Apostolos Antonacopoulos ·
Subhasis Chaudhuri · Rama Chellappa ·
Cheng-Lin Liu · Saumik Bhattacharya ·
Umapada Pal (Eds.)

LNCS 15310

Pattern Recognition

27th International Conference, ICPR 2024
Kolkata, India, December 1–5, 2024
Proceedings, Part X

10 Part X

ICPR 2024 INDIA

IAPR

Springer

MOREMEDIA

Apostolos Antonacopoulos · Subhasis Chaudhuri · Rama Chellappa · Cheng-Lin Liu · Saumik Bhattacharya · Umapada Pal
Editors

Pattern Recognition

27th International Conference, ICPR 2024
Kolkata, India, December 1–5, 2024
Proceedings, Part X

Springer

Editors

Apostolos Antonacopoulos
University of Salford
Salford, UK

Subhasis Chaudhuri
Indian Institute of Technology Bombay
Mumbai, India

Rama Chellappa
Johns Hopkins University
Baltimore, MD, USA

Cheng-Lin Liu
Chinese Academy of Sciences
Beijing, China

Saumik Bhattacharya
IIT Kharagpur
Kharagpur, India

Umapada Pal
Indian Statistical Institute Kolkata
Kolkata, India

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-78191-9 ISBN 978-3-031-78192-6 (eBook)
https://doi.org/10.1007/978-3-031-78192-6

President's Address

On behalf of the Executive Committee of the International Association for Pattern Recognition (IAPR), I am pleased to welcome you to the 27th International Conference on Pattern Recognition (ICPR 2024), the main scientific event of the IAPR.

After a completely digital ICPR in the middle of the COVID pandemic and the first hybrid version in 2022, we can now enjoy a fully back-to-normal ICPR this year. I look forward to hearing inspirational talks and keynotes, catching up with colleagues during the breaks and making new contacts in an informal way. At the same time, the conference landscape has changed. Hybrid meetings have made their entrance and will continue. It is exciting to experience how this will influence the conference. Planning for a major event like ICPR must take place over a period of several years. This means many decisions had to be made under a cloud of uncertainty, adding to the already large effort needed to produce a successful conference. It is with enormous gratitude, then, that we must thank the team of organizers for their hard work, flexibility, and creativity in organizing this ICPR. ICPR always provides a wonderful opportunity for the community to gather together. I can think of no better location than Kolkata to renew the bonds of our international research community.

Each ICPR is a bit different owing to the vision of its organizing committee. For 2024, the conference has six different tracks reflecting major themes in pattern recognition: Artificial Intelligence, Pattern Recognition and Machine Learning; Computer and Robot Vision; Image, Speech, Signal and Video Processing; Biometrics and Human Computer Interaction; Document Analysis and Recognition; and Biomedical Imaging and Bioinformatics. This reflects the richness of our field. ICPR 2024 also features two dozen workshops, seven tutorials, and 15 competitions; there is something for everyone. Many thanks to those who are leading these activities, which together add significant value to attending ICPR, whether in person or virtually. Because it is important for ICPR to be as accessible as possible to colleagues from all around the world, we are pleased that the IAPR, working with the ICPR organizers, is continuing our practice of awarding travel stipends to a number of early-career authors who demonstrate financial need. Last but not least, we are thankful to the Springer LNCS team for their effort to publish these proceedings.

Among the presentations from distinguished keynote speakers, we are looking forward to the three IAPR Prize Lectures at ICPR 2024. This year we honor the achievements of Tin Kam Ho (IBM Research) with the IAPR's most prestigious King-Sun Fu Prize "for pioneering contributions to multi-classifier systems, random decision forests, and data complexity analysis". The King-Sun Fu Prize is given in recognition of an outstanding technical contribution to the field of pattern recognition. It honors the memory of Professor King-Sun Fu who was instrumental in the founding of IAPR, served as its first president, and is widely recognized for his extensive contributions to the field of pattern recognition.

The Maria Petrou Prize is given to a living female scientist/engineer who has made substantial contributions to the field of Pattern Recognition and whose past contributions, current research activity and future potential may be regarded as a model to both aspiring and established researchers. It honours the memory of Professor Maria Petrou as a scientist of the first rank, and particularly her role as a pioneer for women researchers. This year, the Maria Petrou Prize is given to Guoying Zhao (University of Oulu), "for contributions to video analysis for facial micro-behavior recognition and remote bio-signal reading (RPPG) for heart rate analysis and face anti-spoofing".

The J.K. Aggarwal Prize is given to a young scientist who has brought a substantial contribution to a field that is relevant to the IAPR community and whose research work has had a major impact on the field. Professor Aggarwal is widely recognized for his extensive contributions to the field of pattern recognition and for his participation in IAPR's activities. This year, the J.K. Aggarwal Prize goes to Xiaolong Wang (UC San Diego) "for groundbreaking contributions to advancing visual representation learning, utilizing self-supervised and attention-based models to establish fundamental frameworks for creating versatile, general-purpose pattern recognition systems".

During the conference we will also recognize 21 new IAPR Fellows selected from a field of very strong candidates. In addition, a number of Best Scientific Paper and Best Student Paper awards will be presented, along with the Best Industry Related Paper Award and the Piero Zamperoni Best Student Paper Award. Congratulations to the recipients of these very well-deserved awards!

I would like to close by again thanking everyone involved in making ICPR 2024 a tremendous success; your hard work is deeply appreciated. These thanks extend to all who chaired the various aspects of the conference and the associated workshops, my ExCo colleagues, and the IAPR Standing and Technical Committees. Linda O'Gorman, the IAPR Secretariat, deserves special recognition for her experience, historical perspective, and attention to detail when it comes to supporting many of the IAPR's most important activities. Her tasks became so numerous that she recently got support from Carolyn Buckley (layout, newsletter), Ugur Halici (ICPR matters), and Rosemary Stramka (secretariat). The IAPR website got a completely new design. Ed Sobczak has taken care of our web presence for so many years already. A big thank you to all of you!

This is, of course, the 27th ICPR conference. Knowing that ICPR is organized every two years, and that the first conference in the series (1973!) pre-dated the formal founding of the IAPR by a few years, it is also exciting to consider that we are celebrating over 50 years of ICPR and at the same time approaching the official IAPR 50th anniversary in 2028: you'll get all information you need at ICPR 2024. In the meantime, I offer my thanks and my best wishes to all who are involved in supporting the IAPR throughout the world.

September 2024

Arjan Kuijper
President of the IAPR

Preface

It is our great pleasure to welcome you to the proceedings of the 27th International Conference on Pattern Recognition (ICPR 2024), held in Kolkata, India. The city, formerly known as 'Calcutta', is the home of the fabled Indian Statistical Institute (ISI), which has been at the forefront of statistical pattern recognition for almost a century. Concepts like the Mahalanobis distance, Bhattacharyya bound, Cramer–Rao bound, and Fisher–Rao metric were invented by pioneers associated with ISI. The first ICPR (called IJCPR then) was held in 1973, and the second in 1974. Subsequently, ICPR has been held every other year. The International Association for Pattern Recognition (IAPR) was founded in 1978 and became the sponsor of the ICPR series. Over the past 50 years, ICPR has attracted huge numbers of scientists, engineers and students from all over the world and contributed to advancing research, development and applications in pattern recognition technology.

ICPR 2024 was held at the Biswa Bangla Convention Centre, one of the largest such facilities in South Asia, situated just 7 kilometers from Kolkata Airport (CCU). According to ChatGPT "Kolkata is often called the 'Cultural Capital of India'. The city has a deep connection to literature, music, theater, and art. It was home to Nobel laureate Rabindranath Tagore, and the Bengali film industry has produced globally renowned filmmakers like Satyajit Ray. The city boasts remarkable colonial architecture, with landmarks like Victoria Memorial, Howrah Bridge, and the Indian Museum (the oldest and largest museum in India). Kolkata's streets are dotted with old mansions and buildings that tell stories of its colonial past. Walking through the city can feel like stepping back into a different era. Finally, Kolkata is also known for its street food."

ICPR 2024 followed a two-round paper submission format. We received a total of 2135 papers (1501 papers in round-1 submissions, and 634 papers in round-2 submissions). Each paper, on average, received 2.84 reviews, in single-blind mode. For the first-round papers we had a rebuttal option available to authors.

In total, 945 papers (669 from round-1 and 276 from round-2) were accepted for presentation, resulting in an acceptance rate of 44.26%, which is consistent with previous ICPR events. At ICPR 2024 the papers were categorized into six tracks: Artificial Intelligence, Machine Learning for Pattern Analysis; Computer Vision and Robotic Perception; Image, Video, Speech, and Signal Analysis; Biometrics and Human-Machine Interaction; Document and Media Analysis; and Biomedical Image Analysis and Informatics.

The main conference ran over December 2–5, 2024. The main program included the presentation of 188 oral papers (19.89% of the accepted papers), 757 poster papers and 12 competition papers (out of 15 submitted). A total 10 oral sessions were held concurrently in four meeting rooms with a total of 40 oral sessions. In total 24 workshops and 7 tutorials were held on December 1, 2024.

The plenary sessions included three prize lectures and three invited presentations. The prize lectures were delivered by Tin Kam Ho (IBM Research, USA; King Sun

Fu Prize winner), Xiaolong Wang (University of California, San Diego, USA; J.K. Aggarwal Prize winner), and Guoying Zhao (University of Oulu, Finland; Maria Petrou Prize winner). The invited speakers were Timothy Hospedales (University of Edinburgh, UK), Venu Govindaraju (University at Buffalo, USA), and Shuicheng Yan (Skywork AI, Singapore).

Several best paper awards were presented in ICPR: the Piero Zamperoni Award for the best paper authored by a student, the BIRPA Best Industry Related Paper Award, and the Best Paper Awards and Best Student Paper Awards for each of the six tracks of ICPR 2024.

The organization of such a large conference would not be possible without the help of many volunteers. Our special gratitude goes to the Program Chairs (Apostolos Antonacopoulos, Subhasis Chaudhuri, Rama Chellappa and Cheng-Lin Liu), for their leadership in organizing the program. Thanks to our Publication Chairs (Ananda S. Chowdhury and Wataru Ohyama) for handling the overwhelming workload of publishing the conference proceedings. We also thank our Competition Chairs (Richard Zanibbi, Lianwen Jin and Laurence Likforman-Sulem) for arranging 12 important competitions as part of ICPR 2024. We are thankful to our Workshop Chairs (P. Shivakumara, Stephanie Schuckers, Jean-Marc Ogier and Prabir Bhattacharya) and Tutorial Chairs (B.B. Chaudhuri, Michael R. Jenkin and Guoying Zhao) for arranging the workshops and tutorials on emerging topics. ICPR 2024, for the first time, held a Doctoral Consortium. We would like to thank our Doctoral Consortium Chairs (Véronique Eglin, Dan Lopresti and Mayank Vatsa) for organizing it.

Thanks go to the Track Chairs and the meta reviewers who devoted significant time to the review process and preparation of the program. We also sincerely thank the reviewers who provided valuable feedback to the authors.

Finally, we acknowledge the work of other conference committee members, like the Organizing Chairs and Organizing Committee Members, Finance Chairs, Award Chair, Sponsorship Chairs, and Exhibition and Demonstration Chairs, Visa Chair, Publicity Chairs, and Women in ICPR Chairs, whose efforts made this event successful. We also thank our event manager Alpcord Network for their help.

We hope that all the participants found the technical program informative and enjoyed the sights, culture and cuisine of Kolkata.

October 2024

Umapada Pal
Josef Kittler
Anil Jain

Organization

General Chairs

Umapada Pal	Indian Statistical Institute, Kolkata, India
Josef Kittler	University of Surrey, UK
Anil Jain	Michigan State University, USA

Program Chairs

Apostolos Antonacopoulos	University of Salford, UK
Subhasis Chaudhuri	Indian Institute of Technology, Bombay, India
Rama Chellappa	Johns Hopkins University, USA
Cheng-Lin Liu	Institute of Automation, Chinese Academy of Sciences, China

Publication Chairs

Ananda S. Chowdhury	Jadavpur University, India
Wataru Ohyama	Tokyo Denki University, Japan

Competition Chairs

Richard Zanibbi	Rochester Institute of Technology, USA
Lianwen Jin	South China University of Technology, China
Laurence Likforman-Sulem	Télécom Paris, France

Workshop Chairs

P. Shivakumara	University of Salford, UK
Stephanie Schuckers	Clarkson University, USA
Jean-Marc Ogier	Université de la Rochelle, France
Prabir Bhattacharya	Concordia University, Canada

Tutorial Chairs

B. B. Chaudhuri	Indian Statistical Institute, Kolkata, India
Michael R. Jenkin	York University, Canada
Guoying Zhao	University of Oulu, Finland

Doctoral Consortium Chairs

Véronique Eglin	CNRS, France
Daniel P. Lopresti	Lehigh University, USA
Mayank Vatsa	Indian Institute of Technology, Jodhpur, India

Organizing Chairs

Saumik Bhattacharya	Indian Institute of Technology, Kharagpur, India
Palash Ghosal	Sikkim Manipal University, India

Organizing Committee

Santanu Phadikar	West Bengal University of Technology, India
SK Md Obaidullah	Aliah University, India
Sayantari Ghosh	National Institute of Technology Durgapur, India
Himadri Mukherjee	West Bengal State University, India
Nilamadhaba Tripathy	Clarivate Analytics, USA
Chayan Halder	West Bengal State University, India
Shibaprasad Sen	Techno Main Salt Lake, India

Finance Chairs

Kaushik Roy	West Bengal State University, India
Michael Blumenstein	University of Technology Sydney, Australia

Awards Committee Chair

Arpan Pal	Tata Consultancy Services, India

Sponsorship Chairs

P. J. Narayanan	Indian Institute of Technology, Hyderabad, India
Yasushi Yagi	Osaka University, Japan
Venu Govindaraju	University at Buffalo, USA
Alberto Bel Bimbo	Università di Firenze, Italy

Exhibition and Demonstration Chairs

Arjun Jain	FastCode AI, India
Agnimitra Biswas	National Institute of Technology, Silchar, India

International Liaison, Visa Chair

Balasubramanian Raman	Indian Institute of Technology, Roorkee, India

Publicity Chairs

Dipti Prasad Mukherjee	Indian Statistical Institute, Kolkata, India
Bob Fisher	University of Edinburgh, UK
Xiaojun Wu	Jiangnan University, China

Women in ICPR Chairs

Ingela Nystrom	Uppsala University, Sweden
Alexandra B. Albu	University of Victoria, Canada
Jing Dong	Institute of Automation, Chinese Academy of Sciences, China
Sarbani Palit	Indian Statistical Institute, Kolkata, India

Event Manager

Alpcord Network

Track Chairs – Artificial Intelligence, Machine Learning for Pattern Analysis

Larry O'Gorman	Nokia Bell Labs, USA
Dacheng Tao	University of Sydney, Australia
Petia Radeva	University of Barcelona, Spain
Susmita Mitra	Indian Statistical Institute, Kolkata, India
Jiliang Tang	Michigan State University, USA

Track Chairs – Computer and Robot Vision

C. V. Jawahar	International Institute of Information Technology (IIIT), Hyderabad, India
João Paulo Papa	São Paulo State University, Brazil
Maja Pantic	Imperial College London, UK
Gang Hua	Dolby Laboratories, USA
Junwei Han	Northwestern Polytechnical University, China

Track Chairs – Image, Speech, Signal and Video Processing

P. K. Biswas	Indian Institute of Technology, Kharagpur, India
Shang-Hong Lai	National Tsing Hua University, Taiwan
Hugo Jair Escalante	INAOE, CINVESTAV, Mexico
Sergio Escalera	Universitat de Barcelona, Spain
Prem Natarajan	University of Southern California, USA

Track Chairs – Biometrics and Human Computer Interaction

Richa Singh	Indian Institute of Technology, Jodhpur, India
Massimo Tistarelli	University of Sassari, Italy
Vishal Patel	Johns Hopkins University, USA
Wei-Shi Zheng	Sun Yat-sen University, China
Jian Wang	Snap, USA

Track Chairs – Document Analysis and Recognition

Xiang Bai	Huazhong University of Science and Technology, China
David Doermann	University at Buffalo, USA
Josep Llados	Universitat Autònoma de Barcelona, Spain
Mita Nasipuri	Jadavpur University, India

Track Chairs – Biomedical Imaging and Bioinformatics

Jayanta Mukhopadhyay	Indian Institute of Technology, Kharagpur, India
Xiaoyi Jiang	Universität Münster, Germany
Seong-Whan Lee	Korea University, Korea

Metareviewers (Conference Papers and Competition Papers)

Wael Abd-Almageed	University of Southern California, USA
Maya Aghaei	NHL Stenden University, Netherlands
Alireza Alaei	Southern Cross University, Australia
Rajagopalan N. Ambasamudram	Indian Institute of Technology, Madras, India
Suyash P. Awate	Indian Institute of Technology, Bombay, India
Inci M. Baytas	Bogazici University, Turkey
Aparna Bharati	Lehigh University, USA
Brojeshwar Bhowmick	Tata Consultancy Services, India
Jean-Christophe Burie	University of La Rochelle, France
Gustavo Carneiro	University of Surrey, UK
Chee Seng Chan	Universiti Malaya, Malaysia
Sumohana S. Channappayya	Indian Institute of Technology, Hyderabad, India
Dongdong Chen	Microsoft, USA
Shengyong Chen	Tianjin University of Technology, China
Jun Cheng	Institute for Infocomm Research, A*STAR, Singapore
Albert Clapés	University of Barcelona, Spain
Oscar Dalmau	Center for Research in Mathematics, Mexico

Tyler Derr	Vanderbilt University, USA
Abhinav Dhall	Indian Institute of Technology, Ropar, India
Bo Du	Wuhan University, China
Yuxuan Du	University of Sydney, Australia
Ayman S. El-Baz	University of Louisville, USA
Francisco Escolano	University of Alicante, Spain
Siamac Fazli	Nazarbayev University, Kazakhstan
Jianjiang Feng	Tsinghua University, China
Gernot A. Fink	TU Dortmund University, Germany
Alicia Fornes	CVC, Spain
Junbin Gao	University of Sydney, Australia
Yan Gao	Amazon, USA
Yongsheng Gao	Griffith University, Australia
Caren Han	University of Melbourne, Australia
Ran He	Institute of Automation, Chinese Academy of Sciences, China
Tin Kam Ho	IBM, USA
Di Huang	Beihang University, China
Kaizhu Huang	Duke Kunshan University, China
Donato Impedovo	University of Bari, Italy
Julio Jacques	University of Barcelona and Computer Vision Center, Spain
Lianwen Jin	South China University of Technology, China
Wei Jin	Emory University, USA
Danilo Samuel Jodas	São Paulo State University, Brazil
Manjunath V. Joshi	DA-IICT, India
Jayashree Kalpathy-Cramer	Massachusetts General Hospital, USA
Dimosthenis Karatzas	Computer Vision Centre, Spain
Hamid Karimi	Utah State University, USA
Baiying Lei	Shenzhen University, China
Guoqi Li	Chinese Academy of Sciences, and Peng Cheng Lab, China
Laurence Likforman-Sulem	Institut Polytechnique de Paris/Télécom Paris, France
Aishan Liu	Beihang University, China
Bo Liu	Bytedance, USA
Chen Liu	Clarkson University, USA
Cheng-Lin Liu	Institute of Automation, Chinese Academy of Sciences, China
Hongmin Liu	University of Science and Technology Beijing, China
Hui Liu	Michigan State University, USA

Jing Liu	Institute of Automation, Chinese Academy of Sciences, China
Li Liu	University of Oulu, Finland
Qingshan Liu	Nanjing University of Posts and Telecommunications, China
Adrian P. Lopez-Monroy	Centro de Investigacion en Matematicas AC, Mexico
Daniel P. Lopresti	Lehigh University, USA
Shijian Lu	Nanyang Technological University, Singapore
Yong Luo	Wuhan University, China
Andreas K. Maier	FAU Erlangen-Nuremberg, Germany
Davide Maltoni	University of Bologna, Italy
Hong Man	Stevens Institute of Technology, USA
Lingtong Min	Northwestern Polytechnical University, China
Paolo Napoletano	University of Milano-Bicocca, Italy
Kamal Nasrollahi	Milestone Systems, Aalborg University, Denmark
Marcos Ortega	University of A Coruña, Spain
Shivakumara Palaiahnakote	University of Salford, UK
P. Jonathon Phillips	NIST, USA
Filiberto Pla	University Jaume I, Spain
Ajit Rajwade	Indian Institute of Technology, Bombay, India
Shanmuganathan Raman	Indian Institute of Technology, Gandhinagar, India
Imran Razzak	UNSW, Australia
Beatriz Remeseiro	University of Oviedo, Spain
Gustavo Rohde	University of Virginia, USA
Partha Pratim Roy	Indian Institute of Technology, Roorkee, India
Sanjoy K. Saha	Jadavpur University, India
Joan Andreu Sánchez	Universitat Politècnica de València, Spain
Claudio F. Santos	UFSCar, Brazil
Shin'ichi Satoh	National Institute of Informatics, Japan
Stephanie Schuckers	Clarkson University, USA
Srirangaraj Setlur	University at Buffalo, SUNY, USA
Debdoot Sheet	Indian Institute of Technology, Kharagpur, India
Jun Shen	University of Wollongong, Australia
Li Shen	JD Explore Academy, China
Chen Shengyong	Zhejiang University of Technology and Tianjin University of Technology, China
Andy Song	RMIT University, Australia
Akihiro Sugimoto	National Institute of Informatics, Japan
Qianru Sun	Singapore Management University, Singapore
Arijit Sur	Indian Institute of Technology, Guwahati, India
Estefania Talavera	University of Twente, Netherlands

Wei Tang	University of Illinois at Chicago, USA
Joao M. Tavares	Universidade do Porto, Portugal
Jun Wan	NLPR, CASIA, China
Le Wang	Xi'an Jiaotong University, China
Lei Wang	Australian National University, Australia
Xiaoyang Wang	Tencent AI Lab, USA
Xinggang Wang	Huazhong University of Science and Technology, China
Xiao-Jun Wu	Jiangnan University, China
Yiding Yang	Bytedance, China
Xiwen Yao	Northwestern Polytechnical University, China
Xu-Cheng Yin	University of Science and Technology Beijing, China
Baosheng Yu	University of Sydney, Australia
Shiqi Yu	Southern University of Science and Technology, China
Xin Yuan	Westlake University, China
Yibing Zhan	JD Explore Academy, China
Jing Zhang	University of Sydney, Australia
Lefei Zhang	Wuhan University, China
Min-Ling Zhang	Southeast University, China
Wenbin Zhang	Florida International University, USA
Jiahuan Zhou	Peking University, China
Sanping Zhou	Xi'an Jiaotong University, China
Tianyi Zhou	University of Maryland, USA
Lei Zhu	Shandong Normal University, China
Pengfei Zhu	Tianjin University, China
Wangmeng Zuo	Harbin Institute of Technology, China

Reviewers (Competition Papers)

Liangcai Gao
Mingxin Huang
Lei Kang
Wenhui Liao
Yuliang Liu
Yongxin Shi
Da-Han Wang
Yang Xue
Wentao Yang
Jiaxin Zhang
Yiwu Zhong

Reviewers (Conference Papers)

Aakanksha Aakanksha
Aayush Singla
Abdul Muqeet
Abhay Yadav
Abhijeet Vijay Nandedkar
Abhimanyu Sahu
Abhinav Rajvanshi
Abhisek Ray
Abhishek Shrivastava
Abhra Chaudhuri
Aditi Roy
Adriano Simonetto
Adrien Maglo
Ahmed Abdulkadir
Ahmed Boudissa
Ahmed Hamdi
Ahmed Rida Sekkat
Ahmed Sharafeldeen
Aiman Farooq
Aishwarya Venkataramanan
Ajay Kumar
Ajay Kumar Reddy Poreddy
Ajita Rattani
Ajoy Mondal
Akbar K.
Akbar Telikani
Akshay Agarwal
Akshit Jindal
Al Zadid Sultan Bin Habib
Albert Clapés
Alceu Britto
Alejandro Peña
Alessandro Ortis
Alessia Auriemma Citarella
Alexandre Stenger
Alexandros Sopasakis
Alexia Toumpa
Ali Khan
Alik Pramanick
Alireza Alaei
Alper Yilmaz
Aman Verma
Amit Bhardwaj
Amit More
Amit Nandedkar
Amitava Chatterjee
Amos L. Abbott
Amrita Mohan
Anand Mishra
Ananda S. Chowdhury
Anastasia Zakharova
Anastasios L. Kesidis
Andras Horvath
Andre Gustavo Hochuli
André P. Kelm
Andre Wyzykowski
Andrea Bottino
Andrea Lagorio
Andrea Torsello
Andreas Fischer
Andreas K. Maier
Andreu Girbau Xalabarder
Andrew Beng Jin Teoh
Andrew Shin
Andy J. Ma
Aneesh S. Chivukula
Ángela Casado-García
Anh Quoc Nguyen
Anindya Sen
Anirban Saha
Anjali Gautam
Ankan Bhattacharyya
Ankit Jha
Anna Scius-Bertrand
Annalisa Franco
Antoine Doucet
Antonino Staiano
Antonio Fernández
Antonio Parziale
Anu Singha
Anustup Choudhury
Anwesan Pal
Anwesha Sengupta
Archisman Adhikary
Arjan Kuijper
Arnab Kumar Das

Arnav Bhavsar
Arnav Varma
Arpita Dutta
Arshad Jamal
Artur Jordao
Arunkumar Chinnaswamy
Aryan Jadon
Aryaz Baradarani
Ashima Anand
Ashis Dhara
Ashish Phophalia
Ashok K. Bhateja
Ashutosh Vaish
Ashwani Kumar
Asifuzzaman Lasker
Atefeh Khoshkhahtinat
Athira Nambiar
Attilio Fiandrotti
Avandra S. Hemachandra
Avik Hati
Avinash Sharma
B. H. Shekar
B. Uma Shankar
Bala Krishna Thunakala
Balaji Tk
Balázs Pálffy
Banafsheh Adami
Bang-Dang Pham
Baochang Zhang
Baodi Liu
Bashirul Azam Biswas
Beiduo Chen
Benedikt Kottler
Beomseok Oh
Berkay Aydin
Berlin S. Shaheema
Bertrand Kerautret
Bettina Finzel
Bhavana Singh
Bibhas C. Dhara
Bilge Gunsel
Bin Chen
Bin Li
Bin Liu
Bin Yao
Bin-Bin Jia
Binbin Yong
Bindita Chaudhuri
Bindu Madhavi Tummala
Binh M. Le
Bi-Ru Dai
Bo Huang
Bo Jiang
Bob Zhang
Bowen Liu
Bowen Zhang
Boyang Zhang
Boyu Diao
Boyun Li
Brian M. Sadler
Bruce A. Maxwell
Bryan Bo Cao
Buddhika L. Semage
Bushra Jalil
Byeong-Seok Shin
Byung-Gyu Kim
Caihua Liu
Cairong Zhao
Camille Kurtz
Carlos A. Caetano
Carlos D. Martã-Nez-Hinarejos
Ce Wang
Cevahir Cigla
Chakravarthy Bhagvati
Chandrakanth Vipparla
Changchun Zhang
Changde Du
Changkun Ye
Changxu Cheng
Chao Fan
Chao Guo
Chao Qu
Chao Wen
Chayan Halder
Che-Jui Chang
Chen Feng
Chenan Wang
Cheng Yu
Chenghao Qian
Cheng-Lin Liu

Emanuele Vivoli
Emna Ghorbel
Enrique Naredo
Enyu Cai
Eric Patterson
Ernest Valveny
Eva Blanco-Mallo
Eva Breznik
Evangelos Sartinas
Fabio Solari
Fabiola De Marco
Fan Wang
Fangda Li
Fangyuan Lei
Fangzhou Lin
Fangzhou Luo
Fares Bougourzi
Farman Ali
Fatiha Mokdad
Fei Shen
Fei Teng
Fei Zhu
Feiyan Hu
Felipe Gomes Oliveira
Feng Li
Fengbei Liu
Fenghua Zhu
Fillipe D. M. De Souza
Flavio Piccoli
Flavio Prieto
Florian Kleber
Francesc Serratosa
Francesco Bianconi
Francesco Castro
Francesco Ponzio
Francisco Javier Hernández López
Frédéric Rayar
Furkan Osman Kar
Fushuo Huo
Fuxiao Liu
Fu-Zhao Ou
Gabriel Turinici
Gabrielle Flood
Gajjala Viswanatha Reddy
Gaku Nakano
Galal Binamakhashen
Ganesh Krishnasamy
Gang Pan
Gangyan Zeng
Gani Rahmon
Gaurav Harit
Gennaro Vessio
Genoveffa Tortora
George Azzopardi
Gerard Ortega
Gerardo E. Altamirano-Gomez
Gernot A. Fink
Gibran Benitez-Garcia
Gil Ben-Artzi
Gilbert Lim
Giorgia Minello
Giorgio Fumera
Giovanna Castellano
Giovanni Puglisi
Giulia Orrù
Giuliana Ramella
Gökçe Uludoğan
Gopi Ramena
Gorthi Rama Krishna Sai Subrahmanyam
Gourav Datta
Gowri Srinivasa
Gozde Sahin
Gregory Randall
Guanjie Huang
Guanjun Li
Guanwen Zhang
Guanyu Xu
Guanyu Yang
Guanzhou Ke
Guhnoo Yun
Guido Borghi
Guilherme Brandão Martins
Guillaume Caron
Guillaume Tochon
Guocai Du
Guohao Li
Guoqiang Zhong
Guorong Li
Guotao Li
Gurman Gill

Imran Sarker
Inderjot Singh Saggu
Indrani Mukherjee
Indranil Sur
Ines Rieger
Ioannis Pierros
Irina Rabaev
Ivan V. Medri
J. Rafid Siddiqui
Jacek Komorowski
Jacopo Bonato
Jacson Rodrigues Correia-Silva
Jaekoo Lee
Jaime Cardoso
Jakob Gawlikowski
Jakub Nalepa
James L. Wayman
Jan Čech
Jangho Lee
Jani Boutellier
Javier Gurrola-Ramos
Javier Lorenzo-Navarro
Jayasree Saha
Jean Lee
Jean Paul Barddal
Jean-Bernard Hayet
Jean-Philippe G. Tarel
Jean-Yves Ramel
Jenny Benois-Pineau
Jens Bayer
Jerin Geo James
Jesús Miguel García-Gorrostieta
Jia Qu
Jiahong Chen
Jiaji Wang
Jian Hou
Jian Liang
Jian Xu
Jian Zhu
Jianfeng Lu
Jianfeng Ren
Jiangfan Liu
Jianguo Wang
Jiangyan Yi
Jiangyong Duan
Jianhua Yang
Jianhua Zhang
Jianhui Chen
Jianjia Wang
Jianli Xiao
Jianqiang Xiao
Jianwu Wang
Jianxin Zhang
Jianxiong Gao
Jianxiong Zhou
Jianyu Wang
Jianzhong Wang
Jiaru Zhang
Jiashu Liao
Jiaxin Chen
Jiaxin Lu
Jiaxing Ye
Jiaxuan Chen
Jiaxuan Li
Jiayi He
Jiayin Lin
Jie Ou
Jiehua Zhang
Jiejie Zhao
Jignesh S. Bhatt
Jin Gao
Jin Hou
Jin Hu
Jin Shang
Jing Tian
Jing Yu Chen
Jingfeng Yao
Jinglun Feng
Jingtong Yue
Jingwei Guo
Jingwen Xu
Jingyuan Xia
Jingzhe Ma
Jinhong Wang
Jinjia Wang
Jinlai Zhang
Jinlong Fan
Jinming Su
Jinrong He
Jintao Huang

Jinwoo Ahn
Jinwoo Choi
Jinyang Liu
Jinyu Tian
Jionghao Lin
Jiuding Duan
Jiwei Shen
Jiyan Pan
Jiyoun Kim
João Papa
Johan Debayle
John Atanbori
John Wilson
John Zhang
Jónathan Heras
Joohi Chauhan
Jorge Calvo-Zaragoza
Jorge Figueroa
Jorma Laaksonen
José Joaquim De Moura Ramos
Jose Vicent
Joseph Damilola Akinyemi
Josiane Zerubia
Juan Wen
Judit Szücs
Juepeng Zheng
Juha Roning
Jumana H. Alsubhi
Jun Cheng
Jun Ni
Jun Wan
Junghyun Cho
Junjie Liang
Junjie Ye
Junlin Hu
Juntong Ni
Junxin Lu
Junxuan Li
Junyaup Kim
Junyeong Kim
Jürgen Seiler
Jushang Qiu
Juyang Weng
Jyostna Devi Bodapati
Jyoti Singh Kirar
Kai Jiang
Kaiqiang Song
Kalidas Yeturu
Kalle Åström
Kamalakar Vijay Thakare
Kang Gu
Kang Ma
Kanji Tanaka
Karthik Seemakurthy
Kaushik Roy
Kavisha Jayathunge
Kazuki Uehara
Ke Shi
Keigo Kimura
Keiji Yanai
Kelton A. P. Costa
Kenneth Camilleri
Kenny Davila
Ketan Atul Bapat
Ketan Kotwal
Kevin Desai
Keyu Long
Khadiga Mohamed Ali
Khakon Das
Khan Muhammad
Kilho Son
Kim-Ngan Nguyen
Kishan Kc
Kishor P. Upla
Klaas Dijkstra
Komal Bharti
Konstantinos Triaridis
Kostas Ioannidis
Koyel Ghosh
Kripabandhu Ghosh
Krishnendu Ghosh
Kshitij S. Jadhav
Kuan Yan
Kun Ding
Kun Xia
Kun Zeng
Kunal Banerjee
Kunal Biswas
Kunchi Li
Kurban Ubul

Lahiru N. Wijayasingha
Laines Schmalwasser
Lakshman Mahto
Lala Shakti Swarup Ray
Lale Akarun
Lan Yan
Lawrence Amadi
Lee Kang Il
Lei Fan
Lei Shi
Lei Wang
Leonardo Rossi
Lequan Lin
Levente Tamas
Li Bing
Li Li
Li Ma
Li Song
Lia Morra
Liang Xie
Liang Zhao
Lianwen Jin
Libing Zeng
Lidia Sánchez-González
Lidong Zeng
Lijun Li
Likang Wang
Lili Zhao
Lin Chen
Lin Huang
Linfei Wang
Ling Lo
Lingchen Meng
Lingheng Meng
Lingxiao Li
Lingzhong Fan
Liqi Yan
Liqiang Jing
Lisa Gutzeit
Liu Ziyi
Liushuai Shi
Liviu-Daniel Stefan
Liyuan Ma
Liyun Zhu
Lizuo Jin
Longteng Guo
Lorena Álvarez Rodríguez
Lorenzo Putzu
Lu Leng
Lu Pang
Lu Wang
Luan Pham
Luc Brun
Luca Guarnera
Luca Piano
Lucas Alexandre Ramos
Lucas Goncalves
Lucas M. Gago
Luigi Celona
Luis C. S. Afonso
Luis Gerardo De La Fraga
Luis S. Luevano
Luis Teixeira
Lunke Fei
M. Hassaballah
Maddimsetti Srinivas
Mahendran N.
Mahesh Mohan M. R.
Maiko Lie
Mainak Singha
Makoto Hirose
Malay Bhattacharyya
Mamadou Dian Bah
Man Yao
Manali J. Patel
Manav Prabhakar
Manikandan V. M.
Manish Bhatt
Manjunath Shantharamu
Manuel Curado
Manuel Günther
Manuel Marques
Marc A. Kastner
Marc Chaumont
Marc Cheong
Marc Lalonde
Marco Cotogni
Marcos C. Santana
Mario Molinara
Mariofanna Milanova

Markus Bauer
Marlon Becker
Mårten Wadenbäck
Martin G. Ljungqvist
Martin Kampel
Martina Pastorino
Marwan Torki
Masashi Nishiyama
Masayuki Tanaka
Massimo O. Spata
Matteo Ferrara
Matthew D. Dawkins
Matthew Gadd
Matthew S. Watson
Maura Pintor
Max Ehrlich
Maxim Popov
Mayukh Das
Md Baharul Islam
Md Sajid
Meghna Kapoor
Meghna P. Ayyar
Mei Wang
Meiqi Wu
Melissa L. Tijink
Meng Li
Meng Liu
Meng-Luen Wu
Mengnan Liu
Mengxi China Guo
Mengya Han
Michaël Clément
Michal Kawulok
Mickael Coustaty
Miguel Domingo
Milind G. Padalkar
Ming Liu
Ming Ma
Mingchen Feng
Mingde Yao
Minghao Li
Mingjie Sun
Ming-Kuang Daniel Wu
Mingle Xu
Mingyong Li
Mingyuan Jiu
Minh P. Nguyen
Minh Q. Tran
Minheng Ni
Minsu Kim
Minyi Zhao
Mirko Paolo Barbato
Mo Zhou
Modesto Castrillón-Santana
Mohamed Amine Mezghich
Mohamed Dahmane
Mohamed Elsharkawy
Mohamed Yousuf
Mohammad Hashemi
Mohammad Khalooei
Mohammad Khateri
Mohammad Mahdi Dehshibi
Mohammad Sadil Khan
Mohammed Mahmoud
Moises Diaz
Monalisha Mahapatra
Monidipa Das
Mostafa Kamali Tabrizi
Mridul Ghosh
Mrinal Kanti Bhowmik
Muchao Ye
Mugalodi Ramesha Rakesh
Muhammad Rameez Ur Rahman
Muhammad Suhaib Kanroo
Muming Zhao
Munender Varshney
Munsif Ali
Na Lv
Nader Karimi
Nagabhushan Somraj
Nakkwan Choi
Nakul Agarwal
Nan Pu
Nan Zhou
Nancy Mehta
Nand Kumar Yadav
Nandakishor Nandakishor
Nandyala Hemachandra
Nanfeng Jiang
Narayan Hegde

Narayan Ji Mishra
Narayan Vetrekar
Narendra D. Londhe
Nathalie Girard
Nati Ofir
Naval Kishore Mehta
Nazmul Shahadat
Neeti Narayan
Neha Bhargava
Nemanja Djuric
Newlin Shebiah R.
Ngo Ba Hung
Nhat-Tan Bui
Niaz Ahmad
Nick Theisen
Nicolas Passat
Nicolas Ragot
Nicolas Sidere
Nikolaos Mitianoudis
Nikolas Ebert
Nilah Ravi Nair
Nilesh A. Ahuja
Nilkanta Sahu
Nils Murrugarra-Llerena
Nina S. T. Hirata
Ninad Aithal
Ning Xu
Ningzhi Wang
Niraj Kumar
Nirmal S. Punjabi
Nisha Varghese
Norio Tagawa
Obaidullah Md Sk
Oguzhan Ulucan
Olfa Mechi
Oliver Tüselmann
Orazio Pontorno
Oriol Ramos Terrades
Osman Akin
Ouadi Beya
Ozge Mercanoglu Sincan
Pabitra Mitra
Padmanabha Reddy Y. C. A.
Palaash Agrawal
Palaiahnakote Shivakumara
Palash Ghosal
Pallav Dutta
Paolo Rota
Paramanand Chandramouli
Paria Mehrani
Parth Agrawal
Partha Basuchowdhuri
Patrick Horain
Pavan Kumar
Pavan Kumar Anasosalu Vasu
Pedro Castro
Peipei Li
Peipei Yang
Peisong Shen
Peiyu Li
Peng Li
Pengfei He
Pengrui Quan
Pengxin Zeng
Pengyu Yan
Peter Eisert
Petra Gomez-Krämer
Pierrick Bruneau
Ping Cao
Pingping Zhang
Pintu Kumar
Pooja Kumari
Pooja Sahani
Prabhu Prasad Dev
Pradeep Kumar
Pradeep Singh
Pranjal Sahu
Prasun Roy
Prateek Keserwani
Prateek Mittal
Praveen Kumar Chandaliya
Praveen Tirupattur
Pravin Nair
Preeti Gopal
Preety Singh
Prem Shanker Yadav
Prerana Mukherjee
Prerna A. Mishra
Prianka Dey
Priyanka Mudgal

Qc Kha Ng
Qi Li
Qi Ming
Qi Wang
Qi Zuo
Qian Li
Qiang Gan
Qiang He
Qiang Wu
Qiangqiang Zhou
Qianli Zhao
Qiansen Hong
Qiao Wang
Qidong Huang
Qihua Dong
Qin Yuke
Qing Guo
Qingbei Guo
Qingchao Zhang
Qingjie Liu
Qinhong Yang
Qiushi Shi
Qixiang Chen
Quan Gan
Quanlong Guan
Rachit Chhaya
Radu Tudor Ionescu
Rafal Zdunek
Raghavendra Ramachandra
Rahimul I. Mazumdar
Rahul Kumar Ray
Rajib Dutta
Rajib Ghosh
Rakesh Kumar
Rakesh Paul
Rama Chellappa
Rami O. Skaik
Ramon Aranda
Ran Wei
Ranga Raju Vatsavai
Ranganath Krishnan
Rasha Friji
Rashmi S.
Razaib Tariq
Rémi Giraud
René Schuster
Renlong Hang
Renrong Shao
Renu Sharma
Reza Sadeghian
Richard Zanibbi
Rimon Elias
Rishabh Shukla
Rita Delussu
Riya Verma
Robert J. Ravier
Robert Sablatnig
Robin Strand
Rocco Pietrini
Rocio Diaz Martin
Rocio Gonzalez-Diaz
Rohit Venkata Sai Dulam
Romain Giot
Romi Banerjee
Ru Wang
Ruben Machucho
Ruddy Théodose
Ruggero Pintus
Rui Deng
Rui P. Paiva
Rui Zhao
Ruifan Li
Ruigang Fu
Ruikun Li
Ruirui Li
Ruixiang Jiang
Ruowei Jiang
Rushi Lan
Rustam Zhumagambetov
S. Amutha
S. Divakar Bhat
Sagar Goyal
Sahar Siddiqui
Sahbi Bahroun
Sai Karthikeya Vemuri
Saibal Dutta
Saihui Hou
Sajad Ahmad Rather
Saksham Aggarwal
Sakthi U.

Salimeh Sekeh
Samar Bouazizi
Samia Boukir
Samir F. Harb
Samit Biswas
Samrat Mukhopadhyay
Samriddha Sanyal
Sandika Biswas
Sandip Purnapatra
Sanghyun Jo
Sangwoo Cho
Sanjay Kumar
Sankaran Iyer
Sanket Biswas
Santanu Roy
Santosh D. Pandure
Santosh Ku Behera
Santosh Nanabhau Palaskar
Santosh Prakash Chouhan
Sarah S. Alotaibi
Sasanka Katreddi
Sathyanarayanan N. Aakur
Saurabh Yadav
Sayan Rakshit
Scott McCloskey
Sebastian Bunda
Sejuti Rahman
Selim Aksoy
Sen Wang
Seraj A. Mostafa
Shanmuganathan Raman
Shao-Yuan Lo
Shaoyuan Xu
Sharia Arfin Tanim
Shehreen Azad
Sheng Wan
Shengdong Zhang
Shengwei Qin
Shenyuan Gao
Sherry X. Chen
Shibaprasad Sen
Shigeaki Namiki
Shiguang Liu
Shijie Ma
Shikun Li
Shinichiro Omachi
Shirley David
Shishir Shah
Shiv Ram Dubey
Shiva Baghel
Shivanand S. Gornale
Shogo Sato
Shotaro Miwa
Shreya Ghosh
Shreya Goyal
Shuai Su
Shuai Wang
Shuai Zheng
Shuaifeng Zhi
Shuang Qiu
Shuhei Tarashima
Shujing Lyu
Shuliang Wang
Shun Zhang
Shunming Li
Shunxin Wang
Shuping Zhao
Shuquan Ye
Shuwei Huo
Shuyue Lan
Shyi-Chyi Cheng
Si Chen
Siddarth Ravichandran
Sihan Chen
Siladittya Manna
Silambarasan Elkana Ebinazer
Simon Benaïchouche
Simon S. Woo
Simone Caldarella
Simone Milani
Simone Zini
Sina Lotfian
Sitao Luan
Sivaselvan B.
Siwei Li
Siwei Wang
Siwen Luo
Siyu Chen
Sk Aziz Ali
Sk Md Obaidullah

Sneha Shukla
Snehasis Banerjee
Snehasis Mukherjee
Snigdha Sen
Sofia Casarin
Soheila Farokhi
Soma Bandyopadhyay
Son Minh Nguyen
Son Xuan Ha
Sonal Kumar
Sonam Gupta
Sonam Nahar
Song Ouyang
Sotiris Kotsiantis
Souhaila Djaffal
Soumen Biswas
Soumen Sinha
Soumitri Chattopadhyay
Souvik Sengupta
Spiros Kostopoulos
Sreeraj Ramachandran
Sreya Banerjee
Srikanta Pal
Srinivas Arukonda
Stephane A. Guinard
Su O. Ruan
Subhadip Basu
Subhajit Paul
Subhankar Ghosh
Subhankar Mishra
Subhankar Roy
Subhash Chandra Pal
Subhayu Ghosh
Sudip Das
Sudipta Banerjee
Suhas Pillai
Sujit Das
Sukalpa Chanda
Sukhendu Das
Suklav Ghosh
Suman K. Ghosh
Suman Samui
Sumit Mishra
Sungho Suh
Sunny Gupta
Suraj Kumar Pandey
Surendrabikram Thapa
Suresh Sundaram
Sushil Bhattacharjee
Susmita Ghosh
Swakkhar Shatabda
Syed Ms Islam
Syed Tousiful Haque
Taegyeong Lee
Taihui Li
Takashi Shibata
Takeshi Oishi
Talha Ahmad Siddiqui
Tanguy Gernot
Tangwen Qian
Tanima Bhowmik
Tanpia Tasnim
Tao Dai
Tao Hu
Tao Sun
Taoran Yi
Tapan Shah
Taveena Lotey
Teng Huang
Tengqi Ye
Teresa Alarcon
Tetsuji Ogawa
Thanh Phuong Nguyen
Thanh Tuan Nguyen
Thattapon Surasak
Thibault Napolãon
Thierry Bouwmans
Thinh Truong Huynh Nguyen
Thomas De Min
Thomas E. K. Zielke
Thomas Swearingen
Tianatahina Jimmy Francky Randrianasoa
Tianheng Cheng
Tianjiao He
Tianyi Wei
Tianyuan Zhang
Tianyue Zheng
Tiecheng Song
Tilottama Goswami
Tim Büchner

Tim H. Langer
Tim Raven
Tingkai Liu
Tingting Yao
Tobias Meisen
Toby P. Breckon
Tong Chen
Tonghua Su
Tran Tuan Anh
Tri-Cong Pham
Trishna Saikia
Trung Quang Truong
Tuan T. Nguyen
Tuan Vo Van
Tushar Shinde
Ujjwal Karn
Ukrit Watchareeruetai
Uma Mudenagudi
Umarani Jayaraman
V. S. Malemath
Vallidevi Krishnamurthy
Ved Prakash
Venkata Krishna Kishore Kolli
Venkata R. Vavilthota
Venkatesh Thirugnana Sambandham
Verónica Maria Vasconcelos
Véronique Ve Eglin
Víctor E. Alonso-Pérez
Vinay Palakkode
Vinayak S. Nageli
Vincent J. Whannou De Dravo
Vincenzo Conti
Vincenzo Gattulli
Vineet Padmanabhan
Vishakha Pareek
Viswanath Gopalakrishnan
Vivek Singh Baghel
Vivekraj K.
Vladimir V. Arlazarov
Vu-Hoang Tran
W. Sylvia Lilly Jebarani
Wachirawit Ponghiran
Wafa Khlif
Wang An-Zhi
Wanli Xue
Wataru Ohyama
Wee Kheng Leow
Wei Chen
Wei Cheng
Wei Hua
Wei Lu
Wei Pan
Wei Tian
Wei Wang
Wei Wei
Wei Zhou
Weidi Liu
Weidong Yang
Weijun Tan
Weimin Lyu
Weinan Guan
Weining Wang
Weiqiang Wang
Weiwei Guo
Weixia Zhang
Wei-Xuan Bao
Weizhong Jiang
Wen Xie
Wenbin Qian
Wenbin Tian
Wenbin Wang
Wenbo Zheng
Wenhan Luo
Wenhao Wang
Wen-Hung Liao
Wenjie Li
Wenkui Yang
Wenwen Si
Wenwen Yu
Wenwen Zhang
Wenwu Yang
Wenxi Li
Wenxi Yue
Wenxue Cui
Wenzhuo Liu
Widhiyo Sudiyono
Willem Dijkstra
Wolfgang Fuhl
Xi Zhang
Xia Yuan

Xianda Zhang
Xiang Zhang
Xiangdong Su
Xiang-Ru Yu
Xiangtai Li
Xiangyu Xu
Xiao Guo
Xiao Hu
Xiao Wu
Xiao Yang
Xiaofeng Zhang
Xiaogang Du
Xiaoguang Zhao
Xiaoheng Jiang
Xiaohong Zhang
Xiaohua Huang
Xiaohua Li
Xiao-Hui Li
Xiaolong Sun
Xiaosong Li
Xiaotian Li
Xiaoting Wu
Xiaotong Luo
Xiaoyan Li
Xiaoyang Kang
Xiaoyi Dong
Xin Guo
Xin Lin
Xin Ma
Xinchi Zhou
Xingguang Zhang
Xingjian Leng
Xingpeng Zhang
Xingzheng Lyu
Xinjian Huang
Xinqi Fan
Xinqi Liu
Xinqiao Zhang
Xinrui Cui
Xizhan Gao
Xu Cao
Xu Ouyang
Xu Zhao
Xuan Shen
Xuan Zhou
Xuchen Li
Xuejing Lei
Xuelu Feng
Xueting Liu
Xuewei Li
Xueyi X. Wang
Xugong Qin
Xu-Qian Fan
Xuxu Liu
Xu-Yao Zhang
Yan Huang
Yan Li
Yan Wang
Yan Xia
Yan Zhuang
Yanan Li
Yanan Zhang
Yang Hou
Yang Jiao
Yang Liping
Yang Liu
Yang Qian
Yang Yang
Yang Zhao
Yangbin Chen
Yangfan Zhou
Yanhui Guo
Yanjia Huang
Yanjun Zhu
Yanming Zhang
Yanqing Shen
Yaoming Cai
Yaoxin Zhuo
Yaoyan Zheng
Yaping Zhang
Yaqian Liang
Yarong Feng
Yasmina Benmabrouk
Yasufumi Sakai
Yasutomo Kawanishi
Yazeed Alzahrani
Ye Du
Ye Duan
Yechao Zhang
Yeong-Jun Cho

Yi Huo
Yi Shi
Yi Yu
Yi Zhang
Yibo Liu
Yibo Wang
Yi-Chieh Wu
Yifan Chen
Yifei Huang
Yihao Ding
Yijie Tang
Yikun Bai
Yimin Wen
Yinan Yang
Yin-Dong Zheng
Yinfeng Yu
Ying Dai
Yingbo Li
Yiqiao Li
Yiqing Huang
Yisheng Lv
Yisong Xiao
Yite Wang
Yizhe Li
Yong Wang
Yonghao Dong
Yong-Hyuk Moon
Yongjie Li
Yongqian Li
Yongqiang Mao
Yongxu Liu
Yongyu Wang
Yongzhi Li
Youngha Hwang
Yousri Kessentini
Yu Wang
Yu Zhou
Yuan Tian
Yuan Zhang
Yuanbo Wen
Yuanxin Wang
Yubin Hu
Yubo Huang
Yuchen Ren
Yucheng Xing
Yuchong Yao
Yuecong Min
Yuewei Yang
Yufei Zhang
Yufeng Yin
Yugen Yi
Yuhang Ming
Yujia Zhang
Yujun Ma
Yukiko Kenmochi
Yun Hoyeoung
Yun Liu
Yunhe Feng
Yunxiao Shi
Yuru Wang
Yushun Tang
Yusuf Osmanlioglu
Yusuke Fujita
Yuta Nakashima
Yuwei Yang
Yuwu Lu
Yuxi Liu
Yuya Obinata
Yuyao Yan
Yuzhi Guo
Zaipeng Xie
Zander W. Blasingame
Zedong Wang
Zeliang Zhang
Zexin Ji
Zhanxiang Feng
Zhaofei Yu
Zhe Chen
Zhe Cui
Zhe Liu
Zhe Wang
Zhekun Luo
Zhen Yang
Zhenbo Li
Zhenchun Lei
Zhenfei Zhang
Zheng Liu
Zheng Wang
Zhengming Yu
Zhengyin Du

Zhengyun Cheng
Zhenshen Qu
Zhenwei Shi
Zhenzhong Kuang
Zhi Cai
Zhi Chen
Zhibo Chu
Zhicun Yin
Zhida Huang
Zhida Zhang
Zhifan Gao
Zhihang Ren
Zhihang Yuan
Zhihao Wang
Zhihua Xie
Zhihui Wang
Zhikang Zhang
Zhiming Zou
Zhiqi Shao
Zhiwei Dong
Zhiwei Qi
Zhixiang Wang
Zhixuan Li
Zhiyu Jiang
Zhiyuan Yan
Zhiyuan Yu
Zhiyuan Zhang
Zhong Chen
Zhongwei Teng
Zhongzhan Huang
Zhongzhi Yu
Zhuan Han
Zhuangzhuang Chen
Zhuo Liu
Zhuo Su
Zhuojun Zou
Zhuoyue Wang
Ziang Song
Zicheng Zhang
Zied Mnasri
Zifan Chen
Žiga Babnik
Zijing Chen
Zikai Zhang
Ziling Huang
Zilong Du
Ziqi Cai
Ziqi Zhou
Zi-Rui Wang
Zirui Zhou
Ziwen He
Ziyao Zeng
Ziyi Zhang
Ziyue Xiang
Zonglei Jing
Zongyi Xu

Contents – Part X

Explaining Model Parameters Using the Product Space

Ethan Payne[1], David Patrick[1,2], and Amanda S. Fernandez[1](✉)

[1] The University of Texas at San Antonio, San Antonio, TX, USA
amanda.fernandez@utsa.edu
[2] Texas State University, San Marcos, TX, USA

Abstract. With the increasing interest in explainable attribution for deep neural networks, it is important to consider not only the importance of individual inputs, but also the model parameters themselves. Existing methods, such as Neuron Integrated Gradients [18] and Conductance [6], attempt model attribution by applying attribution methods, such as Integrated Gradients, to the inputs of each model parameter. While these methods seem to map attributions to individual parameters, these are actually aggregated feature attributions which completely ignore the parameter space and also suffer from the same underlying limitations of Integrated Gradients. In this work, we compute parameter attributions by leveraging the recent family of measures proposed by Generalized Integrated Attributions, by instead computing integrals over the product space of inputs and parameters. This usage of the product space allows us to now explain individual neurons from varying perspectives and interpret them with the same intuition as inputs. To the best of our knowledge, ours is the first method which actually utilizes the gradient landscape of the parameter space to explain each individual weight and bias. We confirm the utility of our parameter attributions by computing exploratory statistics for a wide variety of image classification datasets and by performing pruning analyses on a standard architecture, which demonstrate that our attribution measures are able to identify both important and unimportant neurons in a convolutional neural network.

Keywords: Attribution · Saliency · Influence · Integrated Gradients · Expected Gradients · Explainability · Causal Inference · Pruning · Unlearning

Supported by the National Science Foundation under Grant No. 2134237. Code: https://github.com/UTSA-VAIL/Explaining-Model-Parameters-Using-the-Product-Space.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78192-6_1.

1 Introduction

As deep learning architectures grow in size and complexity, the push for explainability of model predictions and performance continues. While existing attribution methods are able to generate importance values for model inputs and extracted features, it is also critical to consider the model parameters themselves in the context of the ambient parameter space.

We first briefly summarize the several types of attributions for clarity of terminology in the following subsections. Figure 1 provides an illustration of these attribution types.

Input attributions assign importance values, or some other value of interest, to each dimension of the input. For the particular case of image recognition this means assigning importance values to each pixel of the input, or ideally to each pixel's color channels individually.

Intermediate Feature Attributions assign importance values to the feature maps generated by layers and modules within a neural network, in a method similar to input attribution. Since these feature maps are simply transformations of the input data, we can conveniently treat them using the same attribution methodology as we used for inputs.

Parameter Attributions facilitate computing importance values for individual model parameters, i.e. the weights and biases in a convolutional neural network. Since the model itself is an entirely different class of object than either the inputs or extracted features, we must develop a new methodology for extending the theory of attributions to the parameter space.

1.1 Our Contribution

In this work, we achieve a more complete and faithful method of parameter attribution, leveraging the reformulated attribution framework of Generalized Integrated Attributions [21]. Similar to Neuron Integrated Gradients [18] and Conductance [6], we assign an integrated measure to a parameter within a model. However, unlike these previous methods, we do not aggregate path-integrated feature attributions, but rather use the generalized volume-integral formulation proposed in [21], and account for the parameter space by integrating over the product space of inputs and parameter values. By integrating over a set in the parameter space, we are able to interpret the resulting parameter attributions using the same theory as for input and feature attributions. Additionally, this formulation allows us to assign unique attribution values to each weight and bias in a convolutional neural network, which was not possible using previous methods.

To ensure that the computed measures reflect the dataset of interest rather than some arbitrary or counterfactual baseline value, we follow the approaches of Expected Gradients [7] and Generalized Integrated Attributions [21] and take the expectation in the input space over a set from the training dataset. Using this new formulation of *Parameter Explanations using the Product Space (PEPS)*, we are able to extend each of the measures proposed in [21] to model parameters in addition to inputs. Our experiments confirm that our measures are able to successfully identify important and unimportant neurons, and we summarize our findings regarding the distribution of these measures for several datasets, model training statuses, and hyperparameter combinations using exploratory plots. Furthermore, our measures are able to un-learn a specific class from the trained model without destroying the model's performance on the remaining classes, and we identify several trends within the distributions of our attributions which might be used for future training diagnostics and improved robustness.

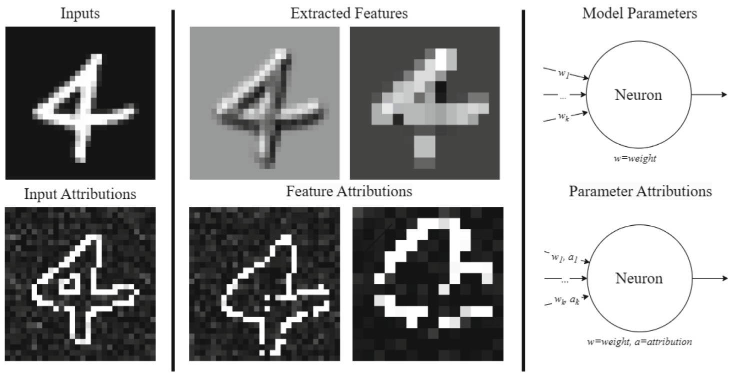


Fig. 1: Mock-up visualization of attributions corresponding to different types of explainable objects. Even once the extracted features cease to resemble the original input, we can still compute attributions using the same methodology as used for input attributions. However, model parameters require a different approach in order to reflect the behavior of the model's own gradient landscape.

2 Related Work

2.1 Input Attributions

Many methods for attributing model predictions to specific features of the input utilize information contained in the model gradients [3,8,19,20,22,24]. Sundararajan et al. devised an attribution method called Integrated Gradients [22] which computes the integral of feature gradients over a linear path from an input to a reference. The value of this integral can then be tracked with respect to each pixel of the input in order to obtain a pixel-wise attribution map. In a similar approach, DeepLift [17] propagates contribution scores according to differences from a reference, which, akin to integrated gradients, requires some justification for correct or appropriate reference values. By incorporating the theory associated with classical Shapley values, Lundberg et al. [14] contextualize several attribution methods such as DeepLift as additive explanations, and uses the theory of SHAP values to unify these different approaches to attribution under a single framework. Ancona et al. [2] show that many gradient-based attribution methods are closely related and can be described with a unified formulation, and propose the metric Sensitivity-n to evaluate these methods.

In contrast to gradient-based methods, other approaches to quantifying feature importance such as the search-based Parallel Local Search (PLS) [10] are able to outperform other state-of-the-art attribution methods. Still, while these search-based methods can be highly useful, they lack much of the a-priori and intuitive explainability offered by gradient-based methods.

A recent reformulation known as Expected Gradients [7] was proposed by Erion et al. to directly improve upon the original method of Integrated Gradients, by computing attributions as an expected value of gradients over the input dataset, thus alleviating the issue of counterfactual baselines. This prompted yet another recent reformulation in the form of Generalized Integrated Gradients [21].

2.2 Intermediate Feature Attributions

It can often be beneficial to consider the intermediate features extracted by machine learning models when collecting attribution information, either for the purpose of improving input attributions as in [3], or in order to assign attributions to model parameters. Using an integration-based method similar to Integrated Gradients, Leino et al. [13] compute an influence-directed attribution measure which can be applied to internal neurons within a model. In two very similar approaches, Shrikumar et al. [18] and Dhamdhere et al. [6] propose Neuron Integrated Gradients and Conductance respectively by applying the principles of Integrated Gradients to the inputs of a given parameter. These feature attributions are then pooled and assigned to the parameter. Although these methods do assign attribution values to individual model parameters, these values are actually aggregated feature attributions rather than representations of the model's own gradient landscape. Additionally, since both of these methods integrate feature gradients over a path in the input space, these aggregated feature attributions are relevant only to this path, entirely neglecting the parameter space itself.

2.3 Parameter Attributions

While methods like [24] intentionally avoid using gradient information when determining neuron importance, our experiments demonstrate that there is a wealth of useful information available within model gradients. Other work on neuron-level analysis has been conducted in the Natural Language Processing (NLP) space [16], but it remains to extend these methods to other domains such as vision. Our method is also similar to the Shapley-value approach of [9], which uses permutations of model parameters in a multi-armed bandit algorithm to determine neuron importance values, as well as the neuron ablation approach of [1] which uses majority voting. While these two methods can be effective for specific use cases, such algorithmic approaches lack the flexibility, intuition, and generalizability of our method for accomodating diverse subjective user needs.

3 Approach

While previous methods consider only integrals within the input space with respect to a single point in parameter space (the model weight state), we instead consider the product space of inputs and parameter values (Fig. 2). By integrating over a set in the product space, we can explain a parameter not only with respect to a given set of inputs, but also with respect to other possible parameter values. We verify the effectiveness of this method in Sect. 4.

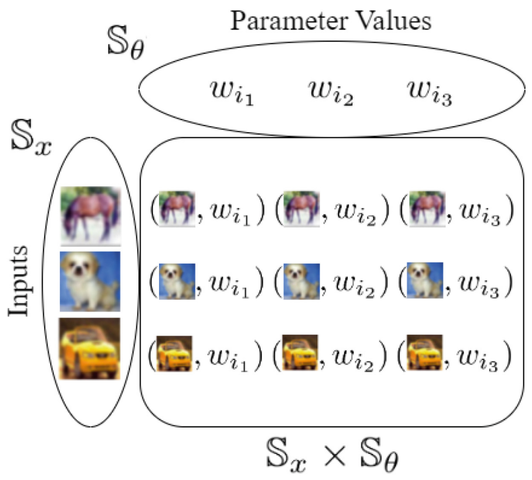

Parameter Values
$\mathbb{S}_\theta$
w_{i_1} w_{i_2} w_{i_3}
$\mathbb{S}_x$
Inputs
$(\,, w_{i_1})$ $(\,, w_{i_2})$ $(\,, w_{i_3})$
$\mathbb{S}_x \times \mathbb{S}_\theta$

Fig. 2: For each neuron, we integrate over the product space of inputs and parameter values in order to collect importance information about that parameter's relation to the set of inputs. Pictured above, we visualize the product space corresponding to three samples for weight w_i and three input samples from the dataset.

3.1 Parameter Explanations Using the Product Space (PEPS)

We first recall the formulation of Generalized Integrated Gradients in Eq. 1 from [21] for an input x and a model F, over a set $\mathbb{S}_x$ in the input space, as well as the even more general Eq. 2 for an attribution function $\mathcal{A}$ and a distribution $p_{\mathbb{S}_x}$:

$$\begin{aligned}\text{GeneralizedIntegratedGrads}(\mathbb{S}_x) &::= \frac{1}{|\mathbb{S}_x|}\int_{\mathbb{S}_x} \nabla F(x)dx \\ &= \mathbb{E}_{\mathbb{S}_x}[\nabla F]\end{aligned} \tag{1}$$

$$\begin{aligned}&\text{GeneralizedIntegratedAttribution}(\mathcal{A}, F, \mathbb{S}_x, p_{\mathbb{S}_x})\\ &\qquad ::= \int_{\mathbb{S}_x} \mathcal{A}(F, x) p_{\mathbb{S}_x}(x) dx\end{aligned} \tag{2}$$

We apply this formulation to model attributions rather than input attributions by treating a parameter or parameter group (such as a convolutional filter) θ as we treated the input x for input attribution. This simultaneous effect of inputs and parameters yields a tuple representing a single point (x, θ) in the input $\times$ parameter product space. We can now compute the gradients of the parameter θ with respect to the input x, so we then integrate over a set of interest $\mathbb{S}_\theta$ and distribution $p_{\mathbb{S}_\theta}$ in the parameter space as well as over a set of interest $\mathbb{S}_x$ and distribution $p_{\mathbb{S}_x}$ in the input space to obtain Eqs. 3 and 4:

$$\begin{aligned}&\text{GeneralizedModelIntegratedGradients}(\mathbb{S}_x, \mathbb{S}_\theta)\\ &\qquad ::= \frac{1}{|\mathbb{S}_x|} \int_{\mathbb{S}_x} \frac{1}{|\mathbb{S}_\theta|} \int_{\mathbb{S}_\theta} \nabla F_\theta(x)\, d\theta dx\\ &\qquad = \frac{1}{|\mathbb{S}_x||\mathbb{S}_\theta|} \int_{\mathbb{S}_x} \int_{\mathbb{S}_\theta} \nabla F_\theta(x)\, d\theta dx\\ &\qquad = \mathbb{E}_{\mathbb{S}_x}\left[\mathbb{E}_{\mathbb{S}_\theta}\left[\nabla F_\theta(x)\right]\right]\\ &\qquad = \mathbb{E}_{\mathbb{S}_x \times \mathbb{S}_\theta}\left[\nabla F_\theta(x)\right]\end{aligned} \tag{3}$$

$$\begin{aligned}&\text{GeneralizedIntegratedModelAttribution}(\mathcal{A}, \mathbb{S}_x, p_{\mathbb{S}_x}, \mathbb{S}_\theta, p_{\mathbb{S}_\theta})\\ &\qquad ::= \int_{\mathbb{S}_x} \left[\int_{\mathbb{S}_\theta} \mathcal{A}(\theta, x) p_{\mathbb{S}_\theta}(\theta) d\theta\right] p_{\mathbb{S}_x}(x) dx\\ &\qquad = \int_{\phi \in \mathbb{S}_x \times \mathbb{S}_\theta} \mathcal{A}(\phi) p_{\mathbb{S}_x \times \mathbb{S}_\theta}(\phi) d\phi\end{aligned} \tag{4}$$

Using this method, we can compute model attributions corresponding to each of the input attribution statistics proposed in [21]: Expected/Integrated Gradients, Gradient Variance, Stability, and Consistency. We compute these attributions for each parameter (weights, biases, etc.), and just as individual pixel attributions can be aggregated to obtain an attribution for an entire image, we can also aggregate parameter attributions as desired to obtain coarser attributions for parameter groups, modules, layers, or the entire model (Eqs. 5 and 6).

$$\text{Attribution(Parameter Group)} = \underset{\theta \in \text{Parameter Group}}{\mathbb{E}}\left[\text{Attributions}(\theta)\right] \tag{5}$$

$$\text{Attribution(Model)} = \underset{\text{Parameter Group} \in \text{Model}}{\mathbb{E}}\left[\text{Attribution(Parameter Group)}\right] \tag{6}$$

Note that this aggregation step is sensitive, as positive and negative measure values can potentially cancel out and obfuscate the true effect at the filter and model-level. For more accurate and thorough attributions, alternative aggregation methods can be applied to combine hierarchically nested attributions, or individual filters and even the entire model might be treated as a single entity for computation of each integrated measure, though this approach may require additional overhead or special considerations with respect to sampling the latent parameter space.

4 Evaluation

We perform two types of quantitative analysis to demonstrate the utility of our method. We first perform pruning experiments to verify that our novel attribution measures are able to identify important and unimportant neurons in a trained model. We then collect additional distributional information for a wider variety of datasets to confirm that our attributions can distinguish between a trained model and an untrained model. For each integrated attribution measure, we select our set $\mathbb{S}_\theta$ to be the ball centered at a parameter of locality radius hyperparameter ε proposed in [21]. We choose our ε values from a large range in our experiments in an attempt to sample both local and nonlocal gradient behavior. We also fix $\mathbb{S}_x$ to be the training set, and follow the same Monte Carlo integration method of [21] in which we sample points from $\mathbb{S}_x$ and $\mathbb{S}_\theta$.

4.1 Pruning

We demonstrate the effectiveness of our method in determining neuron importance by performing a series of pruning experiments (Figs. 3, 4, and 6) as was the approach of [1,9], in which we set a proportion of individual model parameters to zero. If model performance degrades faster or slower when pruning according to ranked attribution values compared to pruning randomly, then we will have successfully identified the important or unimportant neurons respectively. In each experiment, we prune the same proportion of weight and bias parameters from each layer, except for the final output layer which we leave intact.

Pruning Experiments. We perform experiments for the CIFAR-10 dataset [11], shown in Fig. 3, using a ResNet-18 architecture trained for 20 epochs using a standard categorical cross-entropy with learning rate of 0.01 with no momentum or weight decay, and a batch size of 32. We also perform similar experiments for the ImageNet dataset [5], shown in Fig. 4, using a pretrained ResNet-34 architecture. For all pruning experiments, we show the mean F1-Score on the respective test set over 10 replicates and the associated 95% confidence interval. We investigate the effect of our locality radius ε for 32 input sample points and 32 parameter sample points. We also include an investigation of the effect of the number of sample points in the input space and parameter space for in our supplemental information, and we briefly show the benefit of larger sample sizes in Fig. 5. We can notice from Figs. 3 and 4 that our Integrated Gradients and Gradient Variance measures are able to successfully identify important neurons

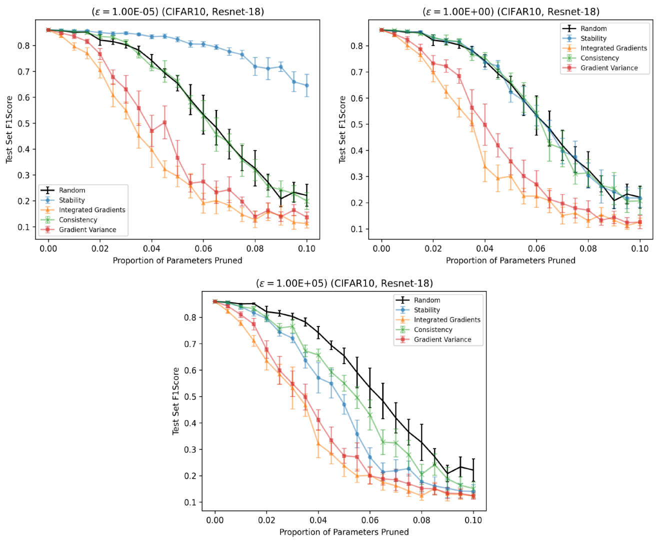


Fig. 3: Pruning of a ResNet-18 model trained on CIFAR-10. We can observe that different choices of locality radius ε reveal different types of attribution information for certain measures. For each of the four measures, we observe statistically significant differences from random pruning, verifying that we have collected information relevant to the model's performance.

for all three of the tested locality radii, but that our Stability measure identifies important neurons for a large radius and unimportant neurons for a small radius. We note the our Consistency measure currently only appears useful for large radii, so this may indicate that we need to explore a wider range of radii and sample points in future studies. We also assume that first multiplying our attributions by -1 will result in the opposite behavior as observed in Figs. 3 and 4, but we must confirm this in future studies.

Pruning to Target a Single Class. We also demonstrate the ability of our method to select for neurons important to a specific class. By fixing $\mathbb{S}_x$ as a specific subset of the input space (i.e. a specific class), we can determine a neuron's importance with respect to that class. We otherwise perform these experiments using the same methodology as Figs. 3 and 4. Shown in Fig. 6 are the two experiments which demonstrate the most selectivity for a single class while preserving performance on the remaining classes, the remaining experiments are available in supplemental information. Our results show resounding success in destroying the model's test set F1-Score for the targeted class, which may prove to be useful for applications related to unlearning [4].

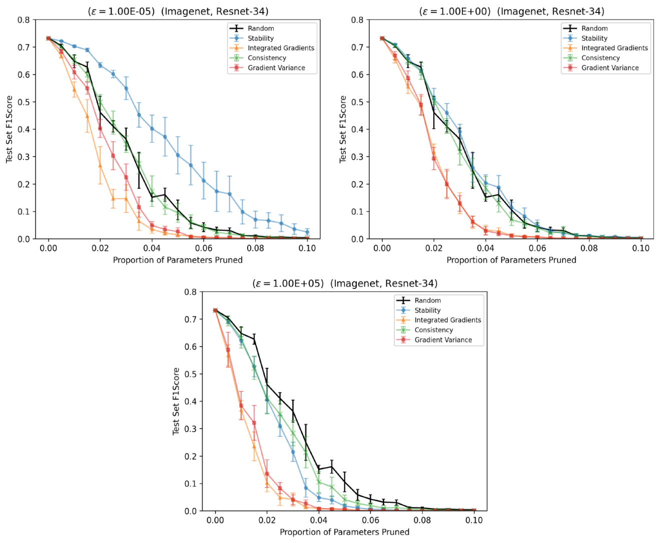


Fig. 4: Pruning of a ResNet-34 model trained on Imagenet. Note the same overall trends for each measure as observed for the CIFAR-10 dataset in Fig. 3, indicating that the utility of our measures successfully generalizes to larger datasets.

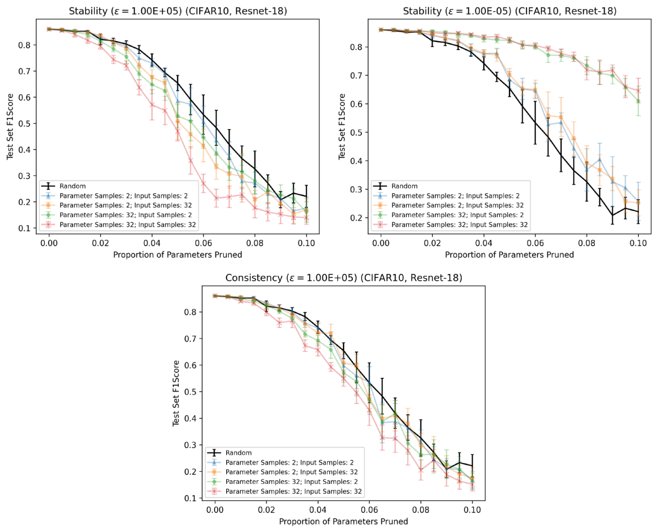


Fig. 5: Pruning of a ResNet-18 model trained on CIFAR-10 for varying input and parameter sample sizes. In general we observe that increased sample size results in more accurate attributions reflecting a neurons relative importance or unimportance.

4.2 Distributional Analysis

We investigate the distribution of our attribution measures for trained and randomly initialized models using several small-scale image classification datasets: CIFAR-10 [11], MNIST [12], FashionMNIST [23], and SVHN [15], using a ResNet-18 architecture trained using the same methodology as for the pruning experiments, except for that on the smaller datasets (MNIST, FashionMNIST, and SVHN) we trained only for 10 epochs. We also include data for partial training on the CIFAR-10 dataset (10 out of 20 epochs) to further illustrate how the distributions converge as the model trains. For these distribution analyses we use 10 sample points from the input space and 20 sample points from the parameter space, and we investigate the distributions of measures corresponding to $\varepsilon = 1.0$ and $\varepsilon = 10^4$.

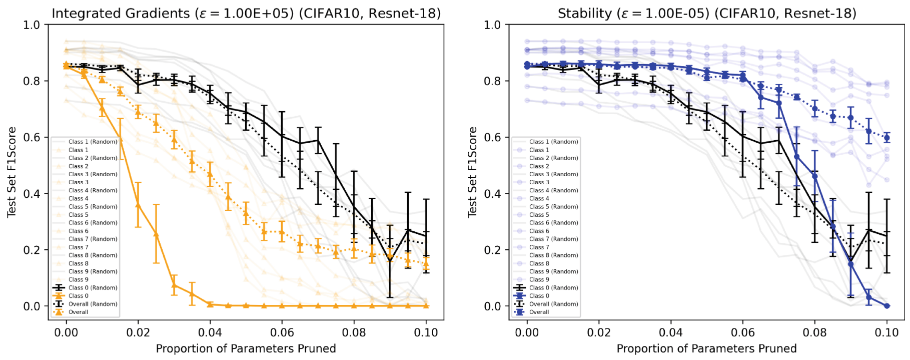


Fig. 6: Single-class targeted pruning of a ResNet-18 model trained on CIFAR-10. Note that we are able to reduce the F1-Score for the targeted class to zero while maintaining $\sim .60$ F1-Score on the overall dataset. This indicates that our measures could be used to very effectively unlearn specific classes as needed for privacy or security applications.

We demonstrate below that the distributions of each of our four integrated model attribution measures are highly dependent on model training status. See Figs. 7, 8, 9, and 10 for histograms and discussion of Integrated Gradients, Gradient Variance, Stability, and Consistency. We can observe distributions resembling several parametric families in the generated histograms, namely the Student's t-distribution for Integrated Gradients, the Gamma distribution for Gradient Variance, and possibly Beta distributions for Stability and Consistency. While we do not in this study fit any parametric distributions to the data, these distinct distributions offer a quantitative and parametric source of model explanation and evaluation.

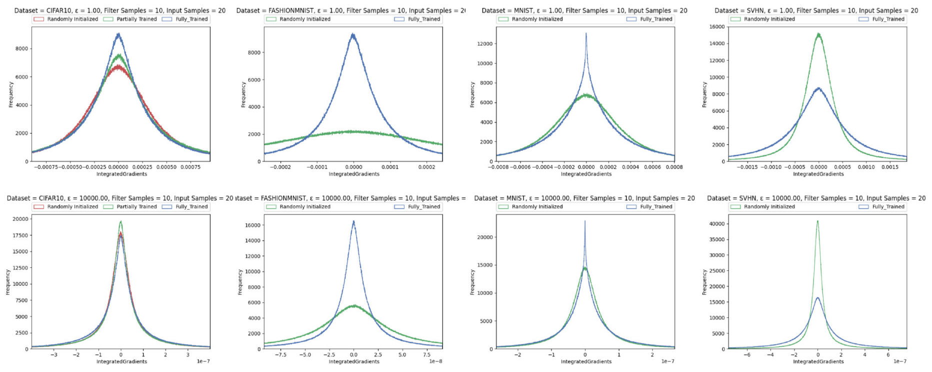


Fig. 7: Parameter-level histograms for the Integrated Gradients measure for models trained on CIFAR-10, FashionMNIST, MNIST, and SVHN datasets. Values outside the .9 quantile are excluded from the histogram range. The attributions appear to be roughly distributed according to a Student's t-distribution for both choices of locality radius ε, which is consistent with the fact that Integrated Gradients is computed as a sample mean of gradients in the Monte Carlo integral assuming gradients in the underlying latent space are normally distributed.

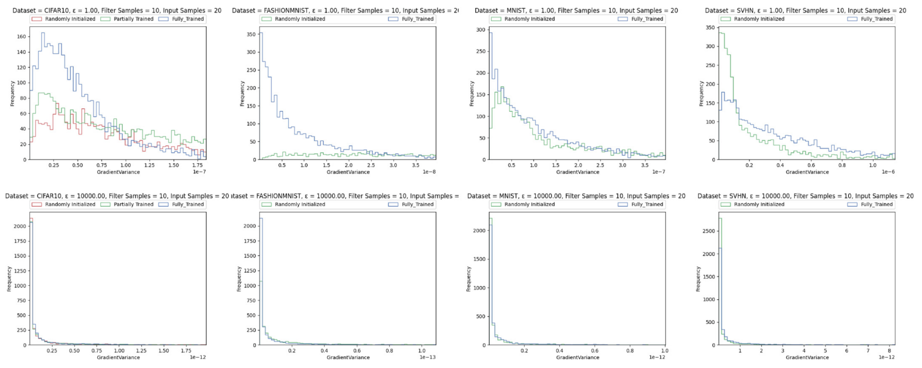


Fig. 8: Filter-level histograms for the Gradient Variance measure for models trained on CIFAR-10, FashionMNIST, MNIST, and SVHN datasets. For these histograms we present filter-level attributions rather than parameter-level attributions due to the large volume of extreme values at the parameter-level. Values outside the .9 quantile are excluded from the histogram range, but even though we are plotting the filter-level distribution, due to the large number of extreme values this range restriction still results in distributions which are difficult to qualitatively evaluate for locality radius $\varepsilon = 10^4$. The attributions appear to be distributed according to a gamma distribution for both choices of locality radius ε, which is consistent with the fact that filter attributions are sums of per-parameter Gradient Variances, which each follow a chi-square distribution assuming that gradients in the underlying latent space are normally distributed.

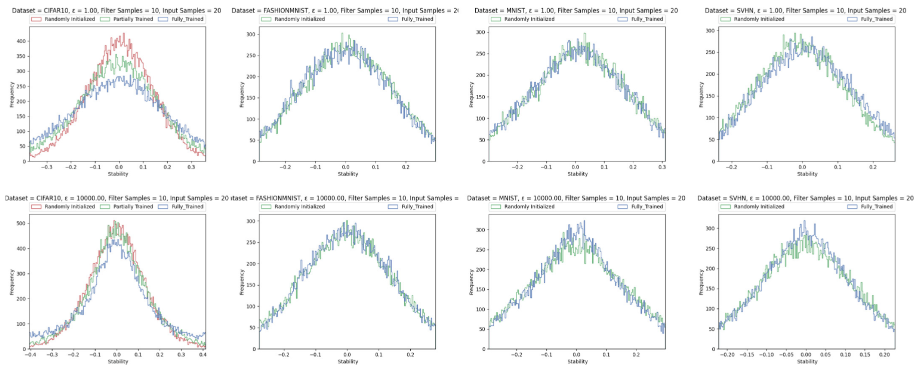


Fig. 9: Parameter-level histograms for the Stability measure for models trained on CIFAR-10, FashionMNIST, MNIST, and SVHN datasets. Values outside the .9 quantile are excluded from the histogram range. The attributions appear to be similarly distributed for both choices of locality radius ε, and while there does appear to be some dependence on model training for the CIFAR-10 ($\varepsilon = 1.0$) results, any additional trends are not immediately qualitatively clear. Since the Stability measure is defined on a bounded support of $[-1, 1]$ as the expectation of a cosine, we should strongly consider distributions such as the beta distribution for development of future statistical tests and inferences based on Stability.

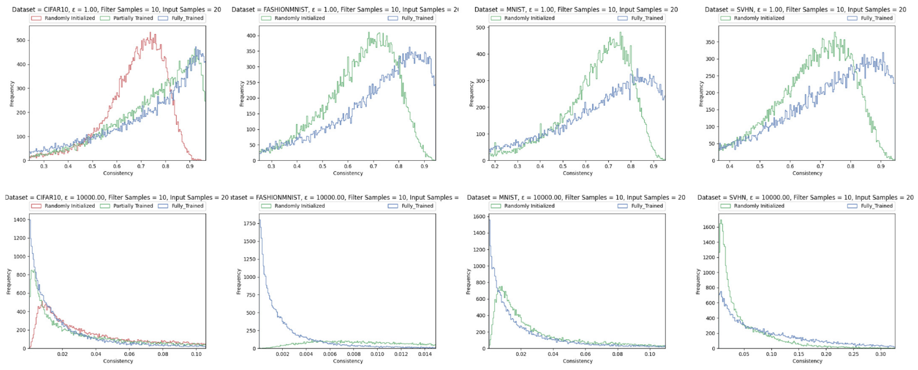


Fig. 10: Parameter-level histograms for the Consistency measure for models trained on CIFAR-10, FashionMNIST, MNIST, and SVHN datasets. Values outside the .9 quantile are excluded from the histogram range. We can note the two distinct paradigms for locality radius $\varepsilon = 1.0$ and $\varepsilon = 10^4$, and a clear distinction between the distributions for trained and untrained models in both cases. Like the Stability measure, since the Consistency measure is defined on a bounded support of $[-1, 1]$, we should be mindful of this when developing any statistical tests.

5 Conclusions

We have developed a novel methodology for explaining model parameters, and have verified its ability to identify important and unimportant neurons. By considering the product space of inputs and parameter values, we are now able to generalize the family of integrated attribution measures proposed in [21] from only input and intermediate feature attribution to also accommodate parameter attribution. We have identified several unique sources of parameter attribution information by using our new formulation to compute the four measures proposed in [21] for varying locality radius. Since the underlying family of attributions is very diverse, our initial success in identifying important neurons justifies a much more thorough investigation into the various relevant hyperparameters (input sample size, parameter sample size, locality radius, etc.) and merits the development of additional measures beyond the four which we have explored in this work. Furthermore, we have demonstrated our method's ability to identify neurons important specifically for single classes. If our parameter attributions are incorporated into more sophisticated unlearning and model reduction methods, we will likely observe even better utility. We have also identified several preliminary trends and patterns with respect to the distributions of Integrated Gradients, Gradient Variance, Stability, and Consistency for trained versus untrained models. Our study confirms that this family of attribution measures is a rich source of relevant model information which begs further study toward the end of both explaining and improving model behavior. Since our methodology is immediately applicable to any machine learning model for which parameter gradients can be computed, and can accommodate any new measures developed using the framework of Generalized Integrated Attributions, we should expect the utility of this method to only increase as additional novel and useful integrated measures continue to be proposed.

5.1 Future Work

While we include much additional pruning data in the supplemental information, we should still explore a wider range of locality radii and perform experiments in which we prune in ascending order of attribution value as opposed to descending value, or in which we sort each parameter using multiple attribution measures at once. Additionally, While we have collected data for the Resnet-18 and Resnet-34 architectures for several image datasets, we can also investigate a wider variety of model architectures and data tasks beyond image recognition.

In the future we may be able to visualize the semantic effect and role of the model's important parameters by inspecting the types of features extracted by these parameters. This, coupled with input and feature attribution, may give us a broader understanding of model attention.

The qualitative trends observed in the distributional study above justify a more rigorous statistical analysis such as Analysis of Variance (ANOVA) in the future to search for higher order and mixed effects. The distributions of each attribution measure can also be fit to parametric distributions such as Student's t, gamma, and beta distributions in order to directly quantify the effect of model training and other hyperparameters. In particular, since we directly derived that Integrated Gradients and Gradient Variance measures should follow Student's t and gamma distributions respectively, we can immediately begin developing statistical tests to explain and improve models based on these two measures. We can additionally continue to collect more data regarding how attributions depend on model training and accuracy. We can similarly continue to investigate how hyperparameters such as the locality radius ε, the training dataset, number of classes, and filter size affect the distribution of attributions. We might also inspect class-wise attribution distributions as a means of further quantifying and explaining how each model responds to a particular class.

Finally, though out-of-scope for this work, future studies should pursue training models using the method of attribution priors proposed by [7]. It is possible that the attribution measures studied in this work could be used to train models more robustly and accurately, so any such opportunities for explainable improvement should be investigated.

Acknowledgements. This material is based upon work supported by the National Science Foundation under Grant No. 2134237. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. Alqahtani, A., Xie, X., Essa, E., Jones, M.W.: Neuron-based network pruning based on majority voting. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 3090–3097 (2021). https://api.semanticscholar.org/CorpusID:233877899
2. Ancona, M., Ceolini, E., Ã–ztireli, C., Gross, M.: Towards better understanding of gradient-based attribution methods for deep neural networks. In: International Conference on Learning Representations (2018). https://openreview.net/forum?id=Sy21R9JAW
3. Barkan, O., Elisha, Asher, Y., Eshel, A., Koenigstein, N.: Visual explanations via iterated integrated attributions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 2073–2084 (October 2023)
4. Chundawat, V.S., Tarun, A.K., Mandal, M., Kankanhalli, M.: Zero-shot machine unlearning. Trans. Info. For. Sec. **18**, 2345–2354 (2023). https://doi.org/10.1109/TIFS.2023.3265506
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
6. Dhamdhere, K., Sundararajan, M., Yan, Q.: How important is a neuron. In: International Conference on Learning Representations (2019). https://openreview.net/forum?id=SylKoo0cKm
7. Erion, G., Janizek, J., Sturmfels, P., Lundberg, S., Lee, S.I.: Improving performance of deep learning models with axiomatic attribution priors and expected gradients. Nat. Mach. Intell. **3**, 1–12 (2021).https://doi.org/10.1038/s42256-021-00343-w

8. Fel, T., et al.: Craft: concept recursive activation factorization for explainability. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2711–2721 (2023)
9. Ghorbani, A., Zou, J.Y.: Neuron shapley: Discovering the responsible neurons. ArXiv **abs/2002.09815** (2020). https://api.semanticscholar.org/CorpusID:211258568
10. Hase, P., Xie, H., Bansal, M.: The out-of-distribution problem in explainability and search methods for feature importance explanations. Adv. Neural Inf. Process. Syst. **34** (2021)
11. Krizhevsky, A.: Learning multiple layers of features from tiny images (2009)
12. LeCun, Y., Cortes, C., Burges, C.: MNIST handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist**2** (2010)
13. Leino, K., Li, L., Sen, S., Datta, A., Fredrikson, M.: Influence-directed explanations for deep convolutional networks. In: 2018 IEEE International Test Conference (ITC), pp. 1–8 (2018)
14. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 4768–4777. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (2017)
15. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning (2011)
16. Sajjad, H., Durrani, N., Dalvi, F.: Neuron-level interpretation of deep NLP models: a survey. Trans. Assoc. Comput. Linguist. **10**, 1285–1303 (2021). https://api.semanticscholar.org/CorpusID:237353268
17. Shrikumar, A., Greenside, P., Kundaje, A.: Learning important features through propagating activation differences. In: Proceedings of the 34th International Conference on Machine Learning - Volume 70, pp. 3145–3153. ICML'17, JMLR.org (2017)
18. Shrikumar, A., Su, J., Kundaje, A.: Computationally efficient measures of internal neuron importance. CoRR arxiv preprint arxiv: abs/1807.09946 (2018)
19. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: visualising image classification models and saliency maps. In: Proceedings of the International Conference on Learning Representations (ICLR). ICLR (2014)
20. Srinivas, S., Fleuret, F.: Full-gradient representation for neural network visualization. Adv. Neural Inf. Process. Syst. **32** (2019)
21. anonymous submission: Generalized integrated gradients. In: Under Review. pp. supplemental materials (2023)
22. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: International Conference on Machine Learning, pp. 3319–3328. PMLR (2017)
23. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms (2017)
24. Zhang, J., Bargal, S.A., Lin, Z., Brandt, J., Shen, X., Sclaroff, S.: Top-down neural attention by excitation backprop. Int. J. Comput. Vision **126**(10), 1084–1102 (2018)

Correlation Weighted Prototype-Based Self-supervised One-Shot Segmentation of Medical Images

Siladittya Manna[1](✉), Saumik Bhattacharya[2], and Umapada Pal[1]

[1] Indian Statistical Institute, Kolkata, India
{siladittya_r,umapada}@isical.ac.in
[2] Indian Institute of Technology Kharagpur, Kharagpur, India
saumik@ece.iitkgp.ac.in

Abstract. Medical image segmentation is one of the domains where sufficient annotated data is not available. This necessitates the application of low-data frameworks like few-shot learning. Contemporary prototype-based frameworks often do not account for the variation in features within the support and query images, giving rise to a large variance in prototype alignment. In this work, we adopt a prototype-based self-supervised one-way one-shot learning framework using pseudo-labels generated from superpixels to learn the semantic segmentation task itself. We use a correlation-based probability score to generate a dynamic prototype for each query pixel from the bag of prototypes obtained from the support feature map. This weighting scheme helps to give a higher weightage to contextually related prototypes. We also propose a quadrant masking strategy in the downstream segmentation task by utilizing prior domain information to discard unwanted false positives. We present extensive experimentations and evaluations on abdominal CT and MR datasets to show that the proposed simple but potent framework performs at par with the state-of-the-art methods.

1 Introduction

Semantic Segmentation is one of the critical applications in computer vision. Applications of semantic segmentation to medical image analysis for assisting medical personnel in disease diagnosis are also plenty. For efficient and reliable analysis of medical images, contemporary deep-learning methods require large-scale datasets annotated by expert medical personnel. However, unlike natural image datasets, obtaining annotated high-quality medical image datasets is time-consuming and labour-intensive. To avoid the scarcity of large-scale datasets in medical image analysis, the few-shot learning paradigm has gained popularity among researchers.

In this work, we adopt the few-shot learning approach for learning to segment organs from MR or CT query scans, from a limited number of given support MR or CT slices and their corresponding ground truth segmentation masks. Depending on the pipeline, few-shot segmentation frameworks can be primarily of two types: prototype feature learning and affinity learning [18]. Prototype feature learning consists of constructing prototypes utilizing the support image and the support mask information. Each

https://doi.org/10.1007/978-3-031-78192-6_2

prototype represents a defined spatial region in the support image. These prototypes are used to find pixels in the query image which are similar to them and are scored accordingly to segment it into foreground and background. Prototype-based features are more robust to noise than pixel-based features [18]. Prototypical methods also drop spatial information, which is important when the variation between support and query images is considerably significant [18]. Prototypical methods are also responsible for losing discriminability because of the masked pooling process to generate prototypes [18]. To address this issue, we create prototypes for foreground and background pixels, which preserve the contextual spatial information required for effective discrimination between the foreground and background pixels. To prevent loss of information, we adopt a correlation-weighted aggregation approach such that the information of all the prototypes corresponding to foreground or background is present in the aggregated prototype.

In PANet [33], a prototypical alignment-based strategy was proposed, wherein the masked support image embedding is mapped to the feature space, and the query mask is predicted by matching the query prototypes to the nearest prototype in the embedding space. However, PANet resorts to a global masked pooling operation, which is not suitable for medical image segmentation, as it can result in the loss of spatial orientation information. In ALPNet [25], which is also a prototype-based framework, a local prototype-based approach is adopted to preserve local information using an adaptive local prototype pooling framework. However, such an approach ignores the global contextual information. In this work, we generate a single prototype for each pixel in the query feature map, that encodes both the global and local spatial context. We resort to a correlation-based aggregation approach, where the prototypes which are similar to a particular query pixel or located close in terms of spatial context, as well as its neighbourhood, will get a higher score than prototypes that are located far away in terms of spatial context. The probability scores are used to generate a weighted prototype for each query pixel. The correlation-based probability weighting scheme allows dynamic prototype generation for each query pixel by giving more weight to contextually related prototypes. In addition to the framework design mentioned above, we also utilize prior domain information to further reduce the effect of false positives in the final downstream task by using *quadrant masking* scheme.

The main contributions of our work are as follows.

- We propose a novel correlation-weighted prototype aggregation-based self-supervised one-way one-shot learning framework for the segmentation of organs from abdominal magnetic resonance or computed tomography scans.
- We also propose a prior domain knowledge-informed quadrant masking scheme for discarding false positives in medical image segmentation tasks.
- Extensive experimental evidence on two datasets on abdominal magnetic resonance imaging and computed tomography shows the efficacy of the proposed simple but potent method.

2 Related Work

2.1 Few Shot Segmentation

One of the first pioneering works in Few Shot Segmentation (FSS) was presented in [28], where a conditioning branch was used to predict weights, which serves as classifier weights for the query image feature obtained from the segmentation branch. The idea was extended in [26] using sparse positive and negative support pixels. In [27], a guided network is used to utilize information from the latent representation from the FSS task to segment query pixels. This work was further improved and extended in [3,29,30,37].

[7] presents one of the first works in the Prototypical Learning paradigm, by fusing prototypes from support images with the query image features using similarity scores. A leap in the paradigm of prototype learning was shown in [33], where the prototype alignment strategy was introduced for maintaining cyclic consistency between the ground truth and the predicted segmentation mask inducing regularizing effect in training. Instead of altering the input structure as in [7,26], the authors in [38] adopted a separate segmentation guidance framework based on similarity. [21] argue that the previous prototype-based methods do not take into account the various appearance of different parts in an object and propose a prototype-based part-aware framework to capture rich and fine-grained features. [35] also pointed out that the primary disadvantage of existing prototype-based methods is the pooling operations which destroy the spatial layout information of the objects, and thereby proposed a prototype mixture model to solve the semantic ambiguity in prototype-based models.

To preserve the spatial correspondence between support and query image pixels, [20] uses a partial optimal transport-based matching. A multi-level variation of the same was done in [32]. [8,18] also aim to solve the same problem. [19,36] aim to capture the intrinsic details to improve segmentation quality. [4,14] attempts to reduce testing bias in the FSS setting. An attempt to improve the discrimination between similar classes is presented in [22].

2.2 Self-supervised Segmentation

The application of self-supervised learning frameworks in segmentation, although limited, follows two paths. One where the pre-training task is different from the downstream task of segmentation, and the other where both are the same. Works like [5,11,13,15,16,23,31,39] uses a pre-training stage to learn representations from the base dataset and then utilises the representation for a downstream semantic segmentation task. [10] uses unsupervised saliency to generate object proposals and then optimizes a contrastive learning objective on the features obtained from the proposals to learn representations for semantic segmentation. For the second type, pseudo-masks are used as the segmentation masks in the pre-training stage. The above strategy is adopted in [1,2,24,25]. In [2], the authors use the output masks of a momentum update net as target pseudo-masks. In [24,25], the pseudo-masks are generated using the Felzenszwalb algorithm [9] and the model using a few-shot learning strategy, the query image is an augmented version of the support image itself.

In our work, to address the issue of false positives and also to preserve the spatial layout information [35], we propose a dynamic prototype-based framework that will weigh the background prototypes according to correlation with the pixel features.

3 Methodology

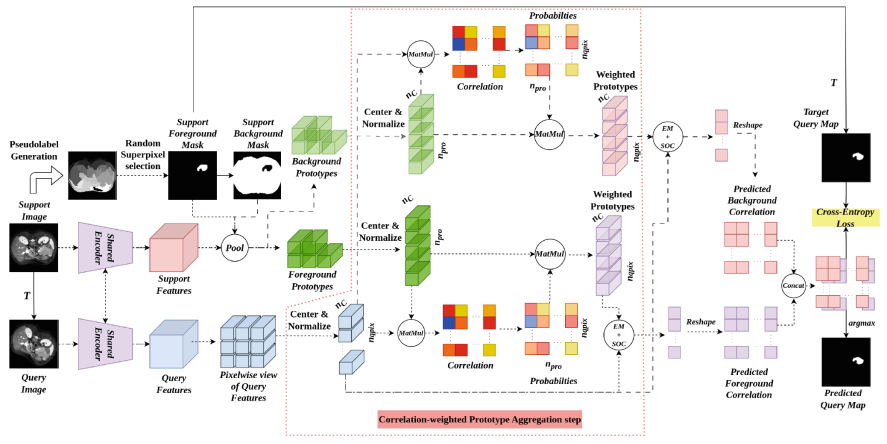


Fig. 1. The figure depicts the entire working principle of the proposed framework. For clarity, we have also indicated the novel proposed correlation-weighted prototype aggregation step using a dotted red bounding box. ***T*** (Color figure online) indicates the transformation applied to the support image to generate the query image only in the pre-training stage. ***Pool*** denotes pooling the feature map of the region denoted by the mask. ***MatMul*** denotes Matrix Multiplication. '***EM+SOC***' denotes Element-wise Multiplication and Sum over Channels. ***Concat*** denotes the concatenation operation. (Best viewed at 300%)

3.1 Problem Definition

In the few-shot learning framework, the dataset is split into two parts, training dataset $\mathcal{D}_{train}$ and testing dataset $\mathcal{D}_{test}$. In both training and testing datasets, each sample consists of the input and the associated ground truth, $(\mathcal{X}, \mathcal{Y})$. In our work, $\mathcal{X}$ and $\mathcal{Y}$ correspond to slices from the abdominal MR or CT scans and the associated superpixel pseudo-masks generated in the pre-training stage, respectively. Whereas, in the evaluation or testing stage, the original ground truth masks for each organ are used with the MR or CT slices. Furthermore, no overlap should be present between the classes present in $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$.

In the few-shot learning framework, we need to consider two sets of data, the *Support* set $\mathcal{S}$ and the *Query* set $\mathcal{Q}$. The *Support* set $\mathcal{S}$ consists of the tuple $\{\mathcal{X}_s^i, \mathcal{Y}_s^i(l)\}_{i=1}^{k}$, where $\mathcal{X}_s^i$ is the i-th sample in the Support set with the segmentation mask $\mathcal{Y}_s^i(l)$ for the class l, where l belongs to the set of novel classes available during the testing phase. The primary objective is to learn an approximate function f which takes as input the support set $\mathcal{S}$ and the query image $\mathcal{X}_q$ and predicts the binary mask $\hat{\mathcal{Y}}_q$ of the unseen classes in $\mathcal{X}_q$, denoted by the support mask $\mathcal{Y}_s(l)$.

The support set $\mathcal{S}$ is a subset of $\mathcal{D}_{train}$. During training, the input to the model is $(\mathcal{S}, \mathcal{X}_q)$. Such a pair is called an *episode*. If during training, the value of k is 1, that is, we use only a single image in the support set, then the learning is known as one-shot learning, which we adopt in this work. If $k > 1$, it is known as few shot learning. If the number of classes is N, then we call it N-way k-shot learning.

3.2 Overview of the Proposed Approach

We propose a self-supervised approach for one-shot learning of segmentation in MR and CT scans. The skeleton of our framework is based on a prototype-based segmentation strategy. For the first step, we use superpixel-based pseudo-segmentation mask generation. In our work, we adopt a dynamic prototype generation approach, one for each query feature map pixel. The dynamic nature of the prototype is the result of the aggregation step using correlation-based probability scores. The final score is obtained by calculating the correlation score of each query pixel to their assigned prototype.

In the downstream segmentation phase, we utilize the slide index information as in [24], to filter out false positives (FP) obtained from other organs or regions on abdominal scans. Furthermore, we also propose to use the spatial location information of the respective organ (l) to segment.

3.3 Generation of Pseudo Segmentation Masks

We follow the same strategy as in [24] for the generation of pseudo-segmentation masks. As stated in [24], superpixel-based segmentation satisfies two properties: for each class, the representations should be clustered to be discriminative under a similarity metric, and the representations should also be invariant across images to ensure robustness. Otherwise, the regions that denote the same class in the support and query images would not be mapped together in the feature space. A superpixel-based clustering strategy ensures that regions with similar pixel features are clustered together. This ensures consistency over each of the pseudo-labels as well.

For every slice in abdominal scans (say $\mathcal{X}_i$), the Felzenszwalb image segmentation algorithm [9] $\mathcal{F}$ is applied to the slice to generate the super pixels, $\mathcal{S}_p = \mathcal{F}[\mathcal{X}^i]$). During self-supervised training, a superpixel is randomly chosen, $l_s \sim \mathcal{U}[0, |\mathcal{S}_p| - 1]$ and converted to a binary mask to be used as the segmentation mask. A sample of the superpixels obtained is shown in Fig. 1.

$$\mathcal{Y}_s = [\mathcal{S}_p \in \{l_s\}] \tag{1}$$

3.4 Feature Extraction

Each episode $(\mathcal{S}, \mathcal{X}_q)$ is passed through the encoder f_θ, which gives us the support and query feature maps, which we denote by $f_\theta(\mathcal{X}_s)$ and $f_\theta(\mathcal{X}_q)$. In our case, the encoder takes an input of dimension $3 \times 256 \times 256$ and outputs a feature map with dimensions $256 \times 32 \times 32$. We use the $deeplab\_v3$ version of ResNet101 available from the $torchvision$ library. To ensure that the output dimensions match the specifications mentioned above, we used $dilation$ in the last two layers of the encoder, similar to [25].

3.5 Correlation Weighted Prototype Aggregation

The principle component of our proposed framework is the prototype aggregation module. This novel correlation-weighted prototype-aggregation module primarily consists of four steps: 1) Prototypes Extraction, 2) Correlation computation, 3) Probability score computation, and 4) Prototype Aggregation. The prototype aggregation steps are done separately for foreground and background.

Prototype Extraction. We do not extract the foreground features by merely doing global average pooling using the support mask $\mathcal{Y}_s$. In this case, we follow the steps described in [24], for extracting the foreground and background prototypes. The first step to obtaining the foreground (or background) prototypes is to downsample the segmentation mask to spatial dimension $H \times W$ using an average pooling operation with a window of spatial dimensions 4×4. However, using an average pooling operation may result in values that are not binary (0 or 1). To get a downsampled binary mask, we threshold the interpolated mask. For the foreground, we select a threshold that is 0.8 times the maximum value of the downsampled mask. For the background, we use a threshold that is equal to the mean of the downsampled mask, following [34].

$$\mathcal{Y}_{s(H\times W)} = [AvgPool_{4\times 4}(\mathcal{Y}_s) > \tau] \tag{2}$$

where $AvgPool$ refers to the Average Pooling operation applied on the binary mask $\mathcal{Y}_s$, and τ is the threshold. However, we find that, for the label sets containing Liver and Spleen, using a threshold of 0.95 for both foreground and background worked better than the aforesaid thresholds (See Table 3). Next, the locations w where the downsampled binarized mask $\mathcal{Y}_{s(H\times W)}$ is non-zero are processed. The pixels in the support feature map $f_\theta(\mathcal{X}_s) \in \mathbb{R}^{D\times H\times W}$, whose locations match those in w, are chosen as prototypes. For the foreground prototypes, we also include the global prototype with the obtained prototypes to avoid an empty set of prototypes resulting from the averaging over the small area of the foreground, following [25].

$$\mathcal{P} = f_\theta(\mathcal{X}_s)[\mathcal{Y}_{s(H\times W)} \in \{1\}] \tag{3}$$

Before calculating the cosine similarity between the prototypes $\mathcal{P} \in \mathbb{R}^{D\times N_{pro}}$ and the pixels of the query feature map $f_\theta(\mathcal{X}_q)$, we subtract the mean of each of the N_{pro} prototypes along the channel dimension.

$$\mathcal{P}^j = \mathcal{P}^j - \frac{1}{D}\sum_{d=1}^{D} \mathcal{P}^j[d] \tag{4}$$

where $\mathcal{P}^j$ is the j-th prototype. The steps mentioned above are done for both the foreground and background prototypes. To obtain the foreground prototypes, we simply take $\mathcal{Y}_s$ as the foreground mask $\mathcal{Y}_s^{FG}$, whereas for the background prototypes, we take $\mathcal{Y}_s^{BG} = 1 - \mathcal{Y}_s^{FG}$ as the background mask.

Query Features Centering. The same mean subtraction operation is also done for the query pixels in the output feature map, as follows,

$$f_\theta(\mathcal{X}_q)_{h,w} = f_\theta(\mathcal{X}_q)_{h,w} - \frac{1}{D}\sum_{d=1}^{D} f_\theta(\mathcal{X}_q)_{h,w}[d] \tag{5}$$

where, $\{h, w\}$ denotes the location of the pixels in the feature map.

Correlation Computation. Having obtained the query feature map $f_\theta(\mathcal{X}_q)$ of dimensions $D \times H \times W$ and prototypes $\mathcal{P}$ of dimensions $D \times N_{pro}$, we proceed to compute the cosine similarity between these entities. This results in a correlation matrix or a cosine similarity matrix $\mathcal{C}$, which has dimensions $N_{pro} \times H \times W$. This 3D matrix represents the correlation score of all the prototypes obtained in the previous step for each pixel in the query feature map.

$$\mathcal{C}(j, h, w) = f_\theta(\mathcal{X}_q)(h, w) \odot \mathcal{P}^j \tag{6}$$

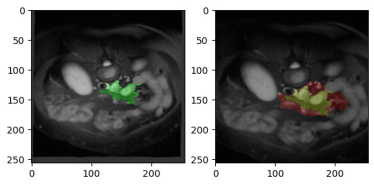
where $\odot$ indicates the operation of the dot product, $f_\theta(\mathcal{X}_q)(h, w)$ denotes the feature at the location (h, w), and $\mathcal{P}^j$ is the j-th prototype. $\mathcal{C}(j, h, w)$ has dimensions $N_{pro} \times H \times W$. The correlation score indicates how similar each prototype is to the query feature map pixels. Intuitively, a prototype $p \in \mathcal{P}$ with a higher correlation score with a pixel on the query feature map $f_\theta(\mathcal{X}_q)(h, w)$ can be said to be more similar than a prototype with a lower correlation score, in terms of feature similarity. As the feature extractor uses dilation, the receptive field of each pixel in the output feature map has a very large receptive field. Hence, contextual or neighbourhood information is encoded in each foreground prototype $\mathcal{P}^{FG}$. This contextual information will help distinguish the region indicated by the support mask $\mathcal{Y}_s$ and the other regions.

Probability Score Computation. The probability of each prototype being similar to a particular query pixel is calculated by taking softmax over the prototypes as follows:

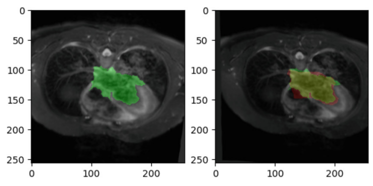
$$\mathcal{M}_{prob}(h, w) = \text{softmax}_{j \in \mathcal{P}}[\mathcal{C}(h, w)/t] \tag{7}$$

where $\mathcal{M}_{prob}(h, w)$ denotes the probability of the prototypes $\mathcal{P}$ with respect to the query pixel at the location (h, w), and t is a temperature parameter. $\mathcal{M}_{prob}(h, w)$ has dimensions $N_{pro} \times H \times W$.

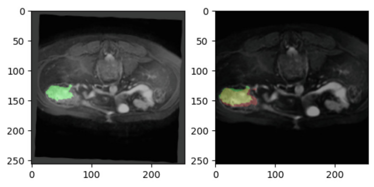
When calculating the scores for the background prototypes $\mathcal{P}^{BG}$, the background query pixels which are spatially close to a particular background query pixel (say, $f_\theta(\mathcal{X}_q)(h, w)$), will yield different correlation scores. This may result in erroneous predictions or increased false positives if we weigh all the prototypes equally. The background prototypes which are spatially farther from $f_\theta(\mathcal{X}_q)(h, w)$ region or feature-wise dissimilar will bring the final score down, thereby increasing false positives. This requires a dynamic prototype that captures contextual information effectively. This is made possible by giving a probabilistic weightage to contextually similar prototypes.



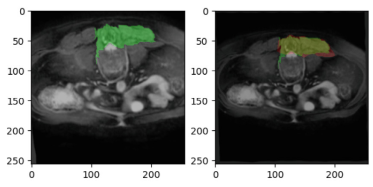
(a) Iteration: 25000



(b) Iteration: 50000



(c) Iteration: 75000



(d) Iteration: 100000

Fig. 2. Predictions in training phase at 25K, 50K, 75K, 100K iterations. The left image in Figs. 2a-2d is the support image $\mathcal{X}_s$ and the support mask is denoted in *green*. The right image in Figs. 2a-2d is the query image. The ground truth is denoted by *green* and the predicted mask is indicated by *red*. (Use 300% zoom for better visibility)

Prototype Aggregation. The weighting scheme necessary for a dynamic and contextual prototype generation is done by aggregating the prototypes on the basis of probability scores obtained from the correlation values between the prototypes and the query pixels. The aggregated dynamic prototype is obtained by a weighted average of all the prototypes using the probabilities obtained in the previous step, as follows:

$$\mathcal{P}_{agg}(h, w) = \sum_{j=1}^{N_{pro}} \mathcal{M}_{prob}(h, w) \cdot \mathcal{P}^j \tag{8}$$

where $\mathcal{P}_{agg}(h, w) \in \mathbb{R}^{D\times1\times1}$ denotes the aggregated prototype for the query pixel at location (h, w), $\mathcal{P}^j \in \mathbb{R}^{D\times1\times1}$ denotes the j-th prototype.

3.6 Mask Prediction

Computing the Final Score. Once we have the aggregated prototype, we can compute the final scores for each query pixel. Thefinal score is calculated by simply calculating the cosine similarity of the aggregated prototype $\mathcal{P}_{agg}(h, w)$ with the pixel feature of the query in location (h, w), as follows.

$$s_{FG}(h, w) = \mathcal{P}_{agg}^{FG}(h, w) \odot f_{\theta}(\mathcal{X}_q)(h, w) \tag{9}$$

$$s_{BG}(h, w) = \mathcal{P}_{agg}^{BG}(h, w) \odot f_{\theta}(\mathcal{X}_q)(h, w) \tag{10}$$

where $s_{FG}(h, w)$ and $s_{BG}(h, w)$ are the scores for the query pixels with respect to the foreground and background prototypes, respectively, and $\mathcal{P}_{agg}^{FG}$ and $\mathcal{P}_{agg}^{BG}$ are the aggregated prototypes for the foreground and background, respectively.

Final Prediction. The final prediction is obtained by choosing the class with the highest probability or the similarity scores for the foreground and background for each query pixel. Thus, the final prediction for each query pixel is obtained as follows:

$$\hat{\mathcal{Y}}_q(h, w) = \underset{\{BG,FG\}}{\text{argmax}}\ \underset{\{BG,FG\}}{\text{softmax}}[s_{BG}, s_{FG}] \tag{11}$$

where $\hat{\mathcal{Y}}_q(h, w)$ is the predicted query mask. A few examples of the query predictions and the associated ground truth from the training stage are shown in Fig. 2.

3.7 Training Pipeline

In each iteration t, we take an episode $((\mathcal{X}_s, \mathcal{Y}_s), \mathcal{X}_q)$ as input, and the model predicts $\hat{\mathcal{Y}}_q$ as output. The query image $\mathcal{X}_q$ is obtained by applying geometric and intensity transformations on the support image $\mathcal{X}_s$, that is, $\mathcal{X}_q = \mathcal{T}_{geo}(\mathcal{T}_{int}(\mathcal{X}_s))$. The pseudo-ground truth is obtained by only applying the geometric transformation $\mathcal{T}_{geo}$ on the support mask $\mathcal{Y}_s$ for a randomly chosen pseudo-superpixel class.

Geometric transformations $\mathcal{T}_{geo}$ consist of affine and elastic transformations to emulate the changing shapes of the class labels in the downstream task. Intensity transformations $\mathcal{T}_{int}$ consist of gamma transformation to account for the varying intensity of the pixels between scans of different patients. The parameters for the geometric and intensity transformation are the same as used in [24].

Since the encoder f_{θ} output is of spatial dimension 32×32, we interpolate the final prediction to 256×256 before calculating the loss using bilinear interpolation. To optimize the model parameters, we minimize the cross-entropy loss $\mathcal{L}_{ssl}^{t}$ for all the query pixels.

$$\mathcal{L}_{ssl}^{t}(\theta) = -\underset{h,w}{\mathbb{E}}\left[\lambda_{h,w}\mathcal{T}_{geo}(\mathcal{Y}_s^t)(h, w)\log\left(\hat{\mathcal{Y}}_q^t(h, w)\right)\right] \tag{12}$$

where $\hat{\mathcal{Y}}_q^t(h, w)$ is the predicted output of $\mathcal{T}_{geo}(\mathcal{Y}_s^t)(h, w)$ taking $\mathcal{X}_q = \mathcal{T}_{geo}(\mathcal{T}_{int}(\mathcal{X}_s))$ as query. (h, w) denotes the location in the predicted query mask or pseudo-ground-truth mask. $\lambda_{h,w}$ denotes the class weight applied during training.

Similar to [24], we also adopt the Cyclic Consistency Regularization (CCR) following [33]. To implement CCR, we interchange query and support images. For the support mask, we use the predicted query output $\hat{\mathcal{Y}}_q$ as the foreground mask, and the support mask $\mathcal{Y}_s$ in the forward iteration is used as the pseudo-ground-truth. The episode in the CCR step consists of $((\mathcal{X}_q, \hat{\mathcal{Y}}_q), \mathcal{X}_s)$. In the CCR step, we use an initial threshold of 0.95 in the prototype extraction step to filter out noisy predictions, following which the aforementioned steps are followed. Otherwise, we see a drop in performance by about 2% in dice score. The CCR loss is represented as

$$\mathcal{L}^t_{reg}(\theta) = - \mathop{\mathbb{E}}_{h,w} \left[\mathcal{Y}^t_s(h, w) \log \left(\hat{\mathcal{Y}}^t_s(h, w) \right) \right] \tag{13}$$

where $\hat{\mathcal{Y}}^t_s(h, w)$ is the predicted output of $\mathcal{Y}^t_q)(\hat{h}, w)$ taking $\mathcal{X}_s$ as a query.

Hence, the total loss is as follows,

$$\mathcal{L}^t = \mathcal{L}^t_{ssl} + \mathcal{L}^t_{reg} \tag{14}$$

To handle the imbalance, we set the class weights at 0.05 for the background pixels or the class label 0, and a weight of 1.0 for the foreground pixels or the class label 1.

Furthermore, it is to be noted that during training, we divide the class labels in abdominal CT or MR into two parts, namely, upper abdomen consisting of *right kidney* and *left kidney*, and lower abdomen consisting of *liver* and *spleen*. When training on slices from the upper abdomen, we do not include slices containing lower abdomen classes and vice versa.

3.8 Validation Without Fine-Tuning

Following [24,25], we evaluate our model on a validation split, without further fine-tuning on the whole dataset. Although we don't fine-tune the model, we do use the class label information from the dataset. For a class label l, we only take the slices $[z_{min}, z_{max}]$ in which l is present for the final predictions.

One-Shot Segmentation. The evaluation strategy follows a one-shot segmentation task. Among the scans in the validation split, the last scan (if arranged in order) is selected as the support scan. The range of slices $[z_{min}, z_{max}]$ in both support and query scans is divided into 3 parts following the evaluation strategy followed in [12]. From the three support scan splits, the middle slice is selected as the support image for the whole of the corresponding query part. The evaluation step then follows the flow of the one-shot segmentation task, that is, a tuple $((\mathcal{X}^p_s, \mathcal{Y}^p_s), \mathcal{X}^{p,i}_q)$ is fed to the pre-trained model f_θ as input to obtain the predicted query mask $\hat{\mathcal{Y}}^{p,i}_q$, where p is the part in which the slices belong and $i \in [z^q_{min}, z^q_{max}]$ is the index of the slices of query scan in which the class label l is present.

Quadrant Masking Scheme. During training, the classes in abdominal MRI or CT datasets are split into two parts, the upper abdomen (right and left kidneys) and the lower abdomen (liver and spleen). During validation, we divide each slice into quadrants. For each class label l, we identify which quadrants are occupied by it. The final predictions obtained from the one-shot segmentation step are masked such that the predictions from the quadrants in which the class label l is present, are considered for the final metric calculation. For example, the class *right kidney* is present in the left half of an MRI or CT slice. Hence, we mask the right half of each slice while making the final prediction. The quadrant masking scheme uses the quadrant information as a piece of soft prior information about the location of the target organ. To the best of our knowledge, no work has employed this scheme before. To fully understand the role of this quadrant masking scheme, we conduct an ablation study in Sec.4.4, where we observe the significant effects of the soft prior knowledge in boosting the segmentation performance.

Validation Metric. To measure the performance of the proposed model, we use the Dice score as a metric, as is usually done in the medical image segmentation literature [24,25,33].

4 Experiments and Results

4.1 Implementation Details

Training and evaluation were implemented using PyTorch. The training was done on a 24GB NVIDIA A5000 GPU. The average training time for each training run consisting of 100K iterations was about 4.5 h. We used a batch size of 1. The initial learning rate of the SGD optimizer was set to $1e-3$ and decayed at 0.95 per 1K iterations.

4.2 Datasets

To demonstrate the effectiveness of the proposed approach, we diversified the input modalities by including both Magnetic Resonance (MR) and Computed Tomography (CT) scans in our work.

For the Magnetic Resonance (MR) modality, we used the Combined Healthy Abdominal Organ Segmentation (CHAOS) Challenge (Task 5) from ISBI 2019 [17]. This dataset contains 20 3D T2-SPIR MRI scans.

For the Computed Tomography (CT) modality, we used data from the 2015 MICCAI Multi-Atlas Abdomen Labeling Challenge (SABS). It contains abdominal scans from 30 patients.

For the experiments, we used a five-fold cross-validation setting, that is, in an experimental run, one-fifth of the dataset (a fold) is used as a validation set while the rest is treated as a training set.

4.3 Results and Comparison

Quantitative Performance Analysis. In this section, we present the results obtained by the proposed model on the two datasets: CHAOS and SABS. From Tables 1 and 2, we can see that the proposed framework outperforms ALPNet [24] and also outperforms several current state-of-the-art methods in several classes. The bold font and the underlined text indicate the best and the second-best performance, respectively. The proposed algorithm outperforms CRAPNet on the CHAOS dataset and produces competitive results on the SABS dataset without any further fine-tuning of hyperparameters. This shows that the dynamic prototype aggregation technique improves the representation learning and generalizability of the model.

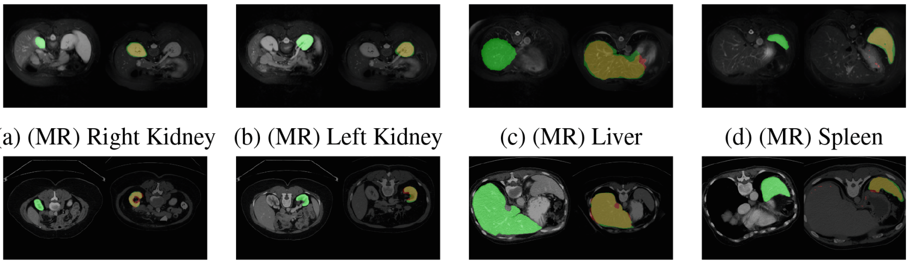
Table 1. DICE score on Abdominal MR (CHAOS) Dataset. Reported Values are with Single Support Scan.

Method	Supervised	RK	LK	Liver	Spleen	Mean
SE-Net [12]	✓	61.32	62.11	27.43	51.80	50.66
Vanilla PANet [33]	✓	38.64	53.45	42.26	50.90	46.33
ALPNet [24]	✓	58.99	53.21	37.32	52.18	50.43
SSL-PANet [24]	×	47.95	47.71	64.99	58.73	54.85
SSL-ALPNet [24]	×	78.39	73.63	73.05	67.02	73.03
CRAPNet [6]	×	**82.77**	<u>74.66</u>	<u>73.82</u>	<u>70.82</u>	<u>75.52</u>
CoWPro (Ours)	×	<u>80.45</u>	**75.30**	**75.77**	**71.51**	**75.56**

Table 2. DICE score on Abdominal CT (SABS) Dataset. Reported Values are with Single Support Scan.

Method	Supervised	RK	LK	Liver	Spleen	Mean
SE-Net [12]	✓	14.34	32.83	0.27	0.23	11.91
Vanilla PANet [33]	✓	17.37	32.34	38.42	29.59	29.43
ALPNet [25]	✓	30.40	34.96	47.37	27.73	35.11
SSL-PANet [24]	×	34.69	37.58	61.71	43.73	44.42
SSL-ALPNet [24]	×	54.82	<u>63.34</u>	**73.65**	60.25	63.02
CRAPNet [6]	×	**67.33**	**70.91**	70.45	**70.17**	**69.72**
CoWPro (Ours)	×	<u>58.99</u>	62.66	<u>73.11</u>	<u>67.97</u>	<u>65.83</u>

The qualitative performance of the proposed model can be seen from the predictions presented in Fig. 3. We can see the predictions for different organs on two different modalities compared to the ground truth. We can see that the model produces segmentation results close to the ground truth.



(a) (MR) Right Kidney (b) (MR) Left Kidney (c) (MR) Liver (d) (MR) Spleen

(e) (CT) Right Kidney (f) (CT) Left Kidney (g) (CT) Liver (h) (CT) Spleen

Fig. 3. Figure showing the predictions obtained for 4 organs, Right Kidney, Left Kidney, Liver, and Spleen for two different modalities MR (CHAOS dataset) and CT (SABS dataset). (green) Ground Truth, (red) Prediction, (yellow) Ground Truth and Prediction overlap. (Use 300% zoom for better visibility) (Color figure online)

4.4 Ablation Studies

Studying the Effect of Fixed Vs. Dynamic Thresholding. The effect of threshold on the performance of the proposed framework can be seen in Table 3, where we see the use of two different types of threshold, dynamic and fixed, for different sets of labels. In the CHAOS dataset, the use of a dynamic thresholding scheme similar to the one used in [34] works better for the labels *right kidney* and *left kidney*, whereas a fixed threshold of 0.95, similar to [25] works better for the other two labels. However, for experiments on SABS, we do not see much variation in performance with the change in thresholds. The dynamic thresholding scheme assigns a threshold of 0.8 times the max value of the downsampled foreground mask, whereas for the background it uses the mean value of the downsampled background mask. The dynamic thresholding scheme allows the model to adapt to the local pixel intensities and infuses a local spatial information in the process. However, for large organs like the spleen and liver, the effect of dynamic thresholding is not positive.

Table 3. Dice score obtained on abdominal CT and MR datasets using different thresholding schemes without Quadrant masking scheme.

Threshold	Abdominal MR				Abdominal CT			
	R. Kidney	L. Kidney	Liver	Spleen	R. Kidney	L. Kidney	Liver	Spleen
Fixed	79.06	72.24	**75.04**	**71.19**	**58.99**	62.47	73.19	66.67
Dynamic	**79.54**	**74.59**	73.87	66.33	58.24	**62.66**	**73.71**	**67.97**

Effect of Quadrant Masking. In Table 4, we observe the effect of the quadrant masking scheme on the segmentation performance on both the MR and CT datasets. Barring a few exceptions, the quadrant masking scheme has improved the dice score for all the organs. The primary reason behind the slight drop in performance can be attributed to the hard quadrant boundary assigned to the corresponding organs.

Table 4. Dice score obtained on abdominal CT and MR datasets with and without quadrant masking scheme with fixed thresholding.

Quadrant Masking	Abdominal MR				Abdominal CT			
	R. Kidney	L. Kidney	Liver	Spleen	R. Kidney	L. Kidney	Liver	Spleen
Yes	**79.66**	**75.30**	**75.77**	**71.51**	**58.99**	**63.31**	73.18	**66.85**
No	79.06	72.24	75.04	71.19	**58.99**	62.47	**73.19**	66.67

Effect of Number of Aggregated Prototypes. The correlation-weighted prototype-aggregation step aims to incorporate the information from all the prototypes to prevent loss of information, as is generally the case in prototype-based methods. However, one may argue, that using all the prototypes may induce an unintended negative effect from anatomically different but semantically similar regions, consequently causing a degradation in performance. In this ablation study, we take the Top-100%, 50%, 10%, 5%, and 2% most similar prototypes to predict foreground or background regions. This study also establishes the efficiency of our model in encoding contextual information, which is evident from the insignificant variation in the dice scores with the decrease in the number of prototypes. In Table 5, we show the dice scores for varying numbers of prototypes in the inference stage for the left and right kidney over all the folds.

Table 5. Dice score for 2×2 and 4×4 averaging window without Quadrant Masking.

Averaging Window		2×2		4×4	
Organ		RK	LK	RK	LK
Percentage of prototypes	100%	79.99	76.22	80.77	72.87
	50%	79.89	76.24	80.72	72.94
	10%	79.49	75.22	80.94	73.81
	5%	78.16	73.46	81.08	74.43
	2%	75.69	70.42	80.68	74.86

Effect of Averaging Window. In Table 5, we observe the effect of changing the averaging window in the prototype generation step. We chose the left and right kidneys from the abdominal MR dataset to study the effect of the averaging window, as the two kidneys are almost similar in shape and size but vary only in the spatial context. We observe that using the same averaging window as in training, that is, an averaging window of 4×4, the performance of the proposed framework is better for the right kidney, than using an averaging window of 2×2. On the contrary, using an averaging window of 2×2 yields better performance than using an averaging window of 4×4 on the left kidney. We believe that this discrepancy in the trend is primarily due to the different spatial contexts of the two organs.

5 Conclusion

In this work, we have presented a prototype-based framework for self-supervised one-shot learning of medical image segmentation tasks. Instead of taking a pre-training representation learning approach, we take a task-learning-based approach. To address the issue of variations in background information between the support and query images, we propose a correlation-based weighting scheme to aggregate the support prototypes according to how related the prototypes are to the query image. Therefore, each pixel on the feature map of the query has a custom prototype. The score for foreground or background is obtained by calculating the cosine similarity of the query feature map pixels with their corresponding prototype. The primary objective of constructing a prototype for each query feature map pixel is to reduce false positives in the predictions by weighing down the contribution of dissimilar prototypes in the final prediction. Despite the limitations of the proposed method, we can see that the proposed method is on par with most contemporary self-supervised segmentation methods.

References

1. Amac, M., Sencan, A., Baran, O., Ikizler-Cinbis, N., Cinbis, R.: MaskSplit: self-supervised meta-learning for few-shot semantic segmentation. In: 2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 428–438. IEEE Computer Society, Los Alamitos, CA, USA (2022). https://doi.org/10.1109/WACV51458.2022.00050, https://doi.ieeecomputersociety.org/10.1109/WACV51458.2022.00050
2. Araslanov, N., Roth, S.: Self-supervised augmentation consistency for adapting semantic segmentation. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 15379–15389. IEEE Computer Society, Los Alamitos, CA, USA (2021). https://doi.org/10.1109/CVPR46437.2021.01513, https://doi.ieeecomputersociety.org/10.1109/CVPR46437.2021.01513
3. Bhunia, A.K., Bhunia, A.K., Ghose, S., Das, A., Roy, P.P., Pal, U.: A deep one-shot network for query-based logo retrieval. Pattern Recogn. **96**, 106965 (2019). https://doi.org/10.1016/j.patcog.2019.106965
4. Chen, J., Gao, B.B., Lu, Z., Xue, J.H., Wang, C., Liao, Q.: APANet: adaptive prototypes alignment network for few-shot semantic segmentation. IEEE Trans. Multimedia **25**, 1–1 (2022). https://doi.org/10.1109/TMM.2022.3174405
5. Chen, L., Bentley, P., Mori, K., Misawa, K., Fujiwara, M., Rueckert, D.: Self-supervised learning for medical image analysis using image context restoration. Med. Image Anal. **58**, 101539 (2019). https://doi.org/10.1016/j.media.2019.101539
6. Ding, H., Sun, C., Tang, H., Cai, D., Yan, Y.: Few-shot medical image segmentation with cycle-resemblance attention. In: 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 2487–2496. IEEE Computer Society, Los Alamitos, CA, USA (Jan 2023).https://doi.org/10.1109/WACV56688.2023.00252, https://doi.ieeecomputersociety.org/10.1109/WACV56688.2023.00252
7. Dong, N., Xing, E.P.: Few-shot semantic segmentation with prototype learning. In: British Machine Vision Conference (2018)
8. Fan, Q., Pei, W., Tai, Y.W., Tang, C.K.: Self-support few-shot semantic segmentation. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision - ECCV 2022, pp. 701–719. Springer Nature Switzerland, Cham (2022)

9. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. Int. J. Comput. Vis. **59**(2), 167–181 (2004)
10. Gansbeke, W.V., Vandenhende, S., Georgoulis, S., Gool, L.V.: Unsupervised semantic segmentation by contrasting object mask proposals. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 10032–10042. IEEE Computer Society, Los Alamitos, CA, USA (oct 2021). https://doi.org/10.1109/ICCV48922.2021.00990, https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.00990
11. Gao, Z., et al.: Unsupervised representation learning for tissue segmentation in histopathological images: from global to local contrast. IEEE Trans. Medical Imaging **41**(12), 3611–3623 (2022). https://doi.org/10.1109/TMI.2022.3191398
12. Guha Roy, A., Siddiqui, S., Pölsterl, S., Navab, N., Wachinger, C.: 'squeeze & excite' guided few-shot segmentation of volumetric images. Med. Image Anal. **59**, 101587 (2020). https://doi.org/10.1016/j.media.2019.101587
13. Guizilini, V., Ramos, F.: Online self-supervised segmentation of dynamic objects. In: 2013 IEEE International Conference on Robotics and Automation, pp. 4720–4727 (2013). https://doi.org/10.1109/ICRA.2013.6631249
14. He, H., Zhang, J., Thuraisingham, B., Tao, D.: Progressive one-shot human parsing. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Event, February 2-9, 2021, pp. 1522–1530. AAAI Press (2021). https://ojs.aaai.org/index.php/AAAI/article/view/16243
15. Hoyer, L., Dai, D., Chen, Y., Koring, A., Saha, S., Gool, L.V.: Three ways to improve semantic segmentation with self-supervised depth estimation. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11125–11135. IEEE Computer Society, Los Alamitos, CA, USA (Jun 2021). https://doi.org/10.1109/CVPR46437.2021.01098, https://doi.ieeecomputersociety.org/10.1109/CVPR46437.2021.01098
16. Ji, X., Vedaldi, A., Henriques, J.: Invariant information clustering for unsupervised image classification and segmentation. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 9864–9873. IEEE Computer Society, Los Alamitos, CA, USA (Nov 2019). https://doi.org/10.1109/ICCV.2019.00996, https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00996
17. Kavur, A.E., et al.: CHAOS challenge - combined (CT-MR) healthy abdominal organ segmentation. Med. Image Anal. **69**, 101950 (2021). https://doi.org/10.1016/j.media.2020.101950
18. Li, G., Jampani, V., Sevilla-Lara, L., Sun, D., Kim, J., Kim, J.: Adaptive prototype learning and allocation for few-shot segmentation. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8330–8339. IEEE Computer Society, Los Alamitos, CA, USA (Jun 2021). https://doi.org/10.1109/CVPR46437.2021.00823, https://doi.ieeecomputersociety.org/10.1109/CVPR46437.2021.00823
19. Liu, J., Bao, Y., Xie, G., Xiong, H., Sonke, J., Gavves, E.: Dynamic prototype convolution network for few-shot semantic segmentation. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11543–11552. IEEE Computer Society, Los Alamitos, CA, USA (Jun 2022). https://doi.org/10.1109/CVPR52688.2022.01126, https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.01126
20. Liu, W., Zhang, C., Ding, H., Hung, T.Y., Lin, G.: Few-shot segmentation with optimal transport matching and message flow. IEEE Trans. Multimedia 1–12 (2022). https://doi.org/10.1109/TMM.2022.3187855
21. Liu, Y., Zhang, X., Zhang, S., He, X.: Part-aware prototype network for few-shot semantic segmentation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12354, pp. 142–158. Springer, Cham (2020)

22. Okazawa, A.: Interclass prototype relation for few-shot segmentation. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision - ECCV 2022, pp. 362–378. Springer Nature Switzerland, Cham (2022)
23. Ouali, Y., Hudelot, C., Tami, M.: Autoregressive unsupervised image segmentation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12352, pp. 142–158. Springer, Cham (2020)
24. Ouyang, C., Biffi, C., Chen, C., Kart, T., Qiu, H., Rueckert, D.: Self-supervision with Superpixels: training few-shot medical image segmentation without annotation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12374, pp. 762–780. Springer, Cham (2020)
25. Ouyang, C., Biffi, C., Chen, C., Kart, T., Qiu, H., Rueckert, D.: Self-supervised learning for few-shot medical image segmentation. IEEE Trans. Med. Imaging **41**(7), 1837–1848 (2022). https://doi.org/10.1109/TMI.2022.3150682
26. Rakelly, K., Shelhamer, E., Darrell, T., Efros, A.A., Levine, S.: Conditional networks for few-shot semantic segmentation. In: International Conference on Learning Representations (2018)
27. Rakelly, K., Shelhamer, E., Darrell, T., Efros, A.A., Levine, S.: Few-shot segmentation propagation with guided networks. arxiv preprint arxiv:abs/1806.07373 (2018)
28. Shaban, A., Bansal, S., Liu, Z., Essa, I., Boots, B.: One-shot learning for semantic segmentation. In: British Machine Vision Conference 2017, BMVC 2017, London, UK, September 4-7, 2017. BMVA Press (2017)
29. Siam, M., Oreshkin, B., Jagersand, M.: AMP: adaptive masked proxies for few-shot segmentation. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 5248–5257. IEEE Computer Society, Los Alamitos, CA, USA (Nov 2019). https://doi.org/10.1109/ICCV.2019.00535, https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00535
30. Siam, M., Oreshkin, B.N.: Adaptive masked weight imprinting for few-shot segmentation. In: Workshop at the International Conference on Learning Representations (ICLR) (2019)
31. Singh, S., et al.: Self-supervised feature learning for semantic segmentation of overhead imagery. In: British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018. pp. 102. BMVA Press (2018). http://bmvc2018.org/contents/papers/0345.pdf
32. Wang, H., Zhang, X., Hu, Y., Yang, Y., Cao, X., Zhen, X.: Few-shot semantic segmentation with democratic attention networks. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12358, pp. 730–746. Springer, Cham (2020)
33. Wang, K., Liew, J.H., Zou, Y., Zhou, D., Feng, J.: PANet: few-shot image semantic segmentation with prototype alignment. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 9196–9205 (2019). https://doi.org/10.1109/ICCV.2019.00929
34. Wu, H., Xiao, F., Liang, C.: Dual contrastive learning with anatomical auxiliary supervision for few-shot medical image segmentation. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision - ECCV 2022, pp. 417–434. Springer Nature Switzerland, Cham (2022)
35. Yang, B., Liu, C., Li, B., Jiao, J., Ye, Q.: Prototype mixture models for few-shot semantic segmentation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12353, pp. 763–778. Springer, Cham (2020)
36. Zhang, B., Xiao, J., Qin, T.: Self-guided and cross-guided learning for few-shot segmentation. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8308–8317. IEEE Computer Society, Los Alamitos, CA, USA (Jun 2021). https://doi.org/10.1109/CVPR46437.2021.00821, https://doi.ieeecomputersociety.org/10.1109/CVPR46437.2021.00821

37. Zhang, K., Zheng, Y., Deng, X., Jia, W., Lian, J., Chen, X.: Guided networks for few-shot image segmentation and fully connected CRFs. Electronics **9**(9), 1508 (2020). https://doi.org/10.3390/electronics9091508, https://www.mdpi.com/2079-9292/9/9/1508
38. Zhang, X., Wei, Y., Yang, Y., Huang, T.S.: SG-One: similarity guidance network for one-shot semantic segmentation. IEEE Trans. Cybern. **50**(9), 3855–3865 (2020). https://doi.org/10.1109/TCYB.2020.2992433
39. Zhu, K., Zhai, W., Zha, Z., Cao, Y.: Self-supervised tuning for few-shot segmentation. In: International Joint Conference on Artificial Intelligence (2020). https://api.semanticscholar.org/CorpusID:215744862

Quantitative Structure-Activity Relationship Modeling for the Prediction of Fish Toxicity Lethal Concentration on Fathead Minnow

R. Kavitha[1](✉) and D. S. Guru[2]

[1] Department of Data Science, CHRIST (Deemed to Be University), Bangalore 560 029, Karnataka, India
kavitha.rajamanii@gmail.com

[2] Department of Studies in Computer Science, University of Mysore, Mysuru 570 006, Karnataka, India
dsg@compsci.uni-mysore.ac.in

Abstract. As there has been a rise in the usage of in silico approaches, for assessing the risks of harmful chemicals upon animals, more researchers focus on the utilization of Quantitative Structure Activity Relationship models. A number of machine learning algorithms link molecular descriptors that can infer chemical structural properties associated with their corresponding biological activity. Efficient and comprehensive computational methods which can process huge set of heterogeneous chemical datasets are in demand. In this context, this study establishes the usage of various machine learning algorithms in predicting the acute aquatic toxicity of diverse chemicals on Fathead Minnow (Pimephales promelas). Sample drive approach is employed on the train set for binning the data so that they can be located in a domain space having more similar chemicals, instead of using the dataset that covers a wide range of chemicals at the entirety. Here, bin wise best learning model and subset of features that are minimally required for the classification are found for further ease. Several regression methods are employed to find the estimation of toxicity LC50 value by adopting several statistical measures and hence bin wise strategies are determined. Through experimentation, it is evident that the proposed model surpasses the other existing models by providing an R^2 of 0.8473 with RMSE 0.3035 which is comparable. Hence, the proposed model is competent for estimating the toxicity in new and unseen chemical.

Keywords: QSAR · Aquatic Toxicity · Lethal Concentrations · LC50 · Fathead Minnow · Pimephales promelas

1 Introduction

REACH (Registration, Evaluation, Authorization and restriction of CHemicals) a regulation of European Union aims to improve the conservation of living organism's health and environment from risks that may be posed by chemicals. To conform with the directives, the chemical companies which involve in drug design, production of consumer goods products and industrial processes must determine and manage or communicate the risk. The toxic concentration of a compound is ascertained by experiments using in vivo or in vitro models [1] measuring the end points. On the other hand, these techniques pose some issues as they are very much time consuming, expensive and demands the living organisms to put to the test. To overcome these challenges, in silico methods [1] have been gaining attention much due to their low cost, time efficiency and particularly without harming any living organism. Consequently, in silico methods flourished as an alternative method for the risk full assessment of chemicals by not posing any adverse effects on animals [1]. Hence, mathematical models that can be used to predict the physicochemical, biological and environmental properties of compounds from the knowledge of their chemical structure has drawn surpassing considerations in the present years.

Quantitative Structure-Activity Relationship (QSAR) modeling is an empirical and statistical approach by which chemical structure is quantitatively correlated with biological activity. The biological activity can be conveyed quantitatively as the concentration of a chemical substance needed to stimulate a certain biological reaction. The modeling can be used to predict toxicity of a chemical compound from the physical characteristics of the structure of chemicals called molecular descriptors. This modeling has several well known applications in estimation of toxicity of chemical compounds, drug discovery, interactions between structural domains of proteins, etc., [2]. Fish Acute Toxicity Syndrome is the functional reactions in fish as a consequence of critical exposure to a lethal concentration of a toxic chemical compound which can produce an unfavorable effect in it. A number of QSAR models have been developed to predict the acute toxicity towards various aquatic organisms by estimating the quantitative parameter usually LC50, which represents the concentration of a chemical compound required to kill 50% of test population in 96 h of administration.

Most of the studies employed similarity-based methods which could not handle huge set of features properly. Also, in consensus modeling, same set of features were used for the entire model. In addition to these, the dataset is varied in nature having diverse chemicals. So, modeling techniques that can better model the portion of data rather than the whole dataset is in need. Subsets of data may exhibit variant properties which might be learned better by different model. This motivated us to drive focus with different viewpoint. In this paper, an approach is proposed where pseudo bins are generated based on the closeness of the target parameter and then finally estimation of LC50 is computed.

The ultimate aim of this study is to estimate the acute toxicity through molecular descriptors in terms of LC50, a lethal concentration of a chemical compound required to kill 50% of test population of Fathead minnow in 96 h of administration. The aim of this study are stated as follows: (i) To develop an effective way of binning the dataset with various levels using bin dependent best learning model and minimal subset of feature selection. (ii) To locate the unseen chemical in a near optimal bin having subset of samples. (iii) To predict LC50 based on sample driven regression model. (iv) Unlike other learning-based models which use a common learning model across all samples of the population; here we recommend adapting best performing learning models for each level and for each bin while predicting LC50 value.

The organization of this paper is as follows. Section 2 reviews the literature on QSAR modeling about aquatic toxicity. In Sect. 3, the proposed methodology is explained. Further, the analysis of model performance is done in Sect. 4. The final section concludes the paper.

2 Literature Review

Most in silico models are trained on extensive, diverse datasets that are gathered from multiple sources and combined into a single database [5].

The QSAR Toolbox is developed by the Organization for Economic Co-operation and Development to aid in risky assessment of chemicals. It contains 53 databases having nearly 100, 000 chemicals with above 3 million measured and recorded data in version 4.5 under four categories as physical chemical properties, environmental fate and transport, ecotoxicological information and human health hazards. The prevalent databases used for experimentations are ECOTOX, ECHA REACH and Aquatic OASIS. The CompTox Chemicals Dashboard is a part of a set of databases and applications developed by Chemical Safety for Sustainability Research Program of US Environmental Protection Agency. These databases aid research efforts in computational toxicology to develop methods to modify how chemicals are currently assessed for potential health risks. The dashboard has chemical toxicity and exposure information for over 900,000 chemicals. ToxValDB is a comprehensive collection of quantitative data from public datasets i.e., 34 unique sources including EPA (risk assessment), ECHA (industrial chemicals, human and eco), EFSA (food additives, human and eco), DOE (Ecotoxicology risk assessments) [15]. The database contains 700,000 records for over 25,000 chemicals. The data is being made public through EPA CompTox Chemicals Dashboard.

TopTox is a software for computing element-specific topological descriptors (ESTDs) for toxicity endpoint predictions and it is available online through Wei Lab, Michigan State University, USA. The preprocessed train and test sets from TopTox were used; where original data is available in ECOTOX, http://chem.sis.nlm.nih.gov/chemidplus/chemidheavy.jsp [6]. ChemIDPlus from National Library of Medicine (NLM), US is an online dictionary of chemicals and its structures. The dataset has end points for four different quantitative toxicity. They are LD50, IGC50, LC50 and LC50-DM. These endpoint measures have been used in toxicology for estimating the toxicity of a given chemical compound on a given population of a given organism [6]. Our study uses the LC50 dataset for the experimentation purpose. LC50 measures the toxicity concentration of a given chemical compound to kill 50% of the test population of Fathead minnow, a species of temperate freshwater fish after 96 h of exposure. The unit of measurement of LC50 is −log10 (T mol/L). The dataset comprises of total 823 chemicals with its 995 descriptors.

Many works in the literature attempted in estimating the LC50 value through similarity-based k-Nearest Neighbor (KNN) regression and linear regression. KNN regression was applied for LC50 estimation and Genetic algorithms were used to select molecular descriptors and to optimize the number of nearest neighbors [1]. Multiple linear regression and artificial neural network were used to predict LC50 [2, 3]. Recursive partitioning was implemented for partitioning on mode of actions as reactive or narcosis before building a regressor [2]. An ensemble stacking QSAR modeling was built to predict both LC50 and points of departure [4]. Two variants of similarity-based algorithms were used in [5] for the prediction of acute aquatic toxicity as distance weighted KNN and Locally weighted Least Squares Kernel regression and it was found that kernel-based regression performs well. In [6], meta ensemble method was proposed by providing five different feature representation and finally the aggregated estimation was retrieved from a deep learning model. A new strategy is introduced in Artificial Neural Network (ANN) for QSAR modeling where residuals of Linear Regression along with molecular descriptors were given as input to the neural network [3]. Both bagging and boosting types of ensemble learning was adopted to predict toxicity on multiple aquatic organisms [7]. In [8], Regression analysis had been applied to examine the structure-activity relationships regarding acute fish toxicity. Here, logP dependent baseline toxicity model was developed. A hybrid quantum particle swarm optimization was used to optimize the key molecular descriptors as well as the parameters of Radial Basis Function and then it was used to estimate the acute toxicity [9].

The chemical compounds were classified based on different narcosis mode of action using Bayesian approach and then regression was applied for the estimation [11]. To overcome the biased assessment of predictor performance, two different strategies of cross validation were proposed as ‘transformation-out’ and ‘solvent-out’ based on reactions going under new conditions [13]. Logistic Regression, Multi-layer perceptron, Random Forest classifiers were employed to analyze various end points having imbalancedness [14]. A variant of 21 types of errors which frequently occurs in QSAR literature were found. Recommendations were suggested for the avoidance of such errors [12]. k-Nearest Neighbor method was used with Mahalanobis distance to measure the similarity between the specific molecule and its neighbors for prediction of toxicity of Daphina Magna. Genetic Algorithm was coupled to select the relevant molecular descriptors [16]. The ability of three global models such as response surface model, stepwise linear regression and neural network to model the toxicity of phenols was investigated and found that linear regression using mechanistically interpretable descriptors perform well as neural network analysis [17]. The tree ensemble Random Forest models for prediction of acute toxicities of three trophic level organisms R. subcapitata, D. Magna and fish were carried out. Genetic algorithm was used for pruning the descriptors [18]. A global prediction model for mode of action and two local models for organic compounds that exhibit narcosis and excess toxicity were developed using genetic algorithm and multiple linear regression [19].

3 Proposed Methodology

The proposed approach for LC50 estimation through molecular descriptors is discussed in this section.

A new approach is proposed in this study for estimating the lethal concentration of aquatic toxicity. The architectural diagram of the proposed model is given in Fig. 1. Data preparation is performed to make the data suitable for the analysis. The observations with invalid non-numeric data, features with constant values are discarded. The dataset is sorted in ascending order based on the target LC50 value. Initially same bin value is assigned to all samples. The entire dataset is made into several initial level bins by fixing a threshold value on first order differences of LC50 (FOD) as in Eq. 1. A new bin is created only if the first order difference between the current and previous sample exceeds the threshold value.

$$Bin(i) = \begin{cases} (Bin(i-1)+1 & LC50(i) - LC50(i-1) > Th1 \\ (Bin(i)) & Otherwise \end{cases} \quad for\, i = 1, 2, \ldots, n \quad (1)$$

where, n is the no. of samples, Bin(i) is the Bin of ith sample and LC50(i) is the Lethal Concentration of ith sample.

Then, successive levels of binning are done when the Weighted Mean Square Difference (WMSD) of a specific bin is greater than a specified threshold which is expressed in Eq. 2. Thus, the respective bins which are to be partitioned into sub bins are identified. Once it is identified, the bins are partitioned into sub bins by fixing a threshold value on first order differences of LC50 of samples of that respective bin.

$$\begin{aligned} \text{WMSD} = {} & \frac{w_i(maxLC50_i - minLC50_i)^2}{\sum W} \\ & > Th_j\, for\, i = 1\, to\, No.\, of\ samples\, in\, respective\, bin \end{aligned} \quad (2)$$

where, Th_j is the threshold at j^{th} level,

w_i is the number of samples in the Bin B_i.

w is the total number of samples.

The above strategy is repeated till a bin violates the condition in Eq. 2.

Various classifiers are considered and trained on the level wise samples based on the bins and the best learning model is found for the recommendation purpose. The features are ranked using various supervised feature selection methods and the scores are recorded. The minimal subset of ranked features required to classify the bins into sub partitions is observed by utilizing the best model identified. Bin wise best learning model and subset of required feature are identified for all levels by applying k-fold cross validation. It is stored for further recommendation purpose.

Once the binning is done, the regression is applied at the leaf nodes so that the final toxicity level LC50 can be estimated. Three different strategies are applied here based on the number of samples present in the specific leaf bins. If there is a single sample in the leaf bin, then there is a direct assignment of LC50 value. If there are two samples present in the leaf bin, then the average of both the LC50 values are taken as predicted LC50. If there are more than two samples in the leaf bin, a different strategy is adopted. Variants of regression models are considered and trained for the bins individually. Then the predicted LC50 values out of the regression models are subjected to further statistical analysis based on different statistical measures. By the way, the finally computed LC50 of each bin is recorded to assign for unseen record which falls into the respective bin in future testing.

The statistical measures utilized in this experimentation are mean, mean of first order difference over cut points, midrange, inter quartile mean and mid hinge. The way these statistical measures computed on predicted values of LC50 are given below from Eq. 3 to Eq. 7.

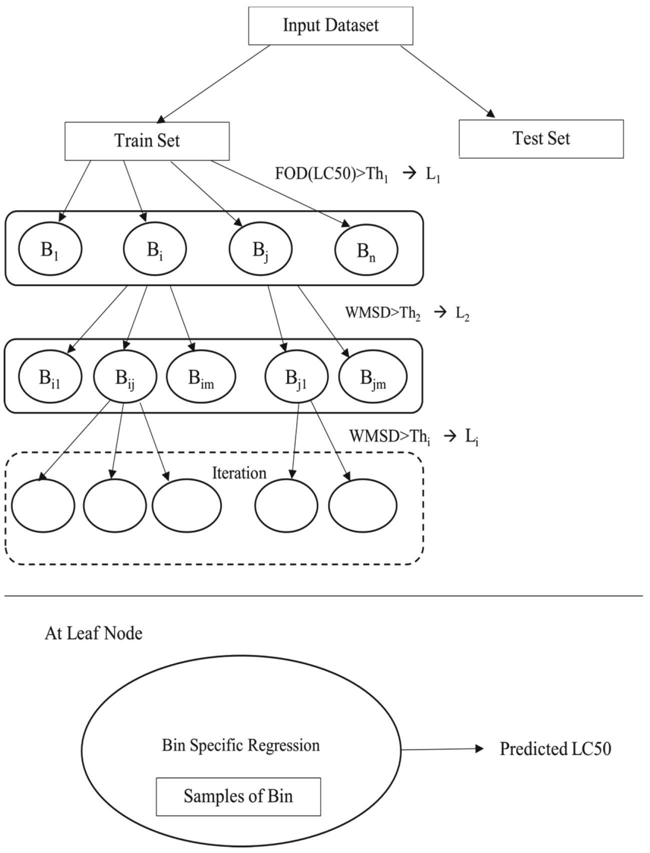


Fig. 1. Schematic Representation of the Proposed Model

Measure 1: Mean

$$\sum_{i=1}^{m} pred\_LC50(i)/m \tag{3}$$

where 'm' is the no. of regression models

Measure 2: Mean of First Order Difference Over Cut Points

$$\forall_i FOD_i = pred\_LC50(i+1) - pred\_LC50(i),\, i = 1, .., m$$

$$\frac{\sum_{i=j}^{k} pred\_LC50(i)}{k}, i = pred\_LC50\big[j\big] to\, pred\_LC50[k] \tag{4}$$

where, $j = \max(FOD)\, and\, k = SecondMax\,(FOD)$

Measure 3: Mid Range

$$\big[\max(pred\_LC50) - \min(pred\_LC50)\big]/2 \tag{5}$$

Measure 4: Inter Quartile Mean

$$\frac{\sum_{i=j}^{k} pred\_LC50(i)}{k}, i = pred\_LC50[j]\, to\, pred\_LC50[k] \quad (6)$$

where, $j = 25\text{th percentile}(Pred\_LC50)\ and\ k = 75th\,percentile(pred\_LC50)$

Measure 5: Mid Hinge

$$[25\text{th percentile}(pred\_LC50) - 75th\,percentile(pred\_LC50)]/2 \quad (7)$$

Thus, the proposed method helps in locating the test data in the near optimal bin. Therefore, the respective attached LC50 value of the bin is assigned for the new chemical.

4 Experimental Results

4.1 About Dataset

As it is mentioned earlier, the compiled dataset is taken from the study [4]. There are totally 995 molecular descriptors as predictor features characterizing a chemical compound. The target feature LC50 records the toxicity of a given compound causing death in 50% of test Fathead minnow, a species of temperate freshwater fish within 96 h of exposure. The initial size of the dataset is 823×995. In some samples, certain features contain invalid characters. Specifically, some continuous-type features are recorded with the text value. After the removal of those invalid data among the predictors, the original dataset dimension is reduced to 642×995. Then variance is found to eliminate the features with constant values by fixing variance threshold as 0. The number of features removed in this case is 153. The remaining dataset is taken for the further processing.

The acute toxicity of the data ranges from 0.15 to 9.261 mmol/L. The data distribution significantly impacts the statistical quality of any predictive model. Therefore, a histogram is generated to check if the endpoints contain poorly distributed or empty regions. The histogram developed indicates normality with a mean of 4.3605, standard deviation (SD) of 1.3987 respectively. The median is found to be 4.2555 and as the mean and median are closer to each other, the dataset is more or less evenly distributed from the lowest to highest values of LC50 preserving the normality. It is expressed in Fig. 2. The x-axis represents the quantity of lethal concentration LC 50 in mmol/L. The y-axis represents the normal distribution as density of probability.

4.2 Model Performance

The train and test set are split in the ratio of 80:20 for model development and validation respectively. The entire train set is initially partitioned into several bins based on the Eq. 1 by fixing a FOD threshold Th1 on subsequent sample target value difference as 0.05. Then, the bins are checked for possibility of dividing it into further and subsequent binning happened in the next levels. The subsequent binning is done using Eq. 2. The various thresholds fixed such as WMSD and FOD on different levels upon experimentation is given in Table 1.

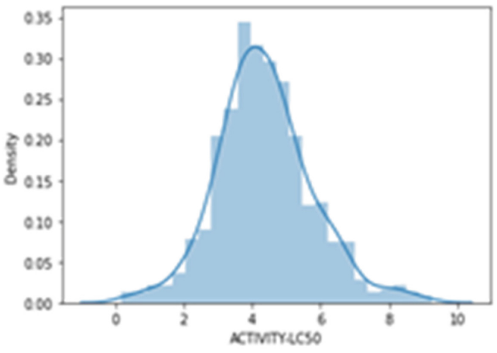


Fig. 2. Normal Distribution of Target Variable Lethal Concentration LC50

Table 1. Threshold values for FOD and WMSD

Level	FOD Threshold	WMSD Threshold
Level 1	0.05	–
Level 2	0.03	0.0001
Level 3	0.01	0.001

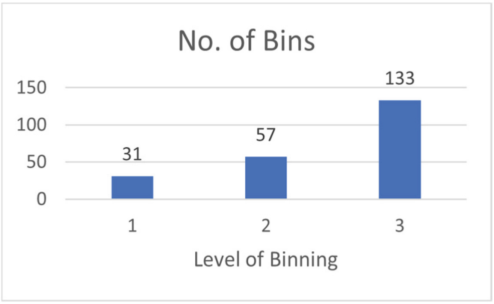


Fig. 3. Level wise number of bins

The following bar chart in Fig. 3 shows the number of bins generated in each level.

For each level of binning, 3 different classifiers are learnt such as K-Nearest Neighbor Classifier (KNN), Support Vector Machine Classifier (SVM) and Random Forest Classifier (RF). Thus, the bin wise best learning model is identified as the nature of the samples taking part in the learning is different as per the bins. Similarly, for each bin, a set of feature selection technique is applied and a best feature selection technique is identified based on the F1-score and the subset of features for the testing purpose. The features with rank 0 are discarded and other features are taken into account and a model is trained based on it to find the optimal subset of features. In the first level binning, a set of 23 descriptors are selected. Then in the subsequent levels, for each large bin which are partitioned into sub bins, best learning model and subset of important features are found. It is represented in Table 3. But the performance analysis was done through experimentation on all classifiers and feature selection techniques that were taken for the analysis. The overall performance is given in the below Table 2.

Table 2. Performance of Classifiers with respect to bins

Level	Bin Number	Learning Model/Feature Selection Technique	Chi2	Mutual Info Classif	f-Classif
1	Overall	KNN-C	0.801	0.8069	0.7815
		SVC	0.8127	0.8127	**0.8147**
		RFC	0.7952	0.801	0.8127
2	14	KNN-C	0.865	0.8606	0.8671
		SVC	**0.884**	0.8798	0.882
		RFC	0.8735	0.8692	0.8735
	15	KNN-C	0.8146	0.8071	0.7941
		SVC	**0.886**	0.8606	0.87
		RFC	0.8602	0.85	0.8572
	16	KNN-C	0.8159	0.8206	0.8241
		SVC	0.8712	0.8708	0.8803
		RFC	0.88	0.8727	**0.882**
3	15:06	KNN-C	0.8055	0.7973	0.7945
		SVC	**0.8274**	0.8203	0.819
		RFC	0.8219	0.8192	0.8192
	15:07	KNN-C	0.8278	0.83	0.8174
		SVC	0.8883	**0.8917**	0.8862
		RFC	0.8652	0.8719	0.8613
	15:08	KNN-C	0.8206	0.8303	0.8194
		SVC	0.8471	**0.85**	0.8392
		RFC	0.8197	0.8374	0.8063

Four different ways of performance evaluation has been done. Each experiment is conducted with 15 trials, and the mean value is calculated. The training dataset undergoes 5-fold cross-validation. Additionally, a 15 $\times$ 2 cross-validation approach is employed, consisting of 15 repeats of 5-fold cross-validation.

Ground Truth Associated with Bins

Each bin has non-overlapping range values. Each chemical in the test set is passed over the proposed tree generated and reaches the leaf bin. The correctness of the bin

Table 3. Bin wise optimal number of features with learning model and feature selection technique

Level	Bin Number	Learning Model	Feature Selection Technique	No. of features required for binning
1	Overall	SVC	f_classif	23
2	14	SVC	Chi2	5
	15	SVC	Chi2	3
	16	RF	f_classif	12
3	15–6	SVC	Chi2	89
	15–7	SVC	mutual_info_classif	56
	15–8	SVC	mutual_info_classif	6

assignment is checked using Eq. 8. Then the error rate and accuracy of the bin assignment are calculated accordingly.

$$Bin(Chemical) = \begin{matrix} (Correct) \;\; if\; LC50(Chemical) > Min\,\mathrm{LC50}(Bin)\; and \; \geq Max\,\mathrm{LC50}(Bin) \\ (Incorrect) \qquad\qquad\qquad Otherwise \end{matrix} \tag{8}$$

The mean error rate and mean accuracy are found to be 9.41 and 90.59 respectively.

Performance Through Regression

The test set is taken sample by sample and admitted to the testing process. Hence, the exact bin for the test sample is found based on the bin wise best learning model by using only the recommended subset of features. Once it reaches the leaf bin, the attached LC50 value is assigned for that chemical.

In Table 4, we present the statistics that represents the performance of the proposed model. Q^2 for internal validation, Coefficient of determination (R^2) for external validation, Root Mean Square Error (RMSE), Mean Absolute Error (MAE) and Ratio of Performance to Deviation (RPD) are computed by comparing experimental value and predicted value. Internal cross validation refers to the 5-fold cross validation done using training dataset and external validation refers to the 20% of data that is kept aside as test set.

After binning, regressors are applied individually for the purpose of comparison of performance with the proposed model without computing statistical measures for estimation. The R^2 score obtained are; 0.7752, 0.7986 and 0.7418 for KNN Regression, Support Vector Regression and Random Forest Regression respectively. But the proposed model provides R^2 score of 0.8305 which is comparatively higher than applying the individual regressors. Hence, it is found that, sample driven bin wise application of regression along with statistical computation performs well in estimating the final LC50 value.

Table 4. Performance Evaluation of the proposed model

Model Evaluation Method	Proposed Model	Existing Study [6]
Q^2 for Internal Cross Validation	0.8953	–
R^2 for External Validation	0.8305	0.792
RMSE	0.3044	0.668
MAE	0.2965	0.479
RPD	6.9546	–

Evaluation Through Range Interval

Once the sample in the test set reaches the respective leaf bin, the closest range of LC50 (either minimum or maximum) value of that specific bin is assigned as the final LC50. Then, the coefficient of determination (R^2) is calculated. The mean R^2 and mean RMSE are 0.8473 and 0.3035 respectively.

Evaluation Through Regression Within Bin

Linear regression is performed in each leaf bin and the model is stored for the testing purpose. Once the sample arrives at the leaf bin, using that stored regression model, the LC50 value is estimated. The mean R^2 and RMSE values are 0.8014 and 0.4158 respectively.

The R^2 and RMSE metrics, obtained from an independent test set sourced from existing research [6], exhibit values of 0.792 and 0.668 respectively. Through experimentation in this study, the R^2 and RMSE values are 0.8473 and 0.3035 respectively which are comparable.

From the experimentation, it is inferred that, bin wise learning model and subset of features are very much useful as they are sample driven and hence contributes in improving the overall accuracy.

5 Conclusion

The aim of this research is to estimate the aquatic toxicity value, LC50, for Fathead Minnow using machine learning models, leveraging the molecular descriptors of chemical compounds. Bin wise best learning model and minimal subset of features needed for further classification into bins are identified. The main advantage of this approach is instead of using the entire dataset for the estimation spanning a broad range of chemicals, it uses only the data most similar to the given fitting point. Then, the regression models which best suits for individual bins are identified and through various statistical measures, the final estimation of LC50 is done. The proposed model outperforms the application of individual machine learning model for this QSAR study. Also, the performance of the proposed model is compared with the existing study and it is evident that our model performs efficiently in terms of evaluation metrics R^2 0.8473. The findings from this study provide further evidence of the benefits of using statistical measures on the aggregated output of the regressors. Hence, this study precisely characterizes the relationship between the response and predictor variables in heterogeneous aquatic toxicology data. Study on association amongst molecular descriptors and finding out clustering of descriptors for effective way of reducing features could be our future interesting work.

References

1. Cassotti, M., Ballabio, D., Todeschini, R., Consonni, V.: A similarity-based QSAR model for predicting acute toxicity towards the fathead minnow (Pimephalespromelas). SAR QSAR Environ. Res. **26**(3), 217–243 (2015)
2. In, Y., Lee, S.K., Kim, P.J., No, K.T.: Prediction of acute toxicity to fathead minnow by local model based QSAR and global QSAR approaches. Bull. Korean Chem. Soc. **33**(2), 613–619 (2012)
3. Devillers, J.: A new strategy for using supervised artificial neural networks in QSAR. SAR QSAR Environ. Res. **16**(5), 433–442 (2005)
4. Sheffield, T.Y., Judson, R.S.: Ensemble QSAR modeling to predict multispecies fish toxicity lethal concentrations and points of departure. Environ. Sci. Technol. **53**, 12793−12802 (2019)
5. Gajewicz-Skretna, A., Furuhama, A., Yamamoto, H., Suzuki, N.: Generating accurate in silico predictions of acute aquatic toxicity for a range of organic chemicals: towards similarity-based machine learning methods. Chemosphere **280**, 130681 (2021)
6. Karim, A., et al.: Quantitative toxicity prediction via meta ensembling of multitask deep learning models. ACS Omega **6**, 12306−12317 (2021)
7. Singh, K.P., Gupta, S., Kumar, A., Mohan, D.: Multispecies QSAR modeling for predicting the aquatic toxicity of diverse organic chemicals for regulatory toxicology. Chem. Res. Toxicol. **27**, 741−753 (2014)
8. Nendzat, M., Russomi, C.L.: QSAR modelling of the ERL-D fathead minnow acute toxicity database. Xenobiotica **27**(2), 147–170 (1991)
9. Wang, Y., Chen, X.: A joint optimization QSAR model of fathead minnow acute toxicity based on a radial basis function neural network and its consensus modeling. RSC Adv. **10**, 21292 (2020)
10. Liu, H., Setiono, R.: Chi2: feature selection and discretization of numeric attributes. In: 7th IEEE International Conference Proceedings on Tools with Artificial Intelligence, pp. 388–391 (1995)
11. Lozano, S., Lescot, E., Halm, M.-P., Lepailleur, A., Bureau, R., Rault, S.: Prediction of acute toxicity in fish by using QSAR methods and chemical modes of action. J. Enzyme Inhibit. Med. Chem. **25**(2), 195–203 (2010)
12. Dearden, J.C., Cronin, M.T.D., Kaiser, K.L.E.: How not to develop a quantitative structure–activity or structure–property relationship (QSAR/QSPR). SAR QSAR Environ. Res. **20**(3–4), 241–266 (2009)
13. Rakhimbekova, A., et al.: Cross-validation strategies in QSPR modelling of chemical reactions. SAR QSAR Environ. Res. **32**(3), 207–219 (2021)
14. Lovrić, M., Malev, O., Klobučar, G., Kern, R., Liu, J.J., Lučić, B.: Predictive capability of QSAR models based on the CompTox zebrafish embryo assays: an imbalanced classification problem. Molecules **26**, 1617 (2021)
15. Judson, R.: ToxValDB: Compiling Publicly Available In Vivo Toxicity Data. Presented at EPA's Computational Toxicology Communities of Practice Monthly Meeting, RTP, NC, (2018)

16. Cassotti, M., Ballabio, D., Consonni, V., Mauri, A., Tetko, I.V., Todeschini, R.: Prediction of acute aquatic toxicity towards Daphnia magna by using the GA-kNN method. ATLA-Alternatives to Laboratory Animals **42**, 31–41 (2014)
17. Enoch, S.J., Cronin, M.T.D., Schultz, T.W., Madden, J.C.: An evaluation of global QSAR models for the prediction of the toxicity of phenols to Tetrahymena pyriformis. Chemosphere **71**, 1225–1232 (2008)
18. Toma, C., Cappelli, C.I., Manganaro, A., Lombardo, A., Arning, J., Benfenati, E.: New models to predict the acute and chronic toxicities of representative species of the main trophic levels of aquatic environments. Molecules **26**, 6983 (2021)
19. Wu, X., Zhang, Q., Hu, J.: QSAR study of the acute toxicity to fathead minnow based on a large dataset. SAR QSAR Environ. Res. **27**(2), 147–164 (2016)

SuperSimpleNet: Unifying Unsupervised and Supervised Learning for Fast and Reliable Surface Defect Detection

Blaž Rolih(✉), Matic Fučka, and Danijel Skočaj

Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia
br9136@student.uni-lj.si, {matic.fucka,danijel.skocaj}@fri.uni-lj.si

Abstract. The aim of surface defect detection is to identify and localise abnormal regions on the surfaces of captured objects, a task that's increasingly demanded across various industries. Current approaches frequently fail to fulfil the extensive demands of these industries, which encompass high performance, consistency, and fast operation, along with the capacity to leverage the entirety of the available training data. Addressing these gaps, we introduce SuperSimpleNet, an innovative discriminative model that evolved from SimpleNet. This advanced model significantly enhances its predecessor's training consistency, inference time, as well as detection performance. SuperSimpleNet operates in an unsupervised manner using only normal training images but also benefits from labelled abnormal training images when they are available. SuperSimpleNet achieves state-of-the-art results in both the supervised and the unsupervised settings, as demonstrated by experiments across four challenging benchmark datasets. Code: https://github.com/blaz-r/SuperSimpleNet.

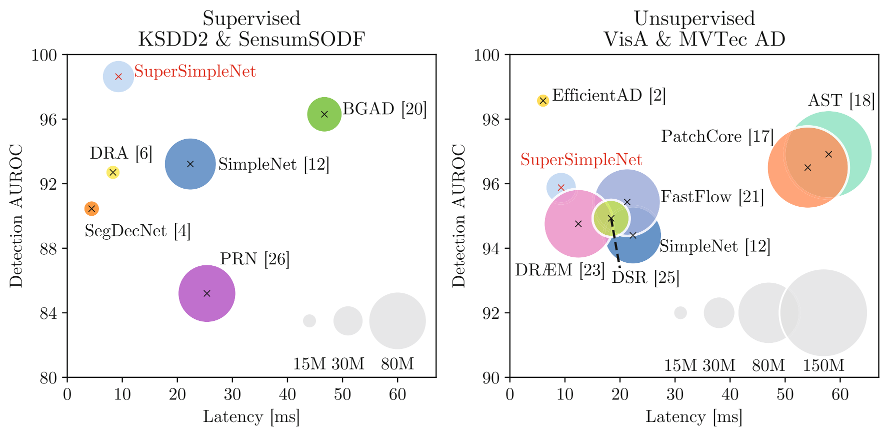
Keywords: Surface Defect Detection · Surface Anomaly Detection · Industrial Inspection · Supervised Learning · Unsupervised Learning

1 Introduction

A critical aim of the manufacturing process is to achieve high-quality products and increased efficiency. Surface Defect Detection (SDD) plays a pivotal role in this pursuit, as it aims to identify and classify defects or irregularities on the surface of manufactured components. Traditional manual inspection methods are time-consuming, subjective, and prone to human error. In contrast, automated SDD systems offer the potential for real-time monitoring, precise defect localisation, and improved product quality. The integration of deep learning algorithms into SDD systems [3,4,16,28] has shown promising results, indicating

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78192-6_4.

their potential to revolutionise quality control processes and streamline manufacturing operations.



Requirement	[18]	[21]	[17]	[23]	[26]	[16]	[10]	[14]	[6]	[4]	[2]	[20]	[25]	[12]	**Ours**
Unsupervised	✓	✓	✓	✓							✓	*	✓	✓	✓
Supervised					✓	✓	✓	✓	✓	✓		✓	*	*	✓
Speed						✓	-	-	✓	✓	✓				✓

Fig. 1. Model comparison for both the supervised (KSDD2 [4] and SensumSODF [16]) and the unsupervised (MVTec AD [3] and VisA [28]) setting. The Y-axis represents the anomaly detection performance measured in AUROC, and the X-axis represents inference time in milliseconds using an NVIDIA Tesla V100S (more details in Sect. 4.4). The size of the circles represents the model's parameter size. Additionally, the table below indicates whether each model meets specific speed requirements (if its inference time is below 10ms) and whether it is capable of working in the unsupervised and/or the supervised setting. If a model is designed specifically for either the supervised or the unsupervised setting but theoretically applicable to the other, we marked the opposing cell with a '*'. Two methods (marked with '-') lack publicly available code, preventing us from assessing their speed. SuperSimpleNet stands out as the only model meeting all criteria.

To bridge the gap between academic research and real-world manufacturing processes, developed models must meet all industry-defined requirements. These requirements fall into two main categories: performance and flexibility. Performance requirements pertain to the model's anomaly detection efficacy and its inference time. The model should exhibit a high anomaly detection rate alongside rapid inference capabilities. Although the performance aspects of these requirements have been extensively studied [2, 12], flexibility requirements have largely been overlooked. Flexibility requirements are concerned with the model's adaptability to various training regimes encountered in actual manufacturing settings. Different levels of annotations are available for various objects during training, requiring the model to be capable of using all available data effectively. This implies the necessity for a model to be trainable in both the supervised and the unsupervised setting, a feature seldom found in existing methods [20,25]. Another important yet often overlooked requirement is the stability of training, which should lead to consistent detection results, regardless of the specific training run. Unfortunately, a high level of consistency is not met by many existing methods. Our objective was to devise a method that fully meets these criteria: (i) detection and localisation performance, (ii) low inference time in both (iii) the supervised and the unsupervised setting, and (iv) a stable and consistent learning process.

To meet all specified requirements, we introduce SuperSimpleNet, an innovative model that builds on the foundation laid by SimpleNet [12]. While SimpleNet has demonstrated great performance in the unsupervised setting, achieving these results consistently requires multiple training runs. This limitation, coupled with the industry's demand for efficient and more resilient models, prompted us to refine the training approach and architecture of the original model. These enhancements have rendered it more robust and suitable for practical applications.

The main contributions of our work are as follows:

– We propose SuperSimpleNet - a strong discriminative defect detection model tailored to meet industry standards. We optimised the originally proposed architecture and introduced a novel synthetic anomaly generation procedure, resulting in a more stable learning process and improved performance. With an inference time of 9.3 ms and a throughput of 268 images per second, SuperSimpleNet outspeeds most contemporary models while achieving state-of-the-art defect detection results.
– We have extended the architecture, initially designed for the unsupervised setting, to incorporate abnormal training images and utilise available labels. Additionally, we've integrated a separate classification head into the model, which helps the model consider the image's global context. This unification of unsupervised and supervised approaches significantly boosts anomaly detection capabilities, positioning SuperSimpleNet among the top performers in both unsupervised and supervised scenarios. This versatility renders it highly suitable for industrial applications.

We have performed extensive experiments on four challenging datasets. First, we show that SuperSimpleNet achieves state-of-the-art results in the supervised setting on two standard defect detection datasets – SensumSODF and KSDD2, with an AUROC of 97.8% and a detection AP of 97.4%, respectively. Then, we show that SuperSimpleNet achieves state-of-the-art results in the unsupervised setting on two standard anomaly detection datasets – MVTec AD and VisA, with an AUROC of 98.4% and 93.4%, respectively. With the state-of-the-art results achieved in both scenarios, as depicted in Fig 1, we demonstrate the versatility of SuperSimpleNet and its suitability for real-life scenarios.

2 Related Work

Unsupervised methods
Unsupervised methods have become a cornerstone in surface anomaly detection because they can detect outliers in settings where labelled defect data is scarce or non-existent. To address such scenarios, various paradigms of models emerged. The *reconstructive approaches* made the first attempts, which train an autoencoder-like network [24] with the idea that the model will successfully reconstruct only the anomalous regions whilst leaving the normal regions intact. Methods under this paradigm also used different kinds of generative networks, such as GANs [1] or transformers [15]. The successful reconstruction of anomalous regions does not always hold, leading to an overall bad performance.

Another prominent paradigm involves *leveraging the features* extracted from a pretrained network, such as a ResNet [8]. The extracted features are then used to learn normality by utilising approaches such as a normalising flow [18,21], a memory-bank [17], a student-teacher architecture [27] or distillation [5].

The remaining set of approaches, the *discriminative methods*, are trained using synthetic anomalies. These anomalies can be generated on images [7,23,27], while the recent SimpleNet [12] generates them in latent space by perturbing the entire copy of features with noise. Another, more sophisticated approach to generating synthetic anomalies in the latent space is introduced in DSR [25], where a Perlin noise mask conditions the area of anomalies.

While these methods have demonstrated success, their complexity in design and implementation often hinders efficient execution, leading recent research efforts to also prioritise efficiency [2]. Even though these methods show great results in the unsupervised setting, most of them lag behind on datasets [4] curated for the supervised setting, as they cannot utilise labelled defects during training.

Supervised methods
Although defective samples are initially rare, their availability in industrial settings increases over time. However, the unsupervised methods often aren't designed to utilise such data effectively. Consequently, supervised anomaly detection is prevalent in industrial settings to maximise performance. Industrial-grade methods such as SegDecNet [4], TriNet [16], and MaMiNet [14] leverage both normal and anomalous samples, with the capability to incorporate image-level labels in a weakly-supervised setting, thus alleviating the work of manual pixel-precise annotation. A downside of this approach is that anomalous data is still needed, albeit not annotated, and presents quite a different paradigm from unsupervised learning.

There have been attempts to use one-class classification methods for anomaly detection, enabling unsupervised and supervised training. However, methods such as Deep SAD [19] and FCDD [13] demonstrate poor performance compared to recent methods, especially in the unsupervised setting.

Other recent approaches, such as BGAD [20], PRN [26], and DRA [6], have extended the supervised approach with synthetic anomaly generation. The main disadvantage of such methods is that they require many known anomalous samples to outperform recent unsupervised methods, and their performance in a strictly unsupervised setting may be inadequate. Furthermore, these methods generate anomalies at the image level, although recent advancements in data augmentation could benefit from latent space generation. Additionally, the complexity of architectures, e.g. BGAD and PRN, leads to longer inference times, potentially limiting their applicability in industrial settings.

3 SuperSimpleNet

The proposed SuperSimpleNet, as illustrated in Fig. 2, builds upon the foundation laid by SimpleNet [12]. It begins with feature extraction via a pretrained convolutional network (detailed in Sect. 3.1), followed by upscaling and pooling processes designed to encapsulate the neighbouring context. Subsequently, these features are adapted to a common latent space via a feature adaptor (further described in Sect. 3.2). A notable enhancement over the original model is the introduction of an innovative approach for generating synthetic anomalies, playing a pivotal role in the model's enhanced performance across both unsupervised and supervised scenarios. This advancement is primarily attributed to the creation of anomaly regions at the feature level, employing a binarised Perlin noise mask (expounded in Sect. 3.3). The refined features are then funnelled into the segmentation and classification modules (outlined in Sect. 3.4).

This method exclusively depends on the synthetically produced masks and labels in the unsupervised setting. However, the inclusion of synthetic anomalies markedly elevates the system's efficacy also in the supervised setting, particularly when integrated with actual ground truth data (discussed in Sect. 3.3 and Sect. 3.5). During inference, the framework operates in a seamless end-to-end fashion (Sect. 3.6).

Subsequent sections will provide a thorough exposition of each module, encompassing training and inference details, ensuring a comprehensive understanding of the system's architecture and functionality.

3.1 Feature Extractor

Following the design principles of SimpleNet [12], we employ a ResNet-like [8] convolutional neural network pretrained on ImageNet as the feature extractor. Specifically, we utilise a WideResNet50 [22], extracting features from its 2nd and 3rd layers. Due to ResNet-like networks' architectural characteristics, output features are of relatively low resolution. This limits the effective detection of smaller anomalies and compromises the segmentation precision.

To effectively address this challenge, we have extended the base SimpleNet extractor by incorporating a new upscaling strategy prior to feature concatenation. Our methodology introduces an additional layer of upscaling, effectively doubling the previously applied scaling factor. As a result, layer 3 is enlarged by

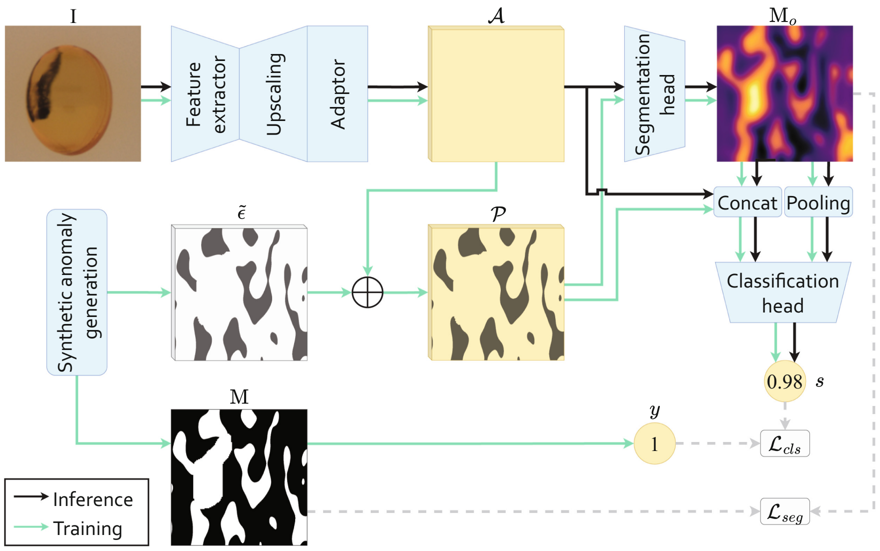


Fig. 2. SuperSimpleNet's architecture. Features are first extracted, upscaled, and adapted. During training, synthetic anomalies are generated in latent space by adding Gaussian noise to the adapted feature map $\mathcal{A}$. The noise is limited to regions generated by binarised Perlin mask and non-anomalous regions (depicted by $\tilde{\epsilon}$). The perturbed feature map $\mathcal{P}$ is then used as the input for the segmentation head to predict an anomaly mask M_o. The predicted anomaly mask M_o and the perturbed feature map $\mathcal{P}$ are then used as the input for the classification head, producing the anomaly score s. The produced anomaly score s and the predicted mask M_o are during the training supervised by the anomaly mask M and y, where y is set to 1 if the image contains an anomaly (synthetic or real) and to 0 otherwise. During inference, the anomaly generation phase is omitted, and M_o and s are produced directly from the adapted feature map $\mathcal{A}$.

a factor of 4, while layer 2 undergoes a doubling in size. This approach ensures both layers achieve equal dimensions, thus allowing for a seamless concatenation.

Afterwards, as in SimpleNet [12], the neighbouring context is encapsulated through the application of local average pooling, employing a mean kernel of size 3×3. This step yields an upscaled feature map where every element is enriched with information from its surroundings.

3.2 Feature Adaptor

While the representations from pretrained backbones can transfer well to the task of anomaly detection [9], a feature adaptor as in SimpleNet [12] is employed to further improve the features for the task. It is implemented as a simple linear layer, producing the adapted features, denoted as $\mathcal{A}$.

3.3 Feature-Space Anomaly Generation

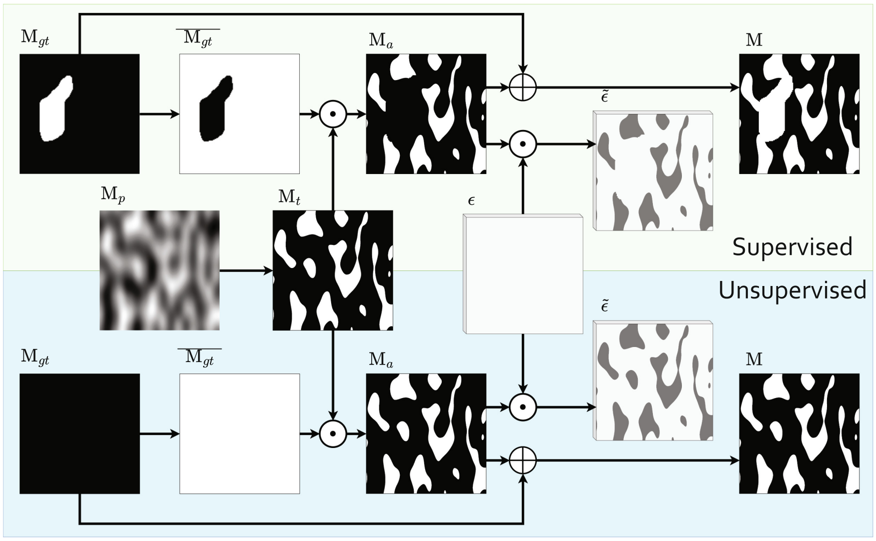


Fig. 3. Synthetic anomaly generation. Synthetic anomaly masks M_a are generated using Perlin Noise. In the unsupervised setting, Gaussian noise is added to all the regions denoted by the thresholded Perlin Noise mask M_t. In contrast, in the supervised setting, noise is omitted from the regions with actual anomalies, denoted by M_{gt}. The final anomaly mask M is constructed from M_a and M_{gt}, and holds information on both where the Gaussian Noise is added and where the actual anomalies lie.

The anomaly generation process (visualised in Fig. 3) begins with the creation of a Perlin binary anomaly mask M_t by thresholding a generated Perlin noise image M_p (similarly as in [7,23,27]). All the regions containing actual anomalies, delineated by M_{gt}, are removed from M_t, resulting in M_a (Fig. 3 – green section). In the unsupervised setting, M_{gt} is always empty during training (Fig. 3 – blue section). Subsequently, Gaussian noise ϵ, sampled from the Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, is selectively applied only within the confines defined by M_a, as illustrated in Fig. 3, and then added to the adapted features $\mathcal{A}$ to produce the perturbed feature map $\mathcal{P}$. To increase the stability of the training regime, the duplication of the adapted features is retained, but unlike the original SimpleNet, noise is applied to both the original and the copy following the process described above. Through this refined strategy, SuperSimpleNet achieves a higher level of precision in anomaly simulation, focusing on creating more realistic, spatially coherent, yet highly randomised anomalous regions. This inherent randomness safeguards the model against overly depending on specific patterns, which might not be representative of unseen data.

A direct method for the *supervised learning setting* might involve substituting the synthetically generated anomaly mask, M_a, with the actual ground truth anomaly mask, M_{gt}. Yet, we recognised that the defects present in the training dataset often fail to fully encompass the vast spectrum of potential defects. To address this limitation, we supplement the model with additional synthetic anomalies created using the methodology employed in the unsupervised framework to represent the defect distribution more comprehensively. Gaussian noise is added solely to non-defective areas to ensure the model gets as much actual defect data as possible. Consequently, the anomaly masks M for the supervised learning scenario are formulated from a blend of authentic ground truth data and synthetic anomalies, enriching the model's exposure to a wider array of defect variations and bolstering its detection and generalisation capabilities.

3.4 Segmentation-Detection Module

We have further extended the architecture to enhance anomaly detection performance by introducing a *classification head*, D_{cls}, while maintaining the *segmentation head*, D_{seg}, as it was in SimpleNet. The classification head's design is straightforward, consisting of a single 5×5 convolutional block and a linear layer. This structure allows the model to understand the global semantics of the image, reducing the overall count of false positives. It also makes it easier for the model to detect significant changes in small areas, thereby increasing the detection rate of smaller anomalies that might have gone unnoticed before.

As displayed in Fig. 2, an anomaly mask M_o is first produced using the segmentation head. This mask M_o is then concatenated with the adapted feature map $\mathcal{A}$ (or noise-augmented feature map $\mathcal{P}$ during training) and used as input for the convolutional block of the classification head. Both, the output from the convolutional block and the anomaly map M_o undergo average pooling and max pooling, after which they are combined and fed into the final linear layer, producing an image-level anomaly score s.

3.5 Loss Function

The truncated $\mathcal{L}_1$ loss is used for the segmentation head. The loss consists of two cases defined in Eq. 1 where th is the truncation term preventing overfitting (in our case, 0.5):

$$l_{i,j} = \begin{cases} max(0, th - D_{seg}(P_{i,j})); \text{ if M[i, j]} = 1 \\ max(0, th + D_{seg}(P_{i,j})); \text{ otherwise} \end{cases} . \tag{1}$$

The total truncated $\mathcal{L}_1$ loss, denoted by $\mathcal{L}_{1t}$, is computed by averaging the loss $l_{i,j}$ across all elements within the predicted anomaly mask. To better guide and stabilise the training process, we additionally incorporate focal loss [11] due to its improved performance in scenarios with unbalanced data, resulting in the formulation of the final segmentation loss, denoted as $\mathcal{L}_{seg}$:

$$\mathcal{L}_{seg} = \mathcal{L}_{1t} + \mathcal{L}_{foc} . \tag{2}$$

For the classification loss $\mathcal{L}_{cls}$, the focal loss [11] is employed:

$$\mathcal{L}_{cls} = \mathcal{L}_{foc} \ . \tag{3}$$

The final loss is the sum of the segmentation and classification loss:

$$\mathcal{L} = \mathcal{L}_{seg} + \mathcal{L}_{cls} \ . \tag{4}$$

The target in segmentation loss is the anomaly mask M, which delineates the regions with synthetic and real anomalies. The target anomaly label y for classification loss is derived from M, i.e. y is set to 1 if the image contains an anomaly (synthetic or real) and to 0 otherwise.

3.6 Inference

During inference, the network predicts the anomaly map and anomaly score by bypassing the anomaly generation phase. The segmentation head outputs an anomaly mask M_o. This mask undergoes interpolation to match the input image's size and is further refined by applying a Gaussian filter with $\sigma = 4$, yielding the final anomaly map. The anomaly score for each image is given by the value s, produced by the classification head D_{cls}.

4 Experiments

An extensive evaluation of the proposed method is performed in both the supervised and the unsupervised setting. The setup is described first, followed by the results.

4.1 Datasets

The performance in the supervised setting is evaluated on two real-world, reliable, and well-annotated datasets: Sensum Solid Oral Dosage Forms (SensumSODF) [16] and Kolektor Surface-Defect Dataset 2 (KSDD2) [4]. SensumSODF consists of two categories, each a different type of solid oral dosage form: a capsule and a softgel. Both categories contain normal and annotated anomalous samples with defects of varying complexity and size. SensumSODF does not contain a predefined train-test split. Due to that, we followed an already defined protocol [16], involving a 3-fold cross-validation. KSDD2 is constructed from images of production items captured using a visual inspection system. Both train and test split contain normal and precisely annotated anomalous samples with many in-distribution defects. Figure 4 shows some examples from both datasets.

The unsupervised regime is evaluated on two established datasets: MVTec AD [3] and VisA [28]. MVTec AD encompasses 15 categories, while VisA comprises 12 different categories. Each category consists of a training set with only normal images and a test set with normal and pixel-precise annotated anomalous images. Anomalies present in both datasets are of various types, shapes, and scales. Figure 5 shows some examples from both datasets.

4.2 Evaluation Metrics

Evaluation metrics depend on the used dataset and recent literature. Image-level performance for SensumSODF, MVTec AD, and VisA is evaluated using the Area Under the Receiver Operator Curve (AUROC). Recent works have strayed away from using pixel-level AUROC for the pixel-level evaluation on these three datasets and rather opted for Area Under the Per-Region Overlap (AUPRO). We chose the same. In the case of KSDD2, a vast majority of recent works evaluate the image and pixel-level performance using Average Precision (AP_{det} and AP_{loc}). Once again, we followed the suite of previous works.

4.3 Implementation Details

The model is trained for 300 epochs with a batch size of 32 using the AdamW optimiser. The adaptor module has a learning rate set to 10^{-4}, while both the segmentation and classification head have it set to $2 * 10^{-4}$ with a weight decay of 10^{-5}. To improve the inference time of the feature adaptor and the segmentation head, we exchange the feature vector reshaping and the consequent fully connected layers with a simple 1×1 convolution, which yields the same results.

The learning rate scheduler is utilised to enhance training stability, multiplying the learning rate by 0.4 after 240 and 270 epochs. To further stabilise the training, the gradient is adjusted by stopping the gradient flow from the classification head to the segmentation head in the unsupervised setting and clipping the gradient in the supervised setting to norm 1.

Following SimpleNet [12], Gaussian noise is sampled from $\mathcal{N}(0, \sigma^2)$ with $\sigma = 0.015$. Perlin noise is binarised using a threshold of 0.6 for VisA, KSDD2, and SensumSODF, generating thinner and smaller anomalies. Since MVTec AD contains larger anomalies, a threshold of 0.2 is used. Synthetic anomalies are added to 50% of the images.

All input images are normalised using ImageNet normalisation. MVTec AD and VisA use image dimensions of 256×256 without center-crop. For KSDD2, following the original protocol, a 232×640 resolution is used. For SensumSODF, capsule and softgel categories have resolutions of 192×320 and 144×144, respectively. To extend the anomalous samples, flipping is used as in [16] where batches are composed of equal anomalous and normal samples, as in [4]. As stated in Subsect. 4.1, we followed predefined train-test splits for KSDD2 [4], MVTec AD [3], and VisA [28], while we used 3-fold cross-validation for SensumSODF, as defined in the original paper [16].

For comparison, we extended the original SimpleNet to support training in a supervised manner. This was done by changing the loss design to consider the labelled defects, i.e. predicting the defective regions inside the ground truth mask (M_{gt}) as anomalous. All other parameters are kept the same as the original [12].

The metrics are calculated based on the model resulting from the final epoch in both settings and all categories. For SuperSimpleNet and SimpleNet, the average performance of 5 runs with different seeds is reported, along with the corresponding standard deviations.

4.4 Experimental Results

Results in the supervised setting. We compared our method with the current state-of-the-art methods for the supervised setting: SegDecNet [4], DRA [6], BGAD [20], PRN [26], TriNet [16] and SimpleNet [12]. The results on SensumSODF are displayed in Table 1, where SuperSimpleNet achieves the best result with a mean anomaly detection AUROC of 97.8%, surpassing the previous state-of-the-art by 0.9 percentage points (p.p.), reducing the error by 29%. Anomaly detection and localisation results in a supervised setting on KSDD2 are shown in Table 2. SuperSimpleNet achieves a state-of-the-art AP_{det} of 97.4%. We hypothesise that SuperSimpleNet achieves such a high performance in the supervised setting due to the classification head, which can efficiently learn to capture more global information due to the presence of real and synthetic anomalies during training.

Table 1. Results of Supervised anomaly detection (AUROC) and localisation (AUPRO) on the SensumSODF dataset.

	[4]	[6]	[20]	[26]	[16]	[12]	Ours
Detection	83.4	90.1	94.3	80.6	96.9	88.4 (± 1.84)	97.8 (± 0.13)
Localisation	75.2	-	97.0	66.0	-	89.6 (± 1.14)	93.0 (± 0.54)

Table 2. Results for supervised anomaly detection (AP_{det}) and localisation (AP_{loc}) on the KSDD2 dataset.

	[25]	[4]	[6]	[20]	[26]	[14]	[10]	[12]	Ours
Detection	95.2	95.4	89.3	92.7	78.6	96.2	99.9	93.5 (± 1.05)	97.4 (± 0.25)
Localisation	85.5	67.6	-	76.5	48.2	-	-	75.9 (± 2.40)	82.1 (± 0.50)

Figure 4 shows qualitative results of our supervised model on KSDD2 [4] and SensumSODF [16]. It can be discerned that SuperSimpleNet improves upon previous methods in two aspects: first, it outputs more precise masks in comparison to its competitors PRN [26], BGAD [20] and its baseline SimpleNet [12]. PRN outputs masks that majorly underestimate the total area of the defect, whilst BGAD heavily overestimates it. The more precise masks are due to the upscaling module and improved training.SuperSimpleNet improves upon both and reduces

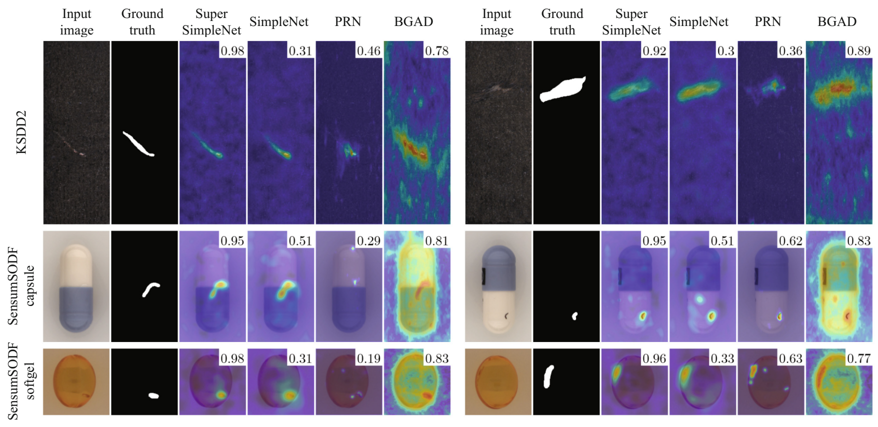


Fig. 4. Qualitative comparison of anomaly maps produced in the supervised setting: the input image, the ground truth, and the overlaid anomaly map for SuperSimpleNet, SimpleNet, PRN, and BGAD. The first row displays two samples from KSDD2; the second and third are SensumSODF capsule and softgel respectively. The anomaly score is displayed in the top right corner of each overlaid anomaly map. The anomaly score from the classification head proves to be more reliable than the established maximum value of the anomaly mask.

the area of the false positive regions produced by the baseline SimpleNet [12]. Secondly, SuperSimpleNet estimates the image-level anomaly score more accurately than its competitors due to its classification head. While SimpleNet and PRN often predict scores below 0.5, SuperSimpleNet predicts scores near the upper limit.

Results in the unsupervised setting. We compared our method with the current state-of-the-art methods for the unsupervised setting: AST [18], DSR [25], EfficientAD [2], FastFlow [21], Patchcore [17], DRÆM [23] and SimpleNet [12]. Table 3 reports performance on the MVTec AD dataset.

SuperSimpleNet achieves state-of-the-art performance with mean anomaly detection of 98.4%. Results on the VisA dataset are shown in Table 4. SuperSimpleNet reaches state-of-the-art performance with anomaly detection AUROC of 93.4%. Whilst the classification head can't learn as efficiently as in the supervised setting, the improved synthetic anomaly generation improves the models' discriminative capabilities, leading to better downstream anomaly detection. Qualitative results of the unsupervised model on MVTec AD [3] and VisA [28] are shown in Fig. 5. Like the supervised model, SuperSimpleNet outputs more precise masks than SimpleNet and outputs higher anomaly scores. Similarly, SuperSimpleNet outputs a lower rate of false positives in the background than SimpleNet.

Table 3. Anomaly detection and localisation (AUROC/AUPRO) on MVTec AD dataset.

	[18]	[25]	[2]	[21]	[17]	[23]	[12]	Ours
Carpet	98.3/89.4	99.9/92.4	99.3/92.7	97.5/92.9	98.7/93.0	97.0/92.9	97.8/92.3	98.4/92.3
Grid	98.7/79.7	100/88.9	99.9/88.9	100/96.0	98.2/91.9	99.9/98.4	99.6/93.5	99.3/93.1
Leather	100/90.4	99.6/97.3	100/98.3	100/99.1	100/96.9	100/97.8	100/96.0	100/96.9
Tile	99.1/72.0	100/85.6	100/85.7	99.9/87.3	100/87.9	99.6/98.5	98.8/85.4	99.7/84.2
Wood	99.2/71.9	92.2/84.5	99.5/90.2	98.9/93.1	99.1/85.7	99.1/93.5	97.3/77.7	99.3/84.7
Bottle	100/86.0	99.8/95.9	100/95.7	100/89.3	100/94.0	99.2/97.0	100/92.9	100/90.4
Cable	98.0/75.6	96.6/88.5	95.2/92.5	93.9/89.9	99.5/94.1	91.8/75.6	98.9/90.6	98.1/88.5
Capsule	98.8/88.1	96.7/89.8	97.9/97.6	98.1/95.4	98.5/93.4	98.5/91.0	98.0/91.3	98.7/92.3
Hazelnut	100/89.5	99.5/94.7	99.4/95.7	98.9/95.6	100/95.1	100/98.6	99.3/89.4	99.8/94.5
Metal nut	97.8/75.6	99.8/91.8	99.6/94.4	99.6/92.3	99.9/94.1	98.7/94.0	99.1/88.3	99.5/90.9
Pill	99.0/71.7	98.3/95.9	98.6/96.1	96.7/93.9	95.1/93.9	98.9/88.2	97.0/93.3	98.1/94.2
Screw	99.1/87.1	95.8/91.3	97.0/96.4	84.5/89.7	97.3/94.6	93.9/98.2	88.7/91.3	92.9/95.3
Toothbrush	97.5/67.1	100/95.8	100/93.3	89.2/87.0	95.3/85.7	100/90.3	89.9/91.8	92.2/85.1
Transistor	98.9/90.6	93.5/78.9	99.9/91.2	98.5/92.0	99.8/94.8	93.1/81.4	99.5/90.0	99.9/91.5
Zipper	99.1/83.2	99.7/91.1	99.7/93.4	98.5/93.7	99.2/94.7	100/96.2	99.4/94.5	99.6/93.1
Average	98.9/81.2	98.1/90.8	99.1/93.5	96.9/92.5	98.7/92.7	98.0/92.8	97.6/90.5	98.4/91.1

Stability comparison. We also evaluated the stability of the training. A comparison in anomaly detection performance and the standard deviation between several training runs with SimpleNet is shown in Fig. 6. SuperSimpleNet improves the performance of SimpleNet and reduces the standard deviation between several training runs, making it more reliable and robust. The same holds even for the unsupervised setting for which SimpleNet was specifically designed.

Computational efficiency. Fig. 1 showcases the balance between performance and computational efficiency, measured using an NVIDIA Tesla V100S. SuperSimpleNet is the only model designed to work well in both supervised and unsupervised regimes. At the same time, SuperSimpleNet offers great anomaly detection performance while fulfilling the low inference time requirement with an inference time of 9.3 ms and a throughput of 268 images per second. The protocol from [2] is followed to measure computational efficiency. A more detailed description and additional metrics are presented in Supplementary Material.

Table 4. Anomaly detection and localisation (AUROC/AUPRO) on VisA dataset.

	[18]	[25]	[2]	[21]	[17]	[23]	[12]	Ours
Candle	99.4/94.1	86.4/79.7	98.4/95.7	96.8/93.7	98.6/96.4	92.7/92.7	92.5/89.8	97.1/93.6
Capsules	85.4/68.2	93.4/74.5	93.5/96.9	83.0/89.3	76.4/57.5	90.2/85.4	78.9/84.8	81.5/80.1
Cashew	95.1/79.1	85.2/61.5	97.2/94.2	90.0/84.7	97.9/88.8	85.5/67.6	91.9/82.6	93.0/86.3
Chewing gum	100/78.5	97.2/58.2	99.9/83.1	99.8/86.8	98.9/75.9	95.2/58.0	99.0/84.1	99.3/87.7
Fryum	99.0/58.7	93.0/65.5	96.5/86.7	98.6/72.9	94.8/80.8	88.6/80.4	95.4/90.3	96.8/79.3
Macaroni 1	93.9/87.2	91.7/57.7	99.4/99.0	94.8/94.1	95.8/75.0	94.2/86.3	94.2/97.3	93.1/95.9
Macaroni 2	72.1/80.4	79.0/52.2	96.7/98.8	80.5/87.3	77.7/49.1	86.6/96.3	71.8/85.9	75.0/89.7
PCB1	99.2/89.3	89.1/61.3	98.5/97.1	95.5/91.2	98.9/91.3	75.9/61.1	92.5/88.7	96.9/92.9
PCB2	98.4/85.7	96.4/84.9	99.5/95.0	96.1/87.0	97.1/85.7	98.9/76.2	93.6/89.0	97.5/85.4
PCB3	97.4/87.7	97.0/79.5	98.9/94.0	94.0/77.6	96.3/73.2	94.4/83.5	92.6/90.3	94.4/83.0
PCB4	99.7/80.1	98.5/62.1	98.9/92.7	98.4/88.6	99.4/88.7	98.6/73.4	97.9/81.6	98.4/87.3
Pipe fryum	99.4/89.4	94.3/80.5	99.7/94.7	99.6/89.0	99.7/94.5	97.6/74.6	94.6/91.1	97.6/88.0
Average	94.9/81.5	91.8/68.1	98.1/94.0	93.9/86.8	94.3/79.7	91.5/78.0	91.2/88.0	93.4/87.4

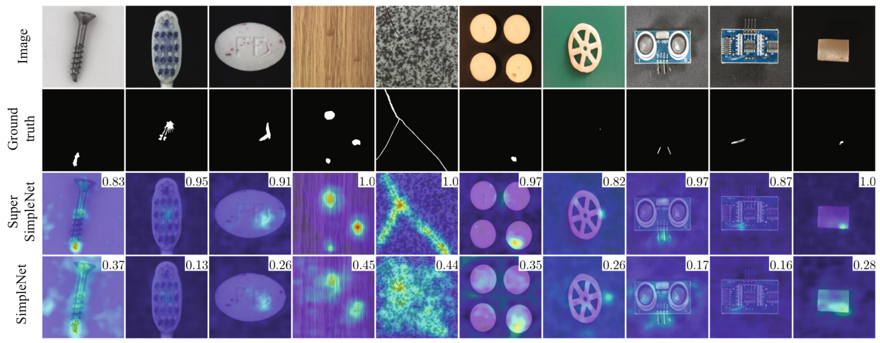


Fig. 5. Qualitative comparison of anomaly maps produced by unsupervised SuperSimpleNet and SimpleNet. The top row shows the input anomalous image. The second row displays the ground truth anomaly mask. The third and fourth rows contain anomaly maps generated by SuperSimpleNet and SimpleNet, respectively. The anomaly score is displayed in the top right corner of each anomaly map.

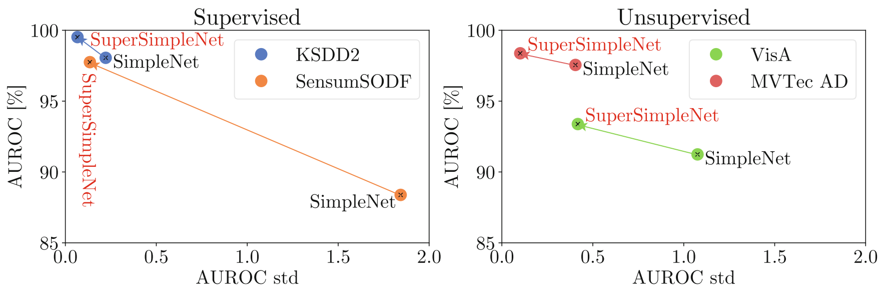


Fig. 6. Comparison of SuperSimpleNet with SimpleNet in anomaly detection in terms of AUROC and its standard deviation on all four datasets.

5 Ablation Study

To determine the contributions of each component in SuperSimpleNet, each newly introduced module is evaluated by excluding it from the architecture. Quantitative results are shown in Table 5, while the qualitative results are shown in Fig. 7.

Table 5. Ablation study results on anomaly detection and localisation (AUROC / AUPRO) in the supervised setting (mean value of results on SensumSODF and KSDD2) and the unsupervised setting (mean value of results on MVTec AD and VisA), as well as the average of both settings. *SSN* stands for SuperSimpleNet, while *SN* stands for SimpleNet.

Method	Anomaly generation		Architecture			*Super.*		*Unsup.*		*Average*	
	SuperSimpleNet	SimpleNet	Upscale	Cls. head	Opt. train.	Det.	Loc.	Det.	Loc.	Det.	Loc.
SSN **(Ours)**	✓		✓	✓	✓	98.6	95.6	95.9	89.3	97.3	92.4
$SSN_{no\_upscale}$	✓			✓	✓	98.2	93.0	94.9	88.3	96.6	90.6
SSN_{no_cls}	✓		✓		✓	95.5	96.2	96.2	89.6	95.9	92.9
$SSN_{no\_cls\&SN\_anom}$		✓	✓		✓	94.9	96.6	96.0	89.1	95.4	92.8
SSN_{old_train}	✓		✓	✓		98.2	93.5	92.7	85.3	95.5	89.4
$SSN_{overlap}$	Overlap		✓	✓	✓	98.3	95.9	95.9	89.3	97.1	92.6
SSN_{SN_anom}		✓	✓	✓	✓	97.9	96.5	88.0	87.5	93.0	92.0
SSN_{no_anom}			✓	✓	✓	98.1	89.8	-	-	-	-
SN_{SSN_anom}	✓					91.3	91.0	94.6	86.2	93.0	88.6
SN		✓				93.2	93.2	94.4	89.3	93.8	91.2

Upscaling module. Excluding the upscaling of features ($SSN_{no\_upscale}$) causes a decline in detection performance, a 0.4 p.p. decrease for the supervised setting and a 1.0 p.p. in the unsupervised setting. A noticeable decline also occurs in the localisation performance with a drop of performance by 2.6 p.p. and 1.0 p.p. for the supervised setting and the unsupervised setting, respectively. The results suggest the importance of feature size in the final predictions, as also visible from less precise segmentation in Fig. 7.

Classification head. The exclusion of a classification head (SSN_{no_cls} - where the anomaly score s is obtained as the maximum value from the anomaly map) notably reduces anomaly detection performance (by 3.1 p.p) in the supervised setting. This is due to stronger discriminative capabilities inside the classification head, which are beneficial when both synthetic and real data are available. The decreased detection performance is also indicated by lower anomaly scores in Fig. 7. However, excluding the classification head in the unsupervised setting leads to slight improvement (by 0.3 p.p). This is because the strong discriminative properties can hinder performance when relying solely on synthetic data. The gradient also flows from the classification head to the segmentation head in the supervised setting, adjusting the anomaly map to be more suitable for clas-

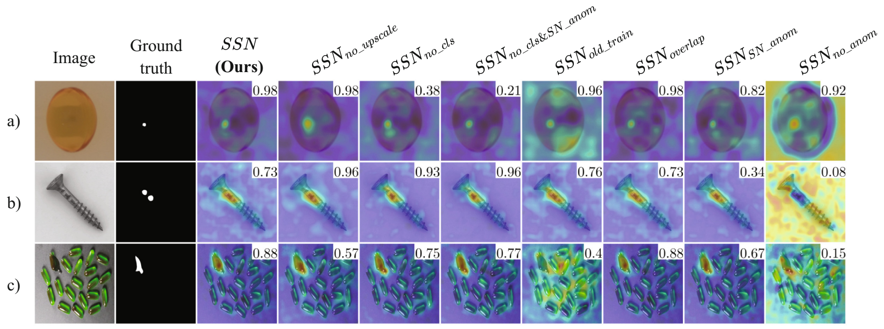


Fig. 7. Qualitative comparison of anomaly maps produced by different versions of SuperSimpleNet from ablation study. a) shows a sample from the supervised setting; b) and c) show two samples from the unsupervised setting. The anomaly score is displayed in the top right corner of each overlaid anomaly map. Observing both the anomaly map and the anomaly score provides an insight into how each component contributes towards our final model.

sification. Removing the classification head thus slightly increases localisation performance.

Improved training. The effect of upgrading the loss, incorporating a scheduler, and gradient adjustments can be deduced from the SSN_{old_train} experiment in the table. These modifications notably impact both detection (by 0.4 p.p. and by 3.2 p.p.) and localisation (by 2.1 p.p. and by 4.0 p.p.). Unsupervised performance is particularly improved, largely due to a more stable training process, which prevents poor final results like the capsules in Fig. 7 – row c.

Addition of synthetic anomalies to anomalous regions. In the supervised setting, synthetic anomalies are exclusively generated within non-anomalous regions. The results of the $SSN_{overlap}$ experiment showcase the impact of anomaly generation without this constraint. This approach leads to slightly worse detection performance (0.3 p.p.). We hypothesise that augmenting already anomalous regions leads to the loss of genuine anomalous information. Since this change only applies to the supervised setting, the unsupervised results remain unaffected.

Anomaly mask generation. The importance of anomaly mask generation was evaluated by using the feature duplication strategy from SimpleNet. SimpleNet copies the features and adds the noise to the entirety of the copy. The strategy from SimpleNet (SSN_{SN_anom}) leads to a decrease in detection performance in both the supervised (by 0.7 p.p.) and the unsupervised setting (by 7.9 p.p.). We hypothesise the major decline in unsupervised performance is due to the incompatibility of this mask generation strategy with the classification head, as it struggles to efficiently learn defect bordering regions. This hypothesis is further supported by the $SSN_{no\_cls\&SN\_anom}$ experiment, where SimpleNet strategy is used for SuperSimpleNet with the classification head removed. This improves the unsupervised performance but leads to poor supervised performance, indicating that our anomaly generation strategy is crucial for good simultaneous supervised and unsupervised performance when using a classification head.

Synthetic anomaly generation strategy. To evaluate the importance of synthetic anomalies in the supervised setting, only real anomalies were used during training. As evident from the SSN_{no_anom} experiment in the table, the detection performance achieves a decline of 0.5 p.p., whilst the localisation performance achieves a major decline of 5.8 p.p. The results indicate the importance of synthetic anomalies during training. The results from the unsupervised setting are omitted due to the inability of the model to learn a boundary without the presence of synthetic anomalies during training, also seen in Fig. 7.

6 Conclusion

A novel discriminative anomaly detection model, SuperSimpleNet, has been proposed to meet the industry's requirements (performance, speed, robustness, and stable training). It offers the flexibility to be trained in both the supervised and the unsupervised setting, making full use of all available training data. This ability is rarely achieved in previous methods. The proposed model is also robust, achieving a predictable performance independent of the training run. The efficiency of SuperSimpleNet is validated in both the supervised and the unsupervised setting. In the supervised setting, the method is evaluated on two well-established benchmarks, SensumSODF and KSDD2, achieving 97.8% AUROC on SensumSODF and 97.4% AP_{det} on KSDD2. On SensumSODF, SuperSimpleNet surpasses all previous methods by 0.9%. SuperSimpleNet also achieves state-of-the-art results in the unsupervised setting on two well-established benchmarks, MVTec AD and VisA, with 98.4% and 93.4% AUROC, respectively. It achieves state-of-the-art results whilst holding an inference time of 9.3 ms and a throughput of 268 images per second.

Limitations and future work. SuperSimpleNet mostly struggles in the unsupervised setting with categories containing multiple objects. SuperSimpleNet is also heavily dependent on the used backbone, as the magnitude of the Gaussian noise needs to be adjusted for each backbone. Also, anomaly detection performance deteriorates if the backbone fails to extract meaningful features. In the future, we will extend the model to also operate in weakly-supervised and mixed-supervised settings. This means we will have pixel-level annotations for only a subset of defective images while having image-level annotations for the entire training set. Such a change will further improve direct usability and alleviate the need for pixel-level annotations. The results also indicate that combining the knowledge from the unsupervised and the supervised domain is a viable step for the anomaly detection field in the future.

Acknowledgements. This work was in part supported by the ARIS research project L2-3169 (MV4.0), research programme P2-0214 and the supercomputing network SLING (ARNES, EuroHPC Vega).

References

1. Akcay, S., Atapour-Abarghouei, A., Breckon, T.P.: Ganomaly: semi-supervised anomaly detection via adversarial training. In: Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14, pp. 622–637. Springer (2019)
2. Batzner, K., Heckler, L., König, R.: EfficientAD: accurate visual anomaly detection at millisecond-level latencies. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 128–138 (2024)
3. Bergmann, P., Batzner, K., Fauser, M., Sattlegger, D., Steger, C.: The MVTec anomaly detection dataset: a comprehensive real-world dataset for unsupervised anomaly detection. Int. J. Comput. Vision **129**(4), 1038–1059 (2021)
4. Božič, J., Tabernik, D., Skočaj, D.: Mixed supervision for surface-defect detection: from weakly to fully supervised learning. Comput. Ind. **129**, 103459 (2021)
5. Deng, H., Li, X.: Anomaly detection via reverse distillation from one-class embedding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9737–9746 (2022)
6. Ding, C., Pang, G., Shen, C.: Catching both gray and black swans: open-set supervised anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7388–7398 (2022)
7. Fučka, M., Zavrtanik, V., Skočaj, D.: TransFusion – a transparency-based diffusion model for anomaly detection. In: European Conference On Computer Vision (2024)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference On Computer Vision And Pattern Recognition, pp. 770–778 (2016)
9. Heckler, L., König, R., Bergmann, P.: Exploring the importance of pretrained feature extractors for unsupervised anomaly detection and localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2916–2925 (2023)
10. Li, Y., Wu, X., Li, P., Liu, Y.: Ferrite beads surface defect detection based on spatial attention under weakly supervised learning. IEEE Trans. Instrum. Meas. **72**, 1–12 (2023)
11. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference On Computer Vision, pp. 2980–2988 (2017)
12. Liu, Z., Zhou, Y., Xu, Y., Wang, Z.: Simplenet: a simple network for image anomaly detection and localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 20402–20411 (2023)
13. Liznerski, P., Ruff, L., Vandermeulen, R.A., Franks, B.J., Kloft, M., Müller, K.R.: Explainable deep one-class classification. In: International Conference on Learning Representations (2021)
14. Luo, X., Li, S., Wang, Y., Zhan, T., Shi, X., Liu, B.: MaMiNet: memory-attended multi-inference network for surface-defect detection. Comput. Ind. **145**, 103834 (2023)
15. Pirnay, J., Chai, K.: Inpainting transformer for anomaly detection. In: Sclaroff, S., Distante, C., Leo, M., Farinella, G.M., Tombari, F. (eds.) Image Analysis and Processing - ICIAP 2022, pp. 394–406. Springer International Publishing, Cham (2022)
16. Rački, D., Tomaževič, D., Skočaj, D.: Detection of surface defects on pharmaceutical solid oral dosage forms with convolutional neural networks. Neural Comput. Appl. **34**(1), 631–350 (2021)

17. Roth, K., Pemula, L., Zepeda, J., Schölkopf, B., Brox, T., Gehler, P.: Towards total recall in industrial anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14318–14328 (2022)
18. Rudolph, M., Wehrbein, T., Rosenhahn, B., Wandt, B.: Asymmetric student-teacher networks for industrial anomaly detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 2592–2602 (2023)
19. Ruff, L., et al.: Deep semi-supervised anomaly detection. In: International Conference on Learning Representations (2020)
20. Yao, X., Li, R., Zhang, J., Sun, J., Zhang, C.: Explicit boundary guided semi-push-pull contrastive learning for supervised anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 24490–24499 (2023)
21. Yu, J., et al.: Fastflow: Unsupervised Anomaly Detection and Localization via 2D Normalizing Flows. arXiv preprint arXiv:2111.07677 (2021)
22. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: Proceedings of the British Machine Vision Conference 2016. British Machine Vision Association (2016)
23. Zavrtanik, V., Kristan, M., Skočaj, D.: Dræm-a discriminatively trained reconstruction embedding for surface anomaly detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 8330–8339 (2021)
24. Zavrtanik, V., Kristan, M., Skočaj, D.: Reconstruction by inpainting for visual anomaly detection. Pattern Recogn. **112**, 107706 (2021)
25. Zavrtanik, V., Kristan, M., Skočaj, D.: DSR–a dual subspace re-projection network for surface anomaly detection. In: European conference on computer vision, pp. 539–554. Springer (2022)
26. Zhang, H., Wu, Z., Wang, Z., Chen, Z., Jiang, Y.G.: Prototypical residual networks for anomaly detection and localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16281–16291 (2023)
27. Zhang, X., Li, S., Li, X., Huang, P., Shan, J., Chen, T.: DeSTSeg: segmentation guided denoising student-teacher for anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3914–3923 (2023)
28. Zou, Y., Jeong, J., Pemula, L., Zhang, D., Dabeer, O.: SPot-the-difference self-supervised pre-training for anomaly detection and segmentation. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXX, pp. 392–408. Springer (2022)

Augmentation of Human Activity Data: Convert, Generate, Transform

Nilah Ravi Nair[1](✉), Arthur Matei[2], Dennis Krön[2], Fernando Moya Rueda[3], Christopher Reining[1], and Gernot A. Fink[2]

[1] Chair of Material Handling and Warehousing, TU Dortmund University, Dortmund, Germany
nilah.nair@tu-dortmund.de
[2] Department of Computer Science, TU Dortmund University, Dortmund, Germany
arthur.matei@tu-dortmund.de
[3] MotionMiners GmBH, Dortmund, Germany

Abstract. Data-driven neural network models trained on human motion data facilitate human activity recognition and identity verification applications. However, large annotated and processed human motion datasets are scarce, leading to the overfitting of models to training data. Thus, it is important to investigate data augmentation techniques to generate additional data to facilitate model generalisation. In addition, the choice of augmentation techniques severely impacts the performance of self-supervised learning. Thus, this work experiments and evaluates various augmentation techniques on seven sensor-based human activity datasets. Three supervised neural network models and one self-supervised learning model were experimented with. We note that due to the high variability in the performance of algorithmic augmentation techniques on time-series human activity datasets, generative data is highly influential in this domain.

Keywords: Human activity recognition · Augmentation · Generation · Human motion · Self-supervised learning

1 Introduction

With the advent of Internet-of-Things (IoT), smart devices for daily living, and sensor-based technologies for industries, time-series data has increased interest in research and development. Sensor-based time-series data facilitate activity recognition, subject re-identification, health and ergonomy analysis, and monitoring with the help of neural networks. However, the availability of annotated datasets is often limited [23]. In addition, dataset imbalance is a common occurrence. As a result, neural network models tend to overfit the training data. To

Nilah Ravi Nair, Arthur Matei : These authors contributed equally to this work.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78192-6_5.

overcome these issues, data augmentation (DA) techniques are of interest [11]. Authors of [7,11] note that, compared to image data, time-series data augmentation techniques are not extensively explored. In addition, the applicability of time-series data augmentation is specific to the dataset and neural network used [11]. It is unclear what features of the dataset or the neural network lead to the variation in the impact of the augmentation technique.

Further, [9,12] suggests that the algorithmic time-series augmentation techniques must be explored before focusing on generative models for synthetic data creation. For instance, [9] notes that traditional augmentations may perform better than Generative Adversarial Networks (GANs), as GANs may radically change the structure of the time-series data even though the data utilized may fool the discriminator. Synthetic data generation is not limited to generative networks. With the support of motion capture (MoCap) data, a concept of virtual inertial measurement units (IMUs) is used to obtain synthetic sensor readings from a location of the MoCap skeleton [16]. Consequently, a cumulative evaluation of the augmentation techniques focused on activity datasets may help develop an augmentation application strategy.

Augmentation strategies are integral to self-supervised learning (SSL). Due to the expensive nature of annotated sensor-based human activity data, a shift of interest from supervised to self-supervised learning is visible [30]. As augmentation techniques play an essential role in self-supervised learning, exploring and possibly developing augmentation application strategies for time-series human activity data could benefit the development of robust self-supervised learning for activity recognition [27]. This work aims to answer the following research questions.

RQ1: How do neural network models respond to the augmentations in the human activity training data?
RQ2: How can one choose the data augmentation techniques for their dataset for a specific network model?
RQ3: Is it better to focus on generative models than algorithmic methods for tackling the problems of dataset imbalance and model generalisation?
RQ4: Which augmentation techniques benefit self-supervised learning?

The paper is organised as follows: Sect. 2 explores the recent work on augmentation of time-series datasets, including their application to SSL. Section 3 elaborates on the various augmentation techniques experimented with in this work. Section 5 and Sect. 6 explain the experiments' results and the derived conclusions, respectively.

2 Related Work

The authors of [11] survey various time-series data augmentation techniques focusing on magnitude, time and frequency domain transformations and pattern mixing. The augmentation techniques were evaluated on various networks, such as VGG, ResNet, MLP, and LSTM networks. Slicing, window warping, and dynamic guided warping (DGW) showed a positive impact. Multi-layer perceptron (MLP) and Long Short Term Memory (LSTM) presented significant performance degradation in the presence of augmented data. The authors note that effective augmentation techniques vary depending on the model and dataset. [5] performed experimentation of eight datasets of time-series representative value and noted that rotation, in combination with permutation, improves network performance. Scaling, magnitude warping, and time warping were effective augmentation techniques. However, augmentation techniques specific to HAR datasets are of interest to account for the activity performed in the time-series recording.

Few works focus on generating synthetic data from the human movement recordings of videos and motion capture systems. [10] proposed pose reconstruction to synthetic IMU data. Virtual sensors placed on an SMPL mesh of a MoCap sequence facilitated synthetic data generation. The authors use a bidirectional RNN to train a Deep Inertial Poser network to learn the mapping between human poses and IMU data. Using finite differences, synthetic poses were computed through forward kinematics and their respective linear accelerations. The authors of [1] proposed transfer learning of pose annotations from video datasets to support HAR on IMU datasets. The authors use the pose annotations from videos as multi-channel time series data of human movements. Then, the authors apply a smooth piece-wise spline interpolation to the human joint pose sequences. These interpolations are considered to be synthetic IMU data. The synthetic data is used to train the network as part of transfer learning. [1] showed that source and target datasets of transfer learning, with similar activities, improved performance.

Lastly, extending upon IMU generation data from pose or motion sequences, [14] proposed an IMUGPT, which uses ChatGPT to generate prompts to generate 3D human motion sequences using inverse kinematics; the motion sequences are converted to virtual IMU data.

SSL on unlabelled human activity data has the potential to facilitate and improve HAR. For instance, [30] applied SSL on a large activity tracker dataset with 700.000 person-days of unlabelled data. A ResNet-V2 network with 18 layers and a 1D convolutions model outperformed the baselines on seven benchmark datasets with an average accuracy improvement of 18.4%. Augmentation techniques focusing on the temporal dependencies of human motion, such as reversing in time, permutation, time-warping and weighted sampling augmentation, were evaluated. Following the work of [28], [27] applied noise, scaling, 3D rotation, reversing, flipping, warping, shuffling channels, and perturbation as augmentation for cross-domain HAR using SSL.

These related works point towards a distinct gap in having a holistic view of the effect of augmentation techniques on supervised and unsupervised neural networks and their impact on new models of neural networks, such as transformers. Furthermore, exploring and comparing generative and kinematic transformation as part of dataset augmentation is interesting. Consequently, this work focuses on bridging the above-mentioned gaps.

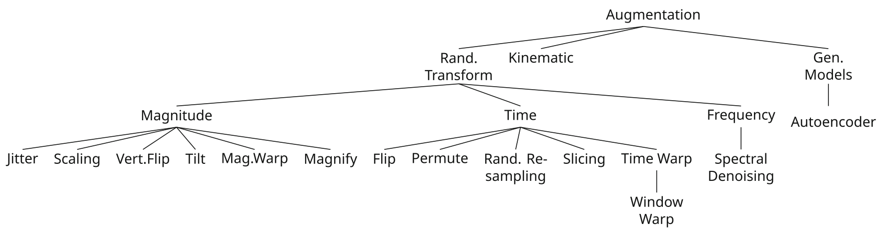


Fig. 1. Data Augmentations applied in this work.

3 Augmentation Techniques

This work explores and analyses various augmentation techniques, specifically for supervised and self-supervised learning, for human activity recognition as an application. Thus, we address three data augmentation groups for multi-channel time-series HAR: random transformations, generative models, and kinematics from pose data[1]. Further, the work attempts to outline the effect of these techniques on different use cases, such as dataset balancing, enhancement, variation, and even learning strategies in specific, self-supervised learning. To further elaborate, the goal is to support ML practitioners and/or researchers in identifying which augmentation technique would be ideal for their goal – for example, if a dataset requires class or subject data balancing within a dataset, directly exploring generative methods may be more valuable to augment the training dataset than exploring random transformations on the dataset. Figure 1 presents the augmentation techniques utilized in this work.

Random Transformation. This work refers to random transformation as an algorithmic augmentation technique. As discussed in [11], random transformations refer to a function $\mathbf{f}(.)$ applied on a segment $\mathbf{x}$ to obtain a modified segment $\mathbf{x}'$. The implementation of these algorithmic augmentations, unless mentioned otherwise, are adapted from [11,12].

Generative Augmentations. Considering use cases where the datasets are imbalanced, generative models are helpful as an augmentation strategy, creating samples of underrepresented activity classes. Given the understanding from [9] that synthetic data generated from GAN may be structurally different from time-series data, an autoencoder design was chosen for the generative model. The autoencoder model can learn the structural features of the original time-series data while introducing randomness in motion. A tCNN architecture developed from the work of [20] was used as the encoder. The encoder consists of four convolutional layers with no pooling layer and ReLU activations. The fully connected layers are replaced with 1×1 convolutional layers to facilitate the data

[1] The code and parameters of the networks are available on https://github.com/nilahnair/ICPR2024_DataAugmentation.

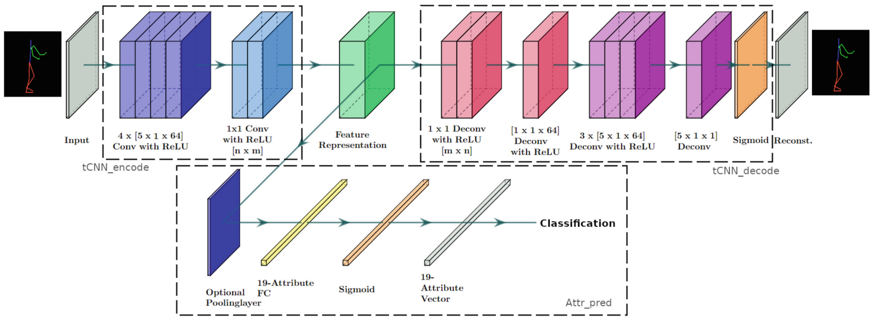


Fig. 2. Architecture of the temporal autoencoder. The grey dashed boxes indicate the parts of the autoencoder referenced in the text. Parts of $tCNN_{encode}$: 🟦: four 1×5 temporal convolutions based on the baseline tCNN. 🟦: additional 1×1 convolution layers. n and m represent the amount of feature maps for the two 1×1 convolution layers. Parts of $tCNN_{decode}$: 🟥: three 1×1 Deconvolution layer. n and m represent the amount of feature maps for the first and second 1x1 deconvolution layers. 🟥: four 5×1 Deconvolution layers. Parts of $Attr_{pred}$: 🟦: optional Poolinglayer. 🟨: fully connected layer with 19 Neurons for attribute vector calculation. 🟧: Sigmoid activation function. (Color figure online)

reconstruction from the computed feature representation. This change helps preserve the time information of the data for synthetic data generation [15]. The encoder will be indicated as $tCNN_{encode}$.

The decoder is tasked with reconstructing the data from the feature representation generated by the encoder. It has six deconvolutional layers with ReLU activation layers. The final decoder layer comprises Sigmoid activations, as the output range is expected to be between zero and one. This method is a consequence of the normalisation of the input data. It is to be noted that, due to the ReLU activation in the encoder, the reconstruction is a consequence of linear regression. This decoder part will be called $tCNN_{decode}$.

Inspired by [29], we propose a fully connected layer parallel to the decoder layer followed by a multi-head output layer. Thus, the extracted feature representation is passed to a fully connected layer with Sigmoid activation for computing an attribute representation of activity classes as in [21] or softmax activation function [20]. This parallel layer will be called $Attr_{pred}$ (Fig. 2).

The autoencoder has two parallel output layers - $tCNN_{decode}$ and $Attr_{pred}$. Traditionally, reconstruction is trained using a Mean Squared Error (MSE) loss function. The authors of [22] used Binary Cross Entropy (BCE) loss for $Attr_{pred}$. Thus, the autoencoder is trained with a combined loss function, MSE plus BCE.

Kinematic Augmentations. Inspired by the inverse kinematics calculation in robotics [4], the authors in [18] propose a method to compute inertial measurements from the derivation of sequences of joint poses of a human, e.g., from marker-based Motion Capturing System (marker-based MoCap). These derivatives will act as a Synthetic On-body Devices (SOBDs), located on the human. The SOBDs will provide linear acceleration $\mathbf{a}$ and angular velocity $\boldsymbol{\omega}$ of the human joints. Figure 3 shows an example of a SOBD located on left wrist of the LARa$_{MoCap}$ [21], a marker-based MoCap dataset. The linear acceleration of the SOBDs is computed by a combination of the linear acceleration and the angular rotation of the joint to a global or given origin frame $\{\{\mathsf{O}\}\}$[2]. Equation (1) presents the acceleration ${}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\mathbf{a}\,(\mathsf{t})$ equation for a frame attached to the joint $\{\mathsf{J}\}$ with respect to the origin frame. ${}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\boldsymbol{\omega}\,(\mathsf{t})$ and ${}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\dot{\boldsymbol{\omega}}\,(\mathsf{t})$ are the angular velocity and acceleration of the frame attached to the joint $\{\mathsf{J}\}$ with respect to a origin frame; and $\mathbf{p}_\mathsf{j}$ the vector from joint $\{\mathsf{J}\}$ to the given origin frame.

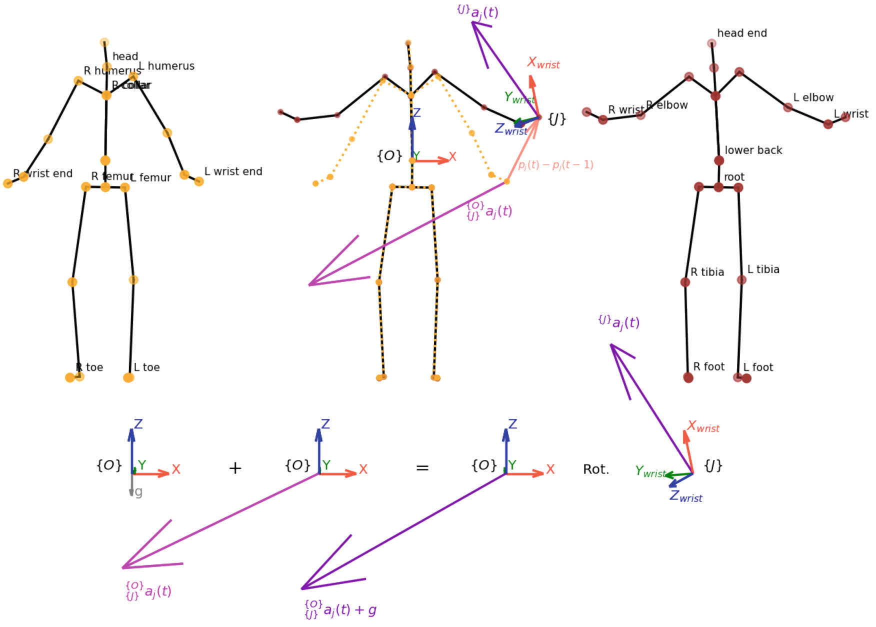


Fig. 3. Example of the linear acceleration of a Synthetic On-body Device (SOBD) from marker-based MoCap from two consecutive poses of the joint $\{\mathsf{J}\}$ located on left wrist of the LARa-MoCap (LARa-M) dataset. A frame attached to joint $\{\mathsf{J}\}$ is represented by the unitarian vectors towards (X, Y, Z), specified by 🟥, 🟩, 🟦. The linear acceleration ${}^{\mathsf{O}}_{\{\mathsf{J}\}}\mathbf{a}_\mathsf{j}\,(\mathsf{t})$ on origin frame is presented in 🟥. The vector from pose $\mathbf{p}_{\{\mathsf{J}\}}\,(\mathsf{t}-1)$ to pose $\mathbf{p}_{\{\mathsf{J}\}}\,(\mathsf{t})$ is given in 🟥. The linear acceleration of the joint $\{\mathsf{J}\}$ given in origin frame O and frame $\{\mathsf{J}\}$, including the gravity ${}^{\{\mathsf{J}\}}\mathbf{a}_\mathsf{j}\,(\mathsf{t})$ is shown in 🟪. The gravity vector $\mathbf{g}$ is given in ■.

[2] The origin frame $\{\mathsf{O}\}$ depends on the specifications of a marker-based MoCap system.

$$ {}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\mathbf{a}_{\mathsf{j}}(\mathsf{t}) = {}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\mathbf{a}(\mathsf{t}) + {}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\boldsymbol{\omega}(\mathsf{t}) \times \left({}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\boldsymbol{\omega}(\mathsf{t}) \times {}^{\{\mathsf{O}\}}\mathbf{p}_{\mathsf{j}}(\mathsf{t})\right) + {}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\dot{\boldsymbol{\omega}}(\mathsf{t}) \times {}^{\{\mathsf{O}\}}\mathbf{p}_{\mathsf{j}}(\mathsf{t}), \quad (1) $$

${}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\mathbf{a}(\mathsf{t})$ is computed from the second derivative of a smooth spline approximation of degree three on a small time-interval from a sequence of joint-pose estimations. Similarly, the first and second derivative of a cubic spline interpolation from a sequence of joint rotations is used to compute ${}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\boldsymbol{\omega}(\mathsf{t})$ and ${}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\dot{\boldsymbol{\omega}}(\mathsf{t})$. This approach differs from the SOBDs using finite differences for human pose estimation in that it assumes that local temporal neighbourhoods are likely to be correlated. Specifically, a SOBD dataset is created from the derivatives of overlapping sequences of human-joint recordings. Gravity influences real IMU data. Thus, ${}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\mathbf{a}_{\mathsf{j}}(\mathsf{t})$ is summed with gravity in the direction of Z and is rotated to its corresponding joint frame resulting in

$$ {}^{\{\mathsf{J}\}}\mathbf{a}_{\mathsf{j}}(\mathsf{t}) = {}^{\{\mathsf{J}\}}_{\{\mathsf{O}\}}\mathbf{R}(\mathsf{t}) \cdot \left({}^{\{\mathsf{O}\}}_{\{\mathsf{J}\}}\mathbf{a}_{\mathsf{j}}(\mathsf{t}) + \mathbf{g}\right) \quad (2) $$

where, ${}^{\{\mathsf{J}\}}_{\{\mathsf{O}\}}\mathbf{R}(\mathsf{t}) = \mathbf{R}_{\mathsf{z}}(\mathsf{t})\,\mathbf{R}_{\mathsf{y}}(\mathsf{t})\,\mathbf{R}_{\mathsf{x}}(\mathsf{t})$ and $\mathbf{g} = (0, 0, -9.8m/s)^T$.

4 Neural Networks and Training Methodology

This work experiments with two learning strategies: supervised learning and self-supervised learning. For supervised learning, we explored three architectures: convolutional neural network (CNN), LSTM, and CNN-transformer (transformer with input embedding created with a convolutional layer). For self-supervised learning, we focused on CNN. Within the CNN architecture, further structural changes are considered depending on the channel density of the dataset. The CNN for high channel density datasets is the temporal CNN-IMU network from [20]. The CNN-IMU network assigns a branch of four convolutional layers to sensors attached to each human limb (legs, hands and torso). Further, the concatenated features are provided to a multi-layer perception (MLP) to provide activity classifications. In the case of low channel density datasets with a single sensor attached to a single limb, the same network is modified to consist of a block of four convolutional layers, followed by MLP. In contrast, dataset channel density is not considered for the LSTM and CNN-Transformer architectures. The LSTM has four hidden layers of 256 dimensions, followed by two MLP layers and a softmax activation layer. The CNN-Transformer is adapted from the work of [25]. The network consists of a convolutional backbone that generates latent sequence embedding provided to the transformer encoder. The latent sequence is further aggregated and provided to the classifier head. The cross-entropy loss function trains the classifiers for the supervised learning method. Adam's Optimiser was effective in the case of CNN-Transformer, while Root Mean Square Propagation (RMSProp) propagation was effective for the CNN and LSTM architectures. To establish baselines for the study, a hyperparameter study on batch size, learning rate, and epoch was conducted on each network for

Table 1. Datasets used and their features. Here, LARa$_{mc}$, LARa$_{mb}$, and LARa$_{mm}$ stands for LARa MoCap, LARa-mbientlab and LARa-motionminers, respectively. Sampl.Rate refers to the sampling rate chosen by the respective dataset creators. #Subj. refers to the number of subjects and #Act. refers to the number of activity classes in the dataset. Win. Size refers to the window size provided to the network.

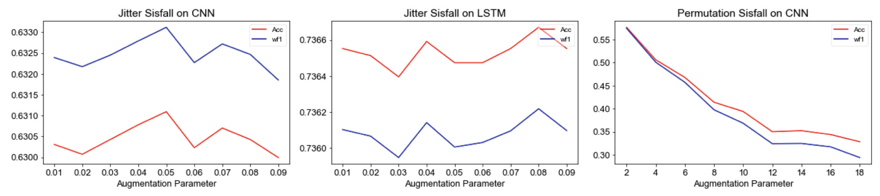
Dataset	#OBDs	#Sensors	Sampl. Rate (Hz)	#Subj.	#Act.	Win. Size	Overlap
MobiAct [24]	1	3	20	50	9	200	50
LARa$_{mc}$ [21]	-	-	200	16	7	200	25
LARa$_{mb}$ [21]	5	3	100	7	7	100	12
LARa$_{mm}$ [21]	3	3	100	7	7	200	25
Motionsense [17]	1	3	50	24	6	200	25
Sisfall [26]	1	3	200	38	15	200	50
MotionMiners [19]	3	3	100	5	4	100	12

the datasets in the experiment. The weights of the autoencoder are initialised using the orthonormal initialisation method. The RMSProp optimisation is used. Gaussian noise with mean $\mu = 0.0$ and standard deviation $\rho = 0.01$ is added to the sensor measurements to simulate sensor inaccuracies [20].

In SSL, earlier successes in applying contrastive learning in computer vision showed the importance of choosing the proper augmentations to create pairs of images depicting the same classes [2]. This work explores solutions for the same fundamental challenge in HAR. We chose the simple siamese representation learning method SimSiam [3], as it has been proven to be simple and efficient without the need for large batch sizes [2], tracking a running queue of previous features [8] or a momentum encoder [6]. To identify the necessary augmentations to successfully apply SSL in HAR, we perform self-supervised pre-training with a fixed setup for every augmentation and every dataset. We use a batch size of 512, a learning rate 0.05 with cosine annealing and train for 256 epochs using the Adam optimizer. Subsequently, linear probing is performed for every pre-trained model with the same configuration and fixed seed. During linear probing, we use a batch size of 50, a constant learning rate of 0.01 and train for 32 epochs. For each combination of dataset and augmentation, we keep the model with the best accuracy on the validation split of the corresponding dataset. Following the standards of linear probing, convolutional layers are not trained to isolate the impact of a given augmentation.

5 Experimental Results

As discussed previously, the effect of data augmentation methods varies based on the networks used for training and the datasets upon which the augmentation takes place [11]. Consequently, this work experiments varied techniques with seven different datasets to account for these factors. The datasets were explicitly selected to vary on the number of sensors,activities, sampling rate, and recording method. Table 1 shows the details of the selected datasets. It is to be noted that as part of the pre-processing of the training data, no changes to the sampling rate of the original datasets were performed.



(a) Jitter on Sisfall test set (b) Jitter on Sisfall test set (c) Permutation on Sisfall test set

Fig. 4. Augmentation parameter analysis on Sisfall dataset on pre-trained CNN and LSTM

5.1 Algorithmic Augmentation

Unlike image data, a major issue with augmenting time-series data is that the impact of augmentation is not human-perceivable. For instance, when adding noise to a segment of activity data recording, it is often unclear to what extent the vital information required for recognising the activity is retained and when the complete information is lost. Thus, in a preliminary experiment, we augmented the test data and ran an augmentation parameter variation testing phase on a network trained without augmentation. The performance of the test data indicated the extent the augmentation technique modified the test data features. Further, this augmentation parameter analysis presented interesting observations. For instance, the Sisfall dataset performs quite well with jitter augmentation in the test set, exhibiting its resistance towards noisy data. However, as Fig. 4 shows, increasing the permutation parameter decreased the accuracy of the test set.

Figure 5 presents the performance of the networks trained on each dataset with one specific augmentation applied randomly at 50% probability. In addition, a network trained without augmentation, referred to as Baseline, is presented for all networks and datasets. We present the results graphically to analyse the variation of performance of the networks on different datasets for each augmentation method. In particular, the performance variation that an augmentation technique brings compared to the baseline. Overall, one can see that with most of the augmentation techniques, a fall in performance compared to the baseline is apparent. In rare cases, the network performance improves for an augmentation technique beyond the baseline.

In general, in alliance with the work of [11], a trend on the impact of the augmentation techniques on the datasets is not present. The number of sensors, the activities included in the dataset, and the labelling affect the performance.

For instance, relatively similar performance can be seen between datasets with a single sensor compared to datasets with more than one sensor. Furthermore, CNN networks seem to be more accepting of augmented data. However, both the gain and drop in performance are minuscule. A variation can be seen in the Sisfall dataset where *time-warping* and *permutation* provide improvement while *slicing* and *vertical flip* further reduce performance. In general, *time-based warping* techniques increase performance. A similar benefit can be seen with the *random resampling* and *spectral denoising* techniques. *Vertical flipping* of the data as part of augmentation severely affects all the datasets except the MotionMiners. A similar but more pronounced impact can be seen on the CNN-Transformer.

Similar to Fig. 5, Fig. 6 presents the augmentation techniques' impact on half the dataset but with augmentations using the same parameters. Interestingly. more pronounced variations in the impact of the augmentation techniques can be found here. In the case of CNN networks, an interesting trend can be noticed. LARa$_{mb}$, LARa$_{mm}$ and MotionMiners, which have sensors placed in var-

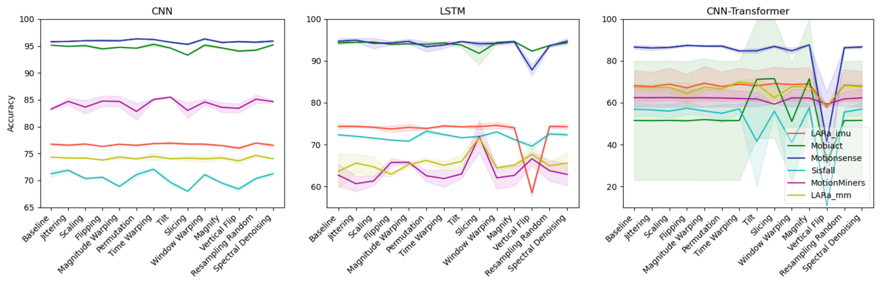


Fig. 5. Implication of augmentation on the respective dataset for CNN, LSTM and CNN-Transformer. 🟥 indicates LARa$_{mb}$, 🟩 indicates Mobiact, 🟦 indicates Motionsense, 🟦 indicates Sisfall, 🟪 indicates MotionMiners, and 🟨 indicates LARa$_{mm}$. (Color figure online)

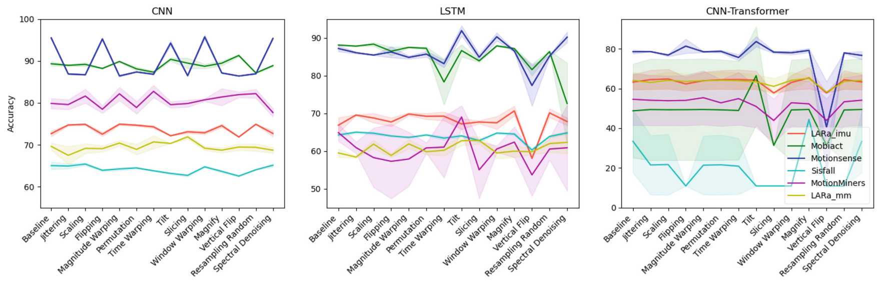


Fig. 6. Implication of augmentation on the respective half dataset for CNN, LSTM and CNN-Transformer. 🟥 indicates LARa$_{mb}$, 🟩 indicates Mobiact, 🟦 indicates Motionsense, 🟦 indicates Sisfall, 🟪 indicates MotionMiners, and 🟨 indicates LARa$_{mm}$. (Color figure online)

ied body locations, drop in performance with *spectral denoising*, while the other datasets with one sensor generally improve performance. Unlike the complete training Motionsense set, the half training set do not show visible improvement with augmentations but rather a steep drop in performance. Unfortunately, Sisfall datasets performance on CNN-Transformer falls rapidly with augmentations techniques except for *magnify* and *spectral denoising* techniques.

5.2 Generative Augmentations

To evaluate the Generative-based augmentation, we first experimented with the temporal autoencoder model to find the best architecture for the reconstruction task. Next, the experiments on creating and using the synthetic data generated by the temporal autoencoder are presented. We choose the LARa$_{mc}$ data for these experiments to facilitate visualisation. The weighted F1 (wF1) score and accuracy (Acc) are computed to measure the extracted features' quality. A semantic attribute vector is calculated for the data to compute the metrics. This vector is then used to classify the activity based on the nearest neighbour search and compute the metrics for the experiments. Thus, human movements can be transformed into discriminative feature representations that can be reconstructed[3].

Having found the hyperparameters of the temporal autoencoder that give high activity classification accuracy (details are given in the supplemental material), we generate synthetic data by introducing Gaussian noise to the feature representation. The LARa$_{mc}$ data is used as an example to facilitate visualisation. To create a balanced dataset, an attempt was made to match the *Handling-Center* activity samples that dominated the LARa dataset, as Fig. 1a in supplementary shows. Thus, copies of the other activity classes were created to match the sample size of *Handling-Center* activity class. These copies are given as the input to the modified temporal autoencoder to generate synthetic data. A total of 208768 samples of synthetic data were generated in which all classes occur equally. In other words, we created 156229 more samples to create a balanced dataset.

To evaluate the quality and impact of the temporal autoencoder-generated synthetic data, the baseline tCNN network is trained for solving HAR using semantic attribute vectors on the LARa$_{mc}$. Classification performance of $wF1[\%]$ raises from 73.62 to 75.22 when balancing the LARa$_{mc}$ using the generative model. Table 2 shows the classification performance on the datasets when training using 50% and 100% of the training set and augmented to obtain a balanced data generated by the generative model–column *Gen.*–, and comparing with no augmentation. The performance of the classification network was found to have mixed results for the datasets, being significantly higher for the *MobiAct* and *Motionsense* when using 100% of the training set, and LARa$_{mm}$ when having available 50% of the training set.

[3] Fig. 1b in supplementary shows that the autoencoder reconstructions are similar to the original input data.

5.3 Kinematic Augmentations

The total acceleration considering the linear and angular changes of the joint poses along time is computed to generate SOBD data. These SOBDs will be source domains for parameter-based transfer learning of HAR. The LARa$_{mc}$ is deployed for creating the subset LARa-Synthetic OBD (LARa-SOBD). The LARa-SOBD is sampled at $200Hz$, $100Hz$, $50Hz$, and $20Hz$ matching the sampling rate of the target datasets.

Table 2. Classification performance in terms of $wF1\%$ CNN-IMU applied on the inertial datasets, when balancing the datasets using generative models (Gen.), and the kinematic technique (Kin.) utilising the $LARa_{mc}$ as a data source. Values in **bold** are significant with respect to the No Augmentation case by means of a permutation test.

	$LARa_{mb}$			$LARa_{mm}$			*MobiAct*			*Motionsense*			*Sisfall*		
$wF1\%$	No Aug.	Gen.	Kin.	No Aug.	Gen.	Kin.	No Aug.	Gen.	Kin.	No Aug.	Gen.	Kin.	No Aug.	Gen.	Kin.
50%	74.40	71.64	**73.77**	42.11	**67.24**	**67.28**	89.62	88.22	87.99	76.57	23.56	**78.93**	58.68	48.64	59.17
100%	74.59	75.00	**76.32**	71.21	71.85	67.08	94.20	**94.55**	**95.25**	86.79	**96.01**	88.04	63.75	17.36	58.92
Acc%	No Aug.	Gen.	Kin.	No Aug.	Gen.	Kin.	No Aug.	Gen.	Kin.	No Aug.	Gen.	Kin.	No Aug.	Gen.	Kin.
50%	74.72	72.75	**74.03**	56.94	**67.33**	**68.76**	89.75	88.11	87.37	76.72	31.42	**79.04**	59.37	52.18	59.74
100%	75.55	75.33	**77.08**	71.89	72.01	67.62	94.38	**94.73**	**95.35**	86.80	**96.07**	88.016	63.98	24.76	59.24

Table 3. Classification performance in terms of $wF1$[%] and Acc[%] from the CNN-IMU applied on the *MotionMiners* dataset when applying the kinematic technique utilising the $LARa_{mocap}$ as data source. Values in **bold** are significant with respect to the No Augmentation case by means of a permutation test.

50%				100%			
No Aug.		Kinematic		No Aug.		Kinematic	
$wF1$[%]	Acc[%]	$wF1$[%]	Acc[%]	$wF1$[%]	Acc[%]	$wF1$[%]	Acc[%]
49.15 ± 7.09	59.29 ± 1.15	$\mathbf{80.45 \pm 0.78}$	$\mathbf{80.64 \pm 0.74}$	82.89 ± 0.58	83.80 ± 0.66	$\mathbf{86.11 \pm 0.21}$	$\mathbf{86.22 \pm 0.50}$

In general, the performance improves according to the proportion of the dataset. The learnt features from human poses and their derivatives can be deployed on inertial data. This suggests that filters learnt short temporal relations from the sequence inputs per channel independently of the rather related domain. Since time sequences of human poses and inertial measurements are physically related, the number of channels differs, and the source and target belong to the same problem, i.e., activity classes. The activities in the LARa$_{mb}$ dataset are performed in a warehouse environment, whereas the source datasets consider activities of daily living. As Table 2 shows, the performance significantly improves when considering the 50% and 100% of the LARa$_{mb}$, 50% of the LARa$_{mm}$ and *Motionsense*—not significantly for *Sisfall*; and 100% of the LARa$_{mm}$ and *MobiAct*.

In the case of the industrial dataset *MotionMiners* with few subjects and only three On-Body Devices (OBDs), the classification performance gets significantly boosted by the use of SOBDs from a $wF1\% = 49.15 \pm 7.09$ to 80.45 ± 0.78 for the 50% training dataset and from $wF1\% = 82.89 \pm 0.58$ to 86.22 ± 0.50 for the 100% of the training set. These improvements show the potential of this technique when data is limited, as Table 3 shows.

Table 4. Test accuracies after linear probing per dataset and augmentations used in self-supervised pre-training. During linear probing, only fully connected layers are trained, while convolutional layers are frozen. Values in **bold** or **bold gray** identify the best augmentation or second and third best augmentation per dataset, respectively.

Augmentation	$LARa_{mb}$		$MotionMiners$		$LARa_{mm}$		*Motionsense*		*MobiAct*		*Sisfall*	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Supervised Baseline	76.74	75.70	83.27	81.68	74.32	72.94	95.78	95.69	95.13	94.90	71.23	70.82
Flipping	57.56	44.28	59.33	44.18	59.18	49.12	91.50	91.48	86.36	84.60	52.71	50.10
Vertical Flip	59.63	46.84	59.15	44.10	57.18	41.60	68.53	59.28	84.04	81.35	14.90	6.12
Jittering	58.80	48.90	64.37	63.35	57.91	43.40	61.50	53.81	**89.03**	**87.82**	**59.74**	**58.88**
Magnify	57.57	43.52	59.51	44.65	57.73	43.80	70.54	63.84	88.17	86.60	**60.24**	**58.92**
Tilt	58.39	47.77	**68.23**	**67.68**	57.18	41.60	24.34	9.53	**92.19**	**91.54**	43.85	38.58
Scaling	**64.69**	**59.78**	62.39	58.10	57.23	41.76	89.40	89.27	**89.24**	**88.08**	51.51	48.25
Permutation	60.10	49.36	**81.31**	**79.33**	59.41	47.01	67.56	60.20	31.34	14.96	57.74	56.64
Resampling Random	**71.83**	**68.15**	**68.54**	**64.80**	**66.36**	**63.57**	87.41	86.55	64.51	52.30	31.67	22.94
Slicing	56.80	41.16	59.33	44.18	57.18	41.60	72.35	68.42	31.34	14.96	58.34	57.34
Spectral Denoising	58.68	47.87	59.32	44.39	58.00	43.72	62.49	54.94	31.34	14.96	49.62	48.48
Magnitude Warping	56.80	41.16	59.34	44.21	**61.38**	**52.52**	**91.96**	**91.98**	31.34	14.96	**58.94**	**57.40**
Time Warping	**65.30**	**60.39**	59.33	44.18	**60.85**	**53.25**	**92.32**	**92.23**	87.93	86.51	10.89	2.14
Window Warping	61.47	53.37	59.33	44.18	57.18	41.60	**92.92**	**92.87**	31.34	14.96	50.97	48.86

5.4 Self-supervised Learning

Following the method explained in Sect. 4, we report test accuracies after linear probing in Table 4. For *Motionsense*, *MobiAct* and *Sisfall*, we used tCNN, while for $LARa_{mb}$,*MotionMiners* and $LARa_{mm}$, we used tCNN-IMU with five and three branches respectively.

For three of the six tested datasets, SSL pre-training achieves comparable results as the supervised baseline, although not exceeding it. Unlike in research from the vision domain [2,13] the results do not show a general recommendation of augmentations across datasets. For example, on *Sisfall* time warping achieve the worst performance, while at the same time, applying time warping achieves second or third best performances on most other datasets. However, less severe, similar behaviour is noticeable when applying augmentations in supervised training (compare Fig. 5). Contrary to supervised learning, choosing the wrong augmentation in SSL leads to a catastrophic drop in possible performance. Additionally, we investigate the quality of learned embedding spaces under the constraint of fewer available annotated data during linear probing by halving the training data. We report these results in the supplementary material. While the results show that with fewer data, the performance drops slightly, the per dataset trends observed in Table 4 are still observable here.

6 Conclusion

This work analyses various augmentation techniques and their impact on three networks and two learning strategies on seven datasets used in varied contexts. We find that neural network models, specifically CNNs, are more receptive to algorithmic augmentation techniques. In contrast, LSTM and CNN-Transformer performance show more variability, thus answering *RQ1* on the response of neural networks to augmentations. In fact, most of the algorithmic augmentation techniques resulted in severe performance drops compared to gains. Thus, unlike in the domain of images, the use of varied augmentation techniques can be deemed not useful in augmenting and increasing the variability within the training data, thus resulting in a non-concrete answer to *RQ2* on the possibility of providing a method to choose the augmentation technique for a specific dataset. However, the results indicate that CNN-Transformers prefer the actual quantity of data to better performance than being provided augmented data. In comparison, a pronounced and stable gain in performance is observed when using generative and kinematic augmentation techniques, specifically when one attempts to use the generated data to fine-tune the networks. As a result, addressing the data imbalance and model generalisation through generative and kinematic models is advisable, answering *RQ3* on choosing between generative and algorithmic augmentation techniques. Finally, the research indicates that similar to supervised learning, one cannot obtain a trend in the performance of augmentation techniques for SSL. However, certain augmentation techniques can achieve performance close to supervised learning, thus answering *RQ4* on the identification of augmentation techniques beneficial for SSL.

As part of future work, further experimentation and evaluation of augmentation techniques with SSL is required. Furthermore, inspired by the availability of accessible and efficient augmentation libraries in computer vision, developing a standardized augmentation library for multi-channel time-series data is desirable, ideally for HAR. In an interesting research direction, augmentation techniques could be fused, e.g., using data generation to create pairs for SSL methods by synthesizing a sample conditioned on original data. Finally, generative networks can be conditioned on deep representations of subjects' inertial data; synthetic data augmentations by interpolating among subjects could be a feasible way of generating additional plausible data.

Acknowledgment. The work in this publication was supported by the German Federal Ministry of Education and Research (BMBF) in the context of the project "LAMARR Institute for Machine Learning and Artificial Intelligence" (LAMARR24B).

References

1. Awasthi, S., Rueda, F.M., Fink, G.A.: Video-based pose-estimation data as source for transfer learning in human activity recognition. In: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 4514–4521. IEEE, Montreal, QC, Canada (2022)
2. Chen, T., Kornblith, S., Swersky, K., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: Proceedings of the 37th International Conference on Machine Learning, pp. 1597–1607 (2020)
3. Chen, X., He, K.: Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15750–15758 (2021)
4. Craig, J.J.: Introduction to Robotics: Mechanics And Control, vol. 3. Pearson Education (2005)
5. Gao, Z., Li, L., Xu, T.: Data augmentation for time-series classification: an extensive empirical study and comprehensive survey (2023)
6. Grill, J.B., et al.: Bootstrap your own latent-a new approach to self-supervised learning. Adv. Neural. Inf. Process. Syst. **33**, 21271–21284 (2020)
7. Hasan, M.A., Li, F., Piet, A., Gouverneur, P., Irshad, M.T., Grzegorzek, M.: Exploring the benefits of time series data augmentation for wearable human activity recognition. In: Proceedings of the 8th International Workshop on Sensor-Based Activity Recognition and Artificial Intelligence, pp. 1–7. ACM, Lübeck Germany (2023)
8. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9729–9738 (2020)
9. Hoelzemann, A., Sorathiya, N., Van Laerhoven, K.: Data augmentation strategies for human activity data using generative adversarial neural networks. In: 2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), pp. 8–13. IEEE, Kassel, Germany (2021)
10. Huang, Y., Kaufmann, M., Aksan, E., Black, M.J., Hilliges, O., Pons-Moll, G.: Deep inertial poser: learning to reconstruct human pose from sparse inertial measurements in real time. ACM Trans. Graph. **37**(6), 1–15 (2018)
11. Iwana, B.K., Uchida, S.: An empirical survey of data augmentation for time series classification with neural networks. PLOS ONE **16**(7) (2021)
12. Iwana, B.K., Uchida, S.: Time series data augmentation for neural networks by time warping with a discriminative teacher. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 3558–3565. IEEE, Milan, Italy (2021)
13. Koßmann, D., Matei, A., Wilhelm, T., Fink, G.A.: Image augmentations in planetary science: implications in self-supervised learning and weakly-supervised segmentation on Mars. In: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 2800–2806 (2022)
14. Leng, Z., Kwon, H., Ploetz, T.: Generating virtual on-body accelerometer data from virtual textual descriptions for human activity recognition. In: Proceedings of the 2023 International Symposium on Wearable Computers, pp. 39–43. ACM, Cancun, Quintana Roo Mexico (2023)
15. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431–3440 (2015)

16. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: SMPL: a skinned multi-person linear model. ACM Trans. Graph. **34**(6), 1–16 (2015)
17. Malekzadeh, M., Clegg, R.G., Cavallaro, A., Haddadi, H.: Mobile sensor data anonymization. In: Proceedings of the International Conference on Internet of Things Design and Implementation, pp. 49–58. ACM, Montreal Quebec Canada (2019)
18. Rueda, F.M.: Transfer learning for multi-channel time-series human activity recognition (2023)
19. Rueda, F.M.: MotionMiners HAR Dataset (Version V1) (2024)
20. Moya Rueda, F., Grzeszick, R., Fink, G., Feldhorst, S., Ten Hompel, M.: Convolutional neural networks for human activity recognition using body-worn sensors. Informatics **5**(2), 26 (2018)
21. Niemann, F., et al.: LARa: creating a dataset for human activity recognition in logistics using semantic attributes. Sensors **20**(15), 4083 (2020)
22. Reining, C., Rueda, F.M., Ten Hompel, M., Fink, G.A.: Towards a framework for semi-automated annotation of human order picking activities using motion capturing. In: 2018 Federated Conference on Computer Science and Information Systems (FedCSIS), pp. 817–821. IEEE, Poznan, Poland (2018)
23. Reining, C., Nair, N.R., Niemann, F., Rueda, F.M., Fink, G.A.: A tutorial on dataset creation for sensor-based human activity recognition. In: 2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), pp. 453–459. IEEE, Atlanta, GA, USA (2023)
24. Röcker, C., Ziefle, M., O'Donoghue, J., Maciaszek, L., Molloy, W.: Institute for systems and technologies of information. In: C.a.C. (eds.): ICT4AWE 2016: proceedings of the International Conference on Information and Communication Technologies for Ageing Well and e-Health: Rome, Italy, April 21-22, 2016. SCITEPRESS - Science and Technology Publications, Lda, Setúbal (2016)
25. Shavit, Y., Klein, I.: Boosting inertial-based human activity recognition with transformers. IEEE Access **9**, 53540–53547 (2021)
26. Sucerquia, A., López, J., Vargas-Bonilla, J.: SisFall: a fall and movement dataset. Sensors **17**(12), 198 (2017)
27. Thukral, M., Haresamudram, H., Ploetz, T.: Cross-domain har: few shot transfer learning for human activity recognition (2023)
28. Um, T.T., et al.: Data augmentation of wearable sensor data for Parkinson's disease monitoring using convolutional neural networks. In: Proceedings of the 19th ACM International Conference on Multimodal Interaction, pp. 216–220. ACM, Glasgow UK (2017)
29. Varamin, A.A., Abbasnejad, E., Shi, Q., Ranasinghe, D., Rezatofighi, H.: Deep auto-set: a deep auto-encoder-set network for activity recognition using wearables (2018)
30. Yuan, H., Chan, S., Creagh, A.P., Tong, C., Clifton, D.A., Doherty, A.: Self-supervised learning for human activity recognition using 700,000 person-days of wearable data (2023)

PostAugment: Adversarial Data Augmentation with Hard Sample Suppression by Incorrect Class Likelihood

Azusa Sawada(✉), Takashi Shibata, Keiko Yokoyama, Shoji Yachida, and Toshinori Hosoi

NEC Corporation, Nakahara-ku Shimonumabe, 1753 Kawasaki, Japan
swd02@nec.com, t.shibata@ieee.org

Abstract. We propose an adversarial automatic data augmentation with hard sample suppression by incorrect class likelihood for time series data. Automatic data augmentation (ADA) is a practical framework when the number of training data is small. In particular, an ADA based on adversarial loss, such as TeachAugment, can improve classification accuracy in optimizing complex transformation parameters with class and instance dependencies. In this paper, we aim to extend adversarial ADA for image recognition to time series data. The critical challenge is that data augmentation for time series data is more difficult than that for images in designing a transformation function that does not cross the discriminant class boundary. When existing ADA frameworks are naively applied to time series data, the transformation function often generates augmented data similar to the incorrect class. As a result, this overly hard-to-identify data degrades the accuracy of a target classifier. In general, this hard-to-identify data near the class boundary has a high likelihood both in the correct and incorrect classes. The proposed method can reduce this hard-to-identify data by introducing a novel fidelity loss that suppresses only the likelihood of the incorrect class. Comprehensive experiments on two datasets demonstrate the effectiveness of the proposed method. The project code is included in the supplemental material.

Keywords: Data augmentation · Time series · Classification

1 Introduction

Data augmentation is one of the most practical approaches in various tasks such as image, time-series signals, and audio when training data is small because deep learning is data-hungry. The training data can be virtually increased by applying random transformations during training. Various transformations are

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78192-6_6.

used for data augmentation, including transformations that preserve semantics and transformations that improve the robustness of the model [24,27].

Automatic data augmentation (ADA), which automatically searches for an appropriate set of transformations (i.e., policy), has been widely studied, mainly in general image recognition. This is because it is difficult to properly select and set the strength of data augmentation among various transformations. A practical approach to efficient search among the enormous policy space is to perform an adversarial strategy [20,25]. The adversarial strategy, which searches for a policy that maximizes the task loss of the target model, allows for policy search that can empirically improve model generalization while reducing the computational load. The point of the adversarial strategy is that it efficiently trains discriminative boundaries by generating data that is difficult to classify for a target model on training. In particular, TeachAugment [25], which leverages a teacher model to moderate the adversarial updates of augmentations, has improved accuracy in optimizing complex transformation parameters with class and instance dependencies.

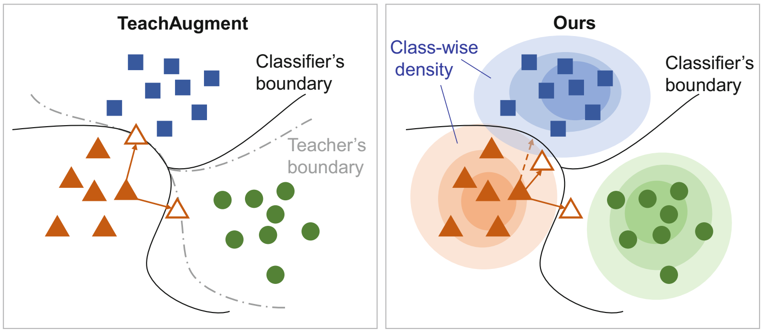


Fig. 1. Concept of augmentation criterions in TeachAugment and our method.

In this paper, we aim to extend those sophisticated adversarial data augmentations for image domain to time series one. Existing adversarial data augmentations, e.g., TeachAugment, are designed for general image classification tasks like CIFAR10/100 [13,14] and ImageNet [5]. A vital assumption in ADA for the images is the ease of designing transformation functions, i.e., it is intuitive and easy for humans to design “transformation functions that do not change the belonging class of each sample." For example, based on visual intuition, we adopt chromatic and geometric transformations as candidates that do not change the belonging class of each sample.

On the other hand, for time-series data, it is difficult to design a suitable transformation function that does not change the belonging class of each sample because this implicit assumption based on human intuition is unavailable. In the time series domain, ambiguous samples located between classes are often generated due to applying adversarial augmentation frameworks that proactively generate samples located around discriminative boundaries, as shown in Fig. 1 (left). The frequent generation of ambiguous samples that are difficult to identify as their class is detrimental to performance. A method that is applicable to time-series data while taking advantage of ADA's strengths for the image domain is strongly needed.

This paper proposes an adversarial automatic data augmentation using hard sample suppression for time series data. The key is suppressing the absolute likelihood value for incorrect classes to avoid hard-to-identify samples. The generation of hard-to-identify data can be reduced by introducing a novel fidelity loss that can suppress only the incorrect class likelihoods because the hard-to-identify samples near the discrimination boundary are the part where the correct and incorrect class likelihoods coexist, as shown in Fig. 1 (right). Comprehensive experiments on two datasets demonstrate the effectiveness of the proposed method.

The contributions of this paper are as follows:

– We propose an adversarial automatic data augmentation using hard sample suppression for time series data.
– We introduce a loss function (Fidelity loss) that suppresses transformations that would change classes to other ones.
– We demonstrate the effectiveness of the proposed method for time series dataset where existing adversarial data augmentation fails to suppress transformations that adversely affect performance.

2 Related Work

The methods for finding effective transformations for data augmentation have been widely explored, mainly in image recognition tasks. One of the earliest methods, AutoAugment [4], searches augmentation policy to get smaller validation loss through reinforcement learning. Although AutoAugment uses a smaller model on a subset of the training set in the search phase, it requires high computational cost in retraining for many times. Subsequent works proposed effective variations: population-based training of augmentation schedules [9], alternation of criterion by density matching [17], and utilizing a shared pre-trained model to warmstart AutoAugment [12]. Gradient-based bi-level optimization further improved the efficiency of the augmentation search [7,8,16,26]. Although most previous works rely on default augmentations as a fixed part of augmentation, DeepAA [29] showed it can search multi-layer transformation from scratch by layer-wise search using gradient matching criterion. SLACK [19] further enabled us to learn the joint probability of multiple transformations directly by the stabilization of gradient updates using a cold-start strategy and a Kullback-Leibler regularization.

As shown in RandAugment [3], optimal augmentation parameters depend on the amount of training data. Adversarial AutoAugment [20] proposed to tune augmentations in online manner during training models on whole train set by updating augmentation to increase a train loss. Similarly, PointAugment [15] updates transformation to increase classification loss to some limit value proportional to loss on original data. Recently, TeachAugment [25] proposed a min-max formulation of joint training of the model and augmentations using a teacher model. It minimizes adversarial loss and classification loss of the teacher model, which can be another model trained on the same data or the exponential moving average of a model on training. This method can suppress indiscriminative data due to strong augmentation without prior knowledge, and also successfully trains class-dependent and instance-dependent augmentations.

While many works on automatic data augmentation are designed for image recognition tasks, several works explore the optimization of data augmentations on time series. There are many transformation candidates for the general time series, as summarized in the survey paper [27]. Fons et al. [6] calculates losses for original and augmented samples by all transformation candidates, and reweight them with trainable parameters or trim ones of top-K and bottom-K values. Rommel et al. [22] showed the importance of class-dependence of augmentations, but their method fails to improve final accuracy on class-dependent settings by gradient-based bi-level optimization.

We focus on adversarial augmentation because it can search data augmentation parameters with all training data, and have been shown to successfully find class-dependent or instance-dependent augmentation.

3 Method

The goal of the automated data augmentation is to find better augmentation to get a target model with high accuracy on unseen test data when trained on train data augmented by the augmentation. Let x and y be input data and label sampled from train data D_{train}, respectively. Train data D_{train} contains N_k samples in class $k \in 1, ..., K$. We denote a learnable augmentation parameterized ϕ as a_ϕ.

In the following, we first describe the overview of the proposed method. Then, the module and loss functions in our method are described in Sects. 3.2, 3.3, and 3.4. Finally, we explain the learnable data augmentation for time-series data in Sect. 3.5.

3.1 Overview

The proposed method "PostAugment" is based on an adversarial data augmentation. The adversarial data augmentation approach does not attempt to maximize performance on validation data created by separating from train data; instead, it maximizes the training loss for a target model on train data while the model is training using all the train data.

TeachAugment [25] updates the augmentation parameters to decrease a loss for the "teacher model" in addition to the adversarial loss for the target model because naive maximization of the training loss may lead to indiscriminative data generation. When augmentations with the updated parameters can change the class labels into different ones, TeachAugment tends to generate augmented samples around inter-class boundaries to balance the teacher and the adversarial losses. As a result, the target models' performance is degraded due to training on indiscriminative samples generated by the augmentations.

In contrast, our method introduces a novel fidelity loss to decrease the likelihood of augmented data being incorrect class using the per-class density estimation on feature space instead of the teacher's prediction based on the teacher model. The overview of our method PostAugment is shown in Fig. 2. Our method iteratively updates a target model with a density estimator and an augmentation model. The keys are the above-mentioned fidelity loss and the isolation of the density estimator training from the augmentation to avoid fitting label-confusing transformation candidates.

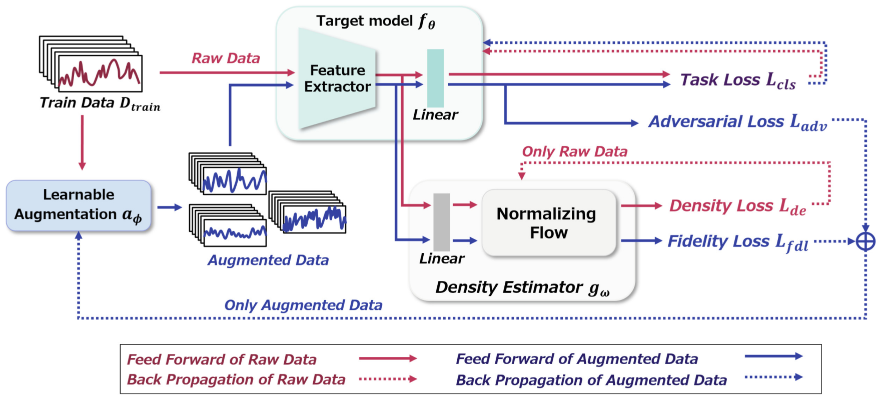


Fig. 2. Overview of the proposed method. Our method trains the density estimator in addition to the target model. Augmentation parameters are updated by backpropagation from adversarial loss and fidelity loss, alternately with the target model and the density estimator.

The detailed training procedure in our method is shown in Appendix A. We employ the following techniques as in [25]: sliced Wasserstein distance regularization for (with coefficient 10.0) and experience replay of the augmentation function a_ϕ.

3.2 Density Estimator for Fidelity Loss

We introduce the per-class density estimator for calculating the fidelity loss based on epistemic uncertainty modeling in the form of Dirichlet distribution. This density estimator module enables us to check if augmented data is similar to data in different classes based on observed train data.

The proposed per-class density estimation module is inspired by PostNets [2], proposed originally for modeling the epistemic uncertainty for out-of-distribution

(OOD) data detection without training on OOD data. The proposed estimation module outputs pseudo counts of each class on a feature space, which can be regarded as parameters of Dirichlet distribution over class probability. The pseudo counts for class k is a density q_k integrated to N_k using normalization flow layers [21]. We utilize this q_k as the likelihood of class k. In our experiments, the architecture of this module is fixed to an eight-layer radial flow.

The proposed per-class density estimation module is attached to a given target model as an additional branch in parallel to the last linear layer to keep the model's test-time architecture unchanged. The density estimation branch has a projection layer to reduce dimension (e.g., to 8 in our implementation) because the feature dimension is often too high for density estimation.

3.3 Loss Functions for Model with Density Estimator

The task loss L_{cls} to update the target model is a cross-entropy loss in this paper. Any classification losses can alternate this choice. We calculate L_{cls} both on raw and augmented samples to ease the density estimation of raw samples.

The density loss to train the density estimation module is below:

$$L_{de}(x,y) = \Psi(1+q_y(x)) - \Psi(\sum_{k=1}^{K}(1+q_k(x))) + \rho H(\mathbf{q(x)}), \tag{1}$$

where $\Psi(\cdot)$ is digamma function and H is entropy regularization term to promote smoothness with coefficient ρ (set to 1.e-5) [2]. The first two terms are derived from an expectation value of cross entropy over Dirichlet distribution. The gradients from this loss are stopped before the projection layer in order not to affect updates of the target model. The feature extractor in the target model is updated independently of the density loss; therefore, this branch may fail to keep fit on train data density. In our experiments on trajectory data in Sect. 4.1, the accuracy for this branch is comparable to that of the target model.

3.4 Loss Functions for Augmentation

The objective of augmentation models is minimizing the sum of two components: adversarial loss L_{adv} and fidelity loss L_{fdl}.

$$L_\phi = L_{adv} + \alpha L_{fdl}, \tag{2}$$

where α is a hyperparameter (set to 10.0 if not mentioned).

The adversarial loss L_{adv} is makes augmentations difficult for the target model, promoting efficient training and better generalization. In the proposed method, we employ the non-saturating loss adopted as in [25]:

$$L_{adv}(x',y) = -\sum_{k=1}^{K} \tilde{y}_k \log\left(1 - f_k(a_\phi(x',y))\right), \tag{3}$$

where $\tilde{y}$ is a soft target, original target label y after label smoothing (with a smoothing parameter 0.1).

The fidelity loss L_{fdl} in Eq. (2) is introduced to suppress transformations that change data difficult to classify. It is difficult to judge the true labels of transformed samples; therefore, we only penalize transformations that make samples similar to raw samples in different classes. We utilize the density estimation results as a likelihood $q_k(x)$ that a transformed sample is in each class k.

$$L_{fdl}(x', y) = \sum_{k \neq y} \log\left(1 + q_k(x')\right). \tag{4}$$

Here, the offset 1 is inserted to prevent gradients of this loss from divergence at small q_k. This loss sufficiently contributes to the updates of augmentation parameters for a minority class in an imbalanced dataset because confusion to majority classes is penalized heavily by q_k with normalization coefficients proportional to population N_k.

3.5 Stochastic Learnable Data Augmentation for Time-Series Data

The learnable augmentation a_ϕ in this paper is designed as sequential transformations of all candidates. The transformations are parameterized through probabilities and magnitudes for candidates. We consider two settings: instance-aware and class-aware.[1]

Instance-Aware. The instance-aware setting is a one-dimensional version of augmentations in TeachAug [25], originally implemented for two-dimensional image data. It consists of color transformation and geometric transformation. The one-dimensional color transformation is changed to amplitude scaling with different scaling coefficients and offsets by input dimensions. The one-dimensional geometric transformation is scaling in the temporal axis. The class-dependent probabilities to apply each transformation are approximated by the differentiable form using Gumbel-Softmax [11] and applied to corresponding magnitudes. The magnitude parameters are sampled as the normalized outputs of multi-layer perceptrons that receive a random noise vector concatenated with a class label, and only for color augmentation, that also receive data for local instance dependency.

Class-Aware. The class-aware setting is a simpler modeling, where the magnitude parameters are sampled from a common distribution for all data in the same classes. We sample a random variable from a Gaussian, normalize it by a sigmoid function, and multiply a learnable magnitude parameter to it. We assigned independent magnitude parameters for the geometric transformation to scaling up and down limits. In this setting, we add jittering, which adds random noise, where the magnitude is the maximum noise amplitude.

[1] We focus on the criterion for augmentation optimization in this paper. We leave improving parameterized candidates for time series augmentation for future work.

Implementation Details. It is important to care about the initialization of augmentation parameters. In the instance-aware setting, all parameters are initialized to zero, which means zero magnitudes and 0.5 probabilities. It enables augmentation updates and training from raw data. In the class-aware setting, the initialization to zero means the magnitudes are set to half the maximum values. The non-zero magnitude initialization must be considered when applying discrete augmentation, such as spatial flip (with gradient approximations).

4 Experiments

We conducted two sets of experiments. The first set is the experiments on our private dataset for trajectory classification. The second set is the experiments on UEA Multivariate Time Series Classification Archive, public time series dataset.

4.1 Trajectory Dataset

Dataset. Trajectory data are tracks of small particles in transparent liquid containers. The classes are defined as "`class0`" for foreign particles, "`class1`" for bubbles, and "`class2`" for scratches and detection noises. Each trajectory is generated by object tracking of videos and consists of eight components: time, horizontal coordinate, vertical coordinate, size, size in log-scale, mean brightness, variance of brightness, and two-dimensional positional encoding (all components are calculated for each detected blob and normalized into $[0, 1]$). Positional encoding here is designed for this dataset to tell if current positions are in a container and under a liquid surface in the respective dimension. All videos capture the same movement of the containers, including two static sections at different poses after rotations to induce the movement of particles in liquid. The trajectory in these static sections "section I" and "section II" are used as different datasets. Their input length is in $[26, 1000]$ and $[20, 340]$, respectively. The test sets (two splits, A and B, for each section) are separated not to include the data extracted from common videos in the training sets. The resulting numbers of samples and example plots are shown in Appx. B.

Experimental Setup. A model architecture used in this experiment is Adaptive Multi-Scale Convolutional Neural Network [23] proposed to classify fragmented time series such as trajectory data. All models are trained by SGD with momentum 0.9 and batch size 32 using cross-entropy loss for 80 epochs. The learning rate is scheduled to warm up linearly to 0.01 in the first 10 epochs and multiplied by 0.1 at 60. We report miss rates (MRs) on the test set at the best and the last epoch as the average of 5 runs with different random seeds.

We compare our method with three baselines: i) *None* trains a target model only with a default augmentation random crop. ii) *Fixed Random* uses all trainable augmentations at random within max magnitudes after random crop. iii) TeachAugment [25] is the existing method that updates all trainable augmentations to reduce the sum of the adversarial loss in Eq. (3) for the target model and the target loss for a teacher model generated by exponential moving average of the target model with decay rate 0.999. In the instance-aware setting, the optimizer for augmentation parameters is AdamW [18] as in TeachAugment experiments. In the class-aware setting, it is changed to SGD with momentum 0.9 because the parameters saturated early in the training stage with only small updates with AdamW. The max range of magnitudes for color, geometric, and jittering transformation are set to $[0.5, 1.5]$, $[0.5, 1.5]$, and $[0.0, 0.5]$, respectively.

Main Results. The results are shown in Table 1. *Fixed Random* increased miss rates compared with *None* in the trajectory dataset. TeachAugment shows better results compared to *Fixed Random* in most cases, and our method achieved much lower miss rates. On section II dataset, there are the case our method cannot reach the performance of None. This may be due to the difficulty to update transformation probabilities to zero because it needs large value.

Table 1. BestMR and LastMR results on our trajectory data. Data augmentation with class parameters is evaluated. Augmentation with class-wise parameters on trajectory data. The best/second results are shown in **bold**/*italic*, respectively.

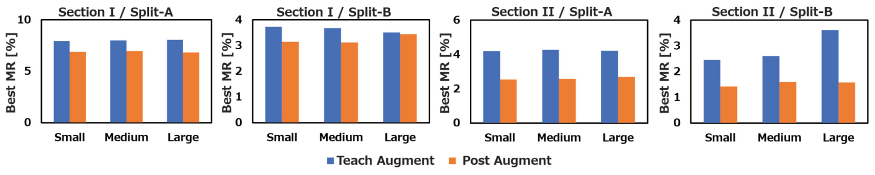
(a) BestMR Results

	Section I		**Section II**		
Method	Split-A	Split-B	Split-A	Split-B	Average
None	*7.120* (.172)	*3.353* (.142)	**2.489** (.138)	**1.410** (.059)	*3.617*
Fixed Random	8.333 (.286)	3.792 (.360)	4.220 (.152)	3.018 (.360)	4.863
TeachAugment	7.994 (.277)	3.676 (.331)	4.278 (.250)	2.604 (.314)	4.660
PostAugment (Ours)	**6.877** (.193)	**3.145** (.151)	*2.546* (.218)	*1.426* (.198)	**3.520**

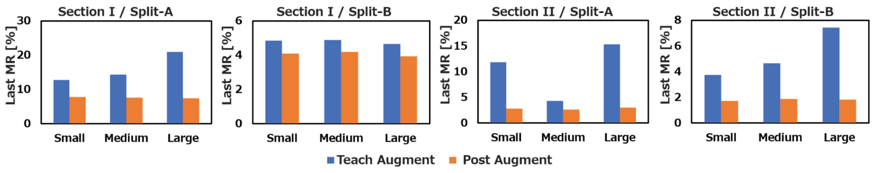
(b) LastMR Results

	Section I		**Section II**		
Method	Split-A	Split-B	Split-A	Split-B	Average
None	*7.977* (.168)	*4.185* (.412)	*2.904* (.293)	**1.592** (.069)	*4.195*
Fixed Random	13.40 (3.41)	4.393 (.661)	4.220 (.152)	4.444 (.140)	6.699
TeachAugment	14.30 (4.53)	4.879 (1.04)	4.278 (.250)	4.643 (1.84)	7.124
PostAugment (Ours)	**7.783** (.201)	**4.092** (.422)	**2.804** (.297)	*1.725* (.237)	**4.129**

Next, we evaluate the initialization dependency by changing the initialization conditions of augmentation parameters. The original setting (*medium init.*) sets all parameters to 0, which means probabilities and magnitudes are set to 0.5 and half-maximum values. They are changed to -1 in *low init.* and 1 in *high init.* setting. We insert three epochs with no augmentation for safe training in *high init.* cases because the class-dependence of augmentations may leak the class label to the target model due to training on mainly augmented data.



(a) BestMR



(b) LastMR

Fig. 3. Initialization dependency of the proposed method and TeachAugment by changing initialization of augmentation parameters. The proposed method outperforms TeachAugment in all initial values.

The resulting miss rates are shown in Fig. 3. We also compare the tendency of parameter updates in Fig. 4. Both methods change parameters little for `class1` with the smallest number of data due to less frequent updates. In our method, the probabilities are updated to lower values for `class0` than `class2` even if it needs larger updates for `class0` with a smaller number of data than `class2`. This result indicates that the loss in our method can suppress transformations well, even for class-imbalanced datasets.

We also compare TeachAugment and our method on the instance-wise setting (Table 2). The results on TeachAugment are almost better than the results on the class-aware setting in Table 1. This result may be caused by zero magnitude initialization in instance-wise setting, which is important for TeachAugment. Although Our method shows a little worse results compared to class-aware setting, it is still better than TeachAugment.

Table 2. Augmentation with instance-wise parameters on trajectory data. The best results are shown in **bold**.

Datasets	Split	BestMR		LastMR	
		TeachA	PostA (ours)	TeachA	PostA (ours)
Section I	A	7.184(.184)	**7.087**(.306)	7.767(.334)	**7.524**(.313)
Section I	B	3.815(.231)	**3.468**(.283)	5.064(.719)	**4.671**(.397)
Section II	A	3.619(.200)	**2.804**(.155)	4.692(.706)	**3.648**(.622)
Section II	B	1.907(.194)	**1.443**(.172)	2.206(.139)	**1.675**(.206)

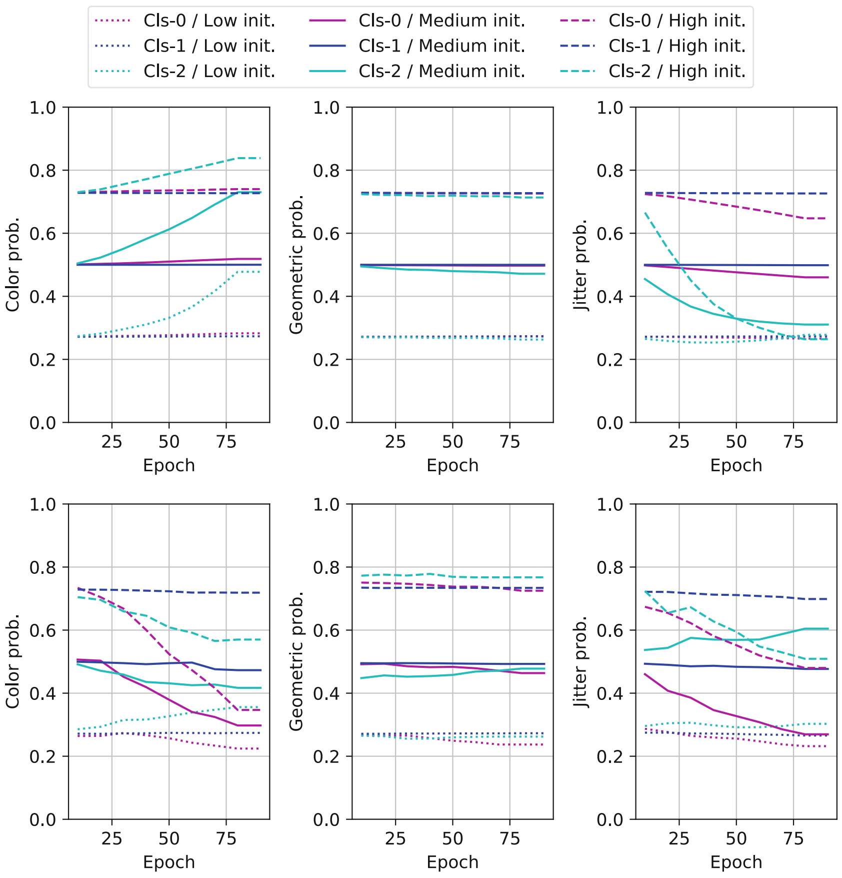


Fig. 4. Parameter updates in TeachAugment(top) and PostAugment(bottom) on split A of section I. Probabilities averaged over 5 runs for different initial values. Each column is corresponding to the respective probability of color, geometric and jittering transformation described in Sect. 3.5.

Ablation Study and Sensitivity Analysis. Table 3 shows the effect of raw data training, training target models both on augmented and original data simultaneously, on our method. There is clear increase of miss rates without raw data training. Our fidelity loss is calculated by density estimators of the raw data features by itself, and density fitting can be harder when the input features extracted by models trained only on augmented data.

The main hyperparameter in our method is α, which determines the relative weight of the fidelity loss to adversarial loss. Figure 5 compares resulting miss rates over different values of α. We also change the weight of the teacher loss in TeachAugment (the original version is corresponding to $\alpha = 1$). The results

Table 3. Ablation study on section I.

Method	Split	BestMR	LastMR
PostAugment	A	6.942(.184)	7.589(.359)
w/o raw data training		7.751(.522)	12.38(2.94)
PostAugment	B	3.121(.216)	4.185(.310)
w/o raw data training		3.861(.240)	4.647(.207)

show the best miss rates are found in $\alpha = 15$ on the average. There are large margins between our PostAugment and TeachAugment even for large α.

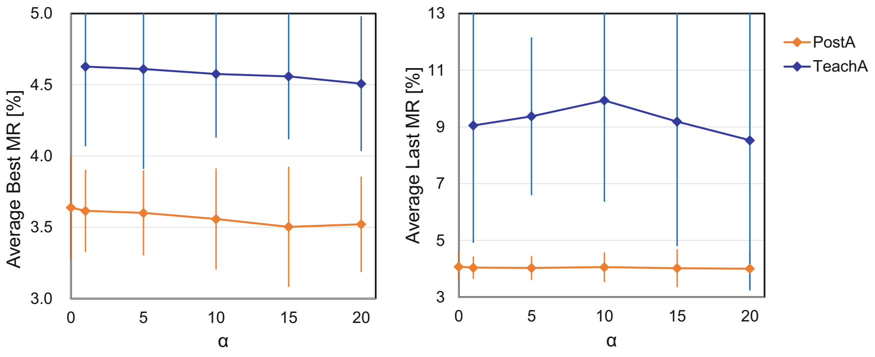


Fig. 5. Sensitivity to coefficient α for the fidelity loss or the teacher loss.

4.2 UEA Multivariate Time Series Classification Archive

Dataset. The UEA multivariate time series archive [1] contains 30 classification datasets with various characteristics, as summarized in Appendix C.

Experimental Setup. We evaluate each augmentation method using InceptionTime [10] without ensembling. The training settings are mostly chosen following the existing work that applied augmentations to the same datasets [28]: the optimizer is AdamW [18] with weight decay rate 10^{-5}, the batch size is 64, the maximum epoch is $1,500$, the base learning rate is initially 0.001 and scheduled by multiplying 0.5 when training loss has not decreased for 50 consecutive epochs down to the minimum value 0.0001. We stopped training if the minimum validation loss was not updated for 100 epochs. All experiments are repeated five times with different splits of the datasets, including the original train test split of the dataset. The rest four splits were generated by random resampling

Table 4. Accuracy comparison on three groups of UEA multivariate time series archive devided by train set sizes. Each column show accuracy on None and accuracy improvement from None on the rest. The best results are shown in bold.

Dataset (#Train<100)	None	Random	RandA	SLACK	TeachA	PostA
StandWalkJump	36.00	-9.33	-5.33	-2.67	+6.67	+**8.00**
AtrialFibrillation	21.33	+10.67	+5.33	+12.00	+12.00	+**14.67**
ERing	86.18	-0.16	-1.97	-69.51	+1.54	+**1.78**
BasicMotions	**100.0**	-0.50	-6.00	-50.00	+**0.00**	-0.50
DuckDuckGeese	63.20	+**4.00**	-0.80	-40.80	+3.20	+3.20
Best counts	1	1	0	0	1	3

Dataset (100<#Train<1000)	None	Random	RandA	SLACK	TeachA	PostA
Cricket	88.59	+3.59	-48.44	+5.02	+**6.88**	+2.03
UWaveGestureLibrary	79.63	+**9.25**	-4.88	+3.62	+5.38	+0.13
Epilepsy	95.79	+1.46	+0.52	+1.89	+**3.48**	+2.40
Handwriting	40.47	+**4.00**	-18.29	+1.98	-12.32	-13.02
RacketSports	**88.82**	-0.56	-1.08	-0.79	-0.07	-1.49
HandMovementDirection	52.50	-18.97	-9.44	+**2.64**	+0.63	-1.16
Libras	80.61	-6.02	-1.25	+3.39	+3.62	+**6.04**
NATOPS	91.53	+1.25	+0.79	+**3.58**	+2.12	+2.36
SelfRegulationSCP2	48.37	+2.03	+4.98	+**8.52**	+7.20	+4.78
Heartbeat	64.90	+1.21	-1.78	+**11.88**	+1.59	+1.52
EthanolConcentration	27.04	-2.02	-3.71	+2.62	+6.97	+**7.22**
PEMS-SF	86.53	-41.61	-44.38	+**3.41**	-9.70	-13.04
SelfRegulationSCP1	79.51	-0.17	-0.73	+**9.84**	+6.40	+3.41
JapaneseVowels	94.41	+0.64	-10.73	+1.26	+1.45	+**2.25**
ArticularyWordRecognition	94.20	-2.41	-0.35	+1.80	+1.28	+**2.03**
MotorImagery	47.03	-2.14	+1.51	+**11.77**	+11.32	+9.67
FingerMovements	60.78	-1.65	-5.87	+1.42	-0.82	+**4.43**
Best counts	1	2	0	7	2	5

Dataset (#Train>1000)	None	Random	RandA	SLACK	TeachA	PostA
CharacterTrajectories	97.54	+0.50	-16.89	+**1.51**	+1.44	+1.39
LSST	43.46	+1.85	-6.12	+**19.19**	-0.15	-11.12
PhonemeSpectra	20.38	+5.02	-0.11	+6.16	+**6.36**	+4.94
FaceDetection	68.82	-5.88	-0.53	+**4.50**	+3.13	+1.60
SpokenArabicDigits	92.14	+5.23	+6.00	+**7.40**	+6.97	+7.19
PenDigits	97.39	+1.75	+0.48	+**2.17**	+1.89	+1.94
InsectWingbeat	65.98	+1.78	-0.37	+**2.91**	+1.33	+1.06
Best counts	0	0	0	6	1	0

from the combined train and test datasets with a preset seed value, such that the new class populations matched the original ones.

We compare our method with TeachAugment [25] and four baselines: *None* trains a target model without any augmentations. *Random* uses all trainable augmentations at random within max magnitudes. RandAugment [3] is a simple grid search (using a quarter of train set for validation here) on reduced augmentation parameters: the number of augmentations to apply at once and the magnitude level of fixed strength of augmentations (five levels here). SLACK [19] is one of the latest methods based on the maximization of a target model's performance on a validation set. SLACK first trains a model on half of the train set without augmentation and searches for augmentation policy by training the pre-trained model with each policy and evaluating it on the rest half. Then, the model is trained with the policy on a whole train set. Note that the augmentation parameterization in SLACK is not the same as that of others.

Results. The results are shown in Table 4, grouped by the number of train samples: less than 100 (top), 100 to 1000 (middle) and more than 1000 (bottom). Our method shows better results on small datasets. In contrast, SLACK is better on large datasets. This result can be explained as the optimization in SLACK, based on the split validation set, degrades on an extremely small dataset due to insufficiency of validation data. In contrast, our PostAugment and TeachAugment can utilize the whole train set to optimize augmentations.

5 Conclusion

We proposed the adversarial automatic data augmentation with hard sample suppression by incorrect class likelihood for time series data. The key challenge is that data augmentation for time series data is difficult to design a transformation function that does not cross the discriminant class boundary. The proposed method can reduce to generate this hard-to-identify data by introducing a novel fidelity loss that suppresses the likelihood of the incorrect class. Comprehensive experiments on two datasets demonstrated the effectiveness of the proposed method.

References

1. Bagnall, A., et al.: The uea multivariate time series classification archive (2018)
2. Charpentier, B., Zügner, D., Günnemann, S.: Posterior network: uncertainty estimation without ood samples via density-based pseudo-counts. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS'20 (2020)
3. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: practical automated data augmentation with a reduced search space (2019)
4. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: learning augmentation policies from data. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)

5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
6. Fons, E., Dawson, P., Zeng, X.J., Keane, J.A., Iosifidis, A.: Adaptive weighting scheme for automatic time-series data augmentation. ArXiv **abs/2102.08310** (2021). https://api.semanticscholar.org/CorpusID:231934182
7. Hataya, R., Zdenek, J., Yoshizoe, K., Nakayama, H.: Faster autoaugment: learning augmentation strategies using backpropagation. In: Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXV. Lecture Notes in Computer Science, vol. 12370, pp. 1–16. Springer (2020)
8. Hataya, R., Zdenek, J., Yoshizoe, K., Nakayama, H.: Meta approach to data augmentation optimization. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 2574–2583 (January 2022)
9. Ho, D., Liang, E., Stoica, I., Abbeel, P., Chen, X.: Population based augmentation: efficient learning of augmentation policy schedules. In: ICML (2019)
10. Ismail Fawaz, H., et al.: Inceptiontime: finding alexnet for time series classification. Data Mining and Knowledge Discovery (2020)
11. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. In: International Conference on Learning Representations (2017). https://openreview.net/forum?id=rkE3y85ee
12. Keyu, T., Chen, L., Ming, S., Luping, Z., Junjie, Y., Wanli, O.: Augment via augmentation-wise weight sharing. In: Advances in Neural Information Processing Systems, vol. 33, pp. 19088–19098 (2020)
13. Krizhevsky, A., Nair, V., Hinton, G.: Cifar-10 (canadian institute for advanced research). http://www.cs.toronto.edu/~kriz/cifar.html
14. Krizhevsky, A., Nair, V., Hinton, G.: Cifar-100 (canadian institute for advanced research)
15. Li, R., Li, X., Heng, P.A., Fu, C.W.: Pointaugment: an auto-augmentation framework for point cloud classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020)
16. Li, Y., Hu, G., Wang, Y., Hospedales, T.M., Robertson, N.M., Yang, Y.: DADA: differentiable automatic data augmentation (2020)
17. Lim, S., Kim, I., Kim, T., Kim, C., Kim, S.: Fast autoaugment. In: Advances in Neural Information Processing Systems (NeurIPS) (2019)
18. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2019). https://openreview.net/forum?id=Bkg6RiCqY7
19. Marrie, J., Arbel, M., Larlus, D., Mairal, J.: Slack: stable learning of augmentations with cold-start and kl regularization. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2023)
20. Mounsaveng, S., Vazquez, D., Ayed, I.B., Pedersoli, M.: Adversarial learning of general transformations for data augmentation (2019)
21. Rezende, D., Mohamed, S.: Variational inference with normalizing flows. In: Bach, F., Blei, D. (eds.) Proceedings of the 32nd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 37, pp. 1530–1538. PMLR, Lille, France (07–09 Jul 2015). https://proceedings.mlr.press/v37/rezende15.html
22. Rommel, C., Moreau, T., Paillard, J., Gramfort, A.: CADDA: class-wise automatic differentiable data augmentation for EEG signals. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=6IYp-35L-xJ

23. Sawada, A., Miyagawa, T., Ebihara, A.F., Yachida, S., Hosoi, T.: Convolutional neural networks for time-dependent classification of variable-length time series. 2022 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2022). https://api.semanticscholar.org/CorpusID:250408203
24. Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. J. Big Data (2019). https://doi.org/10.1186/s40537-019-0197-0, https://doi.org/10.1186/s40537-019-0197-0
25. Suzuki, T.: Teachaugment: Data augmentation optimization using teacher knowledge. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10904–10914 (June 2022)
26. Wang, X., Chu, X., Yan, J., Yang, X.: Daas: differentiable architecture and augmentation policy search (2021)
27. Wen, Q., et al.: Time series data augmentation for deep learning: a survey. In: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence. IJCAI-2021, International Joint Conferences on Artificial Intelligence Organization (Aug 2021). https://doi.org/10.24963/ijcai.2021/631, http://dx.doi.org/10.24963/ijcai.2021/631
28. Yang, H., Desell, T.: Robust augmentation for multivariate time series classification (2022)
29. Zheng, Y., Zhang, Z., Yan, S., Zhang, M.: Deep autoaugmentation. In: ICLR (2022)

Generating Pseudo-Strong Labels from Weak Labels for Distributed Multi-Microphone Sound Event Detection

Vijay John(✉) and Yasutomo Kawanishi

Guardian Robot Project, RIKEN, Seika, Kyoto, Japan
{vijay.john, yasutomo.kawanishi}@riken.jp

Abstract. Annotating frame-level strong labels for training a distributed multi-microphone sound event detection model to both recognize and temporally localize sound events within sequences is difficult and requires considerable time and effort. For a given distributed multi-microphone data, the strong labels identify the sound event categories in the environment along with their start and end times. Conversely, annotating sequence-level weak labels, which denote only the presence of sound events in the multi-microphone data without temporal information, is simpler. However, employing weak labels to train a distributed multi-microphone sound event detection model presents challenges. In this study, we propose leveraging weak labels within a distributed multi-microphone sound event detection framework to identify and temporally locate sound events across the multiple microphones. Our approach initially generates pseudo-strong labels for the distributed multi-microphones using the weak labels provided. Subsequently, a latent embedding estimation model for the audio data are learned using the generated pseudo-strong labels. Using transfer learning, the trained latent embedding estimated model are then integrated within a sound event detection model, which identifies and temporally localize sound events. By integrating the latent embedding estimation model that is learnt from the pseudo-strong labels with the sound event detection model, the proposed framework leverages the knowledge from the weak labels and transfers it for the sound event detection. We evaluated the proposed framework on the MM Office dataset and compared it with state-of-the-art baseline algorithms. The experimental results demonstrate that incorporating weak labels within the sound detection framework enhances the event detection accuracy.

Keywords: Weak Supervision · Distributed Multi-Microphone Sound Event Detection · Metric Learning

1 Introduction

Distributed multiple microphone sound event recognition is the research problem of recognizing and temporally localizing sound events from data captured by

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 98–113, 2025.
https://doi.org/10.1007/978-3-031-78192-6_7

multiple microphones scattered in a target scene. In a distibuted environment, such as office or home, a single source data is limited to sound events in its field-of-view. To effectively recognize sound events localized in different areas or sound events which span across different areas in the target scene a distributed multi-microphone data is required. For example, as shown in Fig 1, in an office environment with multiple rooms, to effectively recognize events, such as "person handing over documents to secretary", multiple microphones distributed across the office is required.

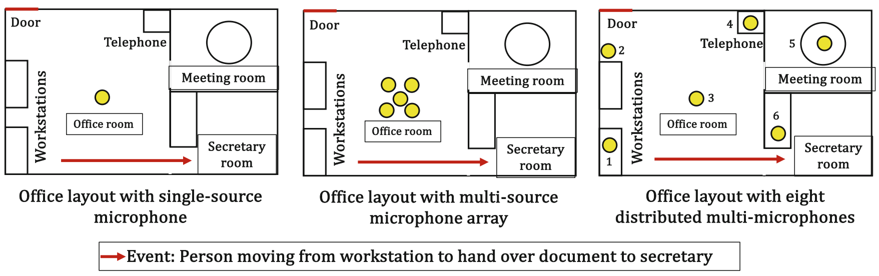

Door
Telephone
Meeting room
Workstations
Office room
Secretary room
Office layout with single-source microphone
Door
Telephone
Meeting room
Workstations
Office room
Secretary room
Office layout with multi-source microphone array
Door
4
Telephone
5
2
Meeting room
Workstations
3
Office room
6
1
Secretary room
Office layout with eight distributed multi-microphones
Event: Person moving from workstation to hand over document to secretary

Fig. 1. An overview of different audio configurations. Unlike the other two configurations, the distributed setup has the capability to capture individual events that span across multiple microphones, as well as events confined to specific areas within the environment.

Table 1. Label properties for sound event detection

Property	Strong Labels	Weak Labels	Pseudo-strong Labels
Num. Samples	Limited (N)	Large (M)	Large (M)
		$M \gg N$	$M \gg N$
Event Start/End Time.	✓	×	✓
Event Uncertainty	×	×	✓
Manually Annotated	✓	✓	×
Generated	×	×	✓

Traditionally, training a model for sound event detection requires strong labels, that annotate sound events along with their temporal information in a single-source audio sequence. However, obtaining these strong labels is a laborious and time-consuming. This challenge is exacerbated in the context of distributed multi-microphone sequences, where annotations necessitate consideration of data from the microphones distributed across different areas in the target scene. Conversely, weak labels, which merely identify sound events in an audio sequence without temporal information, are easier to annotate. Nonetheless, leveraging weak labels for training distributed multi-microphone sound event detection models to estimate sound events along with their temporal characteristics is non-trivial, as weak labels lack temporal information.

In this paper, a novel distributed multi-microphone sound event detection framework that leverages easily available weak labels for sound event detection is proposed. The characteristics of the different labels are shown in Table 1, where the easily available weak labels are devoid of the temporal information, and strong labels with temporal information are limited in quantity. In the proposed framework, to effectively leverage the knowledge from the weak labels for sound event detection, firstly, a pseudo-strong label generation algorithm is proposed to estimate the pseudo-strong labels containing the temporal information for the weak labels. Moreover, owing to the availability of limited strong label data, a transfer learning strategy is adopted, wherein a latent embedding estimation model, trained using the estimated pseudo-strong labels, is integrated within the sound event detection model. By estimating the pseudo-strong labels and integrating the trained latent embedding estimation model within the sound event detection model, the proposed framework effectively leverages weakly labeled data to enhance the robustness and accuracy of sound event detection systems.

The proposed framework operates in three stages. The first step generates the pseudo-strong labels and class confident labels from the distributed multi-microphone data and its weak labels using a pseudo-strong label generation (PSLG) algorithm. Given that pseudo-strong labels are generated from weak labels, there exist inherent uncertainties linked with their generation. These uncertainties are expressed through the utilization of class confident labels. Using them the second step trains a latent embedding estimation (LEE) model, trained by metric learning along with the pseudo-strong label classification. Finally, in the third step, the trained LEE model is integrated into the sound event detection model. The sound event detection model is trained using limited number of distributed multi-microphone data with strong labels which contain the temporal information. The proposed framework leverages the knowledge from easily available weak labels to recognize and localize sound events in distributed multi-microphone data.

The proposed framework is evaluated using the MM Office dataset [16], which comprises distributed multi-microphone data with a lot of weak labels and a few strong labels. The experiment section includes a comparative analysis with the baseline algorithms and a detailed ablation study. Results demonstrate the efficacy of the proposed framework in leveraging weak labels for sound event detection. This study contributes to the literature in the following ways:

- Proposal of a method that leverages weak labels for distributed multi-microphone sound event detection. Existing literature primarily focuses on using weak labels for single-source sound event detection.
- Development of a novel pseudo-strong label generation algorithm to estimate pseudo-strong labels and class confidence labels from weak labeled data.
- Proposal of a novel latent loss function for training the latent embedding estimation model.

The remainder of this paper is structured as follows. Section 2 presents a literature review. Section 3 details the framework. Section 4 discusses the validation of the framework and presents experimental results. Finally, Sect. 5 summarizes the findings and outlines the future research directions.

2 Literature Review

Sound event detection (SED) is the problem of recognizing and localizing sound events in an audio sequence [11]. Deep learning-based SED frameworks report state-of-the-art results using the CNN [4], CRNN [2], and 3D CNN models [10]. To increase the robustness of the SED, and to perform the recognition over a wider area, researchers utilize multiple microphones scattered in a scene for the event detection. The literature on distributed multi-microphone sound event detection (SED) focuses on efficiently utilizing multi-microphone data to recognize and localize sound events [1,3,14]. Casebee et al. [3] propose a deep learning network for sound classification, where the network is trained on a fixed number of channels, and the trained network is tested with varying number of microphones. Phan et al. [14] propose a network, where four subnetworks handling four low-level audio representations are used. The learned embeddings from these subnetworks are used for the multi-microphone sound classification. The aforementioned literature reports state-of-the-art detection accuracy but requires the availability of strong labels.

In recent years, researchers have addressed this limitation by utilizing readily available weak labels to train the SED models [5,9,12,13,15]. Shao et al. [15] propose a multiple instance learning framework using CNN-based audio embeddings to leverage weak labels for the model training. In the work by Miyazaki et al. [12] and Kong et al. [9], a transformer and a self-attention mechanism are used within a weakly-supervised learning formulation. While reporting state-of-the-art performance, the existing literature on leveraging weak labels for SED is primarily restricted to single view audio processing.

Compared to the literature, in this paper, we propose to utilize weak labels for a distributed multi-microphone SED framework. Moreover, a novel pseudo-strong label generation algorithm is first proposed to estimate pseudo-strong labels from the weak labels.

3 Proposed Framework

The proposed distributed multi-microphone SED framework is trained using a dataset obtained from S microphones scattered in a scene. One sample of the data is represented as a set of spectrograms, $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_P)$ with P patches. Here, the p-th patch contains spectrograms from S microphones and is represented as $\mathbf{X}_p = (\mathbf{x}_{p,1}, \ldots, \mathbf{x}_{p,S})$. The strong label that is annotated for every frame to recognize and localize the sound events for $\mathbf{X}$ is given as $\mathbf{g}^\lambda = (\mathbf{g}_1^\lambda, \ldots, \mathbf{g}_P^\lambda)$, where $\mathbf{g}_p^\lambda = (g_{p,1}^\lambda, \ldots, g_{p,C}^\lambda)$, and $\forall c \in \{0, \ldots C\}$, $g_{p,c}^\lambda \in \{0, 1\}$ for the p-th patch in $\mathbf{X}$. Strong label annotation, i.e., frame-level annotation,

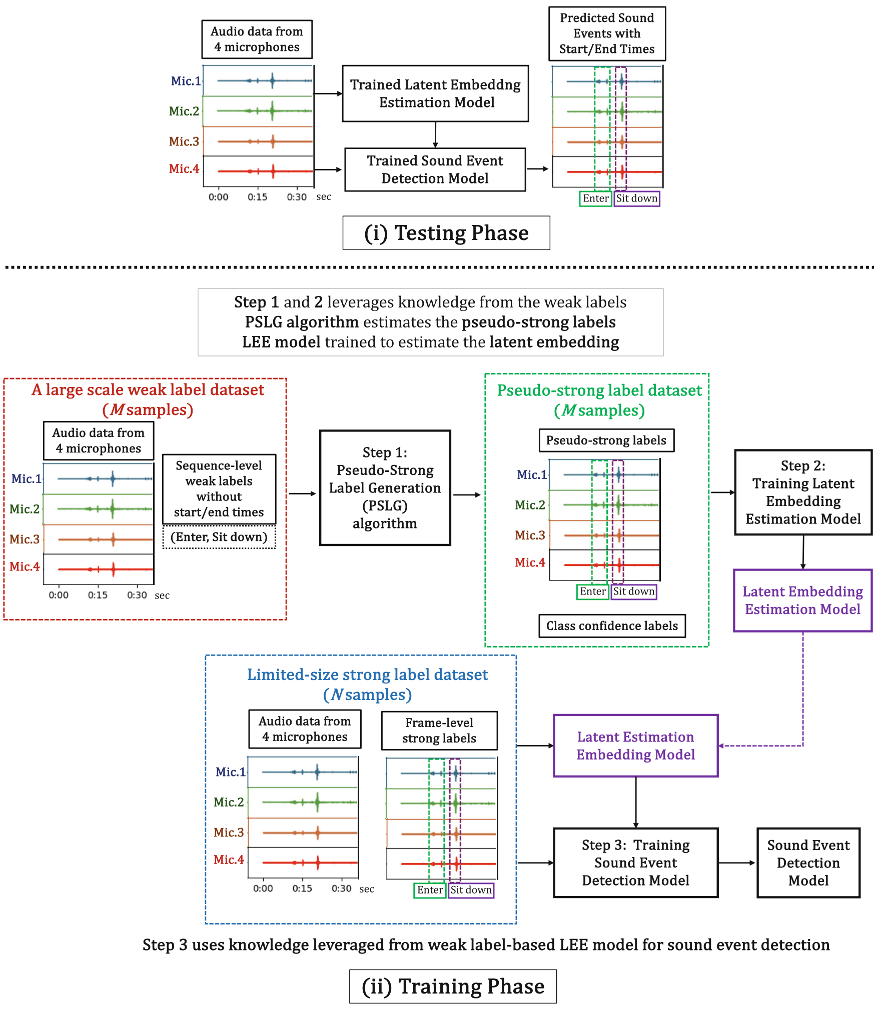

Audio data from 4 microphones
Mic.1
Mic.2
Mic.3
Mic.4
0:00 0:15 0:30 sec
Trained Latent Embeddng Estimation Model
Trained Sound Event Detection Model
Predicted Sound Events with Start/End Times
Enter
Sit down
(i) Testing Phase
Step 1 and 2 leverages knowledge from the weak labels
PSLG algorithm estimates the pseudo-strong labels
LEE model trained to estimate the latent embedding
A large scale weak label dataset (M samples)
Sequence-level weak labels without start/end times
(Enter, Sit down)
Step 1: Pseudo-Strong Label Generation (PSLG) algorithm
Pseudo-strong label dataset (M samples)
Pseudo-strong labels
Class confidence labels
Step 2: Training Latent Embedding Estimation Model
Latent Embedding Estimation Model
Limited-size strong label dataset (N samples)
Frame-level strong labels
Latent Estimation Embedding Model
Step 3: Training Sound Event Detection Model
Sound Event Detection Model
Step 3 uses knowledge leveraged from weak label-based LEE model for sound event detection
(ii) Training Phase

Fig. 2. An overview of the (i) testing phase and the (ii) training phase of the proposed framework.

is laborious to annotate for distributed multi-microphone dataset, while weak label annotation, i.e., sequence-level annotation, is easier. Thus, a large number of weak-labeled data can be obtained. The weak labels that identify the sound events without temporal information for a sequence X are given as $\mathbf{g}^{\omega} = (g_1^{\omega}, \ldots, g_C^{\omega})$, where $\forall c\ in\{1, \ldots C\}, g_c^{\omega} \in \{0, 1\}$, and shared within the patches in the sequence $\mathbf{X}$.

Since the weak labels are easier to annotate, the distributed multi-microphone dataset used to train the proposed framework contains M samples with weak labels and N samples with strong labels, where $M \gg N$. The weakly labeled samples are represented as $\{(\mathbf{X}^{\omega}, \mathbf{g}^{\omega})\}_M$ and are termed as the *weak label* dataset. The strong label samples are represented as $\{(\mathbf{X}^{\lambda}, \mathbf{g}^{\lambda})\}_N$ and are termed the *strong label* dataset.

As shown in Fig 2-i, in the proposed framework, the trained distributed multi-microphone SED model estimates the strong labels for a given multi-microphone audio input. A trained latent embedding estimation model is integrated within the SED model to leverage the knowledge from the weak labels for the strong label prediction. The proposed framework is implemented in three steps (Fig 2-ii). In the first step, the pseudo-strong label generation algorithm generates the pseudo-strong and class confident labels for the *weak label* dataset. Subsequently, using the generated labels the latent embedding estimation model, which is a feature extractor for the next step, is trained. The trained latent embedding estimation model is then integrated into a distributed multi-microphone SED model, which is trained using the *strong label* dataset. Next, we describe each step in detail.

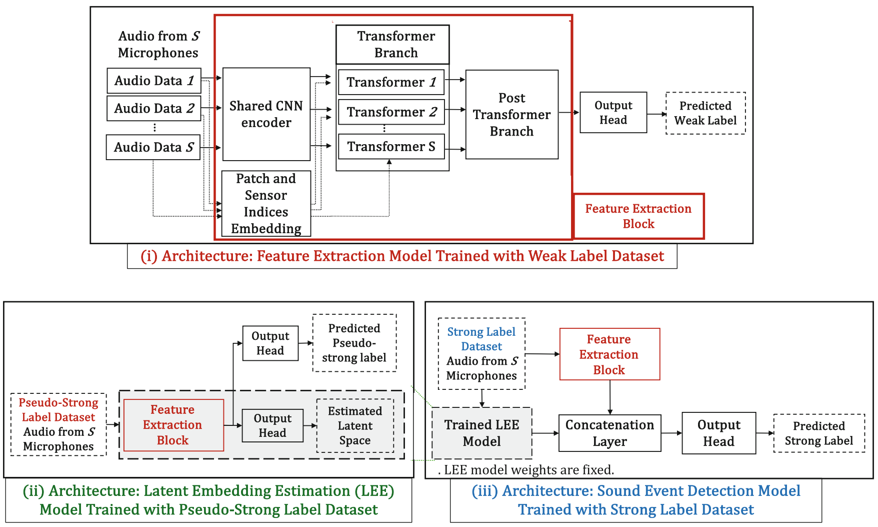

Audio from S Microphones
Audio Data 1
Audio Data 2
Audio Data S
Shared CNN encoder
Patch and Sensor Indices Embedding
Transformer Branch
Transformer 1
Transformer 2
Transformer S
Post Transformer Branch
Output Head
Predicted Weak Label
Feature Extraction Block
(i) Architecture: Feature Extraction Model Trained with Weak Label Dataset
Pseudo-Strong Label Dataset Audio from S Microphones
Feature Extraction Block
Output Head
Predicted Pseudo-strong label
Output Head
Estimated Latent Space
(ii) Architecture: Latent Embedding Estimation (LEE) Model Trained with Pseudo-Strong Label Dataset
Strong Label Dataset Audio from S Microphones
Feature Extraction Block
Trained LEE Model
Concatenation Layer
Output Head
Predicted Strong Label
. LEE model weights are fixed.
(iii) Architecture: Sound Event Detection Model Trained with Strong Label Dataset

Fig. 3. A detailed overview of the architecture of the (i) feature extraction model, (ii) latent embedding estimation model, and (iii) sound event detection model in the proposed framework.

3.1 Pseudo-Strong Label Generation (PSLG) Algorithm

Given the *weak label* dataset, pseudo-strong labels are generated from the available weak labels. As a precursor to the PSLG algorithm, we first formulate a feature extraction model, f_A, to extract a sequence of features $\mathbf{F}^{\omega} = (f_A(\mathbf{x}_i^{\omega})|\mathbf{x}_i^{\omega} \in \mathbf{X}^{\omega})$ for a sequence $\mathbf{X}^{\omega}$. The feature extraction model, f_A, is implemented using multiple blocks termed the *feature extraction block*, which contains a CNN encoder block, embedding block, multi-head attention transformer block, and post-transformer block.

The shared CNN encoder block is shared across the multi-microphone data. The feature extraction model is trained using an output head, which predicts the weak label. The architecture of the feature extraction model, f_A, and its implementation are presented in Fig 3-i and Sect. 4.

Using the set of extracted features and corresponding weak-labels $\{(\mathbf{F}^{\omega}, \mathbf{g}^{\omega})\}_M$, the PSLG algorithm generates pseudo-strong labels $\mathbf{g}^{\theta} = (\mathbf{g}^{\theta}_1,\ldots,\mathbf{g}^{\theta}_P)$, for all $\mathbf{X}^{\omega}$, where $\mathbf{g}^{\theta}_p = (g^{\theta}_{p,1},\ldots,g^{\theta}_{p,C+1})$ and $\forall c \in \{1,\ldots,C+1\}$, $g^{\theta}_{p,c} \in \{0,1\}$ for the p-th patch. The additional class label $C+1$ indicates *uncertain* class in the pseudo-strong label generation. Given that pseudo-strong labels are generated from weak labels, there exist inherent uncertainties linked with their generation. These uncertainties are expressed using a binary class confidence vector denoted as $\eta \in \mathbb{R}^C$, wherein each component η_c assumes a value of zero to denote classes characterized by uncertain generation and one otherwise, indicating certainty. Using η, the PSLG algorithm estimates the patch-level class confidence labels $\boldsymbol{\psi} = (\boldsymbol{\psi}_1,\ldots,\boldsymbol{\psi}_P)$, where $\forall p \in \{1,\ldots,P\}$, $\boldsymbol{\psi}_p \in \{0,1\}$ denotes the confidence of the estimated pseudo-strong label.

The PSLG algorithm first creates clusters, and computes the multivariate Gaussian parameters for every class incrementally. Then, pseudo-strong labels are annotated to all patches in the dataset by comparing with the computed multivariate Gaussian parameters. A detailed description of the PSLG algorithm is presented in Algorithms 1 and 2. In Algorithm 1, the multivariate Gaussian parameters and class certainty vector for the *weak label* dataset are computed. The pseudo-strong and class confidence labels are then computed using Algorithm 2. An illustration of the algorithms are presented in Fig. 4.

3.2 Latent Embedding Estimation (LEE)

The LEE model estimates the latent space by performing a patch-level embedding of the distributed multi-microphone data $\mathbf{X}^{\omega}_p$ using the generated pseudo-strong $\mathbf{g}^{\theta}_p$ and class confidence labels $\boldsymbol{\psi}_p$. The LEE model is implemented using the *feature extraction block* and two output heads. The first output head predicts the pseudo-strong labels for the distributed multi-microphone input and the second output head obtains the patch-level latent embedding of the distributed multi-microphone input. This model is trained by a multi-task learning framework.

To estimate the latent space, an extension of the triplet hard loss is proposed. The original triplet hard loss [6] is used to estimate the latent space by minimizing the distance between the positive pairs and maximizing the distance between the negative pairs. In the proposed extension, a batch hard mining strategy is utilized to build triplets of anchors, "confident" positive data, and negative data, which are used in the extended triplet hard loss calculated as,

$$\mathcal{L}_f = \log_e(1 + e^{\alpha}), \tag{1}$$

$$\alpha = d(\mathbf{a}, \mathbf{p}^{\psi}) - d(\mathbf{a}, \mathbf{n}) \tag{2}$$

Algorithm 1: Algorithm for Calculating Gaussian Parameters

```
Input  : Set of extracted features and weak labels, $\{\mathbf{F}^{\omega}, \mathbf{g}^{\omega}\}_M$
Output: Multivariate Gaussian parameters, $\boldsymbol{\mu} = \{\boldsymbol{\mu}_0, \dots, \boldsymbol{\mu}_C\}$.;
Covariance $\boldsymbol{\Sigma} = \{\boldsymbol{\Sigma}_0, \dots, \boldsymbol{\Sigma}_C\}$.;
Class confidence vector $\eta \in \mathbb{R}^C$
// Initialize class confidence vector and selected classes set
$\eta = \mathbf{0} \in \mathbb{R}^C$; $C_s \leftarrow \{\}$; $C_u \leftarrow \{0, \dots, C\}$;
// Compute multivariate Gaussian parameters for background class
Retrieve patches from sequences with all-zero weak labels.;
Compute $\boldsymbol{\mu}_0$ and covariance $\boldsymbol{\Sigma}_0$ using the retrieved patches;
Update the selected classes set $C_s \leftarrow \{0\}$;
Update the zeroeth index in the class confidence vector $\eta_0 \leftarrow 1$;
// Compute multivariate Gaussian parameters for each class
while $C_s \neq C_u$ do
    // Create an unselected classes set
    $\overline{C_s} = C_u \setminus C_s$;
    for unselected class $c$ in $\overline{C_s}$ do
        Retrieve sequences from $\mathbf{F}^{\omega}$ with $g_c^{\omega} = 1$ and $g_{\bar{c}}^{\omega} = 0 \ \forall \bar{c} \in \overline{Cs} \setminus c$;
        for $p$-th patch in $k$-th sequence do
            // Check if $p$-th patch belongs to $C_s$ or not using
               Mahalanobis distance
            $d_s = \{MahalanobisDistance\left(\mathbf{F}_p^{\omega}, (\mu_{\widehat{c}}, \Sigma_{\widehat{c}})\right), \forall\ \widehat{c} \in C_s\}$
            if all($d >$ threshold in $d_s$) then
                Add $p$-th patch to the class-wise cluster set for unselected class
                $c$;
            end
        end
        Compute $\boldsymbol{\mu}_c$ and covariance $\boldsymbol{\Sigma}_c$ using the class-wise cluster set;
        Update the $c$-th index in class confidence vector $\eta_c \leftarrow 1$;
        Update the selected classes set $C_s \leftarrow C_s \cup c$;
    end
end
```

where, anchor data $\mathbf{a}$ corresponds to the patch embedding in the latent space. The "confident" positive data, $\mathbf{p}^{\psi}$, for a given anchor, corresponds to the patch embedding with the same class label and non-zero class confidence label $\boldsymbol{\psi}_p = 1$. Note that a nonzero confidence label is used only to select the positive data, whereas the negative class data could have any class confidence label.

For a given anchor, the extended triplet hard loss only minimizes the distance to the "confident" positive data while ignoring "unconfident" positive data. On the other hand, the anchor maximizes the distance to all negative data. In the case of an "unconfident" anchor, the positive class being "unconfident" is ignored, while both "confident" and "unconfident" negative data are considered. The architecture and implementation details of the LEE model are presented in Fig 3-ii and Sect. 4.

Algorithm 2: Algorithm for Generating Pseudo-strong Labels

```
Input  : Set of extracted features and weak labels, ${(\mathbf{F}^{\omega}, \mathbf{g}^{\omega})}_M$; Multivariate
         Gaussian distribution parameters, $\boldsymbol{\mu} = \{\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_C\}$ and
         $\boldsymbol{\Sigma} = \{\boldsymbol{\Sigma}_0, \ldots, \boldsymbol{\Sigma}_C\}$, and Class confidence vector $\eta$.
Output: Pseudo-strong labels $\{\mathbf{g}^{\theta}_p\}_M$, where
         $\mathbf{g}^{\theta}_p(m)_M = (g^{\theta}_{p,1}(m), \ldots, g^{\theta}_{p,C+1}(m))$, and their class confidence labels
         $\{\boldsymbol{\psi}_p\}_M$;

for (m) and $\mathbf{g}^{\omega}(m)$ in the weak label dataset do
    $C_s = \{i | g^{\omega}_i(m) = 1\}$;
    // Check the class confidence vector's value for the subset $C_s$
    // If there exists an unconfident class amongst $C_s$
    if $\exists\{\eta_i = 0\}_{i=0}^{C_s}$ then
        for p-th patch in m-th sequence do
            // Class confidence label assignment
            $\boldsymbol{\psi}_p(m) = 0$;
            // Pseudo-strong label assignment
            $\mathbf{g}^{\theta}_{p,i}(m) = \begin{cases} 1 & \text{if } i = C+1 \\ 0 & \text{otherwise} \end{cases}$
        end
    end
    else
        for p-th patch in m-th sequence do
            // Class confidence label assignment
            $\boldsymbol{\psi}_p(m) = 1$;
            // Compute pseudo-strong labels
            $j^* = \text{argmin}_{j \in C_s} \text{MahalanobisDistance}(F^{\omega}(m), (\mu_j, \Sigma_j))$;
            $\mathbf{g}^{\theta}_{p,i}(m) = \begin{cases} 1 & \text{if } i = j^* \\ 0 & \text{otherwise} \end{cases}$
        end
    end
end
```

3.3 Sound Event Detection (SED) Model

The SED model is formulated to recognize and localize the sound events in the distributed multi-microphone data. The trained LEE model is integrated within the SED model to leverage the information learned from the weak labels. In addition to the trained LEE model, the SED model is implemented with the *feature extraction block* and an output head. The output head recognizes and localizes the sound events in the multi-microphone audio data. The SED model is trained using the *strong label dataset*, $(\mathbf{X}^{\lambda}, \mathbf{g}^{\lambda})_N$.

The input to the SED model is the distributed multi-microphone data $\mathbf{X}^{\lambda}$ with P patches. The trained LEE model estimates the latent embedding $\mathbf{B}$

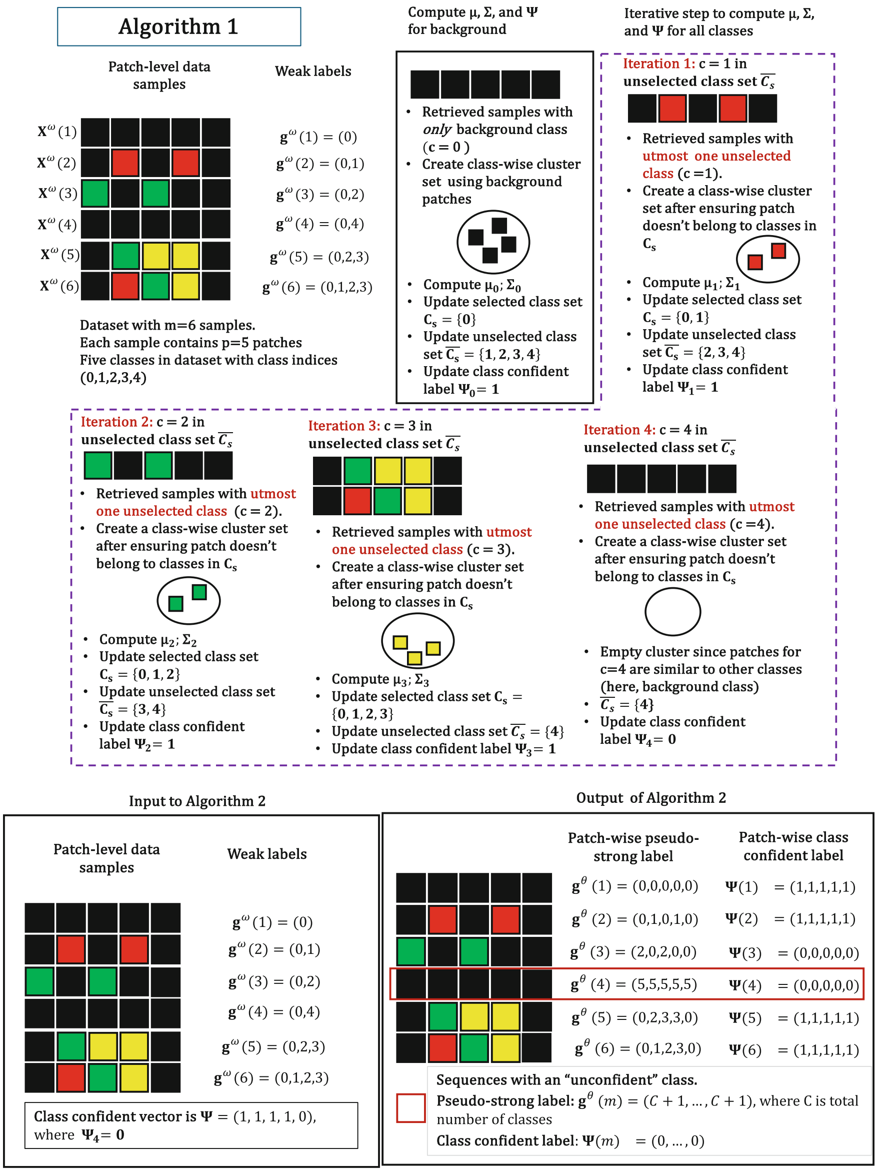

Algorithm 1
Patch-level data samples
Weak labels
$X^\omega(1)$ … $X^\omega(6)$
$g^\omega(1) = (0)$
$g^\omega(2) = (0,1)$
$g^\omega(3) = (0,2)$
$g^\omega(4) = (0,4)$
$g^\omega(5) = (0,2,3)$
$g^\omega(6) = (0,1,2,3)$
Dataset with m=6 samples.
Each sample contains p=5 patches
Five classes in dataset with class indices (0,1,2,3,4)
Compute μ, Σ, and Ψ for background
Retrieved samples with only background class (c = 0)
Create class-wise cluster set using background patches
Compute μ_0; Σ_0
Update selected class set $C_s = \{0\}$
Update unselected class set $\overline{C_s} = \{1,2,3,4\}$
Update class confident label $\Psi_0 = 1$
Iterative step to compute μ, Σ, and Ψ for all classes
Iteration 1: c = 1 in unselected class set $\overline{C_s}$
Retrieved samples with utmost one unselected class (c = 1).
Create a class-wise cluster set after ensuring patch doesn't belong to classes in C_s
Compute μ_1; Σ_1
Update selected class set $C_s = \{0,1\}$
Update unselected class set $\overline{C_s} = \{2,3,4\}$
Update class confident label $\Psi_1 = 1$
Iteration 2: c = 2 in unselected class set $\overline{C_s}$
Retrieved samples with utmost one unselected class (c = 2).
Create a class-wise cluster set after ensuring patch doesn't belong to classes in C_s
Compute μ_2; Σ_2
Update selected class set $C_s = \{0,1,2\}$
Update unselected class set $\overline{C_s} = \{3,4\}$
Update class confident label $\Psi_2 = 1$
Iteration 3: c = 3 in unselected class set $\overline{C_s}$
Retrieved samples with utmost one unselected class (c = 3).
Create a class-wise cluster set after ensuring patch doesn't belong to classes in C_s
Compute μ_3; Σ_3
Update selected class set $C_s = \{0,1,2,3\}$
Update unselected class set $\overline{C_s} = \{4\}$
Update class confident label $\Psi_3 = 1$
Iteration 4: c = 4 in unselected class set $\overline{C_s}$
Retrieved samples with utmost one unselected class (c = 4).
Create a class-wise cluster set after ensuring patch doesn't belong to classes in C_s
Empty cluster since patches for c=4 are similar to other classes (here, background class)
$\overline{C_s} = \{4\}$
Update class confident label $\Psi_4 = 0$
Input to Algorithm 2
Patch-level data samples
Weak labels
Class confident vector is $\Psi = (1,1,1,1,0)$, where $\Psi_4 = 0$
Output of Algorithm 2
Patch-wise pseudo-strong label
Patch-wise class confident label
$g^\theta(1) = (0,0,0,0,0)$ $\Psi(1) = (1,1,1,1,1)$
$g^\theta(2) = (0,1,0,1,0)$ $\Psi(2) = (1,1,1,1,1)$
$g^\theta(3) = (2,0,2,0,0)$ $\Psi(3) = (0,0,0,0,0)$
$g^\theta(4) = (5,5,5,5,5)$ $\Psi(4) = (0,0,0,0,0)$
$g^\theta(5) = (0,2,3,3,0)$ $\Psi(5) = (1,1,1,1,1)$
$g^\theta(6) = (0,1,2,3,0)$ $\Psi(6) = (1,1,1,1,1)$
Sequences with an "unconfident" class.
Pseudo-strong label: $g^\theta(m) = (C+1, \dots, C+1)$, where C is total number of classes
Class confident label: $\Psi(m) = (0, \dots, 0)$

Fig. 4. An illustration of PSLG Algorithms.

from the multi-microphone input. The latent embedding and the output of the *feature extraction block* is concatenated and given as input to the output head. The architecture and implementation details of the sound event detection model are presented in Fig 3-iii and Sect. 4.

4 Experiment

4.1 Dataset

The proposed algorithm is evaluated using the public MM Office dataset [16], which contains weak and strong label datasets. For both the weak and strong label datasets, distributed multi-microphone data is obtained from 8 microphones in an indoor office environment. The weak label dataset contains $M = 720$ distributed multi-microphone sequences obtained from $S = 8$ microphones, for a total of $5,760$ audio sequences. Each distributed multi-microphone data contains weak labels represented by $C = 13$ sound classes without their temporal information. The strong label dataset contains $N = 88$ distributed multi-microphone sequences obtained from 8 microphones, with strong labels represented by 13 sound classes along with their start and end times.

The length of the audio sequences in the weak label dataset varied between 25 and 45 s. The length of the audio sequences in the strong label dataset vary between 1 min and 4 min 50 s. Consequently, the shorter sequences are preprocessed by zero padding to ensure that all audio sequences have a length of 45 s for the weak label dataset and 4min 50 s for the strong label dataset.

4.2 Implementation

Multi-Microphone Input: The distributed multi-microphone input to the proposed framework is represented as the log-Mel-spectrogram. Each log-mel-spectrogram for a particular view is obtained using a sampling rate of 16 000 Hz, hop length 300, and 128 Mel bins. The log-Mel-spectrogram for the weak label dataset is obtained with size $128 \times 2,401$. The spectrogram for the strong label dataset is obtained with size $128 \times 15,370$. In our framework, the spectrogram is split into P patches, with each p-th patch corresponding to a spectrogram patch of size 128×53. The weak label dataset contains 45 patches for the 45 second data, and the strong label dataset contains 290 patches for the 290 seconds.

Feature Extraction Block: The feature extraction block contain the CNN encoder block, embedding block, multi-head attention transformer block, and post transformer block. The CNN encoder block is shared across the multi-microphone data. The encoder is implemented with three 1D convolution layers with $64, 32$, and 32 filters with 3 kernels and ReLU activation, followed by a fully-connected layer with 256 units and ReLU activation.

The embedding block is implemented using Embedding layers for the patch and sensor indices, which embed the patch and sensor indices to a 256-dim vector. The output of the embedding block is added to the output of the feature extraction block.

The transformer block contains eight transformer branches corresponding to each view. Each transformer branch, firstly, computes the multi-head attention using 4 heads. The attention output is then added to the residual transformer input to obtain the *added* attention output. This output is subsequently projected using two fully-connected layers with 32 and 256 units with ReLU and linear activations. The projected output is added to the *added* attention output to obtain the final output.

In the post transformer block, the outputs of the eight transformers are first concatenated, along the view dimension corresponding to the different microphones and a max pooling is performed along it to obtain the *max* features.

Output Head of the Feature Extraction model: The input to the output head is obtained using the Flatten layer, which makes the multi-dimensional *max* features one-dimensional. The output head is implemented with two fully-connected layers with 512 and 13 units with ReLU and sigmoid activation labels. The output head predicts weak labels. The feature extraction model is trained using a binary cross-entropy function.

Output Heads of the LEE model: There are two output heads in the LEE model, and both the inputs correspond to the multi-dimensional *max* features. The first output head is implemented with two fully-connected layers with 512 and 14 units with ReLU and sigmoid activation labels to predict pseudo-strong labels. The output head is trained using a binary cross entropy function. The second output head outputs the latent embedding using a fully-connected layer of 256 units with a linear activation function, followed by L2 normalization. The output head is trained using the extended triplet hard loss function.

Output Heads of the SED model: The input to the output head for the SED model correspond to the concatenated vector of the *max* feature and latent embeddings. The output head is implemented using two fully-connected layers with 512 and 13 units with ReLU and sigmoid activation labels. The SED model is trained using a binary cross-entropy function.

4.3 Training Parameters

The proposed framework was implemented with Tensorflow 2 using NVIDIA 3090 GPUs on an Ubuntu 20.04 desktop. The model parameters are selected empirically with learning rate of 0.001, β_1=0.5 and β_2=0.99 for all three models. The models are trained for 100 epochs.

4.4 Baseline Algorithms

The first baseline, based on John et al. [8], adapted for multi-microphone data, utilizes the *feature extraction block* with an output head in the LEE model to estimate the latent space. The latent space is learnt using a weak latent label loss. The trained LEE model is integrated within the SED model.

The second baseline, adapted from Mei et al. [10], employs 3D CNN with two layers of Conv 3D with 32 filters of size $1 \times 3 \times 3$ for feature extraction from distributed multi-microphone, followed by SED output head.

In the third baseline, the proposed framework is modified, without any latent space learning. The fourth baseline modeled after Boes et al. [2], utilizes the shared CNN encoder block, view-specific RNN with four units, and post-transformer block followed by SED output head. The fifth baseline is similar to the fourth baseline, but the view-specific RNN layers are replaced with LSTM layers.

Finally, the sixth baseline utilizes Audio Spectrogram Transformer (AST) [7] extended to 8 views, with separate transformer branches for each view. The output from the different views are sent to post-transformer block and SED output head.

Only the baseline inspired by John et al.[8] utilizes knowledge transfer from weak label-based learning via a latent space generation model. Other baselines employ transfer learning for knowledge transfer from weak labels. These baselines are initially trained with weak labels using feature extraction output heads (Sect. 4.2). Subsequently, the pre-trained baseline model are fine-tuned with the strong label dataset. For the fine-tuning, the output head of the baseline models are replaced with the SED output head (Sect. 4.2).

Table 2. Comparative Analysis of the Baseline Algorithms

Algorithm	Feature Ext. Model	Transformer	Knowledge Transfer	Latent Loss	Class. Acc
Proposed	CNN-1D	✓	Latent space	Ext. triplet	**86.55**
John et al. [8]	CNN-1D	✓	Latent space	Weak lat.	85.58
CNN 3D. [10]	CNN-3D	×	Transf. learn.	×	82.70
CNN-Trans.	CNN-1D	✓	Transf. learn.	×	85.13
CNN-RNN [2]	CNN-1D	×	Transf. learn.	×	78.50
CNN-LSTM	CNN-1D	×	Transf. learn.	×	78.50
AST [7]	Patch Embed.	✓	Transf. learn.	×	80.41

Table 3. Ablation Study of the Proposed Framework

Algo.	Knowledge Transfer	Cluster Param. (Algo 1)	Dist. Meas. (Algo 2)	LEE model	Latent loss function	Class. Acc
Prop.	Latent space	(μ, Σ)	Mahalanobis	LEE	Ext. triplet	**86.55**
Abl.-A	Latent space	(μ, Σ)	Mahalanobis	ALS	Ext. triplet	84.25
Abl.-B	Latent space	Only μ	Euclidean	LEE	Ext. triplet	84.98
Abl.-C	Latent space	(μ, Σ)	Mahalanobis	LEE	Triplet hard	83.10

4.5 Ablation Study

An ablation study is performed to validate the different components of the proposed framework, as presented in Table 3. In the first variant, Ablation-A, the latent space is generated directly from the multi-source data using an output head which performs the latent embedding using a fully-connected layer of 256 units with a linear activation function, followed by L2 normalization. We term this model, without the feature extraction block, the Audio Latent Space (ALS) model.

In the second variant, Ablation-B, in Algorithm 1 only the cluster mean μ is computed for the cluster parameters. Subsequently, in Algorithm 2, Euclidean distance is used as the distance measure to compute the pseudo-strong labels. In Ablation-C, the latent space is learned using the triplet hard loss instead of the extended triplet loss function.

4.6 Distributed Multi-Microphone Vs Single-Source Dataset

The proposed framework is implemented and validated on the distributed multi-microphone dataset [16]. The proposed framework is implemented using eight microphones of the MM Office dataset. We perform a comparative study of the proposed framework on the MM Office dataset by varying the number of microphones in Table 4. For the experiment studies, the proposed framework is trained and evaluated with the subsets without any modifications. For the evaluation with the individual microphones, the sensor embedding and the post-transformer branches are removed in the different models.

Table 4. Comparative analysis using various subsets in the MM Office dataset

Complete Set.	Set of Four Mics.	Set of Two Mics.	Individual Mics.	
Eight Mics: **86.55**	Odd Mics: 83.89	Mic 1,5: 82.82	Mic 1: 82.63	Mic 5: 83.29
	Even Mics: 85.52	Mic 2,6: 84.95	Mic 2: 83.23	Mic 6: 83.34
		Mic 3,7: 83.28	Mic 3: 82.13	Mic: 7 83.75
		Mic 2,6: 84.69	Mic 4: 84.13	Mic: 8 83.29

4.7 Results and Discussion

Table 2 shows that the proposed framework has better accuracy than the baseline algorithms. We further analyse the result in detail.

Transformer: Compared to baselines using the RNN and the LSTM, the transformer effectively captures the long-range dependencies by employing attention mechanisms. The embedding block in the transformer to address its limitations of being permutation equivariant. The patch and sensor embedding information allows the transformer to process distributed multi-microphone sequences without losing the sensor layout, and patch-wise sequential information. The post-transformer block effectively captures the important information across the different views, while ignoring the unimportant features.

Knowledge Transfer: The proposal to utilize latent space for the knowledge transfer instead of transfer learning is shown to be effective. The proposed framework and John et al. [8] report better performance than the other baselines. These frameworks utilize metric learning, where in the learned latent embeddings, the data from similar classes are embedded close to each other in the latent space, and the data from different classes are kept apart from each other.

Latent Space: In Table 2 and Table 3, the advantages of the proposed extended triplet loss can be observed. The proposed framework with the extended triplet loss reports better accuracy than John et al. [8], which uses the weak latent loss, and Ablation-C, which uses the original triplet hard loss to learn the latent loss. Compared to the other two latent losses, only the extended triplet loss includes the class confidence vector $\boldsymbol{\psi}$ in its loss calculation (Eq. 2) to minimize the distance between the anchor and "confident" positive data.

PSLG Algorithm: In Table 3, an experiment is performed to evaluate the PSLG algorithm. More specifically, the choice to estimate the multivariate Gaussian parameters and the Mahalonobis distance to estimate pseudo-strong labels is evaluated. A comparison of the results of the proposed framework with those of Ablation-B, where only the mean of the clusters is computed, shows that the proposed framework yields better results. This can be attributed to the multivariate Gaussian parameters μ and Σ which represent the cluster distributions better than the mean alone.

LEE Model: Experimentation in Table 3 favors our LSG model employing a standard transformer block in a multi-tasking framework, surpassing performance of the ALS model.

Varying Subsets: The results indicate that employing all available microphones leads to superior performance compared to using different subsets of microphones. This enhanced performance can be attributed to specific events in the MM Office dataset either spanning multiple microphones or being confined to particular ones. Additionally, the findings demonstrate that the framework can be adapted to individual microphones with minimal modifications. Future work will focus on refining and testing this adaptation.

5 Conclusion

In conclusion, this paper introduces a novel framework for distributed multi-microphone sound event detection, utilizing weak labels for sound event detection. Here, pseudo-strong labels are generated from the weak labels, and used to train the latent embedding estimation model, which is subsequently integrated within the sound event detection model. The evaluation on the MM Office dataset demonstrates the effectiveness of the proposed approach compared to baseline algorithms, supported by a comprehensive ablation study. In the future work, we will extend the framework to distributed multimodal data. To facilitate the extension, we will also acquire a distributed multimodal dataset in an indoor home or office environment.

Acknowledgement. This work was partly supported by the Japan Society for the Promotion of Science (JSPS) KAKENHI JP23H03474.

References

1. Adavanne, S., Politis, A., Virtanen, T.: Multichannel sound event detection using 3d convolutional neural networks for learning inter-channel features. arXiv/1801.09522 (2018)
2. Boes, W., Van hamme, H.: Impact of temporal resolution on convolutional recurrent networks for audio tagging and sound event detection. In: Proceedings of the 7th Detection and Classification of Acoustic Scenes and Events 2022 Workshop (DCASE2022), pp. 1–5 (2022)
3. Casebeer, J., Wang, Z., Smaragdis, P.: Multi-view networks for multi-channel audio classification. In: Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 940–944 (2019)
4. Dang, A., Vu, T.H., Wang, J.C.: Acoustic scene classification using convolutional neural networks and multi-scale multi-feature extraction. In: Proceedings of the IEEE International Conference on Consumer Electronics, pp. 1–4 (2018)
5. Deshmukh, S., Raj, B., Singh, R.: Improving weakly supervised sound event detection with self-supervised auxiliary tasks. arXiv/2106.06858 (2021)
6. Fan, X., Jiang, W., Luo, H., Mao, W., Yu, H.: Instance hard triplet loss for in-video person re-identification. Appl. Sci. **10**(6), 1–18 (2020)
7. Gong, Y., Chung, Y.A., Glass, J.: AST: audio spectrogram transformer. In: Proceedings of the INTERSPEECH Conference, pp. 571–575 (2021)
8. John, V., Kawanishi, Y.: Multi-view video-based learning: leveraging weak labels for frame-level perception. arXiv/2403.11616 (2024)
9. Kong, Q., Xu, Y., Wang, W., Plumbley, M.D.: Sound event detection of weakly labelled data with CNN-transformer and automatic threshold optimization. IEEE/ACM Trans. Audio Speech Lang. Process. **28**, 2450–2460 (2020)
10. Mei, P., Yang, J., Zhang, Q., Huang, X.: A method of sound event localization and detection based on three-dimension convolution. In: Proceedings of the 7th International Conference on Image, Vision and Computing, pp. 872–878 (2022)
11. Mesaros, A., Heittola, T., Virtanen, T., Plumbley, M.D.: Sound event detection: a tutorial. IEEE Signal Process. Mag. **38**(5), 67–83 (2021)
12. Miyazaki, K., Komatsu, T., Hayashi, T., Watanabe, S., Toda, T., Takeda, K.: Weakly-supervised sound event detection with self-attention. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 66–70 (2020)
13. Park, C., Kim, D., Ko, H.: Sound event detection by pseudo-labeling in weakly labeled dataset. Sensors **21**(24) (2021)
14. Phan, H., et al.: Multi-view audio and music classification. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 611–615 (2021)
15. Tseng, S., Li, J., Wang, Y., Szurley, J., Metze, F., Das, S.: Multiple instance deep learning for weakly supervised audio event detection. arXiv/1712.09673 (2017)
16. Yasuda, M., Ohishi, Y., Saito, S., Harado, N.: Multi-view and multi-modal event detection utilizing transformer-based multi-sensor fusion. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 4638–4642 (2022)

SITAR: Semi-supervised Image Transformer for Action Recognition

Owais Iqbal[1](✉), Omprakash Chakraborty[1], Aftab Hussain[1], Rameswar Panda[2], and Abir Das[1]

[1] IIT-Kharagpur, Kharagpur, India
owais.iqbal@kgpian.iitkgp.ac.in, opckgp@iitkgp.ac.in, aftabh4004@kgpian.iitkgp.ac.in, abir@cse.iitkgp.ac.in
[2] MIT-IBM Watson AI Lab, Cambridge, USA
rpanda@ibm.com

Abstract. Recognizing actions from a limited set of labeled videos remains a challenge as annotating visual data is not only tedious but also can be expensive due to classified nature. Moreover, handling spatio-temporal data using deep 3D transformers for this can introduce significant computational complexity. In this paper, our objective is to address video action recognition in a semi-supervised setting by leveraging only a handful of labeled videos along with a collection of unlabeled videos in a compute efficient manner. Specifically, we rearrange multiple frames from the input videos in row-column form to construct super images. Subsequently, we capitalize on the vast pool of unlabeled samples and employ contrastive learning on the encoded super images. Our proposed approach employs two pathways to generate representations for temporally augmented super images originating from the same video. Specifically, we utilize a 2D image-transformer to generate representations and apply a contrastive loss function to minimize the similarity between representations from different videos while maximizing the representations of identical videos. Our method demonstrates superior performance compared to existing state-of-the-art approaches for semi-supervised action recognition across various benchmark datasets, all while significantly reducing computational costs. (Project page: https://cvir.github.io/projects/sitar)

Keywords: semi-supervised learning · contrastive learning · action recognition

1 Introduction

Video action recognition (VAR) in computer vision remains a long-standing challenge [3,11,19,33,35]. Earlier approaches relied on optical flow [17,27] or

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78192-6_8.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 114–130, 2025.
https://doi.org/10.1007/978-3-031-78192-6_8

3D convolutions using spatio-temporal kernels [33] for feature extraction. However, recent advancements have introduced transformer architectures [1,8] to address this task. A key challenge in achieving high accuracy for VAR lies in the scarcity of large-scale, meticulously labeled videos. The hunger for annotated data has further aggravated with the advent of 3D video transformers [1] which continuously surpass the performance by 2D vision transformers but requires significantly higher amount of labeled videos to perform. At the same time, 3D Transformers have more parameters, making it computationally more inefficient compared to ConvNets or 2D transformers. Though carefully annotating videos is expensive and labor-intensive, with the proliferation of video cameras and internet millions of videos without labels are available in the wild.

In scenarios with limited labeled training data alongside a substantial amount of unlabeled samples, semi-supervised learning (SSL) [43] has shown significant promise. Despite the inherent promise of semi-supervised video action recognition, it remains a relatively under-explored area compared to its fully supervised counterpart. A straightforward approach of applying image based semi-supervised learning techniques to videos might not be sufficient to bridge the performance gap. The reason being the additional temporal dimension in videos. However, if properly used, *temporal information* can be a friend instead of a foe to semi-supervised video action recognition. Barring a few canonical activities (*e.g.*, walking vs jogging) actions do not change if a video is played fast or slow. A robust video action recognition system should have the capability to identify actions regardless of the speed of the video. Recent works [10,38] demonstrate the effectiveness of training models to be invariant across different play-rates of the same actions. However, these approaches are still supervised and utilise Convolutional Neuron Networks (CNNs) to process the videos. Learning invariances among different versions of same samples as a means of self-supervision has been showing promising results with the emergence of contrastive learning. Studies such as [4,14] have exhibited superior performance compared to supervised learning approaches in image classification. This concept has been extended to videos [7,23], where [23] utilizes temporally distant clips from a video alongside spatial augmentations as positives, while negatives are selected from different videos.

Inspired by the success of utilizing both slow and fast versions of video for supervised action recognition and the success of contrastive learning frameworks [22,23], this work presents a new approach for semi-supervised video action recognition (VAR) that leverages unlabeled video data in both computationally and label efficient manner. Our method differs from traditional approaches by first rearranging video frames into informative super images [24]. Our approach initially transforms a 3D video into a 2D image by rearranging a sequence of input video frames into a super image based on a predetermined spatial layout. These super images capture both spatial and temporal information within video segments, enabling the model to learn representations efficiently. Like images, with super images too, an image transformer can be used.

To leverage temporal information, we introduce a two-pathway model. Each pathway processes temporally augmented versions of the same video (*e.g.*, slow and fast versions). In the fast pathway, the super image is formed using double the number of frames compared to the super image in the slow pathway. Despite this variation, these super images share the same semantic content when they come from the same unlabeled video. The model learns to maximize this semantic similarity. At the same time, the model tries to minimize the similarity between the representations coming from the super images of two different videos in these two pathways. It is achieved by using a contrastive loss [4] formulation. We extensively experiment with three benchmark datasets employing the Swin Transformer [21] as the backbone. Our approach **S**emi-Supervised **I**mage **T**ransformer for **A**ction **R**ecognition (`SITAR`), demonstrates superior performance outperforming previous approaches. Additionally, `SITAR` also exhibits better computational and parameter efficiency compared to existing semi-supervised video action recognition approaches.

2 Related Works

Video Action Recognition. Convolutional Neural Networks (CNNs) [9,10,34] and transformers [1,8,20] have emerged as dominant approaches for action recognition in videos. Although, 3D CNNs [9,10,34] have been dominant in earlier works, their large number of parameters necessitates vast training datasets. Additionally, CNNs limited receptive field hinders accurate motion modeling, hindering performance. Drawing inspiration from the success of transformers in natural language processing, vision-transformers based architectures like TimeSformer [2], MViT [8] and MViTv2 [20] have emerged as powerful alternatives achieving state-of-the-art performance and also efficient arcitectures [26]. Recent works also underscore the ability of transformers [32,40] on fewer data. However, these existing video action recognition approaches are supervised and rely on 3D operations greatly increasing the compute complexity. Our work addresses this challenge by utilizing a 2D Image Transformer in a semi-supervised setting for video action recognition. This strategy aims to not only achieve better performance but also significantly lower the computational requirements.

Semi-supervised learning. While extensive research has yielded successful semi-supervised learning methods in images, directly applying them to VAR proves suboptimal due to the non-exploration of temporal dynamics inherent in videos. Pioneering image-based approaches like Pseudo-Labeling established a foundation, which leverages the predicted confidence scores (softmax probabilities) from the model itself to generate pseudo-labels for unlabeled data. Subsequently, these pseudo-labels are employed to train the network along with a limited set of labeled data. Later works focused on improving pseudo-label quality, *e.g.*, UPS [25], FixMatch [29]. FixMatch utilizes weakly augmented unlabeled instances to generate pseudo-labels and ensures consistent predictions against their strongly augmented counterparts. FixMatch showed superior performance with its influence extending to detection [36] and segmentation [47].

Semi-supervised Video Action Recognition. Despite the emergence of several semi-supervised video action recognition approaches [5,15,28,41,47], most of them primarily extend image-based approaches to videos, ignoring the crucial temporal dynamics inherent in video data. Existing semi-supervised VAR approaches like VideoSSL [15] replaces a 2D ResNet with a 3D ResNet [13] but is limited to using pseudo-labels and distillation from an image CNN when it comes to exploiting unlabelled videos. The marginal gain over pure image-based baselines highlights the limitations of directly transferring image-based methods. With time, more sophisticated video-specific methods, such as TCL [28] have emerged, focusing on exploiting temporal information of the video. MvPL [41] and LTG [39] utilize multi-modal data such as optical flow or temporal gradient information, respectively, to generate high-quality pseudo-labels for training the network. Whereas, CMPL [42] introduces an auxiliary network that requires processing more frames during training, potentially increasing computational complexity. Additionally, all these prior approaches rely on 2D [28] or 3D convolutional neural networks [39,41,42], which can necessitate longer training times. The recent SOTA, SVFormer [40], introduces the exploration of 3D video transformers for semi-supervised action recognition. It incorporates a robust pseudo-labeling framework, utilizing an Exponential Moving Average (EMA) Teacher network, to effectively manage unlabeled video samples.

However, these approaches are computationally expensive. Our method addresses this by constructing informative super images from video frames, enabling efficient contrastive learning with a two-pathway model. This innovative approach significantly reduces computational costs while achieving superior performance on semi-supervised action recognition tasks.

3 Methodology

3.1 Problem Definition

This work addresses the challenge of semi-supervised action recognition in videos using an Image Transformer. The input to the model has two subset - Labeled Videos (D_l) and Unlabeled Videos (D_u). The labeled video set (D_l), comprises of N_l video-label pairs. Each pair (V^i, y^i) represents the i^{th} video and its corresponding activity label, where i ranges from 1 to N_l and each y^i belongs to the label set $Y = \{1, 2, ..., C\}$, representing C distinct action categories. Unlabeled Videos (D_u) comprises a significantly larger set N_u ($\gg N_l$) of videos without labels(U^i), where i ranges from 1 to N_u. we create two different versions of each unlabeled video by playing them at two different frame rates: fast and slow.

3.2 SITAR Framework

The proposed Semi-Supervised Image Transformer for Action Recognition(SITAR) framework, as illustrated in Fig. 2, processes video inputs through two distinct pathways: *primary* and *secondary*. The primary pathway handles fast super image S_f^i, while the secondary pathway handles slow super image S_s^i. Next we describe how exactly the super images for the two pathways are formed.

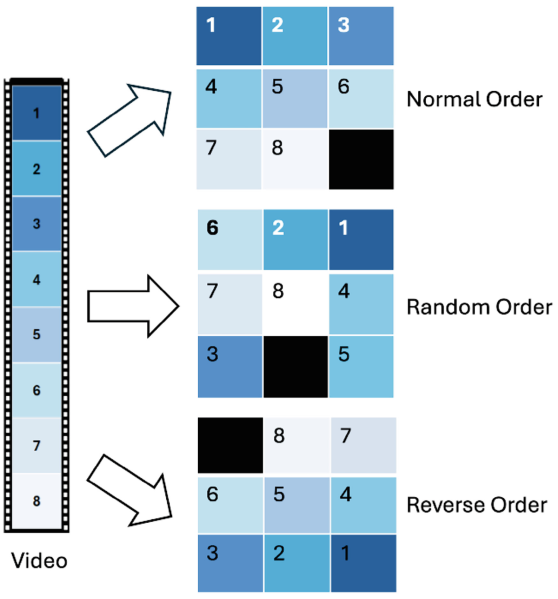


Fig. 1. Construction of Super Image. Aligning frames to form a grid using three different temporal orderings: `normal`, `random`, and `reverse`.

Super Image Construction: For the primary pathway, each video from U^i is represented as a set of M frames i.e., $U_f^i = \{F_{f,1}^i, F_{f,2}^i, \cdots, F_{f,M}^i\}$ sampled uniformly from consecutive non-overlapping segments following the approach in [37]. Similarly, for the secondary pathway, the same video is represented by N frames ($N < M$ and usually $N = \frac{M}{2}$) as $U_s^i = \{F_{s,1}^i, F_{s,2}^i, \cdots, F_{s,N}^i\}$, also sampled uniformly using the same method. The sampled frames are then transformed into purely $2D$ spatial patterns, generating super images. Inspired by the work [24], we create informative super images from the input video frames. This involves arranging the M or N input frames (respectively from the fast (U_f^i) or the slow version (U_s^i)) in a grid format as shown in Fig. 1. The grid size ($m \times m$) is determined by the square root of M, ensuring efficient representation. Any remaining empty spaces in the grid are filled with padding images. The resulting super image for the fast video is denoted as S_f^i, while the super image for the slow video is denoted as S_s^i. Both pathways utilize the same image transformer backbone, denoted by g(.). Next we describe the details of the different training stages within our framework.

Phase 1: Supervised Learning Initially, the Swin Transformer [21] (a 2D Image Transformer backbone) is trained exclusively with the small labeled data set (D_l) by passing it through the Primary Pathway (refer Fig. 2). The representation $g(VS^i)$ of the super image VS^i generated from video V^i in our framework is derived from the logits of the Swin Transformer. We minimize the conventional supervised cross-entropy loss $\mathcal{L}_{sup}$ on the labeled dataset as follows:

$$\mathcal{L}_{sup} = -\sum_{c=1}^{C} (y^i)_c \log(g(VS^i))_c \tag{1}$$

Phase 2: Semi-supervised Learning Starting with this initial backbone obtained under limited supervision, our objective is to develop a model capable

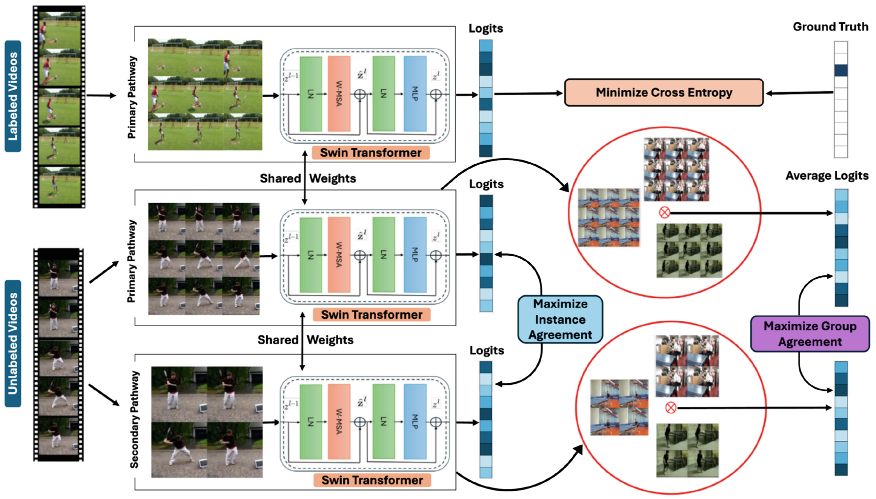


Fig. 2. **SITAR framework:** The proposed framework uses two pathways to process the unlabeled videos, namely, *Primary* and *Secondary*, using an image-transformer backbone and sharing the same weights. The Primary pathway is initially trained with limited labeled data. Then, we generate two versions of super images for the unlabeled videos, one fast, having more frames and other slow, having lower frames and pass them through Primary and Secondary pathways respectively. The training objective is to maximize the agreement between the output predictions of the two pathways. To achieve this, we employ two types of contrastive losses. First, an instance contrastive loss to align the representations of a given unlabeled super image across both the pathways. Second, a group contrastive loss to align the average representations of unlabeled super images grouped using pseudo-labels. During inference, only the Primary pathway is used to indentify actions. (Best viewed in color.)

of leveraging a vast pool of unlabeled videos to enhance activity recognition. To achieve this, we incorporate instance and group contrastive losses as follows:

Instance-Contrastive Loss We utilize the temporal augmentations within unlabeled videos, where each video is transformed into both slow (S_s^i) and fast (S_f^i) super images, and then the pairwise contrastive loss is enforced. In a mini-batch containing B unlabeled videos, the model is trained to align the representation $g(S_f^i)$ of the fast super image of video U^i with $g(S_s^i)$ from the slow super image, forming the positive pair. The remaining $B-1$ videos form negative pairs $g(S_f^i)$ and $g(S_p^k)$, where the representation of the k^{th} video can originate from either pathway ($p \in \{f, s\}$). As these negative pairs consists of different videos with distinct content, the representations from different pathways are diverged through the application of contrastive loss $\mathcal{L}_{ic}$.

$$\mathcal{L}_{ic}(S_f^i, S_s^i) = -\log \frac{h\big(g(S_f^i), g(S_s^i)\big)}{h\big(g(S_f^i), g(S_s^i)\big) + \sum\limits_{\substack{k=1 \\ p \in \{s,f\}}}^{B} \mathbb{1}_{\{k \neq i\}} h\big(g(S_f^i), g(S_p^k)\big)} \tag{2}$$

where, $h(\mathbf{u}, \mathbf{v}) = \exp\big(\frac{\mathbf{u}^\top \mathbf{v}}{||\mathbf{u}||_2 ||\mathbf{v}||_2}/\tau\big)$ represents the exponential of cosine similarity and τ denotes the temperature hyperparameter. The instance-contrastive loss is calculated for all positive pairs, namely $(S_f^i,\ S_s^i)$ and $(S_s^i,\ S_f^i)$, within the minibatch. This loss function aims to minimize similarity not only between different videos within each pathway but also across both pathways.

Group-Contrastive Loss Standard contrastive loss on unlabeled videos can struggle to capture high-level action semantics without class labels. As shown in Fig. 3, it might learn distinct representations for videos with the same action in absence of labels. To address this, we employ group contrastive loss. This approach uses pseudo-labels assigned to unlabeled videos based on the dominant action class (highest activation). Let $\hat{y}_f^i$ and $\hat{y}_s^i$ denote the pseudo-labels respectively of the fast (S_f^i) and slow (S_s^i) super images of the video U^i. Videos with the same pseudo-label within a pathway form a group. The group's representation is the average of its member super image representations (detailed below). This strategy encourages the model to learn more consistent representations for super images with similar actions.

$$R_p^l = \frac{\sum\limits_{i=1}^{B} \mathbb{1}_{\{\hat{y}_p^i = l\}} g(U_p^i)}{T} \tag{3}$$

where $\mathbb{1}$ is a binary function that yields a value of 1 for super images within pathway $p \in \{f, s\}$ whose pseudo-label matches class $l \in Y$. T represents the total number of such super images (with pseudo-label l) in pathway p within the minibatch. Considering the strong agreement between two groups with identical labels in both pathways, it's crucial for these groups to demonstrate comparable features in the feature space. Consequently, within the group-contrastive objective, all pairs (R_f^l, R_s^l) serve as positive pairs, whereas negative pairs consist of pairs (R_f^l, R_p^m) where $p \in \{f, s\}$ and $m \in Y \setminus l$, ensuring that the constituent groups differ in at least one pathway

$$\mathcal{L}_{gc}(R_f^l, R_s^l) = -\log \frac{h(R_f^l, R_s^l)}{h(R_f^l, R_s^l) + \sum\limits_{\substack{m=1 \\ p \in \{s,f\}}}^{C} \mathbb{1}_{\{m \neq l\}} h(R_f^l, R_p^m)} \tag{4}$$

Like instance-contrastive loss, we compute the group-contrastive loss for all positive pairs (R_f^l, R_s^l) and (R_s^l, R_f^l) over the minibatch. This combined loss, along with the supervised loss $\mathcal{L}_{sup}$ on the labeled data (D_l), forms the overall objective function $\mathcal{L}$ used to train our model.

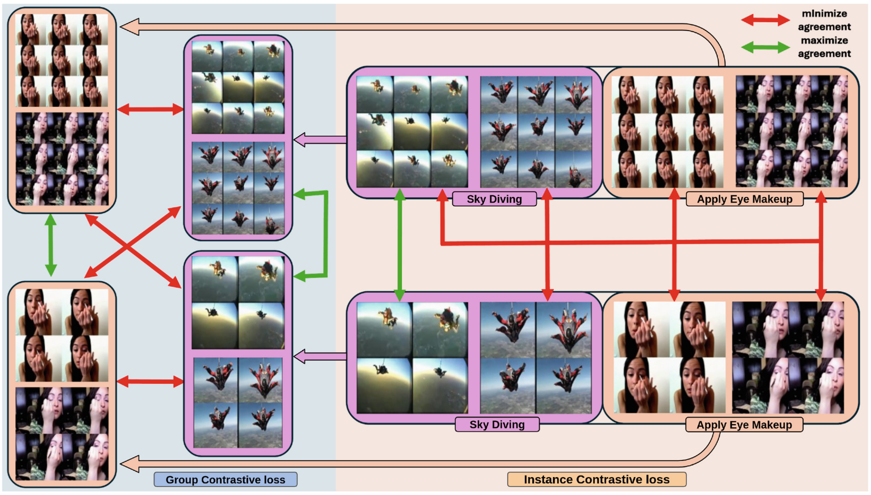


Fig. 3. Instance contrastive loss vs Group contrastive loss. SITAR employs two different contrastive losses to leverage on the unlabeled super images. The Instance contrastive loss maximized the agreement between two instances of the same videos which minimizing the agreement with the other videos in a given mini-batch. This risks the same action samples of the mini-batch to be inadvertently pushed apart (right). To mitigate this we employ a Group contrastive loss, which first groups videos with the same activity class (left) as predicted by high-confidence pseudo-labels. Then the average representation is obtained for each group and the contrastive learning policy is applied at this group level. (Best viewed in color.)

$$\mathcal{L} = \mathcal{L}_{sup} + \gamma * \mathcal{L}_{ic} + \beta * \mathcal{L}_{gc} \tag{5}$$

where, γ is the weight assigned to the instance-contrastive loss and β is the weight of the group-contrastive loss.

4 Results and Discussions

In this section, we first present the experimental settings (Section 4.1). Building on some of the previous works [39,40,42], we show results using different amount of labeled data across three datasets (Section 4.2). Furthermore, we conduct extensive ablation experiments in Section 4.3. For consistency, we utilize the same splits of data as those used by the authors in SVFormer [40].

4.1 Experimental Settings

Datasets. Kinetics-400 [16] is an extensive dataset tailored for human action recognition, containing over 300K clips distributed across 400 distinct classes of human actions. Following the standard practice [40–42], we experimented with two different scenarios in Kinetics-400. These two scenarios assume only 1% and 10% of all the videos are labeled while the rest are unlabeled. This amounts to 6 and 60 labeled training videos per category in this dataset. UCF-101 [30] is another commonly used human action recognition dataset, encompassing a diverse range of categories such as human-human interaction, human-object interaction, playing musical instruments, daily sports *etc.* With $13,320$ video samples, UCF-101 is spread across 101 classes. Consistent with CMPL [42], we assume only 1% and 10% of all the videos to be labeled (resulting in 1 and 10 labeled samples respectively per category) in this dataset also. Additionally, HMDB-51 [18] comprises human motion data encompassing a variety of actions, including facial expressions combined with object manipulation, general body movements, and movements related to human interaction. With 51 categories and a total of $6,766$ videos, HMDB-51 is a comparatively smaller dataset. Adhering to the experimental framework established by LTG [39] and VideoSSL [15], we explore three scenarios assuming 40%, 50%, and 60% of the data to be labeled.

Baselines. We benchmark our method against several baselines and existing semi-supervised VAR approaches. Initially, we investigate a supervised baseline where we report the performance of training a 3D-ResNet-50 only on the labeled data. This acts as the lower bound of performance. Second, we compare with state-of-the-art semi-supervised learning approaches, including FixMatch (NeurIPS'20) [29], VideoSSL (WACV'21) [15], TCL (CVPR'21) [28], ActorCutMix (CVIU'21) [46], MvPL (ICCV'21) [41], CMPL (CVPR'22) [42], LTG (CVPR'22) [39], TACL (TCSVT'22) [31], L2A (ECCV'22) [12] and SVFormer (CVPR'23) [40]. Notably, most of these are computation heavy as 3D videos are processed, unlike our method, which solely relies on 2D Image processing.

Implementation Details. We employ uniform sampling to generate video inputs for our models, dividing each video into multiple segments of equal length. In `SITAR`, we employ uniformly sampled 8-frame segments for the fast super image and 4-frame segments for the slow super image. These images are then fed into the primary and secondary pathways, respectively. The super image size remains constant at (572×572) for both the primary and auxiliary pathways. As a result, the frame size in the fast and slow super images is adjusted to 192 and 288 each side, respectively. Note that initially when the model is trained only with the handful of labeled videos, we pass 8-frame super images from them via the primary pathway. We start with Swin-B and Swin-S backbones [21] pretrained on ImageNet-$21K$ [6]. Initially, we apply random scaling and cropping for data augmentation. Additionally, for Phase-1 of supervised learning, we employ Mixup [45] and CutMix [44] with the respective mixing coefficients of 0.8 and 1.0, respectively following the work [24]. The drop path is set with a rate of 0.1 and label smoothing at a rate of 0.1. Training is conducted using NVIDIA V100 GPUs. For Phase 1 and Phase 2 (refer Sec. 3.2), we conduct experiments for 25 and 50 epochs, respectively, across all datasets. We used AdamW optimizer with a weight decay of 0.05 along with a cosine learning rate scheduler having a base learning rate of 0.0001. For training, we utilize a mini-batch having L_b labeled samples and $\mu \times L_b$ unlabeled samples. We set the values of μ and τ (refer Eqn. 2) to 4 and 0.5 respectively while the γ and β values (refer Eqn. 5) are set to 0.6 and 1, respectively, unless otherwise stated. During inference, only the primary pathway is used with 8 frame super images as inputs. Additional dataset details and experimental results are provided in the supplementary material.

Table 1. Comparisons of performance on UCF-101 and Kinetics-400. We report the top-1 accuracy of SITAR along with different state-of-the-art approaches over two different labeled data proportions. As observed, SITAR achieves the best performance in both the datasets over the different amounts of labeled data. Also, SITAR significantly reduces the compute needs as it uses much less model parameters. The values in **bold** denote the highest top-1 accuracy across all the specified models, respectively.

Method	Backbone	Params	w \ ImgNet	UCF-101		Kinetics-400	
				1%	10%	1%	10%
Supervised	3D-ResNet-50		✓	6.5	32.4	4.4	36.2
FixMatch (NeurIPS'20) [29]	SlowFast-R50		✓	16.1	55.1	10.1	49.4
VideoSSL(WACV'21) [15]	3D-ResNet-18		✓	-	42.0	-	33.8
TCL (CVPR'21) [28]	TSM-ResNet-18			-	-	8.5	-
ActorCutMix (CVIU'21) [46]	R(2+1)D-34		✓	-	53.0	9.02	33.8
MvPL (ICCV'21) [41]	3D-ResNet-50			22.8	80.5	17.0	58.2
CMPL (CVPR'22) [42]	R50+R50-1/4		✓	25.1	79.1	17.6	58.4
LTG (CVPR'22) [39]	3D ResNet 18			-	62.4	9.8	43.8
TACL(TCSVT'22) [31]	3D-ResNet-50		✓	-	55.6	-	-
L2A (ECCV'22) [12]	3D-ResNet-18		✓	-	60.1	-	-
SVFormer-S(CVPR'23) [40]	TimeSformer-S*	81M	✓	31.4	79.1	32.6	61.6
SITAR-S(Ours)	Swin-S	49M	✓	**37.9**	**81.8**	**36.7**	**64.1**
SVFormer-B(CVPR'23) [40]	TimeSformer(Default)	121M	✓	46.3	86.7	**49.1**	**69.4**
SITAR-B(Ours)	Swin-B	87M	✓	**47.0**	**87.1**	39.0	66.5

*Following SVFormer, as TimeSformer only have ViT-B models, SVFormer-S model is implemented from ViT-S.

4.2 Main Results

We evaluate our SITAR's performance on three benchmark datasets: Kinetics-400 [16], UCF-101 [30], and HMDB-51 [18]. As can be seen in Tables 1 and 2 our SITAR-S model consistently achieves superior performance on these datasets compared to the state-of-the-art, SVFormer-S [40], while utilizing significantly less model parameters. When using only 1% labeled data, our SITAR-S outperforms SVFormer-S by a significant margin of 6.5% on UCF-101 and 4.1% on Kinetics-400 (Table 1). This trend continues at the 10% labeled data setting,

Table 2. Comparisons with state-of-the-art methods on HMDB51. We report the top-1 accuracy of SITAR along with different state-of-the-art baselines over three different labeled data proportions. As observed, SITAR achieves the best performance over the different amounts of labeled data.

Method	Backbone	40%	50%	60%
VideoSSL [15]	3D-R18	32.7	36.2	37.0
ActorCutMix [46]	R(2+1)D-34	32.9	38.2	38.9
MvPL [41]	3D-R18	30.5	33.9	35.8
LTG [39]	3D-R18	46.5	48.4	49.7
TACL [31]	3D-R18	38.7	40.2	41.7
L2A [12]	3D-R18	42.1	46.3	47.1
SVFormer-S [40]	TimeSformer-S*	56.2	58.2	59.7
SITAR-S(Ours)	Swin-S	**60.6**	**62.6**	**65.1**
SVFormer-B [40]	TimeSformer(Default)	61.6	64.4	68.2
SITAR-B(Ours)	Swin-B	**63.4**	**65.5**	**68.2**

*Following standard procedure of SVFormer, as TimeSformer only have ViT-B models, SVFormer-S model is implemented from ViT-S.

with improvements of 2.7% and 2.5% for UCF-101 and Kinetics-400, respectively. Similarly, on HMDB-51 (Table2), SITAR-S surpasses SVFormer-S by a substantial margin across all scenarios using different portions of labeled data (40%, 50%, and 60%).

Our larger model, SITAR-B, is at par with SVFormer-B while maintaining a significant parameter efficiency. In UCF-101, SITAR-B performs slightly better in both 1% and 10% labeled data settings compared to SVFormer-B. Although our model is only second to SVFormer-B Kinetics-400, it has a significantly lower number of parameters compared to SVFormer-B. Specifically, our model utilizes 28% fewer parameters than the highest-performing model. On HMDB-51, SITAR-B exhibits improvements of 1.8% and 1.1% in the 40% and 50% settings, respectively, while being at par at 60%. These results highlight the effectiveness of SITAR in achieving high accuracy with reduced model complexity.

4.3 Ablation Results

We perform several ablation studies on HMDB-51 dataset to evaluate the effectiveness of the different components of our SITAR framework.

Impact of Group Contrastive Loss. The importance of group contrastive loss in capturing high-level action semantics is evident from an ablation where we exclude Group Contrastive Loss from our framework (refer Section 3.2) leading to a drop in top-1 accuracy from 60.6% to 60.1% shown in Table 3.

Importance of Contrastive Loss. To evaluate the efficacy of contrastive loss, we compared it to the pseudo-label consistency loss used in FixMatch [29]. As shown in Table 3, training with our contrastive loss achieved a significant improvement in top-1 accuracy on the HMDB-51 dataset, outperforming pseudo-label consistency loss by 2.13%. This finding highlights the effectiveness of contrastive loss in capturing meaningful relationships between unlabeled videos.

Table 3. Ablation Studies on HMDB-51. Numbers show top-1 accuracy with 40% labeled data.

Approch	Top-1 Accuracy
SITAR-S w/o Group-Contrastive Loss	60.1
SITAR-S w/ Pseudo-Label Consistency Loss	58.5
SITAR-S (Ours)	**60.6**

Table 4. Effect of different Frame rates in Super Image. We evaluate on HMDB-51 and report the top-1 accuracy by changing the layout to 4 × 4 and 3 × 3, incorporating 16 and 8 frames in the respective pathways. This was achieved in two ways: by keeping the size of super image constant and by increasing the super image size. The effect of computation on increasing the super image size is depicted in column 'Flops'.

Model	#Frames in Fast SI*	#Frames in Slow SI*	SIze of SI*	Flops(G)	Top-1 Accuracy
SITAR-S	4	3	576	61	59.4
SITAR-S	4	3	768	109	61.9
SITAR-S(Ours)	3	2	576	61	60.6

*SI=Super Image

Ablations on Super Images. In Phase-2 of training, our framework employs super images of layouts 3 × 3 and 2 × 2 in the Primary and Secondary pathways, respectively. Here, we explore the impact of changing the layout to 4 × 4 and 3 × 3, incorporating 16 and 8 frames in the respective pathways (with one extra padded frame in each case). This can be achieved in two ways: Firstly, by increasing the number of frames to 16 and 8 while keeping the super image size constant (*i.e.*, 576 × 576), thus reducing the size of each frame. Secondly, by maintaining the size of each frame constant and increasing the super image size (*i.e.*, 768 × 768). The results in Table 4 demonstrate that maintaining the super image size constant while increasing the number of frames leads to information loss due to the reduction in frame size and reduces Top-1 accuracy (shown in the first row of Table 4). Conversely, in the second experiment, increasing the super image size enhances the Top-1

Table 5. Effect of temporal Order in Super Image. We report the top-1 accuracy for HMDB-51 using 40% labeled data with reverse, random and normal order.

Approch	Top-1 Accuracy
SITAR-S w/ reverse frame order	59.6
SITAR-S w/ random frame order	58.9
SITAR-S (Ours)	**60.6**

accuracy, albeit at the expense of increased computation (model flop going from 61G to 109G) (shown in the second row of Table 4).

In another ablation experiment we examine the effect of the temporal order of frames in the super image. We evaluate our model's performance across three input types with different temporal orders: normal, random, and reverse (refer Fig. 1). In the normal order, the temporal sequence of the video remains unchanged during super image generation. Conversely, for the reverse and random orders, sampled frames are arranged in the super image in reverse and random orders, respectively. Table 5 showcases the impact of altering the input order on the performance of the HMDB-51 dataset. These findings exemplifies the significance of having normal input frame order in model learning for action recognition tasks.

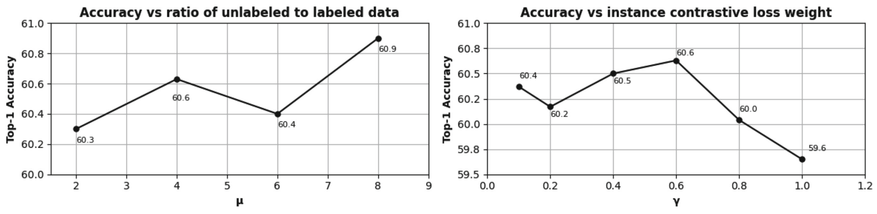


Fig. 4. Effect of hyperparameters on HMDB51, (Left) Varying the ratio of unlabeled data to the labeled data (μ), (Right) Varying the instance-contrastive loss weight (γ)

Impact of Hyperparameters. We investigate the impact of the ratio of unlabeled data to labeled data (μ) and note that setting μ to $2, 4, 6, 8$ with a fixed $\gamma = 0.6$ yields similar outcomes on HMDB-51 (see Fig. 4). However, given the high computational demands associated with scaling μ, we opt to set it to 4 across all experiments to strike a balance between efficiency and accuracy in our model. Additionally, we observe that the weight of the instance-contrastive loss (γ) influences performance in semi-supervised learning. We check for γ by setting it to $0.1, 0.2, 0.4, 0.8, 1$. We find that $\gamma = 0.6$ gives the best result and we used this value of γ throughout in our experiments.

Larger backbones. We conducted additional experiments with a larger swin transformer backbone Swin-L resulting in, SITAR-L, having approximately 195M parameters. We report that the results for SITAR-L which outperforms SVFormer-B (refer Table 6). SITAR-L demonstrates superior performance across different percentages of labeled data in different datasets (barring the scenario when 1% of Kinetics-400 data is used with labels). Note that SITAR-L is also able to outperform SVFormer-B in Kinetics-400 10% labeled data setting which SITAR-B was falling short (refer Table 1).

Table 6. Comparison of performance on UCF-101, HMDB-51 and Kinetics-400 using Swin-L backbone We report the top-1 accuracy of SITAR along with state-of-the-art approach SVFormer-B over different labeled data proportions. As observed, SITAR achieves the best performance in all the datasets over different amounts of labeled data barring Kinetics-400 with 1% labeled data.

Method	Params	UCF-101		HMDB-51			K-400	
		1%	10%	40%	50%	60%	1%	10%
SVFormer-B(CVPR'23) [40]	121M	46.3	86.7	61.6	64.4	68.2	49.1	69.4
SITAR-B(Ours)	87M	47.0	87.1	63.4	65.5	68.2	39.0	66.5
SITAR-L(Ours)	195M	47.0	87.2	64.4	67.7	68.6	44.0	70.1

Flops and Parameters Comparison. In Table 7, we compare the total number of trainable parameters and the total number of operations (FLOPs) between SITAR and the next best approach, SVFormer [40]. It is evident that SITAR-B utilizes 28% fewer parameters compared to SVFormer-B, and the flops count is notably reduced by 45.9%. Despite having fewer parameters and a lower flop count, our model surpasses the Top-1 Accuracy for UCF-101 and HMDB-51 across all labeled data setting (refer Table 1 and 2).

Table 7. Comparison of parameters and FLOPs. We show the comparison of parameters and total operations between our SITAR and next best method SVFormer.

Method	Backbone	Params(M)	FLops(G)
SVFormer-B [40]	TimeSformer(Default)	121	196
SITAR-B(Ours)	Swin-B	**87**	**106**

5 Conclusions

This paper presents a novel approach for semi-supervised action recognition in videos using 2D image transformer. By condensing input frames into a single super image, our method provides a straightforward yet potent means of repurposing image classifiers for action recognition significantly reducing the compute cost. Employing a temporal contrastive learning framework maximizes the similarity between encoded representations of unlabeled videos transformed into super images at varying speeds, while minimizing similarity between different videos. Through the utilization of contrastive loss and exploration of high-level action semantics within video groups, our approach greatly mitigates the need of video annotations. Our work outperforms several competitive approaches across three standard benchmark datasets. These findings magnifies the potential of super images and temporal contrastive learning in advancing video understanding tasks, warranting further exploration and research in this area.

Acknowledgements. This work was partially supported by the SERB Grant CRG/2023/005010. We also acknowledge support from the MIT-IBM Watson AI Lab, which—through MIT Satori Supercomputer, contributed the computational resources necessary to conduct the experiments in this paper.

References

1. AnuragArnab, Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: IEEE International Conference on Computer Vision. pp. 6836–6846 (2021)
2. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: International Conference on Machine Learning. vol. 139, pp. 813–824. PMLR (2021)
3. Brand, M., Oliver, N., Pentland, A.: Coupled Hidden Markov Models for Complex Action Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 994–999 (1997)
4. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning. pp. 1597–1607. PMLR (2020)
5. Dass, S.D.S., Barua, H.B., Krishnasamy, G., Paramesran, R., Phan, R.C.W.: ActNetFormer: Transformer-ResNet Hybrid Method for Semi-Supervised Action Recognition in Videos. arXiv preprint arXiv:2404.06243 (2024)
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255. Ieee (2009)
7. Dorkenwald, M., Xiao, F., Brattoli, B., Tighe, J., Modolo, D.: SCVRL: Shuffled Contrastive Video Representation Learning. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 4132–4141 (June 2022)
8. Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., Feichtenhofer, C.: Multiscale vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 6824–6835 (October 2021)
9. Feichtenhofer, C.: X3d: Expanding architectures for efficient video recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 203–213 (2020)
10. Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019)
11. Gaur, U., Zhu, Y., Song, B., Roy-Chowdhury, A.: A "String of Feature Graphs" Model for Recognition of Complex Activities in Natural Videos. In: IEEE International Conference on Computer Vision. pp. 2595–2602 (2011)
12. Gowda, S.N., Rohrbach, M., Keller, F., Sevilla-Lara, L.: Learn2augment: learning to composite videos for data augmentation in action recognition. In: European Conference on Computer Vision. pp. 242–259. Springer (2022)
13. Hara, K., Kataoka, H., Satoh, Y.: Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6546–6555 (2018)

14. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9729–9738 (2020)
15. Jing, L., Parag, T., Wu, Z., Tian, Y., Wang, H.: Videossl: Semi-supervised learning for video classification. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 1110–1119 (January 2021)
16. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The Kinetics Human Action Video Dataset. arXiv preprint arXiv:1705.06950 (2017)
17. Ke, Y., Sukthankar, R., Hebert, M.: Spatio-temporal Shape and Flow Correlation for Action Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8. IEEE (2007)
18. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: Hmdb: a large video database for human motion recognition. In: 2011 International conference on computer vision. pp. 2556–2563. IEEE (2011)
19. Laxton, B., Lim, J., Kriegman, D.: Leveraging Temporal, Contextual and Ordering Constraints for Recognizing Complex Activities in Video. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8 (2007)
20. Li, Y., Wu, C.Y., Fan, H., Mangalam, K., Xiong, B., Malik, J., Feichtenhofer, C.: Mvitv2: Improved multiscale vision transformers for classification and detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4804–4814 (2022)
21. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: the IEEE/CVF International Conference on Computer Vision. pp. 10012–10022 (2021)
22. Pan, T., Song, Y., Yang, T., Jiang, W., Liu, W.: Videomoco: Contrastive video representation learning with temporally adversarial examples. In: the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11205–11214 (2021)
23. Qian, R., Meng, T., Gong, B., Yang, M.H., Wang, H., Belongie, S., Cui, Y.: Spatiotemporal contrastive video representation learning. In: the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6964–6974 (2021)
24. Quanfu Fan, Richard Chen, R.P.: Can an image classifier suffice for action recognition? In: International Conference on Learning Representations (ICLR) (2022)
25. Rizve, M.N., Duarte, K., Rawat, Y.S., Shah, M.: In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. arXiv preprint arXiv:2101.06329 (2021)
26. Ryali, C., Hu, Y.T., Bolya, D., Wei, C., Fan, H., Huang, P.Y., Aggarwal, V., Chowdhury, A., Poursaeed, O., Hoffman, J., et al.: Hiera: A hierarchical vision transformer without the bells-and-whistles. In: International Conference on Machine Learning. pp. 29441–29454. PMLR (2023)
27. Simonyan, K., Zisserman, A.: Two-stream Convolutional Networks for Action Recognition in Videos. Advances in Neural Information Processing Systems **27** (2014)
28. Singh, A., Chakraborty, O., Varshney, A., Panda, R., Feris, R., Saenko, K., Das, A.: Semi-supervised action recognition with temporal contrastive learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10389–10399 (2021)
29. Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C.A., Cubuk, E.D., Kurakin, A., Li, C.L.: Fixmatch: Simplifying semi-supervised learning with consistency and confidence. Adv. Neural. Inf. Process. Syst. **33**, 596–608 (2020)

30. Soomro, K., Zamir, A., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. CoRR (2012)
31. Tong, A., Tang, C., Wang, W.: Semi-supervised action recognition from temporal augmentation using curriculum learning. IEEE Trans. Circuits Syst. Video Technol. **33**(3), 1305–1319 (2022)
32. Tong, Z., Song, Y., Wang, J., Wang, L.: Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. Adv. Neural. Inf. Process. Syst. **35**, 10078–10093 (2022)
33. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4489–4497 (2015)
34. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6450–6459 (2018)
35. Veeraraghavan, A., Srivastava, A., Roy-Chowdhury, A.K., Chellappa, R.: Rate-invariant Recognition of Humans and their Activities. Transactions on Image Processing **18**(6), 1326–1339 (2009)
36. Wang, H., Cong, Y., Litany, O., Gao, Y., Guibas, L.J.: 3dioumatch: Leveraging iou prediction for semi-supervised 3d object detection. In: the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14615–14624 (2021)
37. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: European Conference on Computer Vision. pp. 20–36. Springer (2016)
38. Xiao, F., Lee, Y.J., Grauman, K., Malik, J., Feichtenhofer, C.: Audiovisual slowfast networks for video recognition. arXiv preprint arXiv:2001.08740 (2020)
39. Xiao, J., Jing, L., Zhang, L., He, J., She, Q., Zhou, Z., Yuille, A., Li, Y.: Learning from temporal gradient for semi-supervised action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3252–3262 (2022)
40. Xing, Z., Dai, Q., Hu, H., Chen, J., Wu, Z., Jiang, Y.G.: Svformer: Semi-supervised video transformer for action recognition. In: CVPR (2023)
41. Xiong, B., Fan, H., Grauman, K., Feichtenhofer, C.: Multiview pseudo-labeling for semi-supervised learning from video. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7209–7219 (2021)
42. Xu, Y., Wei, F., Sun, X., Yang, C., Shen, Y., Dai, B., Zhou, B., Lin, S.: Cross-model pseudo-labeling for semi-supervised action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2959–2968 (2022)
43. Yang, X., Song, Z., King, I., Xu, Z.: A survey on Deep Semi-supervised Learning. IEEE Trans. Knowl. Data Eng. **35**(9), 8934–8954 (2022)
44. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6023–6032 (2019)
45. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: Mixup: Beyond Empirical Risk Minimization. In: International Conference on Learning Representations (2018)
46. Zou, Y., Choi, J., Wang, Q., Huang, J.B.: Learning representational invariances for data-efficient action recognition. Comput. Vis. Image Underst. **227**, 103597 (2023)
47. Zou, Y., Zhang, Z., Zhang, H., Li, C.L., Bian, X., Huang, J.B., Pfister, T.: Pseudoseg: Designing pseudo labels for semantic segmentation. In: International Conference on Learning Representations (2021)

Enhancing Graph Topology and Clustering Quality: A Modularity-Guided Approach

Xiaotian Zhuang, Yongyu Wang(✉), Shiqi Hao, and Xiaoyang Wang

JD Logistics, Beijing101111, China
wangyongyu1@jd.com

Abstract. Current modularity-based community detection algorithms attempt to find cluster memberships that maximize modularity within a fixed graph topology. Diverging from this conventional approach, our work introduces a novel strategy that employs modularity to guide the enhancement of both graph topology and clustering quality through a maximization process. Specifically, we present a modularity-guided approach for learning sparse graphs with high modularity by iteratively pruning edges between distant clusters, informed by algorithmically generated clustering results. To validate the theoretical underpinnings of modularity, we designed experiments that establish a quantitative relationship between modularity and clustering quality. Extensive experiments conducted on various real-world datasets demonstrate that our method significantly outperforms state-of-the-art graph construction methods in terms of clustering accuracy. Moreover, when compared to these leading methods, our approach achieves up to a hundredfold increase in graph construction efficiency on large-scale datasets, illustrating its potential for broad application in complex network analysis.

Keywords: Modularity · Graph · Clustering

1 Introduction

Graph-based methodologies are pivotal in numerous machine learning and data mining tasks, owing to the graph's inherent strength in encapsulating the intricate structures and interrelationships within data sets. The efficacy of a graph-based approach hinges on the integrity of the graph itself, as the quality of the graph profoundly influences the algorithm's solution quality. Over the past decades, a variety of graph construction (or learning) techniques have been introduced.

The k-nearest neighbor (k-NN) graph, for instance, is widely adopted due to its simplicity and effectiveness in capturing the local manifold structure of data. In a k-NN graph, each node is linked to its k nearest neighbors, which helps to maintain robustness against outliers. Nevertheless, the fixed-size neighborhood used in k-NN graphs can restrict their ability to represent the global

Y. Wang and X. Zhuang—These authors contributed equally and are co-first authors.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 131–142, 2025.
https://doi.org/10.1007/978-3-031-78192-6_9

manifold structure adequately. The ϵ-neighborhood graph offers an alternative by connecting each node to all other nodes within a specified distance ϵ. However, selecting an appropriate ϵ value is challenging and sometimes impractical, especially when clusters within the data vary significantly in size. [1] suggested enhancing the k-NN graph by extracting consensus information to prune noisy edges, where edges with a consensus value below a certain threshold are eliminated. While this method mitigates the effect of noisy connections, it may also inadvertently discard valuable structural information. More recent research has explored the use of graph signal processing (GSP) techniques to refine graph learning methods. These approaches aim to estimate sparse graph Laplacians more accurately. For example, [2] introduced a method that constrains the precision matrix to be a graph Laplacian and maximizes the posterior estimate of a Gaussian Markov Random Field (GMRF), incorporating an $l1$-regularization term to maintain graph sparsity. Additionally, [3] developed an approach that employs approximate nearest-neighbor methods to decrease the number of variables in the optimization process, thereby enhancing computational efficiency. However, these advanced Laplacian estimation techniques typically require computational time on the order of $O(n^2)$ for each iteration, which poses a significant challenge for their application in large-scale real-world problems. Therefore, efficiently constructing a high-quality graph remains a challenging task.

In this paper, we propose leveraging modularity to enhance the quality of the graph. Modularity is a key concept in community detection algorithms. In these algorithms [4–6], for a given community network, the aim is to find a clustering division that maximizes modularity as the outcome of community partitioning. Unlike the conventional use of modularity in these methods, we introduce a framework that utilizes modularity to optimize the graph's topological structure. Specifically, we use modularity to iteratively identify and remove redundant and incorrect edges. In our framework, the quality of graph-based clustering algorithms and the graph's topological structure mutually reinforce each other, improving alternately. When the iterative process can no longer increase modularity, we output the final graph topology. Running graph clustering algorithms on this optimized graph topology can significantly enhance clustering accuracy. Experiments on multiple real-world benchmark datasets of large scale indicate that, compared to state-of-the-art graph construction methods, our approach can build graph topologies that significantly improve the accuracy of graph clustering algorithms. Moreover, our graph construction is hundreds of times more efficient than the state-of-the-art methods.

Another contribution of this article is the empirical quantification of the relationship between modularity and clustering accuracy through the method we propose, thus substantiating the modularity theory. In community detection tasks, most clustering divisions lack a ground truth, or they involve multi-label partitions with extremely complex overlap relationships[7,8]. In many research papers on community detection, the value of modularity is directly used as the quantitative evaluation [4–6], which is clearly not very persuasive. Therefore, how to quantitatively evaluate the effectiveness of community detection algorithms remains a highly controversial issue, with no consensus reached thus far [7,8]. This paper introduces a modularity-based graph clustering algorithm that enables the use of standard datasets with a unique ground truth label for each sample to obtain the quantitative relationship between modularity and clustering accuracy. Experimental results on the Pendigits dataset using our algorithm indicate that modularity and clustering accuracy increase synchronously.

2 Preliminary

2.1 Modularity

Modularity, denoted by Q, is a metric used in community detection to assess the quality of a network division. It measures the strength of division of a network into communities by comparing the density of edges within communities to the density of edges between communities. The modularity Q is defined as [4]:

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j), \tag{1}$$

where A_{ij} represents the adjacency matrix of the network, with $A_{ij} = 1$ if there is an edge between nodes i and j, and $A_{ij} = 0$ otherwise. The degree of node i is represented by k_i, and m is the total number of edges in the network. The Kronecker delta function $\delta(c_i, c_j)$ is 1 if nodes i and j belong to the same community, and 0 otherwise.

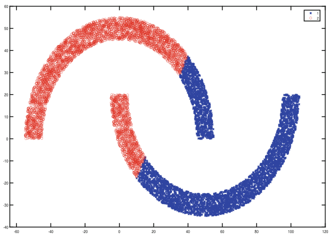
Community detection algorithms such as the Louvain Method [5] and the Leiden Method [6] leverage modularity to identify meaningful communities within complex networks. Modularity serves as a quality measure that quantifies the degree to which a network can be divided into distinct communities. These methods iteratively optimize modularity by reassigning nodes to communities, with the goal of maximizing the difference between observed and expected intra-community connections. This process results in the detection of cohesive groups of nodes that exhibit higher connectivity within their respective communities compared to what would be expected by chance. These algorithms efficiently uncover community structures, making them valuable tools in network analysis, social sciences, biology, and other domains where understanding network organization is essential.

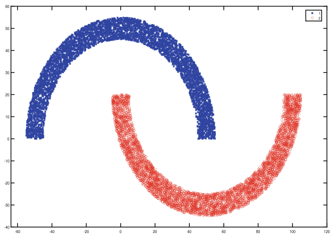
2.2 The Significance of Graph Topology in Clustering

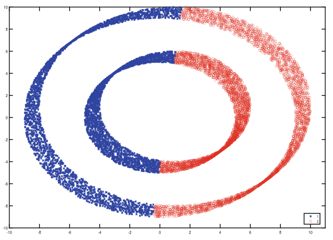
Graph topology plays a crucial role in detecting non-convex and linearly non-separable patterns. To illustrate this, we use the most fundamental clustering task in artificial intelligence as an example.

The widely adopted k-means algorithm, a non-graphical clustering method, primarily relies on distance metrics to minimize the total distances between data points and their respective cluster centroids. However, it often falls short in providing satisfactory clustering results for complex datasets, exemplified by well-known scenarios like the two moons and two circles datasets. As shown in Figure 1, in the two moons dataset, where the ground truth clusters correspond to two slightly entangled moon-shaped regions, k-means fails to produce accurate clustering results. Similarly, in the two circles dataset, where the ground truth clusters correspond to two concentric circles, k-means yields incorrect results.

In contrast, graph-based algorithms, such as spectral clustering, leverage the connectivity information embedded in the graph topology. This allows them to overcome the limitations of k-means and generate more accurate clustering results for complex, non-linear data distributions.


(a) k-means of the two moons data set


(b) Spectral clustering of the two moons data set


(c) k-means of the two circles data set

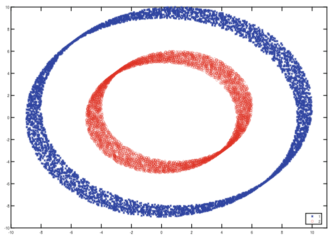

(d) Spectral clustering of the two circles data set

Fig. 1. Clustering results generated by non-graphical and graph-based clustering methods.

3 Methods

3.1 Overview

In broad terms, our approach draws inspiration from recent modularity-based community detection methods. However, it deploys modularity in a distinct manner to accomplish a unique objective. Unlike existing methods that employ modularity to guide node clustering within a fixed and predefined graph topology, our method introduces a novel perspective.

Existing modularity-based algorithms such as Leiden and Louvain algorithms iteratively adjust the cluster membership of individual nodes, seeking to reach the point of maximal modularity. Once this maximum modularity is achieved, the algorithm terminates, and the resulting cluster memberships are considered the final clustering results. In contrast, our approach deviates from the conventional paradigm by redefining the role of modularity within the clustering process. Instead of the traditional approach of altering node cluster memberships within a fixed graph topology to maximize modularity, we aim to optimize the topology based on a given clustering result by maximizing modularity. Once we obtain an improved graph topology through this optimization process, running clustering algorithms on this enhanced topology enables us to achieve more accurate clustering results.

3.2 Algorithm Details

Let us delve deeper into the definition of modularity as given in (1) in Section 2.1.

When edges between two clusters are removed, it directly affects the modularity calculation by altering both the adjacency matrix A_{ij} and the total number of edges m. Generally, removing inter-cluster edges tends to increase modularity since modularity rewards network structures with dense intra-cluster connections and sparse inter-cluster connections.

More specifically, removing edges between different communities reduces the $\frac{k_i k_j}{2m}$ term in the equation because m (the total number of edges) decreases, and since $\delta(c_i, c_j) = 0$ for nodes in different communities, no subtraction of actual existing edge contributions occurs. Consequently, the removal of edges between communities usually leads to an increase in modularity, reflecting a more distinct community structure. However, there is a limit to this increase. If too many edges are removed, the network may become too fragmented, and communities may become disconnected, at which point modularity may no longer be a good measure. Therefore, removing edges to increase modularity should be done without compromising the connectivity within communities.

Based on the above analysis, we propose utilizing modularity to direct the optimization of graph topology and the enhancement of clustering accuracy.

The initial challenge in our method is that at the beginning, we neither have a sufficiently optimized topology nor highly accurate clustering results. If we proceed to remove edges between different clusters based on inaccurate cluster memberships, this will not only fail to optimize the topology but may also result in the removal of numerous correct edges, thereby worsening the topology. To address this, for each cluster, we calculate the centroid and identify the cluster that is furthest away by comparing the distances to the centroids of all other clusters, and then we remove all edges between it and the cluster that is furthest away. Although the initial accuracy of the clustering results may not be high, the points within a cluster and those in the cluster that is furthest away are still very likely to not belong to the same class. Therefore, removing the edges between them is a safe and confident approach.

Subsequently, after removing the edges, we rerun the clustering algorithm on this optimized topology. The clustering algorithm should yield more accurate results on the optimized topology. Then, based on this more accurate clustering result, we search for the furthest cluster from each cluster and remove the edges between them.

Based on the aforementioned process, the quality of the graph topology and the accuracy of graph-based clustering enhance each other, improving iteratively.

3.3 Termination Criteria

In our method, the quality of the graph and the accuracy of the clustering algorithm are alternately and synergistically improved. Initially, for a given graph, we optimize the graph topology by removing edges based on the clustering results. Then, we use the optimized topology to further improve the accuracy of the graph-based clustering method. This process is iterative. At the end of each iteration, we calculate and record the modularity of the graph. We then compare this modularity with that at the end of the previous iteration. If we observe that the modularity no longer increases or if it reaches the preset maximum number of iterations, we stop the process.

The complete algorithm flow has been shown in Algorithm 1.

Algorithm 1 Modularity guided graph topology optimization

Input: A given graph G, number of divisions p. **Output:** The optimized graph.

```
1: while the modularity of the graph does not increase compared to the previous
   iteration or the maximum number of iterations is reached do
2:     Perform clustering algorithm to divide the latest graph into $p$ clusters;
3:     for each cluster $C_i$ do
4:         Find the farthest cluster $C_j$ from it based on the distance between cluster
           centroids;
5:         Remove all the edges between $C_i$ and $C_j$;
6:     end for
7:     Check the termination criterion.
8: end while
```

3.4 Complexity Analysis

In Algorithm 1, the time complexity is dominated by the clustering algorithm employed. It should be noted and emphasized that our method is not designed for any specific clustering algorithm, but rather it is a generalized optimization approach applicable to all graph-based clustering algorithms. Currently, the latest research progress have reduced the time complexity of representative graph-based clustering algorithms, such as spectral clustering, to near-linear [14]. Therefore, if we use the spectral clustering algorithm for clustering in Algorithm 1, the complexity of Algorithm 1 is near-linear.

4 Experiment

In this paper, we construct the initial graph topology using the most commonly employed k-Nearest Neighbors (k-NN) method and perform clustering with the classic spectral clustering algorithm. The value of k is set to 10 in the k-NN method. Experiments are performed using MATLAB running on the Laptop.

4.1 Data Sets

COIL-20 includes 1,440 grayscale images of 20 different objects. Each object is placed on a motorized turntable against a black background, and the turntable is rotated through 360 degrees to capture images at 5-degree intervals, resulting in 72 images per object. These images are down-sampled to a resolution of 32x32 pixels, leading to each image being represented by 1,024 attributes. This dataset is commonly used for object recognition and classification tasks due to its controlled environment and diverse object representations.

PenDigits includes 7,494 handwritten digit samples collected from 44 writers. Each writer contributed multiple instances of the digits 0 through 9, resulting in a diverse and extensive collection of handwritten digits. Each digit is represented by 16 attributes that capture the pen-tip coordinates as the digit is drawn. This dataset is widely used for evaluating handwriting recognition algorithms and has a good balance of inter-class diversity and intra-class variability.

USPS includes 9,298 images of handwritten digits from the USPS postal service. Each digit image is resized to a 16x16 pixel grayscale image, resulting in 256 attributes per image. This dataset is well-known for its use in machine learning and pattern recognition research, particularly in the context of digit classification. The USPS dataset provides a challenging testbed for algorithms due to variations in handwriting styles and digit shapes.

4.2 Comparison Methods

We compare the proposed method against both the baseline and the state-of-the-art graph learning (construction) methods by applying the same spectral clustering algorithm to the graphs constructed by these various methods and evaluating the accuracy of the clustering results. The methods being compared include:

1) k-NN graph: the most widely used graph construction method. Each node is connected to its k nearest neighbors.

2) Consensus k-NN graph [1]: the state-of-the-art graph edge selection methods for improving the performance of k-NN graph. It improves the robustness of the k-NN graph by using the consensus information from different neighborhoods of a given k-NN graph.

3) LGSS [3]: the most recent progress in graph learning field from the GSP perspective. It can automatically select the parameters of the model for achieving the desired graph properties.

4) Spectral edge sparsification method [15]: The state-of-the-art method for detecting non-critical, misleading, and superfluous edges in the graph via spectral analysis.

4.3 Evaluation Metric

We measure the quality of clustering with two metrics: clustering accuracy (ACC) and normalized mutual information (NMI) between the clustering results generated by clustering algorithms and the ground-truth labels provided by the data sets. The two metrics are defined as follows:

Clustering Accuracy The clustering accuracy is defined as:

$$ACC = \frac{\sum\limits_{j=1}^{n} \delta(y_i, map(c_i))}{n}, \tag{2}$$

where n is the number of data instances in the data set, y_i is the ground-truth label provided by the data set,and C_i is the label generated by the clustering algorithm. $\delta(x, y)$ is a delta function defined as: $\delta(x, y)$=1 for $x = y$, and $\delta(x, y)$=0, otherwise. $map(\bullet)$ is a permutation function that maps each cluster index c_i to a ground truth label, which can be realized using the Hungarian algorithm [10]. A higher value of ACC indicates better clustering quality.

Normalized Mutual Information For two random variables P and Q, normalized mutual information is defined as [13]:

$$NMI = \frac{I(P, Q)}{\sqrt{H(P)H(Q)}}, \tag{3}$$

where $I(P, Q)$ denotes the mutual information between P and Q, while $H(P)$ and $H(Q)$ are entropies of P and Q. In practice, the NMI metric can be calculated as follows [13]:

$$NMI = \frac{\sum_{i=1}^{k}\sum_{j=1}^{k} n_{i,j}\log(\frac{n \cdot n_{i,j}}{n_i \cdot n_j})}{\sqrt{(\sum_{i=1}^{k} n_i \log\frac{n_i}{n})(\sum_{j=1}^{k} n_j \log\frac{n_j}{n})}}, \tag{4}$$

where n is the number of data points in the data set, k is the number of clusters, n_i is the number of data points in cluster C_i according to the clustering result generated by algorithm, n_j is the number of data points in class C_j according to the ground truth labels provided by the data set, and $n_{i,j}$ is the number of data points in cluster C_i according to the clustering result as well as in class C_j according to the ground truth labels. The NMI value is in the range of [0, 1], while a higher NMI value indicates a better matching between the algorithm generated result and ground truth result.

4.4 Clustering Quality Results

We perform spectral clustering algorithm on the graphs generated by the four graph construction methods. The clustering accuracy results and NMI results are shown in Table 1 and Table 2, respectively.

As shown in Table 1, the proposed method can consistently lead to dramatic performance improvement over the given graph. By applying our optimization method on k-NN graph, it achieves more than 6%, 12% and 16% clustering accuracy gains on the three data sets, respectively. For the Pendigits dataset, our method has 12% clustering accuracy gain over the second-best method. The superior clustering results clearly demonstrate the effectiveness of the proposed method. As shown in Table 2, for both the Pendigits and the USPS data sets, our method provides best NMI results among all the compared methods. Based on the experimental results, our method outperforms the spectral sparsification approach on all three datasets. Both the spectral sparsification method and our method involve constructing a sub-graph of the original graph. However, the spectral sparsification method achieves this by performing spectral domain analysis to identify and remove edges that have little impact on the key spectral properties of the original graph. In contrast, our method removes edges that, when eliminated, can increase the modularity of the graph. Compared to spectral sparsification, which merely eliminates redundant connections, our approach directly promotes the clarification of the graph's clustering structure. Therefore, our algorithm can lead to a greater improvement in clustering accuracy.

Table 1. Clustering Accuracy (%)

Data Set	k-NN	Consensus	LGSS	Spectral Spar	Our method
COIL20	75.72	81.60	85.49	76.27	82.22
PenDigits	74.36	71.08	74.53	83.26	86.42
USPS	64.31	68.54	81.50	70.74	80.55

Table 2. NMI

Data Set	k-NN	Consensus	LGSS	Spectral Spar	Our method
COIL20	0.86	0.90	0.95	0.86	0.88
PenDigits	0.79	0.79	0.77	0.78	0.81
USPS	0.79	0.81	0.84	0.81	0.85

4.5 Graph Construction Time Results

To assess the efficiency of graph construction, we report the time required for building graphs with both our method and current state-of-the-art methods, as shown in Table 3.

Table 3. Graph learning (construction) time (Seconds)

Data Set	Consensus	LGSS	Our method
COIL20	2.43	13.56	6.32
PenDigits	172.51	1085.43	7.87
USPS	574.28	2074.78	5.01

It can be seen that our method is much more efficient compared to other methods. For the Pendigits data set, our method achieved **22X** and **138X** times speedup over the Consensus method and the LGSS method, respectively. For the USPS data set, our method achieved **115X** and **415X** times speedup over the Consensus method and the LGSS method, respectively. These results indicate that, compared with state-of-the-art graph construction methods, our graph optimization approach not only yields higher-quality graphs that significantly enhance clustering accuracy but also greatly improves efficiency.

4.6 Quantitative Empirical Validation of the Relationship Between Modularity and Clustering Quality

Although many algorithms in the field of community detection optimize clustering based on modularity, to the best of our knowledge, there have been no empirical quantitative results regarding the relationship between modularity and the quality of clustering to date. In this paper, we design experiments based on our proposed method and provide quantitative evidence confirming the relationship between modularity and clustering effectiveness.

The clustering accuracy and modularity results through iterations of our method have been shown in Figure 2 and Figure 3, respectively. It can be observed that the clustering accuracy and modularity increase concurrently, which substantiates the validity of the modularity theory.

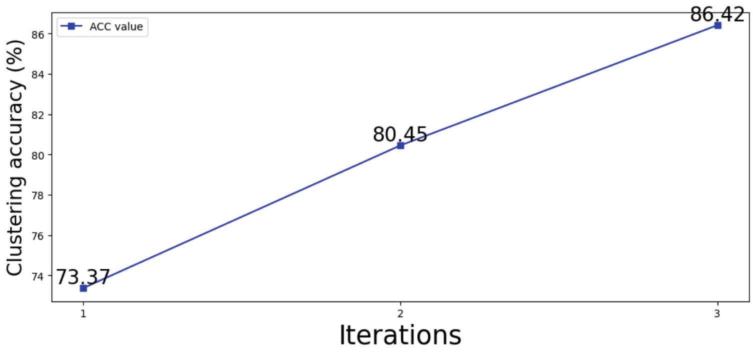


Fig. 2. Clustering accuracy through iterations of the Pendigits data set.

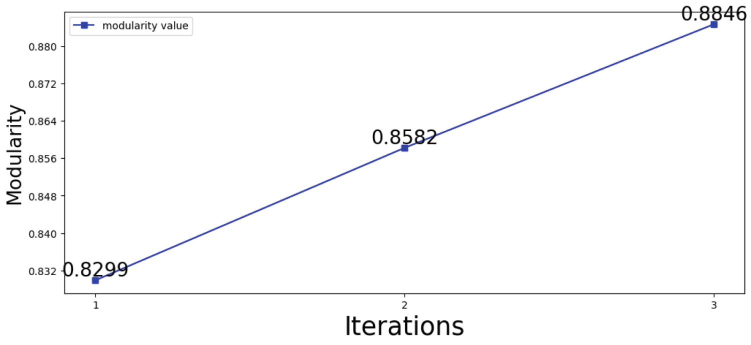


Fig. 3. Modularity through iterations of the Pendigits data set.

5 Conclusion

In this work, we present a modularity-guided graph topology optimization method. We show that the graph topology learning problem can be solved by iteratively identifying and removing redundant and misleading edges to increase the modularity of the graph. Unlike traditional modularity-based approaches which focus on updating cluster-membership of samples to maximize the modularity, our method aims to optimize the graph topology through the modularity maximization process. When comparing with state-of-the-art graph construction (learning) approaches, our approach is more efficient and leads to substantially improved solution quality of graph-based clustering. The paper also validates the modularity theory through quantitative empirical evidence of the relationship between modularity and clustering quality.

References

1. V. Premachandran and R. Kakarala, “Consensus of k-nns for robust neighborhood selection on graph-based manifolds,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1594–1601
2. Egilmez, H.E., Pavez, E., Ortega, A.: Graph learning from data under laplacian and structural constraints. IEEE Journal of Selected Topics in Signal Processing **11**(6), 825–841 (2017)
3. V. Kalofolias and N. Perraudin, “Large scale graph learning from smooth signals,” in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=ryGkSo0qYm
4. Newman, M.E.: Modularity and community structure in networks. Proc. Natl. Acad. Sci. **103**(23), 8577–8582 (2006)
5. Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech: Theory Exp. **2008**(10), P10008 (2008)
6. Traag, V.A., Waltman, L., Van Eck, N.J.: From louvain to leiden: guaranteeing well-connected communities. Sci. Rep. **9**(1), 5233 (2019)
7. Yang, J., Leskovec, J.: Structure and overlaps of ground-truth communities in networks. ACM Transactions on Intelligent Systems and Technology (TIST) **5**(2), 1–35 (2014)
8. Harenberg, S., Bello, G., Gjeltema, L., Ranshous, S., Harlalka, J., Seay, R., Padmanabhan, K., Samatova, N.: Community detection in large-scale networks: a survey and empirical evaluation. Wiley Interdisciplinary Reviews: Computational Statistics **6**(6), 426–439 (2014)
9. Chen, W.-Y., Song, Y., Bai, H., Lin, C.-J., Chang, E.Y.: Parallel spectral clustering in distributed systems. IEEE Trans. Pattern Anal. Mach. Intell. **33**(3), 568–586 (2011)
10. C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998
11. V. D. Silva and J. B. Tenenbaum, “Global versus local methods in nonlinear dimensionality reduction,” in *Advances in neural information processing systems*, 2003, pp. 721–728
12. V. Kalofolias, “How to learn a graph from smooth signals,” in *Artificial Intelligence and Statistics*, 2016, pp. 920–929
13. A. Strehl and J. Ghosh, “Cluster ensembles—a knowledge reuse framework for combining multiple partitions,” *Journal of machine learning research*, vol. 3, no. Dec, pp. 583–617, 2002
14. Y. Wang, “Improving spectral clustering using spectrum-preserving node aggregation,” in *2022 26th International Conference on Pattern Recognition (ICPR)*. IEEE, 2022, pp. 3063–3068
15. Y. Wang and Z. Feng, “Towards scalable spectral clustering via spectrum-preserving sparsification,” in *33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022*. BMVA Press, 2022

Leveraging Data from Vast Unexplored Seas: Positive Unlabeled Learning for Refining Prediction Area in Good Fishing Ground Prediction

Haruki Konii[1](✉), Teppei Nakano[1,2], Yasumasa Miyazawa[3], and Tetsuji Ogawa[1]

[1] Department of Communications and Computer Engineering, Waseda University, Tokyo, Japan
konii@pcl.cs.waseda.ac.jp
[2] Intelligent Framework Laboratory, Tokyo, Japan
[3] Application Laboratory, Japan Agency for Marine-Earth Science and Technology, Yokohama, Japan

Abstract. This paper proposes a Positive Unlabeled (PU) learning approach to narrow down the prediction area in the context of good fishing ground prediction. PU learning, a type of semi-supervised learning, is particularly suitable for scenarios characterized by limited positive examples and abundant unlabeled data, as often encountered in fishing ground prediction tasks, where the explored sea areas identified as productive fishing spots are treated as positive instances and the vast unexplored sea areas as unlabeled data. Conventional methods often struggle to accurately model the characteristics of good fishing grounds, resulting in overly broad prediction areas or overly restrictive constraints. To tackle this challenge, we present a PU learning-based method designed to identify negative examples from the unlabeled data and consequently refine the area predicted as positive. Specifically, we train a prediction model for fishing duration, which can be considered a surrogate indicator of good fishing grounds. Subsequently, we apply this model to predict the fishing duration for unexplored areas; those areas exhibiting short fishing durations are deemed reliable negative examples. By incorporating both past positive examples and selected negative examples into a binary classification framework for predicting good fishing grounds, we aim to fine-tune the prediction area. To the best of our knowledge, this study represents the first application of PU learning in the domain of good fishing ground prediction. Experimental comparisons conducted using data from bullet tuna trolling validate the efficacy of the proposed methodology.

Keywords: Positive unlabeled learning · Good fishing ground prediction · Bullet tuna trolling

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 143–158, 2025.
https://doi.org/10.1007/978-3-031-78192-6_10

1 Introduction

Predicting optimal fishing locations is essential for assisting fishers in decision-making, thereby enhancing operational efficiency. This is expected to result in increased catch yields while minimizing fuel costs and time spent on exploration and movement. To achieve precise predictions of favorable fishing spots, many studies have employed machine learning methodologies [1,2,5–7,10,11,17–19,21]. Many of these approaches have focused on forecasting favorable fishing locations by leveraging local catch data and oceanographic information within specific regions. However, obtaining local fishing yields typically requires equipping fishing vessels with expensive sensors, which presents a challenge for many types of fishing activities. Moreover, the observable data typically stems solely from the GPS log positions of productive fishing grounds, which are considered positive instances in the prediction of good fishing grounds. In areas where the vessel is not in motion, no information is available about whether a particular location constitutes a good fishing ground. These challenges pertain to constructing prediction models using a limited number of positive instances and a majority of unlabeled data, a scenario known as positive unlabeled (PU) learning problems. Despite these challenges, there are currently no instances of PU learning being applied to the prediction of good fishing grounds.

PU learning, a specialized case of semi-supervised learning, involves learning from a limited number of positive examples and a large amount of unlabeled data [3]. Good fishing ground prediction is framed as a PU learning problem because areas of the sea lacking fishing records cannot be designated as negative examples but rather remain unlabeled. Such tasks, comprising positive examples and unlabeled data, are prevalent in real-world scenarios such as medical diagnosis, anomaly detection, and landslide prediction, attracting considerable attention in recent years [4,16,20,22–24]. One method for leveraging unlabeled data in PU learning is through two-step techniques [3,13]. This involves first identifying reliable negative examples from the unlabeled data and then using labeled positive examples alongside these reliable negative examples for learning. In this paper, we propose applying two-step techniques to the task of predicting good fishing grounds.

Furthermore, existing methods for predicting favorable fishing locations have not explicitly considered the quality of fishing grounds, such as catch volume. These approaches have employed convolutional autoencoders and determined the suitability of a sea area as a good fishing ground based on the magnitude of reconstruction errors [6]. If the error is small, the sea area is classified as a good fishing ground; otherwise, it is deemed unsuitable. However, due to the scarcity of negative examples during training and the omission of factors like catch volume, these methods tended to overpredict a large area as positive (i.e., over-detection), which did not sufficiently assist fishers in decision-making. To address this issue, the attribute-dependent thresholding (ADT) model was introduced, incorporating prior knowledge such as location and depth [10]. Since bullet tuna fishing tends to occur in similar geographical areas and depths, this model succeeded in narrowing down the prediction area. However, it overly constrained the area by relying too heavily on prior knowledge, leading to predictions limited to areas with a high frequency of past good fishing grounds and missing the opportunity to identify a broader range of unknown spots where fish might be found.

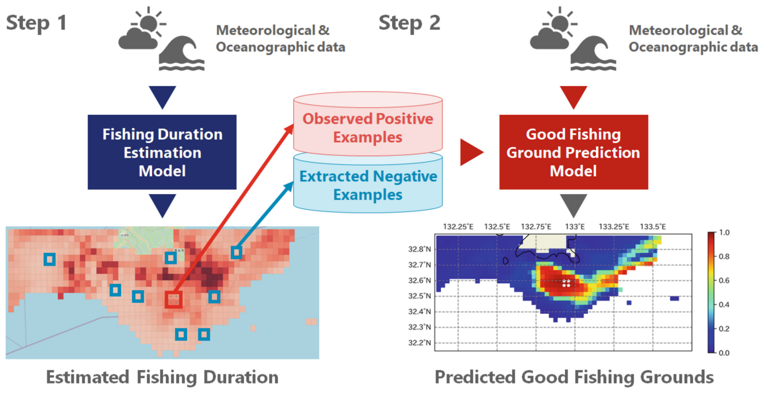

Step 1
Meteorological & Oceanographic data
Fishing Duration Estimation Model
Estimated Fishing Duration
Observed Positive Examples
Extracted Negative Examples
Step 2
Meteorological & Oceanographic data
Good Fishing Ground Prediction Model
Predicted Good Fishing Grounds

Fig. 1. Overview of proposed two-step method for PU learning-based development of good fishing ground predictor. Step 1 involves creating model to predict fishing duration and using it to extract negative examples from unlabeled data. Step 2 constructs binary classifier to differentiate each sea area into good and poor fishing locations using observed positive and estimated negative examples.

This study aims to leverage a significant volume of unlabeled data to narrow down the prediction area without relying on prior knowledge such as geographical locations and depths. To accomplish this, we propose a PU learning-based approach for predicting good fishing grounds, as depicted in Fig. 1. The proposed method consists of two primary steps: *i)* developing a model to predict fishing duration, specifically the time spent fishing in a particular sea area, and extracting negative examples from unlabeled data based on the predicted duration (where a longer duration indicates a more favorable location), and *ii)* constructing a binary classifier to distinguish each sea area into good and poor fishing spots. By utilizing historical positive examples and the estimated negative examples, our aim is to enhance the discriminative capability in identifying good fishing grounds, thereby refining the prediction area. To validate the effectiveness of our proposed method, we use operational data from bullet tuna trolling activities conducted between 2014 and 2017. Although the proxy prediction task in step 1 may seem very specific and limited, our methodology of constructing a predictor based on features related to the final task and subsequently extracting examples using that predictor can be broadly applicable to many PU learning real-world applications. For instance, in healthcare predictive analytics, our methodology could be applied to predict patient outcomes based on historical health records. The proxy prediction task could be adapted to forecast intermediate health indicators, which subsequently inform long-term health outcomes. Therefore, the insights gained from this experiment are anticipated to not only contribute to PU learning for predicting good fishing grounds but also address various PU problems encountered in real-world scenarios.

The remainder of this paper is structured as follows. Section 2 discusses the data employed, encompassing meteorological and oceanographic data, catch data, and operational information. Section 3 explains the details of the proposed two-step method for PU learning in good fishing ground prediction. Section 4 demonstrates the effectiveness of the proposed method. Finally, Sect. 5 provides a summary of the paper.

2 Data

In this study, we utilize data related to the bullet tuna trolling activities carried out off the coast of Tosa-Shimizu City, Kochi Prefecture, Japan, from October 2014 to March 2017. This data comprises fishing vessel trajectories obtained from GPS, statistical information regarding favorable fishing locations derived from these trajectories, and meteorological and oceanographic data during fishing operations. In this case, the monitored sea area is divided into a grid with dimensions of approximately 2.6 km north-south and 2.4 km east-west, and predictions for productive fishing grounds are made for each grid cell. The grid size is determined by the resolution of the oceanographic simulations employed to gather oceanographic data.

2.1 Fishing Vessel Trajectories and Past Good Fishing Grounds

The GPS trajectories of fishing vessels are utilized to identify productive fishing grounds, gather statistical data on previously productive fishing grids, and estimate the duration of fishing activities within each grid. In the absence of local catch data, we consider fishing duration as a proxy for catch volume.

The actual fishing locations were determined from the GPS trajectory data of the fishing vessels. An example of this data is illustrated in Fig. 2a, where the red line depicts the trajectory. The GPS trajectory data comprises time, latitude, longitude, and heading (the direction in which the bow is facing), recorded every second. During bullet tuna trolling, vessels consistently execute left turns while fishing, facilitating the identification of fishing locations based on changes in heading.

As depicted in Fig. 2b, positive labels were assigned to roughly 80% of the consecutive points where fishing occurred, spanning from the beginning to the end of the operation. This assumption is based on the notion that approximately 80% of these points represent stable fishing spots where fish are reliably caught, as indicated by feedback from fishermen.

Figure 3 illustrates the distribution of previously observed productive fishing grounds. From this figure, it is apparent that areas with a high frequency of productive fishing grounds are clustered near the fishing port at coordinates (32°8’N, 133°E) and along the coast between (32°5’N–32°6’N, 132°75’E–133°E). A comparison between the predicted good fishing ground locations and the actual distribution of productive fishing grounds is conducted in Sect. 4.3.

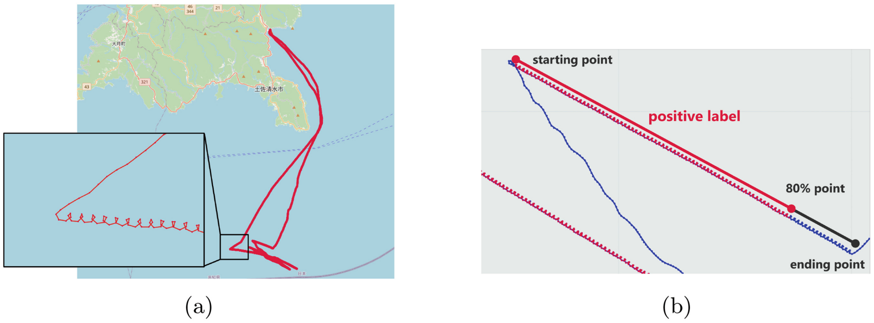


Fig. 2. (a) Illustration of fishing vessel's trajectory obtained from GPS sensors on November 4, 2014. Continuous left turns are executed during fishing operations. (b) Example of positive label annotation. Positive labels were attributed to approximately 80% of points spanning from starting point to ending point of operation.

2.2 Oceanographic and Meteorological Information

The experimental features encompassed oceanographic parameters such as water temperature, salinity, north-south current velocity, and east-west current velocity, along with meteorological data. The oceanographic parameters were sourced from the JCOPE-T reanalysis dataset [8], characterized by a horizontal grid resolution of 1/36-degree. This dataset is compiled by assimilating data from satellite altimetry, sea surface temperature, in-situ temperature, and salinity measurements into an ocean circulation model. The JCOPE ocean forecasting system, which generates this dataset, regularly updates its forecasts by incorporating as much observation data and satellite altimetry data as feasible [14,15].

The data utilized for predicting good fishing grounds includes:

– Spatial range: 32–33°N, 132–134°E
– Data-extracted Depth: 0, 5, 10, 20, 30, 40, 50 m
– Period: 2014/10/01–2017/3/31
– Variables: north-south current velocity, east-west current velocity, potential water temperature, salinity

The oceanographic data comprises an 11×11 grid surrounding the target prediction grid, encompassing seven depths ranging from 0 m to 50 m, and three hours of data up to two hours before the observation and prediction time, resulting in a total of 2541 dimensions. The size of each area is approximately 2.6km $\times$ 2.4km, which corresponds to the dimensions of each grid depicted in Fig. 7.

Meteorological data were sourced from the Shimizu Observatory, located nearest to the fishing area in Tosashimizu, Kochi Prefecture. Table 1 enumerates the 18 data types utilized in the experiments, which are accessible via the website of the Japan Meteorological Agency [9]. Wind direction data, provided in

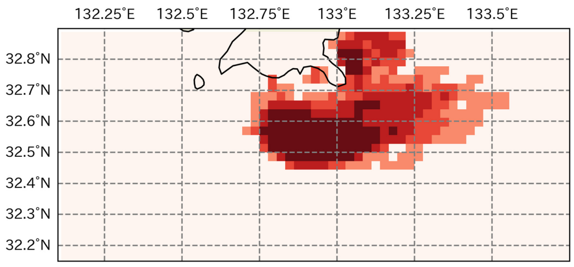


Fig. 3. Distribution of past good fishing grounds. Each grid is color-coded according to percentiles of values, indicating 0, 25, 50, 75, and 100 percentiles. Darker red indicates higher frequency, while lightest red indicates no observed good fishing grounds. (Color figure online)

Table 1. Meteorological information used.

Attribute	Details	Dim
Temperature [°C]	Ave., Max., Min.	3
Humidity [%]	Ave., Min.	2
Precipitation [mm]	Total, Max. in 1 h, Max. in 10 min.	3
Mean atmospheric pressure [hPa]	Local, Sea level	2
Wind speed [m/s]	Ave., Max., Instantaneous max.	3
Wind direction [cos, sin]	Max., Instantaneous max.	4
Sunshine hours [h]	Sunshine hours	1
Total		18

azimuth, were pre-processed into cosine and sine components. The meteorological data input into the model encompassed two days of observations from the day prior and two days preceding the prediction date, resulting in a 36-dimensional feature.

3 PU Learning for Good Fishing Ground Prediction

This section provides a detailed procedure for the proposed PU learning methodology. Areas with short fishing durations can be regarded as negative instances, indicating areas that are not conducive to good fishing grounds. Thus, fishing duration data is estimated from GPS trajectories (Sect. 3.1) and utilized to train a model for estimating fishing duration (Sect. 3.2). Subsequently, this model is employed to estimate fishing durations in unexplored sea areas, with areas exhibiting short durations identified as reliable negative instances. These estimated reliable negative instances, along with observed positive instances, are utilized to construct a model for predicting good fishing grounds via binary classification, discerning whether an area qualifies as a productive fishing ground (Sect. 3.3).

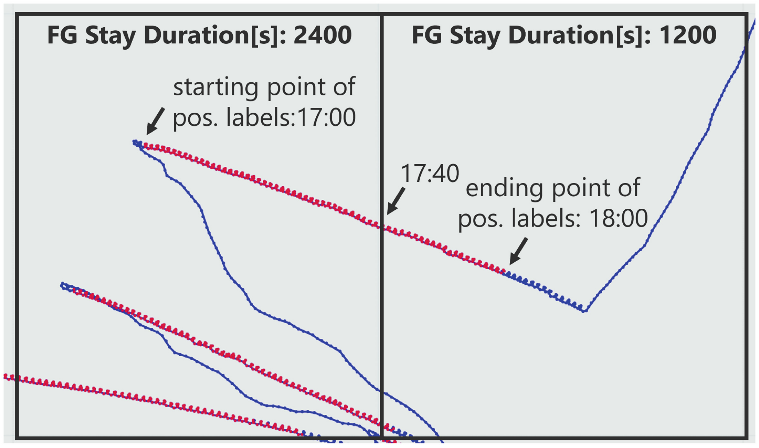

FG Stay Duration[s]: 2400
FG Stay Duration[s]: 1200
starting point of pos. labels:17:00
17:40
ending point of pos. labels: 18:00

Fig. 4. Example of fishing duration acquisition: Duration, measured in seconds, of positive instances is obtained for each grid on hourly basis.

3.1 Estimation of Fishing Duration

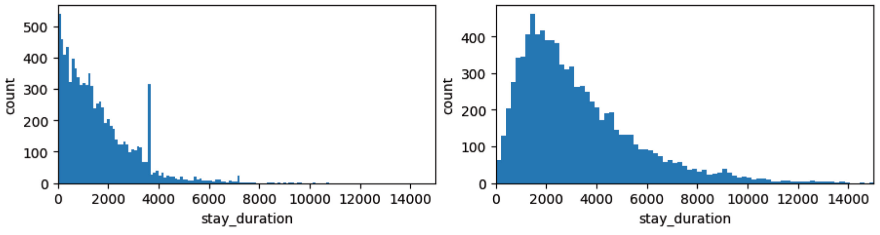
As depicted in Fig. 4, we aggregate the positive examples for each designated grid and calculate the duration of positive examples (referred to as fishing duration) in seconds for each grid on an hourly basis.

Figure 5a illustrates the obtained distribution of fishing duration, which is subject to distortions due to limitations in data acquisition. This distribution exhibits a skewed shape, with a peak in the count near 0 s that decreases as the duration increases, spiking again at 3600 s. The high frequency near 0 s is attributed to the timing or grid boundaries, as fishing activity is aggregated hourly for each grid. The spike at 3600 s corresponds to the maximum duration a vessel can continue fishing within a single grid, calculated as 60 min $\times$ 60 s $=$ 3600 s. Any duration exceeding 3600 s indicates multiple vessels engaging in fishing activities simultaneously. Consequently, the fishing duration obtained through this process cannot directly serve as an indicator of good fishing grounds.

To address the constraints in data acquisition and allow the duration to serve as an indicator of good fishing grounds, temporal and spatial smoothing is employed. During the smoothing process, if fishing activity is detected in the grid or its vicinity one hour prior or if there is concurrent fishing activity in the vicinity, the fishing duration is extended. This adjustment facilitates duration elongation in areas with continuous or neighboring fishing activities.

Let (x, y) denote a grid and t represent the time. We define $R(x, y, t)$ as the actual fishing duration without smoothing and $S(x, y, t)$ as the duration with smoothing. In this case, the detailed smoothing process is formulated as

$$S(x,y,t) = R(x,y,t) + \alpha S(x,y,t-1) + \beta \sum_{\substack{i=x-1 \\ i\neq x}}^{x+1} \sum_{\substack{j=y-1 \\ j\neq y}}^{y+1} R(i,j,t), \tag{1}$$



(a) Without smoothing (b) With smoothing

Fig. 5. Distribution of fishing duration: Horizontal axis represents fishing duration, while vertical axis represents count. Due to constraints in data acquisition, Fig. 5a exhibits spikes near 0 s and prominent spike at 3600 s (one hour). In contrast, Fig. 5b, with smoothing applied, displays more natural shape without prominent spikes.

where at time t, α signifies the coefficient for the smoothed fishing duration of grid (x, y) at time $t-1$, and β represents the coefficient for the actual fishing duration of neighboring grids at time t. This equation facilitates temporal and spatial smoothing, incorporating the smoothed fishing duration from one hour before and the actual fishing duration of neighboring grids at time t into the actual fishing duration of grid (x, y) at time t.

Figure 5b illustrates the distribution of fishing durations after applying smoothing with $\alpha = 0.5$ and $\beta = 0.25$. In comparison to Fig. 5a, there is no longer a spike around 3600 s, and the graph exhibits a more natural shape with a peak around 1800 s. To determine the values of α and β, fishing duration data from the training dataset was utilized. Nine different conditions, varying the values of α and β, were tested. The combination that appeared smoothest was selected based on visual assessment. This smoothing procedure has rendered it viable to regard the duration of stay in fishing grounds as an indicator of good fishing locations. Hence, $S(x, y, t)$ is employed as the fishing duration at grid (x, y) and time t.

3.2 Extraction of Negative Instances from Unlabeled Data

The accurate detection of negative examples from unlabeled data is crucial for the success of PU learning based on the two-step approach.

Fishing Duration Modeling. Firstly, our objective is to develop a model that estimates fishing duration for each grid using corresponding meteorological and oceanographic data. As discussed in Sect. 3.1, fishing duration can act as a proxy indicator for identifying good fishing grounds. Therefore, grids with longer fishing durations in historical positive data are likely to represent good fishing grounds, while those with shorter durations are less likely to be productive. Consequently, when the model predicts a small value (indicating a short fishing duration), the corresponding grid is classified as a negative example (representing a poor fishing ground).

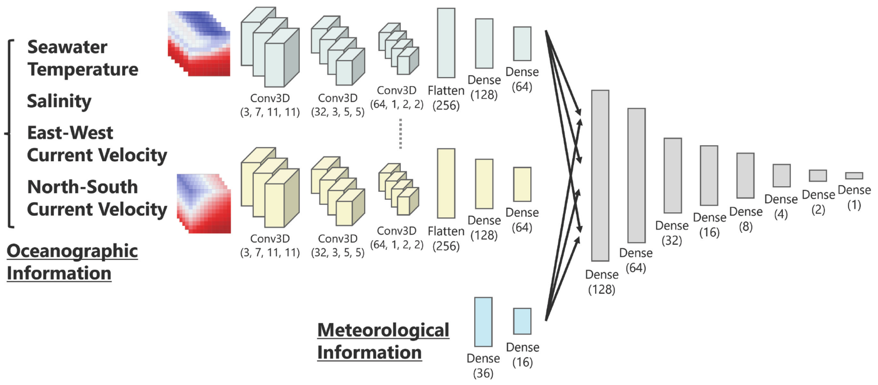


Fig. 6. Network architecture of fishing duration estimation model and good fishing ground prediction model. Final output of fishing duration estimation model is non-negative numerical value, while for prediction of good fishing grounds, output falls between 0 and 1.

The architecture of the fishing duration estimation model is depicted in Fig. 6. The model comprises a combination of three-dimensional convolutional layers and one-dimensional linear layers. Oceanographic parameters such as water temperature, salinity, and north-south/east-west current velocity are processed by the three-dimensional convolution layers, while the linear layers handle meteorological data. By representing oceanographic information in a three-dimensional convolution with three channels, the model is designed to capture the spatio-temporal characteristics associated with good fishing grounds. The final output of the model is the fishing duration, which is a non-negative numerical value. Note that no activation function is applied for the final layer in this model.

Extraction of Negative Examples from Unlabeled Data. The developed fishing duration estimation model is applied to identify reliable negative examples from unlabeled data. Initially, the meteorological and oceanographic data from unlabeled grids, representing unobserved sea areas, serve as inputs to the fishing duration estimation model. Subsequently, a weighted random sampling is conducted based on the output of the model, favoring grids with lower values to be more likely extracted as reliable negative examples. For each date and time with at least one positive example, an equivalent number of negative examples are sampled. For example, if there are four positive examples at 11:00 on November 9, 2014, four negative examples are drawn from the unlabeled grids.

3.3 Good Fishing Ground Prediction as Binary Classification

We train the model for predicting good fishing grounds using both previously observed positive examples and estimated negative examples within the framework of PU learning using two-step techniques. In this stage, all past positive examples are treated as positive examples, regardless of their fishing durations. This approach is guided by two primary considerations. Firstly, previously observed positive examples are inherently more reliable than estimated negative examples. Secondly, given the limited availability of positive examples, it is imperative to gather more positive instances to improve performance. By incorporating reliable negative examples derived from unlabeled data into the training process for binary classification, we anticipate an enhancement in discriminative performance and a more refined prediction area (i.e., further narrowing down the prediction area).

Figure 6 depicts the architecture of the good fishing ground prediction model. This architecture closely resembles that of the fishing duration estimation model, except for the final output. The primary difference between the two models lies in the final layer: the fishing duration estimation model performs regression, and this model performs classification. The activation function for the final layer in this model is a Sigmoid function, so the final output ranges between 0 and 1. Positive examples are labeled as 1, while extracted negative examples are labeled as 0.

4 Good Fishing Ground Prediction Experiment

To assess the efficacy of the proposed approach in narrowing down the prediction area of productive fishing grounds using unlabeled data, experiments were conducted using operational data from bullet tuna trolling from 2014 to 2017. Given the preference of bullet tuna for shallow waters, the target prediction area encompasses waters with a depth of 800 m or less within the region of 32–33°N and 132–134°E.

4.1 Experimental Setup

In the experiments, the data from October 2014 to March 2017 was partitioned into five data folds, and five-fold cross-validation was conducted to account for the annual and seasonal variations in the patterns of good fishing grounds. Table 2 presents the fold ID, corresponding period, along with the number of positive examples and extracted negative examples for each fold. In each iteration, the four folds were used for training, and the remaining fold was used for testing, ensuring complete separation of training and testing folds to prevent data overlap and leakage. The four folds were used for training and negative sample extraction in Step 1, and for training in Step 2. The remaining fold was used exclusively for testing in Step 2.

We conducted a comparison between modeling techniques employing the proposed PU learning and conventional inlier modeling, which utilizes only positive examples. For inlier modeling, we explored both attribute independent training (AIT) and attribute dependent training (ADT) [6,10], the latter being suitable for refining prediction results. These two models (AIT and ADT) were implemented using PyTorch and trained for 250 epochs with a learning rate of 0.0001.

Table 2. Details of each fold: Positive examples consist of observed data, whereas negative examples are extracted from unlabeled data.

Fold ID	Period	# Positive examples	# Negative examples
A01	2014/10–2015/03	1151	1151
A02	2015/04–2015/05	1178	1178
A03	2015/06–2015/12	1097	1097
A04	2016/02–2016/09	1484	1484
A05	2016/10–2017/03	1594	1594

These hyperparameters were selected based on preliminary experiments using the A1 fold. The proposed model underwent training using five different epochs, specifically 15, 20, 25, 30, and 35.

4.2 Evaluation Metrics

The evaluation metric for predicting fishing duration is the mean absolute error (MAE). For the binary classification of good fishing ground prediction, we utilize the Area Under the Curve (AUC), where the ratio of grids estimated to be positive (RGEP) is plotted on the x-axis and the recall is plotted on the y-axis, with varying thresholds. Here, the RGEP and recall are defined as follows:

- **Ratio of grids estimated to be positive (RGEP)** $Pr[f(X) = 1]$: This metric indicates how well the predicted area of good fishing ground is narrowed down (lower values are desirable).
- **Recall** $r = Pr[f(X) = 1|Y = 1]$: This metric indicates how well actual good fishing grounds are detected (higher values are desirable).

Here, $f(X) = 1$ denotes that the input area X is predicted as a good fishing ground, and $Y = 1$ indicates that the actual area is indeed a good fishing ground. A metric $r^2/RGEP$ calculated from the above metrics is often utilized as an alternative to the F score in PU problems where negative examples are unavailable [12]. However, unlike in [12], we utilize the AUC described below instead of the F-score because the optimal threshold for determining good fishing grounds varies annually and seasonally, making AUC more appropriate than the F-score for fundamental technology development.

The RGEP serves as a substitute for the precision when comprehensive observation of good fishing ground data is impractical. Typically, classification problems employ the precision and recall as evaluation metrics. However, the precision cannot be computed in this task because there is no means to verify whether the predicted grid is genuinely positive or not. In its place, the RGEP is employed since it still allows for the determination of the proportion of the area predicted as positive.

Table 3. Fishing duration estimation results for A01 to A05.

Fold ID	Pred value [s]	True value [s]	MAE [s]
A01	1908.8	2688.5	1541.7
A02	2105.2	3531.1	2092.2
A03	2218.1	3648.8	2280.1
A04	2970.8	3853.0	2078.4
A05	1531.5	2917.2	1893.5

The AUC represents the area under the curve plotted with the RGEP on the x-axis and the recall on the y-axis for each threshold. As the predicted area of positive examples becomes narrower and more past positive examples are identified as positive, the AUC approaches one. Hence, values closer to one are desirable. The AUC, derived from the RGEP and recall, is well-suited for assessing data comprising positive examples and unlabeled data and finds application in landslide susceptibility prediction evaluations [4].

4.3 Experimental Results

Fishing Duration Estimation and Negative Example Extraction. Table 3 lists the results of experiments conducted on predicting smoothed fishing duration. The “Pred value” column presents the mean of predicted values for positive examples, “True value” represents the mean of actual values for positive examples, and “MAE” signifies the mean of mean absolute error between predicted and actual values for positive examples.

The data in Table 3 shows that across all folds, the mean of predicted values consistently undershoots those of the actual values. Moreover, the MAEs are relatively high compared to typical regression problems. Given this insufficient performance of the fishing duration estimation model, relying solely on it for predicting good fishing grounds is impractical. Instead, we primarily utilized this model to derive negative examples from unlabeled data. Therefore, this significant error may be considered negligible.

Table 2 lists the results of extracting negative examples from unlabeled data, with the same quantity as the positive examples for each fold.

Binary Classification for Predicting Good Fishing Grounds. Table 4 presents the results of binary classification for identifying good fishing grounds using past positive and extracted negative examples. Predictions were made for periods A01 to A05 using seven models.

The results in Table 4 highlight that the proposed PU learning models consistently achieved the highest values (highlighted in bold) for each period (A01 to A05), surpassing the performance of the existing AIT and ADT models. Notably, the top-performing model, PU learning model (30 epochs), demonstrated a performance improvement of 13.6% (0.660→0.750) compared to the baseline ADT

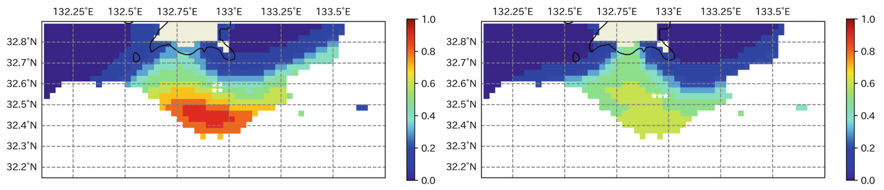
Table 4. Good fishing ground prediction results (AUC).

Model Name	A01	A02	A03	A04	A05	Mean	SD
inlier (AIT)	0.759	0.546	0.645	0.587	0.772	0.662	0.101
inlier (ADT)	0.704	0.567	0.708	0.563	0.756	0.660	**0.089**
PU (15 epochs)	**0.890**	0.510	**0.821**	0.631	0.800	0.730	0.156
PU (20 epochs)	0.884	0.531	**0.821**	0.652	0.797	0.737	0.143
PU (25 epochs)	0.871	0.567	0.806	**0.677**	0.816	0.747	0.124
PU (30 epochs)	0.856	**0.617**	0.775	0.659	**0.844**	**0.750**	0.108
PU (35 epochs)	0.839	0.571	0.756	**0.677**	0.765	0.722	0.102

model. This quantitative evaluation confirms the effectiveness of the proposed approach, demonstrating its capability to narrow down the prediction area without relying on geographical location and depth.

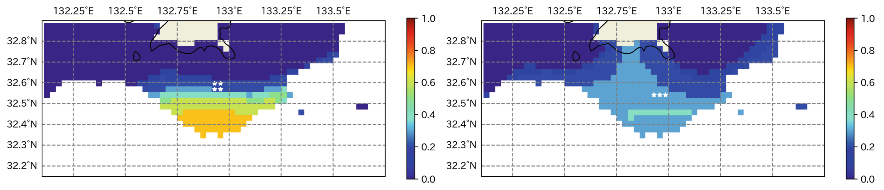
Example of Good Fishing Ground Prediction Maps. Figure 7 presents the success and failure cases of the prediction maps generated by three models, with the A01 period used as the test data. The success case corresponds to the prediction map at 06:00 on March 28, 2015, while the failure case corresponds to the prediction map at 10:00 on October 30, 2014. Focusing on the success cases, Fig. 7.1(a) illustrates the prediction map generated by the AIT model, indicating a wider area predicted as positive. Subsequently, Fig. 7.2(a) presents the prediction map produced by the ADT model. The implementation of the new thresholding system resulted in a reduction in the proportion of the predicted area compared to the overall target area (i.e., RGEP), effectively narrowing down the prediction area. However, the past positive examples were not identified as positive examples, leading to a low recall rate. In contrast to these models, Fig. 7.3(a) demonstrates the success of the proposed PU learning model in narrowing down the prediction area while maintaining a high recall rate. Furthermore, when compared to the distribution of past good fishing grounds depicted in Fig. 3, the proposed model succeeded in predicting areas with low frequencies of good fishing grounds. Thus, it is confirmed that the PU learning-based approach, utilizing unlabeled data as negative examples, enables narrowing down the prediction area without relying on prior knowledge such as location and depth.

Conversely, when examining the failure cases, Figs. 7.1(b), 7.2(b), and 7.3(b) did not successfully predict fishing grounds, as the past positive examples were not identified as high-likelihood spots. The primary cause for this failure is believed to be the deviation in sea patterns from the usual ones. While the prediction results of the proposed PU learning method missed the past fishing grounds, it still identified areas close to them as positive. Therefore, it can be concluded that the proposed PU learning method is capable of predicting sea areas more effectively than conventional methods, even in failure cases.



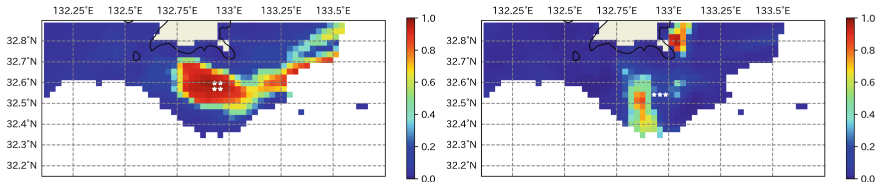
(a) Success case (b) Failure case

1. Inlier modeling (AIT)



(a) Success case (b) Failure case

2. Inlier modeling (ADT)



(a) Success case (b) Failure case

3. Proposed PU learning (15 epochs)

Fig. 7. Success and failure cases of prediction maps (A01, March 28, 2015, 06:00 and A01, October 30, 2014, 10:00, respectively). White stars on heatmap denote actual observed good fishing grounds at this time. Grids shaded from red to blue depict predicted target grids, where red signifies high likelihood of being good fishing grounds and blue suggests low likelihood. Grids without color are located outside target area. (Color figure online)

5 Conclusion

In this paper, we introduced a PU learning-based method designed to precisely narrow down the prediction area for good fishing grounds without relying on prior knowledge such as location and depth. Leveraging the two-step techniques framework of PU learning, we extracted negative examples from unlabeled data and integrated them into the learning process. The experimental results demonstrated a notable 13.6% enhancement in performance compared to conventional methods, affirming the efficacy of our proposed approach in accurately delimiting the prediction area without the need for prior knowledge.

Acknowledgment. The authors would like to thank JAMSTEC for providing the ocean condition simulation data and the Bulletin of Kochi Prefectural Institute of Fisheries for providing the trajectory data of the fishery for this study.

References

1. Adibi, P., et al.: Predicting fishing effort and catch using semantic trajectories and machine learning. In: Multiple-Aspect Analysis of Semantic Trajectories: First International Workshop, pp. 83–99 (2020)
2. Armas, E., Arancibia, H., Neira, S.: Identification and forecast of potential fishing grounds for anchovy (Engraulis ringens) in northern Chile using neural networks modeling. Fishes **7**(4), 204 (2022)
3. Bekker, J., Davis, J.: Learning from positive and unlabeled data: a survey. Mach. Learn. **109**(4), 719–760 (2020)
4. Fang, Z., Wang, Y., Niu, R., Peng, L.: Landslide susceptibility prediction based on positive unlabeled learning coupled with adaptive sampling. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **14**, 11581–11592 (2021)
5. Fu, A., Patil, K.R., Iiyama, M.: Region proposal and regression network for fishing spots detection from sea environment. IEEE Access **9**, 68366–68375 (2021)
6. Horiuchi, Y., Nakano, T., Miyazawa, Y., Ogawa, T.: Inlier modeling-based good fishing ground detection for efficient bullet tuna trolling using meteorological and oceanographic information. In: OCEANS 2022-Chennai, pp. 1–6 (2022)
7. Iiyama, M., Zhao, K., Hashimoto, A., Kasahara, H., Minoh, M.: Fishing spot prediction by sea temperature pattern learning. In: 2018 OCEANS-MTS/IEEE Kobe Techno-Oceans (OTO), pp. 1–4 (2018)
8. Japan Agency for Marine-Earth Science and Technology Application Laboratory. https://www.jamstec.go.jp/jcope/vwp/kochi/. Accessed 27 March 2024
9. Japan Meteorological Agency. https://www.data.jma.go.jp/obd/stats/etrn/. Accessed 27 March 2024
10. Konii, H., Nakano, T., Miyazawa, Y., Ogawa, T.: Narrow down forecast range: using knowledge of past operations and attribute-dependent thresholding in good fishing ground prediction. In: OCEANS 2023-Limerick, pp. 1–7 (2023)
11. Lee, M.a., Yang, W.C., Hung, I.C., Teng, S.Y.: Predicting winter potential fishing zones of albacore tuna (Thunnus alalunga) using maximum entropy models and remotely sensed data in the South Indian ocean. In: Proceedings 2018 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 957–959 (2018)
12. Lee, W.S., Liu, B.: Learning with positive and unlabeled examples using weighted logistic regression. In: ICML, vol. 3, pp. 448–455 (2003)
13. Liu, B., Dai, Y., Li, X., Lee, W.S., Yu, P.S.: Building text classifiers using positive and unlabeled examples. In: Third IEEE International Conference on Data Mining, pp. 179–186 (2003)
14. Miyazawa, Y., et al.: Temperature profiling measurements by sea turtles improve ocean state estimation in the Kuroshio-Oyashio confluence region. Ocean Dyn. **69**, 267–282 (2019)

15. Miyazawa, Y., et al.: Assimilation of high-resolution sea surface temperature data into an operational nowcast/forecast system around Japan using a multi-scale three-dimensional variational scheme. Ocean Dyn. **67**, 713–728 (2017)
16. Mu, H., Sun, R., Yuan, G., Shi, G.: Positive unlabeled learning-based anomaly detection in videos. Int. J. Intell. Syst. **36**(8), 3767–3788 (2021)
17. Mugo, R., Saitoh, S.I.: Ensemble modelling of skipjack tuna (Katsuwonus pelamis) habitats in the western north pacific using satellite remotely sensed data; a comparative analysis using machine-learning models. Remote Sens. **12**(16), 2591 (2020)
18. Mugo, R., et al.: Identification of skipjack tuna (Katsuwonus pelamis) pelagic hotspots applying a satellite remote sensing-driven analysis of ecological niche factors: a short-term run. PLoS ONE **15**(8), e0237742 (2020)
19. Shimura, T., Sonogashira, M., Kasahara, H., Iiyama, M.: Fishing spot detection using sea water temperature pattern by nonlinear clustering. In: OCEANS 2019-Marseille, pp. 1–4 (2019)
20. Wu, B., Qiu, W., Jia, J., Liu, N.: Landslide susceptibility modeling using bagging-based positive-unlabeled learning. IEEE Geosci. Remote Sens. Lett. **18**(5), 766–770 (2020)
21. Yoon, Y.J., et al.: An artificial intelligence method for the prediction of near-and off-shore fish catch using satellite and numerical model data. Korean J. Remote Sens. **36**(1), 41–53 (2020)
22. Zhang, J., Wang, Z., Meng, J., Tan, Y.P., Yuan, J.: Boosting positive and unlabeled learning for anomaly detection with multi-features. IEEE Trans. Multimedia **21**(5), 1332–1344 (2018)
23. Zhang, K., Zhuang, X.: ShapePU: a new PU learning framework regularized by global consistency for scribble supervised cardiac segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 162–172 (2022)
24. Zuluaga, M.A., Hush, D., Delgado Leyton, E.J.F., Hoyos, M.H., Orkisz, M.: Learning from only positive and unlabeled data to detect lesions in vascular CT images. In: Fichtinger, G., Martel, A., Peters, T. (eds.) MICCAI 2011. LNCS, vol. 6893, pp. 9–16. Springer, Heidelberg (2011)

SAda-Net: A Self-supervised Adaptive Stereo Estimation CNN For Remote Sensing Image Data

Dominik Hirner[1](✉) and Friedrich Fraundorfer[1,2]

[1] Institute of Computer Graphics and Vision, Graz University of Technology, Graz, Austria
{dominik.hirner,office,fraundorfer}@icg.tugraz.at

[2] Remote Sensing Technology Institute (IMF), German Aerospace Center (DLR), Cologne, Germany
https://www.tugraz.at/institute/icg/home ,
https://www.dlr.de/encontact-dlr@dlr.de

Abstract. Stereo estimation has made many advancements in recent years with the introduction of deep-learning. However the traditional supervised approach to deep-learning requires the creation of accurate and plentiful ground-truth data, which is expensive to create and not available in many situations. This is especially true for remote sensing applications, where there is an excess of available data without proper ground truth. To tackle this problem, we propose a self-supervised CNN with self-improving adaptive abilities. In the first iteration, the created disparity map is inaccurate and noisy. Leveraging the left-right consistency check, we get a sparse but more accurate disparity map which is used as an initial pseudo ground-truth. This pseudo ground-truth is then adapted and updated after every epoch in the training step of the network. We use the sum of inconsistent points in order to track the network convergence. The code for our method will be made available after acceptance at https://github.com/thedodo/SAda-Net

Keywords: stereo vision · deep learning · satellite images · aerial images · disparity estimation · remote sensing

1 Introduction

Stereo Vision has been a major topic of computer vision for many years. The goal of stereo vision is to extract 3D information of a scene using two neighboring images showing the same scene taken from different camera poses. 3D scene information is useful for many important applications, such as robotics, autonomous driving, 3D scene reconstructions or virtual and augmented reality. 3D reconstruction of urban aerial images is particular important for a multitude of use-cases, such as urban planning, environmental monitoring or disaster management and prevention.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 159–175, 2025.
https://doi.org/10.1007/978-3-031-78192-6_11

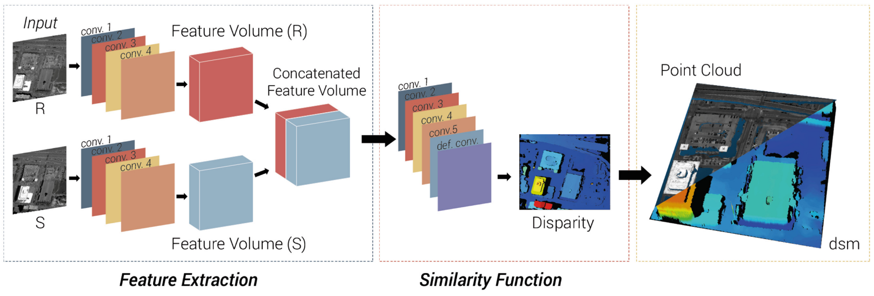


Fig. 1. Overview of our method. The input consists of two stereo rectified satellite image tiles. Our method does not need any additional input in order to be trained. We use the satellite stereo pipeline software (s2p) in order to project our depth estimation into world coordinates and create a point cloud as well as a digital surface model (dsm).

The stereo estimation algorithm consists of four main steps, namely: feature extraction, matching cost calculation, disparity estimation and disparity refinement. While traditional methods, such as SGM [1] or MGM [2] use handcrafted functions and features for the stereo estimation algorithm, recent work has seen major improvements by exchanging one or all of the steps using deep learning. Deep-learning approaches, especially the use of convolutional neural networks, in short CNNs have been successfully applied to many areas of computer vision, often improving upon traditional methods in speed and performance. One of the biggest drawbacks of deep learning methods however, is the reliance on accurate ground-truth data. Creating such ground truth data is time consuming and expensive and therefore reliable ground-truth data sources are not available for many domains. Furthermore, the generality of networks trained on a specific domain is questionable. Previous works, such as the work from L. Hu et al. [3] have shown that the generality of models is especially difficult for satellite images when trained with data from different continents. The creation of such ground truth for specific scenes is costly and often many hours of manual labour are needed for it. Especially creating detailed ground-truth data for the 3D-reconstruction task of urban scenes is often a futile task, as the scene will likely have changed by the time the data set is completed, as can be seen in Fig. 2. In this example, taken from the DFC2019 [11] dataset, a building can be seen in the image that is not present in the ground truth disparity data.

We tackle this problem by creating a training routine that is completely independent from any such created ground truth and only uses the rectified panchromatic images as input in order to guide the training process. Other such self-supervised stereo methods use the photo-consistency loss for training by warping one image using the estimated disparity map to warp the other image and then calculating the similarity between the warped image and itself. Using this loss on its own however will lead to some artefacts, especially for homogeneous areas [4]. Therefore, many works suggest to use a combined loss, for instance combining the photo-consistency loss with some sort of regularization

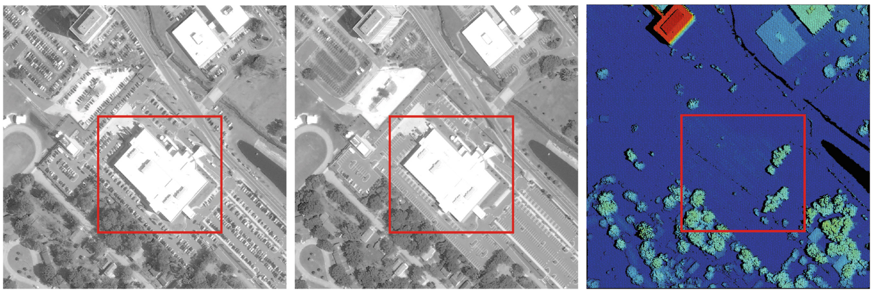


Fig. 2. This image shows one example of the 2019 IEEE Data Fusion Contest (DFC2019) [11] data set where the scene has changed between capturing the ground truth data and the image data. F.l.t.r.: rectified reference image, rectified second image and ground truth disparity map. The large building visible in the panchromatic images is missing in the disparity map.

term, like disparity smoothness or correspondence consistency between the prediction for the left and right image frame [4] [5]. Instead of the photo-consistency loss, we use a 'pseudo-ground truth' created by our initialization step. We use the left-right consistency check [6] in order to remove inconsistent points from the disparity map and consider the remaining disparity values as already correct. This pseudo ground-truth is then consistently updated after each training step. The evolution of such a ground truth over the epochs can be seen in Fig. 6. Furthermore we show that the amount of inconsistent points correlates strongly with the amount of incorrect points and can therefore be used to track the training process of the network. We show that while some consistent points in the ground truth are wrong, it does not influence the overall accuracy of the training process. An overview of the whole method is illustrated in Fig. 1.

In summary, our contributions are as follows:

- We create a novel training scheme based on the left-right consistency check that does not rely on any ground-truth data, making our training truly self-supervised.
- We create a CNN structure that is simple, lightweight and usable on most commercial hardware and yields good results for many different use-cases. With only $495K$ total trainable weights, it is lightweight when compared to other deep learning stereo methods which often use millions of trainable parameters.
- We show that our method produces good results on difficult real life scenes taken from the WorldView-3 satellite.

2 Related Work

Our work is based on previous works on stereo methods and self-supervised machine learning.

Traditional stereo methods use handcrafted features and similarity functions in order to match corresponding image points between two rectified image frames of the same scene. They can be grouped into three major groups, differing in their approach. **Local approaches** [7] [8] are fast but in general lack the accuracy of other methods. **Global methods** [9] [10] on the other hand have high accuracy, however their computational complexity make them unsuitable for many real life tasks. **Semi-global** approaches are a good trade-off between accuracy and computational complexity and are therefore the most popular used methods.

Semi-global matching (SGM) [1] from H. Hirschmüller approximates a 2D smoothness constraint by using multiple 1D line optimizations for each pixel in order to refine the overall accuracy of the disparity estimation.

G. Facciolo et al. [2] improves on this principle, by creating and using different aggregation elements. Instead of using the 16 cardinal directions for cost propagation, such as described by the Semi-global matching method, he creates and uses more complex structures. This helps with the block artefacts that can be produced by the update scheme of Semi-global matching. Even with the advances of machine-learning based methods, More global matching (MGM) is still a viable method that produces good results for real life examples and is therefore still a popular method in the remote-sensing community.

Deep Learning Stereo Estimation has been an successful endeavour in the last decade. One of the first deep learning based stereo methods is by J. Zbontar and Y. LeCun called MC-CNN [16]. In this work they popularized the shared-weight siamese network structure for stereo estimation. Variations of this structure has since been adapted by many state-of the art learning based stereo methods [17–21,24]. In their work, J. Zbontar and Y. LeCun furthermore define the training task as a binary classification task, where matching image patches are defined as the positive classes and non-matching patches are defined as the negative classes. The training goal is then to maximize the similarity of positive class samples while minimizing the similarity of negative samples.

GC-Net [21] uses 3D convolutions in order to regularize the cost-volume and uses soft-argmin for subpixel accuracy. Ga-Net [19] improves upon the results of GC-Net by getting rid of many of the time- and memory-intensive 3D convolutions. Instead they introduce a cost-aggregation block that uses less 3D convolutions and consists of a semi-global and local aggregation part in order to refine the cost-volume.

In their work J.R. Chang et al. [20] use a pyramid stereo matching network. They first use a pyramid pooling structure in order to improve context information for trained image features. Afterwards they create a cost-volume using this trained features. In the last step, the cost-volume is fed into stacked hourglass module using 3D-convolutions and a regression task is used for the final depth prediction.

J. Li et al. uses convolutional GRUs in a multi-level fashion in order to improve information propagation across the image. Their method, called RAFT-stereo [24] uses iterative refinement, traditionally used for optical flow estimation for the stereo task. They extract correlation features from the images at different resolutions and use the recurrency of the GRUs in order to iteratively improve the disparity estimation.

Following MC-CNN [16], our method uses a variation of the shared-weight siamese network structure. We use the same definition of the training classes as MC-CNN. We reformulate the min-max problem as a hinge-loss function. This is the same formulation that has been used in FC-DCNN [17] and FCDSN-DC [18].

Self-supervised Machine Learning refers to the task of learning a model without having corresponding human-annotated labels. In recent years, this method of training has seen success on a wide area of computer vision tasks including depth estimation. MonoDepth by C. Godard et al. [4] is among the most popular self-supervised depth estimation networks. While in their case, inference can be done on a single image frame (Monocular Depth Estimation), the method is trained on two image frames (Binocular or Stereo Depth Estimation). C. Godard et al. improves upon this method with their second version called MonoDepthv2 [5]. In particular they use an improved photo-consistency loss and introduce a mask that finds image points without any ego-motion. For the final prediction they use multi-scale images. With these additions they produce considerably better results.

Ma F. et al. create a self-supervised Sparse-to-Dense deep learning approach for monocular depth prediction in their work [29]. They create a deep regression network that takes sparse 3D LiDAR (Light Detection and Ranging) points for supervision and use stereo color images and the photometric warping loss together with a smoothing loss in order to predict a dense depth map.

P. Knöbelreiter et al. in their work [30] use their pre-trained CNN-CRF [33] model in order to generate ground truth data for the Vaihingen dataset for 3D reconstruction [32]. They first create the disparity maps using their pre-trained model, then remove inconsistent points using the left-right consistency check. The so created disparity maps are then used in order to fine-tune their already trained network. This leads to an improvement of the reconstructed scene.

In this work, instead of a warping or photo-consistency loss we introduce an adaptive pseudo ground-truth loss. We directly predict occluded points (i.e. points without ego-motion) by using the left-right consistency check [6] and do not rely on any additional input data.

3 Remote-Sensing Stereo Matching

In this work we use the area of interest (AOI) from the 2019 IEEE Data Fusion Contest (DFC2019) [11]. Furthermore we use the satellite stereo pipeline s2p [13] for the tiling, rectification and projection step.

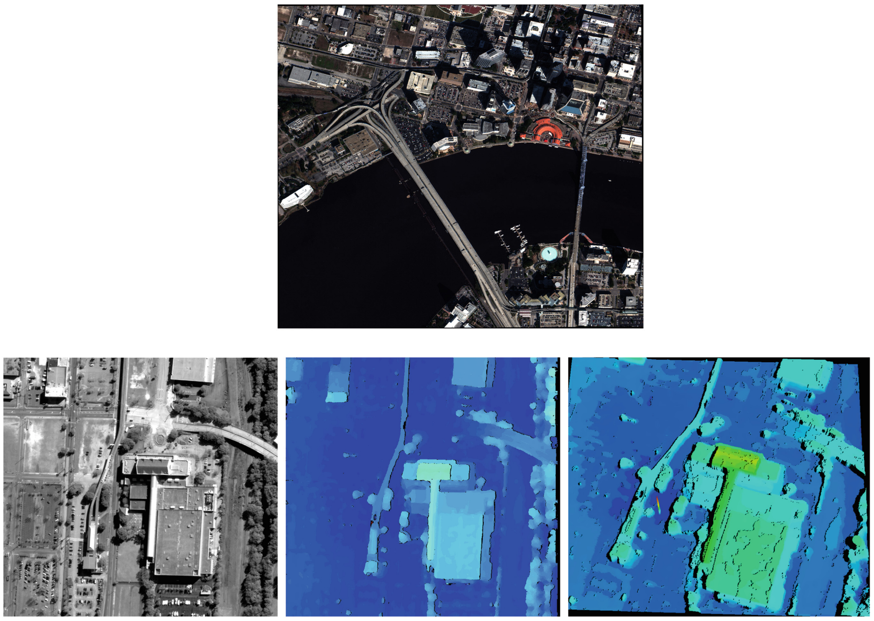


Fig. 3. First row: AOI from Jacksonville, Florida USA using the RGB bands of the multispectral image instead of the panchromatic image for the sake of better visualization. This image was captured using the WorldView-3 [26] satellite in january 2015. Second row f.l.t.r: one reference tile of the area of interest, the predicted disparity map and the projected digital surface model that resulted from it.

This dataset also provides a digital surface model (dsm) for the AOI, which was created using a LiDAR scan. We use this provided dsm for metric evaluation in our work. For the evaluations, the metrics as described by M.Bosch in their work [25] is used. An RGB image of the area of interest used in this evaluation, as well as an example of an extracted rectified reference tile, the disparity map predicted by our method and the resulting projected dsm can be seen in Fig. 3.

For this area, a ground truth dataset for the stereo matching task has been created. Rectified image pairs together with corresponding disparities have been released to the public. However, we only use the ground truth for the calculation of the end-point error, not for training itself. For the training, we use satellite images from the SpaceNet challenge dataset [27] of the same area of interest. To this end we use the panchromatic, not the multispectral images provided by the WorldView-3 satellite [26] as this band has a ground resolution of 0.31 m per pixel while the eight-band multispectral imagery has a ground resolution of 1.24 m per pixel.

4 Training Loop and Network

Figure 4 is a visual representation of how the training loop of our method works. First, two stereo rectified satellite images are put into our CNN which creates the disparity estimation for each frame. In the next step, the left-right consistency check [6] is used in order to get rid of inconsistent points which creates the sparse disparity map. This sparse disparity map is then used for the training patch creation for the current iteration. Using this patch, the hinge-loss is calculated and then back-propagated through the network. Figure 5 shows more detailed illustrations of our network (orange dashed box) and how the patches used for training are created (green dashed box).

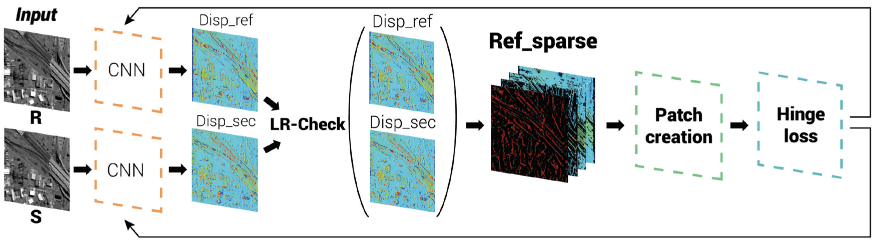


Fig. 4. Schematic of our training loop.

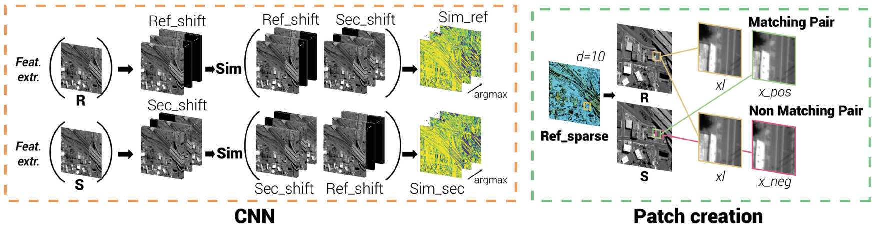


Fig. 5. Detailed illustration of the CNN and patch creation part of our training loop.

Our network consists of two main parts. The first part learns rich and deep image features of the satellite images, the second trains similarity functions between extracted features, improving upon handcrafted similarity functions.

The output of the feature extraction network part is a 60-dimensional image, e.g. $H \times W \times 60$. For the sake of interpretability, we continue to show the panchromatic image in Fig. 5. The trained deep feature images of the reference and the second image are concatenated and used as input for the similarity function part, the output is a 1 dimensional similarity measurement e.g. $H \times W \times 1$.

The green dashed box shows how the sparse disparity map of the current iteration is used for training patch creation. First, a patch around a random point with a consistent disparity value (in the current sparse ground truth) is chosen. The corresponding patch (in the same location) from the panchromatic reference image is extracted. Then the disparity value is used in order to get the position of the matching patch in the second image. A random small offset along the horizontal axis is then added for the non matching crop of the second image. After each training epoch, the sparse disparity maps are updated again and used as pseudo ground truth for the next epoch.

4.1 Implementation Details

We implement our method using Python3, pytorch 1.8.0 [15] and Cuda 12.4. We use gdal 2.4.2 for the manipulation of geo-tiffs and OpenCV [22] for other image manipulation. A single NVIDIA GeForce RTX 3090 consumer grade GPU is used for training. The network is trained using the Adam optimizer [14] with a learning rate of 6.0×10^{-5}. We use randomly cropped patches from the reference and second image with a size of 11×11 and a total batch-size of 500 for training. They are trained using a hinge-loss which maximizes the similarity between matching image patches and minimizes the similarity between close non-matching patches. Let s_+ be the similarity between two matching image patches extracted from the reference and the second image and s_- be the similarity between two non-matching image patches, then the loss is defined as a hinge-loss, as seen in Eq. 1.

$$loss = max(0, 0.2 + s_- - s_+). \tag{1}$$

Following previous works called FCDSN-DC [18] abd FC-DCNN [17] this loss is implemented using ReLU [31], which leads to a slight reformulation. The adapted loss can be seen in Eq. 2.

$$loss = ReLU(s_+ - s_- - 0.2). \tag{2}$$

4.2 Pseudo Ground-Truth

Our self-supervised training is based on an adaptive sparse-to-dense update scheme. For the initial step, the disparity map is calculated using the panchromatic image features from the satellite image with the cosine similarity function. Then, the left-right consistency check [6] is used in order to remove inconsistent points. The left-right consistency check is defined in Eq. 3.

$$|D^L(x, y) - D^R(x - d, y)| > 1.1. \tag{3}$$

This evolution of the predicted disparity map can be seen in Fig. 6. It shows that even if you start with a very sparse disparity map as initial ground truth, our update scheme produces dense and accurate maps after only 300 training epochs. Furthermore, it shows that in early epochs of the training process, only easy to match image points, such as corners or edges of foreground objects (i.e. roads and buildings) are consistent. Using this strongly matching pixels as a starting point, the network slowly learns more ambiguous image features. In theory, if no strong matching image points are found in the initial disparity

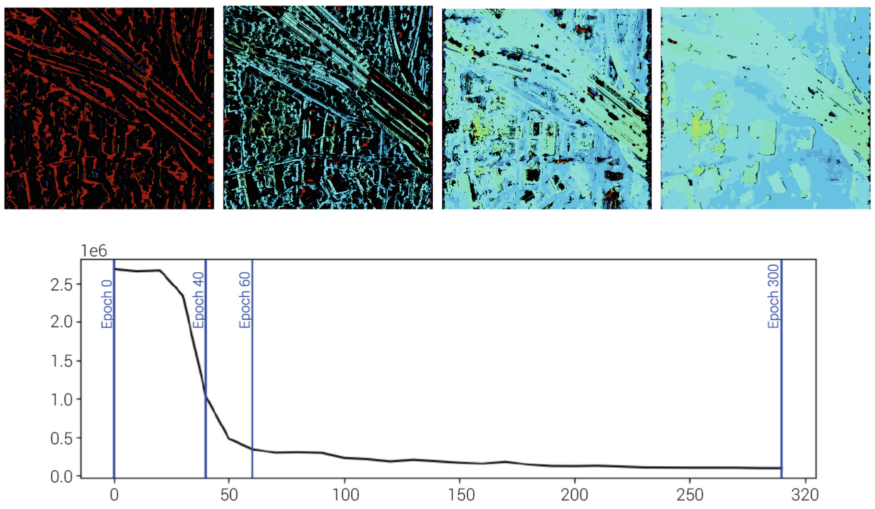


Fig. 6. Evolution of the disparity map used as ground truth for self-supervision. F.l.t.r.: Epoch 0, Epoch 40, Epoch 60 and Epoch 300. The evolution of the number of inconsistent points can be seen in the second row plot. The blue lines indicate the shown disparity maps. (Color figure online)

map, the network could get stuck early during training, however this has never happened in practise during our experiments. The second row of Fig. 6 shows the total number of inconsistent points used for the tracking. In our experiments, if the number of inconsistent points increases for 50 consecutive epochs, we stop the training process.

We argue, that the total sum of such removed points correlates with the amount of incorrect points and this can therefore be used for tracking network convergence and early-stopping. In order to show this, we conduct an experiment on 20 randomly chosen image pairs for the training split and 20 randomly chosen image pairs for the test split from the 2019 IEEE Data Fusion Contest 2019 (DFC2019) [11] challenge. We use the end-point error, as defined in Eq. 4 in order to track the accuracy of the disparity map. The threshold τ gives the number of how close the prediction has to be to the ground truth in order to be counted as correct. For example when using the 4-point error ($\tau = 4$), every predicted point which is within the range of ± 4 to the ground truth is counted as being a correctly predicted image point.

$$\sum |D_{gt} - D_{pred}| >= \tau. \tag{4}$$

4.3 Sub-pixel Enhancement

We follow the work of V.C. Miclea et al. [28] for our sub-pixel enhancement scheme. While many methods use machine learning to learn sub-pixel residuals for the disparity estimation, the feasibility and accuracy of those methods in a self-supervised training framework is an open research question to the best of our knowledge. The algorithm we use in our method is defined as follows:

$$d_{subpx} = \begin{cases} d_{Int} - 0.5 + arctan(\frac{ld}{rd}), & \text{if } ld \leq rd \\ d_{Int} - 0.5 + arctan(\frac{rd}{ld}), & \text{otherwise} \end{cases}$$

$$ld = c_{d-1} - c_d. \tag{5}$$

$$rd = c_{d+1} - c_d.$$

Here, d_{Int} is the chosen integer disparity value for the image point we want to get the sub-pixel value for. Depending on the implementation, this is either the position with the highest or lowest value in the cost volume. This cost (or similarity) is defined as c_d. Following c_{d-1} is the cost of the next pixel to the left of the highest matching pixel and c_{d+1} is the next pixel to the right of the most similar image point in the second image. In our implementation, the sub-pixel enhancement is used on the consistent points not removed by the left-right consistency check. It is not used in order to find more inconsistent points in the disparity map.

5 Experiments

In this section, we compare the scores produced by our method with the scores of other popular deep learning based methods and the baseline method used by the s2p [13] software MGM [2]. We use the publicly available trained weights for the other deep learning methods. We do not retrain or fine-tune the well-known state-of-the-art methods, because we want to show their accuracy if there is no possibility of fine-tuning or training from scratch because no ground truth data exists for the specific scene. Even though ground truth depth data is available for this particular AOI, it is missing for most remote-sensing data.

Table 1. Comparison on Jacksonville data

Method	MGM [2]	SAdaNet (ours)	PSMNet [20]	RaftStereo [24]
recall	0.894	**0.906**	0.800	0.828
precision	0.843	0.836	**0.891**	0.829
jaccardIndex	0.767	**0.769**	0.731	0.707
f-score	0.868	**0.870**	0.8445	0.828

We extend the s2p [13] pipeline, where the default matching algorithm is switched with our method or other deep learning based matching methods respectively for the experiment shown in Table 1.

As Table 1 shows our self-supervised method outperforms all other evaluated methods in the evaluated area of interest with the exception of precision, where PSMNet is slightly better. However PSMNet produces a less complete digital surface model, which can be seen by the lower completeness and f-score. Therefore we argue that our method still compares favourably. A result of our method, showing parts of the digital surface model of Jacksonville can be seen in Fig. 7.

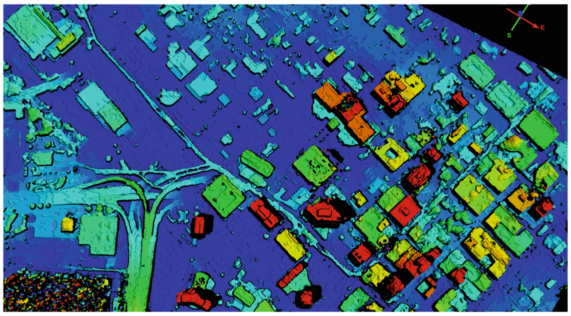


Fig. 7. Part of the digital surface model of the evaluated area created by our method.

5.1 Ablation Study

In this section we perform some ablation studies in order to show the validity and impact of our method. We perform every experiment on the same real life scenes taken from the WorldView-3 satellite as before, namely an area in Jacksonville, Florida USA. First, we show our method only using the feature extractor part of the network with cosine similarity and no sub-pixel enhancement. Then, we use the feature extractor as well as the trained similarity part and no sub-pixel enhancement. The last step then shows the accuracy of feature extractor, trained similarity and sub-pixel enhancement together.

Table 2. Ablation study on Jacksonville data

Method	cosine similarity	trained similarity	trained similarity + sub-pixel enhancement
recall	0.896	0.902	**0.906**
precision	**0.837**	0.834	0.836
jaccardIndex	0.763	0.765	**0.769**
f-score	0.865	0.867	**0.870**

Table 2 shows that the scores improve with each step, with the exception of the precision. This is expected as only using the trained feature extraction part of our method also leads to the sparsest reconstruction out of all three experiments conducted. As the whole area of interest is rather large, it is more difficult to show small differences between the methods. Therefore, to better illustrate the difference between only using the feature extractor and using the feature extractor plus trained similarity function, we show the disparity map of a small section from this ablation study created by these two methods in Fig. 8. Here one can see, that only training the feature extractor leads to a less complete reconstruction (inconsistent points are illustrated in black) when compared to training both the feature extractor as well as the similarity function.

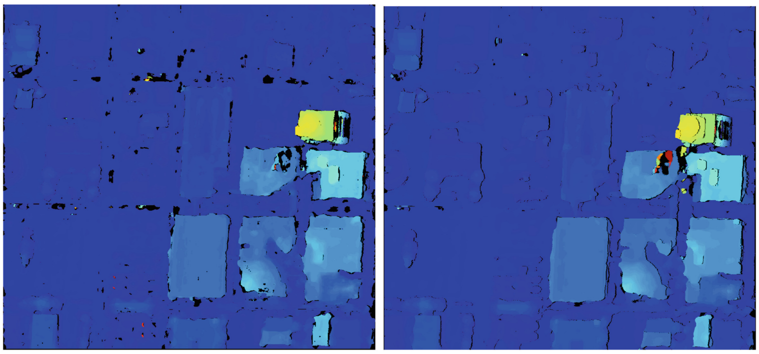


Fig. 8. Left: Disparity map created using only the feature extractor network. Right: Disparity map created using both the trained feature extractor and the trained similarity function. Missing predictions are marked in black.

5.2 Point Error Analysis

To further motivate the impact of our modules we conduct another experiment using the end-point metric and completion score in percentage from the available ground truth disparity from the DFC2019 dataset [11]. Instead of projecting our prediction into 3D space and comparing it to the ground-truth digital surface model, where each point which is within 1 m of the ground truth is counted as correct, we use the 4-point error and the 2-point error as to compare the 2D disparity maps for accuracy. For this τ in Eq. 4 is set to 4 and 2 respectively. We furthermore omit the sub-pixel enhancement for the evaluation, as we do not use it to detect additional inconsistent points, instead using it to refine consistent points in the disparity map. For this evaluation we select 20 random images from the dataset and remove samples with structures missing in the ground truth, such as Fig. 2 or interference created by passing planes. Due to the difficulty of creating ground truth disparities for aerial imagery, the ground truth disparity of flat surfaces in the disparity map observed in the data set can vary by up to four pixels. Therefore we argue that the 4-point error is the best measurement for this specific dataset. However as the sub-pixel enhancement impacts lower threshold end-point errors more, we also report on the 2-point error in this experiment.

The first column of Table 3 shows the accuracy of our trained feature extractor with the cosine similarity function. The second column shows the improvement of the accuracy when the similarity function is trained as well. The last column shows the improvement in accuracy when the sub-pixel enhancement as defined in Eq. 5 is used on the consistent points of the previous experiment.

Table 3 shows the improvement in accuracy for each added step of our method for the train split as well as the test split. One can see that the accuracy of the method is stable for untrained samples, only sacrificing slightly on accuracy and completeness. Furthermore it shows that the sub-pixel enhancement improves the accuracy of higher threshold end-point errors, such as the 4-point error, only slightly. However it has an impact on the accuracy of lower threshold end-point errors, such as the 2-point error. In order to motivate the correlation between the number of inconsistent points with the number of wrongly predicted points,

Table 3. Ablation study on DFC2019 data

Method	cosine similarity	trained similarity	trained similarity + sub-pixel enhancement
Train			
4-PE	12.064	8.979	**8.514**
2-PE	27.376	24.124	**18.079**
completion	85.755	**90.245**	–
Test			
4-PE	15.692	13.399	**13.396**
2-PE	32.753	30.191	**24.557**
completion	82.748	**84.603**	–

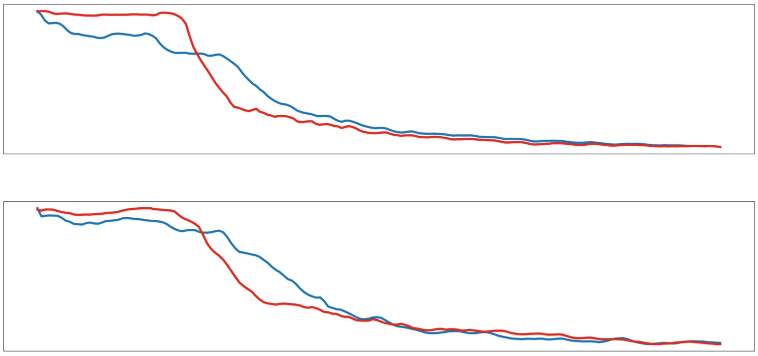


Fig. 9. Evolution of the number of inconsistent points (blue) and the end-point error of the training split (first row) and test split (second row). (Color figure online)

Fig. 9 shows the evolution of both for the train and test split of this experiment. The blue line shows the total amount of inconsistent points which are removed using the left-right consistency check of the predicted output of the network after each training step. The red line shows the total amount of wrongly predicted image points after each training step. Figure 9 visualizes the correlation between the two metrics and that they converge to the same local minimum.

5.3 Additional Datasets

In order to show the generality and robustness of our method, we perform additional experiments on datasets from different domains. To this end, we use the same hyperparameters, stopping criteria and training setup as for the previous experiments. We show qualitative and quantitative results for this experiments. As we want to cover a broad range of different domains, we train and evaluate our method on a dataset with indoor scenes, namely Middlebury [34] as well as a dataset for autonomous driving called KITTI2015 [35].

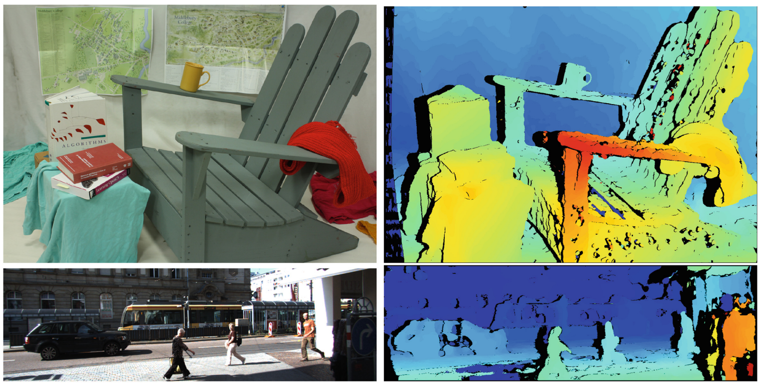


Fig. 10. Qualitative results of our method. From left to right, top to bottom: left image of an indoor scene from the Middlebury dataset, the resulting disparity map after training, left image of a driving scene from the KITTI2015 dataset, the resulting disparity map after training

Table 4. Results of our method on datasets from different domains

Dataset	4-PE	3-PE	2-PE	1-PE
Middlebury	18.054	19.820	22.526	29.196
KITTI2015	5.613	8.260	16.337	41.51

Table 4 shows different end-point errors of our method for Middlebury and KITTI2015. Figure 10 shows the left image and the predicted disparity map of our method for this image for one example of the Middlebury dataset and one example of the KITTI2015 dataset after training.

6 Conclusion

In this work, we have presented a fully self-supervised adaptive stereo method based on deep learning for remote-sensing application we call **S**elf-Supervised **ad**aptive convolutional neural **net**work or in short SAda-Net. We introduce a novel self-supervised training method which is based on adaptively updating the ground truth that is created by our network in each training step. To this end, we remove noisy and incorrect points from the map using the left-right consistency check. We argued for the feasibility of tracking inconsistent points in order to track the training process and network convergence if no other information is present. We then evaluated our method on a challenging real-life satellite imagery from the WorldView-3 satellite. We have shown that our method is able to compete with other state-of the art methods. While fine-tuning the trained weights of the evaluated methods on the new scenes can improve the accuracy of the specific method, we argue that accurate ground truth is often missing and too expensive to create. We therefore argue that our method, that does not rely on costly ground truth data but rather can use any satellite imagery for training is a step towards truly autonomous stereo vision for remote sensing.

References

1. Hirschmueller, H.: Accurate and efficient stereo processing by semi-global matching and mutual information. IEEE Computer Society Conf. Comput. Vis. Pattern Recog. (CVPR 2005), vol. 2, pp. 807-814 (2005)
2. Facciolo, G., et al.: MGM: A significantly more global matching for stereovision. In: BMVC 2015 (2015)
3. Hu, L., et al.: Model generalization in deep learning applications for land cover mapping, arXiv preprint arXiv:2008.10351 (2020)
4. Godard, C., et al.: Unsupervised monocular depth estimation with left-right consistency. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
5. Godard, C., et al.: Digging into self-supervised monocular depth estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2019)
6. Chang, C., et al.: On an analysis of static occlusion in stereo vision. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 722-723 (1991)
7. Zabih, R., Woodfill, J.: Non-parametric local transforms for computing visual correspondence. In: Eklundh, J.-O. (ed.) ECCV 1994. LNCS, vol. 801, pp. 151–158. Springer, Heidelberg (1994). https://doi.org/10.1007/BFb0028345
8. Min, D., Sohn, K.: Cost aggregation and occlusion handling with WLS in stereo matching. IEEE Trans. Image Process. **17**(8), 1431–1442 (2008)
9. Ohta, Y., Kanade, T.: (1985) Stereo by intra-and inter-scanline search using dynamic programming. IEEE Trans. Pattern Anal. Mach. Intell. **2**, 139–154 (1985)

10. Sun, J., et al.: Stereo matching using belief propagation. IEEE Trans. Pattern Analy. Mach. Intell. **25**(7), 787–800 (2003)
11. Le Saux, B., et al.: IEEE Dataport Data Fusion Contest 2019 (DFC2019) (2019). https://dx.doi.org/10.21227/c6tm-vw12
12. Mari, R., et al.: Disparity estimation networks for aerial and high-resolution satellite images: a review. Image Process. Line **12**, 501–526 (2022)
13. De Franchis, C., et al.: An automatic and modular stereo pipeline for pushbroom images. ISPRS Annals Photogrammetry, Remote Sensing Spatial Inform. Sci. **2**, 49–56 (2014)
14. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
15. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. Adv. Neural. Inf. Process. Syst. **32**, 8024–8035 (2019)
16. Zbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. J. Mach. Learn. Res. **17**, 1–32 (2016)
17. Hirner, D., Fraundorfer, F.: FC-DCNN: a densely connected neural network for stereo estimation. In: 2020 25th International Conference on Pattern Recognition (ICPR). IEEE (2021)
18. Hirner. D., Fraundorfer, F.: FCDSN-DC: an accurate and lightweight convolutional neural network for stereo estimation with depth completion. In: 2022 26th International Conference on Pattern Recognition (ICPR). IEEE (2022)
19. Zhang, F., et al.: Ga-net: Guided aggregation net for end-to-end stereo matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2019)
20. Chang, J.R., Chen, Y.S.: Pyramid stereo matching network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
21. Kendall, A., et al.: End-to-end learning of geometry and context for deep stereo regression. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
22. G. Bradski. *Open Source Computer Vision Library.* 2015
23. Wang, H., et al.: PVStereo: pyramid voting module for end-to-end self-supervised stereo matching. IEEE Robot. Autom. Lett. 6(3), 4353-4360 (2021)
24. Lipson, L., et al.: Raft-stereo: Multilevel recurrent field transforms for stereo matching. In: 2021 International Conference on 3D Vision (3DV). IEEE (2021)
25. Bosch, M., et al.: Metric evaluation pipeline for 3d modeling of urban scenes. Inter. Arch. Photogrammetry, Remote Sensing Spatial Inform. Sci. **42**, 239–246 (2017)
26. Longbotham, N., et al.: Measuring the spatial and spectral performance of WorldView-3. Hyperspectral Imaging and Sounding of the Environment. Optica Publishing Group (2015)
27. SpaceNet on Amazon Web Services (AWS). "Datasets." The SpaceNet Catalog. Last modified October 1st (2018). https://spacenet.ai/datasets/, Accessed 5 November 2024
28. Miclea, V.C., et al.: New sub-pixel interpolation functions for accurate real-time stereo-matching algorithms. In: 2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP). IEEE (2015)
29. Ma F., et al.: Self-supervised sparse-to-dense: self-supervised depth completion from lidar and monocular camera. In: 2019 International Conference on Robotics and Automation (ICRA). IEEE (2019)
30. Knöbelreiter, P., et al.: Self-supervised learning for stereo reconstruction on aerial images. In: IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium. IEEE (2018)

31. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML 2010) (2010)
32. Rottensteiner, F., et al.: Isprs test project on urban classification and 3d building reconstruction. Commission III-Photogrammetric Computer Vision and Image Analysis (2013)
33. Knobelreiter, P., et al.: End-to-end training of hybrid CNN-CRF models for stereo. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
34. Scharstein, D., et al.: High-resolution stereo datasets with subpixel-accurate ground truth. In: German Conference on Pattern Recognition. Springer, Cham (2014)
35. Menze, M., Geiger, A.: Object scene flow for autonomous vehicles. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2015)

Enhancing the Quality of Pseudo Labels in 2D Human Pose Estimation via a Debiasing-Teacher Approach

Naihao Wang, Shushi Hong, Kexin Meng, and Ruirui Li(✉)

Beijing University of Chemical Technology, Beijing, China
{wangnaihao,hongshushi,mengkexin,liruirui}@mail.buct.edu.cn

Abstract. The process of labeling data for intricate and specialized downstream tasks, including 2D human pose estimation, necessitates extensive expertise and can be financially burdensome. Consequently, pseudo-labels are increasingly becoming favored alternatives. In the context of 2D human pose estimation, frequent inter or intra-class imbalances often exhibit diverse locating challenges for poses due to variations in scale, motion, and occlusion. Such imbalances can potentially cause data bias and initiate self-training bias, ultimately resulting in a model collapse and increased model errors. Despite previous research efforts that have aimed to rectify model collapses through augmentation techniques and enhance model robustness via dual network training, two substantial issues remain unaddressed: (1) the learned representations often suffer from imbalanced distribution, and (2) this imbalance results in an increased amount of incorrect pseudo-labeling. To address these critical limitations, this paper introduces a novel reverse re-balancing module. This module optimizes representations to solve data bias caused by upstream imbalances. To underscore the module's efficacy, we have also proposed a data augmentation method tailored to it, identified as adaptive joint CutPaste. Furthermore, to mitigate the second highlighted problem, we propose deploying co-guessing techniques to amalgamate the pseudo labels. The effectiveness of our newly developed approach, Debiasing-Teacher, is scrutinized using two benchmark datasets designed explicitly for pose estimation—MSCOCO and MPII. Compared to preceding methods, our approach shows significant improvements, particularly in limited annotation scenarios. The source codes for Debiasing-Teacher can be accessed at https://github.com/wangnaihao/Debias-Teacher.

Keywords: 2D human pose · representation optimization · adaptive masking · debiased learning

R. Li—This work was supported in part by National Natural Science Foundation of China under Grant No. 62101021.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78192-6_12.

1 Introduction

2D human pose estimation serves a crucial role in understanding human movements. This concept focuses on accurately identifying the position coordinates of body joints within an image, providing initial comprehension of 2D human poses. Nevertheless, identifying different poses substantial challenges due to scale, movement, and occlusion variations. To ensure precise pose estimation in particular and convoluted downstream tasks, high-quality annotation data are essential for model training. Yet, acquiring key point data through annotation can be laborious and expensive, requiring specialized knowledge. Consequently, alternative solutions like pseudo-labeling approaches are prevalent in practical applications.

In both semi-supervised learning and noisy label learning, pseudo-labeling techniques can be integrated during the training process for iterative optimization. However, if data distribution is skewed, it introduces bias in the generation of pseudo-labels and exacerbates bias during model training. As early as 2021, Xie et al. [17] illustrated a pressing issue of model collapse when directly implementing semi-supervised techniques based on consistency to the 2D human pose estimation. This problem predominantly originates from the data imbalance in human postures, causing models to fail to recognize keypoints and predict all pixels in an image as the background. Xie et al. proposed the strong and weak data augmentation for keypoints to counteract the model collapse and regulate their consistency. Although this data augmentation improved the model collapse to a certain degree, the problem of inaccurate pseudo-labels persisted. Inspired by noisy label learning, Huang et al. [5] established two teacher networks simultaneously to increase pseudo-labeling's robustness. Nonetheless, there has been limited discourse on the bias of pseudo-labels and the model, and effective solutions have yet to be proposed.

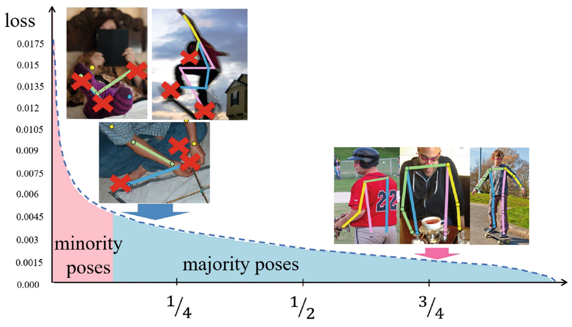


Fig. 1. Loss per image of Xie's Dual [17] on COCO's VAL dataset, categorized by OSTU's [11] thresholding into minority and majority subclasses.

To better illustrate, we plot the loss distribution of Xie's Dual [17] model on the COCO validation dataset, which was semi-supervised on 1K annotated data.

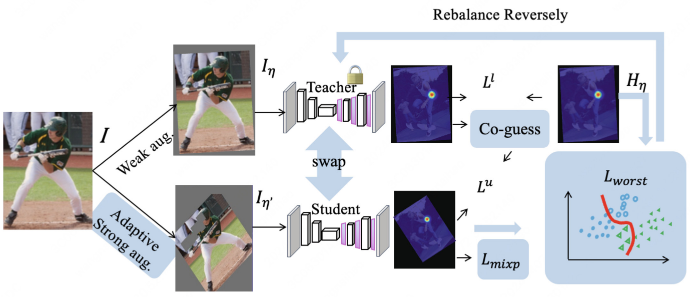


Fig. 2. The overall architecture of Debiasing-Teacher.

We use the OSTU [11] algorithm to classify the loss distribution data into two subclasses as follows: red represents the minority subclass and blue represents the majority subclass. It is worth noting that about one-eighth of the samples have large errors in keypoint locations. A more focused observation shows that these samples often focus on rarer 2D human poses or poses perceived from less common angles. This indicates that the representations obtained by learning human pose images are distributed unbalanced. The model exhibits a preferential bias towards frequently encountered scenarios. Pseudo-label estimates from skewed models further reinforce this bias during training, leading to confirmation bias.

Inspired by the concepts of adversarial learning, we propose a reverse rebalancing module to address the issue of uneven representation distribution, thereby mitigating model bias. Accompanying this module, we introduce an adaptive masking data augmentation technique aimed at reducing the variation in learning difficulty among samples. To alleviate the impact of incorrect pseudo-labels on model training, we further propose a collaborative guessing module based on role-swapping learning within a teacher-student network framework, which significantly improves the accuracy of pseudo-labels. We conducted experiments using different numbers of labeled samples on two 2D human pose estimation benchmark datasets to validate the effectiveness of our debiased teacher method. The results demonstrate that our approach surpasses the current state-of-the-art semi-supervised methods for 2D human pose estimation [5,19]. Our contributions can be summarized as follows:

– Proposal of a reverse re-balancing module to optimize the biased feature distribution.
– Development of an adaptive joint CutPaste data augmentation method based on keypoints' confidence scores.
– Introduction of teacher-student network role-swapping training and co-guessing mechanisms to enhance the robustness of the model and the quality of pseudo labels.

2 Related Work

2.1 2D Human Pose Estimation

Heatmap-based methods have recently garnered increased attention for their application to human pose estimation. The advantages of these methods relate to their capacity for preserving spatial positional information, aligning with the design of Convolutional Neural Networks (CNN), and ultimately boosting prediction accuracy. Research endeavors have also examined the multiscale nature inherent to the human body, leading to the introduction of diverse network architectures such as Simple Baseline [16], HRNet [18] and DNANet [22]. These architectures serve to augment contextual semantic perception. Furthermore, investigations into the correlation between heatmaps and coordinates have led to reductions in quantization errors common to heatmap-based strategies, thus enhancing keypoint localization precision. However, these methods stand to lose their effectiveness in scenarios with a deficiency of labeled data. This research narrows its focus to 2D human pose estimation in contexts with limited annotation. A top-down heatmap regression approach forms the basis of its methodology.

2.2 Semi-supervised 2D Pose Estimation Based on Pseudo-Labeling

Semi-supervised methods, such as MixMatch [2] and FixMatch [12] generally adhere to the consistency principle, which suggests that even if a sample is perturbed, the network should still produce outputs similar to the original sample. Suitable data augmentation methods can provide such perturbations, optimizing the decision boundary and reducing the tendency for it to pass through high-density regions of the data distribution. Xie et al. [17] first introduced consistency learning to the field of 2D human pose estimation. They discovered that due to data imbalance, they were unable to effectively learn the diversity of different poses, and due to collapse, all unlabeled images were mistakenly recognized as background. To address this, they proposed a technique called joint CutOut, which enhances keypoint masking and enables the weakly augmented teacher network to supervise the strongly augmented student network. While this method effectively prevents mode collapse, there still exist discrepancies between the learned decision boundary and the true decision boundary, resulting in unavoidable noisy pseudo labels. Huang et al. [5] employed dual networks to improve the quality of pseudo labels. These articles posit that model bias is a key factor that affects the quality of pseudo labels and the accuracy of the model.

2.3 Learning on Imbalanced Data

It is widely acknowledged that models trained on imbalanced data tend to exhibit bias towards the majority classes, which have abundant examples, while neglecting the minority classes, which have limited samples. Several strategies have been proposed to mitigate this bias, such as re-sampling, reweighting, and two-stage training. These techniques rely on using labels to rebalance the biased model. In contrast, the study of semi-supervised learning (SSL) on imbalanced data has been relatively insufficient. Imbalanced data poses additional challenges for SSL, as the lack of label information prevents the re-balancing of the unlabeled set. Although a handful of recent studies [3] [14], have addressed this issue, further exploration is still needed on how to better avoid and eliminate bias, especially with the existence of pseudo-labels.

3 Preliminary

The objective of 2D human pose estimation is to detect the positions of K body joints in the image I. Most top-down approaches assume a Gaussian distribution for the localization error and convert the problem into estimating K Gaussian heatmaps $H = \{h_1, .., h_K\}$. Each heatmap represents the probability of a joint being located at a specific position in the image. For the ground truth position of the k-th joint at (u_k, v_k), the probability value at a point (u, v) on the heatmap is calculated as:

$$h_k(u,v) = e^{-\frac{\left((u-u_k)^2+(v-v_k)^2\right)}{2\sigma^2}}. \tag{1}$$

Here, σ is the variance parameter and e is the exponential operator. During inference, the position of the k-th joint is determined by selecting the coordinates corresponding to the maximum value on the k-th heatmap.

In semi-supervised 2D human pose estimation, the labeled and unlabeled training data are defined as follows: $\mathcal{L} = \{(I^l, H^l)\}_{l=1}^{N}$ for labeled data and $\mathcal{U} = \{(I^u)\}_{u=i}^{M}$ for unlabeled data. Here, N represents the number of labeled samples and M represents the number of unlabeled samples. The parameters of the teacher and student networks are denoted as θ and δ respectively. f is the neural network function. The augmentation methods applied to the teacher and student networks are represented by η and $\eta\prime$ respectively. The Gaussian heatmaps obtained after applying the augmentation transformation to the teacher network are denoted as H_η. The supervised loss on labeled data can be defined as:

$$L^{\mathcal{L}} = \mathbb{E}_{i\in\mathcal{L}} \|f(I_\eta, \theta) - H_\eta\|^2. \tag{2}$$

In unannotated data, by leveraging the assumption of consistency, the pseudo labels generated by a teacher network using weakly augmented samples are used to supervise the learning of strongly augmented samples by a student network. Therefore, the unsupervised loss is defined as:

$$L^{\mathcal{U}} = \mathbb{E}_{i\in\mathcal{U}} \|f(I_\eta, \theta) - f(I_{\eta'}, \delta)\|^2. \tag{3}$$

In summary, the overall loss function for semi-supervised pose estimation is the weighted sum of these two losses, where α represents the weight of the unsupervised loss.

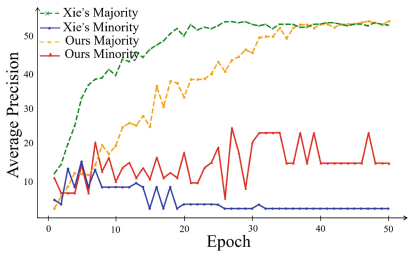
$$L_{total} = L^{\mathcal{L}} + \alpha \cdot L^{\mathcal{U}}. \tag{4}$$

4 Debiasing-Teacher

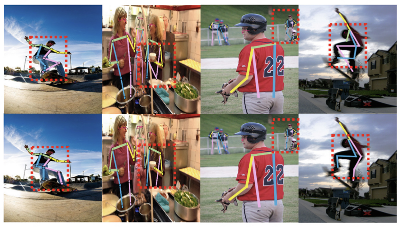
Figure 2 illustrates the comprehensive structure of the Debiasing-Teacher method. It showcases three crucial enhancements over the approach suggested by Xie et al. [17]: the introduction of a module that equalizes representation distribution by optimizing worst-case regression loss; the employment of an adaptive strong augmentation scheme that relies on confidence scores, accompanied by random feature mixup augmentation; and the betterment of pseudo-label quality through the application of the dual network role swapping and co-guessing technique. Detailed expositions of these improvements will be provided in the following subsections.

4.1 Reverse Re-Balancing

The human body exhibits a power-law distribution of poses, resulting in significant variations in the difficulty of recognizing keypoints across different poses. This leads to a severe intra-class imbalance, causing data bias. This deviation becomes more pronounced as the model is trained in the context of semi-supervised 2D human pose estimation tasks with pseudo-labeling errors, leading to a decline in its ability to generalize and affecting its robustness in real-world scenarios.



(a) Comparison of mAP



(b) Comparison of result visualization

Fig. 3. Mean average precision curves for minority and majority subclasses during training: a comparison between our approach and Xie et al.'s Dual [17], using 1K COCO annotations and ResNet18.

As Fig. 1 depicted, we analyzed Xie's model on COCO's validation set, examining the loss per image and dividing the data into two subclasses - minority and majority. Throughout the training process, we tracked the changes in the mAP for both subclasses, as illustrated in Fig. 3. As anticipated, the mAP for the minority class not only failed to improve but displayed a downward trend. Our findings indicate that minorities lack attention during training and are affected by pseudo-label errors, leading to incorrect predictions of joint positions.

To mitigate this bias, from a feature's aspect, we propose a reverse rebalancing module. This module optimizes the representation distribution and reduces the generation of erroneous pseudo labels from the upstream. Unlike classification tasks, pose estimation is a regression localization task for K keypoints, where there is no classification head or hard decision boundary. Therefore, the core idea of the reverse re-balancing module is not to find the worst classification boundary but to define a new reverse regression loss and optimize the worst projection header that maps features to heatmaps. In our approach, we aimed to model the worst-case scenario for the prediction of unlabeled samples while striving to maintain accurate predictions for labeled samples. For the unlabeled samples, we compared the pseudo-labels predicted by the model with those predicted by the projection headers. The projection header that demonstrated the largest discrepancy between the model's pseudo-labels and its own pseudo-labels was identified as the "worst header." Let g represent the learned header. By optimizing the following equation, we can derive the worst-case header, g_{worst}:

$$g_{worst} = \underset{g}{argmax}(L^{\mathcal{U}}(\theta, g, \hat{H}_\eta) - L^{\mathcal{L}}(\theta, g, H_\eta)). \tag{5}$$

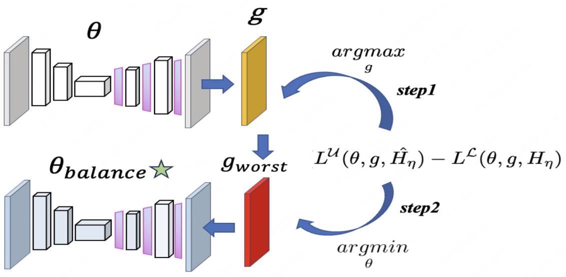
In this context, θ represents the parameters of the Teacher's feature encoding network, H_η denotes the weakly augmented Gaussian heatmaps of annotated samples, and $\hat{H}_\eta$ indicates the weakly augmented pseudo heatmaps of unlabeled samples. The worst header is obtained by minimizing the regression loss on annotated data while maximizing it on unlabeled data.

This worst header is crucial as it defines the worst-case loss, denoted as L_{worst}. To mitigate these adverse scenarios, we implemented a reverse rebalancing mechanism to steer the model's predictions away from the worst-case situations. In the reverse optimization process, we optimize the network parameters θ by minimizing the defined L_{worst} loss:

$$L_{worst} = L^{\mathcal{U}}(\theta, g_{worst}, \hat{H}_\eta) - L^{\mathcal{L}}(\theta, g_{worst}, H_\eta)). \tag{6}$$

Figure 4 illustrates the reverse re-balancing module, which was initially inspired by the principles of adversarial training. Both the worst-case predictor and the target predictor utilize a shared feature extractor, ensuring feature consistency across different prediction tasks. This module promotes the generation of robust and compact feature representations for unlabeled samples, enabling them to achieve minimal error rates even when paired with the least effective projection head. Within the latent space, the module facilitates the clustering of similar samples while maintaining clear distinctions between dissimilar ones. This approach effectively mitigates disparities in feature learning that commonly arise due to variations in pose and other confounding factors.

Since pseudo-labels are dynamically changing, the worst-case projection header also evolves during the training process, and both are trained simultaneously. The dynamic nature of the target predictor allows it to continue training even after the worst-case header has converged, thereby enhancing the model's accuracy and fairness.



θ
g
$\underset{g}{argmax}$
step1
$\theta_{balance}$
g_{worst}
$L^{\mathcal{U}}(\theta, g, \hat{H}_\eta) - L^{\mathcal{L}}(\theta, g, H_\eta)$
step2
$\underset{\theta}{argmin}$

Fig. 4. Reverse re-balancing module. Step1: optimization of the worst header. Step2: reverse optimization of network encoding parameters via the worst header.

4.2 Adaptive Augmentation

Joint CutPaste. In 2D human pose estimation tasks, the uneven distribution of features is often caused by the imbalance in sample difficulty. To address this issue, we present an adaptive data augmentation method that utilizes confidence scores. This approach effectively categorizes training samples as either easy or hard, and applies specific augmentation strategies accordingly. By doing so, it achieves a balanced learning difficulty across different image samples.

Let's assume that there are K keypoints in the i-th image, and the highest probability value in the predicted heatmaps corresponding to the k-th keypoint is denoted as p_k. The confidence score for the i-th image is defined as follows:

$$C_i = \frac{1}{\mathrm{K}} \sum_{k=1}^{\mathrm{K}} p_k. \tag{7}$$

In each batch, we employ the Gaussian Mixture Model (GMM) to fit the standardized confidence scores, obtaining the final difficulty coefficient:

$$\text{degree } = \mathrm{GMM}\left(\frac{C_i - \min\limits_{\mathcal{B}}(C_i)}{\max\limits_{\mathcal{B}}(C_i) - \min\limits_{\mathcal{B}}(C_i)}\right). \tag{8}$$

In our experiment, $\mathcal{B}$ represents the current batch of samples, and $GMM()$ corresponds to the process of fitting a Gaussian mixture distribution. Samples with a degree greater than 0.5 are classified as easy, while those with a degree less than or equal to 0.5 are classified as hard. Our study reveals that increasing the number of easy samples, also known as naive samples, does not significantly improve the model's accuracy. Conversely, hard samples display low confidence in regression heatmaps. Supervising the student network's learning using these low-quality pseudo-labeling heatmaps can result in the acquisition of incorrect information and the emergence of biases during subsequent training. Therefore, we randomly remove P keypoints from the easy samples to increase the learning difficulty. Simultaneously, we eliminate Q keypoints from the hard samples and replace them with some high-confidence easy keypoints. The values of P and Q are determined through experiments, which are described in detail in the Appendix.

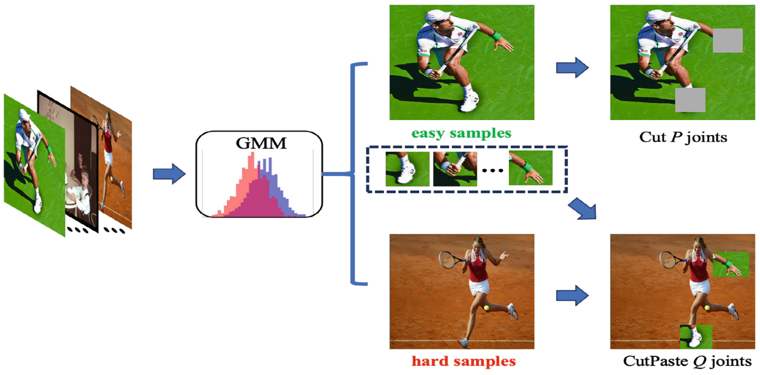


Fig. 5. Pipeline of joint CutPaste.

Random Feature Mixup. Mixup is a widely used technique in semi-supervised learning to enhance model performance by promoting a linear understanding of training samples. This technique improves robustness and generalization. When working with 2D images of human body poses, simply augmenting data at the data level may lead to similar combinations of features, which can worsen the imbalance in data distribution. For instance, mixing two similar samples in the feature space may still result in a similar sample. To address this issue, we adopt a random feature mixup augmentation. In this approach, we randomly select a feature layer from the forward feedback network and mix two random samples at that specific layer. The specific method of random mixup was inspired by [13]. And the implementation details can be found in the appendix.

Swap Role Training and Co-Guessing. The proposed training framework retains the original parameters of the teacher network, which do not undergo gradient updates. To modify the teacher network, we utilize a training approach termed role swapping. This method involves interchanging the parameters of the teacher network and the student network. Post this interchange, the student network inherits the parameter values from the teacher network, enabling its continued learning. Simultaneously, the teacher network is augmented with the parameter values originating from the student network. Based on these networks' predictions, we introduce a simple yet efficient method named co-guessing to produce pseudo labels. This process involves fusing the predicted heatmaps from both the teacher and student networks at a specific ratio, thereby creating the blended heatmap serving as pseudo labels. This strategy enables both networks to contribute towards pseudo-label prediction, consequently mitigating the effect of any erroneous memories in the student network during subsequent training.

5 Experiment

5.1 Dataset

COCO [8]. The COCO dataset consists of 200K images, which are divided into four subsets: TRAIN, VAL, TEST-DEV, and TEST-CHALLENGE. Additionally, it includes 123K unlabeled data called WILD. We randomly select 1K, 5K, and 10K labeled data from the TRAIN subset. In some experiments, we use the remaining images from the TRAIN subset as unlabeled data. In other experiments, we use the entire training set as labeled data and WILD as unlabeled samples. The results are reported in terms of mAP (Average AP over 10 OKS thresholds) evaluation metric.

MPII. [1] and **AI-Challenger** [15] are two widely used datasets in the field. MPII consists of approximately 25K images and 40K annotated human instances. AI Challenger, on the other hand, contains 210K data samples with 370K images that include annotations for 14 keypoints. In our approach, we utilize the MPII dataset as the labeled data and the AI Challenger dataset as the unlabeled data. To evaluate our method, we adopt the PCKh@0.5 metric.

Table 1. mAP of different methods on COCO [8] with 1K, 5K and 10K labels are used.

Method	1K	5K	10K	All
Supervised [16]	31.5	46.4	51.1	67.1
PesudoPose [17]	37.2	50.9	56.0	–
DataDistill [10]	37.6	51.6	56.6	–
Cons [17]	42.1	52.3	57.3	–
Dual [17]	44.6	55.6	59.6	–
SSPCM [5]	46.9	57.5	60.7	–
Denoising [19]	47.58	58.04	61.3	–
Ours	**50.8**	**60.1**	**61.8**	–

5.2 Data Augmentation

For weak augmentation, we adopt the same method as previous studies [5], which randomly rotates an angle between -30 and 30°C and randomly scales a factor between 0.75 and 1.25. For strong augmentation, the range of random rotation angles is increased to -60 to 60°C, and on top of random rotation and scaling, the joint CutPaste augmentation proposed in Sect. 4.2.1 is added.

Table 2. Results of using different network structures for the teacher network and the student network on COCO [8].

Method	Teacher	Student	1K	5K	10K
Supervised [16]	-	ResNet18	31.5	46.4	51.1
Supervised [16]	-	ResNet50	34.4	50.3	56.3
Dual [17]	ResNet18	ResNet18	44.6	55.6	59.6
Dual [17]	ResNet50	ResNet50	48.7	61.2	65.0
Dual [17]	ResNet50	ResNet18	47.2	57.2	60.4
SSPCM [5]	ResNet18	ResNet18	46.9	57.5	60.7
SSPCM [5]	ResNet50	ResNet50	49.4	61.6	65.4
SSPCM [5]	ResNet50	ResNet18	48.3	58.9	61.9
Denoising [19]	ResNet18	ResNet18	47.58	58.04	61.3
Denoising [19]	ResNet50	ResNet50	50.1	62.14	65.36
Ours	ResNet18	ResNet18	50.4	59.8	61.5
Ours	**ResNet50**	**ResNet50**	**54.5**	**64.0**	**67.2**
Ours	ResNet50	ResNet18	51.1	62.3	65.0

5.3 Comparison with SOTA Methods

We employed ResNet18 as the backbone and performed initial comparative experiments on the COCO dataset. The experiments used 1K, 5K, and 10K annotated training data, while the remaining data in TRAIN was considered unannotated. Table 1 displays the performance of our approach in comparison to the state-of-the-art methods. As anticipated, the results achieved solely using labeled data were the worst. In contrast, our proposed method outperformed the latest semi-supervised 2D human pose estimation method, achieving improvements of 3.9% mAP, 2.6% mAP, and 1.1% mAP with 1K, 5K, and 10K labeled data, respectively.

To evaluate how the backbone network's architecture affects model performance, a second series of experiments was conducted. In these tests, either the teacher or student ResNet18 network was replaced with a deeper ResNet50 network. The results of these experiments are summarized in Table 2. As with the previous set of experiments, this series adopted the same settings, but placed emphasis on contrasting the influence of network architecture in four distinct methodologies: supervised learning, Dual Method by Xie et al., SSCPM, and our methodology. The findings suggest that using a deeper network bolsters the localization accuracy across all procedures, particularly when both the teacher and student networks employ the deeper model. In our methodology, replacing ResNet18 with ResNet50 led to an increase of 4.1% mAP, 4.2% mAP, and 5.7% mAP at scales of 1K, 5K, and 10K, respectively.

We conducted large-scale experiments in semi-supervised pose estimation. These experiments used the entire TRAIN set from COCO as labeled data and the WILD set was utilized as unlabeled data. Four diverse backbone network architectures were included in these experiments. A comparative analysis was conducted using the VAL data, comprising of 6352 images. The aggregate results are shown in Table 3. Regardless of the backbone network architecture used, our proposed method consistently outperformed the other three methods. This indicates that the superiority of our method mainly stems from the methodological aspect and is independent of the chosen backbone network.

To underscore the efficacy of our method, we conducted an extensive comparison comprising all respective methods, using the mAP and mAR metrics on the complete TESTING-DEV dataset from COCO, as detailed in Table 4. The HRNetW48 backbone network was utilized in the model training phases by Dual, SSPCM, and our method. Our novel method, in conjunction with DARK, surpassed the runner-up SSPCM+DARK by 0.4% and 0.3% in the mAP and mAR categories respectively, hence achieving a mAP of 77.9% and a mAR of 82.7%.

Table 5 displays the results of experiments conducted on MPII and AI-Challenger. The MPII training set served as the labeled data, while the AI-Challenger was utilized as the unlabeled data. The evaluation was carried out on the MPII test set, employing the PCKh@0.5 metric for assessment. Our method was compared with recent approaches and it was found that our technique had the superior accuracy in identifying the keypoints of various poses, boasting a total precision of 93.6%. Remarkably, we made a significant advancement in the challenging ankle joint, improving the precision from 87.1% to 87.6%. This underscores that our proposed method effectively minimizes bias and boosts model generalization.

5.4 Ablation Study

The efficacy of the proposed method was assessed using a series of ablation studies carried out on the annotated COCO 1K, 5K, and 10K data, with tests executed on the VAL set. ResNet18 was employed for the experiments. As a baseline, the study used our technique, deducting one of the four modules in every cycle: reverse re-balancing, joint CutPaste, random feature Mixup, and co-guessing. The intention behind this was to monitor any drop in the model's mean accuracy. The findings are displayed in Table 6. The results reveal that each module played a part in the efficiency of the method, with reverse re-balancing and random feature Mixup having a more pronounced influence.

5.5 Discussion on the Debiasing Effects

Our approach achieves a better balance between majority and minority, while effectively reducing bias through the use of a re-balancing module and random feature Mixup augmentation. The results were compared with the method proposed by Xie et al. and our approach in terms of mAP changes of minority and majority groups during the training process, as shown in Fig. 3a. It can be observed that our method shows small changes in mAP for majority subclasses, while significantly improving mAP for minority subclasses. Figure 3b also demonstrates the superiority of our method.

5.6 Discussion on Potential for Overfitting

This study employs a teacher-student framework to leverage unlabeled data and uses consistency regularization to prevent the model from overfitting to the labeled data. Given the use of pseudo-labeling, the model could potentially overfit to erroneous information in the pseudo-labels within the supervised branch. To address the errors in pseudo labels, we utilize a co-guessing approach. While our method mitigates some risks, it does not eliminate bias or the risk of overfitting.

Table 3. Results on the COCO [8] VAL set when using all images from the TRAIN set as the labeled set and all images from the WILD set as the unlabeled set.

Method	Network	AP	AP.5	AR	AR.5
Supervised [16]	ResNet50	70.9	91.4	74.2	92.3
Dual [17]	ResNet50	73.9	92.5	77.0	93.5
SSPCM [5]	ResNet50	74.2	92.7	77.2	93.8
Denoising [19]	ResNet50	74.1	–	–	–
Ours	**ResNet50**	**74.6**	**93.5**	**77.5**	**94.1**
Supervised [16]	ResNet101	72.5	92.5	75.6	93.1
Dual [17]	ResNet101	75.3	93.6	78.2	94.1
SSPCM [5]	ResNet101	75.5	93.8	78.4	94.2
Denoising [19]	ResNet101	75.7	–	–	–
Ours	**ResNet101**	**75.7**	**94.0**	**78.6**	**94.4**
Supervised [16]	ResNet152	73.2	92.5	76.3	93.2
Dual [17]	ResNet152	75.5	93.6	78.5	94.3
SSPCM [5]	ResNet152	75.7	93.7	78.6	94.5
Denoising [19]	ResNet152	76.0	–	–	–
Ours	**ResNet152**	**76.4**	**93.9**	**79.3**	**94.6**
Supervised [16]	HRNetW48	77.2	93.5	79.9	94.1
Dual [17]	HRNetW48	79.2	94.6	81.7	95.1
SSPCM [5]	HRNetW48	79.4	94.8	81.9	95.2
Denoising [19]	HRNetW48	79.4	–	–	–
Ours	**HRNetW48**	**79.6**	**94.9**	**82.1**	**95.3**

5.7 Discussion on Limitations

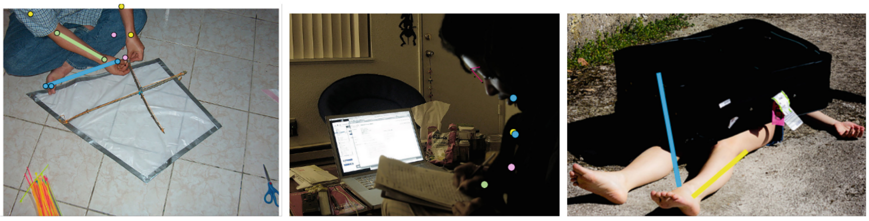


Fig. 6. Failure cases on incomplete poses.

Table 4. Comparison to the SOTA methods on the COCO [8] TEST-DEV dataset. We use COCO TRAIN set as the labeled set and COCO WILD set as the unlabeled set.

Method	NetWork	Input Size	AP	AR
SB [16]	ResNet50	256 × 192	70.2	75.8
HRNet [18]	HRNetW48	384 × 288	75.5	80.5
MSPN [7]	ResNet50	384 × 288	76.1	81.6
DARK [20]	HRNetW48	384 × 288	76.2	81.1
UDP [4]	HRNetW48	384 × 288	76.5	81.6
Dual [17] (+DARK)	HRNetW48	384 × 288	77.2	82.2
SSPCM [5] (+DARK)	HRNetW48	384 × 288	77.5	82.4
Ours (+DARK)	**HRNetW48**	**384 × 288**	**77.9**	**82.7**

Semi-supervised methods are universally acknowledged to grapple with intrinsic challenges stemming from limited annotations and highly imbalanced data distributions. Our approach, while striving to mitigate these issues to a considerable extent, does not entirely eliminate them. Specifically, when the input pose is in a dim environment or when the joints overlap, as shown in the fault situation in Fig. 6, the pseudo-annotations generated by the model may be inaccuracy. This leads to the degradation of the performance of the worst-case projection header, thereby compromising the efficacy of the reverse rebalancing representation. As a result, the inaccuracies in pseudo-annotations can detrimentally affect the accuracy of the worst-case projection header. To address this in our study, we employ a relatively high learning rate with the intent of minimizing the impact of erroneous pseudo-labels on the worst-case projection header. This strategy serves to preserve the operational effectiveness of the reverse rebalancing module. Additionally, we integrate a co-guessing mechanism to mitigate the frequency and severity of errors in the pseudo-annotations. This mechanism facilitates a more

Table 5. Comparisons on the MPII [1] dataset. The training set of the MPII dataset is labeled data, and the AI Challenger [15] dataset is unlabeled data. The size of the input image is 256×256.

Method	Hea	Sho	Elb	Wri	Hip	Kne	Ank	Total
Newell et al.(Stacked Hourglass). [9]	98.2	96.3	91.2	87.1	90.1	87.4	83.6	90.9
Xiao et al.(SB) [16]	98.5	96.6	91.9	87.6	91.1	88.1	84.1	91.5
Ke et al. [6]	98.5	96.8	92.7	88.4	90.6	89.4	86.3	92.1
Sun et al.(HRNet) [18]	98.6	96.9	92.8	89	91.5	89	85.7	92.3
Zhang et al. [21]	98.6	97.0	92.8	88.8	91.7	89.8	86.6	92.5
Xie et al.(Dual) [17]	98.7	97.3	93.7	90.2	92.0	90.3	86.5	93.0
Huang et al.(SSPCM) [5]	98.7	97.5	94.0	90.6	92.5	91.1	87.1	93.3
Ours	**98.7**	**97.6**	**94.2**	**90.9**	**92.7**	**91.5**	**87.6**	**93.6**

collaborative and refined process of annotation correction, thereby enhancing the overall robustness of our semi-supervised framework. Specifically, when the input pose is in a dim environment or when the joints overlap, as shown in the fault situation in Fig. 6, the pseudo-annotations generated by the model may be inaccuracy.

Table 6. Ablation Study on COCO 1K. The Backbone network is ResNet18.

Method	1K	5K	10K
w/o Re-balance	49.5	59.1	60.3
w/o Joint CutPaste	49.8	59.3	60.8
w/o Random feature Mixup	48.7	57.6	60.1
w/o Co-Guessing	50.4	59.8	61.5
Ours	**50.8**	**60.1**	**61.9**

6 Conclusion

In our study, we introduce an innovative solution, Debiasing-Teacher, explicitly engineered to tackle bias-related issues. This solution employs a process known as reverse re-balancing, utilizes effective data augmentation methods like adaptive joint CutPaste, and incorporates a technique known as random feature Mixup. Further, to augment pseudo-label production, we propose the concept of co-guessing. By performing comprehensive experiments, our approach exceeds current methods and achieves state-of-the-art performance.

References

1. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2d human pose estimation: new benchmark and state of the art analysis. In: Proceedings of the IEEE Conference on computer Vision and Pattern Recognition, pp. 3686–3693 (2014)
2. Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C.A.: Mixmatch: a holistic approach to semi-supervised learning. Adv. Neural Inform. Process. Syst. **32** (2019)
3. Chen, B., Jiang, J., Wang, X., Wan, P., Wang, J., Long, M.: Debiased self-training for semi-supervised learning. Adv. Neural. Inf. Process. Syst. **35**, 32424–32437 (2022)
4. Huang, J., Zhu, Z., Guo, F., Huang, G.: The devil is in the details: delving into unbiased data processing for human pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5700–5709 (2020)
5. Huang, L., et al.: Semi-supervised 2d human pose estimation driven by position inconsistency pseudo label correction module. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 693–703 (2023)
6. Herrmann, C.: Robust image stitching with multiple registrations. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11206, pp. 53–69. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01216-8_4
7. Li, W., et al.: Rethinking on multi-stage networks for human pose estimation. arXiv preprint arXiv:1901.00148 (2019)
8. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
9. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 483–499. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_29
10. Radosavovic, I., Dollár, P., Girshick, R., Gkioxari, G., He, K.: Data distillation: Towards omni-supervised learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4119–4128 (2018)
11. Sezgin, M., Sankur, B.l.: Survey over image thresholding techniques and quantitative performance evaluation. J. Electronic Imaging **13**(1), 146–168 (2004)
12. Sohn, K., et al.: Fixmatch: simplifying semi-supervised learning with consistency and confidence. Adv. Neural. Inf. Process. Syst. **33**, 596–608 (2020)
13. Verma, V., et al.: Manifold mixup: better representations by interpolating hidden states. In: International Conference on Machine Learning, pp. 6438–6447. PMLR (2019)
14. Wang, X., Wu, Z., Lian, L., Yu, S.X.: Debiased learning from naturally imbalanced pseudo-labels. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14647–14657 (2022)
15. Wu, J., et al.: Large-scale datasets for going deeper in image understanding. In: 2019 IEEE International Conference on Multimedia and Expo (ICME), pp. 1480–1485. IEEE (2019)
16. Xiao, B., Wu, H., Wei, Y.: Simple baselines for human pose estimation and tracking. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11210, pp. 472–487. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01231-1_29

17. Xie, R., Wang, C., Zeng, W., Wang, Y.: An empirical study of the collapsing problem in semi-supervised 2d human pose estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 11240–11249 (2021)
18. Yu, C., et al.: Lite-hrnet: a lightweight high-resolution network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10440–10450 (2021)
19. Yu, Z., Wang, M., Chen, Y., Favaro, P., Modolo, D.: Denoising and selecting pseudo-heatmaps for semi-supervised human pose estimation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 6280–6289 (2024)
20. Zhang, F., Zhu, X., Dai, H., Ye, M., Zhu, C.: Distribution-aware coordinate representation for human pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7093–7102 (2020)
21. Zhang, H., et al.: Human pose estimation with spatial contextual information. arXiv preprint arXiv:1901.01760 (2019)
22. Zhang, K., et al.: Dnanet: de-normalized attention based multi-resolution network for human pose estimation. arXiv preprint arXiv:1909.05090 (2019)

GraPix: Exploring Graph Modularity Optimization for Unsupervised Pixel Clustering

Sonal Kumar(✉), Arijit Sur, and Rashmi Dutta Baruah

Indian Institute of Technology Guwahati, Guwahati, India
{k.sonal,arijit,r.duttabaruah}@iitg.ac.in

Abstract. A vision transformer learns high-quality patch embeddings during the self-supervised training, which plays a crucial role in many unsupervised downstream tasks like object localization, object detection, and sparse semantic segmentation. Such downstream tasks explore the various properties of the patch affinity graph to achieve state-of-the-art performance in an unsupervised setting. However, the true potential of the patch affinity graph is yet to be harnessed for the dense semantic segmentation task. The existing works show that modularity is an essential property of a graph, which reflects the strength of the existing graph partitions. We argue that joint optimization of feature clustering in patch embedding space and graph modularity in node attribute space leads to a smooth training convergence and achieves better results. In this paper, we propose a novel end-to-end unsupervised learning method called GraPix, which utilizes the hidden property of patch embeddings extracted from a self-supervised vision transformer for the dense semantic segmentation task. The GraPix constructs an affinity graph based on patch similarities in their embedding space. Next, it learns highly discriminative centroid embeddings for dense semantic segmentation with our novel joint feature clustering and graph modularity optimization objective. The experiment results show that GraPix outperforms the state-of-the-art method on the SUIM dataset and achieves the second-best performance on the Cityscapes dataset. Also, we perform a detailed ablation to justify the choice of model components and hyperparameters. The code is available at https://github.com/SonalKumar95/GraPix.

Keywords: Self-supervised learning · Vision transformer · Dense semantic segmentation · Graph Modularity optimization · Pixel clustering

1 Introduction

An image segmentation task assigns a label to each pixel of an image so that pixels with a similar label share some common semantic properties. Semantic

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 193–209, 2025.
https://doi.org/10.1007/978-3-031-78192-6_13

segmentation is the more generalized form of image segmentation in which the labels are consistent throughout images of the whole dataset. It is one of the core computer vision tasks with direct application in various fields like remote sensing, surveillance, underwater exploration [1], autonomous driving [2], medical imaging [3,4], etc. The applicability of the computer vision approach in such applications depends on the accuracy of the semantic segmentation task for domain-specific datasets. Existing state-of-the-art (SOTA) models [5–7] achieve very high-level performance in the presence of supervision or manual annotation. However, the manual pixel-wise annotation of each image from a large dataset is not only time-consuming and cost-ineffective, but it also requires domain expertise [8–10].

In recent years, self-supervised learning has been proposed to alleviate the dependency of the deep learning model's training on manually annotated datasets [11–14]. Yet performing semantic segmentation without manual annotation is much more complex than other unsupervised tasks like image classification, object detection, and object localization. Various unsupervised semantic segmentation (USS) methods [8–10,15–20] have been proposed to achieve better accuracy on top of the existing approaches. An end-to-end learning method [8–10,21] jointly learns pixel embedding and clustering objectives. The two-step learning method [18] utilizes the pre-trained model or visual priors to learn highly discriminative pixel embeddings with a novel loss function in the first step. Later, it utilizes pixel clustering methods to generate segmentation masks. Based on the segmentation mask, there are two major categories of semantic segmentation task, i.e., sparse [15,18] and dense [8–10,18]. A sparse semantic segmentation (SSS) task only labels pixels belonging to the object within a scene. Unlike that, the dense semantic segmentation (DSS) task labels each pixel within a scene, ranging from stuff to things. However, in comparison to sparse semantic segmentation, achieving good performance for dense semantic segmentation without manual annotations is challenging. Only a few existing methods [8–10] are capable of performing unsupervised DSS tasks efficiently.

The introduction of self-supervised learning in the vision transformer (ViT) provides a new direction for applying representation learning to perform complex unsupervised downstream tasks like semantic segmentation, object localization [22–24], and object detection [25]. In contrast with CNN architecture, a ViT learns high-quality patch embeddings with the self-attention mechanism [26]. Some of the existing approaches [27–30] utilize these patch embeddings as an alternative to the corresponding input images to perform unsupervised downstream tasks. Initially, the Lost framework [22] introduced the potential of patch embedding extracted from a self-supervised ViT for the single object localization task. It extracts the patch embedding for an image and computes a patch affinity graph based on the pair-wise similarity of the patch embeddings. Later, it applies the graph-based method to localize the object within an object-centric image. Following this, various SOTA approaches for object localization [23,24] with a novel graph-based method are proposed to achieve superior results. Motivated by this, some of the unsupervised semantic segmentation (USS) tasks are also proposed to utilize patch affinity graphs and outperform the existing methods for the sparse semantic segmentation (SSS) task [10]. However, unlike sparse semantic segmentation (SSS), the true potential of patch embeddings is not harnessed for the dense semantic segmentation task (DSS) in an unsupervised setting.

In contrast with the existing approach [8–10,15–18,20], this paper explores the potential of patch embedding extracted from a self-supervised ViT for end-to-end unsupervised learning for DSS tasks. We propose a novel end-to-end unsupervised learning method, GraPix, which utilizes the patch embeddings from self-supervised ViT with a novel loss function, $\mathcal{L}_{UnSeg}$, for unsupervised DSS task. The $\mathcal{L}_{UnSeg}$ loss is a combination of feature clustering and graph modularity objectives. Modularity is an important property of a graph [31,32], which reflects the strength of the existing partition in a graph. A graph with a high modularity value has dense intra-partition connections and sparse inter-partition connections. For example, the paper [32] proposed the Louvain clustering algorithm to harness the modularity property of a graph for community detection in a large network. Later, a spectral approximation for modularity computation is proposed in paper [33] for graph clustering with Graph Neural Network (GNN). The major contributions of our proposed methods are as follows:

1. A novel end-to-end unsupervised learning method for dense semantic segmentation task, named GraPix, has been proposed, which trains a projection head and cluster embedding over a frozen self-supervised ViT backbone.
2. A novel loss function, $\mathcal{L}_{UnSeg}$ loss, has been introduced to utilize feature clustering objective in patch embedding space and graph modularity objective in node attribute space.

2 Related Work

The semantic segmentation task without the manual annotation can be seen as a pixel clustering problem. Various SOTA methods for sparse [15,18,34] and dense [8–10,20] semantic segmentation are proposed with this idea. The IIC [8] framework is proposed as a preliminary effort for the unsupervised DSS task. It utilizes a CNN backbone and maximizes the mutual information between neighbor patches to learn quality pixel embeddings. Several other end-to-end methods with CNN backbone are proposed with the idea of maximizing agreement on pixel level [21] or segment level [9] with a novel objective function. Alternatively, some of the bottom-up approaches [2,15,16] are also proposed to utilize the visual priors or patch embeddings to decompose the scene into visual groups and utilize mutual information or contrastive objective functions to learn meaningful pixel embeddings.

Also, a few works [10,18,35] utilize feature representations or patch embeddings extracted from self-supervised ViT to perform the unsupervised semantic segmentation task. The MaskDistill method [18] leverages the idea of the LOST framework [22] to generate a single object mask for an image dataset and use it as a pseudo-ground truth to train a segmentation model from scratch. Due to the limitation of the LOST framework of localizing only a single object in a scene, the application of MaskDistill is limited to the unsupervised SSS task. The paper [20] proposes a data-driven bottom-up approach, DatUS, to leverage the Louvain clustering algorithm and patch embeddings for generating the pseudo-segmentation masks of scene-centric images. Also, The STEGO method [10] utilizes the features correspondence property of feature representation of images extracted from self-supervised ViT to train a segmentation head for the unsupervised DSS task. Unlike that, the DeepCut framework [35] utilizes the patch embedding of an input image extracted from self-supervised ViT to construct a patch affinity graph. Later, it utilizes spectral approximation of graph clustering algorithms like correlation and NCut clustering to train a Graph Neural Network (GNN) [36]. The GNN model of the DeepCut method needs to be trained separately for each scene to avoid over-fitting. Hence, the application of DeepCut is limited to single object segmentation (foreground and background separation), semantic part segmentation (image segmentation), and object localization. Like MaskDistill, STEGO, and DeepCut methods, our proposed method, GraPix, also utilizes the patch embedding extracted from a vision transformer. To overcome their limitation, we propose an *Unsupervised Segmentation Loss* to optimize the feature clustering and modularity objective jointly. It allows us to learn a high-quality centroid embedding for unsupervised dense semantic segmentation with the proposed architecture in Fig. 1.

3 Proposed Method

This paper proposes a novel end-to-end unsupervised learning approach for the DSS task, termed GraPix. In contrast with the existing approach, GraPix learns highly discriminative centroid embeddings for the unsupervised DSS task with a joint feature clustering and graph modularity optimization objective. Figure 1 presents a high-level overview of the GraPix architecture. The major components of the proposed architecture are a Self-supervised Representation Learning (SRL) Setup, Trainable Cluster Centroids ($\mathcal{C}_{\mathcal{Q}}$), and Unsupervised Segmentation Loss ($\mathcal{L}_{UnSeg}$).

Also, we introduce additional unsupervised training steps, i.e., Self-label and KNN training [37], to boost the performance of the proposed method.

3.1 Self-supervised Representation Learning (SRL) Setup

The *SRL* setup has an image augmentation strategy $\mathcal{T}$, a pre-trained ViT encoder $\mathcal{E}_{\mathcal{Q}}$, and a trainable projection head $\mathcal{P}_{\mathcal{Q}}$. The image augmentation strategy generates five crops of each training image, followed by the photometric and geometric transformation. The five-cropping of the dataset is performed before training. It expands the volume of the training dataset by the multiple of five, i.e., generates 5N views from N training images. The self-supervised ViT encoder $\mathcal{E}_{\mathcal{Q}}$ acts as a feature extractor and is frozen during the training steps. The $\mathcal{E}_{\mathcal{Q}}$ decomposes each view $\in \mathbf{R}^{H \times W}$ from i_{th} batch in a $((H/P) * (W/P))$ number of patches before processing, where the dimension of each patch is $P \times P$. It

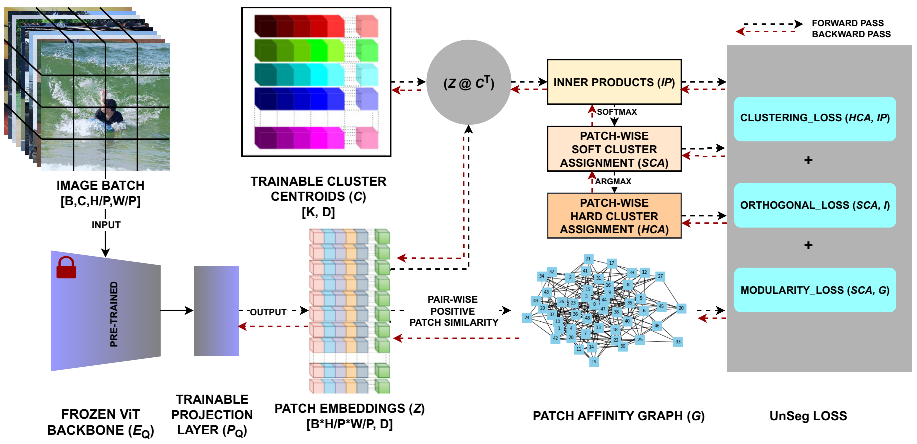


IMAGE BATCH [B,C,H/P,W/P]
INPUT
PRE-TRAINED
FROZEN ViT BACKBONE (E_Q)
TRAINABLE PROJECTION LAYER (P_Q)
OUTPUT
TRAINABLE CLUSTER CENTROIDS (C) [K, D]
PATCH EMBEDDINGS (Z) [B*H/P*W/P, D]
(Z @ C^T)
PAIR-WISE POSITIVE PATCH SIMILARITY
INNER PRODUCTS (IP)
SOFTMAX
PATCH-WISE SOFT CLUSTER ASSIGNMENT (SCA)
ARGMAX
PATCH-WISE HARD CLUSTER ASSIGNMENT (HCA)
PATCH AFFINITY GRAPH (G)
FORWARD PASS
BACKWARD PASS
CLUSTERING_LOSS (HCA, IP)
+
ORTHOGONAL_LOSS (SCA, I)
+
MODULARITY_LOSS (SCA, G)
UnSeg LOSS

Fig. 1. A high-level overview of the GraPix architecture.

extracts one CLS token and $((H/P) * (W/P))$ number of patch embedding for each view. Next, the trainable projection head $\mathcal{P_Q}$ (a linear/non-linear MLP) maps each patch embeddings of i_{th} batch to the compatible dimension (Dim) to apply the proposed Unsupervised Segmentation Loss. Hence, The $\mathcal{P_Q}$ maps $(B * (H/P) * (W/P))$ number of patch embedding from i_{th} batch to projected patch embedding matrix $\mathcal{Z}_i \in \mathbf{R}^{(B*(H/P)*(W/P)\times Dim)}$. Here, B indicates the batch size, i.e., the number of images in i_{th} batch.

The projected patch embedding matrix, $\mathcal{Z}_i$, is further processed to construct the patch affinity graph, i.e., $\mathcal{G}_i$. The nodes of the patch affinity graph represent patches of the i_{th} batch. A pair of nodes in $\mathcal{G}$ share an edge if the corresponding patch embeddings have a positive pair-wise similarity value.

3.2 Trainable Cluster Centroids ($\mathcal{C_Q}$)

The Trainable Cluster Centroids, $\mathcal{C_Q} \in \mathbf{R}^{K\times Dim}$, is a randomly initialized embedding layer. It consists of a K number of embeddings, each representing a centroid feature vector of a category/class from the segmentation dataset. The dimension of a centroid feature vector is the same as a patch embedding form $\mathcal{Z}_i$, i.e., Dim.

$$\mathcal{IP}_i = \mathcal{Z}_i @ \mathcal{C_Q}^T \tag{1}$$

$$\mathcal{SCA}_i = softmax(\mathcal{IP}_i) \tag{2}$$

$$\mathcal{HCA}_i = argmax(\mathcal{SCA}_i) \tag{3}$$

We perform dot product between $\mathcal{Z}_i$ and $\mathcal{C_Q}$ to compute the Inner Product, $\mathcal{IP}_i \in \mathcal{R}^{(B*(H/P)*(W/P)\times K)}$ for each patch of i_{th} batch using Eq. 1. A softmax operation is performed over the $\mathcal{IP}_i$ to compute the Soft Cluster Assignment of each patch of i_{th} batch, i.e., $\mathcal{SCA}_i$ using Eq. 2. Further, we compute patch-wise Hard Cluster Assignment, i.e., $\mathcal{HCA}_i$ by assigning cluster id with maximum probability in $\mathcal{SCA}_i$ to each patch of i_{th} batch using Eq. 3. The $\mathcal{IP}_i$, $\mathcal{SCA}_i$, and $\mathcal{HCA}_i$ are utilized by $\mathcal{L}_{\text{UnSeg}}$ loss in the next section.

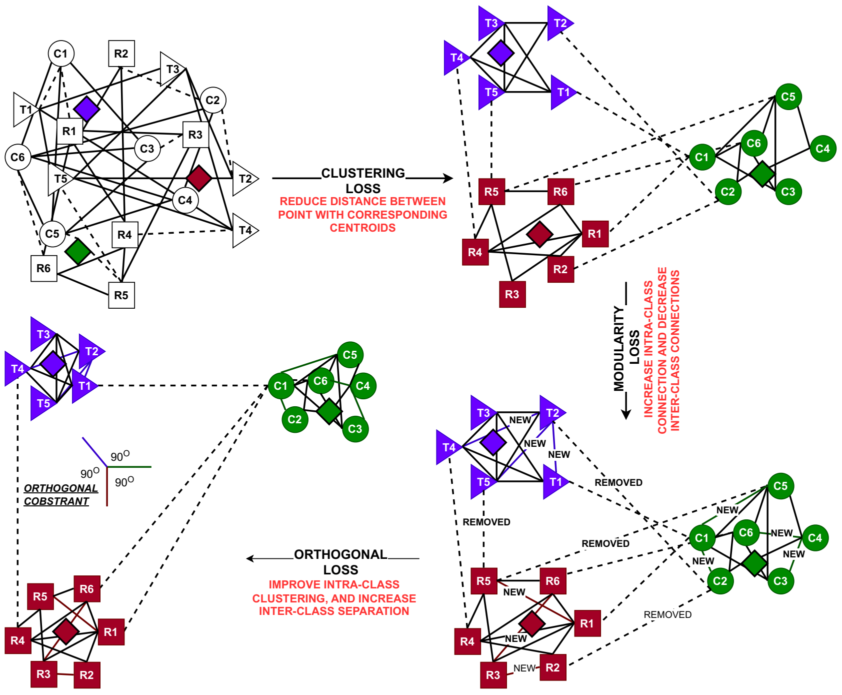

CLUSTERING LOSS
REDUCE DISTANCE BETWEEN POINT WITH CORRESPONDING CENTROIDS
MODULARITY LOSS
INCREASE INTRA-CLASS CONNECTION AND DECREASE INTER-CLASS CONNECTIONS
ORTHOGONAL LOSS
IMPROVE INTRA-CLASS CLUSTERING, AND INCREASE INTER-CLASS SEPARATION
ORTHOGONAL COBSTRANT
90°
NEW
REMOVED

Fig. 2. The individual contribution of clustering, modularity, and orthogonal loss for unsupervised training of GraPix architecture. The squire, circle, and triangle represent the patch embeddings of three different visual groups. The diamonds with blue, green, and red colors represent centroid embeddings of three different clusters. The solid lines represent intra-class connections, and the dotted lines represent inter-class connections. The lines with NEW and REMOVED tags present newly added and discarded connections, respectively. (Color figure online)

3.3 Unsupervised Segmentation Loss ($\mathcal{L}_{UnSeg}$)

As shown in Eq. 4, the $\mathcal{L}_{UnSeg}$ Loss is a combination of modularity $\mathcal{L}_{mod}$, clustering $\mathcal{L}_{clust}$, and orthogonal $\mathcal{L}_{ortho}$ loss. Figure 2 shows the individual contribution of $\mathcal{L}_{clust}$, $\mathcal{L}_{mod}$, and $\mathcal{L}_{ortho}$ for unsupervised training the proposed GraPix architecture. The $\mathcal{L}_{UnSeg}$ aims to optimize the feature clustering and graph modularity objectives jointly with the GraPix architecture.

$$\mathcal{L}_{\text{UnSeg}} = \mathcal{L}_{\text{mod}} + \mathcal{L}_{\text{clust}} + \mathcal{L}_{\text{ortho}} \tag{4}$$

The $\mathcal{L}_{mod}$ loss computes the modularity of patch affinity graph $\mathcal{G}_i$ of i_{th} batch using Eq. 5. The graph modularity [31,32] represents the strength of the partitions in the $\mathcal{G}_i$. Maximizing $\mathcal{L}_{mod}$ loss increases intra-partition and decreases inter-partition connections within the patch affinity graph $\mathcal{G}_i$.

$$\mathcal{L}_{\text{mod}} = (-1) * (1/2 * \mu) * trace((\mathcal{SCA}_i^T @ \mathcal{Q}_i) @ \mathcal{SCA}_i) \tag{5}$$

The Eq. 5 presents a spectral approximation for graph modularity [33] computation. Here, $\mathcal{SCA}_i$, and $\mathcal{HCA}_i$ are hard and soft cluster assignments of patches from i_{th} batch, Adj_i is the corresponding adjacency matrix of patch affinity graph $\mathcal{G}_i$ (refer Eq. 8), $\mathcal{S}_i^+$ is positive pair-wise similarity matrix of patches from i_{th} batch (refer Eq. 9), I denotes identity matrix, P denotes the set of patches from i_{th} batch, $\mathcal{D}_i$ is node degree matrix, λ is hyper-parameter (resolution coefficient) to control the size of output partition of the graph, μ is the total degree of patch affinity graph $\mathcal{G}$, and *trace( )* denotes the trace of a matrix' Also, in the given equations, $< \cdot >$ denotes dot product, $*$ denotes scalar multiplication, @ denotes matrix multiplication, and superscript T denotes the transpose of the corresponding matrix.

$$\mathcal{Q}_i = \mathcal{S}_i^+ - (\lambda * (D_i @ D_i^T) * (1/2 * \mu_i) \tag{6}$$

$$\mathcal{S}_i = (\mathcal{Z}_i @ \mathcal{Z}_i^T) - < (\mathcal{Z}_i @ \mathcal{Z}_i^T) \cdot I_{|P|\times|P|} > \tag{7}$$

$$Adj_i = \begin{cases} adj(r,s) = 0, & \mathcal{S}_i \leq 0 \\ adj(r,s) = 1, & \mathcal{S}_i > 0 \end{cases} \tag{8}$$

$$\mathcal{S}_i^+ = < \mathcal{S}_i \cdot Adj_i > \tag{9}$$

$$\mathcal{D}_i = \sum_{r=1} Adj_i(r,s) \tag{10}$$

$$\mu_i = (1/2) * \sum_{r=1} \sum_{s=1} Adj_i(r,s) \tag{11}$$

We observe that only maximizing the modularity of $\mathcal{G}_i$ leads to an unstable training sequence and does not converge to high-quality discriminative centroid embeddings $\mathcal{C}_i$. Hence, we introduce clustering loss $\mathcal{L}_{clust}$ to stabilize the modularity optimization. The $\mathcal{L}_{clust}$ computes the negative average distance between the points and their nearest centroid in patch embedding space using 12. Maximizing the $\mathcal{L}_{clust}$ loss brings patches sharing the same partition close in patch embedding space.

$$\mathcal{L}_{\text{clust}} = (-1) * < \text{OneHotEncode}(\mathcal{HCA}_i) \cdot \mathcal{IP}_i > \tag{12}$$

Additionally, we introduce the orthogonal loss [37], $\mathcal{L}_{ortho}$, in the Unsupervised Segmentation Loss. The orthogonal loss enforces the orthogonality constraints such that the patch embeddings from different categories become orthogonal in feature space. It improves the intra-class clustering and increases inter-class separation in patch embedding space using Eq. 13. The $\mathcal{L}_{\text{ortho}}$ loss computes cross entropy between the graph partition similarity matrix, i.e., $(\mathcal{SCA}_i^T @ \mathcal{SCA}_i)$ and the identity matrix $\mathcal{I} \in \mathcal{R}^{K \times K}$, where K is the total number of partition or total categories in the dataset.

$$\mathcal{L}_{\text{ortho}} = \text{Cross-entropy}((\mathcal{SCA}_i^T @ \mathcal{SCA}_i), I_{K \times K}) \tag{13}$$

Algorithm 1 summarises the training procedure of the proposed method, i.e., GraPix.

Algorithm 1 GraPix

```
1: Input D: dataset, $E_Q$: pre-trained ViT encoder, $\mathcal{P}_Q$: projection layer, $\mathcal{C}$: centroid
   embeddings, totalItter: number of iteration, T: an augmentation strategy
2: Freeze $\mathcal{E}_Q$
3: Initialize $\mathcal{P}_Q$ and $\mathcal{C}$ with random weights.
4: for $e = 0$ to totalItter do
5:     Sample mini-batch of size N from D
6:     Sample augmented views of each image in mini-batch (V) with T
7:     Extract Patch embedding, i.e., $\mathcal{Z} \leftarrow \mathcal{P}_Q(E_Q(V))$
8:     Compute inner products ($\mathcal{IP}$) using Eq. 1
9:     Compute patch-wise soft cluster assignment ($\mathcal{SCA}$) using Eq. 2
10:    Compute patch-wise hard cluster assignment ($\mathcal{HCA}$) using Eq. 3
11:    Compute $\mathcal{L}_{mod}$, $\mathcal{L}_{clust}$, and $\mathcal{L}_{ortho}$ using Eqs. 5, 12, and 13, respectively.
12:    Compute total loss, i.e., $\mathcal{L}_{UnSeg} = \mathcal{L}_{mod} + \mathcal{L}_{clust} + \mathcal{L}_{ortho}$
13:    Backprop $\mathcal{L}_{UnSeg}$
14: end for
15: Output $\mathcal{P}_Q$, $\mathcal{C}$
```

3.4 Additional Unsupervised Training Steps

In the previous section, the GraPix method learns discriminative centroid embeddings. The learned centroid embedding can be further utilized to generate class-level quality supervision from the dataset to improve the performance of the proposed GraPix architecture. We introduce two additional unsupervised training steps: Self-label and KNN training. They are performed with the pre-trained GraPix architecture and corresponding loss function given in Eq. 14, 15. The Self-label and KNN training are motivated by the SCAN framework [37] for the unsupervised image clustering task.

Self-label Training: It utilizes the confident patch samples of each batch for unsupervised fine-tuning of the pre-trained GraPix architecture with a masked cross-entropy loss, $\mathcal{L}_{SL}$. A patch from a i_{th} batch is confident if the corresponding softmax probability for the assigned cluster exceeds some pre-defined threshold, δ. The masked cross-entropy loss, $\mathcal{L}_{SL}$, is defined in Eq. 14. It computes the cross entropy loss between the hard cluster assignment of confident patches ($\mathcal{HCA}_i$) and its augmented view ($\mathcal{HCA}_i^{Aug}$).

$$\mathcal{L}_{SL} = \text{Cross-entropy}(\mathcal{HCA}_i^{Aug}, \mathcal{HCA}_i) \tag{14}$$

KNN Training: The KNN training maximizes the similarity between the soft cluster assignment $\mathcal{SCA}$ of each patch and its k-nearest neighbors ($\mathcal{N}_p$) from i_{th} batch. As defined in Eq. 15, The KNN Loss, $\mathcal{L}_{KNN}$, has two parts. The right part of the equation computes entropy loss, which ensures uniform prediction across categories in the dataset. The left part computes the average dot product between $\mathcal{SCA}$ of each patch with its k-nearest neighbors.

$$\mathcal{L}_{KNN} = (-1/|P|) \sum_{p \in P} \sum_{n \in \mathcal{N}_p} \log\big(< \mathcal{SCA}_i(p) \cdot \mathcal{SCA}_i(n) > \big) + \alpha * \sum_{k \in K} q_i^k \log q_i^k \tag{15}$$

$$\text{Here,} \quad q_i^k = (1/|P|) \sum_{p \in P} \mathcal{SCA}_i^k(p) \tag{16}$$

Here, P denotes a set of patches from $i_t h$ batch, $\mathcal{N}_p$ denotes k-nearest neighbors of p_{th} patch, $\mathcal{SCA}_i(p)$ denotes the soft cluster assignment of p_{th} patch, α is an entropy term (a scalar value), $\mathcal{SCA}_i^k(p)$ id probability of p_{th} patch to be assigned cluster ID k.

4 Experiments

This section presents the results of our empirical validation conducted to assess the efficacy of GraPix. We also conducted an ablation study to justify the design choice for the GraPix architecture and method. Our experiments were implemented using the PyTorch framework on an NVIDIA DGX Station A100 GPU with 80G memory.

4.1 Experimental Settings

We evaluate the performance of GraPix on two benchmark datasets: SUIM [1] and Cityscapes [38]. SUIM and Cityscapes are well-known scene-centric datasets for dense semantic segmentation tasks. SUIM is a collection of 1635 color images (1535 training and 110 testing images) collected from the underwater environment. The color images are distributed across eight visual groups, which are further grouped into six-course categories [1]: human divers, Robots (AUVs/ROVs/Instruments), Reefs/Invertebrates, Fish/Vertebrates, Wrecks/Ruins, Aquatic-plants/Sea-grass/Sea-floor/Rock/Background. Cityscapes is a collection of 24,500 street color images (approximately 21,000 training and 3500 testing images) from 50 different cities. It recognizes objects from thirty visual groups, which are further categorized into twenty-seven-course categories [9].

We employ a frozen ViT-B/8 vanilla encoder [26], which is trained with the ImageNet-1k dataset using the DINO framework [29]. A linear/non-linear projection head is trained over the ViT backbone to undertake the representation learning task, and an embedding layer is deployed to learn the centroids for a predefined number of classes/clusters (K). The Projection Head $\mathcal{P}_Q$ yielded a final output of 70-dimensional vectors, which is similar to the size of centroid embedding vector $c_q \in \mathcal{C}_Q$. We adopted specific hyper-parameters for both datasets, including a resolution coefficient λ of 1.0/0.05, ADAM optimizer with a learning rate of $5e-4/1e-4$ for MLP $\mathcal{P}_Q$ and $5e-3/1e-3$ for embedding layer $\mathcal{C}_Q$. We train the STEGO [10] model from scratch with the ViT-b/8 backbone to compute the results on the SUIM dataset.

Table 1. Results over SUIM and Cityscapes datasets for unsupervised dense semantic segmentation task. The highest value in each column is in bold format, and the second-highest value is underlined.

Dataset	Method	MiOU	PAcc.	MF1
SUIM	STEGO [10]	23.88	53.08	32.82
	DatUS [20]	28.48	64.67	39.52
	GraPix	28.98	64.06	41.00
	GraPix+SL	<u>30.02</u>	64.43	<u>42.67</u>
	GraPix+KNN	29.97	**65.48**	42.18
	GraPix+KNN+SL	**30.78**	<u>65.40</u>	**43.51**
Cityscapes	IIC [10]	6.4	47.9	–
	MDC [10]	7.1	40.7	–
	PiCIE [10]	12.3	<u>65.5</u>	–
	STEGO [10]	**21.0**	**73.2**	**27.70**
	GraPix	14.33	64.89	19.52
	GraPix+SL	14.43	51.91	19.54
	GraPix+KNN	<u>14.54</u>	51.54	<u>19.69</u>
	GraPix+KNN+SL	14.48	51.14	19.63

Like prior work [9,10], we resize each validation image to 320 pixels along both dimensions, i.e., height and width, followed by 320×320 center crop. In an unsupervised setting, we utilize Hungarian matching [39] to map the K number of predicted clusters to the C number of ground truth classes. Unless stated, we consider $K = C$ for the evaluation process.

4.2 Comparison with State-of-the-Art Methods

Quantitative results: Table 1 presents the comparative analysis of our proposed method, GraPix, and state-of-the-art methods for the unsupervised dense

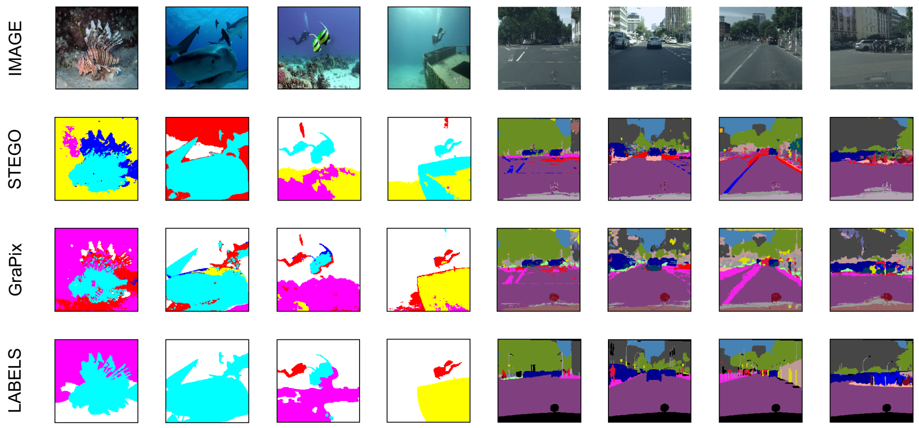


Fig. 3. A visual comparison of mask generated from STEGO, GraPix, and ground truth mask for a few sample images from the validation set.

semantic segmentation task, i.e., STEGO (ViT-B/8). The SL and KNN denote the self-label and KNN training steps, respectively. Notably, Table 1 shows that GraPix outperforms the STEGO on the SUIM and achieves the second-best performance on the Cityscapes dataset. The additional unsupervised training steps further improve the performance of GraPix on both datasets. The GraPix with both additional unsupervised training steps, i.e., GraPix+KNN+SL, achieves an improvement of +6.9, +12.32, and +10.69 on MiOU, Pixel Accuracy, and Mean F1 Score metrics, respectively, on SUIM dataset.

Qualitative Results: Fig. 3 presents a visual comparison between the segmentation masks generated using STEGO and GraPix with Ground truth masks for sample images taken from the validation set. It is evident that our proposed method learns to generate highly informative segmentation for images from different domains without using any manual annotations. Also, it can be observed in Fig. 3 that the unsupervised methods for semantic segmentation are prone to over-segmentation of scenes, i.e., identifying more objects compared to the ground truth masks. One of the reasons for over-segmentation could be that the self-supervised training generates supervision from the unlabeled dataset. Hence, for more complex datasets like cityscapes, where several objects with varying shapes and sizes are present, over-segment may lead to performance degradation.

4.3 Ablation Study

Next, we perform a detailed study to understand the effect of augmentation strategy, projection head, and hyper-parameters on the performance of GraPix, as well as additional unsupervised training steps.

Table 2. Ablation on projection layer (Proj.), crop type (Crop), and resolution coefficient (λ) for GraPix training with batch size (BS) equal to eight.

Proj.	Crop	λ	SUIM			Cityscapes		
			MiOU	PAcc.	MF1	MiOU	PAcc.	MF1
LIN	None	0.05	**28.98**	**64.06**	**41.00**	14.31	**64.89**	18.98
LIN	None	1.0	27.10	61.90	38.42	13.61	44.86	18.79
LIN	Five	0.05	24.97	57.81	34.56	13.88	41.73	**19.52**
LIN	Five	1.0	22.57	52.90	34.54	12.53	52.36	17.54
Non-LIN	None	0.05	25.74	61.61	35.84	12.92	44.32	17.79
Non-LIN	None	1.0	24.06	51.02	35.18	12.34	37.23	17.87
Non-LIN	Five	0.05	25.17	51.99	36.41	14.26	54.89	19.16
Non-LIN	Five	1.0	25.00	56.64	35.99	**14.33**	51.70	19.45

4.4 GraPix Training

Based on the findings presented in Table 2, a linear projection head (LIN) demonstrates effectiveness on small to moderately-sized datasets such as SUIM, while non-linear projection heads are more appropriate for larger and more complex datasets like Cityscapes. Also, the images from the Cityscapes data consist of many small to large size objects in a single scene. To reduce the complexity of such a complex scene, we apply a five-cropping strategy, which decomposes an image into five sub-images of similar dimensions [10]. Table 2 shows that the training with a five-cropped dataset dominates the performance in all metrics for complex datasets. Also, the resolution coefficient λ is used to control the size of the graph partitions. The results suggest that the smaller value of the resolution coefficient λ is more suitable for the dense semantic segmentation task.

Table 3 illustrates the influence of batch size on GraPix performance. The number of nodes in the patch affinity graph of the GraPix method is directly proportional to the batch size. The results from Table 3 suggest that the size of the intermediate patch affinity graph may have some influence on the overall training. For both datasets, a batch size of eight is found to be suitable.

Table 3. Ablation on different batch sizes (BS) with dataset-wise best-performing version of GraPix from Table 2.

BS	SUIM			Cityscapes		
	MiOU	PAcc.	MF1	MiOU	PAcc.	MF1
4	21.66	48.99	32.70	14.18	55.88	19.36
8	**28.98**	**64.06**	**41.00**	**14.31**	**64.89**	**18.98**
16	25.26	52.24	36.39	12.67	48.17	17.61
32	22.14	54.65	32.34	13.93	43.26	19.25

4.5 Additional Unsupervised Training Steps

To study the effectiveness of additional unsupervised training steps, we performed individual and combined training of self-label and KNN training after the completion of GraPix with its best-performing version from Table 2.

Table 4. Ablation on different crop types for Additional Training Steps (ATS), i.e., Self-Label and KNN training step with batch size equal to eight.

ATS	Crop	SUIM			Cityscapes		
		MiOU	PAcc.	MF1	MiOU	PAcc.	MF1
Self-label	None	29.92	64.06	42.48	14.43	**51.91**	19.54
	Five	**30.02**	**64.43**	**42.67**	**14.44**	51.80	**19.56**
KNN	None	**29.97**	**65.48**	**42.18**	14.35	50.61	19.52
	Five	29.75	64.78	42.06	**14.42**	**50.78**	**19.60**

Table 5. Ablation on top-k neighbors for KNN training with batch size equal to eight.

Top-K	SUIM			Cityscapes		
	MiOU	PAcc.	MF1	MiOU	PAcc.	MF1
10	29.00	64.14	41.01	14.36	51.26	19.43
15	29.43	64.22	41.70	14.36	51.26	19.43
20	**29.97**	**65.48**	**42.18**	14.42	50.78	19.60
25	29.09	64.55	42.05	**14.54**	**51.54**	**19.69**

It can be observed from Table 4 that the Self-label training favors the five-cropping for both datasets. Unlike that, the KNN training favors the five-cropping with only the complex dataset, i.e., Cityscapes. Also, as shown in Table 5, the choice of k number of neighbors does not have a noticeable impact on the performance of KNN training for both datasets. In Table 6, we present the performance of different combinations of self-label and KNN training over GraPix training. The additional training steps are performed with and without a five-cropped dataset. It is evident from the table that performing KNN training followed by Self-label training has the best performance as compared to all combinations. Since the additional training steps are performed over the unsupervised pre-trained model. Their performance depends upon the strength of supervision obtained from the unsupervised feature representations of the training dataset. Hence, the improvement gain may or may not be significant depending upon the complexity and the nature of the dataset used for additional training.

Table 6. Ablation of the training sequence for additional unsupervised training steps, i.e., Self-label and KNN training with different crop types of training data.

Method	SUIM			Cityscapes		
	MiOU	PAcc.	MF1	MiOU	PAcc.	MF1
GraPix+SL (None)+KNN(None)	30.38	64.43	43.03	14.47	51.13	**19.68**
GraPix+SL (None)+KNN(Five)	30.31	64.18	42.98	14.46	51.29	19.66
GraPix+SL (Five)+KNN(Five)	30.54	64.33	43.38	14.47	**51.49**	19.65
GraPix+KNN (None)+SL(None)	29.43	64.15	41.73	14.44	50.89	19.61
GraPix+KNN (None)+SL(Five)	**30.78**	**65.40**	**43.51**	14.43	51.07	19.61
GraPix+KNN (Five)+SL(Five)	30.33	64.86	39.21	**14.48**	51.14	19.63

5 Conclusion

This paper proposes a novel end-to-end unsupervised learning method, GraPix, for the dense semantic segmentation of scene-centric images from unlabeled datasets. The GraPix learns highly discriminative centroid embedding with joint optimization of feature clustering in patch embedding space and graph modularity in node attribute space. Also, we introduce additional unsupervised training steps to boost the performance of the proposed architecture after GraPix training. We perform experiments on multiple datasets from different domains to justify the generality of the proposed method. The GraPix outperforms the state-of-the-art method on the SUIM dataset and achieves the second-best performance on the Cityscapes dataset. Finally, we perform a detailed ablation to understand the influence of different design choices for proposed architecture and hyper-parameters on overall performance.

Also, It's crucial to emphasize that the performance of our proposed method can be improved with the advanced methodology in the future. In the unsupervised domain, it is difficult to match the self-supervised model outcome for the computer vision task with the human-level annotation. Further research is required to bridge the gap between supervised and unsupervised methods.

References

1. M. J. Islam, C. Edge, Y. Xiao, P. Luo, M. Mehtaz, C. Morse, S. S. Enan, and J. Sattar, "Semantic segmentation of underwater imagery: Dataset and benchmark," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1769–1776
2. A. Vobecky, D. Hurych, O. Siméoni, S. Gidaris, A. Bursuc, P. Pérez, and J. Sivic, "Drive&segment: Unsupervised semantic segmentation of urban scenes via cross-modal distillation," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVIII*. Springer, 2022, pp. 478–495

3. P. Wang, S. Liu, and J. Peng, "Ast-net: Lightweight hybrid transformer for multimodal brain tumor segmentation," in *2022 26th International Conference on Pattern Recognition (ICPR)*. IEEE, 2022, pp. 4623–4629
4. A. Mahbod, G. Schaefer, R. Ecker, and I. Ellinger, "Automatic foot ulcer segmentation using an ensemble of convolutional neural networks," in *2022 26th International Conference on Pattern Recognition (ICPR)*. IEEE, 2022, pp. 4358–4364
5. L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818
6. Guo, M.-H., Lu, C.-Z., Hou, Q., Liu, Z., Cheng, M.-M., Hu, S.-M.: Segnext: Rethinking convolutional attention design for semantic segmentation. Adv. Neural. Inf. Process. Syst. **35**, 1140–1156 (2022)
7. A. Daydar, A. Pramanick, A. Sur, and S. Kanagaraj, "Segmentation of tibiofemoral joint tissues from knee mri using mtra-unet and incorporating shape information: Data from the osteoarthritis initiative," *arXiv preprint* arXiv:2401.12932, 2024
8. X. Ji, J. F. Henriques, and A. Vedaldi, "Invariant information clustering for unsupervised image classification and segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9865–9874
9. J. H. Cho, U. Mall, K. Bala, and B. Hariharan, "Picie: Unsupervised semantic segmentation using invariance and equivariance in clustering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 794–16 804
10. M. Hamilton, Z. Zhang, B. Hariharan, N. Snavely, and W. T. Freeman, "Unsupervised semantic segmentation by distilling feature correspondences," *arXiv preprint* arXiv:2203.08414, 2022
11. S. Roychowdhury, S. A. Sontakke, L. Itti, M. Sarkar, M. Aggarwal, P. Badjatiya, N. Puri, and B. Krishnamurthy, "Sherlock: Self-supervised hierarchical event representation learning," in *2022 26th International Conference on Pattern Recognition (ICPR)*. IEEE, 2022, pp. 2672–2678
12. K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9729–9738
13. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. Adv. Neural. Inf. Process. Syst. **33**, 9912–9924 (2020)
14. T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607
15. W. Van Gansbeke, S. Vandenhende, S. Georgoulis, and L. Van Gool, "Unsupervised semantic segmentation by contrasting object mask proposals," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 052–10 062
16. Mirsadeghi, S.E., Royat, A., Rezatofighi, H.: Unsupervised image segmentation by mutual information maximization and adversarial regularization. IEEE Robotics and Automation Letters **6**(4), 6931–6938 (2021)
17. Y. Wang, W. Zhuo, Y. Li, Z. Wang, Q. Ju, and W. Zhu, "Fully self-supervised learning for semantic segmentation," *arXiv preprint* arXiv:2202.11981, 2022
18. W. Van Gansbeke, S. Vandenhende, and L. Van Gool, "Discovering object masks with transformers for unsupervised semantic segmentation," *arXiv preprint* arXiv:2206.06363, 2022

19. L. Pauletto, M.-R. Amini, and N. Winckler, "Se2nas: Self-semi-supervised architecture optimization for semantic segmentation," in *2022 26th International Conference on Pattern Recognition (ICPR)*. IEEE, 2022, pp. 54–60
20. S. Kumar, A. Sur, and R. D. Baruah, "Datus: Data-driven unsupervised semantic segmentation with pre-trained self-supervised vision transformer," *IEEE Transactions on Cognitive and Developmental Systems*, pp. 1–14, 2024
21. J.-J. Hwang, S. X. Yu, J. Shi, M. D. Collins, T.-J. Yang, X. Zhang, and L.-C. Chen, "Segsort: Segmentation by discriminative sorting of segments," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7334–7344
22. O. Siméoni, G. Puy, H. V. Vo, S. Roburin, S. Gidaris, A. Bursuc, P. Pérez, R. Marlet, and J. Ponce, "Localizing objects with self-supervised transformers and no labels," November 2021
23. O. Siméoni, C. Sekkat, G. Puy, A. Vobeckỳ, É. Zablocki, and P. Pérez, "Unsupervised object localization: Observing the background to discover objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3176–3186
24. Y. Wang, X. Shen, S. X. Hu, Y. Yuan, J. L. Crowley, and D. Vaufreydaz, "Self-supervised transformers for unsupervised object discovery using normalized cut," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 543–14 553
25. A. Bar, X. Wang, V. Kantorov, C. J. Reed, R. Herzig, G. Chechik, A. Rohrbach, T. Darrell, and A. Globerson, "Detreg: Unsupervised pretraining with region priors for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 605–14 615
26. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint* arXiv:2010.11929, 2020
27. P. Zhou, Y. Zhou, C. Si, W. Yu, T. K. Ng, and S. Yan, "Mugs: A multi-granular self-supervised learning framework," *arXiv preprint* arXiv:2203.14415, 2022
28. J. Zhou, C. Wei, H. Wang, W. Shen, C. Xie, A. Yuille, and T. Kong, "ibot: Image bert pre-training with online tokenizer," *arXiv preprint* arXiv:2111.07832, 2021
29. M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint* arXiv:2304.07193, 2023
30. M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9650–9660
31. Newman, M.E.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E **74**(3), 036104 (2006)
32. Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech: Theory Exp. **2008**(10), P10008 (2008)
33. Tsitsulin, A., Palowitch, J., Perozzi, B., Müller, E.: Graph clustering with graph neural networks. J. Mach. Learn. Res. **24**(127), 1–21 (2023)
34. R. Harb and P. Knöbelreiter, "Infoseg: Unsupervised semantic image segmentation with mutual information maximization," in *Pattern Recognition: 43rd DAGM German Conference, DAGM GCPR 2021, Bonn, Germany, September 28–October 1, 2021, Proceedings*. Springer, 2022, pp. 18–32

35. A. Aflalo, S. Bagon, T. Kashti, and Y. Eldar, "Deepcut: Unsupervised segmentation using graph neural networks clustering," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 32–41
36. N. Veldt, D. F. Gleich, and A. Wirth, "A correlation clustering framework for community detection," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 439–448
37. Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Proesmans, M., Van Gool, L.: SCAN: Learning to Classify Images Without Labels. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12355, pp. 268–285. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58607-2_16
38. M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223
39. Kuhn, H.W.: The hungarian method for the assignment problem. Naval Research Logistics (NRL) **52**(1), 7–21 (2005)

SODAWideNet++: Combining Attention and Convolutions for Salient Object Detection

Rohit Venkata Sai Dulam(✉) and Chandra Kambhamettu

Video/Image Modeling and Synthesis (VIMS) Lab, University of Delaware, Newark, DE 19716, USA
{rdulam,chandrak}@udel.edu

Abstract. Salient Object Detection (SOD) has traditionally relied on feature refinement modules that utilize the features of an ImageNet pre-trained backbone. However, this approach limits the possibility of pre-training the entire network because of the distinct nature of SOD and image classification. Additionally, the architecture of these backbones originally built for Image classification is sub-optimal for a dense prediction task like SOD. To address these issues, we propose a novel encoder-decoder-style neural network called **SODAWideNet++** that is designed explicitly for SOD. Inspired by the vision transformers' ability to attain a global receptive field from the initial stages, we introduce the Attention Guided Long Range Feature Extraction (AGLRFE) module, which combines large dilated convolutions and self-attention. Specifically, we use attention features to guide long-range information extracted by multiple dilated convolutions, thus taking advantage of the inductive biases of a convolution operation and the input dependency brought by self-attention. In contrast to the current paradigm of ImageNet pre-training, we modify 118K annotated images from the COCO semantic segmentation dataset by binarizing the annotations to pre-train the proposed model end-to-end. Further, we supervise the background predictions along with the foreground to push our model to generate accurate saliency predictions. SODAWideNet++ performs competitively on five different datasets while only containing 35% of the trainable parameters compared to the state-of-the-art models. The code and pre-computed saliency maps are provided at https://github.com/VimsLab/SODAWideNetPlusPlus.

Keywords: Salient Object Detection · Self Attention · Dilated Convolutions

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78192-6_14.

1 Introduction

Salient Object Detection (SOD) requires identifying the objects that catch a viewer's immediate attention from visual data. Saliency is vital for many areas of Computer Vision, including Semantic Segmentation [25], Person Identification [12], etc. The first methods for SOD [1,10] relied on hand-crafted priors like color, contrast, etc. With the advent of Deep Learning, the emphasis has shifted towards developing Deep Learning based solutions for SOD.

The fundamental idea of Deep Learning models for SOD involves building novel feature refinement modules that rely on semantic features extracted using ImageNet pre-trained backbones. However, these backbones are primarily designed for image classification and do not fully meet the intricate needs of SOD. SOD often involves analyzing images with multiple objects, unlike the single-object focus typical in image classification datasets. This discrepancy can result in less optimal saliency predictions. Additionally, the standard approach of only integrating the backbone with the refinement modules during the fine-tuning phase misses a critical chance to pre-train these components together, which could enhance overall model performance. To better address these challenges, our models are designed specifically for SOD from the ground up, allowing for the entire network to be pre-trained simultaneously. A crucial part of our approach involves adapting the COCO semantic segmentation dataset [16] for SOD by converting the segmentation labels to saliency labels. Although significantly smaller than other pre-training datasets like ImageNet, our results illustrate the advantages of pre-training the entire model.

Moving toward the architectural innovations that underpin our model, it is crucial to recognize the dramatic shift from traditional Convolutional Neural Networks [9,24] to Vision Transformers (ViTs) [4], which have significantly advanced the benchmarks across various computer vision tasks. Unlike CNNs, which are constrained by their local receptive fields and often fail to adapt to the unique characteristics of different inputs, ViTs excel by capturing global relationships and input-specific details through self-attention. As a consequence of the local receptive field, CNNs typically employ a hierarchical approach to extracting global features that rely on extreme downsampling of the input. While broadening the receptive field, this process results in a substantial loss of detail, a critical drawback for tasks requiring high-resolution outputs. Additionally, the convolution operation in CNNs is designed to detect common patterns across different instances, which does not suffice for capturing attributes unique to individual instances. Several studies [2,36] have attempted to overcome these limitations but often incur high computational costs. To furnish a CNN with the ability to utilize instance-specific information, we employ self-attention and use it to guide the features extracted by convolutional operations.

Pre-trained on the modified COCO dataset, we propose **SODAWideNet**++, a modified SODAWideNet [5] architecture that seamlessly integrates Attention into convolutional components to extract local and global features. To extract global features, we combine dilated convolutions from the Multi-Receptive Field Feature Aggregation Module (MRFFAM) and Attention from

Multi-scale Attention (MSA) modules and propose Attention guided Long Range Feature Extraction (AGLRFE). Furthermore, we modified the Local Processing Module (LPM) by adding Attention and thus proposed an Attention-enhanced Local Processing Module (ALPM) to extract local features. Finally, we reuse the Cross Feature Module (CFM) to combine features from both the proposed components. We summarize our contributions below -

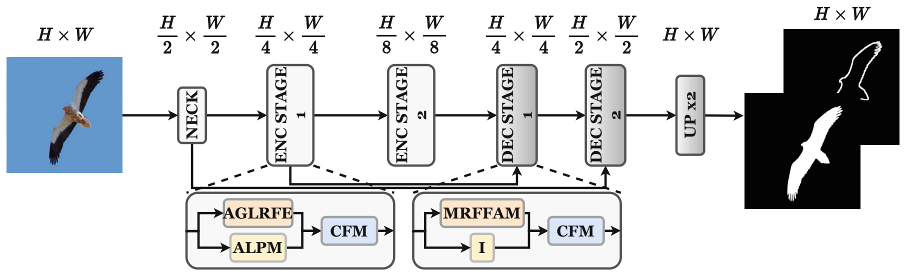
1. We propose SODAWideNet++, a convolutional model utilizing self-attention to extract global features from the beginning of the network.
2. We modify the famous COCO semantic segmentation dataset to generate binary labels and use it to pre-train the proposed model.
3. We propose AGLRFE to extract features from large receptive fields using dilated convolutions and Self-Attention.
4. Unlike prior works, we supervise foreground and background predictions for increased accuracy.

2 Related Works

2.1 Salient Object Detection with ImageNet Pre-Training

PiCAN [17] developed a contextual attention module to attend to essential context locations for every pixel from Resnet-50 features. BASN [21] uses an encoder-decoder-style network initialized by a Resnet-34 model and a boundary refinement network on top to produce accurate saliency predictions with crisp boundaries. (F3-N) [27] uses a Resnet-50 network to extract semantic features refined by a cascaded feedback decoder (CFD) and cross-feature module to produce saliency outputs. VST [18] is the first work to propose a vision transformer-based SOD model. PSG uses a loss function that creates auxiliary saliency maps based on the morphological closing operation to generate accurate saliency maps incrementally. ET [35] uses an energy-based prior for salient object detection. RCSB [11] refines features extracted from a Resnet-50 [9] backbone using Stage-wise Feature Extraction (SFE) and a few novel loss functions for SOD. CSF-R2 [8] proposes a flexible convolutional module named gOctConv to utilize multi-scale features for SOD. EDNet [29] presents a novel downsampling technique to learn a global view of the whole image to generate high-level features for SOD. PGN uses a combination of Resnet and Swin [19] models to generate saliency maps. ICON [37] introduces a diverse feature aggregation (DFA) component to aggregate features with various receptive fields and increase the feature diversity. TR [13] uses an EfficientNet backbone and attention-guided tracing modules to detect salient objects. LDF [28] proposes a label decoupling framework to detect salient objects. The authors disintegrate the original saliency map into body and detail maps to concentrate on the central object and object edges separately. PA-KRN [31] uses a knowledge review network to first identify the salient object and then segment it. RMF [3] proposes a novel Recurrent Multi-scale transformer that utilizes a transformer and multi-scale refinement architectures for SOD. SR [34] proposes a novel framework that enhances global context modeling and detail preservation to generate accurate saliency predictions. VSC [20] proposes a foundational model for SOD that uses programmable prompts to generate saliency predictions.

3 Proposed Method



$H \times W$
$\frac{H}{2} \times \frac{W}{2}$
$\frac{H}{4} \times \frac{W}{4}$
$\frac{H}{8} \times \frac{W}{8}$
$\frac{H}{4} \times \frac{W}{4}$
$\frac{H}{2} \times \frac{W}{2}$
$H \times W$
$H \times W$
NECK
ENC STAGE 1
ENC STAGE 2
DEC STAGE 1
DEC STAGE 2
UP x2
AGLRFE
ALPM
CFM
MRFFAM
I
CFM

Fig. 1. The proposed architecture **SODAWideNet++** contains two branches, one to extract global features using AGLRFE (f_{lr}) and the other to extract local features using ALPM (f_{sr}). These global and local features pass through CFM, producing the output of an encoding stage. The decoding stages also consist of two parallel paths, MRFFAM, to decode features through multiple receptive fields and an Identity operation.

The proposed model is an improved version of the design principles introduced in [5]. This section will first explain the SODAWideNet [5] model, which serves as the basis for SODAWideNet++. Then, we will introduce the core components of SODAWideNet++, namely the Attention Guided Long Range Feature Extraction (AGLRFE) and the Attention Local Pooling Module (ALPM). Finally, we will discuss the specific loss function used in our model and other essential elements of our design strategy.

3.1 SODAWideNet

SODAWideNet, as described in [5], employs an encoder-decoder architecture to generate saliency predictions. The encoder is composed of three parallel pathways: the Multi-Receptive Field Feature Aggregation Module (MRFFAM), the Multi-Scale Attention (MSA), and the Local Processing Module (LPM). These pathways are engineered to capture both global and local features concurrently. The MRFFAM extracts and consolidates semantic information from multiple receptive fields using large dilated convolutions, thereby strengthening the model's capability to identify objects of varying sizes. Similarly, MSA utilizes self-attention to extract global features in a hierarchical manner, keeping in mind the computational complexity of the attention operation. The LPM extracts local features using smaller 3×3 convolutions and multiple maxpooling operations. The Cross-Feature Module (CFM) merges MRFFA, MSA, and LP module features. This module contains a series of convolutions that selectively combine features due to the distinct nature of features obtained from the three components. Each convolution consists of a 3×3 convolution operation followed by a Group Normalization layer and GELU activation function. In the decoding phase, the architecture features two parallel paths, MRFFAM and Identity, which work together to decode the encoded features and generate the final saliency outputs. The overall architecture consists of two such encoder and decoder blocks. We reuse these decoder blocks in the proposed SODAWideNet++ model.

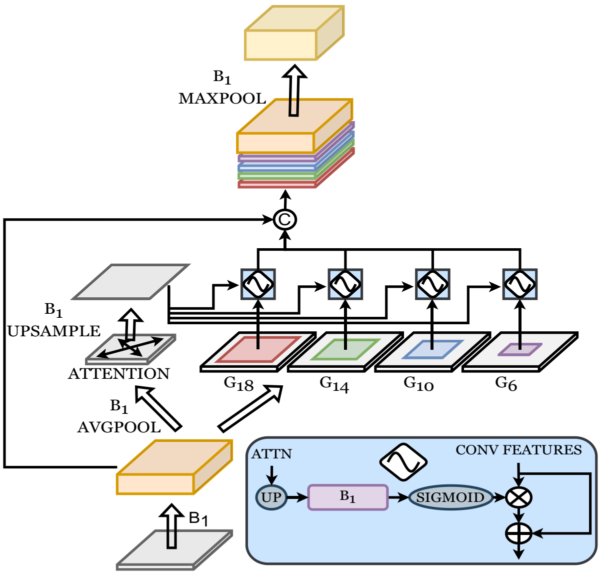


Fig. 2. Attention-guided Long-Range Feature Extraction module (AGLRFE) consists of two branches: a collection of dilated convolutions with different dilation rates to extract long-range convolution features and a self-attention block. We reduce the spatial resolution of the input before the Self-Attention block using Average Pooling and refine them using a series of convolution operations. Attention features are then upsampled to the exact resolution as the convolution features from the dilated convolutions. Then, using a series of convolution layers, we bring its channel size to one and pass it through a Sigmoid layer. These features refine our long-range convolution features, thus inducing input-reliance.

3.2 SODAWideNet++

SODAWideNet++ builds upon the foundational architecture of SODAWideNet, maintaining distinct branches for extracting local and global features. While SODAWideNet has demonstrated commendable performance when trained from scratch, it still lags behind state-of-the-art models in terms of overall efficacy. In SODAWideNet++, we refine the approach of capturing long-range features by merging the functionalities of dilated convolutions and attention into a single, streamlined module named Attention-Guided Long Range Feature Extraction (AGLRFE). This module is engineered to generate long-range, input-dependent convolutional features, thereby enhancing the robustness of the feature representation. Additionally, we incorporate attention into the Local Processing Module (LPM) to increase input dependency, rebranding it as the Attention-enhanced LPM (ALPM). The Cross-Feature Module (CFM) remains unchanged and is tasked with the integration and refinement of both local and global features. The decoder structures in SODAWideNet++ and SODAWideNet are identical, preserving the architectural consistency across both models.

3.3 Attention Guided Long Range Feature Extraction (AGLRFE)

Vision transformers have achieved significant success due to the effectiveness of Self-Attention in extracting important semantic features across large receptive fields. On the other hand, conventional CNNs achieve a global receptive field by downsampling the input hierarchically, which can lead to the loss of critical features and increase the model's parameters.

To overcome these limitations, MRFFAM employs multiple dilated convolutions with large dilation rates to expand the receptive field of our network. The input is divided into chunks, and each chunk is input to a dilated convolution with a specific dilation rate. The output of the MRFFAM block is to concatenate the outputs of dilated convolutions along with the input in the channel dimension. This resultant feature map is either sent through a downsampling layer or a series of convolutions, depending on whether it is in the encoder or the decoder.

However, the intrinsically input-dependent nature of Self-Attention is absent in a convolution operation, thus restricting the effectiveness of a CNN. Thus, to leverage the strengths of convolutions and Self-Attention, we propose a hybrid module, modifying the MRFFAM block to contain an Attention operation. Specifically, we use dilated convolutions to capture long-range convolutional features and Self-Attention to derive input-specific features and use them to guide the convolution features. Below, we illustrate particular components of AGLRFE followed by a sequence of operations.

$$\begin{aligned} B_i(x) &= convB_i(convB_i(x)) \\ G_i(x) &= conv_i(conv_i(x)) \end{aligned}$$

$convB_i$ implies a 3×3 convolution with dilation i followed by Batch Normalization and GELU activation function, whereas $conv_i$ implies a 3×3 convolution with dilation i followed by Group Normalization and GELU activation function.

As seen in Fig. 2, the input to an AGLRFE block X is transformed using a series of 3×3 convolution layers.

$$feat = B_1(X)$$

Next, there are two parallel paths: one path computes attention features, and the other computes convolutional features of different receptive fields. To compute the attention features, *feat* is downsampled using an Average Pooling layer (*AvgPool*), which is then transformed by a series of convolution layers.

$$f_{attention} = Attn(B_1(AvgPool(feat)))$$

$$Attn = softmax\left(\frac{K \times Q}{\sqrt{d_k}}\right)V$$

where K, Q and V are computed using $conv_1(feat)$. Once computed, the attention features guide the outputs of various dilated convolutions in the following manner -

$$f_{conv} = \{f_i + (f_i \times \sigma(B_1(\uparrow f_{attention})))\} \quad \text{where} \quad i \in \{6, 10, 14, 18, 22\}$$

where $f_i = G_i(feat)$ indicate the dilated convolution features, $\uparrow$ indicates bilinear upsampling, and σ indicates a sigmoid operation. To influence the convolutional features, the attention features are upsampled to the same spatial resolution, then reducing their channels to 1 and, finally, passing through a sigmoid layer. These logits are multiplied with the convolutional features, thus filtering crucial per-channel information and inducing an input-dependent nature. Finally, all the refined features are concatenated $\{\}$ in the channel dimension. Thus, the output of an AGLRFE is obtained by concatenating the newly generated convolution features and $feat$ and passing these features through a max pooling *MaxPool* layer followed by a series of convolution layers.

$$f_{lr} = B_1(MaxPool(\{f_{conv}, feat\}))$$

3.4 Attention-Enhanced Local Processing Module (ALPM)

The Local Processing Module (LPM) in [5] employs 3×3 convolutions to extract local features crucial for precise saliency predictions. This module uses a series of max-pooling layers to identify discriminative features from small neighborhoods, enhancing the model's ability to focus on relevant details. To augment this structure with an added layer of input-specific adaptability, we have refined the LPM by incorporating a Self-Attention mechanism. This Self-Attention is strategically applied to the feature map with the smallest spatial resolution within the LPM, enabling the module to obtain input characteristics from the most informative feature map.

$$fx = MAXPOOL(X)$$
$$feat = B_1(MAXPOOL(fx))$$
$$fy = B_1(\{\uparrow (feat + \mathbf{Attn}(feat)), fx\})$$
$$f_{sr} = B_1(fx) + fy$$

Above, we illustrate the series of operations in LPM along with the modification highlighted in **bold**.

3.5 Loss Function

Following [5], SODAWideNet++ produces saliency and contour outputs. Specifically, AGLRFE, ALPM, CFM, MRFFAM, and CFMD outputs are used to generate saliency maps. Additionally, the outputs of MRFFAM and CFMD also produce contour maps. Thus, we create a custom loss to supervise these outputs with their ground truth, as shown in Eqs. 1 and 2.

$$\begin{aligned} L_{salient} = \sum_{i=1}^{2}(&L^{sal}_{AGLRFE_{(i)}} + L^{sal}_{ALPM_{(i)}} + L^{sal}_{CFM_{(i)}} \\ &+L^{sal}_{MRFFAM_{(i)}} + L^{sal}_{CFMD_{(i)}}) \end{aligned} \tag{1}$$

$$\begin{aligned} L_{contour} &= L^{con}_{MRFFAM_{(1)}} + L^{con}_{MRFFAM_{(2)}} \\ &+L^{con}_{CFMD_{(1)}} + L^{con}_{CFMD_{(2)}} \\ L^{con} &= 0.001 \cdot L_{BCE} + L_{dice} \end{aligned} \tag{2}$$

$L_{salient}$ and $L_{contour}$ imply the total Saliency loss and the total Contour loss, respectively. L_{BCE} and L_{dice} are the Binary Cross Entropy and Dice loss, respectively. Unlike other methods, we supervise foreground and background saliency maps. Hence, L^{sal} is divided into two parts. The foreground loss denoted by L_{fg} is the same as the loss function in SODAWideNet, whereas L_{bg} is the newly added background supervision. For COCO pre-training, we set β to one, and for DUTS, we set it to 0.5.

$$\begin{aligned} L^{sal} &= L_{fg} + \beta \cdot L_{bg} \\ L_{fg} &= \alpha^{fg} \cdot L_{bce} + \alpha^{fg} \cdot L_{iou} + \alpha^{fg} \cdot L_1 \\ L_{bg} &= \alpha^{bg} \cdot L_{bce} + \alpha^{bg} \cdot L_{iou} + \alpha^{bg} \cdot L_1 \end{aligned} \tag{3}$$

where L_{bce}, L_{iou}, and L_1 are per-pixel Binary Cross Entropy, Intersection-over-Union, and L1 losses, respectively. α^{fg}_{ij} and α^{bg}_{ij} are per-pixel weights determined for each pixel during foreground and background supervision. Per-pixel weights ensure that wrong predictions for specific pixels are penalized heavily. Below, we illustrate the procedure to compute these values. Both α^{fg} and α^{bg} have the exact spatial resolution as the output. Hence, we compute pixel-wise loss in Eq. 3 and multiply them with α.

$$\begin{aligned} \alpha^{fg}_{ij} &= max(GT_{ij})_{31\times31} \\ \alpha^{bg}_{ij} &= BackgroundGT_{ij} \\ BackgroundGT &= \mathbb{1} - GT \end{aligned}$$

α^{fg}_{ij} for a pixel at spatial location (i, j) is calculated by finding the largest value in a 31×31 window centered on that pixel location. Whereas α^{bg}_{ij} equals the pixel's intensity value in the background map. Figure 3 indicates the values of α_{ij}^{fg} and α_{ij}^{bg} superimposed on the input. Finally, from Eqs. 1, and 2, the total loss to train our model is given as

$$L_{total} = L_{salient} + L_{contour} \tag{4}$$

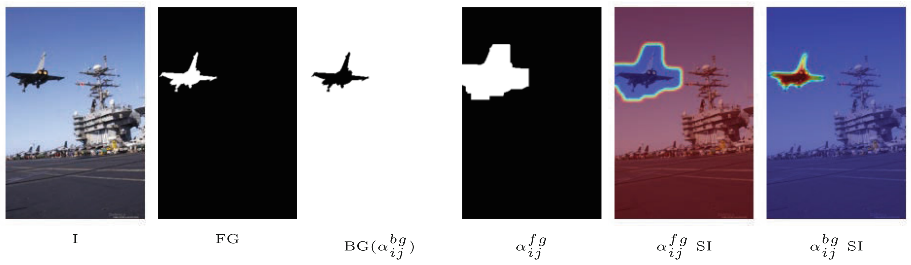


Fig. 3. In the above figure, we visually illustrate the pixels that receive a higher weight for loss computation as shown in Eq. 3. The third image illustrates in white the pixels that receive a higher weight for the background loss, and the fourth image illustrates the pixels receiving a higher weight for the foreground loss. The last two images depict the important pixels (in blue) superimposed (SI) on the input image. (Color figure online)

4 Experiments and Results

4.1 Datasets

We pre-train our model on the modified COCO dataset of 118K annotated images and ground truth pairs. We further augment it using horizontal and vertical flipping, taking the pre-training data size to 354K images. Then, we fine-tune our model on the DUTS [26] dataset, containing 10,553 images for training. We augment the data using horizontal and vertical flipping to obtain a training dataset of 31,659 images. We use five datasets to evaluate the proposed model. They are DUTS-Test [26] consisting of 5019 images, DUT-OMRON [32] which consists of 5168 images, HKU-IS [14] which consists of 4447 images, ECSSD [23] which consists of 1000 images and PASCAL-S [15] dataset consisting of 850 images.

4.2 Implementation Details

For COCO pre-training, we train our model for 21 epochs. The LR is set at 0.001 and multiplied by 0.5 after 15 epochs. For SOD fine-tuning, we train our model for a further 11 epochs with the same LR as the COCO stage. LR is multiplied by 0.1 after five epochs. We follow the procedure used in U-Net [22] to

Table 1. Comparison of our method with 17 other methods in terms of max F-measure F_{max}, MAE, S_m, E_m, and F_w on DUTS-TE, DUT-OMRON, and HKU-IS datasets.

Method	Params. (M)	DUTS-TE [26]					DUT-OMRON [32]					HKU-IS [14]				
		F_{max}	MAE	S_m	E_m	F_w	F_{max}	MAE	S_m	E_m	F_w	F_{max}	MAE	S_m	E_m	F_w
PiCAN$_{\text{CVPR'18}}$ [17]	47.22	0.860	0.051	0.869	0.862	0.755	0.803	0.065	0.832	0.841	0.695	0.918	0.043	0.904	0.936	0.840
BASN$_{\text{CVPR'19}}$ [21]	87.06	0.860	0.048	0.866	0.884	0.803	0.805	0.056	0.836	0.861	0.751	0.928	0.032	0.909	0.946	0.889
F3-N$_{\text{AAAI'20}}$ [27]	26.5	0.891	0.035	0.888	0.902	0.835	0.813	0.053	0.838	0.870	0.747	0.937	0.028	0.917	0.953	0.900
LDF$_{\text{CVPR'20}}$ [28]	25.15	0.898	0.034	0.892	0.910	0.845	0.820	0.051	0.838	0.873	0.752	0.939	0.027	0.919	0.954	0.904
PA-KRN$_{\text{AAAI'21}}$ [31]	68.68	0.907	0.033	0.900	0.916	0.861	0.834	0.050	0.853	0.885	0.779	0.943	0.027	0.923	0.955	0.909
VST$_{\text{ICCV'21}}$ [18]	44.48	0.890	0.037	0.896	0.892	0.828	0.825	0.058	0.850	0.861	0.755	0.942	0.029	0.928	0.953	0.897
PSG$_{\text{TIP'21}}$ [33]	25.55	0.886	0.036	0.883	0.908	0.835	0.811	0.052	0.831	0.870	0.747	0.938	0.027	0.919	0.906	0.958
ET$_{\text{NeurIPS'21}}$ [35]	118.96	0.910	0.029	0.909	0.918	0.871	0.839	0.050	0.858	0.886	0.788	0.947	0.023	0.930	0.961	0.920
RCSB$_{\text{WACV'22}}$ [11]	27.90	0.889	0.035	0.878	0.903	0.840	0.810	0.045	0.820	0.856	0.723	0.938	0.027	0.918	0.954	0.909
CSF-R2$_{\text{TPAMI'22}}$ [8]	36.53	0.890	0.037	0.890	0.897	0.823	0.815	0.055	0.838	0.861	0.734	0.935	0.030	0.921	0.952	0.891
EDN$_{\text{TIP'22}}$ [29]	42.85	0.895	0.035	0.892	0.908	0.845	0.828	0.048	0.846	0.876	0.770	0.941	0.026	0.924	0.956	0.908
PGN$_{\text{CVPR'22}}$ [30]	72.70	0.917	0.027	0.911	0.922	0.874	0.835	0.045	0.855	0.887	0.775	0.948	0.024	0.929	0.961	0.916
ICON-R$_{\text{TPAMI'22}}$ [37]	33.09	0.892	0.037	0.889	0.902	0.837	0.825	0.057	0.844	0.870	0.761	0.939	0.029	0.920	0.952	0.902
TR5$_{\text{AAAI'22}}$ [13]	31.30	0.916	0.026	0.909	0.927	0.883	0.834	0.042	0.847	0.880	0.787	0.947	0.022	0.930	0.961	0.922
SR$_{\text{TMM'23}}$ [34]	220	0.925	0.024	0.921	0.924	0.886	0.838	0.043	0.859	0.884	0.782	0.951	0.023	0.934	0.962	0.918
RMF$_{\text{ACMMM'23}}$ [3]	87.52	0.931	0.023	0.925	0.933	0.900	0.861	0.040	0.877	0.904	0.819	0.957	0.019	0.940	0.968	0.934
VSC$_{\text{CVPR'24}}$ [20]	74.72	0.931	0.024	0.926	0.931	0.897	0.861	0.042	0.876	0.899	0.813	0.957	0.021	0.940	0.965	0.930
Ours	**26.58**	**0.917**	**0.029**	**0.910**	**0.916**	**0.870**	**0.848**	**0.045**	**0.868**	**0.896**	**0.796**	**0.950**	**0.024**	**0.932**	**0.961**	**0.917**
Ours-M	**6.66**	**0.901**	**0.035**	**0.898**	**0.907**	**0.846**	**0.844**	**0.048**	**0.861**	**0.888**	**0.784**	**0.949**	**0.025**	**0.932**	**0.960**	**0.915**
Ours-S	**1.67**	**0.887**	**0.039**	**0.886**	**0.898**	**0.824**	**0.834**	**0.051**	**0.854**	**0.882**	**0.772**	**0.941**	**0.028**	**0.925**	**0.955**	**0.904**

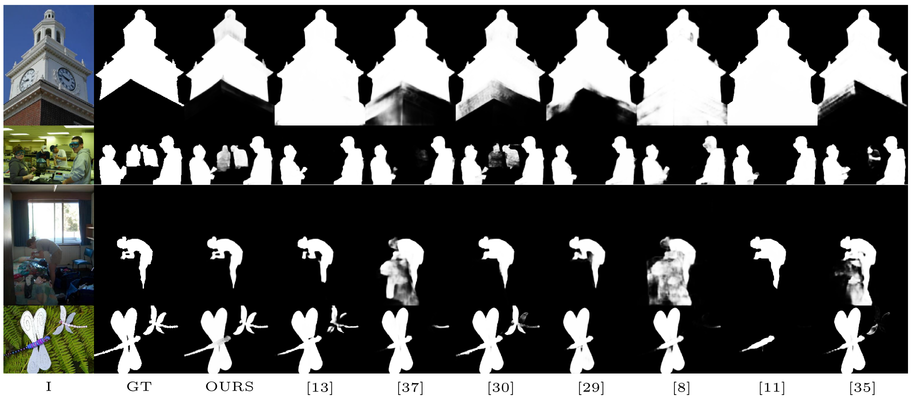


Fig. 4. In the above figure, we visually compare our results against other models.

initialize the weights of our model. Images are resized to 384×384 for training and testing. The predictions for the background saliency supervision are generated by multiplying the pre-sigmoid predictions with −1, thus turning the negative values into positive and positive values into negative. The evaluation metrics for comparing our works with prior works are the Mean Absolute Error(MAE), maximum F-measure, the S-measure [6], the E-measure [7], and the weighted F-measure. The lower the MAE and the higher the F_{max}, S_m, E_m, and F_w scores, the better the model.

Table 2. Comparison of our method with 17 other methods in terms of max F-measure F_{max}, MAE, S_m, E_m, and F_w measures on ECSSD and PASCAL-S datasets.

Method	Params. (M)	ECSSD [23]					PASCAL-S [15]				
		F_{max}	MAE	S_m	E_m	F_w	F_{max}	MAE	S_m	E_m	F_w
PiCAN$_{\text{CVPR'18}}$ [17]	47.22	0.935	0.046	0.917	0.913	0.867	0.868	0.078	0.852	0.837	0.779
BASN$_{\text{CVPR'19}}$ [21]	87.06	0.942	0.037	0.916	0.921	0.904	0.860	0.079	0.834	0.850	0.797
F3-N$_{\text{AAAI'20}}$ [27]	26.5	0.945	0.033	0.924	0.927	0.912	0.882	0.064	0.857	0.863	0.823
LDF$_{\text{CVPR'20}}$ [28]	25.15	0.950	0.034	0.924	0.925	0.915	0.887	0.062	0.859	0.869	0.829
PA-KRN$_{\text{AAAI'21}}$ [31]	68.68	0.953	0.032	0.928	0.924	0.918	-	-	-	-	-
VST$_{\text{ICCV'21}}$ [18]	44.48	0.951	0.033	0.932	0.918	0.910	0.890	0.062	0.871	0.846	0.827
PSG$_{\text{TIP'21}}$ [33]	25.55	0.949	0.031	0.925	0.928	0.917	0.886	0.063	0.858	0.863	0.830
ET$_{\text{NeurIPS'21}}$ [35]	118.96	0.959	0.023	0.942	0.933	0.937	0.900	0.055	0.876	0.869	0.863
RCSB$_{\text{WACV'22}}$ [11]	27.90	0.944	0.033	0.921	0.923	0.916	0.886	0.061	0.857	0.858	0.834
CSF-R2$_{\text{TPAMI'22}}$ [8]	36.53	0.950	0.033	0.930	0.928	0.910	0.886	0.069	0.862	0.855	0.818
EDN$_{\text{TIP'22}}$ [29]	42.85	0.951	0.032	0.927	0.929	0.918	0.891	0.065	0.860	0.867	0.832
PGN$_{\text{CVPR'22}}$ [30]	72.70	0.960	0.027	0.918	0.932	0.929	0.904	0.056	0.874	0.878	0.849
ICON-R$_{\text{TPAMI'22}}$ [37]	33.09	0.950	0.032	0.929	0.929	0.918	0.888	0.066	0.860	0.861	0.828
TR5$_{\text{AAAI'22}}$ [13]	31.30	0.956	0.027	0.933	0.926	0.931	0.907	0.051	0.878	0.875	0.859
SR$_{\text{TMM'23}}$ [34]	220	0.962	0.025	0.941	0.935	0.932	-	-	-	-	-
RMF$_{\text{ACMMM'23}}$ [3]	87.52	0.964	0.020	0.947	0.938	0.946	-	-	-	-	-
VSC$_{\text{CVPR'24}}$ [20]	74.72	0.965	0.021	0.949	0.934	0.942	0.912	0.051	0.885	0.870	0.863
Ours	**26.58**	**0.957**	**0.029**	**0.935**	**0.927**	**0.922**	**0.901**	**0.062**	**0.875**	**0.868**	**0.844**
Ours-M	**6.66**	**0.952**	**0.033**	**0.930**	**0.26**	**0.915**	**0.895**	**0.065**	**0.866**	**0.865**	**0.834**
Ours-S	**1.67**	**0.946**	**0.037**	**0.923**	**0.924**	**0.906**	**0.884**	**0.070**	**0.857**	**0.860**	**0.818**

4.3 Quantitative and Qualitative Results

Tables 1 and 2 detail the quantitative performance of SODAWideNet++ compared to 17 other state-of-the-art models. Notably, on the DUTS-TE, DUT-OMRON, and HKU-IS datasets, our model achieves competitive scores in F_{max}, S_m, and E_m measures which signify highly confident and accurate predictions. Especially while SODAWideNet++ (Ours) uses significantly fewer parameters (**33%**, **35%**, and **13%** of trainable parameters compared to the state-of-the-art RMF [3], VSC [20], and SR [34], respectively). Additionally, the smaller models SODAWideNet++-M (Ours-M) perform considerably well and surpass the older state-of-the-art models such as VST and RCSB. Similarly, the smallest model, SODAWideNet++-S (Ours-S), also illustrates great performance and can be useful in parameter-constrained situations.

The visual results, as depicted in Fig. 4, further substantiate the robustness of SODAWideNet++ across diverse scenarios, including images with large foreground objects, scenes containing multiple objects (notably in the second and fourth rows), and environments characterized by complex backgrounds (third and fourth rows). These results highlight the model's ability to detect salient objects in challenging visual conditions.

5 Ablation Experiments

Through ablation experiments, we delve into the effects of the proposed design choices on the proposed model. All reported numbers are on the DUTS test split.

5.1 SODAWideNet vs. SODAWideNet++

We performed a comparative analysis of SODAWideNet and SODAWideNet++ performance, evaluating both models from scratch and after pre-training on the modified COCO dataset, ensuring comparisons were made against models of similar size for consistency. SODAWideNet++ integrates the Multi-Receptive Field Feature Aggregation Module (MRFFAM) and Multi-scale Attention (MSA) into a single module, the Attention Guided Long Range Feature Extraction (AGLRFE). This integration reduced redundancy when extracting long-range features. Additionally, incorporating background supervision in SODAWideNet++ enhances its ability to distinguish between foreground and background areas, leading to more precise saliency results. Table 3 illustrates the quantitative performance.

Table 3. SODAWideNet vs. SODAWideNet++.

Pre-training mechanism	Model	Size (in M)	F_{max}	MAE
Scratch	SODAWideNet	9.03	0.883	0.043
Mod. COCO	SODAWideNet	9.03	0.899	0.035
Scratch	SODAWideNet++	6.66	0.881	0.043
Mod. COCO	SODAWideNet++	6.66	0.901	0.035

5.2 ImageNet vs. COCO Pre-training

Table 4 compares our proposed models' performance when pre-trained on either the ImageNet or the modified COCO dataset. The results show an improvement with COCO pre-training, where the model experiences a 1.2% increase in performance compared to ImageNet. This significant enhancement can be attributed to the fact that ImageNet pre-training tends to separate the development of the backbone and the feature refinement modules during the pre-training phase, leading to a disconnect in feature interpretation between the encoding and decoding stages when fine-tuning for SOD. Also, training the model from scratch achieves decent performance, highlighting the effectiveness of incorporating self-attention into a convolutional neural network.

Table 4. ImageNet vs. COCO vs. Training from scratch.

Pre-training mechanism	Model	F_{max}	MAE
Scratch	SODAWideNet++	0.890	0.039
ImageNet	SODAWideNet++	0.905	0.036
Mod. COCO	SODAWideNet++	0.917	0.029

5.3 Pre-training Another Model Using the Modified COCO Dataset

We pre-train the PGN [30] model using the modified COCO dataset without their ImageNet pre-trained weights. Tables 5 and 6 contain a quantitative evaluation of the COCO pre-trained PGN, ImageNet pre-trained PGN, and PGN trained from scratch on the DUTS dataset. We have included our model for comparison. Pre-training using the COCO dataset significantly outperforms training from scratch and delivers competitive results against the ImageNet pre-trained PGN model without optimal hyperparameters.

Table 5. Comparison of pre-training (COCO vs. ImageNet) of PGN method.

Method	Params. (M)	DUTS-TE					DUT-OMRON					HKU-IS				
		F_{max}	MAE	S_m	E_m	E_m	F_{max}	MAE	S_m	E_m	E_m	F_{max}	MAE	S_m	E_m	E_m
PGN-S	72.70	0.823	0.060	0.833	0.851	0.731	0.779	0.068	0.809	0.837	0.690	0.909	0.042	0.891	0.934	0.852
$\text{PGN}_{\text{CVPR'22}}$ [30]	72.70	0.917	0.027	0.911	0.922	0.874	0.835	0.045	0.855	0.887	0.775	0.948	0.024	0.929	0.961	0.916
$\text{PGN-COCO}_{\text{CVPR'22}}$ [30]	72.70	0.882	0.040	0.879	0.896	0.820	0.803	0.057	0.828	0.859	0.732	0.931	0.031	0.912	0.948	0.732
Ours	**26.58**	**0.917**	**0.029**	**0.910**	**0.916**	**0.916**	**0.848**	**0.045**	**0.868**	**0.896**	**0.916**	**0.950**	**0.024**	**0.932**	**0.960**	**0.916**

Table 6. Comparison of pre-training (COCO vs. ImageNet) of PGN method.

Method	Params. (M)	ECSSD					PASCAL-S				
		F_{max}	MAE	S_m	E_m	F_w	F_{max}	MAE	S_m	E_m	F_w
PGN-S	72.70	0.916	0.054	0.891	0.907	0.855	0.839	0.094	0.812	0.824	0.750
$\text{PGN}_{\text{CVPR'22}}$ [30]	72.70	0.960	0.027	0.938	0.932	0.929	0.904	0.056	0.874	0.878	0.849
$\text{PGN-COCO}_{\text{CVPR'22}}$ [30]	72.70	0.940	0.039	0.913	0.915	0.896	0.878	0.069	0.852	0.864	0.818
SODAWideNet++	**26.58**	**0.957**	**0.029**	**0.935**	0.927	0.922	**0.901**	**0.062**	**0.875**	**0.870**	0.845

5.4 Ablation Experiments Corresponding to ALGRFE, ALPM, CFM, and MRFFAM

Table 7 provides a quantitative comparison of the impact of each component of the proposed architecture. All these models are pre-trained on the modified COCO dataset and fine-tuned on the DUTS dataset. The absence of AGLRFE (row one) reduces the model's ability to capture long-range dependencies, leading to the lowest performance across all configurations. The removal of ALPM (row two) reduces the model's ability to capture local features. Similarly, removing CFM (row three) causes a similar decline in performance due to the lack of a complex way to integrate local and global features. Also, removing MRFFAM (row four) on the decoder side results in degraded performance, reinforcing the importance of using various receptive fields to decode features. The inclusion of all the four components produces the best model.

Table 7. Influence of individual components in SODAWideNet++.

AGLRFE	ALPM	CFM	MRFFAM	F_{max}	MAE
×	✓	✓	✓	0.906	0.035
✓	×	✓	✓	0.907	0.033
✓	✓	×	✓	0.908	0.032
✓	✓	✓	×	0.906	0.032
✓	✓	✓	✓	**0.917**	**0.029**

5.5 Difference from the Previous Loss Pipelines

Previous works [3, 20, 34] concentrated on supervising the foreground (saliency) maps (fg), while our approach involved supervising both the foreground and background (bg) maps. Our findings, as presented in Table 8, clearly demonstrate that incorporating background supervision (fg + bg) leads to a noticeable performance improvement compared to using foreground-only supervision, reinforcing the effectiveness of our approach.

Table 8. Influence of background (bg) supervision on SODAWideNet++.

Loss	F_{max}	MAE
fg + bg	0.917	0.029
fg	0.912	0.031

6 Conclusion

In conclusion, our SODAWideNet++ framework integrates the strengths of vision transformers and convolutional networks through the novel AGLRFE module. By using dilated convolutions paired with self-attention mechanisms, our model combines the inductive biases of convolutions and the dynamic, input-specific capabilities of attention mechanisms. This combination identifies salient objects across varied scenes and conditions. We used binarized annotations from the COCO dataset to train the model instead of traditional ImageNet pre-training. This tailored approach aligns directly with the nuances of SOD tasks, resulting in a model with competitive performance across multiple datasets. Our results demonstrate the effectiveness of our proposed pre-training approach and model design choices, where we achieve competitive performance against state-of-the-art models such as RMF [3] while only containing 35% of trainable parameters.

References

1. Cheng, M.M., Mitra, N.J., Huang, X., Torr, P.H., Hu, S.M.: Global contrast based salient region detection. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 569–582 (2014)
2. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 764–773 (2017)
3. Deng, X., Zhang, P., Liu, W., Lu, H.: Recurrent multi-scale transformer for high-resolution salient object detection. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 7413–7423 (2023)
4. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
5. Dulam, R.V.S., Kambhamettu, C.: Sodawidenet-salient object detection with an attention augmented wide encoder decoder network without imagenet pre-training. In: International Symposium on Visual Computing. pp. 93–105. Springer (2023)
6. Fan, D.P., Cheng, M.M., Liu, Y., Li, T., Borji, A.: Structure-measure: A new way to evaluate foreground maps. In: Proceedings of the IEEE international conference on computer vision. pp. 4548–4557 (2017)
7. Fan, D.P., Gong, C., Cao, Y., Ren, B., Cheng, M.M., Borji, A.: Enhanced-alignment measure for binary foreground map evaluation. arXiv preprint arXiv:1805.10421 (2018)
8. Gao, S.H., Tan, Y.Q., Cheng, M.M., Lu, C., Chen, Y., Yan, S.: Highly efficient salient object detection with 100k parameters. In: ECCV (2020)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
10. Jiang, B., Zhang, L., Lu, H., Yang, C., Yang, M.H.: Saliency detection via absorbing markov chain. In: Proceedings of the IEEE international conference on computer vision. pp. 1665–1672 (2013)
11. Ke, Y.Y., Tsubono, T.: Recursive contour-saliency blending network for accurate salient object detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 2940–2950 (January 2022)
12. Kim, H., Joung, S., Kim, I.J., Sohn, K.: Prototype-guided saliency feature learning for person search. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4865–4874 (June 2021)
13. Lee, M.S., Shin, W., Han, S.W.: Tracer: Extreme attention guided salient object tracing network (student abstract). In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 12993–12994 (2022)
14. Li, G., Yu, Y.: Visual saliency based on multiscale deep features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5455–5463 (June 2015)
15. Li, Y., Hou, X., Koch, C., Rehg, J.M., Yuille, A.L.: The secrets of salient object segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 280–287 (2014)
16. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common Objects in Context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48

17. Liu, N., Han, J., Yang, M.H.: Picanet: Learning pixel-wise contextual attention for saliency detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3089–3098 (2018)
18. Liu, N., Zhang, N., Wan, K., Shao, L., Han, J.: Visual saliency transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 4722–4732 (October 2021)
19. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10012–10022 (2021)
20. Luo, Z., Liu, N., Zhao, W., Yang, X., Zhang, D., Fan, D.P., Khan, F., Han, J.: Vscode: General visual salient and camouflaged object detection with 2d prompt learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17169–17180 (2024)
21. Qin, X., Zhang, Z., Huang, C., Gao, C., Dehghan, M., Jagersand, M.: Basnet: Boundary-aware salient object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 7479–7489 (2019)
22. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
23. Shi, J., Yan, Q., Xu, L., Jia, J.: Hierarchical image saliency detection on extended cssd. IEEE Trans. Pattern Anal. Mach. Intell. **38**(4), 717–729 (2015)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
25. Sun, W., Zhang, J., Barnes, N.: Inferring the class conditional response map for weakly supervised semantic segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 2878–2887 (2022)
26. Wang, L., Lu, H., Wang, Y., Feng, M., Wang, D., Yin, B., Ruan, X.: Learning to detect salient objects with image-level supervision. In: CVPR (2017)
27. Wei, J., Wang, S., Huang, Q.: F^{3}net: fusion, feedback and focus for salient object detection. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 12321–12328 (2020)
28. Wei, J., Wang, S., Wu, Z., Su, C., Huang, Q., Tian, Q.: Label decoupling framework for salient object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 13025–13034 (2020)
29. Wu, Y.H., Liu, Y., Zhang, L., Cheng, M.M., Ren, B.: Edn: Salient object detection via extremely-downsampled network. IEEE Transactions on Image Processing (2022)
30. Xie, C., Xia, C., Ma, M., Zhao, Z., Chen, X., Li, J.: Pyramid grafting network for one-stage high resolution saliency detection. In: CVPR (2022)
31. Xu, B., Liang, H., Liang, R., Chen, P.: Locate globally, segment locally: A progressive architecture with knowledge review network for salient object detection. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 3004–3012 (2021)
32. Yang, C., Zhang, L., Lu, H., Ruan, X., Yang, M.H.: Saliency detection via graph-based manifold ranking. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3166–3173 (2013)
33. Yang, S., Lin, W., Lin, G., Jiang, Q., Liu, Z.: Progressive self-guided loss for salient object detection. IEEE Trans. Image Process. **30**, 8426–8438 (2021). https://doi.org/10.1109/TIP.2021.3113794

34. Yun, Y.K., Lin, W.: Towards a complete and detail-preserved salient object detection. IEEE Transactions on Multimedia pp. 1–15 (2023). https://doi.org/10.1109/TMM.2023.3325731
35. Zhang, J., Xie, J., Barnes, N., Li, P.: Learning generative vision transformer with energy-based latent space for saliency prediction. In: 2021 Conference on Neural Information Processing Systems (2021)
36. Zhu, X., Hu, H., Lin, S., Dai, J.: Deformable convnets v2: More deformable, better results. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9308–9316 (2019)
37. Zhuge, M., Fan, D.P., Liu, N., Zhang, D., Xu, D., Shao, L.: Salient object detection via integrity learning. IEEE Transactions on Pattern Analysis and Machine Intelligence (2022)

FL-PSeC: Federated Learning-Pseudo Labeled Medical Image Segmentation with Personalized Class Balancing Semi-supervised Approach

Ishu Priya(✉) and C. Krishna Mohan

Indian Institute of Technology, Hyderabad, Hyderabad, India
cs24resch01001@iith.ac.in, ckm@cse.iith.ac.in

Abstract. Medical image segmentation plays a pivotal role in modern healthcare applications, significantly assisting healthcare professionals in making accurate diagnosis. However, limited annotated data hinders the development of robust medical image segmentation models due to strict patient privacy regulations. Semi-supervised learning offers a solution by leveraging both labeled and unlabeled data, enabling models to learn from larger datasets while capitalizing on limited labeled data. Despite its benefits, semi-supervised learning continues to face concerns related to data privacy. Federated learning (FL) addresses these concerns by enabling multiple parties to collaboratively train a model without sharing their local data. As the need for data-efficient and privacy-preserving machine learning techniques grows, semi-supervised federated learning emerges as a potential paradigm to tackle this issue. Our work underscores the importance of semi-supervised federated learning approaches that can effectively leverage both labeled and unlabeled data distributed across multiple sites. However, non-Independent and Identically Distributed (non-IID) data and class data imbalance remain significant challenges in semi-supervised federated learning, hindering overall performance. To address these issues, we propose the Federated Learning-Pseudo labeled Segmentation with Class balance (FL-PSeC) framework, which focuses on tackling class imbalance across clients' local data by assigning higher weightage to underrepresented classes when generating annotations from unlabeled data. This strategy not only improves the IID nature of the data but also leads to enhanced global model performance. Our experiments on HAM10000, BUS, BUSIS, and UDIAT datasets demonstrate the efficacy of FL-PSeC in adapting to a class imbalance in FL, outperforming existing methods.

Keywords: Federated Learning · Semi-Supervised Learning · Psuedo-Labeling · Annotated Data · Medical Image Segmentation · Class Imbalance

1 Introduction

Medical image segmentation [23] is a pivotal task within image content analysis, aiding computer-aided diagnosis by identifying the type of lesion and pinpointing precise areas of interest [16]. Image segmentation, being a significant area of medical imaging, has evolved into a highly active and evolving challenge in the field of medical research. It involves dividing an image into distinct areas on the basis of the relationship between target and non-target regions, facilitating identifying specific objects within the image [7]. Over the years, deep learning has significantly emerged dominant in this domain and has been applied in a wide range of medical image analysis tasks such as skin lesion segmentation [5], lung nodule simulation [11] etc.

Accuracy and robustness of image segmentation become more crucial in the case of medical images, which play a vital role in medical diagnosis and treatment planning [17]. Deep Learning models generally require a substantial amount of training data for better generalization and performance [9]. However, in fully-supervised semantic segmentation models, the optimal scenario involves acquiring pixel-level annotated images from a wide range of diverse sources. However, in real-world scenarios, acquiring pixel-level annotation for all the images from various sources is often unattainable in medical image segmentation. The critical reasons for this are: firstly, the annotation cost and, secondly, the stringent protocols for sensitive data sharing.

Numerous ML algorithms require substantial amounts of data, yet data are distributed across various organizations while safeguarded by privacy regulations [14]. Medical institutions house vast amounts of sensitive patient information. FL is essential for medical image analysis, allowing collaborative model training across institutions while preserving data privacy. This approach ensures accurate segmentation models without the need for centralized data sharing [22], as depicted in Fig. 1.

In the age of AI, collaborative learning through data sharing among institutions offers efficient model building. However, centralized data storage, processing, and analysis pose challenges. FL offers an alternative: models are brought to data sources for in-house training, enabling collaborative learning without centralizing datasets. Participants train models locally and share parameters for aggregation on a centralized server, ensuring data privacy while managing data differently [22].

There are three different types of ML frameworks based on the amount of annotations in the training data, as shown in Fig. 2.

Supervised ML: Supervised ML plays a pivotal role in medical imaging by utilizing labeled data to train algorithms for specific tasks such as medical image classification, where the model learns to categorize images into different classes as depicted in Fig. 2a. In this approach, to indicate the presence or absence of particular features or conditions, medical images are annotated or labeled by experts. These labeled images are then used to train the algorithm to recognize patterns and make predictions on new, unseen images [19].

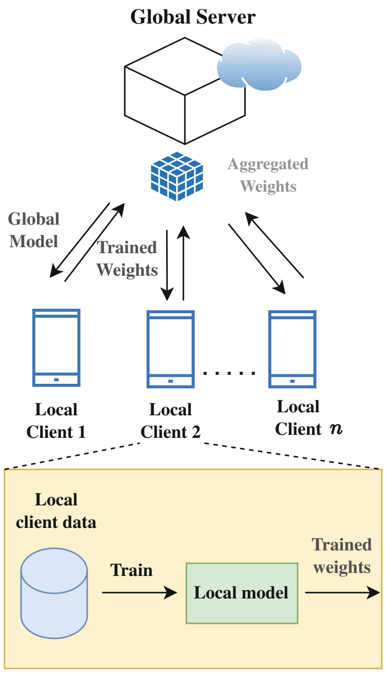


Fig. 1. Federated learning framework

Unsupervised ML: Unsupervised ML, depicted in Fig. 2b, in medical imaging is valuable for discovering patterns, similarities, and structures within unlabeled data. This approach is not dependent on predefined labels but aims to uncover inherent relationships and groupings within the images [2]. Clustering, a key application of unsupervised learning, is particularly used to identify groups of images comprising similar characteristics. In medical imaging, this often implies discovering different subtypes of diseases or conditions on the basis of shared features among images. Additionally, unsupervised learning can be used for anomaly detection, where the algorithm learns the normal patterns within the data and accordingly identifies any deviations as potential abnormalities.

Semi-supervised ML: Semi-supervised ML balances the use of labeled and unlabeled data, offering advantages in scenarios where obtaining large amounts of labeled data is challenging, such as medical image data [4,13]. This approach combines the benefits of supervised learning with the ability to leverage the abundance of unlabeled data. The model is trained using a small set of labeled images and a larger set of unlabeled images, as shown in Fig. 2c. By learning from both data types, the model improves its ability to generalize and thus, make accurate predictions on new, unseen images [19,35].

Annotating medical images with pixel-level precision requires significant expertise, making it a challenging and resource-intensive task. This poses a

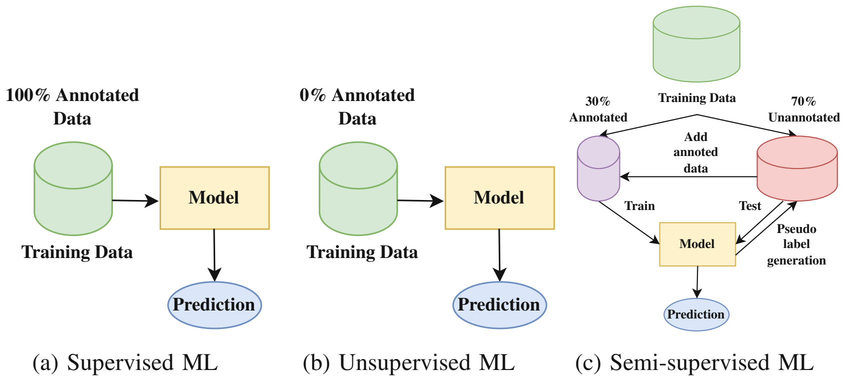


Fig. 2. Overview of different types of machine learning frameworks based on the availability of annotated data.

substantial hurdle for many medical institutions, resulting in a limited availability of images with detailed annotations. Consequently, most datasets in the medical field are comprised of images that are either not labeled or have incomplete annotations. This lack of detailed annotations hampers the development of accurate and reliable segmentation models, crucial for disease diagnosis and treatment planning tasks [15,32]. The need for semi-supervised FL arises from the dual challenges of limited annotated data and stringent privacy regulations in medical image analysis. In a typical FL setup, pixel-level annotations, which are necessary for image segmentation model training, might not be accessible to all the local clients. This poses a challenge in effectively utilizing weakly-labeled and unlabeled data for model learning. To address this, certain federated semi-supervised learning methods require clients to exchange additional information while learning from weakly labeled and unlabeled training data [26].

Addressing these challenges, a critical need arises for a clinical approach to maximizing available supervision. Institutions collaborate to enhance segmentation models using their expertise and annotations without directly sharing patient data. This ensures privacy while advancing accurate medical image segmentation. Current studies often overlook the variations in the availability of supervision, which is common in clinical settings where different clients possess varying levels of labeled data [26].

We propose a novel framework called Federated Learning-Pseudo labeled Segmentation with Class balance (FL-PSeC) that tackles the issue of class imbalance within local datasets of participating devices by prioritizing underrepresented classes during pseudo-label generation. This approach addresses the varying class distributions across devices, promoting uniformity and reducing bias in locally trained models. By assigning increased weights to minority classes, FL-PSeC aims to minimize class imbalance, which is crucial for preventing biased models favoring majority classes. Additionally, it improves the similarity of local data distributions, enhancing the Independent and Identically Distributed (IID) nature of the data. This step ensures the global model can be generalized effectively across diverse datasets. With reduced class imbalance and improved IIDness, FL-PSeC fosters stable model convergence during training to enhance overall model accuracy. The technique facilitates a global model with better performance across all classes and devices. The summary of our main contributions is given below:

- Our proposed framework, FL-PSeC, addresses concerns related to data privacy by allowing multiple parties to collaboratively train a model without sharing their local data along with tackling the challenge of limited annotated data in medical image segmentation
- FL-PSeC introduces a novel strategy to handle class imbalance across clients' local data by assigning higher weightage to underrepresented classes during pseudo-label generation. This approach enhances the IID nature of the data and leads to improved global model performance.
- Our experiments on HAM10000, BUS, BUSIS, and UDIAT datasets demonstrate the efficacy of FL-PSeC in adapting to class imbalance in federated learning and outperforming existing methods, which validates the practicality of your proposed approach.

2 Related Work

In this section, we discuss the state-of-the-art approaches related to federated semi-supervised learning and medical image segmentation. Table 1 compares key characteristics and attributes between existing State-of-the-art methods and our method in the federated semi-supervised learning. Wicaksana *et al.* [26] proposed a label-agnostic unified FL framework to carry out medical image segmentation. It integrates mixed image labels, including pixel-level, bounding box, and image-level class labels, from local clients to update the federated model. Yang *et al.* [28] proposed a solution to the challenge of domain shift in COVID region segmentation using federated and semi-supervised learning. Leveraging a multi-national database of 1704 chest CT scans from three countries, the study addresses the variability in data and annotations by introducing a novel federated semi-supervised learning technique. Further, Das *et al.* [6] introduces the MedPFL framework to analyze and mitigate privacy risks in federated learning (FL) applied to medical image analysis. It highlights the significant privacy risks in FL for processing medical images, demonstrating vulnerabilities to privacy attacks.

Limitations of Existing Works: Many previous studies have applied semi-supervised learning techniques to enhance model performance but have not adequately addressed the non-Independent and Identically Distributed (non-IID) data and class imbalance challenges inherent in federated learning. Furthermore, existing works often neglect data privacy concerns in the medical domain, focusing primarily on medical image segmentation, as demonstrated in Table 1. To address these limitations, our work aims to improve medical image segmentation in federated learning by addressing class imbalance issues through personalized weighting of pseudo labels in a semi-supervised learning approach on a per-client basis. This approach enables the problem to be solved locally, ultimately leading to improved global performance while preserving data privacy.

Table 1. Comparison of key characteristics and attributes between existing State-of-the-art methods and our method in the federated semi-supervised learning.

S.No.	Author, Year	Federated learning	Semi-supervised ML	Medical data	Privacy-preserving techniques	Data distribution heterogeneity	Handling Class imbalance issue
1	Gharibi ***et al.*** [8], 2021	✓	×	×	✓	×	×
2	Korkmaz ***et al.*** [12], 2022	✓	×	✓	✓	×	×
3	Zhang ***et al.*** [33], 2022	✓	×	✓	✓	✓	×
4	Hidayat ***et al.*** [10], 2023	✓	×	×	✓	✓	×
5	Das ***et al.*** [6], 2023	✓	×	✓	×	×	×
6	Yang ***et al.*** [29], 2023	×	✓	×	×	×	×
7	Chen ***et al.*** [4], 2024	×	✓	✓	×	×	×
8	Schmarje ***et al.*** [21], 2021	×	✓	✓	×	×	×
9	Sahu ***et al.*** [20], 2023	×	✓	✓	×	×	×
10	Wang ***et al.*** [25], 2024	×	✓	✓	×	×	×
11	**FL-PSeC (Ours)**, 2024	✓	✓	✓	✓	✓	✓

3 Background

This section defines the concepts related to this work to understand the rest of the paper.

- **Privacy-Preserving Techniques:** In the context of FL, privacy-preserving techniques such as federated averaging with local differential privacy [10] and privacy-preserving aggregation [8,12] are crucial for ensuring the confidentiality and security of sensitive data stored on various devices or servers.
- **Data Distribution Heterogeneity:** It refers to the variation in the underlying data distributions across different sources or domains in ML tasks. Overcoming this challenge is crucial for models to generalize effectively across diverse datasets, ensuring reliable real-world performance [31,33].
- **Handling Class Imbalance Issue:** Class imbalance arises where the number of instances belonging to one class is significantly lower than those of the other classes in a dataset. This can lead to biased model performance, as algorithms tend to favor the majority class.

4 Proposed Methodology

In the realm of FL, the challenges of class imbalance and non-IID data distributions persist as formidable obstacles. These issues need to be addressed using methodologies that mitigate data biases as well as enhance the generalization capabilities of the global model across diverse local datasets.

Our FL-PSeC method provides a comprehensive approach to tackle the challenge of unlabeled imbalanced data in FL by strategically integrating class balancing techniques and data distribution alignment strategies as shown in Fig. 3. We now present a detailed overview of the proposed methodology.

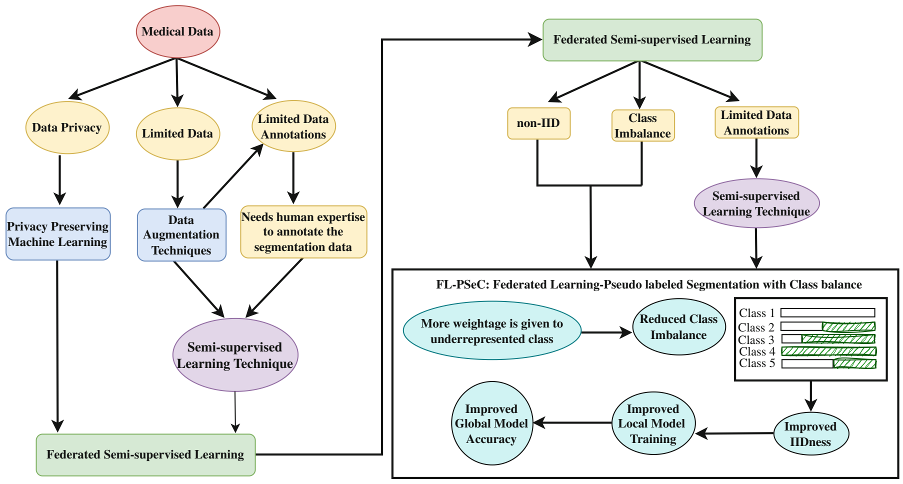

Medical Data
Data Privacy
Limited Data
Limited Data Annotations
Privacy Preserving Machine Learning
Data Augmentation Techniques
Needs human expertise to annotate the segmentation data
Semi-supervised Learning Technique
Federated Semi-supervised Learning
Federated Semi-supervised Learning
non-IID
Class Imbalance
Limited Data Annotations
Semi-supervised Learning Technique
FL-PSeC: Federated Learning-Pseudo labeled Segmentation with Class balance
More weightage is given to underrepresented class
Reduced Class Imbalance
Class 1
Class 2
Class 3
Class 4
Class 5
Improved IIDness
Improved Local Model Training
Improved Global Model Accuracy

Fig. 3. Challenges in the medical image segmentation and the proposed FL-PSeC framework.

In our FL setup, we consider a total of n clients, each client C_k $(k \in [1, n])$ holds private local data D_k with a size of $|D_k| = l_k$. This data is partially annotated, allowing us to split it into two parts: annotated data D_k^A and unannotated data D_k^U, where $|D_k^A| \ll |D_k^U|$. We assume that the data distribution among clients is non-IID, meaning the data can vary significantly between clients. To ensure a valid FL protocol, we enforce a minimum threshold λ for the number of samples per client by randomly splitting the annotated training dataset among clients such that $l_k \geq \lambda$. We follow a similar process for the unannotated data, ensuring class imbalance and non-IIDness. The goal of the central server is to learn a global weight parameter w_G by iteratively aggregating model updates from clients within n-dimensional parameter space and evaluating the results on the remaining test data. For each client C_k, we define three vectors:

– $\Gamma \in \mathbb{R}^c$ represents the class-wise contribution in the local client training data.

- $\Omega \in \mathbb{R}^c$ represents the size of the pseudo labels generated by our semi-supervised learning algorithm for each client.
- $\mathcal{W} \in \mathbb{R}^c$ represents the FL-PSeC class balance weighting vector.

Here, c represents the number of classes in the dataset. Further details on our FL-PSeC model will be provided in subsequent sections.

4.1 FL-PSeC Psuedo Label Generation

Inspired by Fedmix [26], our approach aims to optimize the utilization of all available data and enhance the reliability of local model updates through pseudo label generation. To achieve this, we leverage various levels of labels to extract valuable information even in the absence of pixel-level annotations. By incorporating multiple labelling types, such as bounding box annotations or image-level class labels, we amplify and refine useful signals obtained from pseudo supervision. This multi-label strategy enables effective model training in scenarios where fine-grained annotations are scarce, ensuring that critical information is captured and incorporated into the learning process. As a result, the model's overall global performance is improved by optimizing the available data and

Algorithm 1 Proposed FL-PSeC method

```
Input: $G_t$ - Global model; $l$ - loss function; $\eta$ - learning rate;
Output: $\Delta C_t^i$ - Local model updates of the Client(s)
 1: $n \leftarrow$ total number of clients
 2: procedure CLIENT_EXEC($n$, $G_t$)
 3:     for $k = 1$ to $n$ do                          ▷ Loop through number of total clients
 4:         Replace local model with global model: $C_t^k \leftarrow G_t$
 5:         $\Gamma^k = [\gamma_1^k, \ldots, \gamma_c^k]$, s.t. $\gamma_i^k = \frac{|cl_i^k|}{\sum_{i=1}^{c} |cl_i^k|}$, for $i \in [1, c]$
 6:         $\Omega^k = [\omega_1^k, \ldots, \omega_c^k]$, s.t. $\omega_i^k = |\hat{cl}_i^k|$, for $i \in [1, c]$
 7:         $\mathbf{W}^k = [w_1^k, \ldots, w_c^k]$, s.t. $w_i^k = 1 - \gamma_i^k$, for $i \in [1, c]$
 8:         $\Omega^k = \Omega^k * \mathcal{W}^k$
 9:         $D_k^A \leftarrow \Omega^k$
10:         for $b = 1$ to batches in $D_k^A$ do      ▷ Loop through batches in client's local annotated dataset
11:             $C_{t+1} = C_{t+1} - \eta \Delta l(C_t, b)$     ▷ train local model for E local epochs
12:             Calculate local client's update: $\Delta C_{t+1} = C_{t+1} - C_t$
13:     Scale up updates: $\Delta C_{t+1}^i = \alpha_i \Delta C_{t+1}^i$
14:     return local updates $\Delta C_{t+1}^i$ to central server
15: procedure MAIN( )
16:     for $t = 1$ to $T$ do
17:         Share $G_t$ to all the clients                    ▷ Server execution
18:         Receive updates from clients: $\Delta C_{t+1}^i$
19:         Perform model averaging according to $w_{t+1}^G = w_t^G + \sum_{n \in N} \alpha_n \Delta C_{t+1}^n$
20:         Update the global model ($G_t$): $G_{t+1}$
21:         Perform global model testing on $D_{test}$ data
22:         Client_exec($n$, $G_t$)                ▷ Function call for client execution
```

mitigating the reliance on pixel-level annotations. This method ultimately contributes to the enhanced quality of learned representations, as demonstrated by [26].

Based on the cross-pseudo supervision approach described in [3], FedMix trains two differently initialized models, $F_1(.)$ and $F_2(.)$, co-supervising each other with pseudo labels when pixel-level labels are not available. The training image X is input to both models to generate pseudo labels Y_1 and Y_2 respectively. These pseudo labels are then refined and denoted as $\hat{Y}_1$ and $\hat{Y}_2$ which are used for training each local client. The refinement strategies [26] for different types of labels are as follows:

- **Pixel-Level Labels:** Since pixel-level labels are directly available, no pseudo labels are generated. Therefore, $\hat{Y}_1 = \hat{Y}_2 = Y_{gt}$ represents the ground truth labels.
- **Bounding Box Labels:** For each pair of predictions, $Y_1 = F_1(X)$ and $Y_2 = F_2(X)$ refinement is performed based on the corresponding bounding box label. Specifically, $\hat{Y}_1 = Y_1 * Y_{bbox}$ and $\hat{Y}_2 = Y_2 * Y_{bbox}$
- **Image-Level Class Labels:** Each pair of predictions is refined using the image-level label Y_{img}. This results in $\hat{Y}_1 = Y_1 * Y_{img}$ and $\hat{Y}_2 = Y_2 * Y_{img}$. This refinement helps filter out images that correspond solely to the background class, treating them as outliers and excluding them from the training process.
- **No Labels (Unsupervised):** In the absence of any supervision, the predictions are used directly, setting $Y_1 = \hat{Y}_1$ and $Y_2 = \hat{Y}_2$

The pseudo labels are then taken into a Ω vector such that for each client C_k we calculate

$$\Omega^k = [\omega_1^k, \ldots, \omega_c^k], \text{s.t. } \omega_i^k = |\hat{cl}_i^k|, \text{for } i \in [1, c],$$

where $|\hat{cl}_i^k|$ is the number of pseudo labels generated by semi-supervised learning method for i^{th} class of the k^{th} client.

4.2 Personalized Class Balancing in FL-PSeC

First, we calculate the class-wise contribution in the local client (C_k) training data by determining the values of the Γ vector:

$$\Gamma^k = [\gamma_1^k, \ldots, \gamma_c^k], \text{s.t. } \gamma_i^k = \frac{|cl_i^k|}{\sum_{i=1}^{c} |cl_i^k|}, \text{for } i \in [1, c],$$

where $|cl_i^k|$ represents the number of samples of i^{th} class in the k^{th} client. Next, we compute the FL-PSeC class balance weighting vector $\mathbf{W}^k$ for client C_k as follows:

$$\mathbf{W}^k = [w_1^k, \ldots, w_c^k], \text{s.t. } w_i^k = 1 - \gamma_i^k, \text{for } i \in [1, c].$$

Finally, we update the semi-supervised generated pseudo labels vector Ω^k corresponding to client C_k using the calculated weighting vector:

$$\Omega^k = \Omega^k * \mathcal{W}^k.$$

The updated Ω_k pseudo labels are then added to the annotated data D_k^A of the local client C_k for further local model training in the next communication round. An essential aspect of our approach is the dynamic nature of the pseudo label generation process within the semi-supervised learning algorithm. During each communication round, the vector Ω is updated with newly generated pseudo labels. Our FL-PSeC method adaptively updates the weighting vector $\mathcal{W}$ based on these changes, promoting better class balance and enhancing the overall effectiveness of the pseudo labeling process. This iterative refinement contributes to the improved performance of the federated learning system.

5 Experimental Results

Dataset: We have performed the experiments on two segmentation tasks. One for skin lesion segmentation and other one for breast tumor segmentation. The dataset used to perform the skin lesion segmentation experiments is HAM10000 [24] comprising a total of 10015 dermatoscopic images of skin lesions images collected from a total of 7479 patients from four different sources, distributed among four different clients, namely Rosendahl and Vidir (Old, Modern and Molemax).

Three publicly available datasets - BUSIS [27] comprising 562 cancer images, BUS [1] which consists 133 healthy and 647 cancerous images, and UDIAT [30] comprises 163 cancerous images - are utilized for the breast tumor segmentation task. Each dataset is regarded as a different client resulting in a three client FL setting.

Evaluation Metrics and Implementation Details: To assess the image segmentation task, we have used Dice coefficient (DC) [26]. The chosen baseline segmentation model combines UNet architecture and group normalization (group norm). Four types of class labels are included in our experiments: pixel-level, bounding box, image level, and unlabeled. The model training process utilizes the Adam optimizer for 300 epochs. The training is conducted with a batch size of 16 and a learning rate 0.001. We have followed the experimental settings as done in [26]. The model is trained for four different clients for skin lesion segmentation task and three different clients for breast tumor segmentation. In every training iteration, each client involved utilizes its locally accessible data to conduct local model updates for a single training epoch.

Table 2 presents the comparative results on the HAM10000 dataset following the specified setup from [26]. Notably, in the [L, L, L, L] supervision category, the FedAdaptAgg method exhibits superior performance. As our approach falls under the semi-supervised FL framework, we have included the remaining methods LL, FedAvg, and FedAdaptAgg in this configuration, as depicted in Table 2. Additionally, in the [U, U, L, U]supervision category, our proposed

FL-PSeC method outperforms existing methods such as FedRGD and FedST. The iterative augmentation of class-balanced data evidently fosters IIDness in the dataset, consequently enhancing overall global performance. Furthermore, in the [B, B, L, B] supervision scenario, we note a similar trend. However, the [B, B, L, B] setup presents a mixed supervision context in terms of labels compared to the other two, showcasing our method's efficiency in achieving better performance in terms of DC%.

Table 3 displays the comparative results on the BUS, BUSIS, and UDIAT datasets following the specified configuration. Notably, in the [L, L, L] supervision category, both the LL and FedAdaptAgg methods demonstrate superior performance across various client types. Similar to above experiment, we have included the remaining methods LL, FedAvg, and FedAdaptAgg in this analysis, as our approach operates within the semi-supervised FL framework (Table 3). Furthermore, in the [U, U, L] supervision category, our proposed FL-PSeC method surpasses existing approaches such as FedRGD and FedST following a similar trend of HAM10000 dataset. However, the [B, B, L] setup presents a mixed supervision context compared to the others, highlighting our method's efficiency in achieving superior global performance in terms of DC%.

Table 2. Quantitative results for skin lesion segmentation on HAM10000 dataset across various learning frameworks and supervision settings. The first column indicates client types considered in the experiments: U for clients with unlabeled data, B for clients with bounding box labels, and L for clients with pixel-level labels. C_i represents client's index.

Supervision [C1, C2, C3, C4]	Method	DC%				
		C1	C2	C3	C4	Avg.
[L,L,L,L]	LL [26]	88.7 ± 0.4	92.9 ± 0.4	93.0 ± 0.3	92.6 ± 0.3	91.8 ± 0.3
	FedAvg [18]	88.4 ± 0.6	92.6 ± 0.5	**95.7 ± 0.8**	95.0 ± 0.1	92.9 ± 0.4
	FedAdaptAgg [26]	**89.9 ± 0.5**	**94.1 ± 0.6**	95.2 ± 0.8	**95.0 ± 0.1**	**93.6 ± 0.2**
[U,U,L,U]	LL	74.9 ± 1.0	73.0 ± 1.2	93.0 ± 1.2	91.1 ± 1.7	83.0 ± 0.7
	FedRGD [34]	71.7 ± 2.6	70.1 ± 2.8	92.1 ± 3.0	89.8 ± 0.5	80.9 ± 2.2
	FedST [28]	77.2 ± 0.6	75.2 ± 1.1	94.7 ± 0.6	91.3 ± 0.6	84.6 ± 0.3
	FedMix [26]	77.9 ± 0.6	**75.8 ± 0.5**	95.4 ± **0.2**	92.0 ± 0.5	85.3 ± 0.4
	FL-PSeC (Ours)	**78.1 ± 0.4**	75.6 ± 0.2	**95.6 ± 0.7**	**92.3 ± 1.2**	**85.4 ± 2.5**
[B,B,L,B]	FedMix	85.7 ± 0.2	89.4 ± 0.7	**96.2 ± 0.3**	93.5 ± 0.3	91.2 ± 0.1
	FL-PSeC (Ours)	**85.9 ± 1.1**	**89.7 ± 0.3**	95.7 ± 0.8	**93.7 ± 0.4**	**91.3 ± 0.7**

Table 3. Quantitative results for breast tumor segmentation task across various learning frameworks and supervision settings. The first column indicates client types considered in the experiments: U for clients with unlabeled data, B for clients with bounding box labels, and L for clients with pixel-level labels. C_i represents client's index.

Supervision [C1, C2, C3]	Method	DC%			
		C1	C2	C3	Avg.
[L,L,L]	LL [26]	**78.3 ± 2.8**	**91.6 ± 0.7**	83.4 ± 2.3	84.4 ± 1.7
	FedAvg [18]	77.3 ± 2.3	91.3 ± 0.7	85.6 ± 1.3	84.7 ± 0.7
	FedAdaptAgg [26]	77.9 ± 1.7	90.9 ± 1.0	**86.6 ± 1.1**	**85.1 ± 0.4**
[U,U,L]	LL	65.9 ± 3.1	85.4 ± 1.1	83.4 ± 2.3	78.2 ± 1.4
	FedRGD [34]	60.6 ± 4.4	80.7 ± 3.2	83.5 ± 4.4	74.9 ± 4.0
	FedST [28]	67.3 ± 1.7	85.0 ± 1.6	83.2 ± 3.5	78.5 ± 1.9
	FedMix [26]	**68.3 ± 1.8**	87.8 ± 0.6	85.6 ± 1.9	80.6 ± 0.7
	FL-PSeC (Ours)	68.1 ± 0.2	**88.1 ± 0.4**	**85.9 ± 1.2**	**80.7 ± 0.6**
[B,B,L]	FedMix	70.0 ± 0.6	**89.9 ± 0.4**	89.6 ± 0.1	83.2 ± 0.3
	FL-PSeC (Ours)	**70.3 ± 0.7**	89.7 ± 1.1	**89.9 ± 0.1**	**83.3 ± 1.9**

6 Conclusion

In this paper, we emphasize the importance of semi-supervised learning in advancing accurate medical image segmentation models within the constraints of limited annotated data due to stringent patient privacy regulations. Our proposed framework, FL-PSeC, offers a novel solution to tackle challenges such as non-IID data and class data imbalance in the semi-supervised federated learning setting. By assigning higher weightage to underrepresented classes during pseudo-label generation, FL-PSeC effectively addresses these issues, resulting in an improved IID nature of the data and significantly enhancing model performance. FL-PSeC demonstrates the capability to adapt to varying levels of label availability across different client sites, ensuring efficient utilization of the available data for creating more accurate and generalizable segmentation models. The conducted experiments substantiate the effectiveness of FL-PSeC in outperforming state-of-the-art methods by a considerable margin, revealing its potential to revolutionize the field of medical image segmentation. Overall, FL-PSeC presents a promising direction for advancing semi-supervised federated learning in medical image segmentation, paving the way for more efficient and privacy-preserving healthcare AI systems. In the future, we plan to explore the possibility of extending our approach to multi-modal medical imaging, investigating its applicability to other healthcare tasks, and incorporating advanced techniques for handling data heterogeneity and privacy-preservation in federated learning.

References

1. Al-Dhabyani, W., Gomaa, M., Khaled, H., Fahmy, A.: Dataset of breast ultrasound images. Data Brief **28**, 104863 (2020)
2. Bhimavarapu, U., Chintalapudi, N., Battineni, G.: Brain tumor detection and categorization with segmentation of improved unsupervised clustering approach and machine learning classifier. Bioengineering **11**(3), 266 (2024)
3. Chen, X., Yuan, Y., Zeng, G., Wang, J.: Semi-supervised semantic segmentation with cross pseudo supervision. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2613–2622 (2021). https://doi.org/10.1109/CVPR46437.2021.00264
4. Chen, Y., Mancini, M., Zhu, X., Akata, Z.: Semi-supervised and unsupervised deep visual learning: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **46**(3), 1327–1347 (2022)
5. Codella, N.C.F., et al.: Skin lesion analysis toward melanoma detection: a challenge at the 2017 international symposium on biomedical imaging (ISBI), hosted by the international skin imaging collaboration (ISIC). In: 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), pp. 168–172 (2018). https://doi.org/10.1109/ISBI.2018.8363547
6. Das, B.C., Amini, M.H., Wu, Y.: Privacy risks analysis and mitigation in federated learning for medical images. In: 2023 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 1870–1873. IEEE (2023)
7. Gao, Q.: Medical image segmentation algorithm based on deep learning and convolutional neural network. In: 2023 IEEE International Conference on Integrated Circuits and Communication Systems (ICICACS), pp. 01–05 (2023). https://doi.org/10.1109/ICICACS57338.2023.10100142
8. Gharibi, M., Bhagavan, S., Rao, P.: FederatedTree: a secure serverless algorithm for federated learning to reduce data leakage. In: 2021 IEEE International Conference on Big Data (Big Data), pp. 4078–4083. IEEE (2021)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
10. Hidayat, M.A., Nakamura, Y., Arakawa, Y.: Efficient and secure: privacy-preserving federated learning for resource-constrained devices. In: 2023 24th IEEE International Conference on Mobile Data Management (MDM), pp. 184–187. IEEE (2023)
11. Jin, D., Xu, Z., Tang, Y., Harrison, A., Mollura, D.: CT-realistic lung nodule simulation from 3D conditional generative adversarial networks for robust lung segmentation. In: Frangi, A., Schnabel, J., Davatzikos, C., Alberola-Lopez, C., Fichtinger, G. (eds.) 21st International Conference, Granada, Spain, 16–20 September 2018, Proceedings, Part II, pp. 732–740. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00934-2_81
12. Korkmaz, A., Alhonainy, A., Rao, P.: An evaluation of federated learning techniques for secure and privacy-preserving machine learning on medical datasets. In: 2022 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), pp. 1–7. IEEE (2022)
13. Li, M.X., Liu, Y., Liu, Q., Chen, S.L., Chen, F., Yin, X.C.: Semi-supervised fine-grained classification with web data via noisy sample selection. In: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 5024–5030. IEEE (2022)

14. Li, Q., et al.: A survey on federated learning systems: vision, hype and reality for data privacy and protection. IEEE Trans. Knowl. Data Eng. **35**(4), 3347–3366 (2023). https://doi.org/10.1109/TKDE.2021.3124599
15. Li, Y., Luo, L., Lin, H., Chen, H., Heng, P.A.: Dual-consistency semi-supervised learning with uncertainty quantification for Covid-19 lesion segmentation from CT images. In: de Bruijne, M., et al. (eds.) Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part II 24, pp. 199–209. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-87196-3_19
16. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3431–3440 (2015). https://doi.org/10.1109/CVPR.2015.7298965
17. Masood, S., Sharif, M., Masood, A., Mussarat, Y., Raza, M.: A survey on medical image segmentation. Curr. Med. Imaging Rev. **11**, 3–14 (2015). https://doi.org/10.2174/157340561101150423103441
18. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)
19. Reiß, S., Seibold, C., Freytag, A., Rodner, E., Stiefelhagen, R.: Every annotation counts: multi-label deep supervision for medical image segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9532–9542 (2021)
20. Sahu, H., Kashyap, R., Dewangan, B.K.: Hybrid deep learning based semi-supervised model for medical imaging. In: 2022 OPJU International Technology Conference on Emerging Technologies for Sustainable Development (OTCON), pp. 1–6. IEEE (2023)
21. Schmarje, L., Santarossa, M., Schröder, S.M., Koch, R.: A survey on semi-, self-and unsupervised learning for image classification. IEEE Access **9**, 82146–82168 (2021)
22. Sohan, M.F., Basalamah, A.: A systematic review on federated learning in medical image analysis. IEEE Access **11**, 28628–28644 (2023). https://doi.org/10.1109/ACCESS.2023.3260027
23. Tran, M., Vo-Ho, V.K., Le, N.T.: 3DConvCaps: 3DUnet with convolutional capsule encoder for medical image segmentation. In: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 4392–4398. IEEE (2022)
24. Tschandl, P., Rosendahl, C., Kittler, H.: The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. Sci. Data **5**(1), 1–9 (2018)
25. Wang, C., et al.: An interpretable and accurate deep-learning diagnosis framework modelled with fully and semi-supervised reciprocal learning. IEEE Trans. Med. Imaging **43**(1), 392–404 (2023)
26. Wicaksana, J., et al.: FedMix: mixed supervised federated learning for medical image segmentation. IEEE Trans. Med. Imaging (2022)
27. Xian, M., et al.: A benchmark for breast ultrasound image segmentation (BUSIS). Infinite Study (2018)
28. Yang, D., et al.: Federated semi-supervised learning for Covid region segmentation in chest CT using multi-national data from China, Italy, Japan. Med. Image Anal. **70**, 101992 (2021)
29. Yang, X., Song, Z., King, I., Xu, Z.: A survey on deep semi-supervised learning. IEEE Trans. Knowl. Data Eng. **35**(9), 8934–8954 (2022)
30. Yap, M.H., et al.: Automated breast ultrasound lesions detection using convolutional neural networks. IEEE J. Biomed. Health Inform. **22**(4), 1218–1226 (2017)

31. Zaccone, R., Rizzardi, A., Caldarola, D., Ciccone, M., Caputo, B.: Speeding up heterogeneous federated learning with sequentially trained superclients. In: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 3376–3382. IEEE (2022)
32. Zhang, D., Guo, G., Zeng, W., Li, L., Han, J.: Generalized weakly supervised object localization. IEEE Trans. Neural Netw. Learn. Syst. **35**(4), 5395–5406 (2022)
33. Zhang, R., Li, H., Hao, M., Chen, H., Zhang, Y.: Secure feature selection for vertical federated learning in eHealth systems. In: ICC 2022-IEEE International Conference on Communications, pp. 1257–1262. IEEE (2022)
34. Zhang, Z., et al.: Improving semi-supervised federated learning by reducing the gradient diversity of models. In: 2021 IEEE International Conference on Big Data (Big Data), pp. 1214–1225. IEEE (2021)
35. Zhao, Y.X., Zhang, Y.M., Song, M., Liu, C.L.: Multi-view semi-supervised 3D whole brain segmentation with a self-ensemble network. In: Shen, D., et al. (eds.) Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, 13–17 October 2019, Proceedings, Part III 22, pp. 256–265. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32248-9_29

Improved Hypergraph Laplacian Based Semi-supervised Support Vector Machine

Reshma Rastogi(✉) and Dev Nirwal

Machine Learning and Statistical Inference (MLSI) Lab, Department of Computer Science, South Asian University, Delhi, India
reshma.khemchandani@sau.ac.in, devnirwal@students.sau.ac.in

Abstract. Many real-life systems are framed as networks which are used to model interactions between complex entities. However, in many scenarios these interactions may not be pairwise, rather should described as higher-order interactions. For such a scenario Hypergraphs are preferred rather than simple Laplacian. Therefore, to decide whether Laplacian, Hypergraph or both could be used for the given system could lead to a problem which needs to be addressed properly. In this paper, we present a semi-supervised framework which considers weighted combination of Hypergraph and Laplacian information for pattern classification. Numerical experiments on seven binary, five multi-class and four multilabel datasets along with MNIST-fashion dataset validate the efficacy of proposed algorithm.

Keywords: Support Vector Machines · Hypergraphs · Laplacian · Semi-supervised

1 Introduction

Support vector machine (SVM) proposed by Vapnik [1], a parallel plane classifier, is a supervised learning algorithm which among all possible supporting hyper-planes identifies a maximum margin classifier for the pattern classification problem. Depending upon the structure of data, either hard margin or soft margin SVMs are obtained via solving a Quadratic Programming Problem (QPP) in the dual space. There exist many non-parallel classifier like generalized eigen value proximal support vector machine (GEPSVM) [2] and Twin Support Vector Machines (TWSVM) [3]. GEPSVM tends to solve pair of eigenvalue-eigenvector problem and Twin support vector machines [3] which is similar to SVM but solves pair of QPPs each smaller in size as compared to QPP solved via SVM, which makes it four times faster than SVM. Although SVM based classifiers offers several benefits, the primary drawback is the requirement for a substantial volume of labelled training data. However, in practice, it is challenging to obtain adequately labelled data; which is true for any real life application.

Semi-supervised learning (SSL) combines both supervised as well as unsupervised learning to solve many real-life problems. SSL has its applications in the field of computer vision, medical field, and text classification, where it is challenging to have labelled data. Underlying distribution of unlabelled data captures structural information which in turn enhance performance of a learning algorithm [4]. Numerous techniques exist for SSL, such as graph-based approaches, co-training, self-training, and Gaussian mixture models. In many SSL methods, the most prevalent assumptions about data distributions are the Cluster and Manifold assumption. In the cluster assumption, we can divide the data into several discrete clusters; the data points within a cluster are more likely to have the same output labels. Algorithms such as Transductive SVM [5] or S^{3}VM [6] uses the cluster assumption to find the optimal hyperplane in the SSL setting. Data points located on a manifold structure has significantly lower dimension compare to the input space [7].

Laplacian Support Vector Machine (LapSVM) [8] algorithm tries to learn the underlying manifold structure from the unlabel data and combines it with traditional SVM. A regularization term was considered to keep the decision function and manifold structure smooth, which avoids overfitting. On the lines of LapSVM [8], several other variations such as Laplacian p-norm proximal SVM (LapPPSVM) [9], Laplacian twin support vector machine (LapTSVM) [10], Laplacian least square twin support vector machine (Lap-LSTSVM) [11], and LapLSTSVM [12], exists. All the aforementioned models considers pair-wise relation of the data while constructing a graph Laplacian which may in general not true for real-world applications. The relation between objects may be binary and complex in reality, which is the biggest drawback of LapSVM models. Hypergraph based SVM(HGSVM) is a graph-regularized manifold learning algorithm for semi-supervised setting discussed in [13].

Motivated from the above studies and in order to capture the underlying structure of the data, we propose a new method, termed as, Improved Hypergraph Laplacian Support Vector Machine (IHLSVM) in which we construct both graph Laplacian, Hyper-graph and consider weighted combination of both them to find a new regularization term which further enhance the performance of semi-supervised SVM. Thus, we first construct a hyper-graph [14] that looks up to higher and complex relations using the hypergraph similarity matrix of the original data and Laplacian that considers the pair-wise relation between the data and finally the propose model that considers the weighted combination of both. Unlike LapSVM, HGSVM constructs only hypergraph and thus ignores the scenario wherein data may have pair-wise relation which otherwise is captured in graph Laplacian and thus can lead to overfitting. On the other hand, IHLSVM considers weighted combination of Laplacian as well as Hypergraph and based on underlying data distribution the weight parameter is tuned.

2 Preliminaries

2.1 Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm with many applications in image classification, regression, etc. SVM attempt to minimize empirical error while maximizing the distance between two bounding hyperplanes. Given a set of training data $X = \{(x_1, y_1)...(x_l, y_l)\}$ where $x_i \in R^n$, $y_i \in \{-1, +1\}, i = 1, 2, ..l$. The SVM solves following quadratic programming problem

$$\min_{w,b,\xi} \frac{1}{2}w^T w + C\sum_{i=1}^{l}\xi_i \tag{1}$$
$$\text{s.t. } y_i(w^T x_i + b) \geq 1 - \xi_i$$
$$\xi_i \geq 0, i = 1, 2, ..., l$$

where the slack variable (error term) $\xi = (\xi_1, \xi_2, ..., \xi_l)^T$ corresponds to misclassification error in training data. The separating hyperplane is given by

$$w^T x + b = 0 \tag{2}$$

where $w \in R^n$ and $b \in R$.

2.2 Manifold Learning

The $X = \{(x_i, y_i)|x_i \in R^n\}$ be the set of l-labelled data where n is number of dimension for the training data and $y_i \in \{+1, -1\}$ and the set $U = \{x_i, i = l + 1, l+2.., m\}$ is set of unlabel data. Hence, the dataset $S = X \cup U$, where the first l data points are labelled, and the rest of the u data points are unlabel. $K \in R^{m \times m}$ is the kernel matrix for m points corresponding to $k(x_i, x_j)$ $k : X \times X \to R$. L is the graph Laplacian associated with given data S and is given by

$$L = D - W \tag{3}$$

where W is the adjacency matrix and D is the diagonal matrix with the degree of each node ($d_{ii} = \sum_{j=1}^{m} w_{ij},\ w_{ij}$ is the entry from matrix W). The target function that the learning algorithm must estimate is denoted as f, such that $f : X \to R$ where f is the vector of the n values on training data.

Manifold learning [15] is a non-linear dimensionality reduction technique in which we assume the high dimensional data generally lie on a lower dimension manifold. The semi-supervised manifold regularization framework can be defined as

$$f^* = \underset{f \in H_k}{\operatorname{argmin}} \sum_{i=1}^{l} V(x_i, y_i, f) + \delta_A ||f||_A^2 + \delta_I ||f||_I^2 \tag{4}$$

where $V(x_i, y_i, f)$ is the loss function, $||f||_A^2$ is the ambient norm. i.e., the norm of the function f in the Reproducing Kernel Hilbert Space(RKHS) enforces a smoothness condition on the possible outcome. $||f||_I^2$ is the intrinsic norm of the function f in the low dimensional manifold, which enforces a smoothness along the samples and is learned from the combination of labelled and unlabel data. The tradeoff parameter δ_A and δ_I are hyper-parameters that control the weight of ambient and intrinsic norm. According to the Representer theorem [8] the function f^* will expand in m terms as:

$$f^* = \sum_{i=1}^{m} \alpha_i^* k(x_i, x) \tag{5}$$

2.3 Laplacian Support Vector Machine

LapSVM [8] is based on the principle of manifold learning and using representer theorem solve the following optimization similar to (4).

$$\min_{\alpha \in R^m, \xi \in R^l} \sum_{i=1}^{l} \xi_i + \gamma_A \alpha^T K \alpha + \gamma_I \alpha^T KLK\alpha$$

$$s.t.\ y_i(\sum_{j=1}^{n} \alpha_i k(x_i, x_j) + b) \geq 1 - \xi_i, \quad i = 1, 2..., l$$

$$\xi_i \geq 0, i = 1, 2..., l$$

where L is the regularised Graph Laplacian of the dataset derived in (3).

2.4 Hypergraph Support Vector Machine

LapSVM algorithm assumes a pairwise association between the samples which in general doesn't seem to hold in practice. Graph Laplacian is unable to capture multivariate and higher-order relationships between the samples. Authors in [13] express high order relationship with Hypergraph which captures association between two or more than two vertices. To construct a Hypergraph Support Vector Machine (HGSVM) [16] problem, the main task, like any other graph-regularized manifold learning problem, is to find a matrix representation of the graph. Authors in [16] have constructed Laplacian regularized Hypergraph matrix H [16] (similar to L matrix in LapSVM). Given a graph $G = (V, E)$, $V = \{v_1, v_2, ..., v_n\}$ and $E = \{e_1, e_2, ..., e_n\}$ are the vertex and hyper-edges set. We can define a vertex-edge matrix F such that

$$F = f(v, e) = \begin{cases} 1, & v \in e \\ 0, & v \notin e \end{cases} \tag{6}$$

The degree of hyperedge $\delta(e)$ is defined as the number of vertices a hyperedge contains.

$$\delta(e) = \sum_{v \in e} f(v, e)$$

$$d(v) = \sum_{v \in e, e \in E} w(e) = \sum_{e \in E} w(e) f(v, e)$$

The following formula defines the hyperedge weight $w(e)$:

$$w(e) = \frac{1}{\delta(e)(\delta(e)-1)} \sum_{\{v_i,v_j\}\in e} exp\left(-\frac{||x_i - x_j||^2}{\mu}\right)$$

Similar to a simple graph, the Hypergraph's Laplacian matrix can be defined as [17]:

$$H = D_v - FWD_e^{-1}F^T \tag{7}$$

where D_v, D_e, W are the diagonal matrices composed of $d(v), \delta(e)$ and $w(e)$ respectively.
According to Zhou's [18], the Laplacian regularized Hypergraph matrix can also be defined as

$$H = I - D_v^{-\frac{1}{2}}FWD_e^{-1}F^TD_v^{-\frac{1}{2}} \tag{8}$$

Thus, the formulation of HGSVM is quadratic programming problem given by

$$\min_{\alpha\in R^n,\xi\in R^l} \sum_{i=1}^{l}\xi_i + \gamma_A\alpha^TK\alpha + \gamma_I\alpha^TKHK\alpha$$

$$s.t.\ y_i(\sum_{j=1}^{n}\alpha_i k(x_i,x_j)+b) \geq 1-\xi_i, \quad i = 1,2...,l$$

$$\xi_i \geq 0, i = 1,2...,l$$

where H is the Laplacian regularized Hyper-Graph of the dataset as defined in (8). The solution of the aforementioned optimization is obtained by solving one quadratic programming problem followed by solving system of linear equations which is not scalable.

3 Proposed Method

3.1 Improved Hypergraph Laplacian SVM (IHLSVM)

In this section, we first propose to solve HGSVM i.e. the aforementioned quadratic programming problem in primal space by replacing hinge loss with square-hinged loss function, which in turn lead to solving following unconstrained optimization problem in the primal space.

$$\begin{aligned}\min_{\alpha\in R^m,b\in R}\frac{1}{2}(\sum_{i=1}^{l} max(1-y_i(k_i^T\alpha+b),0)^2 + \delta_A\alpha^TK\alpha + \\ \delta_I(\alpha^TK+1^Tb)H(K\alpha+1b))\end{aligned} \tag{9}$$

Since the above optimization is a convex problem, Newton's method is used to find the optimal solution whose complexity is $O(n^3)$.

The graph Laplacian L carries the pairwise relationship between the training instances, while the multivariate and complex relations are carried by the

Hyper-graph Laplacian matrix H. However it is difficult to identify which representation would be optimal for the given dataset. Therefore, in this section, we define improved Hypergraph Laplacian matrix which is weighted linear combination of Laplacian as well as Hypergraph Laplacian and solve it as unconstrained optimization problem which can be further solved in primal space using Preconditioned Conjugate Gradient method [19] whose order of complexity is $O(kn^2)$, where k is number of iterations and n is the number of training samples.

$$IHL = \lambda L + (1 - \lambda)H \tag{10}$$

In the IHL matrix, L is from (3) and H matrix is from (8). λ is the hyperparameter which work as a balancing factor between both of them.

Thus considering squared-hinged loss function

$$V(x_i, y_i, f) = max(1 - y_i(k_i^T \alpha + b), 0)^2,$$

regularizer term as $||f||_A^2 = \alpha^T K \alpha$, $|f||_I^2 = \alpha^T K(IHL)K\alpha$. IHLSVM problem would solve the following unconstrained optimization problem

$$\begin{aligned} \min_{\alpha \in R^m, b \in R} \frac{1}{2}(\sum_{i=1}^{l} max(1 - y_i(k_i^T \alpha + b), 0)^2 + \delta_A \alpha^T K \alpha + \\ \delta_I(\alpha^T K + 1^T b)(IHL)(K\alpha + 1b)) \end{aligned} \tag{11}$$

We can solve (11) using Preconditioned Conjugate Gradient (PCG) method [19] . The detailed algorithm is mentioned below.

Algorithm 1 Algorithm for IHLSVM

```
Dataset; $S = X \cup U = \{x_t, y_t\}_{t=1}^{l} \cup \{x_t\}_{t=l+1}^{l+u}$, $X$ denotes a set of $l$ labeled data,
$U$ denotes set of $u$ unlabeled examples and parameters set $\sigma, \gamma_1, \gamma_2, \lambda$; classsifier
$f(x) = sign(\sum_{i=1}^{l+u} \alpha_i^* K(x, x_i)$
Initialisation: Calculate Graph Laplacian $L$;
Calculate the Laplacian Regularised Hypergraph $H = I - D_v^{-\frac{1}{2}} FWD_e^{-1}F^T D_v^{-\frac{1}{2}}$;
Calculate Improved Hypergraph Laplacian matrix $IHL = \lambda L + (1 - \lambda)H$;
Find kernel function $K(x_i, x_j)$ ;
Calculate $\alpha^*$ by solving primal problem (11) by using PCG Method;
```

3.2 Multi-category and Multi-label Classification

In semi-supervised learning, multi-class or multi-label classification is a challenging task due to the unknown geometrical structure of the data. There are many problems like speech recognition [20], face recognition [21], and many staged diseases which are multi-class problems. There are many methods to solve multi-class like OVO(One-vs-One) [22], OVR(One-vs-Rest) [23] or DAG for multi-class classification [24], and Binary Relevance for multi-label classification. In this paper, we have adopted the OVR classification strategy for multiclass and binary relevance [25] for multi-label classification.

4 Experiments and Results

We ran a large set of experiments to analyse the Improved Hypergraph Laplacian Support Vector Machine (IHLSVM) proposed solution strategies and compared them with other algorithms like LapSVM and HGSVM. In this section, we discuss data sets, our experimental protocol and the details of parameter selection strategy. We perform numerical experiments on UCI datasets. All the experiments are performed in MATLAB R2023a on a PC with AMD Ryzen 7 5800U with Radeon Graphics 1.90 GHz and 8GB RAM.

4.1 UCI Datasets

We ran tests on 12 UCI [26] benchmark datasets to look into the classification performance of our proposed approach with the square-hinge loss version of LapSVM and HGSVM. Table 1 presents specifics of the 12 UCI benchmark datasets and multilabel datasets[1]. Every dataset is standardised with mean zero and standard deviation 1.

Table 1. UCI datasets

Dataset	Size	Features	Classes
Australian	690	14	2
Breast Cancer	277	9	2
CMC	1473	9	2
German	1000	24	2
Hearts	270	13	2
Ionosphere	351	34	2
WDBC	569	30	2
Ecoli	3186	3	3
Dna	3186	180	3
Glass	214	9	7
Iris	150	4	3
Wine	178	13	3

[1] https://www.uco.es/kdis/mllresources.

Results for Binary Classification. We have compared IHLSVM with square-hinge loss version of LapSVM and HGSVM, on seven UCI binary datasets. We have use 10-fold cross-validation to report the accuracy of models, where two regularisation parameter have range $\{10^{-6}, 10^{-5}, ..., 1, 10, 100\}$ and kernel parameters for RBF have the range $[0, 1]$. Table 2 report the mean accuracy with standard deviation. We can observe that the classification performance of our

Table 2. Classification Accuracy Results for Binary Datasets

Dataset	Unlabelled	LapSVM	HGSVM	IHLSVM
Australian	0%	80.29 ±6.70%	80.00 ±6.62%	**80.87 ±6.10%**
	10%	81.01 ±5.48%	80.87 ±5.33%	**81.16 ±5.63%**
	30%	80.87 ±5.19%	81.01 ±5.18%	**81.45 ±4.20%**
	50%	81.30 ±4.50%	81.30 ±4.50%	**81.45 ±4.62%**
German	0%	71.90 ±4.70%	72.10 ±4.43%	**72.50 ±4.43%**
	10%	71.30 ±4.42%	71.30 ±4.57%	**71.80 ±4.44%**
	30%	72.00 ±3.50%	72.00 ±3.83%	**72.20 ±2.97%**
	50%	70.80 ±5.35%	71.00 ±4.99%	**71.10 ±4.89%**
Ionosphere	0%	94.32 ±3.24%	94.31 ±3.28%	**94.60 ±3.36%**
	10%	94.02 ±3.68%	**94.02 ±3.68%**	93.74 ±3.75%
	30%	92.04 ±3.69%	92.04 ±3.69%	**92.33 ±3.75%**
	50%	92.03 ±4.19%	92.03 ±4.60%	**92.32 ±4.45%**
Breast Cancer	0%	70.74 ±5.80%	**70.74 ±5.80%**	71.10 ±5.71%
	10%	71.14 ±6.20%	71.14 ±6.20%	**71.85 ±6.19%**
	30%	69.70 ±9.04%	63.92 ±7.91%	**70.04 ±7.73%**
	50%	71.85 ±3.64%	67.87 ±7.72%	**72.21 ±4.13%**
CMC	0%	66.05 ±4.04%	65.57 ±4.38%	**66.12 ±3.84%**
	10%	66.46 ±3.42%	66.19 ±3.48%	**66.52 ±3.90%**
	30%	65.92 ±3.72%	65.37 ±2.75%	**66.05 ±3.21%**
	50%	65.17 ±4.15%	65.24 ±3.84%	**65.44 ±4.81%**
Hearts-statlog	0%	76.30 ±8.04%	75.19 ±6.99%	**77.41 ±7.29%**
	10%	76.67 ±8.56%	75.93 ±8.42%	**78.89 ±8.38%**
	30%	79.26 ±5.58%	78.52 ±9.53%	**80.37 ±6.54%**
	50%	80.00 ±6.10%	80.37 ±5.80%	**81.48 ±7.20%**
WDBC	0%	96.14 ±3.77%	95.09 ±3.39%	**96.32 ±4.09%**
	10%	96.66 ±3.14%	95.08 ±3.18%	**96.84 ±2.96%**
	30%	96.66 ±2.40%	96.49 ±1.85%	**96.84 ±1.61%**
	50%	96.32 ±2.67%	96.49 ±2.48%	**97.02 ±2.20%**

Table 3. Classification Accuracy Results for Multi-class Datasets

Dataset	Unlabelled	LapSVM	HGSVM	IHLSVM
Ecoli	0%	85.71 ±3.76%	85.71 ±3.77%	**86.01 ±3.92%**
	10%	85.11 ±3.84%	84.52 ±3.77%	**85.71 ±3.77%**
	20%	85.71 ±3.90%	85.40 ±4.68%	**86.30 ±4.30%**
	30%	81.23 ±5.88%	81.23 ±6.51%	**83.32 ±3.76%**
Iris	0%	92.00 ±5.06%	93.33 ±5.27%	**94.67 ±5.06%**
	10%	92.67 ±5.48%	91.33 ±9.60%	**95.33 ±3.80%**
	20%	92.67 ±10.90%	90.00 ±8.16%	**94.67 ±5.06%**
	30%	92.67 ±7.96%	94.00 ±4.35%	**94.67 ±4.47%**
Wine	0%	98.30 ±1.55%	98.32 ±1.54%	**98.87 ±1.54%**
	10%	97.73 ±2.39%	97.75 ±1.26%	**98.30 ±1.55%**
	20%	98.86 ±1.56%	98.87 ±1.54%	**99.43 ±1.28%**
	30%	93.81 ±2.37%	93.25 ±1.56%	**93.83 ±1.21%**
Glass	0%	66.36 ±7.80%	66.38 ±6.97%	**67.30 ±8.66%**
	10%	64.99 ±5.92%	64.03 ±3.74%	**65.46 ±5.80%**
	20%	58.89 ±8.55%	57.96 ±8.75%	**60.29 ±8.81%**
	30%	53.28 ±6.90%	52.34 ±6.26%	**53.73 ±6.28%**
dna	0%	50.25 ±1.95%	50.22 ±1.96%	**50.38 ±1.93%**
	10%	50.22 ±1.90%	50.16 ±1.72%	**50.25 ±1.74%**
	20%	50.66 ±1.97%	50.56 ±1.78%	**50.75 ±1.87%**
	30%	51.07 ±1.84%	51.13 ±1.75%	**51.22 ±1.66%**

Table 4. Comparison of Results for MNIST Fashion

Unlabelled	LapSVM	HGSVM	IHLSVM
0 %	76.57± 0.79	76.48± 1.41	**76.89± 0.84**
10 %	**76.45± 0.84**	75.88± 0.81	76.41± 0.72
20 %	75.07± 0.98	75.38± 1.18	**75.43± 1.10**
30 %	74.55± 1.56	**74.75± 0.52**	74.70± 1.41
40 %	73.91± 0.96	73.21± 1.82	**74.16± 0.98**
50 %	72.62± 1.28	71.82± 1.23	**72.88± 1.25**

proposed IHLSVM is better than that of the other two methods on all binary datasets. In addition, we investigate the classification performances of IHLSVM under $0\%, 10\%, 30\%, 50\%$ unlabelled data used while training.

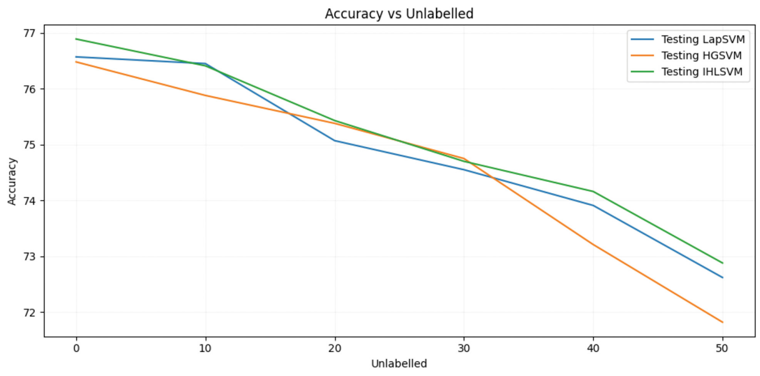


Fig. 1. Results for MNIST Fashion

Table 5. Description of Multi-label Datasets

Dataset	Instances	Features	Labels	Card	Domain
Birds	645	260	19	1.014	Audio
CAL500	502	68	174	26.044	Music
Flags	194	19	7	3.392	Image
Enron	1702	1001	53	3.378	text

Table 6. Comparison of Results on Multi-label Datasets

Dataset	Metric	LapSVM	HGSVM	IHLSVM	LapSVM	HGSVM	IHLSVM
Unlabelled			0%			10%	
Birds	HL(↓)	0.0458	0.0468	**0.0452**	0.0458	0.0474	**0.0455**
	F1(↑)	0.6158	0.6095	**0.62**	0.6097	0.5927	**0.6131**
	RL(↓)	0.1429	0.1524	**0.1429**	**0.1167**	0.1362	0.1262
	AP(↑)	0.7675	**0.7669**	0.7632	0.7908	**0.8069**	0.7901
	AUC(↑)	0.6726	0.6731	**0.6743**	**0.6494**	0.6382	0.6481
Flags	HL(↓)	0.3084	0.3077	**0.3062**	0.3056	0.3027	**0.3019**
	F1(↑)	0.664	0.6641	**0.6671**	0.6768	0.6794	**0.6821**
	RL(↓)	0.0571	0.0571	**0.0571**	0.1143	0.1143	**0.1143**
	AP(↑)	**0.7014**	0.6979	0.6983	**0.6881**	0.6849	0.6803
	AUC(↑)	0.5773	**0.5789**	0.5787	0.5928	0.5952	**0.5987**
Cal500	HL(↓)	**0.1386**	0.1388	0.1387	0.1381	0.138	**0.138**
	F1(↑)	0.3086	**0.3101**	0.3096	0.3054	**0.3072**	0.3071
	RL(↓)	**0.0959**	0.1055	0.1055	**0.1243**	0.1339	0.1255
	AP(↑)	**0.9901**	0.9886	0.9892	**0.991**	0.9898	0.9902
	AUC(↑)	0.4712	**0.4715**	0.4714	0.4718	**0.4726**	0.4725
Enron	HL(↓)	0.0583	0.0585	**0.0583**	0.0586	0.058	**0.0574**
	F1(↑)	0.4266	**0.4282**	0.4272	0.3592	0.4405	**0.441**
	RL(↓)	0.2412	0.2538	**0.2412**	0.1764	0.1724	**0.1724**
	AP(↑)	**0.9304**	0.9295	0.9295	**0.9311**	0.9291	0.9289
	AUC(↑)	**0.486**	0.4854	0.4858	0.4868	0.4871	**0.4889**

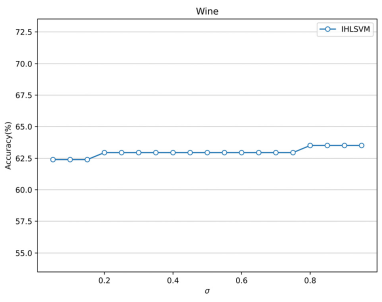
Table 7. Comparison of results on Multilabel datasets

Dataset Unlabelled	Metric	LapSVM	HGSVM 30%	IHLSVM	LapSVM	HGSVM 50%	IHLSVM
Birds	HL(↓)	0.0477	0.048	**0.047**	0.0531	0.0537	**0.053**
	F1(↑)	0.5519	0.5581	**0.5635**	**0.498**	0.4945	0.4959
	RL(↓)	0.0581	0.0771	**0.0386**	0.0776	0.0781	**0.0581**
	AP(↑)	0.8153	**0.8353**	0.8137	0.8432	**0.8698**	0.8486
	AUC(↑)	0.624	0.6201	**0.6265**	**0.5873**	0.5801	0.5854
Flags	HL(↓)	0.3175	0.3168	**0.3146**	0.318	0.3225	**0.3158**
	F1(↑)	0.6672	0.6689	**0.6705**	0.6713	0.6671	**0.6748**
	RL(↓)	0.0571	0.0571	**0.0571**	0.0286	0.0286	**0.0286**
	AP(↑)	**0.7097**	0.7084	0.7055	0.7189	**0.727**	0.7175
	AUC(↑)	0.5774	0.578	**0.5792**	0.5745	0.5701	**0.5759**
Cal500	HL(↓)	**0.139**	0.1395	0.1395	0.14	**0.1394**	0.1401
	F1(↑)	0.3101	0.3097	**0.311**	0.3064	0.308	**0.3091**
	RL(↓)	**0.0754**	0.0827	0.079	0.0685	**0.0553**	0.0649
	AP(↑)	**0.9916**	0.9912	0.9909	0.9919	**0.9923**	0.9915
	AUC(↑)	**0.4737**	0.4731	0.4733	0.4752	**0.4764**	0.4755
Enron	HL(↓)	0.0583	0.0587	**0.0583**	0.0611	**0.0584**	0.0586
	F1(↑)	0.3252	**0.4251**	0.3258	0.1326	**0.3815**	0.2721
	RL(↓)	0.216	0.2161	**0.2119**	**0.1341**	0.1425	0.1425
	AP(↑)	0.942	**0.9424**	0.9418	0.9478	**0.9459**	0.9438
	AUC(↑)	**0.4863**	0.4848	0.4862	0.4694	0.4745	**0.476**

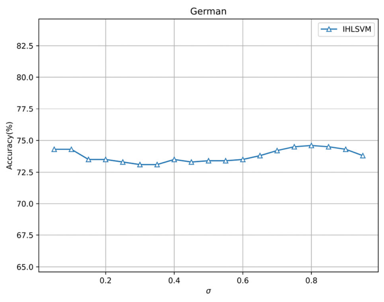
Results for Multi-class Classification. We investigate the classification performance of our proposed method on five UCI multi-class datasets and have evaluate its performance on Fashion MNIST data. We have used OVR [23] strategy and compare it with square-hinge loss version of LapSVM and HGSVM. We are using 10-fold cross-validation to report the classification accuracy. Each experiment is also repeated ten times. Table 3 reports the mean accuracy and its standard deviation. We can observe that the classification performance of our proposed IHLSVM is better than that of the other two methods on all multi-class datasets. In addition, we investigate the classification performances of IHLSVM under $0\%, 10\%, 20\%, 30\%$ unlabelled data used while training.

Fashion-MNIST[2] is a dataset of Zalando's article images-consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28×28 grayscale image, associated with a label from 10 classes. Zalando intends Fashion-MNIST to serve as a direct drop-in replacement for the original MNIST dataset for benchmarking machine learning algorithms. It shares the same image size and structure of training and testing splits. We investigate the classification performances of IHLSVM under $0\%, 10\%, 20\%, 30\%40\%, 50\%$ unlabelled data in the training dataset. Mean and standard deviation are reported in Table 4 and Fig. 1 illustrates the performance for various percentage of unlabelled data. Figure 2 reports ablation study to discuss performance of proposed algorithm (IHLSVM) for different values of hyperparamters i.e. kernel parameter (σ), δ_A, δ_I, λ.

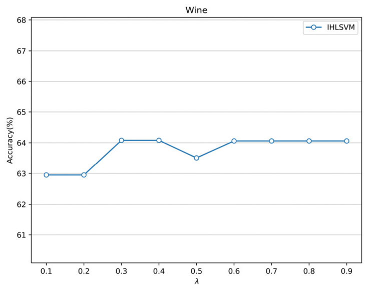
2 https://tech.zalando.comt.



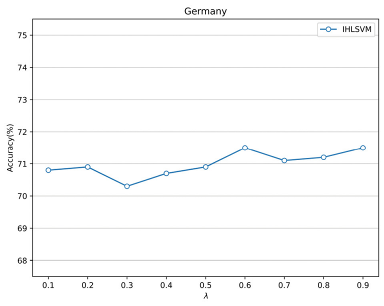
(a) Kernel Sensitivity Analysis for Wine Dataset



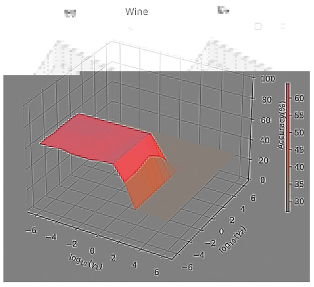
(b) Kernel Sensitivity Analysis for German Dataset



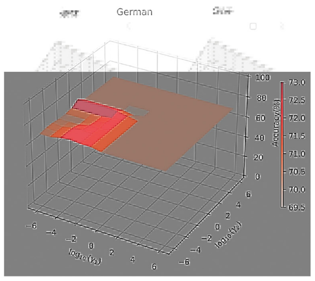
(c) Lambda Sensitivity Analysis for Wine Dataset



(d) Lambda Sensitivity Analysis for German Dataset



(e) δ_A and δ_I Sensitivity Analysis for Wine Dataset



(f) δ_A and δ_I Sensitivity Analysis for German Dataset

Fig. 2. Sensitivity Analysis

Results for Multi-label Classification. We investigate the classification performance of our proposed method along with square hinge loss version of LapSVM and HGSVM on four multi-label datasets i.e. Birds, Cal500, Flags and Enron discussed in Table 5. We have used the five evaluation criteria, Hamming Loss, Average Precision, Ranking Loss, AUC and F1 Score, to compare the performance of multi-label classification algorithms and results are discussed in Table 6- Table 7 for performances of IHLSVM under $0\%, 10\%, 30, 50\%$ unlabelled data used while training.

5 Conclusions

This paper proposes a novel semi-supervised method called improved hypergraph regularized semi-supervised support vector machine (IHLSVM), which explores the multivariate manifold structure embedded in data to establish a classifier. It considers weighted combination of graph Laplacian and Hypergraph to handle a scenario where it is difficult to identify individually which representation works well for the given data. To speed up the training procedure of the proposed algorithm, Square Hinge loss function is deployed and Preconditioned Conjugate Gradient method is used for solving the optimization problem. Results on binary, multi-category and multi-label datasets proves the efficacy of proposed algorithm. We have also shown its application on MNIST Fashion dataset. Ablation study supports the robustness of the proposed algorithm. Future line of work could be explore the possibility of establishing the relationship for multi-label datasets where along with feature, label space also exhibit higher order relationship.

References

1. V. N. Vapnik, "The Nature of Statistical Learning Theory," SpringerLink, 2000
2. Mangasarian, O.L., Wild, E.W.: Multisurface proximal support vector machine classification via generalized eigenvalues. IEEE Trans. Pattern Anal. Mach. Intell. **28**(1), 69–74 (2006)
3. Jayadeva, R. Khemchandani, and S. Chandra, "Twin support vector machines for pattern classification", IEEE Trans on Pattern Analysis and Machine Intelligence, vol. 29, pp. 905-910, 2007
4. Z. Song, X. Yang, Z. Xu, and I. King, "Graph-based Semi-supervised Learning: A Comprehensive Review", IEEE Transactions on Neural Networks and Learning Systems, vol. 34, no. 11, 2023
5. T. Joachims, "Transductive Learning via Spectral Graph Partitioning", in Proceedings of Twentieth International Conference on Machine Learning, pp. 290-297, 2003
6. Bennett, K., Demiriz, A.: Semi-Supervised Support Vector Machines. Adv. Neural. Inf. Process. Syst. **11**, 08 (2001)
7. Izenman, A.J.: Introduction to manifold learning. WIREs Comput. Stat. **4**(5), 439–446 (2012)
8. Melacci, S., Belkin, M.: Laplacian Support Vector Machines trained in the primal. J. Mach. Learn. Res. **12**, 1149–1184 (2011)

9. Tan, J., Zhen, L., Deng, N., Zhang, Z.: Laplacian p-norm proximal support vector machine for semi-supervised classification. Neurocomputing **144**, 151–158 (2014)
10. Qi, Z., Tian, Y., Shi, Y.: Laplacian twin support vector machine for semi-supervised classification. Neural Netw. **35**, 46–53 (2012)
11. Chen, W.-J., Shao, Y.-H., Deng, N.-Y., Feng, Z.-L.: Laplacian least squares twin support vector machine, for semi-supervised classification. Neurocomputing **145**, 465–476 (2014)
12. Khemchandani, R., Pal, A.: Multi-category laplacian least squares twin support vector machine. Appl. Intell. **45**(2), 458–474 (2016). https://doi.org/10.1007/s10489-016-0770-6
13. Sun, Y., Ding, S., Guo, L., Zhang, Z.: Hypergraph regularized semi-supervised support vector machine. Inf. Sci. **591**, 400–421 (2022)
14. D. Zhou, J. Huang, and B. Schölkopf, "Learning with Hypergraphs: Clustering, Classification, and Embedding", Advances in Neural Information Processing Systems, vol. 19, 2006
15. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples. J. Mach. Learn. Res. **7**(85), 2399–2434 (2006)
16. S. Saito, D. Mandic, and H. Suzuki, "Hypergraph p-Laplacian: A Differential Geometry View", Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1, 2018
17. S. Huang, M. Elhoseiny, A. Elgammal, and D. Yang, "Learning Hypergraph-regularized Attribute Predictors", arXiv [cs.CV]. 2015
18. D. Zhou, J. Huang, and B. Schölkopf, "Learning with Hypergraphs: Clustering, Classification, and Embedding", vol. 19, pp. 1601-1608, 2006
19. Melacci, S., Belkin, M.: Laplacian Support Vector Machines Trained in the Primal. J. Mach. Learn. Res. **12**, 1149–1184 (2011)
20. Jin, Y., Li, P.: performance and robustness of bio-inspired digital liquid state machines: A case study of speech recognition. Neurocomputing **226**, 145–160 (2017)
21. J. Wei, Z. Jian-qi, and Z. Xiang, "Face recognition method based on support vector machine and particle swarm optimization", Expert Syst. Appl., vol. 38, pp. 4390-4393, 04 2011
22. Galar, M., Fernández, A., Barrenechea, E., Herrera, F.: DRCW-OVO: Distance-based relative competence weighting combination for One-vs-One strategy in multi-class problems. Pattern Recogn. **48**(1), 28–42 (2015)
23. R. Rifkin and A. Klautau, "In Defense of One-Vs-All Classification", Journal of Machine Learning Research, vol. 5, pp. 101-141, 12 2004
24. Platt, J., Cristianini, N., Shawe-Taylor, J.: Large Margin DAGs for Multiclass Classification. Adv. Neural. Inf. Process. Syst. **12**, 03 (2000)
25. Zhang, M.-L., Li, Y.-K., Liu, X.-Y., Geng, X.: Binary relevance for multi-label learning: an overview. Front. Comp. Sci. **12**(2), 191–202 (2018). https://doi.org/10.1007/s11704-017-7031-7
26. M. Kelly, R. Longjohn, K. Nottingham, " The UCI Machine Learning Repository", https://archive.ics.uci.edu

Context Mutual Evolution Network for Weakly Supervised Surface Defect Detection

Xiaoheng Jiang[1,2,3], Penghui Xiao[1], Feng Yan[1], Yang Lu[1,2,3](✉), Shaohui Jin[1,2,3], and Mingliang Xu[1,2,3](✉)

[1] School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou 450001, China
{iexumingliang,ieylu}@zzu.edu.cn

[2] Engineering Research Center of Intelligent Swarm Systems, Ministry of Education, Zhengzhou 450001, China

[3] National Supercomputing Center in Zhengzhou, Zhengzhou 450001, China

Abstract. Weakly supervised object detection methods have achieved great success in natural scenes. However, they confront challenges such as small and weak appearances when detecting surface defects in industrial scenes. To address this problem, we propose a global-local Context Mutual Evolution Network (CMENet) for weakly supervised surface defect detection. CMENet is a dual-branch network consisting of a CNN branch and a Transformer branch, in which a context mutual evolution (CME) module is introduced in multiple blocks to enhance the feature representation ability of the network through the global and local information interaction. The CME module helps the network focus on defective areas and suppress background interference via a Feature Cluster Attention (FCA) module. The FCA adaptively selects some meaningful tokens as keys and values by a learnable clustering module to calculate attention. Extensive experiments on the DAGM 2007, KolektorSDD2, and Magnetic Tile defect datasets demonstrate that our method achieves promising performance compared with other state-of-the-art methods.

Keywords: Surface defect detection · weakly supervised detection · Context mutual evolution · feature cluster attention

1 Introduction

Surface defect detection is pivotal in industrial manufacturing and production processes. The majority of defect detection methods [29,32] based on deep learning rely heavily on abundant pixel-level annotations for model training. However, obtaining sufficient pixel-level labels consumes a significant amount of time

This work was supported in part by the National Natural Science Foundation of China under Grant 62172371, U21B2037, 62102370; in part by Natural Science Foundation of Henan Province under Grant 232300421093.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 256–270, 2025.
https://doi.org/10.1007/978-3-031-78192-6_17

and resources. In recent years, weakly supervised detection methods [9,12,27] have developed rapidly, which only use image-level classification labels to train a model and obtain segmentation results by the Class Activation Maps (CAMs) [39] method. However, existing weakly supervised detection methods mainly target natural scene images, exhibiting constraints in complex defect imagery. It is difficult for these methods to activate complete defect areas. Defects such as scratches in industrial products often resemble background texture, making it difficult for models to locate the area of the defect. Furthermore, the scale of defective areas in industrial products is significantly lower than that of objective in natural scenes. Figure 1 illustration demonstrates common challenges in surface defect detection, including scale variations, low contrast, and background interference.

To address the above problems, we propose a global-local Context Mutual Evolution Network(CMENet) for weakly supervised surface defect detection. CMENet adopts a dual-branch architecture consisting of the transformer branch and CNN branch, which are used to learn global features and local features, respectively. Global information is beneficial for activating complete defect objects, while local information is necessary for activating some defect details. To take full use of global and local features, the Context Mutual Evolution (CME) module is designed to suppress background noise and retain defect details. The CME module enhances the feature representation through the attention map that is generated by aggregating global and local features. In addition, the Feature Cluster Attention (FCA) module is introduced in the CME to strengthen the representation of global and local features. The FCA adaptively selects some meaningful tokens as keys and values by a learnable clustering module to calculate attention.

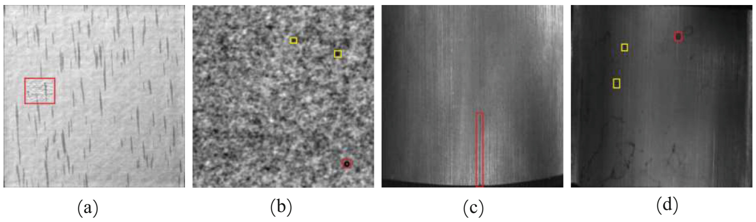


Fig. 1. Surface defect detection presents numerous challenges, including scale variations, low contrast, and background noise. (a) and (b) represent scale-varying defects. (c) and (d) represent low-contrast defects. (b) and (d) represent background noise. Defects and background noise are marked with red and yellow rectangles respectively, highlighting their location in the image. (Color figure online)

The main contributions of this paper are as follows:

1. We propose a global-local Context Mutual Evolution Network (CMENet) for weakly supervised surface defect detection, which learns both global and local features to activate complete defect objects and details.

2. A Context Mutual Evolution (CME) module is proposed to encourage co-evolution of global and local features. In the CME module, the FCA module is proposed to strengthen the representation further by adaptively selecting meaningful tokens for attention calculation.
3. The experiments demonstrate that the proposed weakly supervised defect detection method achieves state-of-the-art performance on three publicly available surface defect datasets (DAGM 2007, KolektorSDD2, and Magnetic tile).

2 Related Works

2.1 Surface Defect Detection

In recent years, the application of deep learning methods has surged in surface defect detection due to advancements in deep learning technology. These methods aim to accurately locate defects, such as object detection [24,37,38] and semantic segmentation [2,10,36]. For example, Cui et al. [6] introduced SDDNet, which propagated intricate details from lower to higher-level feature maps, augmenting small defect forecasting. Li et al. [18] utilized the encoder structure of U-Net to capture multi-scale features, promoting the development of end-to-end defect detection technology for smartphone light guide plates. Lu et al. [20] introduced a Transformer encoder-decoder architecture for end-to-end surface defect detection. Jiang et al. [17] introduced a feature fusion network guided by joint attention to address challenges such as substantial variations in defect scale, intricate backgrounds, and low contrast. However, it's important to note that each of these approaches necessitates costly annotations, whether bounding boxes or pixels.

2.2 Weakly Supervised Semantic Segmentation

The popularity of weakly supervised semantic segmentation is rising in academic and industrial settings. These approaches [21,22] enable pixel-level segmentation solely based on image-level labels. Current weakly supervised methods are usually based on CNN models, which are mainly inspired by the CAMs algorithm proposed by Zhou et al. [40]. Some research focuses on improving the generation process of CAMs to improve the accuracy of localization and segmentation tasks [16,28]. However, affected by the fixed receptive field in CNN, CAMs often only focus on the most discriminative object areas. Much research has been devoted to improving this. Qin et al. [27] introduced a new scheme for activation modulation and recalibration. This approach employed the spotlight branch and the compensation branch to generate weighted CAMs, thereby activating more complete CAMs. Chen et al. [4] introduced a method for locating seeds with structural awareness, enhancing seed region robustness, and developing a prototype model for background awareness to extract hierarchical features. However, they still cannot solve the problem of CNN having difficulty in capturing long-range features between pixels.

The Transformer [30] captures long-distance dependencies of sequences through a self-attention mechanism, which performs well in language sequences processing. However, as deep learning technology developed, researchers began to realize that the potential of the Transformer architecture was not limited to the language domain. Therefore, Vision Transformer (ViT) [8] was proposed, aiming to extend the powerful capabilities of Transformer to computer vision tasks. Recently, some researchers have migrated the Vit architecture to the field of weakly supervised semantic segmentation. Gao et al. [11] proposed TS-CAM, which fully utilizes the self-attention mechanism of Vision Transformer to extract long-distance dependencies and redistribute the semantic associations of patch labels so that each patch label can identify object categories based on relevant information. Xu et al. [34] proposed the MCTformer, which introduces a multi-class label Transformer to learn the interaction between class labels and patch labels by using multi-class labels. Zhu et al. [41] proposed WeakTr, which aims to generate class activation maps (CAMs) by adaptively estimating the importance of attention heads. However, these methods have shown excellent performance on natural images, and their performance is limited in industrial defect scenes.

2.3 Weakly Supervised Surface Defect Detection

Given the challenge of acquiring pixel-level annotations and the high cost associated with labeling surface defects in detection tasks, researchers have explored weakly supervised learning approaches to address this issue. Wu et al. [33] introduced a Siamese network to harmonize image-level and pixel-level supervision, and proposed three loss functions to enhance model accuracy in defect detection. He et al. [13] proposed an encoder to extract features and integrate two decoders to handle interconnected tasks. The primary task aimed to repair defects on textured surfaces, with the secondary task being the identification of the region of interest. Qi et al. [25] proposed two parallel learning modules and a differentiable level set module to achieve accurate defect detection through the interconnection of progressive learning strategies promoted by innovative loss functions. These methods are all based on CNNs, and the detection accuracy is not high due to the lack of global information.

3 Method

3.1 Overview Architecture

As shown in Fig. 2, the CMENet is a dual-branch architecture, consisting of a CNN branch and a Transformer branch. For the CNN branch, we adopt ResNet-50 as the backbone, which is divided into four blocks. The output features of CNN are denoted as $\{F_i\}_{i=1}^{4}$, with resolutions of $1/2^{i+1}$ corresponding to the input image. For the Transformer branch, DeiT-S is adopted as the backbone and is divided into four blocks. The output features of the transformer are denoted as $\{T_i\}_{i=1}^{4}$. For output features F_i and T_i of the same block, they are first fed into the CME block to enhance feature representation. This allows the network to effectively learn more discriminative global and local features, suppressing background noise and highlighting defect details. During the training, both branches use the cross-entropy function as the classification loss function, denoted as L^c_{cls} and L^t_{cls}, respectively.

For the transformer branch, attention maps from all transformer blocks are aggregated together to generate the final attention map. The attention map contains class-to-patch attention map A_{c2p} and patch-to-patch attention map A_{p2p}. In our paper, A_{p2p} is combined with A_{c2p} to generate Class Activation Maps (CAMs) of transformer features $\mathrm{M_T}$. The CAMs from the CNN branch are denoted as $\mathrm{M_C}$. Finally, the $\mathrm{M_C}$ and $\mathrm{M_T}$ are fused to generate the final object localization map.

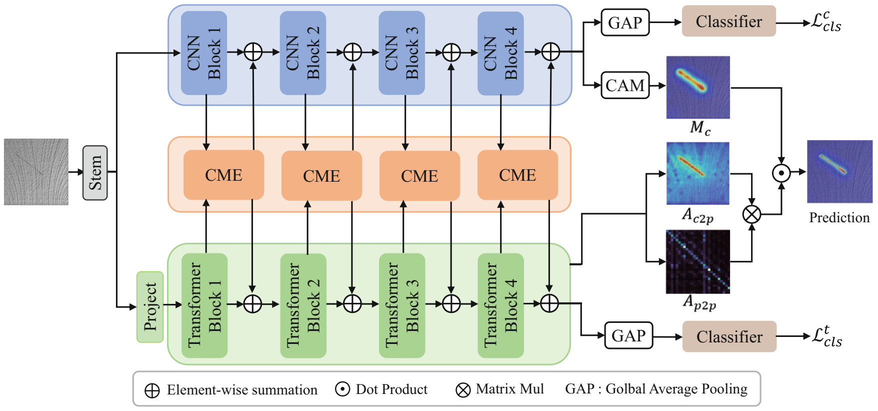


Fig. 2. The pipeline of proposed CMENet. CMENet includes a CNN branch, a Transformer branch, and Context Mutual Evolution (CME) modules. The CME module aims to simultaneously enhance the representation ability of global and local context features through information interaction.

3.2 Context Mutual Evolution Module

CNNs are superior at capturing local features, while transformers are excellent at learning global features. Defects tend to show complex or weak appearance. Therefore, it is difficult for an individual convolutional network or transformer network to capture enough defect information. To this end, the Context Mutual Evolution (CME) module is designed, as shown in Fig. 3, which enhances the representation of both local and global features by mutual learning.

Specifically, the global features (denoted as T_i) from the transformer branch are first reshaped from $\mathbb{R}^{N\times C}$ to $\mathbb{R}^{H\times W\times C}$, where $N = H\times W$. The local features (denoted as F_i) of the CNN branch and the reshaped global features are both fed into the convolution layer to adjust to the same channel number. Then,

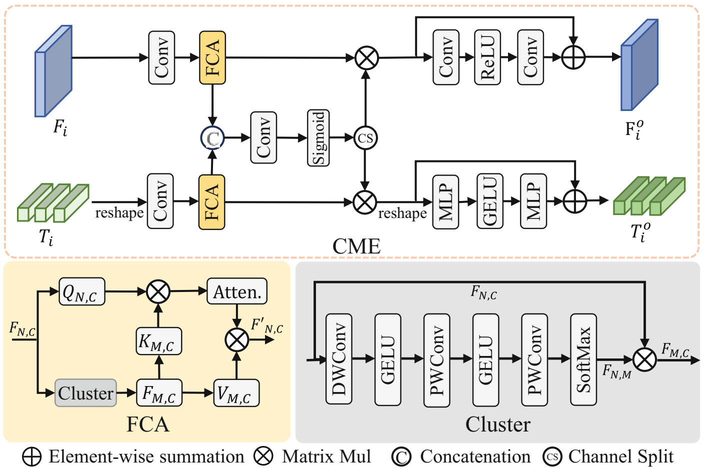


Fig. 3. The illustration of the Context Mutual Evolution (CME) module.

a Feature Cluster Attention (FCA) module is introduced to enhance feature representation. This process is expressed as follows:

$$F_i' = FCA(Conv(F_i)) \tag{1}$$

$$T_i' = FCA(Conv(reshape(T_i))) \tag{2}$$

In FCA, a cluster module is introduced to learn more expressive and meaningful visual features, while greatly reducing computation. To be specific, with input features $F_{N,C}$, we use depth-wise convolution (DWConv) and point-wise convolution (PWConv) to generate clustering features $F_{M,C}$. Mathematically, we have

$$F_{N,C} \xrightarrow[\text{k=7},s=1]{\text{DW}Conv+GELU} \cdot \xrightarrow[\text{k=1},s=1]{\text{PW}Conv+\text{GELU}} U_{N,C}, \tag{3}$$

$$U_{N,C} \xrightarrow[\text{k=1},s=1]{\text{PW}Conv} \cdot \xrightarrow{\text{Softmax}} F_{N,M}, \tag{4}$$

$$F_{M,C} = \text{LN}(F_{N,M}^T \cdot F_{N,C}), \tag{5}$$

The original features and the clustering features are mapped into $Q_{N,C}$, $K_{M,C}$, and $V_{M,C}$, respectively. Then the self-attention is computed as follows:

$$A_{N,M} = \text{Softmax}(\frac{Q_{N,C} \cdot K_{M,C}^T}{\sqrt{C}}), \tag{6}$$

$$Y_{N,C} = A_{N,M} \cdot V_{M,C}, \tag{7}$$

where $M = 100$ in the experiment. Compared with the traditional self-attention module, FCA has lower computation complexity.

Concatenating the enhanced features F_i' and S_i' along the channel dimension, they are then fed into a convolutional layer and a sigmoid function to generate the attention map$A \in \mathbb{R}^{H\times W\times 2C}$. A is split into A_1 and $A_2 \in \mathbb{R}^{H\times W\times C}$ along the channel dimension. A_1 and A_2 are used to weight CNN features F_i' and transformer features T_i', respectively. The weighted F_i' is fed into the convolution layers to generate the final output feature. The weighted T_i' is reshaped and fed into the Multi-Layer Perception (MLP) layers and generates the final output feature after the reshape operation. Mathematically, we have

$$A = Sigmoid((Conv([F_i', T_i']))) \tag{8}$$

$$F_i^o = Convs(F_i' \otimes A_1) + F_i' \tag{9}$$

$$T_i' = Reshape(T_i') \tag{10}$$

$$T_i^o = MLPs(T_i' \otimes A_2) + T_i' \tag{11}$$

where $\otimes$ represents element-wise multiplication operation

4 Experiments

4.1 Datasets

To validate the effectiveness of our defect detection method, the experiments are conducted on three publicly available defect datasets: the dataset DAGM 2007 dataset [15], the KolektorSDD2 dataset [3], and the Magnetic Tile dataset [14]. The detailed information of three datasets is shown in Table 1. The datasets consist of original images with pixel-level annotations. During the training, only image-level labels are used, while pixel-level labels are only used exclusively to evaluate test performance metrics.

4.2 Implementation Details

The network is implemented in PyTorch and runs on RTX 3090 GPU. The parameters of the CNN and Transformer branch are initialized by the pre-trained weights of ResNet50 and DeiT-S on ImageNet [7]. The network is trained on three datasets using the AdamW optimizer, respectively, with an initial learning rate of 0.0005, a batch size of 8, and a total epoch of 45. During the training process, the input image is first resized to 512×512. Then we randomly scale the image with scaling factors such as 0.5, 1, 1.5, and 2 as the input of the network to increase the diversity of the training samples During the test, the CAM is binarized into final segmentation results through a threshold value.

4.3 Evaluation Metrics

In the experiment, four common quantitative evaluation metrics are adopted: mean Intersection over Union (mIoU), Precision, Recall, and F1-measure. Higher mIoU, Precision, Recall, and F1 values mean better model performance.

Table 1. Detailed information of the public dataset. Pos and Neg denote positive (defective) and negative samples, respectively. Resolution indicates the size of the training image

Dataset	Resolution	Train Set		Test Set	
		Pos	Neg	Pos	Neg
DAGM 2007(1-6)	512×512	75	500	75	500
DAGM 2007(7-10)	512×512	150	1000	150	1000
KSDD2	630×230	246	2085	110	894
Magnetic tile	-	196	476	196	476

IoU is the ratio between the intersection and union of predicted and actual values for a specific category. **mIoU** is the mean of IoU of different categories, which is formulated as:

$$\mathrm{mIoU} = \sum_{k=1}^{c} \frac{\mathrm{TP}}{\mathrm{FP} + \mathrm{FN} + \mathrm{TP}} \tag{12}$$

where $c = 2$ in the experiment.

Precision indicates the ratio of correctly predicted positive samples to all predicted positive samples, which is defined as follows:

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{13}$$

Recall indicates the ratio of correctly identified positive samples to all positive samples. It is formally represented as:

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{14}$$

F1-measure represents a weighted mean of precision and recall values. It is formulated as:

$$\text{F1-measure} = \frac{2 * \mathrm{Precision} * \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}} \tag{15}$$

4.4 Experimental Results

To showcase the proposed method's effectiveness, we compare it with other state-of-the-art weakly supervised models for semantic segmentation, containing IRNet [1], AMR [26], MCTF [34], BAS [31], WeakT [41], SIPE [5] and Wave [35].

Results on DAGM 2007. The first three rows of Fig. 4 show varied detection outcomes from distinct methods on the DAGM 2007 dataset. Among these methods, SIPE, IRNet, and AMR demonstrate the ability to identify defects, but with low accuracy. In contrast, the network proposed by us achieves optimal results in challenging defect datasets. Furthermore, as shown in Table 2, our method surpasses other comparison methods and ranks the highest in terms of performance. For instance, compared to the MCTF method, our model shows improvements of 2.2%, 5.9%, 1.4%, and 1.4% in mIoU, precision, recall, and F1, respectively.

Table 2. Quantitative comparisons on DAGM 2007, KSDD2, and Magnetic tile in terms of mIoU, precision, recall, and F1. The optimal outcomes for each dataset are emphasized in bold font.

Dataset	Metrics	IRNet	AMR	MCTF	BAS	WeakT	SIPE	Wave	Ours
DAGM 2007	mIoU	.676	.689	.759	.681	.683	.657	.677	**.781**
	Pr	.443	.478	.643	.441	.485	.425	.450	**.702**
	Re	.663	.673	.738	.704	.619	.646	.654	**.752**
	F1	.521	.558	.687	.543	.544	.503	.533	**.701**
KSDD2	mIoU	.692	.689	.721	.694	.649	.604	.708	**.762**
	Pr	.561	.572	.657	.554	.535	.501	.597	**.727**
	Re	.601	.581	.626	.621	.465	.422	.629	**.676**
	F1	.581	.577	.646	.585	.477	.456	.613	**.691**
Magnetic tile	mIoU	.576	.596	.612	.582	.623	.551	.572	**.675**
	Pr	.341	.363	.514	.354	.502	.324	.354	**.565**
	Re	.552	.572	.605	.563	.561	.531	.531	**.649**
	F1	.403	.416	.567	.409	.519	.514	.524	**.604**

Results on KSDD2. The fourth–sixth rows of Fig 4 show some detection results from various methods on the KSDD2 dataset. The BAS, IRNet, and AMR are interfered with by background, leading to misclassification of background as defects. In contrast, our approach excels at resolving these challenging flaws, producing optimal prediction results. Table 2 further demonstrates this superiority, where our method consistently outperforms other methods. For instance, compared to the MCTF method, our model exhibits enhancements of 4.1%, 7.0%, 5.0%, and 4.5% in mIoU, Precision, Recall, and F1, respectively.

Results on Magnetic Tile. The seventh–ninth rows of Fig. 4 show some detection results of different methods on the Magnetic Tile dataset. The SIPE, IRNet, BAS, and AMR methods encounter challenges in accurately locating

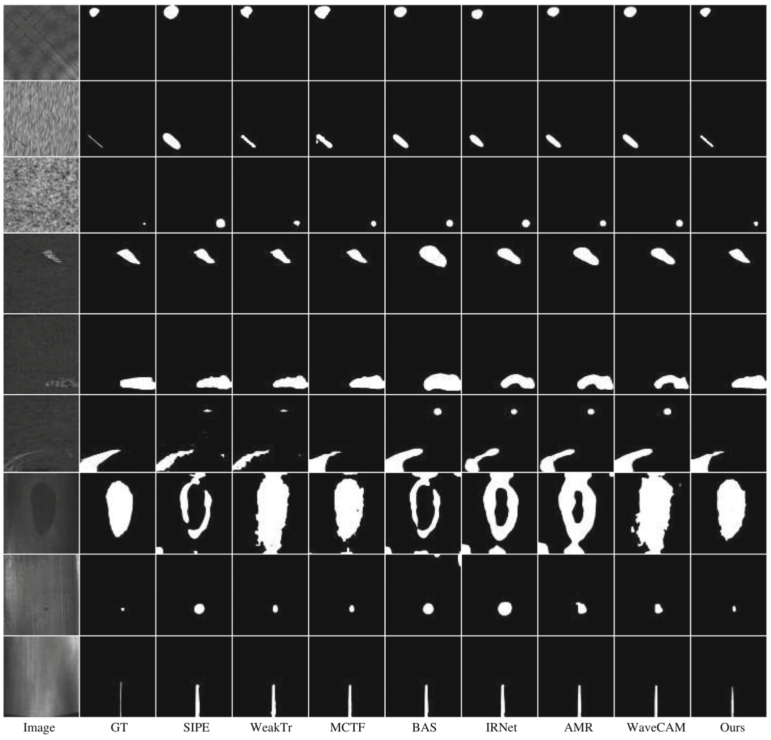


Fig. 4. Qualitative comparison of segmentation results with different methods on DAGM 2007(1st–3rd rows), KSDD2(4th–6th rows), and Magnetic tile(7th–9th rows).

defects. These challenges include incomplete detection of defective areas, overlooking subtle defects, misidentifying defective areas, and especially misclassifying background areas as defects. Furthermore, as shown in Table 2, our approach consistently achieves detection results close to the ground truth (GT) and surpasses comparative methods in performance metrics. For instance, compared to the MCTF method, our model demonstrates improvements of 6.3%, 5.1%, 4.4%, and 3.7% in mIoU, precision, recall, and F1, respectively.

4.5 Ablation Analysis

To showcase the efficacy of each constituent within the proposed network, we conduct a series of ablation experiments about the proposed network with different settings on the above three defective datasets.

Ablation Study for Architecture. To validate the effectiveness of our dual-branch network model, we conduct a series of ablation experiments using architectures with different configurations, including a single CNN branch, a single transformer branch, without CME, and with FCU module of the Conformer [23]

Table 3. Architecture ablation analysis on DAGM 2007, KSDD2 and Magnetic tile.

Architecture	DAGM 2007				KSDD2				Magnetic tile			
	mIoU	Pr	Re	F1	mIoU	Pr	Re	F1	mIoU	Pr	Re	F1
CNN Branch	.653	.537	.468	.486	.675	.532	.601	.562	.554	.321	.530	.393
Vit Branch	.688	.542	.608	.578	.682	.549	.612	.571	.583	.521	.442	.476
w/o CME	.694	.524	.678	.576	.698	.576	.621	.603	.585	.412	.503	.457
w FCU	.713	.642	.618	.632	.707	.542	.618	.576	.598	.541	.374	.442
Ours	**.781**	**.702**	**.752**	**.701**	**.762**	**.727**	**.676**	**.691**	**.675**	**.565**	**.649**	**.604**

Table 4. Module ablation analysis on DAGM 2007, KSDD2 and Magnetic tile.

Module	DAGM 2007				KSDD2				Magnetic tile			
	mIoU	Pr	Re	F1	mIoU	Pr	Re	F1	mIoU	Pr	Re	F1
w/o FCA	.725	.651	.620	.645	.719	.553	.625	.584	.623	.437	.561	.515
w SA	.768	.697	.675	.692	.751	.606	.672	.628	.663	.482	.588	.586
w CA	.765	.687	.672	.682	.748	.612	.675	.636	.661	.486	.592	.591
Ours	**.781**	**.702**	**.752**	**.701**	**.762**	**.727**	**.676**	**.691**	**.675**	**.565**	**.649**	**.604**

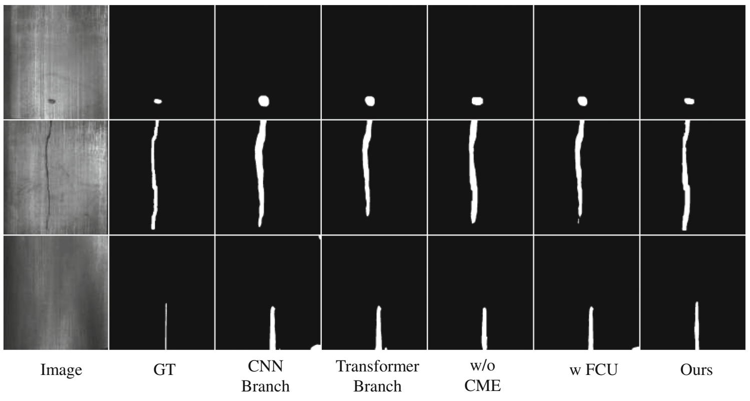


Fig. 5. Visual comparison of segmentation results generated by different network architectures

network. The detailed experimental results are shown in Table 3, and the visual results of the experiment are shown in Fig. 5.

Ablation Study for CME. We employ the model without the FCA module to validate our proposed network module's efficacy. In addition, we analyze and compare the FCA module in CME with the Self-attention and Cross-attention modules, called the variants of Self-attention (w SA) [30] and Cross-attention (w

Table 5. The impact of feature fusion at different blocks of the CME module on DAGM 2007, KSDD2 and Magnetic tile.

Blocks				DAGM 2007				KSDD2				Magnetic tile			
1	2	3	4	mIoU	Pr	Re	F1	mIoU	Pr	Re	F1	mIoU	Pr	Re	F1
✓				.681	.482	.617	.542	.645	.415	.536	.501	.576	.482	.507	.508
✓	✓			.708	.543	.621	.562	.701	.548	.617	.578	.589	.503	.543	.554
✓	✓	✓		.776	.651	.690	.645	.756	.643	.636	.687	.662	.542	.625	.558
✓	✓	✓	✓	.781	.702	.752	.701	.762	.727	.676	.691	.675	.565	.649	.604

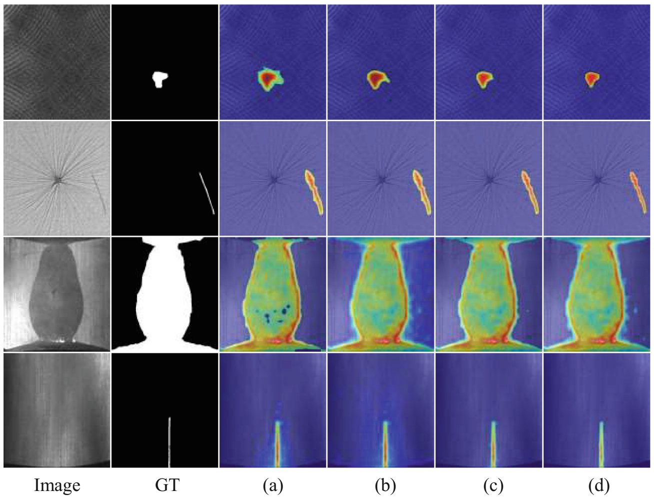


Fig. 6. The CAMs visualization of CME modules used in different blocks. (a)∼(d) represents CAMs generated by the model using the CME module in blocks (1)∼(4).

CA) [19], respectively. We evaluate these variants in terms of mIoU, precision, recall, and F1. The detailed experimental results are shown in Table 4.

Ablation Studies for Different Blocks of CME. To verify the effect of the CME module on the feature fusion of CNN and Transformer branches in different blocks. We use CME modules in different blocks. The detection performance of GLCMENet varies when utilizing CME modules across different blocks, as shown in Table 5. Introducing the CME module in the third block improves the detection performance of the GLCMENet model the most. The detailed experimental results are shown in Table 5, and the visual results of the experiment are shown in Fig. 6.

In our experimental investigations, the local features of the CNN branch and the global features of the Transformer branch evolve with each other using the CME module, and then the CAMs generated by the two branches are fused, achieving the best performance. Particularly noteworthy is the mIoU metric, which shows an incremental enhancement of 6.15% compared to the model without the CME module.

5 Conclusion

In this article, we propose a global-local Context Mutual Evolution Network (CMENet) for weakly supervised surface defect detection. The CMENet introduces a CME module to guide the interaction of global and local features and achieve mutual enhancement. The CME can help the network focus on more defect details and suppress background noise. Experimental results show that CMENet achieves the best performance in three publicly available defective datasets compared with other methods.

References

1. Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2209–2218, 2019
2. Lawrence Amadi and Gady Agam. 2d-pose based human body segmentation for weakly-supervised concealed object detection in backscatter millimeter-wave images. In *International Conference on Pattern Recognition*, pages 124–138. Springer, 2022
3. Božič, J., Tabernik, D., Skočaj, D.: Mixed supervision for surface-defect detection: From weakly to fully supervised learning. Comput. Ind. **129**, 103459 (2021)
4. Qi Chen, Lingxiao Yang, Jian-Huang Lai, and Xiaohua Xie. Self-supervised image-specific prototype exploration for weakly supervised semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4288–4298, 2022
5. Qi Chen, Lingxiao Yang, Jianhuang Lai, and Xiaohua Xie. Self-supervised image-specific prototype exploration for weakly supervised semantic segmentation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4278–4288, 2022
6. Cui, L., Jiang, X., Mingliang, X., Li, W., Lv, P., Zhou, B.: Sddnet: A fast and accurate network for surface defect detection. IEEE Trans. Instrum. Meas. **70**, 1–13 (2021)
7. Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009
8. Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. ArXiv, abs/2010.11929, 2020
9. Banafshe Felfeliyan, Abhilash Hareendranathan, Gregor Kuntze, Stephanie Wichuk, Nils D Forkert, Jacob L Jaremko, and Janet L Ronsky. Weakly supervised medical image segmentation with soft labels and noise robust loss. In *International Conference on Pattern Recognition*, pages 603–617. Springer, 2022

10. Juraj Fulir, Lovro Bosnar, Hans Hagen, and Petra Gospodnetić. Synthetic data for defect segmentation on complex metal surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4423–4433, 2023
11. Wei Gao, Fang Wan, Xingjia Pan, Zhiliang Peng, Qi Tian, Zhenjun Han, Bolei Zhou, and Qixiang Ye. Ts-cam: Token semantic coupled attention map for weakly supervised object localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2886–2895, 2021
12. He, K., Liu, X., Liu, J., Peng, W.: A multitask learning-based neural network for defect detection on textured surfaces under weak supervision. IEEE Trans. Instrum. Meas. **70**, 1–14 (2021)
13. He, K., Liu, X., Liu, J., Peng, W.: A multitask learning-based neural network for defect detection on textured surfaces under weak supervision. IEEE Trans. Instrum. Meas. **70**, 1–14 (2021)
14. Huang, Y., Qiu, C., Yuan, K.: Surface defect saliency of magnetic tile. Vis. Comput. **36**, 85–96 (2020)
15. Jager, M., Knoll, C., Hamprecht, F.A.: Weakly supervised learning of a classifier for unusual event detection. IEEE Trans. Image Process. **17**(9), 1700–1708 (2008)
16. Jiang, P.-T., Zhang, C.-B., Hou, Q., Cheng, M.-M., Wei, Y.: Layercam: Exploring hierarchical class activation maps for localization. IEEE Trans. Image Process. **30**, 5875–5888 (2021)
17. Jiang, X., Yan, F., Yang, L., Wang, K., Guo, S., Zhang, T., Pang, Y., Niu, J., Mingliang, X.: Joint attention-guided feature fusion network for saliency detection of surface defects. IEEE Trans. Instrum. Meas. **71**, 1–12 (2022)
18. Li, Y., Li, J.: An end-to-end defect detection method for mobile phone light guide plate via multitask learning. IEEE Trans. Instrum. Meas. **70**, 1–13 (2021)
19. Hezheng Lin, Xing Cheng, Xiangyu Wu, and Dong Shen. Cat: Cross attention in vision transformer. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2022
20. Xiaofeng Lu and Wentao Fan. Transformer-based encoder-decoder model for surface defect detection. In *2022 the 6th International Conference on Innovation in Artificial Intelligence (ICIAI)*, pages 125–130, 2022
21. Marino, S., Beauseroy, P., Smolarz, A.: Weakly-supervised learning approach for potato defects segmentation. Eng. Appl. Artif. Intell. **85**, 337–346 (2019)
22. Martin Mayr, Mathis Hoffmann, Andreas Maier, and Vincent Christlein. Weakly supervised segmentation of cracks on solar cells using normalized l p norm. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1885–1889. IEEE, 2019
23. Zhiliang Peng, Wei Huang, Shanzhi Gu, Lingxi Xie, Yaowei Wang, Jianbin Jiao, and Qixiang Ye. Conformer: Local features coupling global representations for visual recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 367–376, 2021
24. Athanasios Psaltis, Anastasios Dimou, Federico Alvarez, and Petros Daras. Flow r-cnn: Flow-enhanced object detection. In *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part I*, pages 685–700. Springer, 2021
25. Haochen Qi, Xiangwei Kong, Zhunan Shen, Zhitong Liu, and Jianyi Gu. Progressively learning dynamic level set for weakly supervised industrial defect segmentation. *IEEE Transactions on Instrumentation and Measurement*, 2023
26. Jie Qin, Jie Wu, Xuefeng Xiao, Lujun Li, and Xingang Wang. Activation modulation and recalibration scheme for weakly supervised semantic segmentation. In *AAAI Conference on Artificial Intelligence*, 2021

27. Qin, J., Jie, W., Xiao, X., Li, L., Wang, X.: Activation modulation and recalibration scheme for weakly supervised semantic segmentation. In Proceedings of the AAAI conference on artificial intelligence **36**, 2117–2125 (2022)
28. Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017
29. Tabernik, D., Šela, S., Skvarč, J., Skočaj, D.: Segmentation-based deep-learning approach for surface-defect detection. J. Intell. Manuf. **31**(3), 759–776 (2020)
30. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017
31. Pingyu Wu, Wei Zhai, and Yang Cao. Background activation suppression for weakly supervised object localization. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14228–14237, 2022
32. Xiaojun, W., Qiu, L.T., Xiaodong, G., Long, Z.: Deep learning-based generic automatic surface defect inspection (asdi) with pixelwise segmentation. IEEE Trans. Instrum. Meas. **70**, 1–10 (2020)
33. Xiaojun, W., Wang, T., Li, Y., Li, P., Liu, Y.: A cam-based weakly supervised method for surface defect inspection. IEEE Trans. Instrum. Meas. **71**, 1–10 (2022)
34. Lian Xu, Wanli Ouyang, Mohammed Bennamoun, Farid Boussaid, and Dan Xu. Multi-class token transformer for weakly supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4310–4319, 2022
35. Rongtao, X., Wang, C., Shibiao, X., Meng, W., Zhang, X.: Wave-like class activation map with representation fusion for weakly-supervised semantic segmentation. IEEE Trans. Multimedia **26**, 581–592 (2024)
36. Yang, L., Fan, J., Huo, B., Li, E., Liu, Y.: A nondestructive automatic defect detection method with pixelwise segmentation. Knowl.-Based Syst. **242**, 108338 (2022)
37. Yeung, C.-C., Lam, K.-M.: Efficient fused-attention model for steel surface defect detection. IEEE Trans. Instrum. Meas. **71**, 1–11 (2022)
38. Dehua Zhang, Xinyuan Hao, Dechen Wang, Chunbin Qin, Bo Zhao, Linlin Liang, and Wei Liu. An efficient lightweight convolutional neural network for industrial surface defect detection. *Artificial Intelligence Review*, pages 1–27, 2023
39. Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, 2016
40. Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016
41. Lianghui Zhu, Yingyue Li, Jieming Fang, Yan Liu, Hao Xin, Wenyu Liu, and Xinggang Wang. Weaktr: Exploring plain vision transformer for weakly-supervised semantic segmentation. arXiv preprint arXiv:2304.01184, 2023

Binary-Tree Based Mean-Averaging Estimation for Multi-label Classification

Reshma Rastogi(✉) and Sayanta Chowdhury

Machine Learning and Statistical Inference (MLSI) Lab, Department of Computer Science, South Asian University, Delhi, India
reshma.khemchandani@sau.ac.in, sayanta@students.sau.ac.in

Abstract. Multi-label classification(MLC) is a machine learning problem where each instance may belong to more than one class at the same time. Due to overlapping classes and label-label correlation, solving MLC is very challenging. Further, class imbalance and computational time-complexity are also considered to be major issues. In this paper, we have proposed a novel multi-label classifier that addressed the aforementioned issues; termed as *Binary-Tree based Mean-Averaging estimation for Multi-label classification (BT-MA* (Code is available at: https://github.com/ml-lab-sau/BT-MA*).)*. This proposed classifier takes distinct label-sets meta-feature into account for recovering data imbalance and employs the Divide-and-conquer strategy for resolving time-complexity issue. The experimental results on several benchmark data sets show that our proposed approach *BT-MA* is as competitive as other Multi-label classification approaches.

Keywords: Multi-label classification · Multi-label learning · Label-Subset · Divide-and-conquer algorithm · Meta-feature

1 Introduction

Multi-label classification has emerged as a prominent subject of study within the fields of data mining and machine learning over the past two decades. Multi-label Classification (MLC) is a methodology where each instance is associated with one or multiple class labels at a time. The Multi-label classification approach has found applications in various healthcare sectors, including protein function classification and bio-informatics [2,7,8]. Recently, due to the advancement of generative AI field, researchers in the MLC area also concentrated on multi-label text categorization. [1,2] Presently, a dedicated research team from MBZUAI is actively engaged in the development of a Lib-Multi label library, which is especially for multi-label text categorization [3]. MLC techniques have also been applied to a wide range of issues, including image classification [4], video annotation [9], tag suggestion [5,6], sentiment assessment, information retrieval [10,11], and automated labeling for multimedia content [9].

As observed, multi-label classification has various existing baseline approaches and classifiers. However, the ambiguous nature of multi-label data leads us to

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 271–285, 2025.
https://doi.org/10.1007/978-3-031-78192-6_18

numerous complexities. Dealing with this kind of data requires careful transformation of the complex decision surfaces into more simpler so that the ambiguous problem will explainable to the classical ML approach.

Figure: 1 provides an illustration of overlapping class boundaries within the multi-label data.

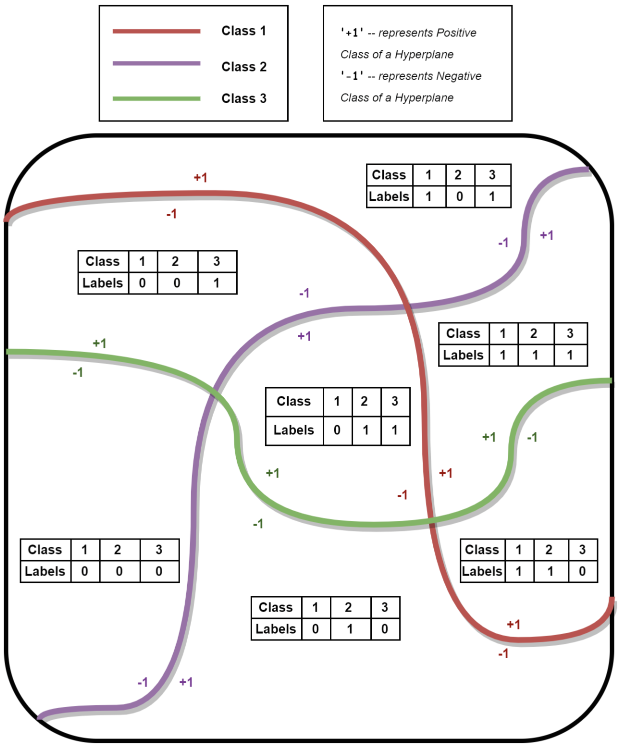

Class 1
Class 2
Class 3
'+1' -- represents Positive Class of a Hyperplane
'-1' -- represents Negative Class of a Hyperplane
Class 1 2 3 / Labels 1 0 1
Class 1 2 3 / Labels 0 0 1
Class 1 2 3 / Labels 1 1 1
Class 1 2 3 / Labels 0 1 1
Class 1 2 3 / Labels 0 0 0
Class 1 2 3 / Labels 1 1 0
Class 1 2 3 / Labels 0 1 0
+1
-1

Fig. 1. Multi-label Class Boundaries

The motivation behind designing our proposed BT-MA algorithm is to introduce a time efficient multi-label model with simpler transformation method which can learn the complex decision surface in more easier steps and will also give faster prediction even with large scale scenario.

The multi-label data can be characterized through the exploration of various meta-features. Among these, *Label Cardinality*, *Label Density* and *Distinct labelsets* are notable examples [12].

Label Cardinality: It is the average number of labels present in per sample [17, 18]. So, $Card(D) = \frac{1}{|D|}\sum_{i=1}^{|D|}|l_i|$.

Label Density: It is represents the average number of labels of the examples exists in D divided by $|L|$ [18,20]. So, $Dens(D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|l_i|}{|L|}$.

Distinct Label-sets: It is the number of different label combinations presents in datasets. The combinations can be represented as a binary vector where i^th position represents as 1 or 0 whether the label present or not [19].

The main contribution in these papers is as follows:

1. We try to incorporate label-based problem transformation techniques through the help of Distinct Label-sets meta-feature, which helps us to transform our multi-label problem into many small and explainable multi-class problems
2. We have used the Mean Average classifier to make it a time-efficient Model for Multi-label classification

2 Related Works

Many standard methodologies for multi-label classification have been introduced. Typically, those are categorized into three fundamental types: i. Transformation-based, ii. Algorithm Adaptation, and iii. Ensemble Technique.

2.1 Transformation-Based Classifiers

In this approach, the objective is to convert multi-label data into binary or multi-class data to make it compatible with traditional machine learning techniques such as Naive Bayesian classifier, Logistic Regression, Support Vector Machines, etc. This method transforms our complex label overlapping issues into a binary, or multi-class problem. Some widely known algorithms of this approach are:
Binary Relevance [13]**:** This converts the multi-label problems into L numbers of binary classification problems through one vs. all strategy. It's final predication output is the union of their prediction. One of it's limitation is it doesn't consider the label similarity. It's basically a brute-force concept to solve MLC problems.
Label Power-set: [20]*:* In this algorithm it converts the multi class problem into single multi-class problem by considering each different label combination as a unique class.
Random k-Label sets [14]*:* This algorithm randomly breaks a large set of labels into n number of small subset of size k which is called k-label-sets.
Calibrated Label Ranking (CLR) [22]*:* This model tackles multi-label classification problem by introducing an artificial label to separate relevant labels from irrelevant ones for each data point. This innovation allows existing pairwise learning techniques, effective for single-label problems, to be applied in the more complex multi-label setting. CLR's strength lies in leveraging these techniques within a multi-label ranking framework.

2.2 Algorithm Adaptation-Based Classifiers

In this approach, the objective is to select a particular learning algorithm and extend its capabilities to handle multi-label data. This extended capability of the learning algorithm might be simple or complicated. However, it primarily depends on the initial formulation of the model and structure of the multi-labels. Many adaptive classifiers utilize well-known machine learning techniques such as Decision trees, Neural networks, and SVM. Some widely recognized algorithms of this approach are:
BP-MLL [2]**:** It is extension of Back Propagation Neural Network to deal with Multi Label Data. It was defined a novel global error function capturing the characteristics of Multi Label Learning which helps model ranking the most relevant label.
ML-KNN [15]**:** It is a extension of popular kNN algorithm to deal with Multi-label Data. In this approach based on the statistical information derived from the neighboring examples, the MAP principle utilized to determine the label set of an unseen example.
Rank-SVM [16]**:** It is a maximum margin approach for multi-label learning. It implemented with the help of kernel trick to incorporate non-linearity.
MLTSVM [23]**:** Chen et al. (2016) introduce the Twin Multi-Label Support Vector Machine (MLTSVM) algorithm, which designed to address the challenge of multi-label classification. MLTSVM leverages the concept of Twin Support Vector Machines (TSVM) [24]. In the binary classification case, TSVM aims to identify two non-parallel hyperplanes, one maximizing the margin for each class while simultaneously minimizing the margin between the classes. Building upon this principle, MLTSVM extends the concept to the multi-label setting.

2.3 Ensemble Technique

In this approach, the objective is to design a particular learning algorithm to handle the special issues of multi-label data, for example: missing label issue, high-dimensionality issue etc. These algorithms may have combination of problem transformation and algorithm adaption approach. Some of the prominent algorithms of this type include:
LCIFS [25]**:** Fan et al. proposed a novel method named Learning Correlation Information for Multi-Label Feature Selection (LCIFS), to address high dimensionality issue of Multi-label Data. *LCIFS* tackles this challenge by concurrently uncovering label correlations and controlling feature redundancy within the data. This joint approach represents a significant advancement in multi-label feature selection, aiming to improve the overall effectiveness of the process.
LRMML [26]**:** Kumar et al. introduce a matrix factorization method for multi-label learning. It leverages a low-rank label assumption to decompose weak-label information. This captures local and global label correlations, while an auxiliary label matrix aids in recovering missing labels.
MLBOTE [27]**:** Building on the concept of borderline oversampling, Teng et al. propose the Multi-Label Borderline Oversampling Technique (MLBOTE)

specifically for addressing class imbalance in multi-label learning tasks. MLBOTE differentiates between various types of informative data points near class borders. It identifies three categories of seed samples: interior, self-borderline, and cross-borderline. Subsequently, the technique employs distinct oversampling mechanisms tailored to effectively handle each category, enhancing the overall effectiveness of the approach.

3 Proposed Model

Let $D = \{(x_k, y_k)\}_{k=1}^{P}$ denotes the multi-label data-set with P samples where x_k is the kth data instance and $y_k = \{0, 1\}^l$ is the associated label-vector with l labels. 1 and 0 indicate the presence or absence of a label respectively for the data instances. Further, the number of 1's in label vector defines the no. of label subset present in a label vector. Using the notation defined, we will discuss the proposed method in following paragraphs.

3.1 Core Methodology

The proposed approach follows a divide-and-conquer strategy based on no. of Distinct Label Subset to solve the multi-label classification problem. The method progressively builds a set of hyperplane for classification and estimating Mean Average of Feature for label assignment process based on label-subset present in each label-vectors.

It successively groups the instances with similar label-subset and then identifies the appropriate label-vector for a data instance by using Mean-Average estimation of Feature. Here, *Mean-Average estimation* refer as, it captures the mean of the Feature value of the training samples along associated labels which it belong to. Later the stored value will used for calculating minimum distance of the testing samples features for making the final label prediction.

The solution approach to build the classifier hyperplane can be viewed as a Binary tree based approach where we successively split the instances and build hyperplane based on label subset. Following paragraph discuss it in detail.

Let P_i represents the number of samples a label subset group $\geq i$ where $i \in \{1, m\}$. m is chosen empirically such that $m \leq l$ and l is the total number of classes. For applying the divide-and-conquer approach, we split the P_i samples into two parts: L_i, the set of samples with label subset $> i$ and R_i, the set of samples with label subset equal(=) to i.

For R_i samples with label subset i, we consider the problem as a multi-class problem where each potential label vector with no. of subset i is consider a pseudo class. To solve the multi-class problem, we employ the Mean Averaging Estimation to identify a classifier hyperplane for each pseudo-label representing a label vector (with i subset). Note the R_i samples drives us towards building classifier hyperplane for ith subset. At this point, we also build a classifier hyperplane to separate the left and right node-clusters. This decision boundary is useful when we choose tree path for previously unseen data instances.

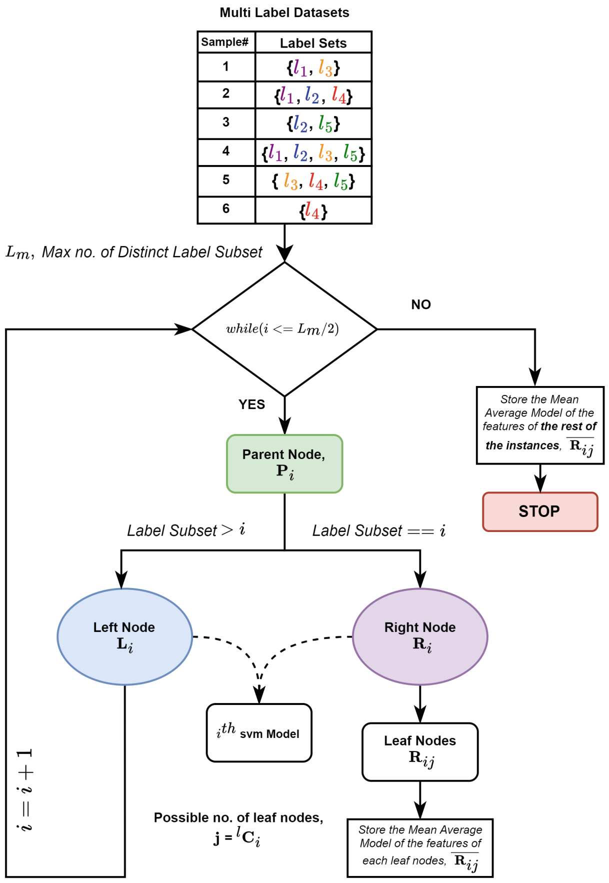

Multi Label Datasets
Sample#
Label Sets
1 $\{l_1, l_3\}$
2 $\{l_1, l_2, l_4\}$
3 $\{l_2, l_5\}$
4 $\{l_1, l_2, l_3, l_5\}$
5 $\{l_3, l_4, l_5\}$
6 $\{l_4\}$
L_m, Max no. of Distinct Label Subset
$while(i <= L_m/2)$
NO
YES
Store the Mean Average Model of the features of the rest of the instances, $\overline{\mathbf{R}_{ij}}$
STOP
Parent Node, $\mathbf{P}_i$
Label Subset $> i$
Label Subset $== i$
Left Node $\mathbf{L}_i$
Right Node $\mathbf{R}_i$
i^{th} svm Model
Leaf Nodes $\mathbf{R}_{ij}$
$i = i + 1$
Possible no. of leaf nodes, j = ${}^{l}\mathbf{C}_i$
Store the Mean Average Model of the features of each leaf nodes, $\overline{\mathbf{R}_{ij}}$

Fig. 2. Flow diagram for the training phase

Since L_i represent the samples with label subset $> i$, we can view these instances as P_{i+1} i.e. the training sample with label subset $\geq (i + 1)$ and apply the approach discussed earlier. It is easy to see that proposed method follows a recursive approach to build appropriate classifier hyperplane successively at each iteration.

Most of the data instances in a multi-label datasets with l possible classes, only have a small fraction of total labels L assigned. Frequently, even the maximum label subset in a multi-label dateset is $L_m << l$. We use this observation, to control the tree-depth for recursion by choosing $i \leq L_m/2$. So, basically we are stopping recursion on half-depth of the tree.

Figure 2 summaries the above discussed divide-and-conquer approach for multi-label learning. The algorithm for building the label-vector specific classifier hyperplane using divide-and-conquer approach is presented in Algorithm 1.

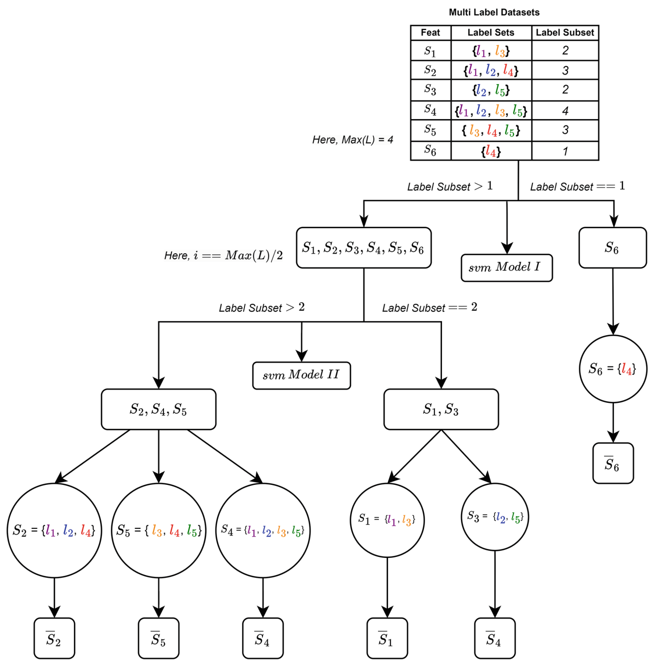


Fig. 3. An illustration of training steps when $L_m = 4$

To further illustrate the proposed method, Fig. 3 shows and example with $L_m = 4$. It also illustrates the split and processing of intermediate and leaf nodes.

In testing phase the datasets passes through the set of trained hyperplane. Then it will enter in the best recommended type of label set combination which is also consider as leaf nodes in our algorithm. For making the final label prediction our model will check the euclidean distance between testing sample feature and the trained label-set's feature. Then specific label-set combination with minimum distance will assign it's label to the testing sample. This label will consider as best recommended label for the testing sample according to our algorithm. To further illustrate the testing steps, Fig. 4 shows an example with four testing samples.

Algorithm 1. Training steps for Binary Tree based Mean-averaging Model

```
Input: The training data $D$
Output: Classifier Hyperplane and Mean Average estimation for different label-vector sets
Initialization: Iterator $i = 1$,
$svmModel(i) = [\ ]$, $\overline{R_{(i,j)}} = [\ ]$,
Max Number of Label Subset, $L_m$ = Max(L)
 1: Make cluster of all the training instances based on it's label subset;
 2: repeat
 3:     Split the samples into Left $(L_i)$ and Right Node $(R_i)$ cluster;
 4:     Left cluster contains instances with label subset $> i$;
 5:     Right cluster contains instances with the label subset $= i$;
 6:     Build a classifier hyperplane between Left$(L_i)$ and Right$(R_i)$ Node-clusters using SVM Model;
 7:     Store the model into $svmModel(i)$;
 8:     Take a Mean Average of features for the right cluster to pick the label vector from one of many same label subset;
 9:     Store the Mean Average to the container $\overline{R_{(i,j)}}$;
10:     $i = i + 1$;
11: until $i \leq (L_m/2)$
12: $RETURN\ svmModel(i), \overline{R_{(i,j)}}$;
```

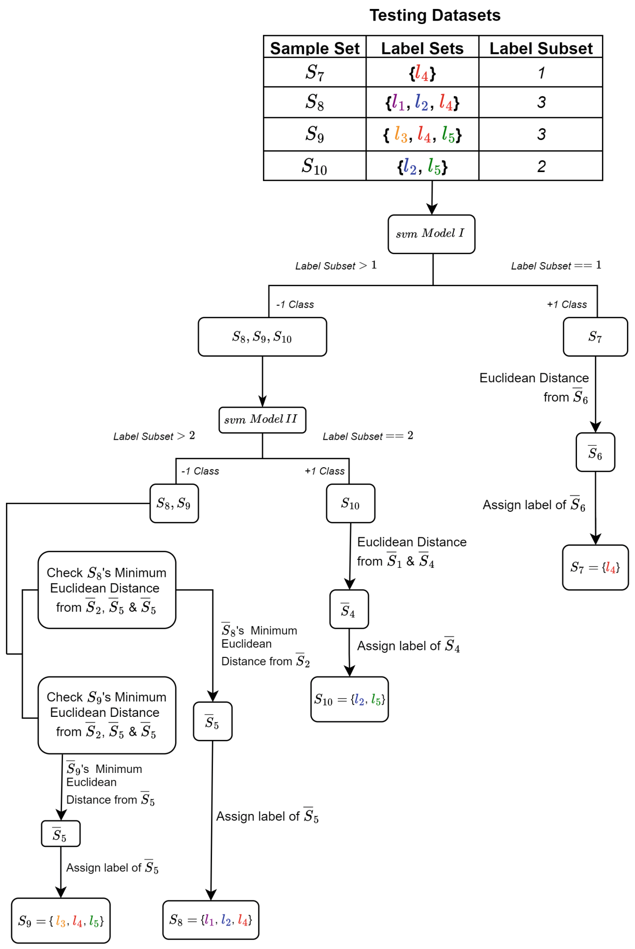
4 Experimental Results

In our experiment, we have performed 5-fold cross-validation on six well-known multi label datasets. Also, we have compared our proposed model with three baseline models BR-SVM, ML-KNN and LSML for the performance evaluation measure. MATLAB R2021a has been used for experiments on a Windows 11 OS with Intel Core i7-8700 CPU @3.20GHz processor and 16 GB RAM.

4.1 Data-Sets Description

The effectiveness of our algorithm we have validate on the six benchmark multi-label data-sets. Those are: Yahoo-Arts[1], Bookmarks[2], Yahoo-Science[3], Bibtex[4], GnegativeGO[5] and PlantGO[6]. Details tabular description of data-sets are given Table 1.

[1] Yahoo-Arts Dataset.
[2] Bookmarks Dataset.
[3] Yahoo-Science Dataset.
[4] Bibtex Dataset.
[5] GnegativeGO Dataset.
[6] PlantGO Dataset.



Testing Datasets

Sample Set	Label Sets	Label Subset
S_7	$\{l_4\}$	1
S_8	$\{l_1, l_2, l_4\}$	3
S_9	$\{l_3, l_4, l_5\}$	3
S_{10}	$\{l_2, l_5\}$	2

svm Model I
Label Subset > 1
Label Subset == 1
-1 Class
+1 Class
S_8, S_9, S_{10}
S_7
Euclidean Distance from $\overline{S}_6$
$\overline{S}_6$
Assign label of $\overline{S}_6$
$S_7 = \{l_4\}$
svm Model II
Label Subset > 2
Label Subset == 2
-1 Class
+1 Class
S_8, S_9
S_{10}
Euclidean Distance from $\overline{S}_1$ & $\overline{S}_4$
$\overline{S}_4$
Assign label of $\overline{S}_4$
$S_{10} = \{l_2, l_5\}$
Check S_8's Minimum Euclidean Distance from $\overline{S}_2$, $\overline{S}_5$ & $\overline{S}_5$
$\overline{S}_8$'s Minimum Euclidean Distance from $\overline{S}_2$
$\overline{S}_5$
Assign label of $\overline{S}_5$
$S_8 = \{l_1, l_2, l_4\}$
Check S_9's Minimum Euclidean Distance from $\overline{S}_2$, $\overline{S}_5$ & $\overline{S}_5$
$\overline{S}_9$'s Minimum Euclidean Distance from $\overline{S}_5$
$\overline{S}_5$
Assign label of $\overline{S}_5$
$S_9 = \{l_3, l_4, l_5\}$

Fig. 4. An illustration of the testing phase considering four example sample set

4.2 Compared Algorithms

BR-SVM[7]: Boutell, Matthew R., et al. [4] proposed a simple way to deal with a multi-label classification tasks by converting the multi-label problem into multiple binary classifications problems where each using a linear Support Vector Machine as a base classifier. This approach trains each class individually, ignoring any label dependencies.

ML-KNN[8]: Zhang, M.L., et al. [15] proposed an algorithm adaptation approach of the original kNN algorithm designed for multi-label learning. This algorithm works based on the traditional KNN algorithm. It also introduces max-

[7] BR-SVM model's code from MLC Toolbox.
[8] ML-KNN model's code from PALM Lab.

Table 1. Summary of Multi-label data-sets

Dataset	*Domain*	*Instances*	*Feature*	*labels*
Arts	text	4498	23146	26
Bookmarks	text	52776	2150	208
Science	text	3865	37187	40
Bibtex	text	4495	1836	159
GnegativeGO	biology	933	1717	8
PlantGO	biology	656	3091	12

imum posterior probability (MAP) for dealing with multi-label classification problems.

LSML[9]: Huang, Jun, et al. [21] proposed a new approach for multi-label classification with missing labels for learning the Label-Specific features. It creates a supplementary label matrix by capturing high-order label correlations from the incomplete label matrix. Then, it learns label-specific data representation for each class label and develops a model by integrating the learned high-order label correlations.

4.3 Evaluation Metrics

We have evaluated the performance of the compared algorithms based on six standard metrics, reported in Table 2 to 7. Given a test data set $T_d = \{x_k, Y_k\}_{k=1}^{N_d}$, where $Y_k \in Y$ is the set of ground truth labels, and $h(x_k)$ is the set of predicted labels for the k^{th} instance. Let $f(x_k, y)$ indicates the confidence score of a point x_k belonging to the label y.

1. **Execution Time:** It reports the total amount of time taken by the algorithm. It reports the unit of time in seconds.

Table 2. Performance analysis of GnegativeGO Data-set

	Evaluation Metrics					
Algorithms	$\downarrow$ CPU Time(Sec)	$\uparrow$ ExactM	$\downarrow$ HamL	$\uparrow$ MacroF1	$\uparrow$ MicroF1	$\uparrow$ AvePre
BT-MA	**0.203**	**0.927**	0.014	0.837	0.947	0.960
BR-SVM	1.044	0.897	0.017	0.818	0.932	0.942
ML-kNN	5.935	0.894	0.019	0.782	0.928	0.945
LSML	3.1321	0.915	**0.013**	**0.956**	**0.949**	**0.978**

[9] LSML model's code from Huang Jun's Site.

2. **Hamming Loss(HamL):** It determines the number of times an instance-label pair has been misclassified, i.e., a label associated with an instance is not predicted or a label not associated with an instance is predicted.

$$HLoss = \frac{1}{N_d}\sum_{k=1}^{N_d}\frac{1}{l}|h(x_k)\Delta Y_k|$$

   Where the symmetric difference between two sets is indicted by Δ, N_d indicates the number of instances and l indicates the number of labels.

3. **Macro F1 (MacroF1):** Macro-F1 is the harmonic mean between precision and recall, where the average is calculated per label and then averaged across all labels. If p_j and r_j are the precision and recall for all $\lambda_j \in h(x_i)$ from $\lambda_j \in y_i$, the macro-F1 is

$$MacroF1 = \frac{1}{Q}\sum_{j=1}^{Q}\frac{2 \times p_j \times r_j}{p_j + r_j}$$

4. **Micro F1 (MicroF1):** It is a more advanced version of the F1 Measure that treats each label vector entry as a single example.

$$MicroF1 = \frac{2\sum_{q=1}^{l}\sum_{k=1}^{N_d} y_{kq}h(x_{kq})}{\sum_{q=1}^{l}\sum_{k=1}^{N_d} y_{kq} + \sum_{q=1}^{l}\sum_{k=1}^{N_d} h(x_{kq})}$$

5. **Average Precision(AvgPre):** This evaluates the average fraction of positive labels that are higher than the particular labels.

$$AvgP = \frac{1}{N_d}\sum_{k=1}^{N_d}\frac{1}{|Y_k|}\sum_{y\in Y_k}\frac{|\{y'|rank_{f(x_k,y')} \leq rank_{f(x_k,y)}, y' \in Y_k\}|}{rank_{f(x_k,y)}}$$

Table 3. Performance analysis of Science Data-set

	Evaluation Matrics					
Algorithms	↓ CPU Time(Sec)	↑ ExactM	↓ HamL	↑ MacroF1	↑ MicroF1	↑ AvePre
BT-MA	1.385	**0.339**	0.039	0.203	**0.378**	0.432
BR-SVM	14.832	0.132	**0.032**	0.076	0.234	0.227
ML-kNN	13.075	0.097	0.034	0.092	0.179	0.184
LSML	**0.944**	0.000	0.992	**0.263**	0.341	**0.606**

Table 4. Performance analysis of Yahoo-Arts Data-set

	Evaluation Metrics					
Algorithms	↓CPU Time(Sec)	↑ ExactM	↓ HamL	↑ MacroF1	↑ MicroF1	↑ AvePre
BT-MA	0.825	**0.326**	0.067	0.261	**0.396**	0.497
BR-SVM	6.816	0.17	**0.054**	0.117	0.285	0.363
ML-kNN	7.582	0.041	0.061	0.052	0.086	0.237
LSML	**0.406**	0.000	0.986	**0.285**	0.345	**0.628**

Table 5. Performance analysis of Bookmarks Dataset

	Evaluation Matrics					
Algorithms	↓ CPU Time(Sec)	↑ ExactM	↓ HamL	↑ MacroF1	↑ MicroF1	↑ AvePre
BT-MA	**7.136**	**0.173**	0.030	0.064	0.101	0.218
BR-SVM	438.837	0.153	**0.009**	0.038	0.179	0.189
ML-kNN	90.775	0.157	**0.009**	0.041	0.186	0.194
LSML	17.1432	0.000	0.999	**0.193**	**0.214**	**0.501**

Table 6. Performance analysis of PlantGO Dataset

	Evaluation Matrics					
Algorithms	↓ CPU Time(Sec)	↑ ExactM	↓ HamL	↑ MacroF1	↑ MicroF1	↑ AvePre
BT-MA	**0.219**	**0.691**	0.047	0.678	0.730	0.774
BR-SVM	2.087	0.569	0.046	0.469	0.698	0.7
ML-kNN	3.929	0.537	0.053	0.501	0.668	0.681
LSML	11.145	0.688	**0.037**	**0.776**	**0.794**	**0.902**

Table 7. Performance analysis of Bibtex Data-set

	Evaluation Matrics					
Algorithms	↓ CPU Time(Sec)	↑ ExactM	↓ HamL	↑ MacroF1	↑ MicroF1	↑ AvePre
BT-MA	**6.909**	0.137	0.029	0.167	0.224	0.275
BR-SVM	188.388	0.103	**0.013**	0.053	0.246	0.222
ML-kNN	66.787	0.037	0.014	0.051	0.186	0.148
LSML	11.198	**0.162**	0.015	**0.473**	**0.507**	**0.612**

5 Conclusion and Discussion

In this paper, we have proposed a Binary tree based Multi-label classifier which works on divide and conquer strategy with mean averaging estimation. The criteria of split the data is based on distinct label subset afterwards it calculate mean average estimation. We have used support vector machine as a classifier at each node. Results on seven multi-label datasets have been reported, which also shows that of our algorithm also gives a competitive performance with respect to the compared algorithm.

The reason of our algorithm's competitive performance:

1. Time Optimization: We have adopt divide and conquer approach due to which in training it requires less amount of time.
2. Intermediate Node Optimization: We have used SVM as a base classifier in intermediate nodes which is very robust classifier. Due to which the model gets better guidance to learn even complex samples in intermediate nodes.
3. Problem Transformation approach: Using the distinct label-set calculation we have converted most of our training samples into binary classification and multi-class classification problems which another reason for better learning capability. Due to our simple transformation technique the training and testing samples are more explainable to our model.
4. Leaf Node Optimization: In leaf nodes, we captures the mean value of features in time of training Phase. In testing phase, we have just look for minimum euclidean distance between the testing sample and mean value of features. Then we have assigned better recommended labels to those samples based on minimum distance.

Since we have adopted the problem transformation approach using distinct label-set approach; for this reason in some label-set combination there may have very less samples. Due to which the intermediate and leaf nodes may suffer.
Future Line of Works

– Our model can easily categories the simpler and more complex multi-label problems. Using this feature it is possible to give special treatment to the complex multi-label problems for getting more competitive performance.

– Also, it can be extended to a label dependency based model using distinct label-subset.

References

1. McCallum, A.K.: Multi-label text classification with a mixture model trained by EM. In: AAAI 99 Workshop on Text Learning, Citeseer (1999)
2. Zhang, M.-L., Zhou, Z.-H.: Multilabel neural networks with applications to functional genomics and text categorization. IEEE Trans. Knowl. Data Eng. **18**(10), 1338–1351 (2006)
3. Liu, J., et al.: ASUS-AICS/LibMultiLabel: A Library for Multi-Class and Multi-Label Text Classification. GitHub (2023). https://github.com/ASUS-AICS/LibMultiLabel
4. Boutell, M.R., Luo, J., Shen, X., Brown, C.M.: Learning multi-label scene classification. Pattern Recogn. **37**(9), 1757–1771 (2004)
5. Katakis, I., Tsoumakas, G., Vlahavas, I.: Multilabel text classification for automated tag suggestion. In: Proceedings of the ECML PKDD 2008 Discovery Challenge, Antwerp, Belgium, pp. 75–83 (2008)
6. Song, Y., Zhang, L., Giles, L.C.: A sparse gaussian processes classification framework for fast tag suggestions. In: Proceeding of the 17th ACM Conference on Information and Knowledge Management, Napa Valley, CA, pp. 93–102 (2008)
7. Clare, A., King, R.D.: Knowledge discovery in multi-label phenotype data. In: De Raedt, L., Siebes, A. (eds.) PKDD 2001. LNCS (LNAI), vol. 2168, pp. 42–53. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44794-6_4
8. Elisseeff, A., Weston, J.: Kernel methods for Multi-labelled classification and Categ orical regression problems. In: Neural Information Processing Systems (2001)
9. Qi, G.-J., Hua, X.-S., Rui, Y., Tang, J., Mei, T., Zhang, H.-J.: Correlative multilabel video annotation. In: Proceedings of the 15th ACM International Conference on Multimedia, Augsburg, Germany, pp. 17–26 (2007)
10. Gopal, S., Yang, Y.: Multilabel classification with meta-level features. In: Proceedings of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Geneva, Switzerland, pp. 315–322 (2010)
11. Zhu, S., Ji, X., Xu, W., Gong, Y.: Multi-labelled classification using maximum entropy method. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, pp. 274–281 (2005)
12. Chekina, L., Rokach, L., Shapira, B.: Meta-learning for selecting a multi-label classification algorithm. In: 2011 IEEE 11th International Conference on Data Mining Workshops, pp. 220–227. IEEE (2011)
13. Boutell, M.R., Luo, J., Shen, X., Brown, C.M.: Learning multi-label scene classification. Pattern Recogn. **37**(9), 1757–1771 (2004)
14. Tsoumakas, G., Vlahavas, I.: Random *k*-Labelsets: an ensemble method for multilabel classification. In: Kok, J.N., Koronacki, J., Mantaras, R.L., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 406–417. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74958-5_38
15. Zhang, M.L., Zhou, Z.H.: ML-KNN: a lazy learning approach to multi-label learning. Pattern Recogn. **40**(7), 2038–2048 (2007)
16. Elisseeff, A., Weston, J.: A kernel method for multi-labelled classification. In: Advances in Neural Information Processing Systems, vol. 14 (2001)

17. Tsoumakas, G., Katakis, I.: Multi-label classification. Data Warehousing and Mining: Concepts, Methodologies, Tools, and Applications: Concepts, Methodologies, Tools, and Applications **3**, 64 (2008)
18. Bernardini, F.C., da Silva, R.B., Rodovalho, R.M., Meza, E.B.M.: Cardinality and density measures and their influence to multi-label learning methods. Submitted to Learning and Nonlinear Models (2014)
19. Tsoumakas, G., Katakis, I.: Multi-label classification: an overview. Int. J. Data Warehous. Min. (IJDWM) **3**(3), 1–13 (2007)
20. Zhang, M.L., Zhou, Z.H.: A review on multi-label learning algorithms. IEEE Trans. Knowl. Data Eng. **26**(8), 1819–1837 (2013)
21. Huang, J., et al.: Improving multi-label classification with missing labels by learning label-specific features. Inf. Sci. **492**, 124–146 (2019)
22. Fürnkranz, J., Hüllermeier, E., Loza Mencía, E., Brinker, K.: Multilabel classification via calibrated label ranking. Mach. Learn. **73**, 133–153 (2008)
23. Chen, W.J., Shao, Y.H., Li, C.N., Deng, N.Y.: MLTSVM: a novel twin support vector machine to multi-label learning. Pattern Recogn. **52**, 61–74 (2016)
24. Khemchandani, R., Chandra, S.: Twin support vector machines for pattern classification. IEEE Trans. Pattern Anal. Mach. Intell. **29**(5), 905–910 (2007)
25. Fan, Y., Liu, J., Tang, J., Liu, P., Lin, Y., Du, Y.: Learning correlation information for multi-label feature selection. Pattern Recogn. **145**, 109899 (2024)
26. Kumar, S., Rastogi, R.: Low rank label subspace transformation for multi-label learning with missing labels. Inf. Sci. **596**, 53–72 (2022)
27. Teng, Z., Cao, P., Huang, M., Gao, Z., Wang, X.: Multi-label borderline oversampling technique. Pattern Recogn. **145**, 109953 (2024)

FAT: Fusion-Attention Transformer for Remaining Useful Life Prediction

Trung Hieu Vu[1,2](✉), Eyad Elyan[1], Will Vorley[2], Joe Goodlad[2], Truong Dang[3], and Tien Thanh Nguyen[3]

[1] School of Computing, Robert Gordon University, Aberdeen, UK
h.vu1@rgu.ac.uk
[2] ADC Energy Ltd., Aberdeen, UK
[3] National Subsea Centre, Robert Gordon University, Aberdeen, UK

Abstract. The prediction of equipment failures in the manufacturing industry, through the estimation of Remaining Useful Life (RUL), is crucial for minimizing downtime and optimizing maintenance planning. However, developing effective RUL prediction models faces challenges, particularly in capturing long-term dependencies in time series data. Traditional Transformer approaches, using multi-head self-attention mechanisms, suffer from high computational complexity and insensitivity to local regions. To address these issues, we propose a novel approach called Fusion-Attention Transformer (FAT) for RUL prediction. Our model integrates two key components: multi-head Logsparse Self-Attention (LSA) and multi-head Auto-correlation Self-Attention (ASA). LSA employs a logarithmic function and a local sparse strategy, reduces computation, and enhances local information. ASA mainly analyzes the seasonality, which reveals periodic fluctuations in the time series. The combined features of LSA and ASA are then fed into a feed-forward neural network for RUL prediction. Extensive experiments on public datasets, reveal that Feature Attention Transformer (FAT) surpasses various state-of-the-art (SOTA) methods. The findings emphasize the enhanced performance achieved by combining different self-attention mechanisms compared to utilizing them individually.

Keywords: Transformers · Remaining Useful Life (RUL) · Attention mechanism · Deep learning

1 Introduction

Prognostics and Health Management (PHM) technology is defined as the monitoring of the deterioration of an asset through detection, diagnosis, and prognostics [1]. Typically, this is aimed at predicting the Remaining Useful Life (RUL) using historical data representing the equipment and the corresponding operations conditions. The approaches of RUL are divided into four categories: physics model-based, statistical model-based, AI-driven, and hybrid approaches [2].

In recent years, the widespread application of Machine Learning (ML) and Deep Learning (DL) techniques, such as convolutional neural networks (CNNs) [5] and long-short-term memory neural networks (LSTMs) [8], has been pivotal in addressing equipment defect prediction [3]. Notably, studies by Li et al. [6] demonstrated the superiority of CNNs and LSTMs over traditional ML methods for predicting defects. J. Zhang et al. [7] proposed a hybrid architecture combining bidirectional gated recurrent and CNN to address spatial-temporal features in predicting Remaining Useful Life (RUL). Wu et al. [9] utilized vanilla LSTM models for RUL prediction, while Huang et al. [10] introduced bidirectional long short-term memory (BiLSTM) networks capturing features from multiple raw sensors and operational conditions. However, limitations exist for both CNN and LSTM methods: (i) CNN-based models using convolution kernel size for processing temporal data struggle to capture long-term dependent information [22]; (ii) LSTM-based models face challenges in building relationships with non-adjacent data points, significantly limiting parallel computing speed during training [15]; and (iii) Both models may encounter issues of vanishing or exploding gradients when processing long-time sequences [11].

Transformer-based architecture models, originating from vanilla Transformer [12], have emerged recently as a powerful solution to handle the above limitations. Using the multi-head self-attention mechanism embedded into the encoder block allows the model to learn the hidden relationship between non-adjacent observations, while still enabling it to fully exploit parallel computation. Additionally, the model employs residual connections to handle challenges such as gradient explosion and vanishing. Ma et al. [14] introduced a Transformer encoder that integrates guiding features to segment modalities for improved sampling. Mo et al. [13] proposed a gated convolutional unit with the Transformer encoder as a backbone to predict RUL. Chen et al. [15] presented a denoising auto-encoder as a pre-processing step before feeding it into the Transformer encoder for RUL prediction. However, conventional transformers still have some challenges. Firstly, the self-attention mechanism requires calculating attention scores over all data points, which increases the computational complexity [23]. Secondly, employing a global point-wise dot-product self-attention mechanism causes the model to be insensitive to local context information [4].

To overcome these challenges, we introduce a novel Fusion-Attention Transformer (FAT) architecture designed for Remaining Useful Life (RUL) predictions of engine systems under complex operational conditions. Our contributions to this paper are summarized as follows:

– We proposed a novel fusion attention-based prognostic model containing three major components. One LSA network is utilized to automatically extract local information features hidden in multiple sensors' signals. Another ASA network is adopted to explore long-term dependencies hidden in the operational condition signals. These combined features are fed into a stack of more dense layers and a single linear regression layer to obtain a final RUL.
– We evaluated the effectiveness of the proposed FAT via an aircraft turbofan engine dataset with four datasets (FD001 - FD004). The proposed method

showcases superior performance over existing Sequence-to-Sequence models and different Transformer-based architectures in multiple operational conditions and fault modes. Furthermore, the integrated model surpasses the individual self-attention components (LSA and ASA) in achieving better results.

The rest of the paper is organized as follows. Section 2 introduces a Multi-head self-attention mechanism and Transformer encoder. Section 3 shows the problem statement and our proposed method. Section 4 discusses experiments and results. Section 5 concludes the paper.

2 Related Work

A brief description of the main component of the vanilla Transformer [12] architecture, including the multi-head attention mechanism and encoder blocks, is presented here.

2.1 Multi-head Self-attention Mechanism

In the single-head attention module, it is assumed that the input vectors $X \in \mathbb{R}^{L \times d_{model}}$ are linearly transformed into three main components: queries matrices $Q \in \mathbb{R}^{L \times d_Q}$, keys matrices $K \in \mathbb{R}^{L \times d_K}$, and values matrices $V \in \mathbb{R}^{L \times d_V}$ where L is the sequence length, d_{model}, d_Q, d_K, d_V are the dimensions of inputs, queries, keys, and values matrices, respectively, as presented in Eq. 1:

$$Q = XW_Q, K = XW_K, V = XW_V \tag{1}$$

where $W_Q \in \mathbb{R}^{d_{model} \times d_Q}$, $W_K \in \mathbb{R}^{d_{model} \times d_K}$, $W_V \in \mathbb{R}^{d_{model} \times d_V}$ are trainable weight matrices. Then, three matrices are used to apply scaled dot-product attention to measure the correlation between data points in the sequence input, as shown in Eq. 2:

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2}$$

where T denotes the transpose operator, the scale factor $\sqrt{d_k}$ is identical to counteract excessively large dot products.

It is also worth noting that a single attention head faces challenges in learning different representations of sub-spaces. Using multi-head attention (h heads) allows the model to learn diverse representations of information (Eq. 3).

$$\begin{gathered} MSA(Q, K, V) = Concat(head_1, \ldots, head_h)W_O \\ \text{where } head_i = Attention(Q_i, K_i, V_i) \end{gathered} \tag{3}$$

where the i-th denotes the attention head, and W_O are learnable weight matrices.

2.2 Transformer Encoder

Encoder blocks are a key component of the Transformer architecture. The encoder in the Transformer architecture typically includes a stack of multiple encoder layers, each of which contains two main sub-blocks: MSA block and feed-forward block. The input vector, integrated with position encoding information, is directed into the MSA block. Subsequently, the output of the MSA block is combined with a residual connection and serves as the input to the layer normalization [16], shown as follows:

$$O_{MSA} = LayerNorm(X + MSA(X)) \tag{4}$$

where O_{MSA} is the output of the MSA block and a residual connection. The output O_{FFN} of the feed-forward block can be calculated after O_{MSA} passes through a fully connected feed-forward network. This network includes two linear transformations with an intervening rectified linear unit (ReLU) activation between them, represented as follows:

$$O_{FFN}(O_{MSA}) = max(0, O_{MSA}W_1 + b_1)W_2 + b_2 \tag{5}$$

where W_1, b_1, W_2, and b_2 denote the learnable parameters between the linear mappings and biases, respectively. It should be emphasized that the RUL prediction studied in this paper is distinct from the challenges of Natural Language Processing (NLP) and Computer Vision (CV). This study mainly focuses on the Encoder block in the Transformer architecture.

3 Methodology

3.1 Problem Statement

In this study, we define the RUL prediction problem. The input features are denoted by $X = \{x_1, x_2, \ldots, x_n\}$, where $x_i \in \mathbb{R}^d$ represents the d-dimensional sensor data collected at i-th time step. Correspondingly, the actual values for the system are denoted by $Y = \{y_1, y_2, \ldots, y_n\}$ where y_i represents the RUL at the i-th time step. This paper aims to establish the best mapping relationship between collected sensor data (input data) and the RUL ground truth, as demonstrated in Eq. 6.

$$Y_i = f(X_i) \tag{6}$$

where f is the mapping function acquired through the training process. To tackle this task, we introduce a novel fusion-attention transformer described in detail in the subsequent sections.

3.2 Overall Architecture of the Proposed Model

The overall architecture of the proposed FAT is demonstrated in Fig. 1. In this section, we intuitively discuss this framework and its model structure.

Regarding the training process, the constant values that do not significantly contribute to the performance of the model are removed. On this basis, we construct the processed time window data and corresponding RUL as input for the model. The fusion-attention block contains two important components: the multi-head Logsparse Self-Attention (LSA) mechanism and the multi-head Auto-correlation Self-Attention (ASA) mechanism. Inside the LSA block is a combination of logarithm and local window strategies, allowing the model to exploit local information [4]. Furthermore, using this mechanism also helps the model to reduce the computational burden, breaking the information bottleneck. The ASA block employs a series-decomposition strategy and auto-correlation mechanism, allowing the model to analyze the time series into seasonal components [21]. Consequently, this approach can minimize the impact of local noise and mine more global information. The information extracted by two components is then integrated, and feed-forward layers are employed to map the fusion features to RUL ground truth.

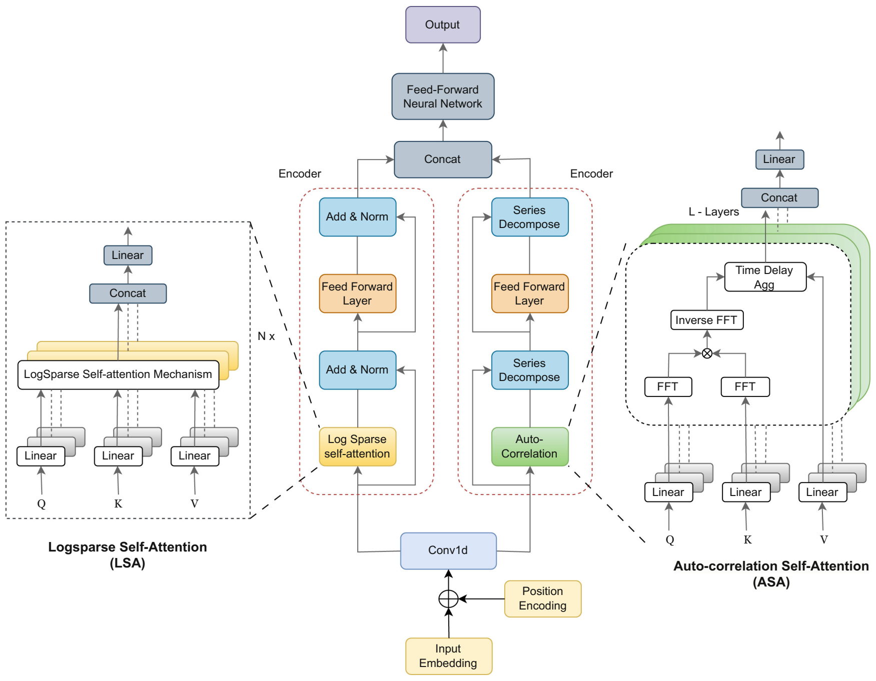


Fig. 1. The framework of the proposed FAT.

3.3 Multi-head LogSparse Self-attention Mechanism

The LSA is designed to deal with long sequences. It introduces a logarithm-based sparse strategy to determine which elements in the attention matrices are relevant for computation, thereby enhancing computational efficiency.

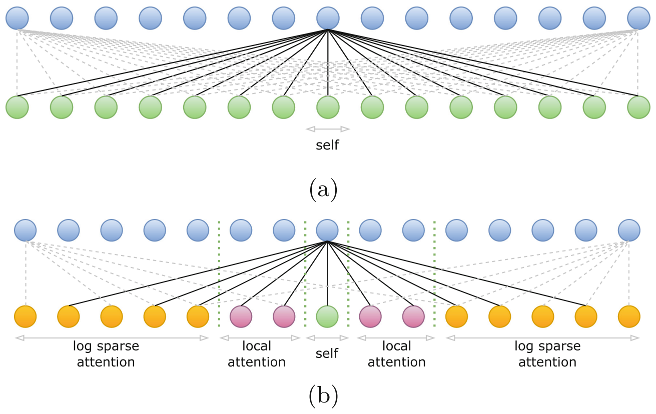


Fig. 2. (a) Self Attention Mechanism (b) Log Sparse Attention Mechanism.

Considering the MSA mechanism in a conventional Transformer, from the k^{th} layer to the $(k+1)^{th}$ layer, it can be assumed that each index cell at the $(k+1)^{th}$ layer will undergo self-attention calculation with all the cells in the k^{th} layer. We consider the index set in a time window length L as $I_l^k = \{1, 2, \ldots, l, \ldots, L\}$, as illustrated in Fig. 2(a). Since this approach connects all cells, its computational complexity is quadratic in the time window length L. To overcome this challenge, we implemented a logarithmic strategy, which selects indexes in window length L based on logarithmic criteria [17]. This strategy obtains a new set of index $I'^k_l = \{l-2^{\lfloor \log_2 L \rfloor}, l-2^{\lfloor \log_2 L \rfloor -1}, \ldots, l-2^0, l, l+2^0, \ldots, l+2^{\lfloor \log_2 L \rfloor -1}, l+2^{\lfloor \log_2 L \rfloor}\}$, where $\lfloor . \rfloor$ floor function rounds a real number down to the nearest integer. It can be shown in Fig. 2(b).

With every cell in each layer, applying logarithmic function results in $O(\log L)$ complexity in the time window length L. Moreover, the author stacks up $O(\log L)$ layers, and after the model can access all the cells' information. A detailed proof can be found at [4]. Using stacked $\lfloor \log_2 L \rfloor + 1$ layers can obtain a comprehensive memory of $O(L(\log_2 L)^2)$ and effectively addresses the bottleneck issue of the conventional self-attention mechanism. Furthermore, for two cells j and l that are far apart $(j < l)$, the number of paths between these two cells can increase exponentially, followed by $O(\log_2(l - j))$, which means the amount of information flow will become richer.

Given a time window length of L, for any $1 \leq l \leq L$, there is at least one path from the cell index j, where $j \in [1, L]$, to cell l if the module is stacked to the $\lfloor (\log_2 L) \rfloor$. For any $1 \leq j \leq l$, it can be demonstrated in [4] that the l-th cell collects information from all preceding cells, including itself. In the situation, $l \leq j \leq L$ where information is received by the cells behind l-th cell, including itself to avoid the loss of information, in details are proven by Zhang et al. [17].

In the domain of RUL forecasting, the later observations significantly influence the prediction of failure points. Therefore, incorporating *local attention*, which allows each cell to densely attend to adjacent cells, enables the extraction of more information from recent observations. Employing local attention, in conjunction with log sparse attention, cannot change its complexity [4]; instead, it facilitates the extraction of additional information from the latest observations, enhancing the accuracy of equipment failure point predictions.

3.4 Auto-correlation Self-attention Mechanism

Conventional attention mechanisms, particularly point-wise self-attention, often counter bottleneck information and cannot directly find reliable dependencies in intricate temporal patterns. To address this limitation, Wu et al. [21] proposed an Auto-Correlation mechanism with series-wise connections with $O(L\log(L))$ complexity, which captures period-based dependencies and aggregate information at the series level. In contrast to the vanilla Time Series Transformer, where attention weights are computed in the time domain and aggregated subsequently point-wise, the auto-correlation mechanism with the primary component, the Auto Correlation block, adopts a distinctive approach.

The auto-correlation self-attention mechanism in the Auto Correlation block exploits the period-based dependencies and accumulates the information at the series level. The mechanism is implemented as follows. The embedding input $X \in \mathbb{R}^{L\times d_{model}}$ are transformed into query matrix $Q \in \mathbb{R}^{L\times d_Q}$, key matrix $K \in \mathbb{R}^{L\times d_K}$, and value matrix $V \in \mathbb{R}^{L\times d_V}$ followed by the Eq. 1. Then, the query matrix and the key matrix are calculated for the autocorrelation, which can be shown as follows:

$$R_{Q,K}(\tau) = \lim_{L\to\infty} \frac{1}{L}\sum_{t=1}^{L} Q_t K_{t-\tau} \tag{7}$$

where $R_{Q,K}(\tau)$ denotes the time-delay similarities between the Q_t and the time delay τ of K_t. It is noted that $R_{Q,K}(\tau)$ are calculated by Fast Fourier Transform (FFT) to ensure efficient computation.

Once the auto-correlation scores are obtained, the subsequent step involves selecting the top k highest auto-correlation subseries, determined by the equation:

$$\tau_1, \tau_2, \ldots, \tau_k = \arg \underset{\tau\in\{1,\ldots,L\}}{TopK} (R_{Q,K}(\tau)) \tag{8}$$

Then, these subseries are normalized by using a softmax function, which can be formalized as:

$$S_{Q,K}(\tau_1), \ldots, S_{Q,K}(\tau_k) = \mathrm{Softmax}(R_{Q,K}(\tau_1), \ldots, R_{Q,K}(\tau_k)) \tag{9}$$

The overall equation of Auto-Correlation is calculated by rolling and the weight aggregation of V value matrix as follows:

$$\text{Auto-Correlation}(Q, K, V) = \sum_{i=1}^{k} Roll(V, \tau_i) S_{Q,K}(\tau_i) \tag{10}$$

where $Roll(X, \tau)$ represents the shifting operation applied to the series X with the time delay τ.

3.5 Our Proposed FAT

FAT is constructed by effectively integrating two dual different attention components: the LSA block and the ASA block. In the LSA block, we employ logsparse attention combined with local attention, efficiently extracting local information. Considering the auto-correlation mechanism in the ASA block, it is designed based on the periodicity of the time series and discovers aggregation of dependencies at the sub-series level. Furthermore, the series decomposition block separates the time series into its trend-cyclical and seasonal components. In the ASA Encoder blocks, the encoder focuses on extracting the seasonal subseries patterns. Then, the combined features reflect the local and global information in the time series. The specific steps of the proposed FAT are outlined as follows:

Step I: The original input features $X \in \mathbb{R}^{L \times k}$ undergo processing steps, accommodating the variable number of sensors k that can alter feature dimensions. A linear function is employed to ensure consistent input, converting the data to a fixed dimension d_{model}. We obtain the embedded i-th input $X_i \in \mathbb{R}^{L \times d_{model}}$. The positional information vectors are injected into the embedded input data, establishing the sequence length's temporal order. When added to the existing representation, these positional information vectors create context vectors. Then, a one-dimensional convolutional layer is designed to create a scale-changing input $X' \in \mathbb{R}^{L \times d_{model}/2}$ and extract high-level patterns before passing into different attention blocks.

Step II: The objective is to integrate two distinct self-attention blocks, specifically the LogSparse and Auto-Correlation encoder blocks. In the LogSparse encoder block, we adopt a logarithmic approach to mitigate computational complexity. The embedded data undergoes feature extraction in the LogSparse Encoder. Utilizing the LogSparse strategy with sparse input vectors facilitates the construction of query, key, and value matrices as in Eq. 2.

The Log Sparse comprises a Log Sparse self-attention layer, residual connection, layer normalization, and a feed-forward neural network (FFN). Local attention is combined with Log Spare attention to capture more information from subsequent time steps. Utilizing both types of attention will create more pathways [4] for the network to learn. Specifically, the Log-local Sparse self-attention layer captures temporal relationships among non-adjacent data within a time window sequence.

Step III: Considering the Auto-Correlation encoder block, the Auto-Correlation Encoder is stacked by encoder layers. Each encoder layer contains two main components: Auto-Correlation block and Series Decomposition block. In the Series Decomposition block, complex time series data is decomposed into trend-cyclical components and seasonal components by adapting the moving average. With the scaled input X', the Series Decomposition block is formalized as:

$$\begin{aligned} X_t &= \mathrm{AvgPool}(\mathrm{Padding}(X')) \\ X_s &= X' - X_t \end{aligned} \tag{11}$$

where X_t and X_s represent the trend components and seasonal components after the decomposition block of the l-th layer, respectively. The Padding$(\cdot)$ function maintains the consistent sequence length after feeding into AvgPool$(\cdot)$.

In the ASA block, it is supposed that the encoder block includes N encoder layers. The l-th encoder layer can be summarized as $(X')^{l}_{en} = \mathrm{Encoder}((X')^{l-1}_{en})$, where $l \in \{1, 2, \ldots, N\}$. The overall equations in ASA blocks are shown as follows:

$$\begin{aligned} S^{l,1}_{en}, \_ &= \mathrm{SeriesDecomp}(\text{Auto-Correlation}((X')^{l-1}_{en}) + (X')^{l-1}_{en}) \\ S^{l,2}_{en}, \_ &= \mathrm{SeriesDecomp}(\text{Feed-Forward}(S^{l,1}_{en}) + S^{l,1}_{en}) \end{aligned} \tag{12}$$

where $(X')^{l}_{en} = S^{l,2}_{en}$ represents the output of the l-th encoder layer and X^{0}_{en} is X'. $S^{l,1}_{en}$ and $S^{l,2}_{en}$ are seasonal components after the first Series Decomposition block and second Series Decomposition block in the l-th encoder layer, respectively. "_" is the removed trend component.

Step IV: The output the LSA ($O_{log} \in \mathbb{R}^{L \times d_{model}/2}$) and the ASA block ($O_{auto} \in \mathbb{R}^{L \times d_{model}/2}$) are concatenated to obtain the final output of the FAT model as follows:

$$O_{FAT} = \mathrm{Concat}(O_{log}, O_{auto}) W_O \tag{13}$$

where Concat$(\cdot)$ is a concatenation operator to combine the output of each block. W_O are learnable weight matrices of the combined output $O_{FAT} \in \mathbb{R}^{L \times d_{model}}$. The combined output is passed into a simple feed-forward layer to obtain RUL prediction. In addition, the relationship between time window length and RUL ground truth at a certain time can suffer from prediction error. It is necessary to design the RUL sequence. We suppose that the predictions of proposed model obtain $\{\hat{y}_m, \hat{y}_{m+1}, \cdots, \hat{y}_{m+L-1}\}$, and the corresponding actual RUL values are $\{y_m, y_{m+1}, \cdots, y_{m+L-1}\}$. The mean squared error (MSE) loss function of FAT on a window length L is defined as:

$$Loss = \frac{1}{L} \sum_{j=m}^{m+L-1} (y_j - \hat{y_j})^2 \tag{14}$$

Table 1. Description of C-MAPSS turbofan engine datasets.

Dataset	C-MAPSS turbofan engine			
	FD001	FD002	FD003	FD004
Training engines	100	260	100	249
Test engines	100	256	100	248
Operating Conditions	1	6	1	6
Fault modes	1	1	2	2

4 Experiments and Results

4.1 Experimental Setup

CMAPSS. The proposed method is evaluated using the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset provided by NASA. This dataset includes simulated degradation data from various engines. As shown in Table 1, the C-MAPSS dataset consists of four datasets: FD001, FD002, FD003, and FD004, which were recorded under various operational conditions and fault modes. Among these datasets, the operating conditions and fault modes of FD001 are the simplest, whereas FD004 presents the most complex and challenging scenarios, encompassing a variety of operational conditions and failure modes. Each sub-dataset consists of 26 variables during the operation of the engines, such as temperature, pressure, fan speed, etc. The detailed dataset description can be found in [18]. The training dataset covers the whole lifespan (run-to-failure) of each engine. During the operation, these aero-engines start to degrade until they reach a failure point (unhealthy status). On the contrary, the data collected from sensors in the test set was randomly selected at some time before engine failure. In this paper, the proposed work is trained on the datasets under various operational conditions and failure modes for RUL prediction, and the output of the FAT model is to predict the RUL value for the engines in the test set. Actual RUL values for these testing observations are provided to validate approaches.

Data Preprocessing. In the C-MAPSS dataset, there are certain measurement data that have constant values. This makes them uninformative in predicting RUL values. For instance, in the FD001 dataset, sensor indices 1, 5, 6, 10, 16, 18, and 19 contain constant values. These constant values are eliminated to prevent adverse effects on the model's learning process. This data processing method is implemented across all four datasets as described in [19].

The collected data from sensors is then normalized using the maximum-minimum normalization method. Equation (15):

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{15}$$

Table 2. Parameters setting of FAT architecture.

Layers	Hyper-Parameters	Configuration
1D Convolution	Kernel size	3
	Padding size	1
Encoder layer	Num of encoder	2
LSA layer	Num of heads	2
ASA layer	Num of heads	2
FFN layer	Hidden units	256

where X_{norm} is the normalized value of the data X, X_{max} and X_{min} denote the maximum and minimum values of the selected data X, respectively.

Time Window Processing. We used a fixed-sliding time window on the time series data with a stride of 1. However, randomly selecting a subset of observations for each engine in the test set limits the size of sequence length L_{seq}. Hence, we use padding to handle this issue. It is assumed that the minimum size of the testing sample on all engines is denoted L_{min}. In case where $L_{min} < L_{seq}$, padding $L_{seq} - L_{min}$ duplicated x_1 in front of $x_{1:L}$. This results in the formation of $\hat{x}_{1:L}$ as $\hat{x}_{1:L} = [x_1, \ldots, x_1, x_{1:L}]$. It is noteworthy that the last element of each sequence sample still remains, corresponding to the actual RUL of the sequence.

Evaluation Metrics and Hyper-Parameters Settings. We used common metrics for evaluating and reporting the performance of RUL models, namely root mean squared error (RMSE) and s-scoring function (S-Score). RMSE measures the average difference between the predicted values and ground truth. The formula is as follows:

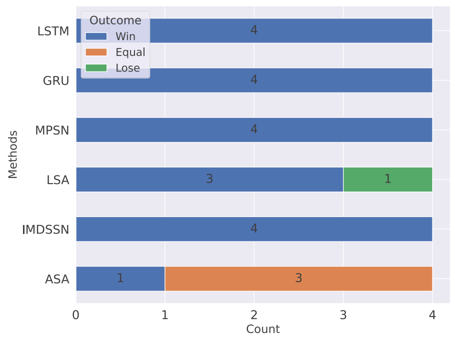
$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{16}$$

where y_i and $\hat{y}_i$ represent the actual value and the predicted value for the i-th sample in test set, respectively. N denotes the total number of testing samples.

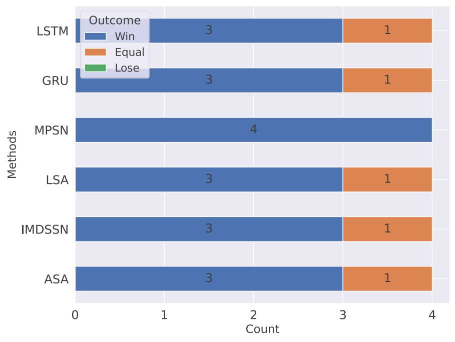
For the RUL task, the scoring function (S-Score) will impose a heavier penalty on predicted values occurring after the actual time of failure and a milder penalty on those predicted before it, which prevents situations where the predicted value significantly exceeds the true value [18]. The S-Score is expressed by:

$$\text{S-Score} = \sum_{i=1}^{N} s_i, \quad s_i = \begin{cases} e^{\frac{y_i - \hat{y}_i}{13}} - 1, & \text{if } \hat{y}_i < y_i \\ e^{\frac{y_i - \hat{y}_i}{10}} - 1, & \text{if } \hat{y}_i \geq y_i \end{cases} \tag{17}$$

Following the work presented at [17], parameters settings were as follows: Time window length was set to 30, batch size = 128, number of epochs was set to 60 with learning rate = 0.001 and dropout rate to 0.1. Adam optimizer was



(a)



(b)

Fig. 3. (a) Mann-Whitney tests between FAT and other approaches, measured by RMSE (b) Mann-Whitney tests between FAT and other approaches, measured by S-Score.

also used, and the MSE was used as the loss function. Similarly, following the same work, the parameters settings for the FAT model architecture are listed in Table 2.

4.2 Experimental Results and Discussion

We conducted experiments to compare the performance of FAT with several SOTA methods. We selected benchmark algorithms from different categories, such as Sequence-to-Sequence models and Transformer-based architectures. We opted for two SOTA architectures of the Sequence-to-Sequence model, including LSTM and GRU, in which we re-implemented the network architecture following the settings in [20]. We also compared with two latest Transformer-based architectures, namely MPSN and IMDSSN [17]. To demonstrate the effectiveness of the proposed fusion architecture, we compared the performance of FAT with those using either the LSA component or the ASA component. The experiments were run on 10 trials to reduce the randomness. All the experiments were implemented on a server with Quadro RTX 6000 and Intel® Xeon(R) Gold 5218R CPU.

We used the Mann-Whitney U test (significance level was set to 0.05) to compare the performance of FAT to each benchmark algorithm on all datasets. The test results are shown in Fig. 3(a) and 3(b). Our method consistently outperformed all other methods. Regarding RMSE, FAT surpassed LSTM, GRU, MPSN, and IMDSSN across all datasets. Specifically, FAT outperformed LSA on FD001, FD002, and FD004 but not on FD003. Compared to ASA, FAT performed better on FD001 and achieved similar performance on the other remaining datasets (FD002, FD003, and FD004). In S-Score, FAT consistently outperformed MPSN across all datasets. Specifically, on three datasets, FD001, FD002, and FD004, FAT surpassed other methods, including LSTM, GRU, LSA, IMDSSN, and ASA, while exhibiting comparable performance on FD003.

Table 3. The comparison with state-of-the-art methods in RMSE. Std: Standard deviation.

	FD001		FD002		FD003		FD004		Average	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Rank
LSTM	15.21 (6)	0.57	27.72 (5)	0.68	14.88 (6)	0.39	29.40 (7)	0.60	21.80	6.00
GRU	15.35 (7)	0.33	27.90 (6)	0.99	14.08 (3)	0.79	28.95 (6)	0.62	21.57	5.50
MPSN [17]	13.91 (5)	0.84	17.05 (3.5)	0.83	15.53 (7)	0.84	21.36 (4)	0.57	16.96	4.88
LSA	12.88 (3)	0.65	18.79 (4)	0.56	**12.61 (1)**	0.42	21.58 (5)	0.58	16.47	3.25
IMDSSN [17]	13.09 (4)	0.51	17.05 (3.5)	0.76	14.24 (5)	1.01	21.07 (3)	0.52	16.36	3.88
ASA	12.68 (2)	0.46	16.29 (2)	0.98	14.11 (4)	0.72	19.61 (2)	1.24	15.67	2.50
FAT (our)	**12.21 (1)**	0.49	**15.57 (1)**	0.84	13.07 (2)	0.54	**18.32 (1)**	1.52	**14.80**	**1.25**

Table 3 shows the mean and standard deviation of RMSE for FAT and benchmark algorithms across experimental datasets. FAT exhibits superior performance on three datasets (FD001, FD002, and FD004). Although FAT ranks second on FD003, the difference in RMSE between FAT and the top-ranking method (LSA) is not significant (13.07 vs. 12.61). FAT outperforms the Transformer-based approach IMDSSN by 6.72% on FD001, 8.68% on FD002, 8.22% on FD003, and 13.05% on FD004. In contrast, traditional Sequence-to-Sequence models (LSTM, GRU) perform the poorest in our experiment, highlighting the superiority of Transformer-based architectures. The average mean of FAT is significantly better than that of ASA and LSA (14.80 vs. 15.67 and 16.47). With an average ranking of 1.25, FAT outperforms ASA (2.50) and LSA (3.25), emphasizing the efficiency of the fusion-attention model over independent attention components.

Table 4. The comparison with state-of-the-art methods in S-Score. Std: Standard deviation.

	FD001		FD002		FD003		FD004		Average	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Rank
LSTM	391.47 (7)	67.05	12953.16 (6)	4050.13	416.76 (2)	97.44	7104.36 (6)	695.48	5216.44	5.25
GRU	369.75 (6)	29.38	17723.32 (7)	8963.73	418.58 (3)	120.99	7441.10 (7)	1431.60	6488.19	5.75
MPSN [17]	368.35 (5)	100.99	1773.56 (4)	530.93	1382.33 (7)	432.78	4589.26 (5)	1133.50	2028.38	5.25
LSA	317.67 (3)	56.19	2552.07 (5)	380.90	**364.13 (1)**	50.67	4355.57 (4)	814.75	1897.36	3.25
IMDSSN [17]	319.32 (4)	33.29	1744.69 (3)	529.90	643.48 (5)	332.22	4214.04 (3)	856.46	1730.38	3.75
ASA	296.74 (2)	53.87	1713.40 (2)	507.68	764.13 (6)	348.84	4101.11 (2)	1660.35	1718.85	3.00
FAT (our)	**247.86 (1)**	25.79	**1375.08 (1)**	506.76	540.85 (4)	302.37	**2329.69 (1)**	869.84	**1123.37**	**1.75**

In the S-Score results (refer to Table 4), FAT achieved the top position on three datasets: FD001, FD002, and FD004, whereas LSA achieved top performance on FD003 (with a margin of approximately 177 points over FAT for S-Score). FAT significantly outperforms LSA on FD001 (by around 70 points), FD002 (by approximately 1177 points), and FD004 (by about 2026 points). Compared to IMDSSN, FAT demonstrates an average mean improvement of 35.08%, with an average rank of 1.75 compared to IMDSSN's 3.75. In contrast to ASA, FAT shows a mean average improvement of 34.64%. The lower S-Score results of FAT indicate that our method yields more early prediction values and fewer late prediction values compared to other methods, providing increased practical applicability.

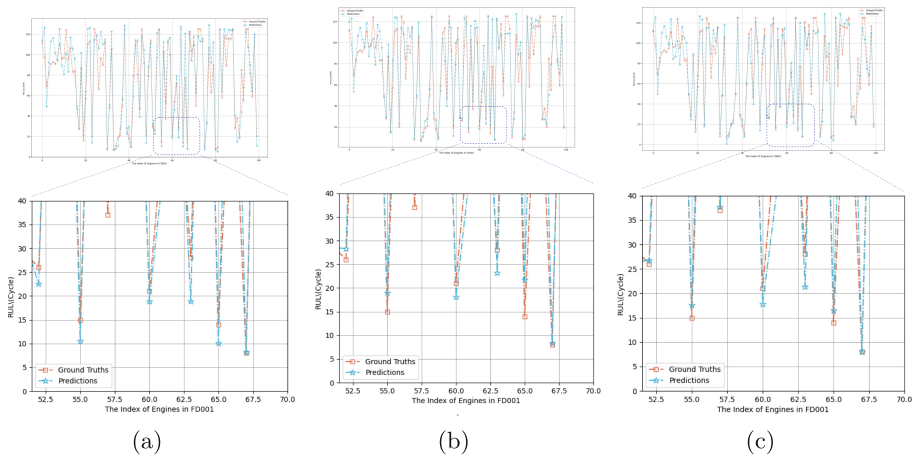


Fig. 4. (a) RUL predictions of MPSN on FD001 (b) RUL predictions of IMDSSN on FD001 (c) RUL predictions of FAT on FD001.

Due to space limitations, we only presented a part of the observations, which compares the proposed FAT method with MPSN and IMDSSN on the FD001. In Fig. 4, the single-point predictions of MPSN, IMDSSN, and the proposed FAT are depicted. We zoomed in on a segment of the prediction charts, revealing that FAT excels in RUL prediction performance. Notably, the predicted values of MPSN surpass the true RUL values, indicating late predictions with potentially adverse consequences (refer to Fig. 4(a)). Conversely, Fig. 4(b) illustrates that IMDSSN predicts values smaller than the true RUL values, signaling its ability to provide early predictions compared to MPSN. FAT further enhances prediction accuracy by generating early prediction values closely aligned with the actual RUL values (see Fig. 4(c)).

Discussion. The proposed FAT method aims to overcome the limitations of traditional transformer architecture, including its quadratic computational complexity, insensitivity to local regions, and periodic time series sequences. This is achieved by integrating LSA blocks and ASA blocks. The LSA blocks reduce computational complexity by focusing on a logarithmically sparse subset of the sequence. The complexity of LSA is given by $O(L(\log L)^2)$, compared to the traditional self-attention of $O(L^2)$. On the other hand, the ASA utilizes an autocorrelation mechanism to discover the similarity of sub-series based on the series periodicity, with a complexity of $O(L \log L)$. Therefore, the overall complexity of the FAT method is $O(L(\log L)^2)$ since $O(L(\log L)^2)$ grows faster than $O(L \log L)$. Although, in practice, the proposed model runs more slowly than component models, it addresses the limitations of the traditional Transformer architecture on time series sequences, such as seasonality or insensitivity to local regions. This has been demonstrated through our experiment to promote RUL prediction performance.

5 Conclusion

The accurate prediction of Remaining Useful Life (RUL) is crucial for ensuring system reliability and safety. The proposed method provides a promising solution to enhance RUL forecasting performance in complex systems. FAT overcomes limitations of conventional Transformers, addressing issues like insensitivity to local regions and computational complexity, resulting in significant performance improvement. Using public datasets, FAT proved to be capable of handling challenges like long-term dependencies and overall degradation trends. It is important to note that the RUL prediction problem addressed in this study is a supervised learning task that requires data with the same distributions for both the training and testing phases. However, the practical systems may not satisfy these requirements, which is a limitation of this work. In future work, we aim to propose a transfer learning-based method to solve real-world problems under the condition of insufficient labeled data.

Acknowledgements. This work received funding from Innovate UK and ADC Energy Limited through the Knowledge Transfer Partnership (KTP) project (No. 13518).

References

1. El-Thalji, I., Jantunen, E.: A summary of fault modelling and predictive health monitoring of rolling element bearings. Mech. Syst. Signal Process. **60**, 252–272 (2015)
2. Wang, Y., Zhao, Y., Addepalli, S.: Remaining useful life prediction using deep learning approaches: a review. Procedia Manufact. **49**, 81–88 (2020)
3. Serradilla, O., Zugasti, E., Rodriguez, J., Zurutuza, U.: Deep learning models for predictive maintenance: a survey, comparison, challenges and prospects. Appl. Intell. 1–31 (2022). https://doi.org/10.1007/s10489-021-03004-y
4. Li, S., et al.: Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
5. Sateesh Babu, G., Zhao, P., Li, X.-L.: Deep convolutional neural network based regression approach for estimation of remaining useful life. In: Navathe, S.B., Wu, W., Shekhar, S., Du, X., Wang, X.S., Xiong, H. (eds.) DASFAA 2016. LNCS, vol. 9642, pp. 214–228. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-32025-0_14

6. Li, X., Ding, Q., Sun, J.Q.: Remaining useful life estimation in prognostics using deep convolution neural networks. Reliab. Eng. Syst. Saf. **172**, 1–11 (2018)
7. Zhang, J., et al.: A parallel hybrid neural network with integration of spatial and temporal features for remaining useful life prediction in prognostics. IEEE Trans. Instrum. Meas. **72**, 1–12 (2022)
8. Huang, C.G., Huang, H.Z., Li, Y.F.: A bidirectional LSTM prognostics method under multiple operational conditions. IEEE Trans. Industr. Electron. **66**(11), 8792–8802 (2019)
9. Wu, Y., Yuan, M., Dong, S., Lin, L., Liu, Y.: Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. Neurocomputing **275**, 167–179 (2018)
10. Huang, C.G., Huang, H.Z., Li, Y.F.: A bidirectional LSTM prognostics method under multiple operational conditions. IEEE Trans. Industr. Electron. **66**(11), 8792–8802 (2019)
11. Xia, J., Feng, Y., Teng, D., Chen, J., Song, Z.: Distance self-attention network method for remaining useful life estimation of Aeroengine with parallel computing. Reliab. Eng. Syst. Saf. **225**, 108636 (2022)
12. Vaswani, A.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
13. Mo, Yu., Wu, Q., Li, X., Huang, B.: Remaining useful life estimation via transformer encoder enhanced by a gated convolutional unit. J. Intell. Manuf. **32**(7), 1997–2006 (2021). https://doi.org/10.1007/s10845-021-01750-x
14. Ma, Q., Zhang, M., Xu, Y., Song, J., Zhang, T.: Remaining useful life estimation for turbofan engine with transformer-based deep architecture. In: 2021 26th International Conference on Automation and Computing (ICAC), pp. 1–6. IEEE (2021)
15. Chen, D., Hong, W., Zhou, X.: Transformer network for remaining useful life prediction of lithium-ion batteries. IEEE Access **10**, 19621–19628 (2022)
16. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
17. Zhang, J., Li, X., Tian, J., Luo, H., Yin, S.: An integrated multi-head dual sparse self-attention network for remaining useful life prediction. Reliab. Eng. Syst. Saf. **233**, 109096 (2023)
18. Saxena, A., Goebel, K., Simon, D., Eklund, N.: Damage propagation modeling for aircraft engine run-to-failure simulation. In: 2008 International Conference on Prognostics and Health Management, pp. 1–9. IEEE (2008)
19. Zhang, J., Jiang, Y., Wu, S., Li, X., Luo, H., Yin, S.: Prediction of remaining useful life based on bidirectional gated recurrent unit with temporal self-attention mechanism. Reliab. Eng. Syst. Saf. **221**, 108297 (2022)
20. Zheng, S., Ristovski, K., Farahat, A., Gupta, C.: Long short-term memory network for remaining useful life estimation. In: 2017 IEEE International Conference on Prognostics and Health Management (ICPHM), pp. 88–95. IEEE (2017)
21. Wu, H., Xu, J., Wang, J., Long, M.: Autoformer: decomposition transformers with auto-correlation for long-term series forecasting. Adv. Neural. Inf. Process. Syst. **34**, 22419–22430 (2021)
22. Zhang, Z., Song, W., Li, Q.: Dual-aspect self-attention based on transformer for remaining useful life prediction. IEEE Trans. Instrum. Meas. **71**, 1–11 (2022)
23. Zeng, A., Chen, M., Zhang, L., Xu, Q.: Are transformers effective for time series forecasting? In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 9, pp. 11121–11128 (2023)

Conditional Probability-Based Feature Embedding for Genomic Sequence Data

Parashjyoti Borah[1] and Aparajita Dutta[2](✉)

[1] Indian Institute of Information Technology Guwahati, Guwahati 781015, Assam, India
parashjyoti@hotmail.com
[2] National Institute of Technology Silchar, Silchar 788010, Assam, India
aparajita.dutta@cse.nits.ac.in

Abstract. Embedding techniques for categorical attributes play a pivotal role in the performance of machine learning inference, especially for downstream analysis of genomic sequence data. Categorical data do not convey quantitative information to directly influence the model parameters in mapping input to the desired output. Two commonly used state-of-the-art embedding techniques are one-hot embedding and binary embedding, which embed the values of the categorical attributes in vectors of 0s and 1s. One-hot treats values of a particular attribute uniformly, or in other words, it considers uniform probability distribution for the random variable of the categorical attribute. On the other hand, binary embedding assigns some ordinal characteristics to the data without considering the probability distribution. Both of these techniques do not embed the underlying semantics of the genomic features. In this study, we propose two conditional probability–based feature embedding techniques for categorical data that utilize the probability distribution of the data. Furthermore, the proposed embedded vector lengths are based on the number of classes in the training data rather than the unique values an attribute can take.

Keywords: Feature Embedding · Gene Sequence · Conditional Probability

1 Introduction

In recent years, the advent of high-throughput sequencing technologies has revolutionized the field of genomics, enabling researchers to gather vast amounts of genomic sequence data with unprecedented speed and accuracy. This wealth of data has opened new avenues for understanding the intricate relationships between genetic information and biological processes, as well as for developing predictive models for various genomic applications.

Traditionally, positional matrices have been widely used to represent genomic sequences, where each position in the sequence is represented by a vector containing information about the presence or absence of certain sequence motifs or features [1, 2]. However, these traditional approaches heavily rely on handcrafted features, which are inherently limited by human knowledge of the problem domain.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 302–314, 2025.
https://doi.org/10.1007/978-3-031-78192-6_20

To overcome the limitations of handcrafted features, researchers have explored alternative approaches such as one-hot encoding and binary feature embeddings. While these methods provide a more flexible representation of genomic sequences, they often lack semantic information about the input data, leading to suboptimal performance in downstream machine learning tasks [3].

Furthermore, distributed feature embeddings such as word2vec have shown promising results in capturing semantic relationships in natural language processing tasks. However, these embeddings are not directly applicable to genomic sequence data and cannot be easily mapped back to the input space, hindering interpretability and feature interpretation [4]. In biological downstream tasks, this mapping is particularly important because mankind has limited existing knowledge regarding the biological processes.

In this manuscript, we propose two novel approaches for generating embeddings for genomic sequence data for the downstream task of identifying splice sites. Our approach aims to circumvent the above mentioned limitations by providing an efficient embedding that is lightweight, faster to compute, and can be seamlessly mapped back to the input space for improved feature interpretation. Although the proposed work analyses the performance of the novel embeddings on a single downstream task of splice site identification, we can apply similar embeddings for identification of several other cell variables like polyadenylation sites, promoter, and enhancer regions. By leveraging the intrinsic structure and patterns present in genomic sequences, our method promises to enhance the interpretability and performance of machine learning models for various genomic applications. Moreover, these methods hold promise for potential application beyond gene sequence data, extending their utility to various other domains.

2 Related Work

Numerous techniques in data analysis and data mining rely on embedding data in a Euclidean space. However, when dealing with symbolic datasets, such as biological sequence data from high-throughput sequencing assays, traditional embedding methods like binary and k-mer count vectors might prove too high-dimensional or coarse to effectively glean insights from the data. Furthermore, approaches like Multidimensional Scaling (MDS) [5] and Node2Vec [6] may not scale well to large datasets since they require a full recomputation of the embedding for unclassified test data.

In the realm of genomic sequence representation, many strategies hinge on features derived from biochemical properties of nucleotides or amino acids, encompassing aspects like thermodynamic stability, structural flexibility, hydrophobicity, charge, and polarity [7–9]. Common methods, such as k-mer count vectors and binary vectors, while simple, often result in excessively high-dimensional representations.

The majority of molecular sequence classification methods rely on manually crafted features, like position-specific scoring matrices (PSSM) [10]. Despite their prevalence, these features, including Position Weight Matrix (PWM) and others like average molecular weight and iso-electric point value, have limitations that can compromise model accuracy [11]. To address these drawbacks, techniques like positional-average molecular weight have been introduced to discern the significance of amino acid positions within protein sequences [12].

Additionally, Natural Language Processing (NLP)-based methods, such as word2vec [13] and doc2vec [14], have gained traction in constructing numeric embeddings for molecular sequences. These approaches, including dna2vec [15] and kmer2vec [16], treat sequences as documents and leverage fixed-length fragments (k-mers) as words. Furthermore, techniques like Gene2vec [17] and SPVec [18] offer alternative representations for genes and protein sequences, respectively. Despite their promising results, there remains scope for enhancing efficiency and facilitating the interpretation of learned features by mapping embeddings back to the input space.

3 Background

We chose the task of splice site prediction to analyze the embeddings. In many eukaryotic organisms, protein-coding genes are fragmented by sequences known as introns. These introns are excised from the exonic sequences through a process called splicing, which takes place concurrently with transcription [19]. This process yields mature substrates that can be translated into proteins. Splicing entails the precise removal of introns at sites known as splice sites or junctions, situated between exons and introns, followed by the joining of exons. Therefore, the identification of splice junctions is crucial for understanding gene structure and functionality.

We compare the proposed embedding technique to the two most widely used embedding techniques that can be mapped back to the input feature space, namely binary embedding and one-hot embedding. Furthermore, we compare our proposed embeddings with the two baseline embeddings for the mentioned classification task using three classifiers, namely support vector machine (SVM) [20], least squares support vector machine (LS-SVM) [21], and one-dimensional convolutional neural network (1D-CNN) [22]. SVM is a highly effective traditional machine learning method known for its versatility in various applications. LS-SVM extends SVM by maximizing the margin between class hyperplanes, but it emphasizes improving the separation between hyperplanes close to the classes rather than just the boundary hyperplanes. In contrast, 1D-CNNs are preferred deep learning techniques for analyzing gene sequence data. 1D-CNNs utilize convolution operations to effectively extract features from the local sequential patterns present in genomic sequences, making them well-suited for genomic data analysis. The baseline methods and classifiers with the notations that will appear in the subsequent discussions are described next.

All the vectors and the matrices are written in boldface. The feature matrix $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N)^T$ is the collection of the input feature vectors $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \cdots, x_{iD})^T$, where N is the cardinality of $\boldsymbol{X}$, D is the dimensionality of the data and $(\cdot)^T$ is the matrix transpose. $\boldsymbol{y} = (y_1, y_2, \cdots, y_N)^T$ is the vector of class labels where $y_i \in \{y_1, \cdots, y_C\}$ and C is the number of classes in the dataset. For any categorical feature j, K_j is the categorical value where $k = 1, 2, \cdots, K_j$ (i.e. feature j can take a value from the K_j possible values).

3.1 Binary Embedding

Binary embedding assigns an integer (usually starting with 1) to each of the categorical values of the attribute and embeds it with its corresponding binary equivalent. The number of digits required to encode any categorical attribute j is $\lceil log_2 K \rceil$. In other words, we represent the binary encoded vector of v_{jk} as $\boldsymbol{b}_{jk} = Binary(v_{jk})$ where $\boldsymbol{b}_{jk} \in \{0, 1\}^{\lceil log_2 K \rceil}$ and its elements are the digits of the binary equivalent of k.

3.2 One-Hot Embedding

One-hot embedding also uses a Boolean encoding technique for any categorical attribute j using K_j digits. The encoded vector $\boldsymbol{o}_{jk}$ has 1 at the k^{th} element and all the other elements are 0s. We represent the one-hot embedded vector as $\boldsymbol{o}_{jk} = Onehot(v_{j,k})$ where $\boldsymbol{o}_{jk} \in \{0, 1\}^{K_j \times 1}$ and its k^{th} element is 1 and all others 0.

3.3 Support Vector Machine (SVM)

Originally SVM [20] is a binary classification method that is based on the idea of maximizing the distance between the boundary hyperplanes of the two classes, which are expressed as $\varphi(\boldsymbol{x})^T \boldsymbol{\omega} + b = \pm 1$ with $+1$ and -1 representing the positive and the negative class respectively. The classifier is built as $\varphi(\boldsymbol{x})^T \boldsymbol{\omega} + b = 0$ at unit relative distance from the boundary hyperplanes. Here, $\boldsymbol{\omega}, b$ are the model parameters and $\varphi(\cdot)$ is a feature mapping function. For points falling on the wrong side of the boundary planes SVM employs the Hinge loss function. The optimization problem of SVM is given below:

$$\begin{gathered} \min_{\boldsymbol{\omega}, \xi_i} \frac{1}{2} \|\boldsymbol{\omega}\|^2 + \lambda \sum_{i=1}^{N} \xi_i \\ s.t. y_i \left(\varphi(\boldsymbol{x})^T \boldsymbol{\omega} + b \right) \geq 1 - \xi_i, for\ i = 1, 2, \cdots, N \\ \xi_i \geq 0. \end{gathered} \tag{1}$$

where ξ_i is the slack variable and λ is the penalty parameter for the trade-off between structural risk and empirical risk. Traditionally, the above optimization problem is transformed into its corresponding dual formulation and the solution is obtained in the dual space.

3.4 Least Squares Support Vector Machine (LS-SVM)

LS-SVM [21] is the least squares version of the original SVM that employs least squares loss function instead of the Hinge loss function in SVM. Additionally, unlike SVM, LS-SVM constrains the class hyperplanes by keeping them proximal to the points of their respective classes with an equality constraint.

$$\begin{gathered} \min_{\boldsymbol{\omega}, \xi_i} \frac{1}{2} \|\boldsymbol{\omega}\|^2 + \lambda \sum_{i=1}^{N} \xi_i^2 \\ s.t. y_i \left(\varphi(\boldsymbol{x})^T \boldsymbol{\omega} + b \right) = 1 - \xi_i, for\ i = 1, 2, \cdots, N. \end{gathered} \tag{2}$$

The squared loss function of LS-SVM and its proximity constraint make it highly sensitive to noise and outliers. Solution for LS-SVM is obtained in the dual space requiring to solve a system of linear equations.

3.5 One-Dimensional Convolutional Neural Network (1D-CNN)

A 1D-CNN [22] architecture comprises an input layer, one or more 1D convolutional layers (each potentially followed by a pooling layer), dense layers, and an output layer. 1D CNNs are characterized by the convolution operation conducted along a single dimension within the convolutional layers. Each 1D convolutional layer is composed of multiple filters (or kernels) that slide over the input data, generating feature maps. Typically, these filters are vectors of dimension ($F \times 1$), where F is the kernel size. Strides are occasionally utilized to enhance the capture of local and global patterns by downsampling the input, reducing data dimensionality, and extracting features across various scales. In 1D CNNs, the stride parameter dictates the step size of the filter's movement along the input sequence. For an input sequence (input vector) $\boldsymbol{x}$, a filter $\boldsymbol{\omega} = (\omega_1, \omega_2, \cdots, \omega_F)^T$ and stride ε, the convolution operation to produce the output vector $\boldsymbol{c} = (c_1, c_2, \cdots, c_z)^T$ can be defined as:

$$c_i = \sum_{j=1}^{F} \omega_j x_{(i-1)\varepsilon+j} \tag{3}$$

where, $i = 1, 2, \cdots, z$ for $z = \frac{N-F}{\varepsilon} + 1$. After convolution, an activation function is applied to introduce non-linearity. Pooling layers are used to reduce the dimensionality of the feature maps, retaining essential features while reducing computational complexity. Subsequent to one or more convolutional and pooling layers, fully connected (dense) layers are used to map the extracted features to the output classes or predictions. These layers are typically used at the end of the network.

4 Proposed Embedding Approaches

In this section, we propose two approaches for embedding categorical data. Unlike the two popular embedding techniques discussed in the above section, the proposed approaches utilize probability information to retain influential characteristics of the attribute values in generalization.

4.1 Embedding Based on Conditional Probability (A Linear Function)

Consider that the attribute j is categorical, which is the random variable R_j that takes values from the set $\{v_{jk} | k = 1, 2, \cdots, K_j\}$ and the conditional probability $P\big(v_{jk} | y_n\big)$ is its probability distribution. Throughout this paper, whenever y is subscripted with n or C, it refers to a particular class among the C classes and not the target of a sample. Now, the conditional probability of v_{jk} given that a training instance belongs to the class y_n, is expressed as,

$$P\big(v_{jk} | y_n\big) = \frac{\left|S_{v_{jk}}^{(n)}\right|}{\left|S_{y_n}\right|} \tag{4}$$

where $S_{v_{jk}}^{(n)} = \{\boldsymbol{x}_i | y_i = y_n, x_{ij} = v_{jk}\}$ and $S_{y_n} = \{\boldsymbol{x}_i | y_i = y_n\}$ and $|\cdot|$ is the cardinality of a set. The embedding vector for v_{jk} is

$$\boldsymbol{p}_{jk} = \left(P\left(v_{jk}|y_1\right), \cdots, P\left(v_{jk}|y_C\right)\right)^T \tag{5}$$

The encoding is based on linear conditional probability function and hence normalization of the attribute may not be required.

4.2 Embedding Based on Exponential Function of Conditional Probability

The proposed embedding technique discussed in the above subsection utilizes the conditional probability function linearly. In other words, the significance of a particular categorical value v_{jk} for a particular attribute j varies linearly with $P\left(v_{jk}|y_n\right)$. However, sometimes the information carried by v_{jk} are not a linear function of $P\left(v_{jk}|y_n\right)$. Here we propose a nonlinear function of conditional probability that exhibits similar properties as the sigmoid function. The probability function is defined below:

$$\hat{P}\left(v_{jk}|y_n\right) = \frac{1}{1 + e^{a\left(1 - b\frac{\left|S_{v_{jk}}^{(n)}\right|}{\left|S_{y_n}\right|}\right)}} \tag{6}$$

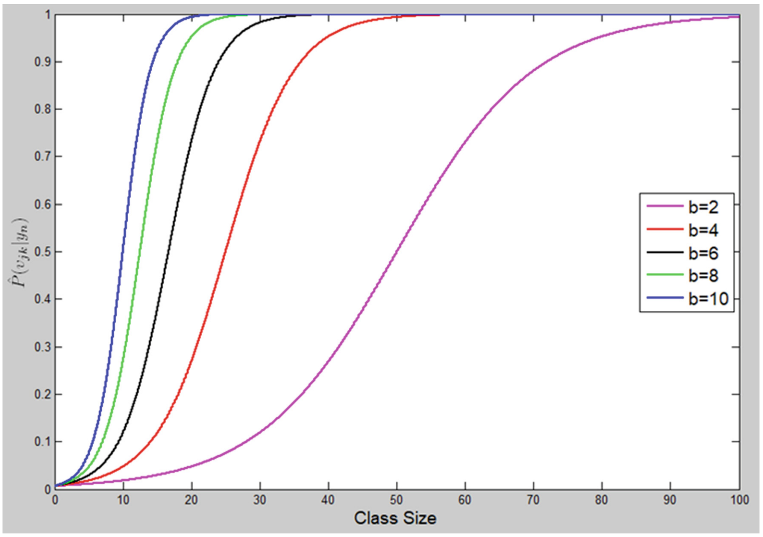
The parameters a and b controls the range, shape and orientation of the function. Although these parameters can be tuned, we suggest $a = 5$ and $b = K_j$. There is a strong relation between these parameters with the function shape and orientation. The function for $a = 5$ and different values of b is shown in Fig. 1. It is observable from Fig. 1 (a) and (b) that the proposed function retains its shape and orientation irrespective of the size of the class y_n. It empirically proves that the proposed function is consistent with class size. Further, by setting $a = 5$ the range is distributed evenly between 0 and 1 among different $\left|S_{v_{jk}}^{(n)}\right|$ with varying b values. By setting $b = K_j$, we can derive the interpretation that, with a higher value of K_j a smaller $\frac{\left|S_{v_{jk}}^{(n)}\right|}{\left|S_{y_n}\right|}$ ration is representative of the inference.

Finally, the embedded vector, whose length is equal to the number of classes in the training data, is expressed as,

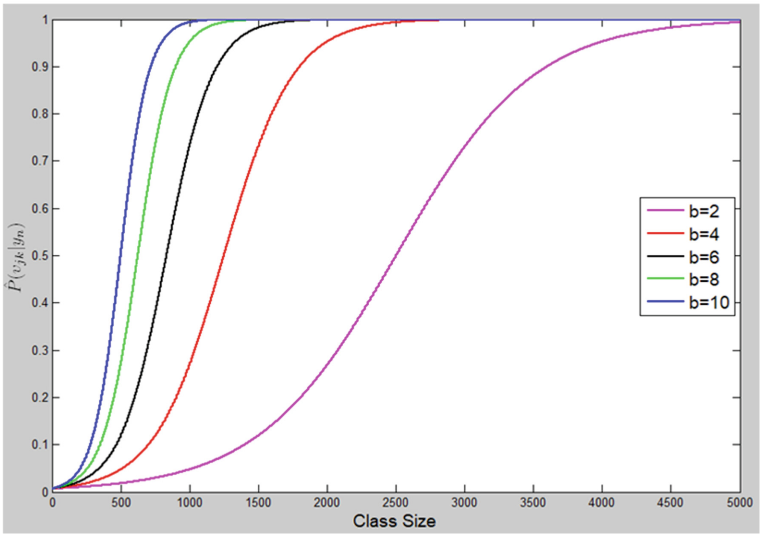
$$\hat{\boldsymbol{p}}_{jk} = \left(\hat{P}\left(v_{jk}|y_1\right), \cdots, \hat{P}\left(v_{jk}|y_C\right)\right)^T \tag{7}$$

4.3 Discussion

The size of the embedded vector for binary embedding of attribute j is determined by $\lceil log_2 K_j \rceil$, while for one-hot embedding, it is K_j. Based on the value of K_j and the number of categorical attributes, dimensionality of the embedded dataset may increase significantly. In contrast, the proposed embedding approaches utilize a vector length C, where C represents the number of classes. In most of the real-world data, it is observed that $K_j \gg C$ and subsequently $\lceil log_2 K_j \rceil > C$. As a result, one advantage of the proposed



(a)



(b)

Fig. 1. Graphical representation of $\hat{P}(v_{jk}|y_n)$ with $a = 5$ and for different values of b. (a) For class size is 100, (b) For class size 5000.

approaches lies in potentially reducing the dimensionality of the embedded data compared to binary and one-hot embeddings. Subsequently, space and computational cost could be significantly reduced with the proposed embedding techniques. This advantage is supported by empirical evidence detailed in the subsequent section of this study.

Furthermore, one-hot embedding uniformly treats each value of a categorical attribute, implying a uniform probability distribution across the values. In contrast, binary embedding assigns an ordinal characteristic to the data by assigning binary equivalents of decimal values without regard to the underlying probability distribution of the categorical variable. However, the proposed approaches in our study take into account the probability distribution of categorical values. This consideration is crucial as it helps preserve valuable information about the influential characteristics inherent in the attribute values during the generalization process. By incorporating these probability distributions, our embeddings aim to capture and retain meaningful patterns and relationships within the data, enhancing the overall effectiveness and interpretability of the models trained on such encoded representations.

5 Empirical Study

To validate the applicability of the proposed approaches, numerical studies are conducted on gene sequence data containing all categorical attributes. In this section, we present first the description of the dataset for this study, then the hardware and software environment under which the study is carried out, along with other experimental settings, and finally, a discussion of the experimental results.

5.1 Dataset

We extracted the introns across all transcripts of the protein coding genes annotated in the latest version of GENCODE [23] annotations (based on human genome assembly version GRCh38). The upstream and downstream flanking regions of 40 nucleotides were extracted at each splice junction, which lies at the two ends of the introns. A flanking region of 40 nucleotides was considered because most of the known splicing features mentioned in the literature lie within this region [24]. This comprises the positive samples of our dataset. Each sample contains 164 nucleotide bases. A nucleotide comprises of the bases A, T, G, C, or N. Therefore, we have considered each nucleotide, in the sequences of the dataset, as a categorical feature. Hence, each sample has a total of 164 features.

The splice junctions mostly contain the consensus pattern GT and AG at the donor [‘5’ end of the intron] and acceptor [‘3’ end of the intron] splice sites, respectively. The negative samples were generated randomly from the genome sequences such that the center of the negative samples contains the most common consensus pattern GT and AG at the splice junctions. However, the junctions were chosen such that they did not belong to the set of true splice junctions. The length was considered same as that of the positive samples. We obtained a total of 293,889 positive samples from the annotations. However, due to resource constraints, we have considered a random sample size of 10,000 samples for all the results and analysis shown below.

5.2 Experimental Setup and Hyperparameter Tuning

Binary embedding and one-hot embedding are commonly preferred embedding techniques for categorical attributes. SVM is one of the most efficient traditional machine learning classification methods and has proven to be efficient in a wide range of applications. LS-SVM is another efficient classification algorithm that is based on SVM’s principle of maximizing the margin between two class hyperplanes. However, unlike the boundary hyperplanes of SVM, LS-SVM maximizes the separation between two hyperplanes that are proximal to the classes. On the other hand, compared to traditional machine learning methods, 1D-CNNs are one of the most widely used deep learning techniques for gene sequence data analysis. Genomic sequence data exhibit sequential features comprising consecutive nucleotides, and 1D-CNNs are particularly well-suited to extract features like local sequential patterns using convolution operations. In this study, we compare the performance of SVM, LS-SVM, and 1D-CNN on the genomic sequence dataset with the proposed embedding techniques against binary and one-hot embedding.

The experiments for this study were conducted using a workstation PC equipped with 128 GB of DDR4 RAM and an Intel Xeon W-2295 18-core processor. The SVM and LS-SVM models were executed using MATLAB 2022 computational software, while the 1D-CNN models were run on Python 3.10.4. To implement the 1D-CNN model, the standard TensorFlow library package is used.

The robust Gaussian kernel is used for nonlinear feature mapping, which is expressed as $f\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = e^{\frac{-\|x_i - x_j\|^2}{2\sigma^2}}$. The kernel parameter σ is tuned from the set $\left\{2^{-2}, 2^{-1}, 2^0, 2^1, 2^2, 2^4, 2^6, 2^8, 2^{10}, 2^{12}\right\}$ and the hyperparameter λ is tuned from the set $\left\{2^{-1}, 2^0, 2^1, 2^2, 2^3, 2^4, 2^5, 2^6, 2^8, 2^{10}\right\}$. On the other hand, the 1D-CNN architecture consists of the following layers in sequence: an input layer, two 1D convolution layers each followed by a max-pooling layer with a pool size of 2, a global max-pooling layer, a dropout layer, a dense layer with 64 neurons, and an output layer. The hyperparameters tuned for the 1D-CNN include batch sizes of $\{16, 32, 64\}$, filter sizes of $\{16, 32, 64\}$ for the first 1D convolution layer and $\{32, 64, 128\}$ for the second, dropout rates of $\{0.1, 0.3, 0.5\}$, and learning rates of $\{0.001, 0.01, 0.1\}$. The ReLU activation function is used for the 1D convolution and dense layers, while the softmax activation function is used for the output layer. The model is trained for up to 100 epochs with early stopping criteria in place. For the convolution operation, we use a stride $\varepsilon = k$, where k is the embedded vector length for the respective embedding. This ensures the convolution operation slides over the embedded genomic sequence with a vector of length k. Consequently, the kernel size is considered as $3k \times 1$ since three consecutive nucleotides carry a biological significance by forming a codon which subsequently codes an amino acid [25]. The model is optimized using the Adam optimizer with binary cross-entropy loss. Accuracy is used as the validation metric.

The hold-out validation strategy is applied for parameter tuning with 75:25 split of the training dataset for train:validation set. The evaluation metrics to evaluate performance of the models on the datasets are accuracy, precision, recall, and F1-score.

5.3 Experimental Results

For the dataset discussed above, data dimensionality after generating the embedding using different embedding techniques is provided in Table 1. The proposed approaches achieve significantly reduced dimensionality as compared to binary and one-hot embedding. Subsequently, space and computational cost could be significantly reduced with the proposed embedding techniques.

Table 1. Dimensionality of embedded data

Embedding	No. of features
One-Hot ($\boldsymbol{o}_{jk}$)	820
Binary ($\boldsymbol{b}_{jk}$)	492
Linear Conditional Probability-based ($\boldsymbol{p}_{jk}$)	328
Nonlinear Conditional Probability-based ($\hat{\boldsymbol{p}}_{jk}$)	328

Generalization performances of SVM, LS-SVM and 1D-CNN on feature embedded test data using different embedding techniques are presented in Table 2, Table 3, and Table 4 respectively. It can be observed from Tables 2, 3 and 4 that the proposed approaches could achieve better or comparable generalization performance for the dataset in this study.

Table 2. Generalization results of SVM on the dataset with different embedding techniques.

Embedding	Accuracy	Precision	Recall	F1-score
Binary ($\boldsymbol{b}_{jk}$)	94.3068	0.95851	0.92623	0.942093
One-Hot ($\boldsymbol{o}_{jk}$)	94.6989	**0.962056**	0.930684	0.94611
Linear Conditional Probability-based ($\boldsymbol{p}_{jk}$)	**94.9751**	0.961764	0.936743	**0.949088**
Nonlinear Conditional Probability-based ($\hat{\boldsymbol{p}}_{jk}$)	94.9305	0.960716	**0.936921**	0.948669

Table 3. Generalization results of LS-SVM on the dataset with different embedding techniques.

Embedding	Accuracy	Precision	Recall	F1-score
Binary ($\boldsymbol{b}_{jk}$)	94.9127	0.958189	0.939237	0.948619
One-Hot ($\boldsymbol{o}_{jk}$)	95.2869	**0.966245**	0.938525	0.952183
Linear Conditional Probability-based ($\boldsymbol{p}_{jk}$)	95.2869	0.964031	**0.940841**	0.952295
Nonlinear Conditional Probability-based ($\hat{\boldsymbol{p}}_{jk}$)	**95.3225**	0.965757	0.939772	**0.952587**

Accuracy is a neural metric when the dataset has balanced class sizes and the dataset in our study has equal class sizes for the positive and the negative class. The proposed embedding achieved the best performance in terms of accuracy. On the other hand, precision, recall and F1-score are more representative of the positive class. Biomedical data are more concerned about correctly classifying the positive class and from the results

Table 4. Generalization results of 1D-CNN on the dataset with different embedding techniques.

Embedding	Accuracy	Precision	Recall	F1-score
Binary ($\boldsymbol{b}_{jk}$)	88.5602	0.897064	0.871169	0.883927
One-Hot ($\boldsymbol{o}_{jk}$)	89.2017	0.90367	0.877584	0.890436
Linear Conditional Probability-based ($\boldsymbol{p}_{jk}$)	**94.1732**	**0.960438**	**0.921418**	**0.940524**
Nonlinear Conditional Probability-based ($\hat{\boldsymbol{p}}_{jk}$)	93.309	0.959886	0.903956	0.931082

the proposed embedding approaches could outperform binary and one-hot embedding in terms of recall and F1-score.

The best accuracy scores are highlighted in boldface in Tables 2, 3, and 4, showcasing the effectiveness and consistency of the proposed approaches as seen in their generalization performances. Interestingly, Table 4 reveals that the 1D-CNN struggles with binary and one-hot embeddings but performs comparably well with the proposed embedding. This issue arises from the inherent limitations of binary and one-hot embeddings in gene sequence data. These embeddings create sparse, high-dimensional representations that are difficult for a 1D-CNN model to learn from effectively, especially with a relatively small dataset of only 10,000 samples. They fail to capture the intricate patterns and relationships within the sequence data, leading to suboptimal feature extraction by the convolutional layers. In contrast, the proposed embedding offers a more compact and informative representation of the gene sequences by utilizing the probability distribution of the data to capture essential features and relationships more effectively. This enhances learning and generalization by the 1D-CNN, resulting in improved performance. The effectiveness of the proposed embedding across both traditional machine learning methods (SVM, LS-SVM) and the deep learning approach (1D-CNN) underscores its robustness and capability to comprehensively capture the underlying structure of the data. We conclude that generalization performance-wise the proposed approaches are promising and applicable in genomic sequence-based classification tasks.

6 Conclusion

In this work, we have proposed two novel embedding techniques based on conditional probability for categorical features. We have analyzed the performance of the proposed embeddings on the task of splice site prediction considering the nucleotides in the genomic sequence as categorical features. Furthermore, we have compared the proposed embeddings with two widely used embedding techniques. We see that the proposed embeddings are lightweight as they comprise less number of features. Hence, they are faster in execution and reduced space complexity. Despite comprising a lesser number of features, they perform comparable to the baseline embeddings. We validated this using three different classifiers, namely SVM, LS-SVM and 1D-CNN. In the future scope of this work, the evaluation of these embeddings can be extended across various classification tasks beyond genomic sequences. Additionally, their performance can be compared with datasets from different domains. Furthermore, their effectiveness with other machine learning techniques including advanced deep learning models and classifiers can be explored to gain further insights into their versatility and applicability. Also, incorporating newer embedding techniques into the comparison will broaden the applicability of the proposed approaches.

References

1. Gershenzon, N.I., Stormo, G.D., Ioshikhes, I.P.: Computational technique for improvement of the position-weight matrices for the DNA/protein binding sites. Nucleic Acids Res. **33**(7), 2290–2301 (2005)
2. Hartmann, H., Guthöhrlein, E.W., Siebert, M., Luehr, S., Söding, J.: P-value-based regulatory motif discovery using positional weight matrices. Genome Res. **23**(1), 181–194 (2013)
3. Seger, C.: An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing (2018)
4. Di Gennaro, G., Buonanno, A., Palmieri, F.A.N.: Considerations about learning Word2Vec. J. Supercomput. **77**(11), 12320–12335 (2021). https://doi.org/10.1007/s11227-021-03743-2
5. Krzanowski, Wojtek. *Principles of multivariate analysis.* Vol. 23. OUP Oxford, 2000
6. Grover, Aditya, and Jure Leskovec. “node2vec: Scalable feature learning for networks.” In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864. 2016
7. Sciabola, S., Cao, Q., Orozco, M., Faustino, I., Stanton, R.V.: Improved nucleic acid descriptors for siRNA efficacy prediction. Nucleic Acids Res. **41**(3), 1383–1394 (2013)
8. Sarda, Deepak, Gek Huey Chua, Kuo-Bin Li, and Arun Krishnan. “pSLIP: SVM based protein subcellular localization prediction using multiple physicochemical properties.” *Bmc Bioinformatics* 6 (2005): 1–12
9. Cai, C. Z., L. Y. Han, Zhi Liang Ji, X. Chen, and Yu Zong Chen. “SVM-Prot: web-based support vector machine software for functional classification of a protein from its primary sequence.” *Nucleic acids research* 31, no. 13 (2003): 3692–3697
10. Ali, Sarwan, Babatunde Bello, Prakash Chourasia, Ria Thazhe Punathil, Yijing Zhou, and Murray Patterson. “PWM2Vec: An efficient embedding approach for viral host specification from coronavirus spike sequences.” *Biology* 11, no. 3 (2022): 418
11. Mohamed, Shakir, David Rubin, and Tshilidzi Marwala. “Multi-class protein sequence classification using fuzzy ARTMAP.” In *2006 IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, pp. 1676–1681. IEEE, 2006
12. Saha, Suprativ, and Tanmay Bhattacharya. “A new protein sequence classification approach using positional-average values of features.” In *Soft Computing: Theories and Applications: Proceedings of SoCTA 2018*, pp. 703–712. Springer Singapore, 2020
13. Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. “Efficient estimation of word representations in vector space.” *arXiv preprint* arXiv:1301.3781 (2013)
14. Le, Quoc, and Tomas Mikolov. “Distributed representations of sentences and documents.” In *International conference on machine learning*, pp. 1188–1196. PMLR, 2014
15. Ng, Patrick. “dna2vec: Consistent vector representations of variable-length k-mers.” *arXiv preprint* arXiv:1701.06279 (2017)
16. Ren, Ruohan, Changchuan Yin, and Stephen S.-T. Yau. “kmer2vec: A novel method for comparing DNA sequences by word2vec embedding.” *Journal of Computational Biology* 29, no. 9 (2022): 1001–1021

17. Du, J., Jia, P., Dai, Y., Tao, C., Zhao, Z., Zhi, D.: Gene2vec: distributed representation of genes based on co-expression. BMC Genomics **20**, 7–15 (2019)
18. Zhang, Yu-Fang, Xiangeng Wang, Aman Chandra Kaushik, Yanyi Chu, Xiaoqi Shan, Ming-Zhu Zhao, Qin Xu, and Dong-Qing Wei. "SPVec: a Word2vec-inspired feature representation method for drug-target interaction prediction." *Frontiers in chemistry* 7 (2020): 895
19. Shomron, Noam, and Carmit Levy. "MicroRNA-biogenesis and Pre-mRNA splicing crosstalk." *BioMed Research International* 2009 (2009)
20. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**, 273–297 (1995)
21. Suykens, Johan AK, Lukas Lukas, Paul Van Dooren, Bart De Moor, and Joos Vandewalle. "Least squares support vector machine classifiers: a large scale algorithm." In *European Conference on Circuit Theory and Design, ECCTD*, vol. 99, pp. 839–842. Citeseer, 1999
22. Gu, Jiuxiang, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu et al. "Recent advances in convolutional neural networks." *Pattern recognition* 77 (2018): 354–377
23. Harrow, Jennifer, Adam Frankish, Jose M. Gonzalez, Electra Tapanari, Mark Diekhans, Felix Kokocinski, Bronwen L. Aken et al. "GENCODE: the reference human genome annotation for The ENCODE Project." *Genome research* 22, no. 9 (2012): 1760–1774
24. Woolfe, A., Mullikin, J.C., Elnitski, L.: Genomic features defining exonic variants that modulate splicing. Genome Biol. **11**, 1–23 (2010)
25. Chaffey, Nigel. "Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K. and Walter, P. Molecular biology of the cell. 4th edn." (2003): 401–401

RADA: Reconstruction Assisted Domain Adaptation for Nighttime Aerial Tracking

Avinash Chouhan[1,2](✉), Mayank Chandak[2], Arijit Sur[2], Dibyajyoti Chutia[1], and Shiv Prasad Aggarwal[1]

[1] North Eastern Space Applications Centre, Umiam 793103, India
[2] Indian Institute of Technology Guwahati, Guwahati, Assam 781039, India
avinash.chouhan@nesac.gov.in

Abstract. Visual object tracking is a popular research area in computer vision due to its diverse applications. Despite the impressive progress made by numerous state-of-the-art trackers on large-scale datasets, visual object tracking at nighttime remains challenging because of low light (brightness) conditions, lack of contrast, very low variability among feature distributions, etc. In addition, the lack of paired (labeled) data for nighttime tracking makes it infeasible for supervised learning based modeling. Unsupervised domain adaptation based tracking can resolve this issue. In this work, we proposed static image style transfer-based Reconstruction Assisted Domain Adaptation (RADA) with adversarial learning for nighttime object tracking. The main contribution of the work is two-fold. First, a reconstruction-assisted adaptation is proposed for domain invariant feature extraction and to achieve input and feature level adaptation. Secondly, static style transfer is used to generate synthetic paired images (video frames) for supervised nighttime modeling for visual object tracking. Style adversarial alignment at multiple levels helped to adapt between the styled source domain and the target domain, which do not require pseudo labels. RADA attained feature and input level adaptation without external model requirements for low-light image enhancement. Static style transfer avoids negative domain transfer and enables domain transfer learning on true labels. The effectiveness of RADA is validated on six benchmark datasets. RADA achieved state-of-the-art results on two benchmark nighttime adaptation datasets with improvements in the range of 3.7% - 11.4%. RADA also attained state-of-the-art results on three other nighttime datasets without target adaptation. The tracking results and model weights are available at https://github.com/chouhan-avinash/RADA/.

Keywords: Reconstruction based adaptation · Adversarial learning · Input and feature level adaptation

1 Introduction

Object tracking is an important computer vision problem that involves object localization by means of a bounding box and its continuous following for a few frames. There have been several recent visual object tracking approaches that show promising results. These include Correlation Filter based approaches [1,2] and Siamese Network based approaches [3–5], and Transformer [6–9] based approaches. Object tracking in nighttime conditions is important, and several approaches [10–17] used low light enhancement based nighttime detection and tracking. It has been observed in the literature that visual object tracking at night time is a quite challenging task due to the scarcity of paired (labeled) images to train the supervised models. Domain adaptation is a technique that extends transfer learning by minimizing the differences between the domain-specific features across different domains. These approaches have gained popularity in semantic segmentation [18,19], object detection [11,20] and, recently, in object tracking [15,21,22]. Several strategies involving domain adaptation have been developed, including addressing domain shifts at both image and instance levels [20] and merging low-light enhancement and object detection models [11]. Some works [21,22] used adversarial learning for domain adaptation in visual object tracking to minimize the domain discrepancy between day and night features and showed promising results in addressing the domain shift problem in object tracking. In this work, we proposed static style transfer based Reconstruction Assisted Domain Adaptation (RADA) for nighttime object tracking. Static style transfer is used to generate synthetic paired images (video frames) for supervised nighttime modeling for visual tracking. The workflow of RADA is shown in Fig 1. Results show that the proposed approach outperforms the recent SOTA schemes.

The organization of this work is as follows. A review of recent related work is presented in Sect. 2, which is followed by proposed scheme details in Sect. 3. The implementation details are presented in Sect. 4 followed by results evaluation in Sect. 5 and ablation analysis in Sect. 6. The work is concluded in Sect. 7.

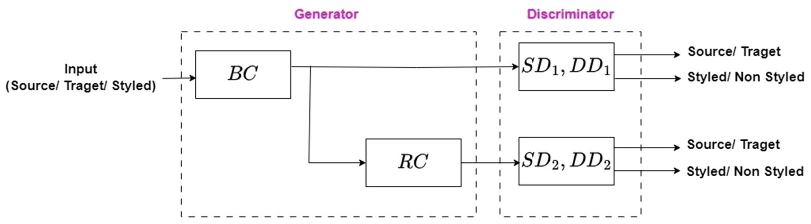


Fig. 1. Workflow of RADA framework. Here, BC is the Backbone Component, RC is the Reconstruction Component, SD is the Style Discriminator, and DD is the Domain Discriminator.

2 Literature Survey

2.1 Visual Object Tracking

The correlation filter based object trackers use a template of the target object to find the most similar regions in the subsequent frames. They often require special handcrafted feature extraction techniques for processing. Galoogahi et al. [1] used background based negative samples extracted through handcrafted features in correlation filters for efficient object tracking. Li et al. [2] proposed varying spatiotemporal regularization, which utilized global and local response variations details in the object tracker. The Siamese network-based approaches focus on finding a generic similarity function. They generally use a pair of identical common weighted neural networks to find the correlation between the target object and the search region and take the benefit of not requiring any unique feature engineering due to the end-to-end learning possible. Chen et al. [3] proposed a siamese network with a unified approach for object classification and bounding box regression. Xu et al. [4] utilized a fully convolutional object tracker with target state estimation and generated classification score for local view instead of predefined anchors. Guo et al. [5] proposed anchor and region proposal free siamese network, which requires fewer hyperparameters. More recently, transformer-based object tracking has been proposed. Lin et al. [6] proposed the use of a full attention-based siamese network with motion tokens to extract motion context. Cui et al. [7,9] proposed tracking framework with repetitive mix attention modules for joint features extraction and relation mapping between target and template frame. They also proposed transformer based hierarchical and non-hierarchical trackers with several pretraining strategies. Ye et al. [8] proposed transformer based a single stream stage tracking framework combining feature extraction and target-template relation modeling and utilizing an early candidate elimination approach for efficient inference.

2.2 Nighttime Object Tracking

Despite the excellent performance achieved in object tracking, these trackers still show a comparatively poor result when adverse conditions (e.g. nighttime) are involved. Li et al. [10] proposed a low light enhancement module with a correlation filter for nighttime tracking, but they are restricted to handcrafted features and could not benefit from end-to-end learning. Sasagawa et al. [11] utilized multiple pretrained models for knowledge distillation from the low light enhancement model to the object detection model. Ye et al. [14] proposed retinex based iterative low light enhancement with joint illuminations and noise estimation for object tracking. They further mitigate the weak collaboration in visual tracking and proposed a transformer [15] with learned curve projection-based image light enhancement for illumination and denoising of low light images. Li et al. [16] used illumination adaptive tracking with low light enhancement and target aware masking. Fu et al. [17] proposed transformer based image enhancer with crafted range and antinoise mask and dynamic parameter adaptation. Zhu et al. [12] proposed night image enhancement followed by a tracking approach that utilized darkness clue to produce a visual prompt. Ma et al. [13] proposed a bilevel adaptation for low-light image enhancement, adaptable to unknown scenes.

Chen et al. [20] proposed image and instance level alignment for domain adaptive object detection. Wu et al. [18] used an adversarial learning-based nighttime image relighting module and segmentation module. Ye et al. [21] explored using adversarial learning for domain adaptation in visual object tracking to minimize the domain discrepancy between day and night features. Their approach has shown promising results in addressing the domain shift problem in object tracking. Zhang [22] proposed progressive style translation using domain invariant content details and separate domain style details. Yao et al. [23] used a segment-anything model-assisted approach with zero-shot learning-based training sample generation for the target night domain. Wu et al. [19] used an adversarial learning-based single-stage domain adaptation approach for semantic segmentation of nighttime images. Lu [24] used a multi-level denoising transformer, which used a self-attentive encoder and decoder with a cross-attention module. Fu et al. [25] proposed a contrastive learning-based domain adaptive network for object tracking with scale information. Lv et al. [26] used Gabor filter-based preprocessing steps before utilizing the adversarial learning-based domain adaptive training network. Sun et al. [27] utilized filtering of high-frequency noise for nighttime image enhancement with a dynamic template-based object-tracking network. Kennerley et al. [28] proposed a student-teacher-based network that combined high and low-confidence pseudo labels as two-phase consistency with nighttime augmentations. Chen et al. [29] proposed a mean teacher-based network, which utilized assignment-based object identification and low light enhancement for unlabelled target domain inputs. Zheng et al. [30] proposed multi-source domain adaptation for satellite videos, which utilized student-based learning with weak re-identification.

Despite the advancements in object tracking using deep learning approaches, there are a few areas where improvements may be possible. Some of such observations are as follows:

1. Image level adaptation is difficult but important for day-to-night domain adaptation.
2. Learned style translation requires additional computational overhead and may deteriorate results in some unpaired cases.
3. Existing trackers often perform poorly in adverse conditions, such as low-light environments that can affect the visual aspect of the tracked object.

3 Proposed Scheme

In this work, we proposed Reconstruction Assisted Domain Adaption (RADA) for nighttime object tracking. Our main contributions are

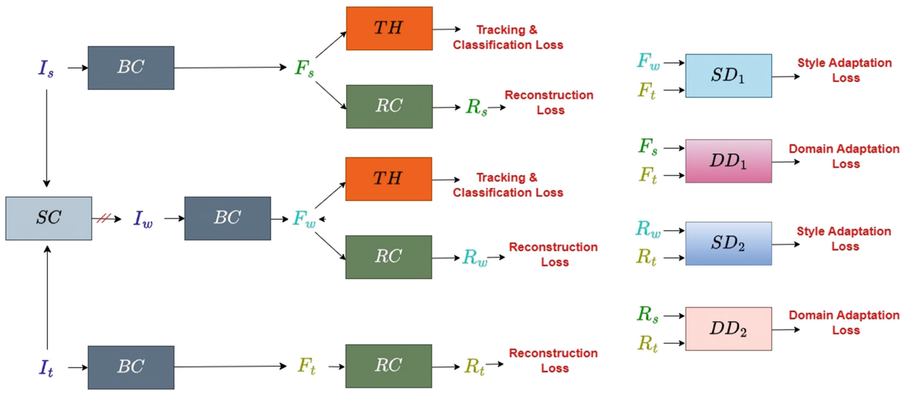


Fig. 2. Illustration of the proposed RADA framework. Here, I_s, I_t, and I_w are search and template image pairs of source, target, and styled source domains. F_s, F_t, and F_w are extracted source, target, and styled source features using the Backbone Component (BC).

1. Reconstruction assisted adaptation is proposed for domain invariant feature extraction and to attain input and feature level adaptation.
2. Style adversarial alignment at multiple level is proposed to adapt between the styled source domain and the target domain, which do not require pseudo labels.

RADA achieved state-of-the-art results on two benchmark dataset for domain adaptive nighttime tracking. The various components of RADA are presented in Fig. 2. It consists of a Backbone Component (BC), a Tracking Head (TH), a Reconstruction Component (RC), and Discriminators (DD_1, DD_2, SD_1, SD_2). Adding these modules aims to improve the robustness of the tracker in adverse conditions, enhance domain adaptation, and utilize the benefits of style transfer for object tracking.

Template and search images are obtained from the input video sequences. BC utilized these inputs and extracts features that TH used for regression and classification when labeled inputs (source image and styled source image) are passed. The feature maps are also passed (in all cases) into the proposed RC, which learns to reconstruct the input images. Discriminators used extracted features and reconstructed outputs for alignment.

3.1 Backbone Component

The backbone component (BC) is used to extract features from the input. We used mixed attentive transformer [7] based common BC for source, target, and styled source inputs. Inputs having search image of size $3 \times 384 \times 384$ template image of size $3 \times 192 \times 192$ are passed to BC for feature extraction. The extracted features have sizes of $1024 \times 24 \times 24$ and $1024 \times 12 \times 12$ for search and template inputs.

3.2 Style Transfer Component

We used Wallis Style Transfer [31,32] for static style transfer of the target domain night image to the source domain day image. It works by blending the style image over the content image as shown in 3. For source image I_s, target image I_t, target styled source image is produced using following equation :

$$I_w = w[\sigma(I_t)(\frac{I_s - \mu(I_s)}{\sigma(I_s)}) + \mu(I_t)] + (1 - w)I_s \quad (1)$$

Here, w is the weight parameter, μ is the mean, and σ is the variance. The style source image I_w with the same annotation is used for model training and extracting domain invariant features. We used $w = 1$ in experimentation for a complete nighttime style transfer.

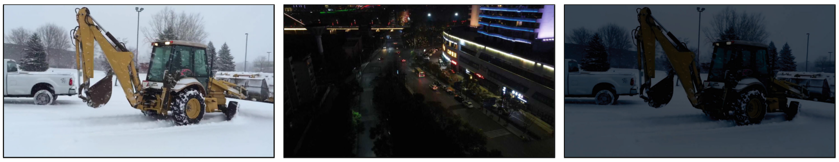

(a) GOT image (b) NAT2021 image (c) Output image

Fig. 3. Visualization of style transfer. Here daytime image is transformed into night-styled image with day content.

3.3 Reconstruction Component

Reconstruction Component (RC) uses multitask learning which helps to learn a common representation of the domains and to attain input-level domain adaptation through reconstruction. Reconstruction Component consists of transposed convolution-based upsampling with a resolution module. It takes features extracted by BC and reconstructs the original passed inputs. RC takes input shapes of $1024 \times 32 \times 32$ and $1024 \times 16 \times 16$ for search and template input and produces a reconstructed output of shape $3 \times 384 \times 384$ and $3 \times 192 \times 192$.

3.4 Tracker Head

Tracker Head (TH) consists of classification and regression branches for object presence identification and bounding box detection. TH takes feature extracted by BC with the shape of $1024 \times 32 \times 32$ and $1024 \times 16 \times 16$ for search and template inputs. The outputs of TH are used for the calculation of tracking and classification losses.

3.5 Domain Discriminators

Domain Discriminators (DD) distinguished source and target at the feature and reconstructed output levels. DD enables feature and input level adaptation between the source and target domain and helps in domain invariant feature representation. DD consists of two discriminators ($DD1$, $DD2$) for feature and reconstructed outputs. $DD1$ takes domain features extracted by BC of shape $1024 \times 32 \times 32$ and $1024 \times 16 \times 16$ for search and template image and produce domain classification output. DD2 used output generated by RC of shape $3 \times 192 \times 192$ and $3 \times 384 \times 384$ for template and search inputs to generate classification output.

3.6 Style Discriminators

Style Discriminators (SD) differentiate between styled source and target at the feature and reconstructed input levels. SD attains feature and input level adaptation and minimizes the difference between target-styled source domain inputs with target domain inputs. SD contains two style discriminators ($SD1$, $SD2$). $SD1$ took BC extracted features for template and search inputs having the shape of $1024 \times 32 \times 32$ and $1024 \times 16 \times 16$. $SD2$ utilized RC outputs of template and search inputs with shapes of $3 \times 192 \times 192$ and $3 \times 384 \times 384$ for style nonstyle classification.

3.7 Loss Function

The total loss (L) for the network in one iteration can be calculated as follows:

$$L = Ls_{tra} + \alpha Ls_{rec} + \beta(Lw_{tra} + \alpha Lw_{rec}) + \alpha Lt_{rec} + \lambda L_{adv} \tag{2}$$

Here Ls_{tra} and Lw_{tra} are tracking loss for source and styled source inputs, which includes classification and regression loss. Ls_{rec}, Lw_{rec}, Lt_{rec} are reconstruction losses on the source, styled source and target inputs. L_{adv} is adverserial loss and loss weight parameters are represented by α, β, and λ.

4 Implementation Details

4.1 Dataset

This work uses the GOT [33] dataset as a source domain dataset. For the target domain, NAT2021 benchmark [21] train set (NAT2021-train) is used without the labels. Performance evaluation is done on NAT2021-test and NAT2021-L-test datasets. NAT2021-train dataset contains 1400 videos with 276081 frames without annotations. NAT2021-test and NAT2021-L-test datasets have 180, 23 videos and 140815, 53564 frames, respectively. NAT2021-L-test involves much longer videos and is used for the long-term tracking evaluation of the trackers. Long-term tracking is a typical scene in visual object tracking and involves many more challenging attributes. RADA performance comparisons are also made on three other nighttime datasets(UAVDark70 [10], UAVDark135 [16], DarkTrack2021 [15]) without target adaptation to validate its efficacy.

4.2 Hyperparameters and Evaluation Metrics

The model is implemented in the PyTorch framework and experiments are run on the system with a V100 (32 GB) graphics card. Models are trained using the SGD with momentum optimizer, with the initial learning rate set to $\gamma = 0.0015$ and the momentum set to $\mu = 0.9$. The learning rate is scheduled logarithmically over the number of trained epochs, starting from 0.0015 to 0.00015. The discriminator is trained using the Adam optimizer with the initial decay rates set to $\beta_1 = 0.9$ and $\beta_2 = 0.99$. The base learning rate is set at 0.005. The entire training is done for a total of 30 epochs. For evaluation, we use the One Pass Evaluation style. We use the same metrics as the baseline tracker for comparison: success rate, precision, and normalized precision. RADA has a computation cost similar to the base backbone model [7] (MACS: 183.89M, FLOPS: 127.81 G, FPS: 27 on RTX 8000). During test time, RADA does not utilize additional modules (reconstruction component, style transfer, and style and domain discriminators).

Table 1. Numerical comparative results for the test set of NAT2021 dataset. The best results are shown in red color, and the second-best results are marked in blue color.

	Source Dataset	Success Rate	N. Precision	Precision
RADA (ours)	GOT [33]	0.567	0.691	0.737
PDST [22]	GOT [33], LaSOT [34]	0.547	0.665	0.76
MT-CAR [29]	GOT [33], VID [35]	0.507	0.592	0.72
MT-BAN [29]	GOT [33], VID [35]	0.494	0.562	0.699
UDAT-CAR [21]	GOT [33], VID [35]	0.483	0.564	0.687
UDAT-BAN [21]	GOT [33], VID [35]	0.469	0.546	0.694
SiamCAR [5]	GOT [33], VID [35]	0.453	0.542	0.663
SiamBAN [3]	GOT [33], VID [35]	0.437	0.509	0.647

Table 2. Numerical comparative results for the test set of NAT2021-L dataset. The best results are shown in red color, and the second-best results are marked in blue color.

	Source Dataset	Success Rate	N. Precision	Precision
RADA (ours)	GOT [33]	0.565	0.667	0.722
PDST [22]	GOT [33], LaSOT [34]	0.507	0.599	0.649
MT-BAN [29]	GOT [33], VID [35]	0.399	0.455	0.556
MT-CAR [29]	GOT [33], VID [35]	0.39	0.442	0.543
SFDT [26]	GOT [33], VID [35]	0.401		0.524
UDAT-CAR [21]	GOT [33], VID [35]	0.376	0.413	0.506
UDAT-BAN [21]	GOT [33], VID [35]	0.352	0.406	0.496
SiamCAR [5]	GOT [33], VID [35]	0.330	0.375	0.477
SiamBAN [3]	GOT [33], VID [35]	0.316	0.366	0.464

Table 3. Numerical comparative results for the other datasets. The best results are shown in red color, and the second-best results are marked in blue color. Here, SR is success rate, NP is normalized precision, P is precision and AO is average of overlap rates. * represents online server evaluation.

Methods	UAVDark135			UAVDark70			DarkTrack2021			Source*		
	SR	NP	P	SR	NP	P	SR	NP	P	AO	$SR_{.5}$	$SR_{.75}$
RADA (ours)	0.615	0.744	0.756	0.615	0.756	0.765	0.567	0.681	0.695	0.762	0.851	0.762
PDST [22]	0.607	0.729	0.737	0.597	0.738	0.761	-	-	-	-	-	-
MT-CAR [29]	-	-	-	0.501	0.595	0.682	-	-	-	-	-	-
DCPT [12]	0.577	0.710	0.717	-	-	-	0.540	0.667	0.646	-	-	-
Dimp50 SCT [15]	0.562	0.710	0.717	-	-	-	0.521	0.677	0.633	-	-	-
UDAT-CAR [21]	0.490	0.617	0.612	0.512	0.592	0.695	0.470	0.570	0.600	0.503	0.576	0.338
HighlightNet [17]	0.424	-	0.539	0.434	0.526	0.616	-	-	-	-	-	-
SAM-DA [23]	-	-	-	-	-	-	0.454	0.529	0.592	-	-	-
MixViT-L [9]	-	-	-	-	-	-	-	-	-	0.757	0.853	0.751

Table 4. Ablation study for network components on NAT2021 test set.

	Success Rate	N. Precision	Precision
BC+SC+RC1+RC2+DD+SD	0.561	0.682	0.721
BC+SC+RC1+RC2+DD	0.552	0.669	0.715
BC+SC+RC1+RC2	0.548	0.667	0.710
BC+SC+RC1	0.546	0.661	0.708
BC+SC	0.545	0.658	0.706
BC	0.532	0.625	0.681

5 Performance Comparison

5.1 Numerical Comparison

NAT2021-Test. A performance comparison of our model with other recent work is presented in Table 1. RADA got the highest success rate and normalized precision values of 0.567 and 0.691. For precision, RADA achieved the second-best result of 0.737, preceded by PDST [22] results of 0.76. The success, normalized precision, and precision plot comparisons of the different trackers on the NAT2021 test dataset are shown in Fig. 6, Fig. 7, and Fig. 8. RADA raises the previous results of success rate and normalized precision by **3.7%** and **3.9%**.

NAT2021-L-Test. Table 2 depicted the comparative results on a large sequence video dataset.RADA significantly outperforms the baseline. In success rate, RADA (0.565) raises the previous result [22] (0.507) by **11.4%**. In precision, RADA (0.722) raises the previous result [22] (0.649) by **11.2%**. In normalized

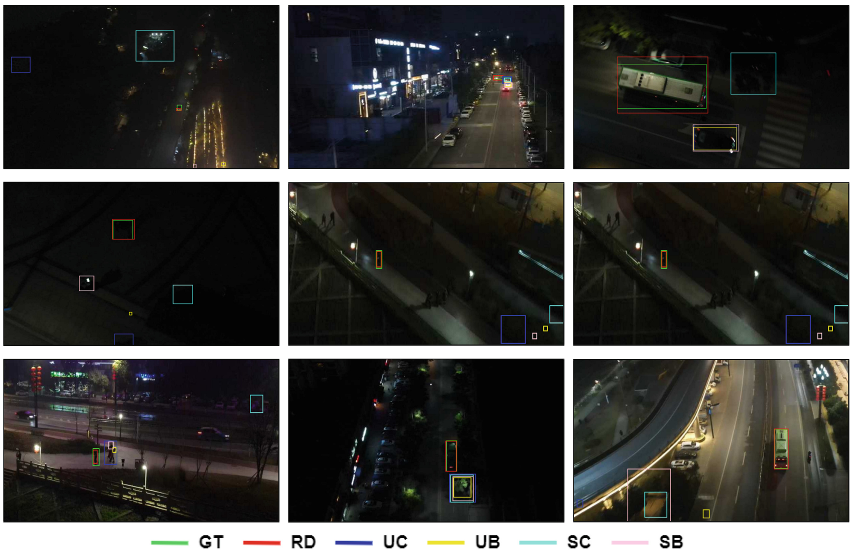


Fig. 4. Visual comparison of results of RADA and baselines on the test set of NAT2021 dataset. Here, GT is ground truth, RD is RADA, UC is UDAT-CAR, UB is UDAT-BAN, SC is SiamCAR, and SB is SiamBAN.

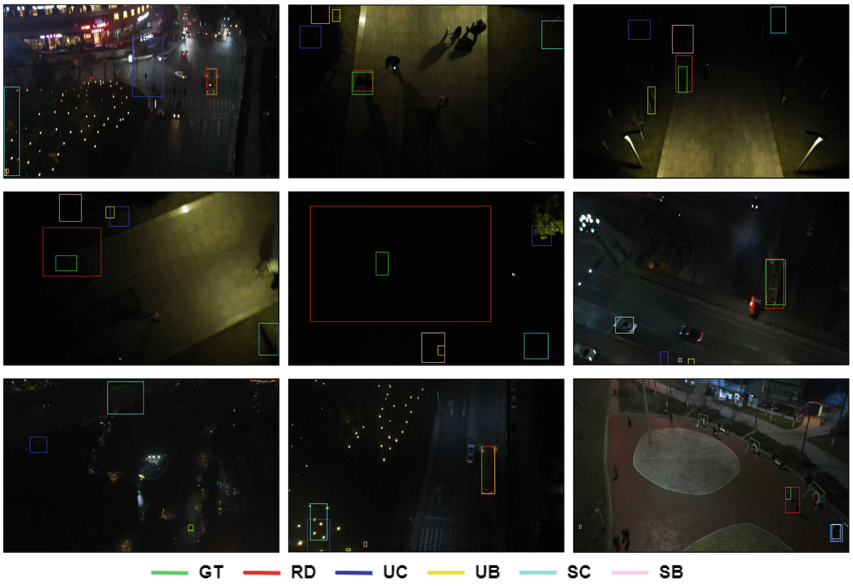


Fig. 5. Visual comparison of results of RADA and baselines on the test set of NAT2021-L dataset. Here, GT is ground truth, RD is RADA, UC is UDAT-CAR, UB is UDAT-BAN, SC is SiamCAR, and SB is SiamBAN.

Table 5. Ablation study for network components on NAT2021-L test set.

	Success Rate	N. Precision	Precision
BC+SC+RC1+RC2+DD+SD	0.532	0.625	0.681
BC+SC+RC1+RC2+DD	0.524	0.613	0.681
BC+SC+RC1+RC2	0.508	0.599	0.653
BC+SC+RC1	0.502	0.587	0.642
BC+SC	0.505	0.599	0.633
BC	0.491	0.564	0.621

Table 6. Ablation study for loss hyperparameter on NAT2021 test set.

α	β	Success Rate	N. Precision	Precision
0	0	0.532	0.625	0.681
0	0.25	0.545	0.658	0.706
0	0.5	0.541	0.652	0.691
0	0.75	0.539	0.649	0.689
0	1	0.537	0.641	0.681
0.25	0.25	0.548	0.667	0.710
0.25	0.5	0.546	0.661	0.706

Table 7. Ablation study for loss hyperparameter on NAT2021-L test set.

α	β	Success Rate	N. Precision	Precision
0	0	0.491	0.564	0.621
0	0.25	0.505	0.599	0.633
0	0.5	0.501	0.587	0.638
0	0.75	0.5	0.579	0.634
0	1	0.49	0.569	0.627
0.25	0.25	0.508	0.599	0.653
0.25	0.5	0.506	0.587	0.651

precision, RADA (0.667) improves the previous result [22] (0.599) by **11.4%**. The success, normalized precision, and precision plot comparisons of the different trackers on the NAT2021L test dataset are shown in Fig. 6, Fig. 7, and Fig. 8.

Other Datasets. RADA performance is also evaluated on three nighttime (UAVDark70 [10], UAVDark135 [16], DarkTrack2021 [15]) and one source (GOT [33]) datasets and qualitative results are presented in Table 3. For other nighttime datasets, RADA attained better results compared to other recent methods

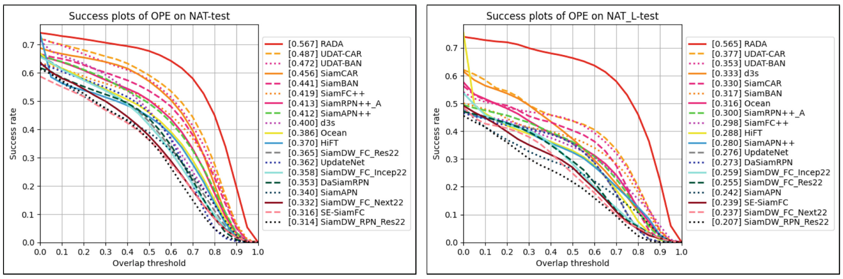


Fig. 6. Success plots of several trackers on the test sets of NAT2021 and NAT2021-L datasets.

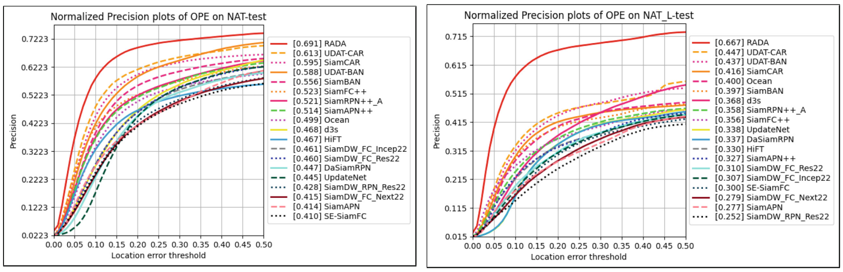


Fig. 7. Normalized precision plots of several trackers on the test sets of NAT2021 and NAT2021-L datasets.

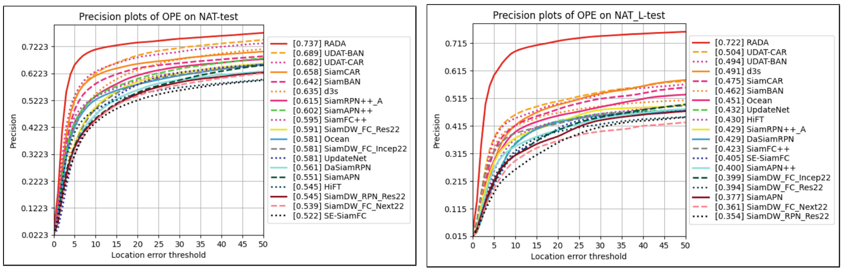
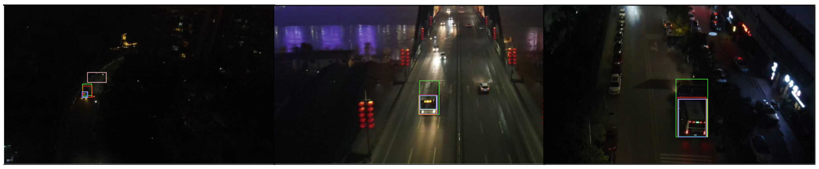


Fig. 8. Precision plots of several trackers on the test sets of NAT2021 and NAT2021L datasets.

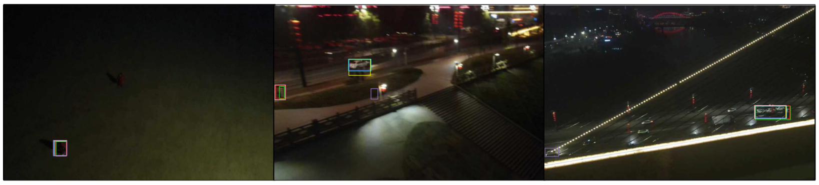
[12,15,17,22,23,29]. RADA also achieved better results on the source dataset compared to baseline methods [9,21].

5.2 Qualitative Comparison

We illustrate some instances from both datasets where our tracker is able to correctly identify the target object and locate it while the other tracker starts making arbitrary bounding boxes or tracking a different object altogether. For



(a) NAT2021-test



(b) NAT2021-L-test

Fig. 9. Visual comparison of ablation study on test sets of NAT2021 and NAT2021-L datasets.

visual comparisons, baseline results of UDAT-CAR [21], UDAT-BAN [21], SiamCAR [5], and SiamBAN [3] are used. Figure 4 visualized results for NAT2021-test datasets. Visual comparison for NAT2021-L-test dataset is shown in Fig. 5.

6 Ablation Study

To demonstrate the effectiveness of the various components of RADA and to find optimal parameter values, extensive ablation comparisons are made on the NAT2021-test and NAT2021L-test datasets. All ablation study experiments are run for 10 epochs. The best result configuration is retrained for 30 epochs to produce the final results presented in Table 1 and Table 2.

6.1 Ablation for Network Components

Table 4 and Table 5 presented network ablation studies for RADA. The initial results of BC are taken as a baseline for comparison. The addition of the Style transfer Component (SC) improved the qualitative results for both datasets. Including a reconstruction component for the source domain and styled source inputs (RC1) further improved the results. We extended the reconstruction component for (RC2) for target input, which further enhanced the results. The addition of adversarial components (style and domain discriminators) significantly improved the previous results produced by BC+SC+RC1+RC2. The visual results of this ablation study are shown in Fig. 9.

6.2 Ablation for Loss Parameters

Table 6 and Table 7 depicted the ablation results for loss weight parameters of α and β. Initially, $\beta = 0$ results are produced with $\alpha = 0$. Experiments are done with increments of 0.25 in both values. From these experiments, we concluded that the best results are achieved using values of 0.25 and 0.25 for α and β.

7 Conclusion

This work proposes reconstruction component-based domain adaptation, which results in input-level and feature-level adaptation. The style transfer component helped in learning target domain representation without generating noisy pseudo levels. Our proposed framework does not require an external nighttime image enhancement model and can produce good results without this step. Detailed qualitative and quantitative evaluations are presented for the proposed model. The results are validated on six benchmark datasets and compared with other recent works.

References

1. Kiani Galoogahi, H., Fagg, A., Lucey, S.: Learning background-aware correlation filters for visual tracking. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 1144–1152 (2017)
2. Li, Y., Fu, C., Ding, F., Huang, Z., Lu, G.: AutoTrack: towards high-performance visual tracking for UAV with automatic spatio-temporal regularization. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11920–11929 (2020)
3. Chen, Z., Zhong, B., Li, G., Zhang, S., Ji, R.: Siamese box adaptive network for visual tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6668–6677 (2020)
4. Yinda, X., Wang, Z., Li, Z., Yuan, Y., Gang, Yu.: SiamFC++: towards robust and accurate visual tracking with target estimation guidelines. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 12549–12556 (2020)
5. Guo, D., Wang, J., Cui, Y., Wang, Z., Chen, S.: SiamCAR: Siamese fully convolutional classification and regression for visual tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6269–6277 (2020)
6. Lin, L., Fan, H., Zhang, Z., Xu, Y., Ling, H.: Swintrack: a simple and strong baseline for transformer tracking. In: Conference on Neural Information Processing Systems (NeurIPS) (2022)
7. Cui, Y., Jiang, C., Wang, L., Wu, G.: MixFormer: end-to-end tracking with iterative mixed attention. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 13598–13608, Los Alamitos, CA, USA. IEEE Computer Society (2022)
8. Ye, B., Chang, H., Ma, B., Shan, S., Chen, X.: Joint feature learning and relation modeling for tracking: a one-stream framework. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T., (eds.) Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXII, vol. 13682. LNCS, pp. 341–357. Springer (2022)

9. Cui, Y., Jiang, C., Gangshan, W., Wang, L.: MixFormer: end-to-end tracking with iterative mixed attention. IEEE Trans. Pattern Anal. Mach. Intell. **46**(6), 4129–4146 (2024)
10. Li, B., Fu, C., Ding, F., Ye, J., Lin, F.: ADTrack: target-aware dual filter learning for real-time anti-dark UAV tracking. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 1–8 (2021)
11. Sasagawa, Y., Nagahara, H.: Yolo in the dark-domain adaptation method for merging multiple models. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16, pp. 345–359. Springer (2020)
12. Zhu, J., et al.: DCPT: darkness clue-prompted tracking in nighttime UAVs. arXiv preprint arXiv:2309.10491 (2023)
13. Ma, L., et al.: Bilevel fast scene adaptation for low-light image enhancement. Int. J. Comput. Vis., 1–19 (2023)
14. Ye, J., Fu, C., Zheng, G., Cao, Z., Li, B.: Darklighter: light up the darkness for UAV tracking. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3079–3085 (2021)
15. Ye, J., Changhong, F., Cao, Z., An, S., Zheng, G., Li, B.: Tracker meets night: a transformer enhancer for UAV tracking. IEEE Rob. Autom. Lett. **7**(2), 3866–3873 (2022)
16. Li, B., Changhong, F., Ding, F., Ye, J., Lin, F.: All-day object tracking for unmanned aerial vehicle. IEEE Trans. Mob. Comput. **22**(8), 4515–4529 (2023)
17. Fu, C., Dong, H., Ye, J., Zheng, G., Li, S., Zhao, J.: HighlightNet: highlighting low-light potential features for real-time UAV tracking. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 12146–12153 (2022)
18. Wu, X., Wu, Z., Guo, H., Ju, L., Wang, S.: DANNet: a one-stage domain adaptation network for unsupervised nighttime semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15769–15778 (2021)
19. Wu, X., Wu, Z., Guo, H., Ju, L., Wang, S.: DANNet: a one-stage domain adaptation network for unsupervised nighttime semantic segmentation. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 15764–15773, Los Alamitos, CA, USA. IEEE Computer Society (2021)
20. Chen, Y., Li, W., Sakaridis, C., Dai, D., Van Gool, L.: Domain adaptive faster R-CNN for object detection in the wild. In: Computer Vision and Pattern Recognition (CVPR) (2018)
21. Ye, J., Fu, C., Zheng, G., Paudel, D.P., Chen, G.: Unsupervised domain adaptation for nighttime aerial tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–10 (2022)
22. Zhang, J., Li, Z., Wei, R., Wang, Y.: Progressive domain-style translation for nighttime tracking. In: Proceedings of the 31st ACM International Conference on Multimedia, MM '23, pp. 7324–7334, New York, NY, USA. Association for Computing Machinery (2023)
23. Yao, L., Zuo, H., Zheng, G., Changhong, F., Pan, J.: SAM-DA: UAV tracks anything at night with SAM-powered domain adaptation (2023)
24. Kunhan, L., Changhong, F., Wang, Y., Zuo, H., Zheng, G., Pan, J.: Cascaded denoising transformer for UAV nighttime tracking. IEEE Robot. Autom. Lett. **8**(6), 3142–3149 (2023)

25. Changhong, F., Li, T., Ye, J., Zheng, G., Li, S., Peng, L.: Scale-aware domain adaptation for robust UAV tracking. IEEE Robot. Autom. Lett. **8**(6), 3764–3771 (2023)
26. Lv, Y., Feng, W., Wang, S., Dauphin, G., Zhang, Y., Xing, M.: Spectral-spatial feature enhancement algorithm for nighttime object detection and tracking. Symmetry **15**(2) (2023)
27. Sun, L., Kong, S., Yang, Z., Gao, D., Fan, B.: Modified Siamese network based on feature enhancement and dynamic template for low-light object tracking in UAV videos. Drones **7**(7) (2023)
28. Kennerley, M., Wang, J.G., Veeravalli, B., Tan, R.T.: 2PCNET: two-phase consistency training for day-to-night unsupervised domain adaptive object detection. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11484–11493, Los Alamitos, CA, USA. IEEE Computer Society (2023)
29. Chen, J., Sun, Q., Zhao, C., Ren, W., Tang, Y.: Rethinking unsupervised domain adaptation for nighttime tracking. In: Luo, B., Cheng, L., Wu, ZG., Li, H., Li, C. (eds.) Neural Information Processing, pp. 391–404. Springer, Singapore (2024)
30. Zheng, X., Cui, H., Xiaoqiang, L.: Multiple source domain adaptation for multiple object tracking in satellite video. IEEE Trans. Geosci. Remote Sens. **61**, 1–11 (2023)
31. Li, X., Luo, M., Ji, S., Zhang, L., Meng, L.: Evaluating generative adversarial networks based image-level domain transfer for multi-source remote sensing image segmentation and object detection. Int. J. Remote Sens. **41**(19), 7343–7367 (2020)
32. Peng, D., Guan, H., Zang, Y., Bruzzone, L.: Full-level domain adaptation for building extraction in very-high-resolution optical remote-sensing images. IEEE Trans. Geosci. Remote Sens. **60**, 1–17 (2021)
33. Huang, L., Zhao, X., Huang, K.: GOT-10k: a large high-diversity benchmark for generic object tracking in the wild. IEEE Trans. Pattern Anal. Mach. Intell. **43**(5), 1562–1577 (2019)
34. Fan, H., et al.: LaSOT: a high-quality benchmark for large-scale single object tracking. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5369–5378 (2018)
35. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (2015)

A Novel Outlier Detection Algorithm Based on Symmetry and Distance Ratio

Haoyu Zhai, Zexuan Fei, and Yan Ma(✉)

College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai 200234, China
ma-yan@shnu.edu.cn

Abstract. Outlier detection has become essential in various fields such as defense monitoring, fiscal anomaly identification, and business industries. Nevertheless, outlier detection methods based on distance or density suffer from certain limitations. The performance of traditional outlier detection algorithms is susceptible to various factors such as data point density, shape, and other factors. To address these challenges, we propose an outlier detection algorithm, which utilizes symmetry ratio and distance ratio to determine the outlier degree of the data point. Moreover, it automatically finds the threshold of outlier degree using the interquartile range method. Experimental results on synthetic and UCI real datasets demonstrate the excellent performance of our algorithm in detecting outliers.

Keywords: Outlier Detection · Outlier Degree · K-Nearest Neighbors · Interquartile Range

1 Introduction

An outlier refers to data points that exhibit significant deviations from other observations or are generated through different mechanisms [1, 2]. Outliers can arise from factors such as measurement errors, data transmission errors, and other sources of variability. Outliers can have an impact on the results of data analysis, potentially leading to erroneous conclusions. Therefore, accurately identifying outliers during the data preprocessing stage is crucial for improving the accuracy and reliability of data analysis.

Outlier detection is an important topic in the field of data mining research, which aims to find data objects that are different from ordinary data points [3, 4]. Outlier detection can be classified into supervised and unsupervised learning approaches based on whether it requires supervision or not [5]. In the supervised scenario, the model predicts the labels of unseen data by learning the difference between normal and abnormal data points, which requires a large amount of accurately labeled data (both normal and abnormal instances). In contrast, unsupervised learning does not depend on labeled information, and numerous recent outlier detection algorithms have been specifically designed for unsupervised learning scenarios [6]. In this paper, we present a novel outlier detection algorithm that leverages the symmetry ratio and distance ratio of data points to quantify their outlier degree. The interquartile range method (IQR) is employed to autonomously determine the threshold for the outlier degree, enabling the identification of data points exceeding this threshold as outliers. In summary, the contributions of this paper are as follows:

- We propose an unsupervised outlier detection algorithm.
- We introduce the concepts of symmetry ratio and distance ratio to represent the outlier degree of data points.
- The IQR method is used to automatically determine the threshold for the outlier degree.
- Our algorithm is not affected by the density and shape of the dataset.

The remainder of this paper is organized as follows. Section 2 introduces the proposed outlier detection algorithm in detail. Section 3 provides the time complexity analysis. Section 4 provides experimental results and analysis on various synthetic and real datasets. Finally, we conclude the paper in Sect. 5.

2 Method

This section provides a detailed description of outlier detection algorithm. We propose the outlier detection algorithm, which consists of three main phases: calculate symmetry ratio, calculate distance ratio, and automatic threshold acquisition.

2.1 Symmetry Ratio

Definition 1 (*k*NN). For a dataset $X_{n\times d} = \{x_1, x_2, x_3, \ldots, x_N\}$ consisting of N data points with d dimensions, the k-nearest neighbors (kNN) of x_i, denoted as $kNN(x_i)$, are defined as follows:

$$kNN(x_i) = \{x_j | d(x_i, x_j) \leq d(x_i, NN_k(x_i))\} \tag{1}$$

where $NN_k(x_i)$ is the kth nearest neighbor of x_i.

Definition 2 (Symmetric point). Assuming $x_j \in kNN(x_i)$, the symmetric point of x_j relative to x_i, denoted as $SYM(x_j|x_i)$, is defined as:

$$SYM(x_j|x_i) = 2 \cdot x_i - x_j \tag{2}$$

Definition 3 (Symmetry Ratio). The symmetry ratio of a data point x_i is defined as:

$$\mathcal{R}_s(x_i) = \frac{\sum_{j=1}^{k} d(\overline{x}_j, NN_1(\overline{x}_j))}{\sum_{j=1}^{k} d(x_i, x_j)} \tag{3}$$

where $x_j \in kNN(x_i)$, $\overline{x}_j = SYM(x_j|x_i)$.

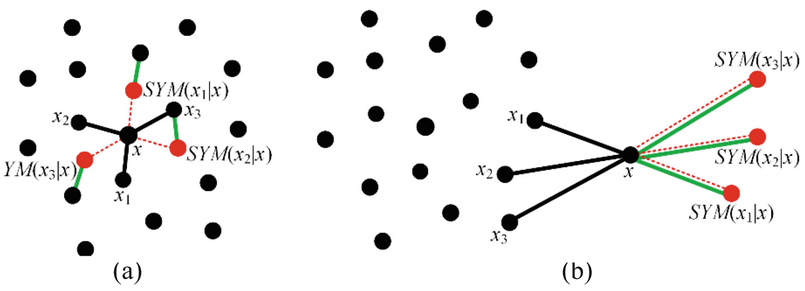


Fig. 1. Two examples of symmetry ratio, where $k = 3$: (a) The data point x is located in the interior, (b) The data point x is an outlier. (Color figure online)

As shown in Fig. 1(a), the data points x_1,x_2, and x_3 are k-nearest neighbors of x. We calculate the symmetric points for x_1, x_2, and x_3 based on Eq. (2), represented by $SYM(x_1|x)$, $SYM(x_2|x)$, and $SYM(x_3|x)$ respectively. These symmetric points are indicated by red points. The points connected to the symmetric points by green lines are the nearest neighbors of the symmetric points. Therefore, The symmetry ratio of data point x is the ratio of the total length of the three green lines to the total length of the three black lines.

Next, we discuss the data point x located on the edge in Fig. 1(b). As x is an outlier, it results in x being the nearest neighbor for the three symmetric points. Consequently, the total length of the three green lines is equal to the total length of the three black lines. The symmetry ratio of x reaches its maximum value of 1. The analysis indicates that the closer the symmetry ratio of a data point is to 1, the more likely it is to be an outlier.

2.2 Distance Ratio

Definition 4 (Distance Ratio). The distance ratio of a data point x_i is defined as:

$$\mathcal{R}_d(x_i) = \frac{\sum_{j=1}^{k} \sum_{m=1}^{k} d(x_j, x_m^j)}{k \cdot \sum_{j=1}^{k} d(x_i, x_j)} \tag{4}$$

where $x_j \in kNN(x_i)$, $x_m^j \in kNN(x_j)$.

Next, we further illustrate the distance ratio. In Fig. 2, the data point x_1 is connected to its three nearest neighbors using blue lines, while the neighbors of x_1 are connected to their three nearest neighbors using red dashed lines. According to Eq. (4), the distance ratio is defined as the ratio between the total length of the red dashed lines and the total length of the blue lines multiplied by the factor k. Since x_1 is not an outlier, there is a relatively small difference in lengths between the blue and red lines. Therefore, $\mathcal{R}_d(x_1)$ approaches 1. Similarly, for the data point x_2 and its nearest neighbors, colored lines are used for connection. As x_2 is an outlier, the length of the blue line is significantly greater than that of the red lines. Thus, we obtain $\mathcal{R}_d(x_2)$ less than 1. Based on the aforementioned analysis, we can draw the following conclusion: when the distance ratio of a data point is less than 1, it is more likely to be considered as an outlier.

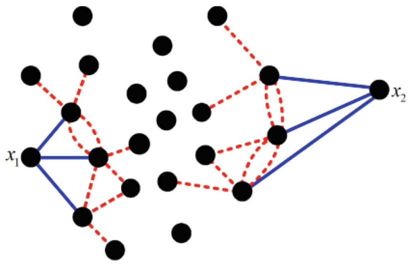


Fig. 2. An example of calculating the distance ratio with $k = 3$. (Color figure online)

Definition 5 (Outlier Degree). The outlier degree of the data point x_i is defined as:

$$\mathcal{O}(x_i) = (\mathcal{R}_s(x_i) - \mathcal{R}_d(x_i))_{\text{NORM}} \tag{5}$$

where NORM represents the normalization process. The greater the value of $\mathcal{O}(x_i)$, the higher the likelihood of x_i being an outlier.

The outlier degree is determined by the symmetry ratio and distance ratio of data points, which reflect the relative values of the neighboring relationships among the data points. These ratios remain unaffected by the density and shape of the data point distribution. Consequently, the outlier degree remains independent of the density and shape of the data point distribution.

2.3 Outlier Degree Threshold

In this section, we utilize the interquartile range (IQR) method to determine the threshold for filtering outliers. The IQR is calculated as the difference between the 75th percentile (Q_3) and the 25th percentile (Q_1) of the data. Assuming the outlier degrees of the data points in the dataset are sorted in ascending order, we can obtain an ordered sequence. Initially, Q_1 and Q_3 are computed using Eqs. (6) and (7). Then, the interquartile range is calculated using Eq. (8). Lastly, the low bound *lb* and up bound *ub* are determined by Eqs. (10) and (11). *ub* can be regarded as the outlier degree threshold. When the outlier degree value of a data point is greater than the upper bound, the data point is considered an outlier.

$$Q_1 = \mathcal{O}_i | i = round(N \times 0.25) \tag{6}$$

$$Q_3 = \mathcal{O}_i | i = round(N \times 0.75) \tag{7}$$

$$IQR = Q_3 - Q_1 \tag{8}$$

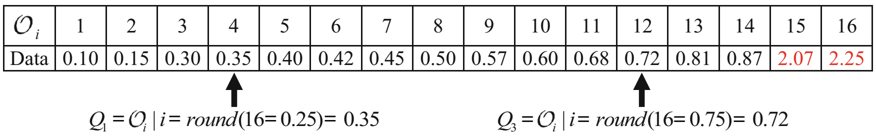
$$lb = Q_1 - r \times IQR \tag{9}$$

$$ub = Q_3 + r \times IQR \tag{10}$$

where r is a parameter used to determine the range of outliers.

Tukey's method for outlier detection sets r to 1.5 [7]. However, there is no statistical basis for this specific value, and it can be adjusted based on the specific application [8]. We will discuss in detail the selection of the parameter r in Sect. 3.

Figure 3 provides an example of calculating the IQR. In this example, we used 16 samples, which are sorted in ascending order. Q_1 and Q_3 are calculated as 0.35 and 0.72 respectively, resulting in an IQR of 0.37. By substituting $r = 1.5$ into Eq. (10), ub is determined as 1.275. Since the values 2.07 and 2.25 are greater than ub, they are considered as outliers and are highlighted as red numbers in Fig. 3.



$\mathcal{O}_i$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Data	0.10	0.15	0.30	0.35	0.40	0.42	0.45	0.50	0.57	0.60	0.68	0.72	0.81	0.87	2.07	2.25

$Q_1 = \mathcal{O}_i \,|\, i = round(16 = 0.25) = 0.35$

$Q_3 = \mathcal{O}_i \,|\, i = round(16 = 0.75) = 0.72$

Fig. 3. An example of calculating IQR. (Color figure online)

Here, we provide outlier detection algorithm in detail.

```
Algorithm 1: Outlier Detection Algorithm
Input: Data set $X = \{x_1, x_2, \quad , x_N\}$, $k$, $r$
Output: Outlier_points
Begin
1. Calculate the distances between all data point pairs and $kNN(x_i)$ for each point;
2. For each data point $x_i$
3.     For each point $x_j$ in $kNN(x_i)$
4.         Calculate the symmetric point of $x_j$ relative to $x_i$ according to Eq. (2);
5.     End For
6.     Calculate the symmetry ratio $R_s(x_i)$ of $x_i$ according to Eq. (3);
7.     Calculate the distance ratio $R_d(x_i)$ of $x_i$ according to Eq. (4);
8.     Calculate the outlier degree $O(x_i)$ of $x_i$ according to Eq. (5);
9. End For
10. Normalize the outlier degree of all data points;
11. Calculate the upper bound ub according to Eq. (6)-Eq. (10);
12. For each data point $x_i$
13.     If $O(x_i) > ub$
14.         $Outlier\_points = Outlier\_points \cup \{x_i\}$;
15.     End If
16. End For
End
```

3 Time Complexity

Below, we analyze the time complexity of the outlier detection algorithm.

Line 1: We can use a kd-tree to compute the k-nearest neighbors for each data point, which requires a time complexity of $O(n \log n)$.

Lines 2–11: It takes approximately $O(kn)$ time to calculate the outlier degree for each data point.

Lines 12–16: It requires $O(n)$ time to determine if each data point belongs to an outlier.

In general, k is smaller than $\log n$. Therefore, the proposed algorithm requires approximately $O(n \log n)$ time.

4 Experimental Results

4.1 Experimental Setup

We conducted a series of experiments to validate the performance of our algorithm. As shown in Table 1, our algorithm was tested on a total of 20 datasets, including 4 two-dimensional synthetic datasets [9] and 16 real datasets obtained from UCI [10].

Table 1. Detail of datasets.

No	Datasets	Size	Dimensionality	Description
1	DS1	788	2	Include seven approximately circular clusters
2	DS2	1268	2	Include four linear clusters
3	DS3	1427	2	Include one curved cluster, three small spherical clusters, and global noise
4	DS4	644	2	Include three curved clusters and one spherical cluster
5	Banknote	1372	4	Features extracted from genuine and forged banknote images
6	Pendigits	10992	16	Pre-processed features derived from handwritten digits
7	Rice	3810	7	Measurements of rice grain features for grain quality assessment
8	WDBC	569	30	Features computed from digitized images of breast masses for breast cancer classification
9	Wine	178	13	Chemical analysis results of wines
10	Wifilocation	2000	7	Wi-Fi signal strength measurements for indoor positioning and localization
11	Ionosphere	351	34	Radar signal data collected from the ionosphere
12	Spambase	4610	57	Email features for spam email classification
13	SyntheticControl	600	60	Synthetic control charts for time series anomaly detection
14	German	1000	20	Credit information of individuals for credit risk assessment
15	Vertebral	310	6	Measurements taken from patients' X-ray images of the spine for spinal diagnosis and classification

(*continued*)

Table 1. (*continued*)

No	Datasets	Size	Dimensionality	Description
16	Breast	286	9	Features extracted from mammogram images for breast cancer diagnosis and classification
17	Yeast	1484	8	Protein localization sites in yeast cells
18	Userknowledge	403	5	User ratings and preferences for different knowledge-based systems
19	Phishing	1353	9	Features extracted from URLs to detect phishing websites
20	CMC	1473	9	Socio-demographic information of married women and their contraceptive method choices

In addition, we utilized four clustering algorithms in our experiments: DPC [11], SMKNN [12], SMMP [13], and CTCEHC [14]. Among these algorithms, DPC is density-based, while the remaining three are graph-based methods. Furthermore, our comparative analysis included four additional denoising algorithms: AutoEncoder [15], IForest [16], LOF [17], and MGBTAI [18]. Each of these algorithms employs a distinct approach for outlier detection:

1. AutoEncoder: This algorithm identifies outliers by analyzing the reconstruction errors generated during the encoding-decoding process.
2. IForest: IForest utilizes a forest of trees and random feature selection to effectively isolate outliers.
3. LOF: LOF determines outliers by considering data point densities and evaluating the local neighborhood density of each data point.
4. MGBTAI: MGBTAI utilizes a multi-generational binary tree structure to identify outliers in the data.

The aforementioned 4 comparative denoising algorithms and 4 clustering algorithms were all executed with default parameters. In our proposed algorithm, the k value for k-nearest neighbors was set to 10, and the r value was set to 0.5.

All experiments were conducted on a computer running the Windows 11 operating system, equipped with an AMD R7-6800H CPU and 16 GB of RAM. The implementation of all algorithms was performed using MATLAB R2022a.

We utilized three metrics, namely Precision (PR), Recall (RE), and F1-Measure, to evaluate the clustering performance [19].

$$PR = \frac{TP}{TP + FP} \tag{11}$$

$$RE = \frac{TP}{TP + FN} \tag{12}$$

$$F = 2 \cdot \frac{PR \cdot RE}{PR + RE} \tag{13}$$

4.2 2D Synthetic Dataset Experiments

The four comparative denoising algorithms and our proposed algorithm were tested on four synthetic datasets. The results on the 2D synthetic dataset are shown in Fig. 4. Normal points are represented by blue dots, while outliers are represented by red dots.

From Fig. 4, it can be observed that AutoEncoder misclassifies some normal points as noise in DS1 and DS3. AutoEncoder, IForest, and LOF, these three denoising algorithms fail to identify the global noise present in DS3. In contrast, both MGBTAI and our algorithm exhibit good performance on DS3, as they are capable of detecting the global noise. However, MGBTAI misclassifies some normal points in DS1 and DS2 as noise. Overall, our proposed algorithm accurately identifies the noise in all four datasets.

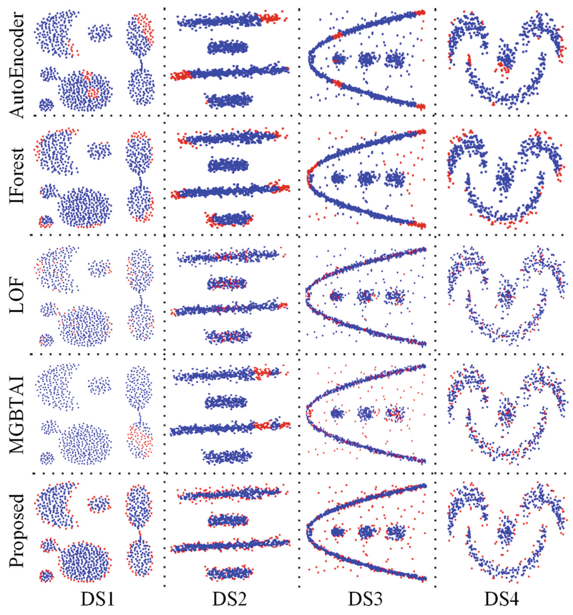


Fig. 4. The Denoising Results on the 2D Synthetic Dataset. (Color figure online)

4.3 Real Dataset Experiments

We conducted clustering experiments on six real datasets using four clustering algorithms. These six datasets include Banknote, Pendigits, Rice, WDBC, Wine, and Wifilocation datasets. The experimental results are presented in Table 2 and Table 3, where the optimal results are indicated in bold. In Table 2 and Table 3, "Origin" indicates clustering performed on the original dataset without any denoising. The remaining rows ("AutoEncoder," "IForest," "LOF," "MGBTAI," and "Proposed") represent clustering on the denoised dataset obtained by applying denoising algorithms. Noise points were assigned to the cluster of their nearest normal points after clustering.

From Table 2 and Table 3, it is evident that the clustering performance on the denoised datasets is significantly better than that on the original datasets. Additionally, the clustering performance is not only influenced by the denoising process but also by the choice of clustering algorithm. For instance, when applying our proposed algorithm for denoising on the Banknote dataset, CTCEHC achieves an F1 score of 0.99. However, when using SMKNN as the clustering algorithm, the F1 score drops to only 0.66. After applying our proposed algorithm for denoising, the four clustering algorithms achieve favorable results on most of the datasets.

To further validate the effectiveness of our proposed algorithm, we conducted clustering experiments on 10 additional real datasets. These 10 additional datasets are listed as the last 10 entries in Table 1. The results are presented in Fig. 5. In Fig. 5, "DPC_0" represents clustering performed on the original dataset using the DPC algorithm. "DPC_1" represents the denoising of the dataset first, followed by clustering the denoised dataset using the DPC algorithm. The same interpretation applies to other algorithms with numerical suffixes. From Fig. 5, it is evident that the clustering performance on the denoised datasets outperforms that on the original datasets.

Table 2. Clustering results on Banknote, Pendigits, and Rice datasets.

	Dataset	Banknote			Pendigits			Rice		
	Metric	PR	RE	F1	PR	RE	F1	PR	RE	F1
DPC	Origin	0.82	0.77	0.79	0.70	**0.69**	**0.70**	**0.91**	0.90	**0.91**
	AutoEncoder	0.58	0.58	0.58	0.58	0.68	0.63	**0.91**	**0.91**	**0.91**
	IForest	**0.98**	**0.98**	**0.98**	**0.71**	**0.69**	**0.70**	**0.91**	**0.91**	**0.91**
	LOF	**0.98**	**0.98**	**0.98**	**0.71**	**0.69**	**0.70**	**0.91**	**0.91**	**0.91**
	MGBTAI	**0.98**	**0.98**	**0.98**	**0.71**	**0.69**	**0.70**	0.85	0.76	0.80
	Proposed	**0.98**	**0.98**	**0.98**	**0.71**	**0.69**	**0.70**	**0.91**	**0.91**	**0.91**
SMKNN	Origin	**0.78**	0.52	0.63	0.67	0.42	0.52	0.73	0.63	0.68
	AutoEncoder	0.75	**0.59**	**0.66**	0.58	0.32	0.41	**0.75**	0.66	**0.70**
	IForest	0.74	0.55	0.63	**0.71**	0.59	0.64	**0.75**	0.66	**0.70**
	LOF	0.74	0.55	0.63	0.54	0.40	0.46	0.75	0.50	0.60
	MGBTAI	0.75	**0.59**	**0.66**	0.64	0.50	0.56	0.71	**0.69**	**0.70**
	Proposed	0.75	**0.59**	**0.66**	**0.71**	**0.69**	**0.70**	**0.75**	0.65	**0.70**

(*continued*)

Table 2. (*continued*)

	Dataset	Banknote			Pendigits			Rice		
	Metric	PR	RE	F1	PR	RE	F1	PR	RE	F1
SMMP	Origin	**0.75**	**0.59**	**0.66**	**0.89**	0.81	0.85	0.77	0.75	0.76
	AutoEncoder	**0.75**	**0.59**	**0.66**	0.85	0.74	0.79	**0.82**	0.77	0.79
	IForest	**0.75**	**0.59**	**0.66**	0.84	0.81	0.83	0.80	0.80	0.80
	LOF	0.27	0.45	0.33	0.85	0.74	0.79	0.54	0.54	0.54
	MGBTAI	**0.75**	**0.59**	**0.66**	0.87	**0.84**	**0.86**	0.81	**0.81**	**0.81**
	Proposed	**0.75**	**0.59**	**0.66**	0.87	**0.84**	**0.86**	0.53	0.53	0.53
CTCEHC	Origin	0.91	0.91	0.91	0.69	0.61	0.65	0.70	0.61	0.65
	AutoEncoder	0.88	0.83	0.86	0.76	0.72	0.74	0.71	0.62	0.66
	IForest	0.85	0.73	0.79	0.85	0.84	0.84	0.70	0.61	0.65
	LOF	0.81	0.75	0.78	0.79	0.78	0.79	0.71	0.61	0.66
	MGBTAI	0.85	0.74	0.79	0.85	0.81	0.83	**0.90**	**0.90**	**0.90**
	Proposed	**0.99**	**0.99**	**0.99**	**0.89**	**0.87**	**0.88**	**0.90**	**0.90**	**0.90**

Table 3. Clustering results on WDBC, Wine, and Wifilocation datasets.

	Dataset	WDBC			Wine			Wifilocation		
	Metric	PR	RE	F1	PR	RE	F1	PR	RE	F1
DPC	Origin	0.31	0.47	0.37	**0.90**	0.90	**0.90**	0.62	0.69	0.65
	AutoEncoder	0.31	0.47	0.37	0.88	0.88	0.88	0.85	0.84	0.84
	IForest	0.34	0.46	0.39	**0.90**	0.90	**0.90**	**0.86**	**0.85**	0.85
	LOF	0.33	0.46	0.39	0.83	0.82	0.82	**0.86**	**0.85**	**0.86**
	MGBTAI	0.33	0.46	0.38	0.82	0.80	0.81	0.62	0.68	0.65
	Proposed	**0.42**	**0.48**	**0.45**	**0.90**	**0.91**	**0.90**	**0.86**	**0.85**	**0.86**
SMKNN	Origin	0.92	0.90	0.91	0.90	0.91	0.91	0.58	0.52	0.55
	AutoEncoder	0.92	0.90	0.91	0.83	0.78	0.80	**0.87**	**0.76**	**0.81**
	IForest	**0.94**	**0.91**	**0.93**	**0.91**	0.91	0.91	**0.87**	**0.76**	**0.81**
	LOF	**0.94**	**0.91**	0.92	**0.91**	0.91	0.91	0.83	0.52	0.64
	MGBTAI	0.90	0.88	0.89	0.81	0.74	0.77	**0.87**	**0.76**	**0.81**
	Proposed	0.92	0.85	0.88	**0.91**	**0.92**	**0.92**	**0.87**	**0.76**	**0.81**

(*continued*)

Table 3. (*continued*)

	Dataset	WDBC			Wine			Wifilocation		
	Metric	PR	RE	F1	PR	RE	F1	PR	RE	F1
SMMP	Origin	0.90	0.79	0.84	0.90	0.91	0.91	0.87	0.76	0.81
	AutoEncoder	0.90	0.80	0.85	0.91	0.92	0.91	**0.94**	**0.94**	**0.94**
	IForest	**0.91**	**0.81**	**0.86**	0.92	0.93	**0.93**	**0.94**	**0.94**	**0.94**
	LOF	0.57	0.54	0.56	0.92	0.93	**0.93**	0.94	0.93	0.94
	MGBTAI	**0.91**	**0.81**	**0.86**	0.89	0.89	0.89	0.87	0.76	0.81
	Proposed	0.57	0.54	0.55	**0.93**	**0.94**	**0.93**	**0.94**	**0.94**	**0.94**
CTCEHC	Origin	0.91	0.82	0.86	0.83	0.81	0.82	0.90	0.85	0.87
	AutoEncoder	0.94	0.91	0.92	0.78	0.74	0.76	**0.95**	**0.94**	**0.95**
	IForest	0.94	0.91	0.92	0.78	0.75	0.76	0.90	0.85	0.88
	LOF	0.93	0.91	0.92	0.84	0.81	0.83	**0.95**	**0.94**	**0.95**
	MGBTAI	**0.95**	**0.93**	**0.94**	**0.91**	**0.92**	**0.92**	**0.95**	**0.94**	0.94
	Proposed	0.91	0.90	0.90	0.84	0.81	0.83	0.90	0.90	0.90

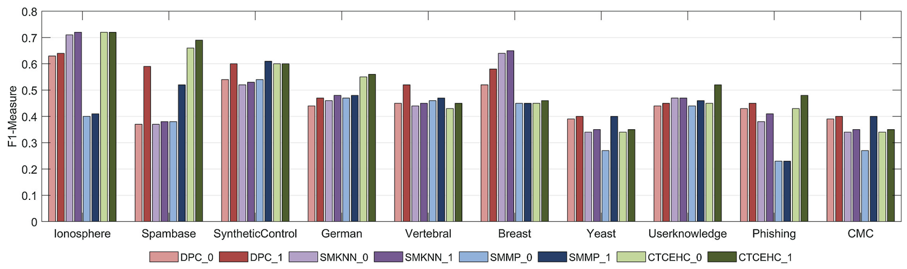


Fig. 5. Clustering results on 10 real datasets using our algorithm and the DPC algorithm.

4.4 Parameter Analysis

The parameters involved in our algorithm are k and r. The value of k in the k-nearest neighbors is generally set to 10. Here, we focus on analyzing the impact of different r values on the clustering performance. Figure 6 illustrates the F1 scores of the DPC algorithm on six datasets as r varies from 0 to 3. From Fig. 6, it can be observed that as r increases, the F1 scores remain relatively stable. The optimal performance is achieved when r is set to 0.5 for all six datasets. Therefore, we choose r as 0.5 in our algorithm.

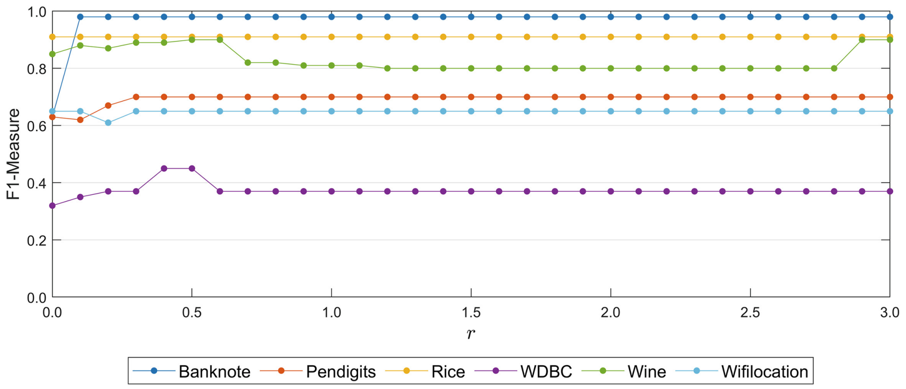


Fig. 6. The effects of parameter r on clustering results.

5 Conclusions

In conclusion, this study presents a novel outlier detection algorithm that addresses the limitations of traditional methods based on distance or density measures. By introducing the concepts of symmetry ratio and distance ratio, our algorithm accurately quantifies the outlier degree of data points. Moreover, the interquartile range method is utilized to automatically determine the threshold for the outlier degree. Experimental results on synthetic and UCI real datasets demonstrate the excellent performance of our algorithm in detecting outliers. It is observed that our algorithm remains robust and unaffected by factors such as data point density and shape. We plan to conduct research on outlier detection algorithms that are tailored for diverse data types and subsequently apply these algorithms to datasets within their respective domains.

References

1. Hawkins, D.M.: Identification of outliers. Springer (1980)
2. Askari, S.: Fuzzy C-Means clustering algorithm for data with unequal cluster sizes and contaminated with noise and outliers: review and development. Expert Syst. Appl. **165**, 113856 (2021)
3. Gupta, M.K., Chandra, P.: A comprehensive survey of data mining. Int. J. Inf. Technol. **12**, 1243–1257 (2020)
4. Boukerche, A., Zheng, L., Alfandi, O.: Outlier detection: Methods, models, and classification. ACM Computing Surveys (CSUR) **53**, 1–37 (2020)
5. Nassif, A.B., Talib, M.A., Nasir, Q., Dakalbab, F.M.: Machine learning for anomaly detection: A systematic review. Ieee Access **9**, 78658–78700 (2021)
6. Verma, K.K., Singh, B.M., Dixit, A.: A review of supervised and unsupervised machine learning techniques for suspicious behavior recognition in intelligent surveillance system. Int. J. Inf. Technol. **14**, 397–410 (2022)
7. Seo, S.: A Review and Comparison of Methods for Detecting Outliers in Univariate Data Sets (2006)
8. Shrifan, N.H.M.M., Akbar, M.F., Isa, N.A.M.: An adaptive outlier removal aided k-means clustering algorithm. J. King Saud Univ. Comput. Inf. Sci. 34, 6365–6376 (2021)

9. Wang, Y., Ma, Y., Huang, H.: A neighborhood-based three-stage hierarchical clustering algorithm. Multimedia Tools and Applications **80**, 32379–32407 (2021)
10. Blake, C.L.: UCI repository of machine learning databases. http://www.ics.uci.edu/~mlearn/MLRepository.html (1998)
11. Rodriguez, A., Laio, A.: Clustering by fast search and find of density peaks. science 344, 1492–1496 (2014)
12. Wang, Y., Ma, Y., Huang, H., Wang, B., Acharjya, D.P.J.I.S.: A split–merge clustering algorithm based on the k-nearest neighbor graph. 111, 102124 (2023)
13. Guan, J., Li, S., He, X., Zhu, J., Chen, J., Si, P.J.I.T.o.P.A., Intelligence, M.: SMMP: a stable-membership-based auto-tuning multi-peak clustering algorithm. 45, 6307–6319 (2022)
14. Ma, Y., Lin, H., Wang, Y., Huang, H., He, X.J.I.S.: A multi-stage hierarchical clustering algorithm based on centroid of tree and cut edge constraint. 557, 194–219 (2021)
15. Yin, C., Zhang, S., Wang, J., Xiong, N.N.J.I.T.o.S., Man,, Systems, C.: Anomaly Detection Based on Convolutional Recurrent Autoencoder for IoT Time Series. 52, 112–122 (2022)
16. Liu, F.T., Ting, K.M., Zhou, Z.-H.J.E.I.I.C.o.D.M.: Isolation Forest. 413–422 (2008)
17. Breunig, M.M., Kriegel, H.-P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: ACM SIGMOD Conference. (Year)
18. Sarkar, J., Saha, S., Sarkar, S.J.A.I.: Efficient anomaly identification in temporal and non-temporal industrial data using tree based approaches. 53, 8562–8595 (2022)
19. Duan, X., Ma, Y., Zhou, Y., Huang, H., Wang, B.: A novel cluster validity index based on augmented non-shared nearest neighbors. Expert Syst. Appl. **223**, 119784 (2023)

Graph Regression Based on Autoencoders and Graph Autoencoders

Francesc Serratosa(✉)

Universitat Rovira i Virgili, Tarragona, Catalonia, Spain
francesc.serratosa@urv.cat
https://webs-deim.urv.cat/francesc.serratosa/

Abstract. Graph Autoencoders are a generalisation of Autoencoders in which the elements are structured as graphs. Since now, both models have been applied separately. This paper introduces AE+GAE, a new model for graph regression that combines both. Graph Autoencoders assume node attributes are related to their local structure. Nevertheless, in some applications, not all the node attributes have the property of being dependent of their local structure. Our method learns which attributes do have this property and which ones do not. This is done by feeding all the attributes to both models and combining their latent domain by a neural network. AE+GAE has been applied to predict the Energy, pIC_{50} and the binding affinity of drugs, which are represented as attributed graphs but could be used in other fields as well. The method demonstrates improved performance compared to other previously presented models.

Keywords: Graph regression · Graph Autoencoders · Graph Neural Networks

1 Introduction

Graph Neural Networks (GNNs) [28] currently demonstrate remarkable achievements across various applications involving the classification or regression of objects represented as attributed graphs. Examples include Graph Convolutional Networks (GCN) [9], Graph Transformers [32], Graphorners [31], and GIN [30]. These models operate under the assumption of a connection between node features and their connected nodes. A prevalent assumption is that connected nodes exhibit similar features, as observed in Graph Convolutional Networks, often associated with low pass filtering. In contrast, GIN is perceived as employing high pass filtering. Therefore, across all these models, there exists an implicit assumption of a relationship between node attributes and the local structure [29].

Supported by the project NextPandemics: "Development of new computational tools for drug discovery against current and future pandemics", PID2022-138327OB-I00.

Existing models for predicting drug potency rely on portraying drugs as graphs and employing a Graph Neural Network (GNN) as the underlying architecture. However, there is a lack of evidence demonstrating a consistent relationship between all atom features (node attributes) and the local structure. For instance, there is no observed connection between the atom type (Nitrogen, Oxygen, etc.) and a propensity to be linked to specific other atom types through particular bonds. In such scenarios, it appears illogical to apply any established GNNs that presuppose this type of relationship.

In this paper, we propose a GNN called AE+GAE, which combines an Autoencoder (AE) and a Graph AutoEncoder (GAE) [9,11] with the aim of automatically selecting if the node features have a local relation or they are independent of their local structure. This model have been applied to drug potency prediction achieving lower mean square error (MSE) than classical architectures such as Neural Networks, GNNs, AEs or GAEs.

In the next section, we present a summarised overview of the models for graph regression applied to drug prediction, which have been involved in this work. In Sect. 3, we explain our approach in detail. In Sect. 4, we show the experimental validation. In Sect. 5, we conclude the paper.

2 Models for Graph Regression

This section summarises some models for graph regression, which we have used to compare our new proposal. We discern between the ones that the input is a vector (Sect. 2.1) and the ones that the input is an attributed graph (Sect. 2.2).

Note that frequently, graphs are used to represent chemical compounds [25]. In these cases, the graph edit distance [20–24,26] was applied to define a distance between chemical compounds and usually, some methods were applied for learning the edit costs [1–3,16,18]. Finally, regression or classification algorithms were applied, depending on the problem at hand [6,17].

2.1 Models Based on Vectors

In these models, the adjacency matrix is not considered and only the vector of node properties are used. Thus, the input is a vector $\boldsymbol{X}$ that each element has the properties of a node. Note that attributed graphs in the database have different number of nodes, but these models need the length of this vector to be constant, then vectors that represent the graphs are enlarged to have all of them the same length n.

Neural Networks. A classical and well known neural network (NN) is applied for regression. The input is $\boldsymbol{X}$, there are several fully-connected layers with the sigmoid function and the last one is a read-out with a linear function.

Autoencoders. An AutoEncoder, AE, is a particular class of NN that is employed in machine learning to capture the most basic representations of an entity. To achieve this, it is trained to reconstruct the input data after having generated an intermediate data called latent space [7,10]. An AE consists of two components: an encoder that converts the input space into a latent space, $\boldsymbol{Z_{sem}}$, and a decoder that converts the lower-dimensional representation back to the original input space. Both encoders and decoders include non-linear activation functions. An unsupervised learning algorithm is used since the loss function is based on minimising the difference between the input data $\boldsymbol{X}$ and the generated data $\hat{\boldsymbol{X}}$.

When this process is finished, a fitting model is trained to deduce the global properties, such as a simple regression or a NN, given the input $\boldsymbol{Z_{sem}}$. Figure 1 shows this architecture.

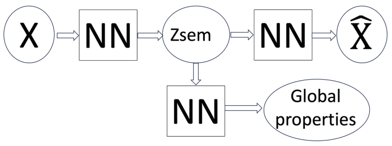


Fig. 1. Regression model based on an AE. Input: Node features X. Z_{sem} is the latent domain. The model reconstructs the node features $\hat{X}$

2.2 Structured Models

These models not only consider the node properties but the existence of binary relations between them. In this way, the input moves from a classical vector to an attributed graph. Equally than the previous models, node properties are represented by a vector $\boldsymbol{X}$ and they are seen as the node features. Binary relations between atoms are represented by a graph adjacency matrix $\boldsymbol{A}$. Thus, we have the element we want to obtain its global properties represented as the graph **G(X,A)**.

Specifically, $\boldsymbol{X} \in \mathbb{R}^{n \times f}$ is a matrix of size $n \times f$, with n being the number of nodes and f being the number of attributes or node features. The adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is of size $n \times n$, where the $\boldsymbol{A}_{i,j} = 1$ if there is an edge between the i^{th} and the j^{th} node and 0 otherwise. In the models we have analysed, graph's edges are unattributed and undirected, meaning that if there is an edge from node i to node j, there is also an edge from node j to node i, which is represented by the equality $\boldsymbol{A}_{i,j} = \boldsymbol{A}_{j,i}$.

Graph Neural Networks. The key idea behind Graph Neural Networks, GNNs, is to use the information from the neighbouring nodes to update the node's representation. This can be accomplished by defining a convolution operation on the graph, which is typically implemented as a weighted sum of the representations of the neighbouring nodes. A learnable weight matrix is often used to determine the weights of this sum. Usually, after this first step of involving nodes and edges, there is a fully-connected NN that is used to infer the global properties. As commented in the introduction, several GNN architectures have been defined, such as Graph Convolutional Networks [9], Graph Transformers [32], Graphorners [31] or GIN [30]. Figure 2 shows the main scheme of this architecture.

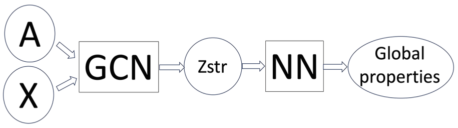


Fig. 2. Regression model based on a GNN. Input: Node features X and adjacency matrix A. Z_{str} is the latent domain.

Graph Autoencoders. Graph AutoEncoders, GAEs are a generalisation of AE in which the input is an attributed graphs instead of a matrix [9]. Just like the classical AE, GAEs are composed of two main parts: an encoder and a decoder. The encoder embeds input graphs through a GNN as defined in [9] returning a latent matrix $\boldsymbol{Z_{str}} \in \mathbb{R}^{n \times b}$ with the graph's unique properties. The number of features in the latent space is b. Equation 1 shows the encoder's function:

$$\boldsymbol{Z_{str}} = GNN(\boldsymbol{X}, \boldsymbol{A}) \tag{1}$$

And Eq. 2 and Eq. 3 show the decoder:

$$\boldsymbol{A^*} = \sigma\left(\boldsymbol{Z_{str}} \boldsymbol{Z'_{str}}\right) \tag{2}$$

$$\boldsymbol{\hat{A}} = binary\left(\boldsymbol{A^*}\right) \tag{3}$$

where $\boldsymbol{\sigma}\left(\cdot\right)$ is the sigmoid function and the symbol $\boldsymbol{'}$ means the transposed matrix. The output of $\boldsymbol{\sigma}\left(\cdot\right)$ is a matrix of real numbers between 0 and 1 that represents the probability of an existing edge in the reconstructed adjacency matrix. Note that in order to deduce the final reconstructed matrix, a *binary* function is applied to $\boldsymbol{\sigma}\left(\cdot\right)$ to discern between non-edge and edge, i.e., zero and one values. Figure 3 shows the main scheme of this architecture.

3 Proposed Approach: AE+GAE

The foundation of the Graph Autoencoder (GAE) approach relies on the premise that knowledge associated with nodes is interconnected with the knowledge associated with edges, and vice versa [29]. This assumption posits a relationship

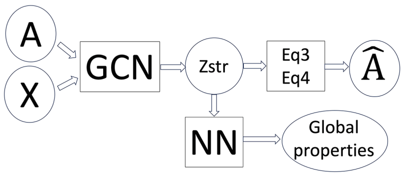


Fig. 3. Regression model based on a GAE. Input: Node features X and adjacency matrix A. Z_{str} is the latent domain. The model reconstructs the adjacency matrix $\hat{A}$

between local structural patterns and node attributes, as discussed in Sect. 1. However, not all node features adhere to this assumption. We have devised a tailored model capable of handling graphs where nodes possess attributes influenced by structural patterns, as well as those unrelated to edges. The architecture is outlined in Sect. 3.1, while Sect. 3.2 provides insights into the learning process.

3.1 Architecture

Specifically, our approach involves inputting the graph matrices $\boldsymbol{X}$ and $\boldsymbol{A}$ into a Graph Autoencoder (GAE) [9]. Simultaneously, we input the node attributes ($\boldsymbol{X}$) into an Autoencoder (AE) [12]. Both modules project their respective data into a latent domain, which is then utilized in any fitting mechanism, such as a Neural Network (NN). The architecture is illustrated in Fig. 4. In our experiments, we utilized the GAE as defined in [8] and summarized in Sect. 2.2, and the AE as defined in [12].

It is crucial to emphasise that both the Autoencoder (AE) and the Graph Autoencoder (GAE) play a role in feature extraction during the encoding stage, which is essential for subsequent prediction tasks. However, our model offers the additional capability to reconstruct the entire graph ($\hat{\boldsymbol{X}}$ and $\hat{\boldsymbol{A}}$), enabling it to function as a graph generative model. This feature proves valuable for validating the efficacy of the encoding process.

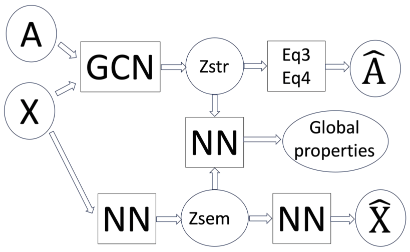


Fig. 4. Regression model based on a AE+GAE. Input: Node features X and adjacency matrix A. Z_{str} and Z_{sem} are the latent domains. The model reconstructs the node features $\hat{X}$ and the adjacency matrix $\hat{A}$.

A comparable approach was outlined in [5]; however, in that instance, the determination of which attributes were fed into each of the models was predefined. In our approach, all features are employed in both models (AE and GAE), and the model autonomously learns their relative importance. In [5], the choice of which node attributes to utilise in the AE and GAE involved a validation process. This process might involve training the model on various subsets of selected attributes (randomly choosing attributes for each architecture) and identifying the combination that yields the lowest loss for both. Alternatively, for specific tasks, one could base this decision on domain knowledge of the problem at hand.

Our architecture's latent space is constructed by merging the latent space of the Autoencoder (AE), denoted as $\boldsymbol{Z_{sem}}$, with that of the Graph Autoencoder (GAE), denoted as $\boldsymbol{Z_{str}}$. Graphs inherently possess the property of being node-position invariant, meaning their structure remains invariant to the order of the nodes. To achieve this invariance, various aggregation methods such as sum, mean, minimum, or maximum can be applied to each feature across all nodes. We opted for the mean calculation as it imparts the architecture with independence from the number of nodes, a process commonly referred to as global average pooling. Consequently, given the $\boldsymbol{z_{str}}$ vectors, the $\boldsymbol{r_{str}}$ vector is generated by computing their mean. Notably, the length of the vector $\boldsymbol{r_{str}}$ remains unaffected by the number of nodes n, indicating that the system can be trained with graphs featuring different node counts.

Ultimately, the combined vector comprising both $\boldsymbol{r_{sem}}$ and $\boldsymbol{r_{str}}$ is employed to feed into a regression module, aiming to ascertain the global property of the graph.

3.2 The Learning Process

The learning process is achieved in two steps. In the first step, given all graphs G^g of the learning database, where $g = 1, ..., k$, the weights of the AE and the GAE are learned. This process is done by an unsupervised learning algorithm such that the aim is to achieve the output graphs to be as much similar as possible to the input graphs in the learning database.

The minimisation criterion is expressed in Eq. 4:

$$\boldsymbol{\mathcal{L}} = \frac{1}{k}\sum_{g=1}^{k} \mathcal{L}_{str}^{g} + \mathcal{L}_{sem}^{g} \tag{4}$$

where,

$$\mathcal{L}_{str}^{g} = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{pos} A_{i,j}^{g} log A_{i,j}^{*g} + w_{neg}(1 - A_{i,j}^{g}) log(1 - A_{i,j}^{*g}) \tag{5}$$

describes the loss function of reconstructing the adjacency matrix (edges of the graph) per each graph G^g, where w_{pos} and w_{neg} are weights included to compensate an unbalancing problem. Usually, there are less nodes connected by edges than nodes non-connected by edges. Thus, the sum of these weights is 1 and they are defined as follows: w_{neg} counts for the number of edges divided by n^2. And $w_{pos} = 1 - w_{neg}$.

Besides,

$$\mathcal{L}^g_{sem} = \frac{1}{n}\sum_{i=1}^{n} norm(X^g_i - \hat{X}^g_i) \tag{6}$$

describes the loss function of reconstructing the node attributes per each graph G^g.

In the second step, given the returned latent vectors $\boldsymbol{r^g_{sem}}$ and $\boldsymbol{r^g_{str}}$ of all graphs G^g in the training set, the regression weights are learned, where $g = 1, ..., k$. This is a supervised algorithm and several architectures can be used, such as NN applied to regression problems or a simple linear regression.

4 Experimental Validation

We applied AE+GAE to predict the pIC_{50}, the binding affinity and the chemical energy in three drug databases. These are three related metrics to having the ability to stop or reduce unwanted biological processes.

On the one hand, half maximal inhibitory concentration, $IC50$, is a measure of the potency of a substance in inhibiting a specific biological function. $IC50$ is a quantitative measure that indicates how much of a particular drug is needed to inhibit a given biological process by 50%. The biological component could be an enzyme, cell or microorganism. The pIC_{50} is computed as $pIC_{50} = -log(IC50)$. Then, higher values of pIC_{50} indicate exponentially more potent inhibitors. pIC_{50} values are typically expressed as molar concentration.

On the other hand, the binding affinity is the interaction of drugs to some proteins. In general, high-affinity drug binding results from greater attractive forces between the drug and its receptor while low-affinity drug binding involves less attractive force. A drug that can bind to a protein might alter the function of the protein and inhibit a given biological process.

Finally, the chemical energy is the energy of chemical substances that is released when the substances undergo a chemical reaction and transform into other substances.

In the following section, we comment the used databases. Then we summarise the architecture setting in Sect. 4.2. After that, in Sect. 4.3 we show the mean square error (MSE) of the predicted pIC_{50} and the binding affinity obtained by our method and five other ones, which have been summarised in Sect. 2.

4.1 Databases

This section summarises the three used databases: QM7, SARS-CoV-2 M-pro database and IEDB-nonamers database. They are publicly available at[1] and they have in common that graphs represent chemical compounds and also that there is a global property per graph.

[1] https://github.com/ASCLEPIUS-URV?tab=repositories.

QM7 Database. This database is a small subset of a quantum mechanics database[2] composed of 7165 molecules of up to 23 atoms per compound. While the original dataset contains information about different features, e.g., Coulomb Matrices, atomisation energies, and atomic charge, only a selected set of the molecular features was used for training the model. The dataset was preprocessed to generate graphs where edges resembled the connected atoms in a molecule, while the atomic energy, the Cartesian coordinates of the atoms and the number of bonds per atom were registered as features of the node. Only the first 200 compounds were used for learning purposes and the rest of them were used for testing. The compound energy is the global property.

SARS-CoV-2 M-Pro Database. The learning dataset consisted of 223 M-pro crystallised structures bound to an inhibitor for which its pIC_{50} is known. The learning set is composed of 160 compounds, 53 of them come from the well know Protein Data Bank (PDB) database[3] and the other 107 structures come from FRAGALYSIS[4] database. The test set is composed of 63 compounds. Although PDB and FRAGALYSIS databases have a lot of components, we have only the pIC_{50} from a biotechnological lab of few of them. This database was used in drug discovery in [4]. The pIC_{50} is the global property.

For a given drug-protein pair, a singular attributed graph is generated, encapsulating the entire drug along with the atoms and bonds of the protein in proximity to any drug atom forming a bond. The graph nodes represent atoms from both the drug and the protein, while graph edges depict bonds present in both entities. Node attributes encompass the three-dimensional positions of atoms, their atomic numbers, the number of bonds, and electric charges. Edges lack attributes and are present if any type of bond exists between atoms. To accommodate a compound's maximum size of "drug + binding site atoms" at 146, all generated graphs are extended to include 146 nodes. The $pIC50$ is considered a global property in this context.

IEDB-Nonamers Database. The database provided here comprises a curated set of samples sourced from the Immune Epitope Database (IEDB) [27]. IEDB is a valuable repository for investigating distinct diseases, empowering researchers to pinpoint and scrutinize epitopes pertinent to their specific research objectives [13]. Additionally, it serves as a crucial resource for training and developing web servers focused on predicting binding interactions between peptides and major histocompatibility complex molecules [15].

This recently established database specifically incorporates peptides of length 9, referred to as nonamers, and focuses on the HLA-A02:01 allele. The selection of HLA-A02:01 was deliberate, given its prevalence and polymorphism among HLA-A molecules in both human populations and within the Immune Epitope

[2] http://quantum-machine.org/datasets/.
[3] https://www.rcsb.org.
[4] https://fragalysis.diamond.ac.uk/viewer/react/preview/target/Mpro.

Database (IEDB). The decision to focus on nonamers stems from their high frequency as the peptide length most commonly binding to the HLA-A*02:01 allele within the IEDB database. In total, 4872 peptides were chosen for inclusion in this database.

The database was utilised to generate 3D compounds through the application of FoldX [19], a software tool that predicts the stability of protein structures and mutations while providing their binding energies. Employing functions such as *RepairPDB*, *BuildModel*, and *AnalyseComplex* within FoldX, the 3D compounds were created, offering a more detailed understanding of the interactions between peptides and the HLA-A02:01 allele. The template for constructing all 3D compounds was the 5ENW structure of HLA-A02:01 [14]. These complexes involve an HLA molecule and a peptide of interest. To adhere to the constraint of a maximum of 87 atoms, all generated graphs were extended accordingly. The learning set comprises 200 compounds, while the training set comprises 500 compounds. Further details about this database can be found in [5].

4.2 Architecture Setting

In this section, we outline the architectures of our model as well as those of the compared models, with a comprehensive summary provided in Sect. 2.

NN (Sect. 2.1): The structure consists of a fully connected neural network with four layers. The input layer encompasses a number of neurons equivalent to the maximum nodes in the database graphs multiplied by the number of node features. The subsequent layers contain half and a quarter of this neuron count for the second and third layers, respectively. The final layer is comprised of a single neuron. tangent sigmoid activation functions are applied to all layers, with the exception of the last layer, where a linear function is employed.

AE (Sect. 2.1): The architecture comprises a fully connected neural network with four layers. The input and output layers share a common number of neurons, determined by the maximum nodes in the database graphs multiplied by the number of features. For the intermediate layers, the number of neurons is set at half of the maximum node count multiplied by the number of features. All layers, except the final one, utilise tangent sigmoid activation functions, while the last layer employs a linear function. The 3D positions of atoms serve as features inputting the autoencoder (AE), while the remaining atom properties are fed into the graph autoencoder (GAE).

GNN (Sect. 2.2): The structure is a combination of a Graph Neural Network (GNN) linked to a fully connected neural network (NN) featuring two layers. The GNN's input length corresponds to the number of nodes, and its width is determined by the number of features. The initial layer of the NN contains neurons equal to the product of the number of nodes and the number of features. The last layer of the NN consists of a single neuron. Activation functions for the first and second layers in the NN are the tangent sigmoid and the linear function, respectively. The architecture of the GNN can be a Graph Convolutional architecture, GCN [9], a Graph Transformer, GT [31] or a GIN [30].

GAE (Sect. 2.2): It consists of an encoder, precisely mirroring the GCN utilised in the GNN model, and a decoder illustrated in Eq. 2 and Eq. 3. Additionally, there is a neural network (NN) responsible for interpreting the latent space, and it aligns with the one elucidated in the GNN model.

AE|GAE [5]: It comprises an Autoencoder (AE), identical to the one in the AE model, and a Graph Autoencoder (GAE), mirroring the GAE model. However, the dimensions of the input matrices in both models depend on the attributes considered independent of the local structure (AE) and those dependent on it (GAE). The GAE's output is concatenated and supplied to a neural network (NN) with two layers. The first layer's neuron count is determined by the number of nodes in the graphs multiplied by the number of features. Furthermore, the second layer consists of a single neuron. Activation functions for the first and second layers are the tangent sigmoid and the linear function, respectively.

AE+GAE (Sect. 3): It consists of the identical Autoencoder (AE) and Graph Autoencoder (GAE) featured in the model from [5]. However, there is a distinction in the input matrix size, which is now determined by the product of the number of node attributes and the number of attribute nodes in both cases. Additionally, the neural network (NN) architecture differs; as both AE and GAE receive input from all attributes of the graph nodes, the number of neurons in the first layer is double the product of the number of nodes and the number of features.

4.3 Analysis of the Prediction

Table 1 shows the MSE and the standard deviation of the predicted Energy in the QM7 database, the pIC_{50} in the SARS-CoV-2 M-pro database and the binding affinity in the IEDB-nonamers database, given several different models and our proposal, which is AE+GAE.

Table 1. MSE and the standard deviation obtained by several models applied to the QM7, SARS-CoV-2 M-pro and EDB-nonamers databases.

Method	QM7	SARS-CoV	EDB-nonamers
NN	25934 (160)	2.0 (1.4)	69.9 (4.2)
AE	15559 (124)	0.72 (0.8)	31.6 (5.6)
GAE	7317 (85)	0.48 (0.7)	17.2 (4.1)
GCN [9]	2328638 (226)	1.46 (1.0)	38.6 (6.2)
GIN [30]	2129385 (217)	1.72 (0.9)	36.9 (6.9)
GT [31]	12718 (26)	1.03 (0.8)	28.7 (5.7)
AE|GAE [5]	9471 (97)	0.55 (0.7)	26.2 (5.1)
AE+GAE (our model)	5113 (71)	0.32 (0.5)	14.9 (3.8)

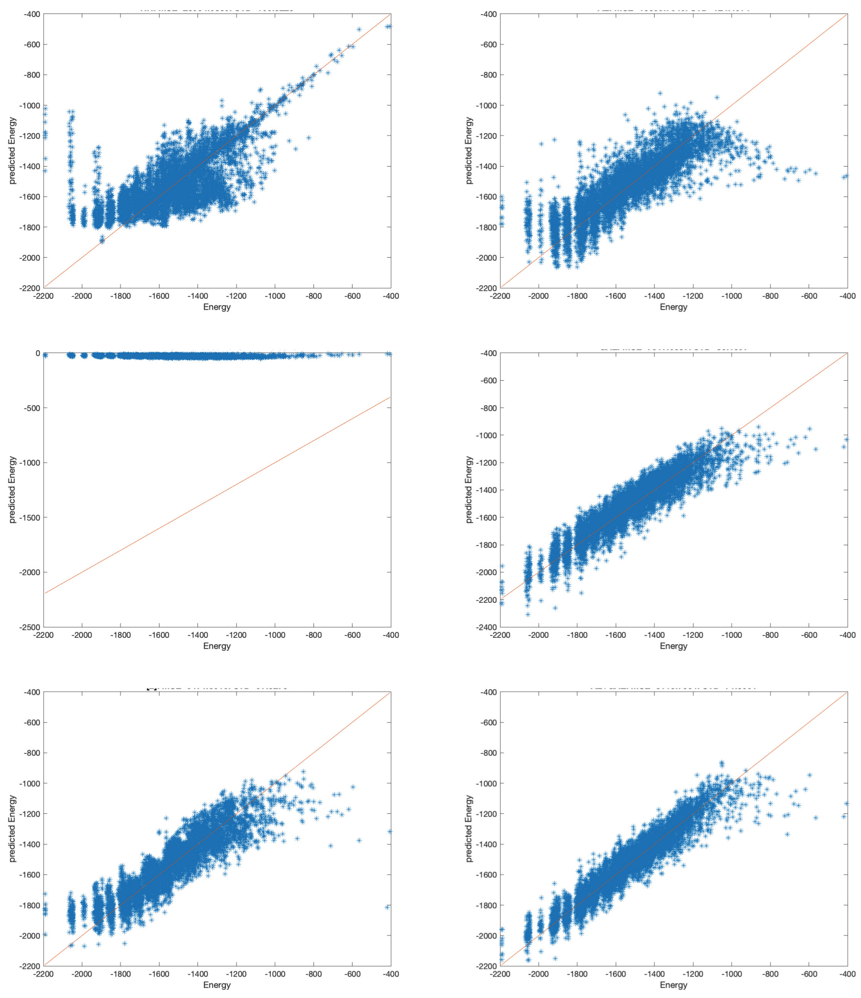


Fig. 5. Predicted versus experimental Energy in QM7 database. Methods: First row: NN and AE. Second row: GNN and GAE. Third row: [4] and AE+GAE

NN and GNN obtain very high MSE, compared to the other models. Considering NN, it is supposed that it is because the bonds between atoms is an important information to keep into the model. Considering GCN and GIN, they are not able to keep the information of the relation between nodes and bonds, thus it seems they are not able to predict both global properties. Note we are not using these models as authors initially suggested since they were presented to perform node classification or regression given only one huge graph. In our case, we have used them to perform graph regression given several graphs. GT obtains good accuracy, although its MSE is higher than GAE.

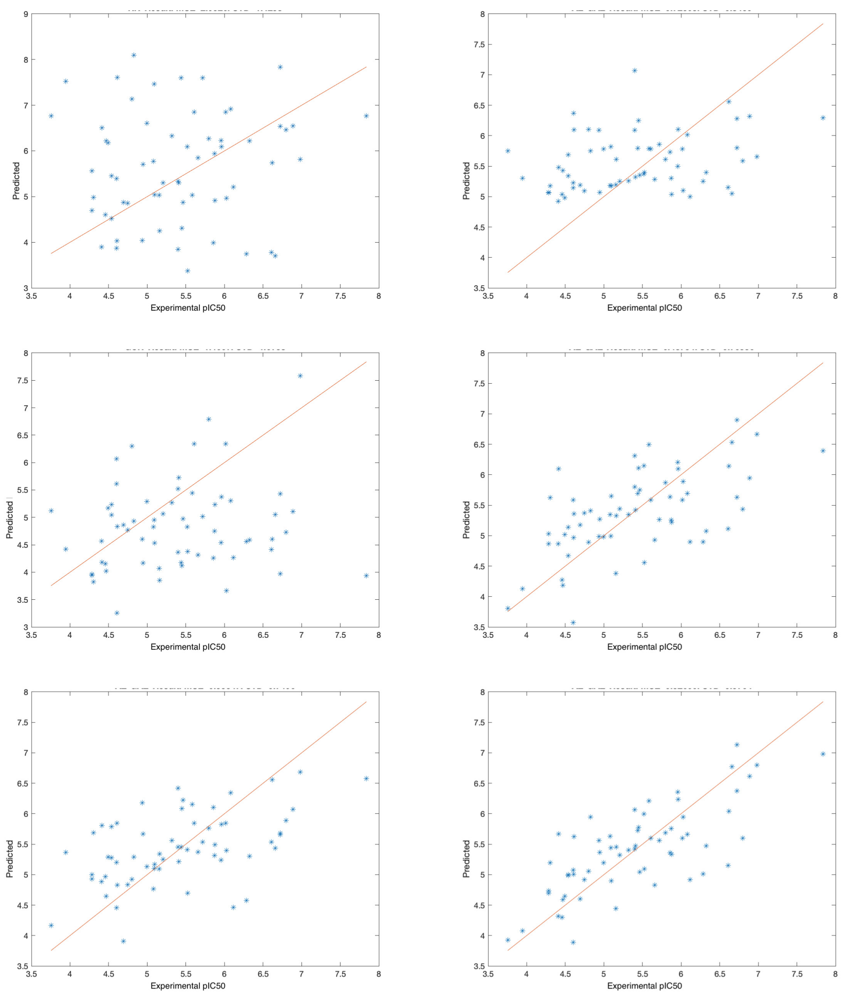


Fig. 6. Predicted versus experimental pIC_{50} in SARS-CoV-2 M-pro database. Methods: First row: NN and AE. Second row: GNN and GAE. Third row: [4] and AE+GAE

Apart from the NN, AE is the other non-structural model that also returns high MSE. Clearly, this fact informs us the bonds between atoms, represented as edges in the graphs, have to be considered in the model. The latent domain in AE is not able to capture all the variability of the initial data. Note the NN is composed of the decoder of the AE connected to a fully connected layer.

Finally, models that incorporate GAEs obtain the lowest MSE. These are [5], GAE and AE+GAE. This means that, in these models, the latent domain is able to capture the semantic and structural information of the data. If we compare

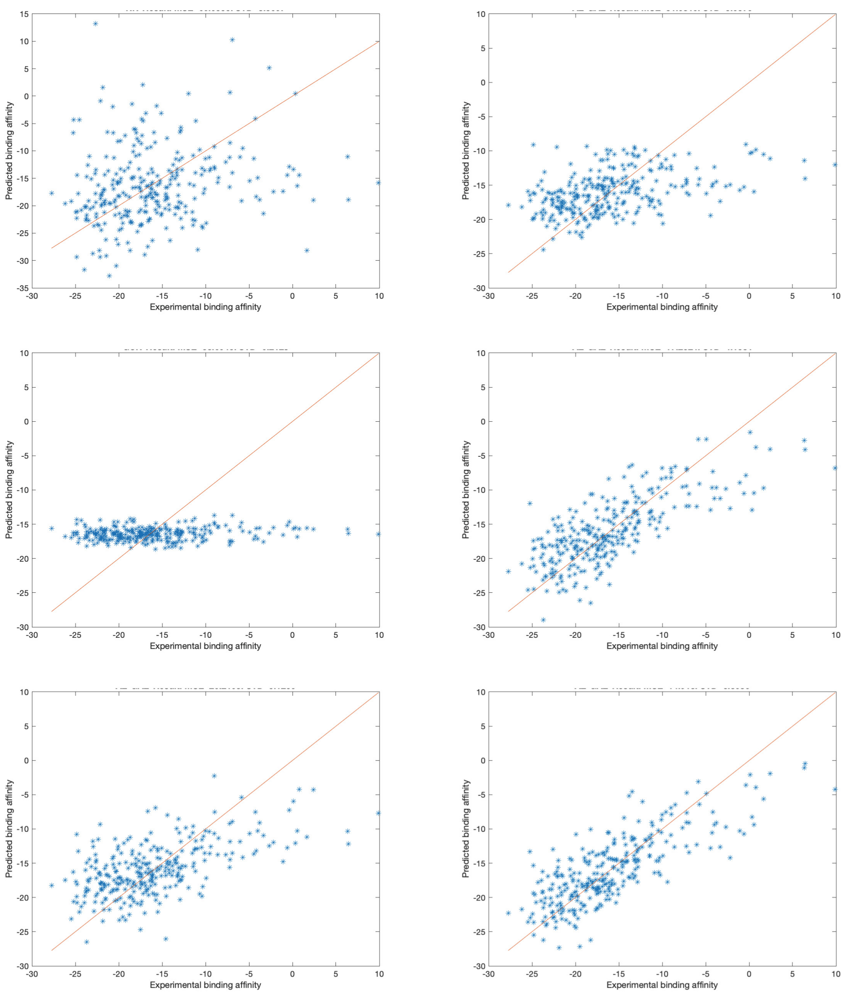


Fig. 7. Predicted versus experimental binding affinity in IEDB-nonamers database. Methods: First row: NN and AE. Second row: GNN and GAE. Third row: [4] and AE+GAE

GAE to AE+GAE, we realise it is worth to incorporate the reconstruction of the node features by AE since AE+GAE returns the lowest MSE. We assume, in this case, the latent domain is able to better capture the variability of the data. As commented in Sect. 2, the main difference between [5] and AE+GAE is that in the first case, the node features are manually split to feed the AE and the GAE. Contrarily, AE+GAE feeds with all the features the AE and the GAE models. Results suggests that the AE+GAE option is the best since the model automatically learns which features depend on the local structure and which ones do not.

Figures 5, 6 and 7 show the predicted versus experimental Energy, pIC_{50} and binding affinity generated by the six methods given QM7 database, ARS-CoV-2 M-pro database and IEDB-nonamers database, respectively. The data in Table 1 has been extracted from this scatter plots. Clearly, it is visible the tendency of following the diagonal line when the GAE is used, and more precisely the AE+GAE, in the three databases.

5 Conclusions

We have presented a graph regression model, called AE+GAE, that has been applied to predict the Energy, the pIC_{50} and the binding affinity. We have selected the drug discovery application because we wanted graphs to have several node attributes with several properties. Moreover, we wanted the node attributes to be independent of the neighbour attributes. Our architecture integrates an Autoencoder and a Graph Autoencoder, both subsequently linked to a Neural Network. A crucial element of our approach lies in the trainable capability to distinguish between the semantic and structural knowledge encoded in the node attributes. This fundamental feature is application-independent, implying that our proposal can be applied across various fields. Empirical experiments demonstrate the model's proficiency in predicting global graph features. Our findings reveal that our model consistently yields lower mean square error compared to other established models. As a future work, we want to model how important is the graph structure versus the graph semantics of each node attribute. That is, to try to define a saliency map of nodes and their attributes.

References

1. Algabli, S., Serratosa, F.: Embedding the node-to-node mappings to learn the graph edit distance parameters. Pattern Recogn. Lett. **112**, 353–360 (2018)
2. Conte, D., Serratosa, F.: Interactive online learning for graph matching using active strategies. Knowledge-Based Systems **205**, 106275 (2020). https://doi.org/10.1016/j.knosys.2020.106275, https://www.sciencedirect.com/science/article/pii/S0950705120304585
3. Cortés, X., Serratosa, F.: Learning graph matching substitution weights based on the ground truth node correspondence. Int. J. Pattern Recognit Artif Intell. **30**(02), 1650005 (2016)
4. Fadlallah, S., Julià, C., García-Vallvé, S., Pujadas, G., Serratosa, F.: Drug potency prediction of sars-cov-2 main protease inhibitors based on a graph generative model. Int. J. Mol. Sci. **24**(10), 8779 (2023)
5. Fadlallah, S., Segura Alabart, N., Julià, C., Serratosa, F.: Splitting structural and semantic knowledge in graph autoencoders for graph regression. In: Graph-Based Representations in Pattern Recognition. pp. 81–91 (2023)
6. Garcia-Hernandez, C., Fernández, A., Serratosa, F.: Ligand-based virtual screening using graph edit distance as molecular similarity measure. J. Chem. Inf. Model. **59**(4), 1410–1421 (2019)

7. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016), http://www.deeplearningbook.org
8. Kipf, T.N.: Deep Learning with Graph-Structured Representations. Ph.D. thesis, University of Amsterdam (apr 2020)
9. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. CoRR **abs/1609.02907** (2016), http://arxiv.org/abs/1609.02907
10. Kramer, M.A.: Nonlinear principal component analysis using autoassociative neural networks. AIChE J. **37**, 233–243 (1991)
11. Le, T., Le, N., Le, B.: Knowledge graph embedding by relational rotation and complex convolution for link prediction. Expert Systems With Applications **214** (2023)
12. Majumdar, A.: Graph structured autoencoder. Neural Networks **106**, 271–280 (OCT 2018). https://doi.org/10.1016/j.neunet.2018.07.016
13. Naveed, M., Ali, U., Aziz, T., Rasool, M., Ijaz, A., Alharbi, M., Essa Alharbi, M., Alshammari, A., Alasmar, A.: A reverse vaccinology approach to design an mrna-based vaccine to provoke a robust immune response against hiv-1. Acta Biochimica Polonica **70**(2) (2023). https://doi.org/10.18388/abp.2020_6696
14. Remesh, S., Andreatta, M., Ying, G., Kaever, T., Nielsen, M., McMurtrey, C., Hildebrand, W., Peters, B., Zajonc, D.: Unconventional peptide presentation by major histocompatibility complex (mhc) class i allele hla-a*02:01:breaking confinement. J Biol Chem **292**(13) (2017). https://doi.org/10.1074/jbc.M117.776542
15. Reynisson, B., Alvarez, B., Paul, S., Peters, B., Nielsen, M.: Netmhcpan-4.1 and netmhciipan-4.0: improved predictions of mhc antigen presentation by concurrent motif deconvolution and integration of ms mhc eluted ligand data. Nucleic Acids Research **48**(W1), W449–W454 (2020). https://doi.org/10.1093/nar/gkaa379
16. Rica, E., Álvarez, S., Serratosa, F.: On-line learning the edit costs based on an embedded model. In: International Workshop on Graph-Based Representations in Pattern Recognition. pp. 121–130. Springer (2019)
17. Rica, E., Álvarez, S., Serratosa, F.: Ligand-based virtual screening based on the graph edit distance. Int. J. Mol. Sci. **22**(23), 12751 (2021)
18. Santacruz, P., Serratosa, F.: Learning the graph edit costs based on a learning model applied to sub-optimal graph matching. Neural Processing Letters pp. 1–24 (2019)
19. Schymkowitz, J., Borg, J., Stricher, F., Nys, R., Rousseau, F., Serrano, L.: The foldx web server: an online force field. Nucleic acids research **33**(Web Server issue) (2005). https://doi.org/10.1093/nar/gki387
20. Serratosa, F.: Fast computation of bipartite graph matching. Pattern Recogn. Lett. **45**, 244–250 (2014)
21. Serratosa, F.: Speeding up fast bipartite graph matching through a new cost matrix. International Journal of Pattern Recognition and Artificial Intelligence **29**, 1550010 (11 2014). https://doi.org/10.1142/S021800141550010X
22. Serratosa, F.: Graph edit distance: Restrictions to be a metric. Pattern Recognition **90**, 250–256 (2019). https://doi.org/10.1016/j.patcog.2019.01.043, https://doi.org/10.1016/j.patcog.2019.01.043
23. Serratosa, F.: A general model to define the substitution, insertion and deletion graph edit costs based on an embedded space. Pattern Recognition Letters **138**, 115–122 (2020). https://doi.org/10.1016/j.patrec.2020.07.010, https://www.sciencedirect.com/science/article/pii/S0167865520302567
24. Serratosa, F.: Redefining the graph edit distance. SN Computer Science **2** (11 2021). https://doi.org/10.1007/s42979-021-00792-5

25. Serratosa, F.: Atena: A web-based tool for modelling metal oxide nanoparticles based on nanofingerprint quantitative structure-activity relationships. Molecules **29**(10), 2235 (2024)
26. Serratosa, F., Cortés, X.: Graph edit distance: Moving from global to local structure to solve the graph-matching problem. Pattern Recogn. Lett. **65**, 204–210 (2015)
27. Vita, R., Mahajan, S., Overton, J.A., Dhanda, S.K., Martini, S., Cantrell, J.R., Wheeler, D.K., Sette, A.: The immune epitope database (iedb): 2018 update. Nucleic acids research **47**(D1) (2018). https://doi.org/10.1093/nar/gky1006
28. Waikhom, L., Patgiri, R.: A survey of graph neural networks in various learning paradigms: methods, applications, and challenges. Artif. Intell. Rev. **56**(7), 6295–6364 (2023)
29. Wang, J., Liang, J., Yao, K., Liang, J., Wang, D.: Graph convolutional autoencoders with co-learning of graph structure and node attributes. Pattern Recognit. **121**, 108215 (2022)
30. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net (2019)
31. Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., Liu, T.: Do transformers really perform badly for graph representation? In: Advances in Neural Information Processing NeurIPS 2021. pp. 28877–28888 (2021)
32. Yun, S., Jeong, M., Yoo, S., Lee, S., Yi, S.S., Kim, R., Kang, J., Kim, H.J.: Graph transformer networks: Learning meta-path graphs to improve gnns. Neural Netw. **153**, 104–119 (2022)

Dissimilarity-Based Graph Embedding: An Efficient GAT-based Approach

Francesco Leonardi(✉) and Kaspar Riesen

Institute of Computer Science, University of Bern, Neubrückstrasse 10, 3012 Bern, Switzerland
{francesco.leonardi,kaspar.riesen}@unibe.ch

Abstract. Graph embedding into vector spaces is a widely used practice in graph-based pattern recognition to bypass the mathematical limitations of the graph domain. Several methods for graph embedding have been proposed in the last decades. Many of them are based on graph kernels which produce implicit embeddings that are sometimes hard to interpret. There are also explicit graph embeddings available that produce more interpretable results. However, these methods are often computationally expensive. In the present paper, we propose an approach that combines *Graph Attention Networks* (GAT) and *Dissimilarity Based Graph Embedding* (DBGE) for the first time. The major goal of our novel method is to produce DBGEs with much smaller computation time than with the original method. To achieve this goal, we train a GAT to learn embeddings from a dissimilarity matrix previously calculated on training data. In an empirical evaluation on four graphs datasets, we observe a considerable time reduction compared to the original embedding technique without any noticeable deterioration in classification performance.

Keywords: Graph Embedding · Graph Edit Distance · Graph Neural Network

1 Introduction and Related Work

The ability to recognize pattern is innate in human beings as we are able, for instance, to recognize the face of a familiar person in a crowd, identify language patterns, or discern recurring behaviour, to name just three examples. In computer science, pattern recognition is the area of research that aims to equip machines with this capacity in order to perform tasks such as image classification [1], fraud detection [2], or social network analysis [3].

Two main approaches are present in the field of pattern recognition, *statistical pattern recognition*, which draws inspiration from probability and statistics, and *structural pattern recognition*, which focuses on the intrinsic structure of data. There is a general consensus that the structural approach is particularly useful when the relationships between entities are crucial for understanding the phenomena under investigation. In structural pattern recognition, often graphs

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 361–374, 2025.
https://doi.org/10.1007/978-3-031-78192-6_24

are employed for formally representing the patterns. In the present paper, we define a graph g as a quadruple $g = (V, E, \mu, \nu)$, where V represents a finite set of nodes, $E \subseteq V \times V$ denotes the set of edges, $\mu : V \rightarrow L_V$ is the node labelling function, and $\nu : E \rightarrow L_E$ is the edge labelling function (where L_V and L_E represent node and edge label alphabets, respectively). In recent years, graphs have been employed in diverse pattern recognition applications (e.g., in recommender systems [4] or in object recognition in 3D scenes [5]).

Research in the field of structural pattern recognition can be subdivided into three areas. A first area is characterized by the concept of *graph matching*, with the overall aim to find correspondences between nodes and edges of two graphs [6]. In this context, we find well-known algorithms such as *Graph Edit Distance* (GED) [7], which allows to compute the dissimilarity between graphs by determining the minimum cost sequence of edit operations required to transform one graph into another. Other well-known approaches for graph matching are based on *spectral methods* [8] or *expectation maximization algorithms* [9].

A second area of structural pattern recognition is characterized by the use of *Graph Neural Networks* (GNNs) [10]. GNNs exploit the concept of message passing by simulating the exchange of information between neighboring nodes. This characteristic allows GNNs to overcome the limitation of static graph representations. In the last years, various GNN architectures, such as *Graph Convolutional Networks* (GCNs) [11] or *Graph Isomorphism Networks* (GINs) [12], have proven to be very effective in performing various tasks, including, for instance, the prediction of molecular properties [13] or community detection in social networks [3].

A third area in structural pattern recognition is based on research on *graph kernels*. This approach extends the kernel concept, originally developed for vector representations, in a natural way to graphs, paving the way for the application of machine learning algorithms to graphs (e.g., for classification, clustering or regression). The variety of kernels developed in recent years attests to the vitality of this field, with proposals ranging from kernels based on *walks* [14] and *sub-graphs* [15] over kernels based on *diffusions* [16] to kernels based on *embeddings* [17]. All of these approaches aim to project graphs into a high-dimensional vector space. A possible differentiating feature for graph kernels is whether the resulting graph embedding is explicit or implicit. Implicit means that only pairs of dot products are available in the embedding space, while explicit means that for each graph we receive a corresponding feature vector.

One explicit graph kernel that turns out to be flexible and powerful is *Dissimilarity-Based Graph Embedding* (DBGE) [18]. However, the main limitation of DBGE stems from the high computational cost, as it depends on graph matching. To mitigate this problem, it has been proposed to use fast, yet approximate, graph matching algorithms such as, for instance, *Bipartite Graph Edit Distance* (BP-GED) [19]. In the present paper, we pursue a novel approach to accelerate DBGE. Concretely, we propose a method that combines for the first time a *Graph Attention Network* (GAT) [20] with DBGE. The basic idea of this combination is that we first run a DBGE on each graph in the training set and then train a GAT on these embeddings to learn and approximate the embedding for future input graphs. The motivation for this approach is that we want to significantly reduce the computation time for the graph embedding while maintaining the accuracy of the algorithms, that work on these learned embeddings, as high as possible.

There are, of course, other approaches that use graph neural networks to extract graph embeddings. One of the best known methods is graph2vec [21], which extracts sub-graphs and learns their representation using skip-grams with negative sampling. Other popular methods are based on Siamese networks [22], which compare pairs of input graphs and learn their similarity. Attention-based graph neural network models [23] have demonstrated good performance by creating embeddings that take into account the relationships between nodes. However, most of these approaches are based on unsupervised methods. In contrast, our method, which relies on DBGE, is supervised and thus easier to interpret than other methods.

The remainder of this paper is organised as follows. First, in Section 2, we present in detail both the computation of the DBGE and the neural network model actually employed. Then, in Section 3 we present a thorough empirical evaluation and details on all parameters used, ensuring reproducibility. Ablation studies are also conducted to understand how different modifications to our method lead to different embeddings. Finally, in Section 4, we conclude our paper and discuss future research activities.

2 Learning Dissimilarity-Based Graph Embeddings

We propose a novel method that learns graph embeddings in vector spaces. Our approach is based on a graph neural network, viz. a *Graph Attention Network* (GAT) [20], that maps the graphs into a space structured as a dissimilarity vector with respect to the graphs in a training set. In other words, our goal is to simulate the *Dissimilarity-Based Graph Embedding* (DBGE) [18] using a graph neural network.

Figure 1 shows the two main components of our novel framework. The first component is responsible to compute DBGEs for both training purposes and ground truth embeddings. The second component consists of a GAT that learns and simulates the embeddings. In the following two subsections we describe these two components, in greater detail.

2.1 Computation of Ground-Truth Embeddings

Our ground truth embedding is defined by the DBGE [18] which allows graphs to be represented as numerical vectors. The basic idea of DBGE is as follows. Let us assume a training set $T = \{g_1, g_2, ..., g_N\}$ with N graphs from an arbitrary graph domain G, some graph prototypes $P = \{p_1, p_2, ..., p_n\} \subseteq T$, as well as an arbitrary dissimilarity measure $d : G \times G \to \mathbb{R}$ are given. Then, the DBGE is computed by the mapping $\varphi : G \to \mathbb{R}^n$ defined as

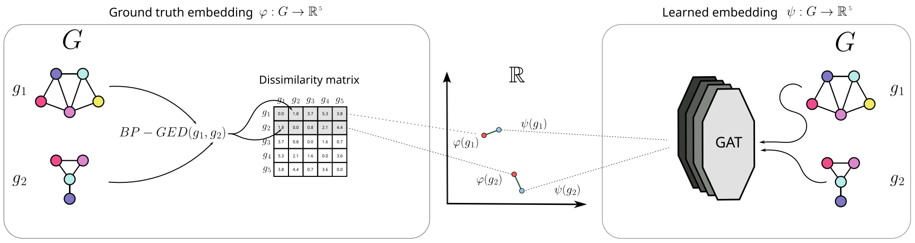


Fig. 1. Computation of the DBGE used as ground truths (see left side of illustration) and then approximating these embeddings using a Graph Attention Network (GAT) (see right side of illustration).

$$\varphi(g) = (d(g, p_1), d(g, p_2), ..., d(g, p_n)),$$

where $d(g, p_i)$ is the graph dissimilarity measured between any graph $g \in G$ and the i-th prototype graph $p_i \in P$.

To apply the DBGE, it is necessary to have a dissimilarity measure for graphs available. In the present paper we employ GED as basic dissimilarity measure (note, however, that any other graph dissimilarity could be used as well). Basically, GED evaluates the minimum number of edit operations required to transform one graph into the other. Optimal algorithms for computing GED have exponential time complexity, making them computationally challenging for large graphs (or even making their use impossible). For this reason, we employ an approximation for the computation of GED, viz. the Bipartite Graph Edit Distance (BP-GED) [19]. BP-GED reduces the problem of GED computation to a linear sum assignment problem, which is in turn solvable with cubic time complexity.

In our approach for DBGE, the initial dataset $D = \{g_1, ..., g_N\}$ is split into two disjoint parts D_{train} and D_{test} with n and m graphs, respectively. We use D_{train} to build a ground-truth embedding and train our model. That is, for each graph $g_i^{train} \in D_{train}$ with $i \in [1, n]$, we calculate the dissimilarity value between g_i^{train} and all other graphs from D_{train} to obtain the following $n \times n$ dissimilarity matrix

$$\mathbf{D}_{\text{train}} = \begin{array}{c} \\ g_1^{train} \\ \vdots \\ g_i^{train} \\ \vdots \\ g_n^{train} \end{array} \begin{array}{c} \begin{array}{ccccc} g_1^{train} & \cdots & g_j^{train} & \cdots & g_n^{train} \end{array} \\ \begin{pmatrix} d_{11} & \cdots & \cdots & \cdots & d_{1n} \\ \vdots & \ddots & & & \vdots \\ \vdots & & d_{ij} & & \vdots \\ \vdots & & & \ddots & \vdots \\ d_{n1} & \cdots & \cdots & \cdots & d_{nn} \end{pmatrix} \end{array} \tag{1}$$

where $d_{ij} = \text{BP-GED}(g_i^{train}, g_j^{train})$.

The graphs from D_{test} are used to assess the embedding quality. Hence, also for all graphs $g_i^{test} \in D_{test}$ with $i \in [1, m]$, we calculate the dissimilarity value between g_i^{test} and all graphs $g_j^{train} \in D_{train}$ to obtain the following $m \times n$ dissimilarity matrix

$$\mathbf{D}_{\text{test}} = \begin{array}{c} \\ g_1^{test} \\ \vdots \\ g_m^{test} \end{array} \begin{array}{c} \begin{array}{ccccc} g_1^{train} & \dots & g_j^{train} & \dots & g_n^{train} \end{array} \\ \begin{pmatrix} d'_{11} & \dots & \dots & \dots & d'_{1n} \\ \vdots & \ddots & & & \vdots \\ d'_{m1} & \dots & \dots & \dots & d'_{mn} \end{pmatrix} \end{array} \tag{2}$$

where $d'_{ij} = \text{BP-GED}(g_i^{test}, g_j^{train})$. Note that in contrast to the original approach [18], we do not perform prototype selection for the present study. This means that we use all available training graphs as a reference point for DBGE. From now on, we refer to this ground truth and reference embedding $\varphi : G \rightarrow \mathbb{R}^n$ as *GED-Embedding* (see left side of Figure 1).

2.2 Architecture of the Graph Attention Network (GAT)

Various GNN architectures have been proposed in the literature, each characterized by a unique approach to aggregate and update the node representations. For instance, *Graph Convolutional Networks* (GCN) [11] apply convolutions to the graph nodes, facilitating the efficient diffusion of information through the graph structure. On the other hand, *GraphSAGE* [24] adopts a sampling and aggregation strategy to ensure network scalability to sizeable graphs. *Graph Attention Networks* (GAT) [20] introduce attention mechanisms to differentially weight information during aggregation, particularly relevant when some relations in the graph are more significant than others.

This last feature of GATs is the reason to adopt it in our framework[1] and we now provide a detailed examination of the GAT architecture used in our work (outlined in Figure 2).

1. The input to our GAT is a labeled graph $g \in G$ that will be embedded.
2. Three levels of Graph Attention Layers (GATConv) [20] are implemented to increase the dimensionality of the node embedding. This phase allows the network to gain an understanding of the relationships between the nodes in the graph and their labels.
3. Three generic pooling operations are employed [25], viz. global sum, global maximum and global mean, used to represent the graph as the sum, maximum and average of the labels of its nodes, respectively.
4. The graph representations pass through fully connected layers, producing a vector called *bottleneck.*
5. Bottleneck is the latent representation of the graph by combining the different generic pooling operations passed through the linear layers.
6. The bottleneck serves as the starting point for the subsequent expansion phase. Four fully connected layers, applied in series, gradually increase the

[1] Our model is implemented using PyTorch 2.2.0.

dimension of the latent space until reaching the output unit count, which corresponds in our scenario to the dimension of the DBGE (i.e., $|D_{train}| = n$). A Dropout layer is introduced to regulate the learning process and mitigate the risk of over-fitting (i.e., contributing to greater generalization of the model). We apply a *Rectified Linear Unit* (ReLU) as nonlinearity function to each layer of the expansion phase, which allows modelling complex relationships between nodes and further generalizing the model.

7. The main objective of the network is to predict the DBGE for each input graph. To achieve this goal, a loss function $\mathcal{L}$ is used that combines two terms, *Mean Squared Error* (*MSE*) and *Kullback-Leibler Divergence* (*KLD*). The first term, MSE, is used to minimize the mean square error between the values predicted by the network and the actual values of the DBGE. The second term, KLD, is useful for considering the overall distribution of the graphs, favoring a less precise representation of the specific value. Formally, in our framework the loss $\mathcal{L}$ is defined by

$$\mathcal{L} = \lambda \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y_i})^2 + (1 - \lambda) \sum_{i=1}^{N} p_i \log \frac{p_i}{q_i}, \tag{3}$$

where
   - N is the number of graphs that are embedded.
   - y_i is the GED-Embedding of graph g_i.
   - $\hat{y_i}$ is the predicted embedding of graph g_i.
   - p_i is the probability distribution of the predicted embedding.
   - q_i is the probability distribution of the GED-Embedding.
   - λ is a free parameter ($0 \leq \lambda \leq 1$) which allows to weight how much the two terms (MSE and KLD) affect the value of $\mathcal{L}$, i.e. whether to focus more on the values of the DBGE or more on the overall distribution.

   Graph embeddings $\psi : G \to \mathbb{R}^n$ that are produced using this process are termed *GAT-Embedding* for the rest of the paper (see right side of Figure 1).

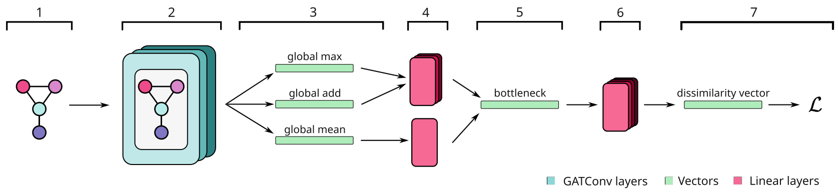


Fig. 2. The GNN architecture to learn the DBGE.

3 Experimental Evaluation

3.1 Experimental Setup and Datasets

To assess the effectiveness of the proposed embedding framework, we focus on evaluating three key metrics, viz. the classification accuracy obtained on the true and learned embeddings, the mean square error between the two embeddings and the computation time needed to embed the graphs using the reference method (GED-Embedding) and our novel approach (GAT-Embedding).

In our evaluation, we use four publicly available datasets [26]:

- DHFR contains 467 graphs representing inhibitors of dihydrofolate reductas from two classes.
- MUTAG comprises 188 graphs representing chemical compounds which are divided into two classes that refer to their mutagenicity on Salmonella typhimurium.
- PROTEINS contains 1,113 graphs representing proteins classified as enzymes or non-enzymes.
- MUTAGENICITY contains 4,337 graphs representing mutagen or nonmutagen molecules.

Each dataset is split into a training and test set with a ratio of 9:1. Next, the GAT is trained on the training data to learn the GED-Embedding. Finally, the GAT inference is performed on the graphs in the test set. For measuring the classification accuracy, we classify both the GED-Embeddings and GAT-Embeddings with three standard classifiers, namely SVM, K-Nearest Neighbors, and Random Forest. These models are optimized on the embeddings of the training sets (for each dataset individually). Note that both the GED-Embeddings and GAT-Embeddings are standardized using the mean and standard deviation of the respective features obtained from embeddings on the training set.

3.2 Validation of Hyperparameters

The neural network to predict the embedding is trained for 500 epochs in total using the Adam optimizer with a learning rate of 0.0001. A batch size of 8 is used for the MUTAG and DHFR dataset, while for the PROTEINS and MUTAGENICITY datasets, batch sizes of 32 are used. Parameter λ, which trades between MSE and KLD, is tested with values $\lambda \in \{0.10, 0.25, 0.50, 0.75, 0.90\}$.

The parameters of the classification algorithms (SVM, K-Nearest Neighbors and Random Forest) are optimized using GridCV with five-folds for each training set. The following configurations are evaluated.

- **SVM:** C $\in \{0.1, 1, 10, 100\}$ using an RBF kernel with $\gamma \in \{0.01, 0.1, 1, auto\}$
- **K-Nearest Neighbors:** $k \in \{3, 5, 7, 10\}$
- **Random Forest:** Number of estimators $\in \{50, 100\}$, max depth $\in \{10, 20, 30\}$, min samples split $\in \{2, 3, 5\}$

3.3 Similarity between GED-Embedding and GAT-Embedding

By means of this first evaluation, we measure how well the proposed GAT-Embedding approximates the true GED-Embedding. For this purpose, we measure the *Root Mean Squared Percent Error* (RMSPE) between the two embeddings on the test sets of all datasets (see results in Table 1).

Table 1. *Root Mean Squared Percent Error* (RMSPE) between GED-Embedding and GAT-Embedding.

		Dataset				
Type	Layers	DHFR	MUTAG	PROTEINS	MUTAGENICITY	Mean
	1	107%	3%	58%	71%	60%
GATConv	2	177%	7%	34%	41%	65%
	3	89%	3%	23%	27%	**36%**
	1	173%	8%	47%	65%	73%
GCNConv	2	122%	4%	58%	61%	61%
	3	116%	4%	34%	39%	48%

For a more complete study, we conduct an ablation study, modifying the architecture of the underlying GNN to observe how the approximations change. Specifically, we modify the initial part of the GNN, which has the goal of extracting information from the nodes of the graph. We choose to consider two types of layers for message passing, GATConv [20] and GCNConv [27]. For our evaluation we evaluate the number of layers in series at the beginning of the GNN to amplify the effect of message passing.

As we increase the complexity of the model by adding layers, the RMSPE is reduced in general (for both GATConv and GCNConv). Comparing GATConv and GCNConv, we observe that GATConv often shows higher RMSPE values than GCNConv. However, when reaching three GATConv layers in series, the RMSPE decreases significantly, being about thirteen percentage points lower than that obtained with three GCNConv layers in series. That is, we observe that the overall best results are achieved with the architecture GATConv using 3 layers with an average *RMSPE* of 36% measured over all datasets. Therefore, we choose to use this architecture for the remaining experiments.

The aim of the next evaluation is to gain insight how the GAT-Embeddings vary with different values for λ that weights the two loss criteria, MSE and KLD (see results in Table 2). It can be seen that high values for λ tend to achieve lower values for the RMSPE in general. This means that the first term (MSE) and not necessarily the second term (KLD) seems to be more important for our optimization. We choose to fix $\lambda = 0.85$ for the remainder of our evaluation.

For a visual and qualitative assessment, we apply a *Principal Component Analysis* (PCA) [28] to both GED-Embeddings and GAT-Embeddings (in order

Table 2. The RMSPE values with the 3-GATConv architecture using different values for parameter λ. The best values achieved for each dataset are shown in bold face.

	Dataset				
λ	DHFR	MUTAG	PROTEINS	MUTAGENICITY	Mean
0.10	414%	8%	49%	60%	133%
0.25	341%	7%	43%	23%	104%
0.50	213%	2%	20%	**20%**	64%
0.75	130%	**2%**	**18%**	29%	45%
0.90	**82%**	3%	24%	28%	**34%**

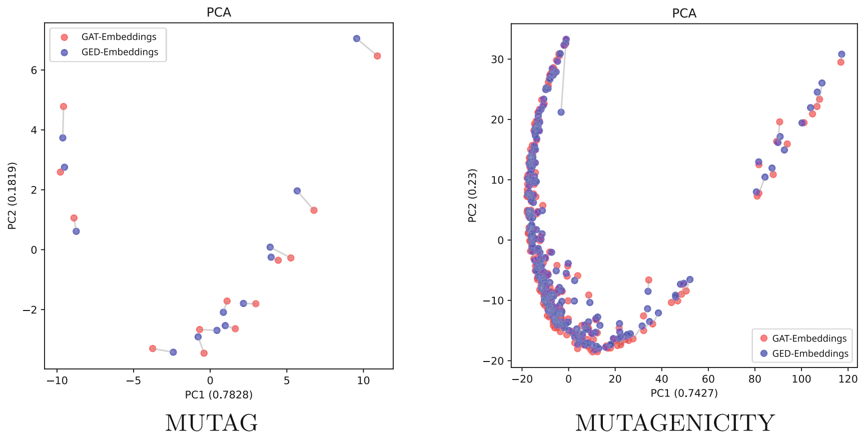


Fig. 3. Projection of GED-Embeddings (blue dots) and GAT-Embeddings (red dots) via PCA into a visualizable, shared latent space. The visual representation includes visualisations of the distances between corresponding vector pairs (using a gray line)(Color figure online).

to reduce the dimension of the embeddings and visualize them). Specifically, PCA is learned from the GED-Embeddings of the test set, and then the trained PCA model is used to project the GAT-Embeddings in the same latent space, allowing a visual comparison of the results. Figure 3 shows the PCA evaluation on the MUTAG and MUTAGENICITY datasets (similar plots are achieved on the other datasets).

It is clearly visible that the predicted and the true embeddings are close to each other, reinforcing the hypothesis that GAT-Embeddings can simulate the GED-Embeddings with quite high precision.

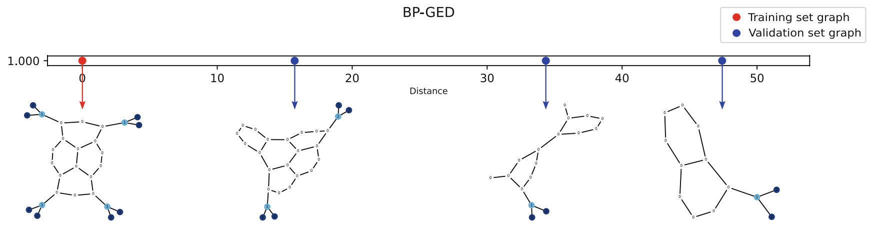


Fig. 4. The upper part of the figure shows the distance values predicted by the GAT-Embedding along the first dimension of the dissimilarity embedding. The red point represents a graph embedding from the training set, while the blue points represent graph embeddings from the test set. The lower part of the figure shows the graphs corresponding to the points.

By having a structured embedding space, in which each dimension corresponds to the dissimilarity to a graph in the training set, we can also visualize the distance of some input graphs to a specific graph in the learned embedding space. This aspect gives our method a sort of interpretability.

Figure 4 shows an example visualization of graphs from the MUTAG dataset. We show the first dimension of the GAT-Embedding, displaying the first graph of the training set and the distances of three graphs of the test set. Note that these distances correspond to the predicted distances using our GAT-Embedding. The resulting distances to the training graph are obviously reasonable, since the similarity of the associated graphs actually decreases with increasing distance to the training graph.

3.4 Classification Task

To evaluate the applicability of the learned GAT-Embeddings, a graph classification task is conducted on the four datasets using the three aforementioned classifiers. To ensure that the results are not influenced by random factors, five iterations are performed for each classifier and dataset. Table 3 reports the mean and standard deviation of the classification accuracy obtained by the three classifiers.

We observe that the original GED-Embeddings tend to achieve better results than the learned GAT-Embeddings (regardless the classification algorithm and dataset). At the same time, however, we can report that on the MUTAG and MUTAGENICITY datasets (and partly also on PROTEINS) quite similar accuracies are achieved by both embeddings. Considering the difficulty of the underlying classification problems, the results obtained are in any case quite promising. This is especially true when we compare the computation times actually needed for carrying out the embeddings (evaluated in the next section).

Table 3. Mean and standard deviation of the classification accuracy achieved by the classification algorithms on the GED-Embeddings and GAT-Embeddings. The best result per dataset and classifier is shown in bold face.

SVM

Dataset	GED-Embeddings	GAT-Embeddings
DHFR	**0.78 (± 1.54)**	0.69 (± 4.62)
MUTAG	**0.85 (± 0.14)**	0.83 (± 0.12)
PROTEINS	**0.80 (± 0.04)**	0.72 (± 0.02)
MUTAGENICITY	**0.75 (± 0.06)**	0.68 (± 0.08)

K-Nearest Neighbors

Dataset	GED-Embeddings	GAT-Embeddings
DHFR	**0.78 (± 2.04)**	0.66 (± 3.86)
MUTAG	0.78 (± 0.13)	**0.83 (± 0.13)**
PROTEINS	**0.80 (± 0.01)**	0.73 (± 0.03)
MUTAGENICITY	**0.71 (± 0.06)**	0.68 (± 0.05)

Random Forest

Dataset	GED-Embeddings	GAT-Embeddings
DHFR	**0.74 (± 1.54)**	0.70 (± 1.33)
MUTAG	0.76 (± 0.09)	**0.83 (± 0.16)**
PROTEINS	**0.79 (± 0.01)**	0.73 (± 0.02)
MUTAGENICITY	**0.76 (± 0.03)**	0.70 (± 0.04)

3.5 Computation Time

Last but not least, Table 4 shows the computational cost (in milliseconds) associated with the calculation of the GED-Embeddings and GAT-Embeddings. For this experiment, 100 graphs are embedded for each dataset and we repeat the embedding 1000 times and show the average computation time.

For the GED-Embedding we use an optimized implementation of *BP-GED* [29] using 8 CPUs. The GAT-Embedding is performed on both the CPU and GPU[2]. In addition, the calculated ratio of the time taken by the GED-Embedding versus the time taken by the GPU based GAT-Embedding is shown in Table 4. The results highlight that the proposed GAT-Embedding confers a significant computational advantage over the optimized GED-Embedding on all tested datasets. That is, our novel approach is about 6 to about 3,500 times faster than the time consuming GED-Embedding.

[2] The CPU used is AMD EPYC 7742. The GPU used is a GeForce RTX 4070 Ti.

Table 4. Computational cost for the calculation of 100 DBGE (in milliseconds)

Dataset	GED-Embeddings (8 CPUs)	GAT-Embeddings (CPU)	GAT-Embeddings (GPU)	Ratio
DHFR	2,799	6.34	2.92	959
MUTAG	10.19	2.17	1.66	6
PROTEINS	3,731	7.95	3.52	1,060
MUTAGENICITY	11,397	3.24	3.21	3,550

4 Conclusion

The basic idea of *Dissimilarity Based Graph Embedding* (DBGE) is to interpret the distances to graph prototypes as features of a graph to be embedded. This type of graph embedding is known to be powerful and flexible, but suffers from the fact that the runtime is largely dependent on graph matching. In this paper, we introduce a novel graph embedding method that learns DBGE by means of a *Graph Attention Network* (GAT). The motivation for combining a GAT with DBGE is that we want to maintain the strong embeddings of DBGE but with much faster computations that operate independently of a graph matching procedure. In an experimental evaluation, we show that our novel method significantly reduces the computation time for DBGEs, making it thousands of times faster than the reference method. This massive speed-up makes DBGE usable in real-time applications for the first time. Furthermore, the analysis of the classification accuracy shows that the DBGE predicted by the GAT can achieve similar performance to those calculated using the original (time consuming) embeddings.

For future work, we see several rewarding avenues that can be pursued. First, it might be useful to implement advanced prototype selections to include more informative and representative graphs for DBGE (currently, no selection of prototypes takes place). These selection strategies could be extended to more challenging targets and projects, e.g., by training networks to produce embeddings that are interoperably applicable on multiple datasets simultaneously. Another interesting line of research could be to replicate our approach with other types of embeddings (e.g., Weisfeiler-Lehman embeddings [30] or others).

Acknowledgements. This study is funded by the Swiss National Science Foundation (SNSF) Project Nr 200020_219388.

The code is available at the link: https://github.com/MonsieurSolver/DBGE-GAT

References

1. Zaïd Harchaoui and Francis R. Bach. Image classification with segmentation graph kernels. In *2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007), 18-23 June 2007, Minneapolis, Minnesota, USA*. IEEE Computer Society, 2007

2. Shehnepoor, S., Togneri, R., Liu, W., Bennamoun, M.: HIN-RNN: A graph representation learning neural network for fraudster group detection with no handcrafted features. IEEE Trans. Neural Networks Learn. Syst. **34**(8), 4153–4166 (2023)
3. Guo, Z., Wang, H.: A deep graph neural network-based mechanism for social recommendations. IEEE Trans. Ind. Informatics **17**(4), 2776–2783 (2021)
4. Shakila Shaikh, Sheetal Rathi, and Prachi Janrao. Recommendation system in e-commerce websites: A graph based approached. In *2017 IEEE 7th International Advance Computing Conference (IACC)*, pages 931–934, 2017
5. Johanna Wald, Helisa Dhamo, Nassir Navab, and Federico Tombari. Learning 3d semantic scene graphs from 3d indoor reconstructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020
6. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. Int. J. Pattern Recognit Artif Intell. **18**(3), 265–298 (2004)
7. Sanfeliu, A., King-Sun, F.: A distance measure between attributed relational graphs for pattern recognition. IEEE Trans. Syst. Man Cybern. **13**(3), 353–362 (1983)
8. Mariá Cristina Vasconcelos Nascimento and André Carlos Ponce de Leon Ferreira de Carvalho. Spectral methods for graph clustering - A survey. *Eur. J. Oper. Res.*, 211(2):221–231, 2011.
9. Luo, B., Hancock, E.R.: Structural graph matching using the EM algorithm and singular value decomposition. IEEE Trans. Pattern Anal. Mach. Intell. **23**(10), 1120–1136 (2001)
10. Zhou, J., Cui, G., Shengding, H., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M.: Graph neural networks: A review of methods and applications. AI Open **1**, 57–81 (2020)
11. Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017
12. Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019
13. Zeren Shui and George Karypis. Heterogeneous molecular graph neural networks for predicting molecule properties. In Claudia Plant, Haixun Wang, Alfredo Cuzzocrea, Carlo Zaniolo, and Xindong Wu, editors, *20th IEEE International Conference on Data Mining, ICDM 2020, Sorrento, Italy, November 17-20, 2020*, pages 492–500. IEEE, 2020
14. Karsten M. Borgwardt, Cheng Soon Ong, Stefan Schönauer, S. V. N. Vishwanathan, Alexander J. Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. In *Proceedings Thirteenth International Conference on Intelligent Systems for Molecular Biology 2005, Detroit, MI, USA, 25-29 June 2005*, pages 47–56, 2005
15. Karsten M. Borgwardt, Tobias Petri, S. V. N. Vishwanathan, and Hans-Peter Kriegel. An efficient sampling scheme for comparison of large graphs. In Paolo Frasconi, Kristian Kersting, and Koji Tsuda, editors, *Mining and Learning with Graphs, MLG 2007, Firence, Italy, August 1-3, 2007, Proceedings*, 2007
16. Risi Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In Claude Sammut and Achim G. Hoffmann, editors, *Machine Learn-*

ing, Proceedings of the Nineteenth International Conference (ICML 2002), University of New South Wales, Sydney, Australia, July 8-12, 2002, pages 315–322. Morgan Kaufmann, 2002

17. Bahonar, H., Mirzaei, A., Sadri, S., Wilson, R.C.: Graph embedding using frequency filtering. IEEE Trans. Pattern Anal. Mach. Intell. **43**(2), 473–484 (2021)
18. Kaspar Riesen and Horst Bunke. *Graph Classification and Clustering Based on Vector Space Embedding*, volume 77 of *Series in Machine Perception and Artificial Intelligence*. WorldScientific, 2010
19. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. Image Vis. Comput. **27**(7), 950–959 (2009)
20. Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *CoRR*, abs/1710.10903, 2017
21. Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. *CoRR*, abs/1707.05005, 2017
22. Sofia Ira Ktena, Sarah Parisot, Enzo Ferrante, Martin Rajchl, Matthew C. H. Lee, Ben Glocker, and Daniel Rueckert. Metric learning with spectral graph convolutions on brain connectivity networks. *NeuroImage*, 169:431–442, 2018
23. Rami Al-Rfou, Bryan Perozzi, and Dustin Zelle. DDGK: learning graph representations for deep divergence graph kernels. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 37–48, 2019
24. William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1024–1034, 2017
25. Yingxin Wu, Xiang Wang, An Zhang, Xiangnan He, and Tat-Seng Chua. Discovering invariant rationales for graph neural networks. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022
26. Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. *CoRR*, abs/2007.08663, 2020
27. Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016
28. Aamir Khan and Hasan Farooq. Principal component analysis-linear discriminant analysis feature extractor for pattern recognition. *CoRR*, abs/1204.1177, 2012
29. Anthony Gillioz and Kaspar Riesen. Improving graph classification by means of linear combinations of reduced graphs. In Maria De Marsico, Gabriella Sanniti di Baja, and Ana L. N. Fred, editors, *Proceedings of the 11th International Conference on Pattern Recognition Applications and Methods, ICPRAM 2022, Online Streaming, February 3-5, 2022*, pages 17–23. SCITEPRESS, 2022
30. Till Hendrik Schulz, Tamás Horváth, Pascal Welke, and Stefan Wrobel. A generalized weisfeiler-lehman graph kernel. *Mach. Learn.*, 111(7):2601–2629, 2022

Quantifying Racial Segregation Through Continuous-Time Quantum Walks

Yutong Jiang[1], Xing Wu[2], and Jianjia Wang[3](✉)

[1] School of Computer Engineering and Science, Shanghai University, Shanghai, China
jiangyutong@shu.edu.cn

[2] Engineering and Science Shanghai Institute for Advanced Communication and Data Science, School of Computer, Shanghai University, Shanghai, China
xingwu@shu.edu.cn

[3] School of AI and Advanced Computing, XJTLU Entrepreneur College(Taicang), Xi'an Jiaotong-Liverpool University, Suzhou 215123, China
Jianjia.Wang@xjtlu.edu.cn

Abstract. With the development of cities, racial segregation is one of the reasons for social inequality on a large scale, it is also a major factor affecting economic development. However, racial segregation has received comparatively limited attention in research. In the paper, we propose a segregation index independent of any parameters(titled Quantum Walk Convergence Time), which calculates the convergence time statistics for accessing different racial categories through quantum walks on complex networks constructed from urban systems. The magnitude of the time statistics represents the degree of racial segregation. The results generated by our method can also be applied to other social factors, including family situation, income level, and education level. We evaluate our method using large-scale real datasets, including statistics from the U.S. Census Bureau and the Office for National Statistics in the UK. The results demonstrate the close correlation between our method and the degree of racial segregation, showing advantages over traditional random walk methods.

Keywords: Quantum walk · Racial segregation · Convergence time · Complex social systems

1 Introduction

Spatial heterogeneity is an important characteristic of complex social systems, and quantifying spatial segregation is a crucial topic within the study of spatial heterogeneity. We can measure and assess the distribution and segregation of different groups in urban or geographical spaces using various statistical methods and indicators. The fundamental framework for quantifying spatial segregation begins with subdividing a geographical area into smaller units, often census tracts or neighborhoods. We regard the distribution of different social groups within these census tracts as given and estimate the tendency of units to cluster around their surroundings. In this context, Massey and Denton[1] proposed five dimensions of segregation: Evenness, which measures the degree of uniformity in the distribution of a group across different communities; Exposure, which assesses the potential for contact with other groups; Concentration, indicating the amount of physical space occupied by a group; Centralization, reflecting the degree to which a group resides near the city center; and Clustering, which measures the tendency for individuals of the same group to reside in adjacent communities.

Previous studies have proposed various segregation indices to quantify social segregation, including metrics based on spatial heterogeneity and spatial clustering[2][3]. However, these methods rely on the scale and size of the study objects, overlook large-scale spatial correlations, or depend on temporary parameters. They often ignore the multi-scale characteristics of urban spatial heterogeneity, resulting in limited accuracy when comparing different systems on an equal basis. Therefore, it is crucial to seek a method capable of comparing spatial heterogeneity at multiple scales equally.

In this paper, we aim to quantify racial segregation in complex urban systems by constructing networks of these systems and performing quantum walks on the generated target graphs to compute convergence time statistics. Analyzing the spatial distribution of quantum walk convergence time, which represents the expected number of steps for a quantum walk starting from an initial node to visit certain categories in the system, reveals that Quantum Walk Convergence Time results from diffusion on the graph. The diffusion outcome depends on the structure of the target graph and node attributes. Quantum Walk Convergence Time is associated with the degree of regional segregation at each node, effectively capturing the manifestation of urban area social segregation at different scales, shapes, and specific micro-level features. It quantifies the level of racial segregation in different areas of the city.

Quantum walks simulate random walks using quantum mechanics, but they differ significantly in their walking states. In random walks, the state vector is real-valued, while in quantum walks, it is complex-valued. This property allows interference between different walking paths. In classical cases, doubly stochastic matrices control the walk's evolution. In quantum cases, unitary matrices control the evolution, making quantum walks reversible. Researchers study quantum walks extensively on various graphs[4][5], such as linear chains, cycle graphs, complete graphs, regular graphs, and grid graphs. In quantum walks, particles propagate along multiple paths simultaneously. The superposition property allows quantum walks to traverse the network faster. Moreover, the reversibility of quantum walks enables the system to correct errors by returning to previous states, improving the accuracy and reliability of the analysis. Through experiments, we compare the convergence time statistics of quantum walks and random walks[6]. The comparison demonstrates that quantum walks perform better and describe racial segregation across different spatial scales more accurately. This provides new insights for optimizing urban planning and social policies.

2 Related work

The section explains how the target graph is constructed. It also proves the relationship between Quantum Walk Convergence Time and racial segregation through virtual experiments.

2.1 Constructing the target graph

First of all, we connect the target regions through code name. Then, we map the blocks in the city to nodes in the network, and if two blocks are adjacent, we connect them with an edge, forming a graph $G = (V, E)$. In graph G, each node i has a corresponding feature x_i. In this paper, x_i represents the racial distribution in the census area associated with node i. Specifically, it can be expressed as $x_i = \{x_{i,1}, x_{i,2}, ..., x_{i,c}\}$, where $x_{i,c}$ represents the number of citizens belonging to racial category c residing in that area.

2.2 Relation

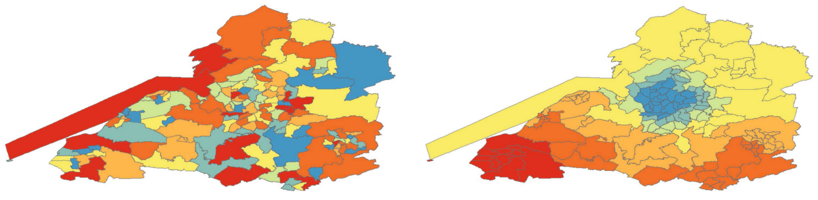
We perform a quantum walk on the target graph $G = (V, E)$, starting from the initial node i_0 with the corresponding quantum state $|\psi(0)\rangle = [0, 0, ..., 1, ..., 0]^T$. Then, we calculate the expected number of steps required for the walker to visit a subset of categories in the graph. This expected number of steps is defined as the Quantum Walk Convergence Time (QWCT). To demonstrate that the statistical data of QWCT can quantify the level of segregation and heterogeneity in a city area, we conduct simulation experiments.

As shown in Fig. 1, we illustrate two extreme fictional racial distribution scenarios in Bristol. The racial distribution in Fig. 1a is randomly uniform, indicating no segregation between specific clusters. If a pedestrian starts from any blue area, they typically encounter a red area after a few steps. In contrast, in Fig. 1b, races in the neighborhoods are artificially organized into distinct spatial clusters, similar to many metropolitan areas. If a walker starts from the large blue cluster in the middle of Fig. 1b, it takes quite a few steps to reach a red area.

This simulation experiment shows that the number of steps needed to access other racial categories after leaving homogeneous clusters on the graph is correlated with the spatial distribution of races. Therefore, we propose a method to quantify the degree of racial segregation in urban areas by conducting quantum walks on the graph. A smaller QWCT value indicates that quantum walks cover nodes of different categories more rapidly in the network, implying that nodes of different categories are relatively easier to access, and segregation is lower.

3 Continuous-time quantum walk

Continuous-time quantum walks, the quantum counterparts of continuous-time random walks, evolve over time between the vertices of a graph. The Schrödinger



(a) No spatial distinction (b) Ethnic cluster distribution

Fig. 1. Virtual distribution of the seven races in the Bristol map, **a** There is no obvious difference in the spatial distribution, and the quantum walk convergence time is small. **b** it forms an obvious cluster distribution, with strong spatial heterogeneity and long quantum convergence time.

equation governs the mathematical structure of quantum walks[7]. Classical random walks model the diffusion process on a graph and prove to be useful tools for analyzing its structure.

In classical random walks, the state vector represents the walker's probability at each vertex of the graph. In quantum systems, the state vector is defined by a vector of complex amplitudes at the graph vertices. The squared norm sums to 1 across all vertices, with no restriction on sign or complex phase. These phase differences produce interference effects. Additionally, in quantum walks, the evolution of the state vector is governed by a complex unitary matrix, while the dynamics of classical random walks are controlled by a stochastic matrix[8].

Farhi and Gutman[9] introduce the concept of continuous-time quantum walks. Unlike discrete-time quantum walks, which require a coin state to decide the walk direction, continuous-time quantum walks do not need such a coin state. In this paper, we apply continuous-time quantum walks rather than discrete-time quantum walks. The former is more suitable for simulating systems with continuous evolution, such as diffusion processes in space.

3.1 Quantum state

Let $G = (V, E)$ be an undirected graph, where V is a set of n vertices, and $E = (V \times V)$ is the set of edges. A quantum state can be represented by a state vector $|\psi\rangle$ in Dirac notation. This state vector contains all the information about the system. It exists in a Hilbert space H spanned by the orthonormal basis states $|i\rangle$, $i \in V$[10]. For example, the state of a two-state system can be represented as:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1}$$

where $\alpha^2 + \beta^2 = 1$, α and β are complex numbers, $|0\rangle$ and $|1\rangle$ are the ground states of the qubit.

3.2 Superposition

Superposition is the combination or addition of two or more basis states to produce another valid state. It describes a quantum system being in a linear combination of multiple states simultaneously. For quantum walks, this means the system can walk multiple paths at the same time. In a graph with N nodes, the quantum state can be represented as a superposition of the basis states of all nodes:

$$|\psi(t)\rangle = \sum_{i=1}^{N} \alpha_i(t)|i\rangle \tag{2}$$

where $|i\rangle$ represents the basis vector corresponding to the i-th node. $\alpha_i(t)$ is the probability amplitude of the i-th node at time t, satisfying the normalization condition:

$$\sum_{i=1}^{N} |\alpha_i(t)|^2 = 1 \tag{3}$$

In quantum walks, a particle jumps in space probabilistically, existing in a superposition of multiple positions. Interference is a phenomenon observed in quantum walks. When a particle moves along different paths, these paths can interfere with each other, leading to the addition or cancellation of probability amplitudes.

3.3 Unitary evolution

In quantum walks, the system's evolution is described by the Schrödinger equation[11]. Specifically, the quantum state of the system changes over time, and this change is determined by the Hamiltonian. In the context of quantum walks, we usually replace the system's Hamiltonian with the graph Laplacian operator L. This replacement allows the evolution of the quantum walk to be described in terms of the graph's structure, where the graph Laplacian L characterizes the connectivity and propagation properties between the nodes of the graph:

$$i\frac{\partial}{\partial t}|\psi(t)\rangle = L|\psi(t)\rangle \tag{4}$$

Among them, the quantum state at time t $|\psi(t)\rangle \in \mathbb{C}^{|\mathbb{V}|}$.

Therefore, the evolution of the quantum walk follows the unitary evolution operator $U(t)$. This implies that the quantum walk is reversible, non-ergodic, and lacks a limiting distribution. The unitary operation outputs probability amplitudes rather than probabilities. For any given initial state, after t unitary operations, the state is given by:

$$|\psi(t)\rangle = e^{-iLt}|\psi(0)\rangle \tag{5}$$

4 Quantum walk on graph

4.1 Quantum walk convergence time

Performing quantum walks on the graph $G = (V, E)$ requires considering two aspects: the particle being in a superposition of quantum states and the selection of the next node. Algorithm 1 illustrates the generation process of Quantum Walk Convergence Time.

Algorithm 1 Quantum Walk Convergence Time

Input: Target graph $G = (V, E)$, graph Laplacian L,start node index *start*, end node index *end*, matrix of node properties *classes*, number of walk iterations *num*, Total number of categories C.

Output: Perform quantum walks on graph G, encountering convergence times $\mathbf{t}$ for different categories.

```
1:  avg_list=[]; {Initialize an empty list}
2:  for $node = start$ to $end$ do
3:      nrc=classes[node]; {Current node properties}
4:      $|\psi(0)\rangle = [0, 0, ..., 1, ..., 0]^T$; {Starting node quantum state}
5:      for $n = 1$ to $num$ do
6:          s=0;
7:          while 0 in $nrc$ do
8:              $U(s) = e^{-iLs}$; {Unitary Evolution Operator $U(s)$}
9:              $|\psi(s)\rangle = e^{-iLs}|\psi(0)\rangle$; {Update quantum state}
10:             $P(u, s) = |\langle u|\psi(s)\rangle|^2$; {Probability of visiting a node u}
11:             $j$ {Select the next node $j$}
12:             $s = s + 1$; {The number of steps to visit each node}
13:             nrc=nrc+classes[j]; {The current and next node's properties}
14:             v = count_nonzero(obs); {The number of categories visited}
15:             return $avg\_list = [v_1, v_2, ..., v_n]$
16:         end while
17:     end for
18:     avg_list=sum(avg_list)/num; {Average categories visited over time}
19:     for $q = 1$ to 100 do
20:         t=0; {Initializing quantum walk convergence time}
21:         $th = (q * C)//100$; {Set threshold}
22:         for $x$ in $avg\_list$ do
23:             t=t+1;
24:             if $x \geqslant th$ then
25:                 break
26:             end if
27:         end for
28:         return t; {Access to different scaled QWCT matrix}
29:     end for
30: end for
```

We describe an algorithm to compute the Quantum Walk Convergence Time. The goal is to calculate the expected number of steps from the initial node to visit a specific proportion of racial categories. Algorithm 1 includes four for loops: the first for loop iterates over initial nodes, the second for loop repeats the walk to calculate averages and reduce errors, the third for loop sets thresholds, and the fourth for loop traverses the list of average visited categories and compares it with the thresholds.

The algorithm's main steps include initialization, quantum walk, and computing the expected convergence time. In step 4, the algorithm creates the initial quantum state based on the node's attribute information. The initial node i_0 corresponds to the qubit in the ground state $|1\rangle$, while the qubits of other nodes are in the ground state $|0\rangle$. During initialization, the algorithm also constructs an empty list to store the number of visited categories, initializes the step count, and sets the node attributes.

In the quantum walk phase, the walker starts from the initial node, performs multiple repetitions, and evolves the quantum state at each time step. In step 9, according to Eq. 5, given the initial state $|\psi(0)\rangle$, the algorithm updates the quantum state superposition to a new state. The evolution process uses the Laplacian matrix. The algorithm traverses each node in the target area and calculates the probability of each node being visited based on the updated quantum state superposition to select the next node.

In the expected convergence time calculation phase, In step 18, we obtain the average number of categories visited starting from different nodes, which increases over time. Then, we set the target category proportion q and compare it with the average number of categories obtained. If it meets or exceeds the threshold, we return the calculated expected convergence time $\mathbf{t}$ as the output.

4.2 Spatial indicators

We find a strong correlation between the QWCT required to access different categories and the degree of racial segregation. In graph G, we initiate the quantum walk from node i_0 and traverse the nodes with steps $t = 0, 1, ..., t$, forming a sequence of visited nodes $i = \{i_0, i_1, ..., i_t\}$. We define the QWCT on node i with category proportion q as the expected value of steps required to encounter a category representing a proportion q of the total categories, starting from node i:

$$Q_i(q) = min\left\{ t | \frac{1}{N} \sum_{n=1}^{N} q_i(t) \geqslant q \right\} \tag{6}$$

Among them, $q_i(t)$ is the proportion of accessing a certain category after spending time t starting from the starting node i_0 and time 0. q is the threshold, and N is the number of quantum walks starting from the starting node.

By averaging Eq. 6, we obtain the corresponding QWCT change trend when accessing different category proportions q from different nodes:

$$\mu(q) = \frac{1}{I}\sum_{i=1}^{I} Q_i(q) \tag{7}$$

For the result obtained from Eq. 6 and the mean value obtained from Eq. 7, we calculate its variance:

$$\sigma(q) = \frac{1}{I}\sum_{i=1}^{I}(Q_i(q) - \mu(q))^2 \tag{8}$$

At the level of local spatial diversity, we measure spatial diversity by calculating the Shannon Diversity Index[12]:

$$H(q) = -\sum_{i=1}^{I} p_i \ln(p_i) \tag{9}$$

Where p_i represents the relative abundance of the $Q_i(q)$ at node i. If the difference in relative abundance of different $Q_i(q)$ increases, the Shannon index will decrease.

In general, a higher value of $\mu(q)$ indicates a more uneven distribution of categories within the system. $\sigma(q)$ measures the variation of QWCT across the entire space. A smaller $H(q)$ value indicates greater disparity in QWCT among neighboring nodes, indicating the presence of local spatial diversity. It's worth noting that these metrics may be influenced to some extent by the characteristics of each node and the size of the graph.

To mitigate these influences, we introduce a null model in this study. The null model is constructed based on the target graph G, where category attributes $\{x_i\}$ are randomly and uniformly assigned to each node. This preserves the edge connectivity and topological structure between nodes while maintaining the overall relative abundance of categories and their distribution within individual regions. However, it disrupts the existing spatial arrangement of categories[13]. We will compute the average deviation of these metrics from the corresponding metrics of the null model to represent spatial distribution:

$$\Delta\mu = \int_0^1 \left|\mu(q) - \mu(q)^{null}\right| dq \tag{10}$$

$$\Delta\sigma = \int_0^1 \left|\sigma(q) - \sigma(q)^{null}\right| dq \tag{11}$$

$$\Delta H = \int_0^1 \left|H(q) - H(q)^{null}\right| dq \tag{12}$$

Where q represents the category proportion, and its value falls within the range of $[0, 1]$.

We refer to $\Delta\mu$,$\Delta\sigma$ and ΔH as spatial heterogeneity, spatial variance and spatial diversity,respectively. $\mu(q)$,$\sigma(q)$, and $H(q)$ deviations from the expected values of the null model constitute a set of fundamental metrics, enabling comparisons across spatial systems with different category counts, shapes, sizes, and features.

4.3 Similarity comparison

To quantify the similarity between the QWCT distributions generated by the original system and the null model through quantum walks, we employ the Jensen-Shannon divergence (JS divergence) to measure the proximity of the model's results to reality[14]. Before delving into that, it's important to understand KL divergence[15], also known as relative entropy, which is a distance measure used to calculate the similarity between two probability distributions $p(x_i)$ and $q(x_i)$:

$$D_{\mathrm{KL}}(p \parallel q) = \sum_{i=1}^{I} p(x_i) \log\left(\frac{p(x_i)}{q(x_i)}\right) \tag{13}$$

Among them, $D_{\mathrm{KL}}(p \parallel q)$ represents the KL divergence of the real system and the null model. The smaller the value, the more similar the estimated probability distribution is to the actual distribution.

The JS divergence also measures the similarity between two probability distributions and shares the property of non-negativity with the KL divergence. However, unlike the KL divergence, its results are symmetric. The formula for calculating the JS divergence is as follows:

$$D_{\mathrm{JS}}(p \parallel q) = \frac{1}{2} D_{\mathrm{KL}}(p \parallel \frac{p+q}{2}) + \frac{1}{2} D_{\mathrm{KL}}(q \parallel \frac{p+q}{2}) \tag{14}$$

The JS divergence ranges from 0 to 1, where a value of 0 indicates identical distributions and a value of 1 indicates completely dissimilar distributions.

We use the JS divergence to compare the similarity between two probability distributions. When the JS divergence approaches zero, it indicates that the two distributions are very similar or overlapping. Conversely, when the JS divergence is larger, it suggests that the distributions differ significantly. Therefore, by calculating the JS divergence, we can quantify the similarity or dissimilarity between the QWCT distributions generated by the original system and the null model through quantum walks.

As shown in Table. 1, we observe that the JS divergence between the real-world spatial system and the null model is close to zero, indicating a small difference in distributions. This occurs because the null model preserves the topological structure of the graph while also maintaining the overall relative abundance and distribution of categories. It only alters the category attributes of each node.

Table 1. The JS divergences between the original systems and null models for the five cities.

	Atlanta	Boston	Cardiff	Dallas	London
$D_{\mathrm{JS}}(p \parallel q)$	0.0096	0.0425	0.0292	0.0053	0.0602

5 Evaluation

In the experiments, we first compare the spatial distribution of convergence time statistics between quantum walks and random walks. Then, we demonstrate the spatial indicators of London, a representative area, to validate the superior performance of quantum walks in quantifying racial segregation.

5.1 Settings

We evaluated our approach using the following datasets:

The Census data from the UK Office for National Statistics, which includes racial distribution information for England and Wales. The data categorizes cities into three levels:LSOAs, OAs, Wards, with a total distribution of 250 racial categories. They are linked by *geo_code* to different regions[16].

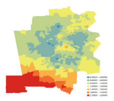
The Census data from the United States Census Bureau, which contains population distribution information for 64 racial categories in different regions. They are linked by $GEOID10$ to different regions[17].

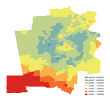
The default settings for numerical simulation in this section are as follows: $q = 0.7$, where q represents a threshold value, and $q * c$ represents the number of categories of interest, where c represents the total number of categories in the area. QWCT on q is defined as the expected number of steps required for the quantum walker to first visit $q * c$ categories starting from the initial node i_0. To avoid errors, we set the number of repeated walks to 1000, denoted as $num = 1000$. The node range is determined by the number of regions divided in the CSV file of the target area. Taking London as an example, the node range is from 0 to 632. It is worth noting that because the UK census divides cities into three levels: LSOAs, OAs, and wards, in this section, experiments related to the UK are based on ward-level divisions. The experimental results in this section are the average of 1000 independent experiments.

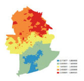
5.2 Comparative experiment

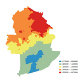
Spatial distribution of QWCT Fig. 2 illustrates the spatial distribution of $\tilde{Q}_i(q) = Q_i(q)/Q_i(q)^{null}$ values generated by quantum walks and traditional random walks for four cities. We observe that blue and green areas represent regions with lower $\tilde{Q}_i(q)$ values, indicating easier access to all ethnicities. Typical regions include the southern area of Cardiff and the central area of Dallas. In reality, Cardiff is a multicultural city attracting populations from various racial and cultural backgrounds, resulting in relatively lower levels of racial segregation compared to some other major cities. Yellow and red areas, on the other hand, represent regions with higher $\tilde{Q}_i(q)$ values. Typical regions include the southwestern area of Atlanta and the northeastern area of Boston. The $\tilde{Q}_i(q)$ values in the northeastern area of Boston are three times higher than those predicted by the null model, indicating a high level of racial segregation.

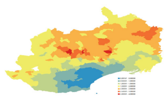
Nevertheless, Fig. 2 shows only partial regional segregation levels. It does not provide a clear comparison of the accuracy of quantum walks and random walks in quantifying segregation levels. Therefore, we analyze three spatial indicators proposed in this paper: $\mu(q)$, spatial variance $\sigma(q)$ and spatial diversity $H(q)$. These indicators allow us to directly compare the performance of quantum walks and random walks.

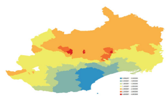

(a) Quantum walk on Atlanta

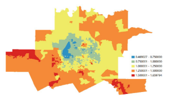

(b) Random walk on Atlanta


(c) Quantum walk on Boston


(d) Random walk on Boston


(e) Quantum walk on Cardiff


(f) Random walk on Cardiff


(g) Quantum walk on Dallas

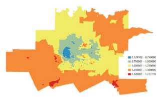
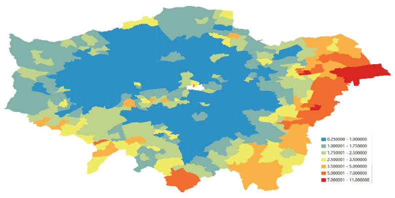

(h) Random walk on Dallas

Fig. 2. The normalized quantum walk convergence time $Q_i(q)/Q_i(q)^{null}$ spatial distribution provides detailed insights into the degree of racial segregation. $q = 0.7$

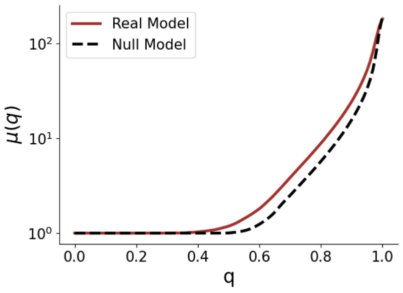
Spatial indicators of London Through Fig. 3, we first analyze the spatial distribution of $\tilde{Q}_i(q)$ values in London. Then we closely examine the behavior of spatial heterogeneity $\mu(q)$, spatial variance $\sigma(q)$, and spatial diversity $H(q)$. London is characterized by strong racial segregation[18]. In fact, some areas of the city clearly show larger $\tilde{Q}_i(q)$ values. In the east area of London, its $\tilde{Q}_i(q)$ value is more than seven times larger than the null model. We also observed an interesting phenomenon. When accessing the last one hundredth category, the random walk often takes an extremely large number of steps to access, while the image of the quantum walk appears relatively smooth. This is one of the advantages of quantum walks.

We understand that a larger value of $\mu(q)$ indicates an uneven distribution of categories within the system. A higher $\sigma(q)$ value reflects categories tending to form segregated homogeneous groups and clusters, quantifying the variation of QWCT across the entire space. A larger $H(q)$ value suggests minimal differences in QWCT between neighboring nodes, indicating local spatial diversity.

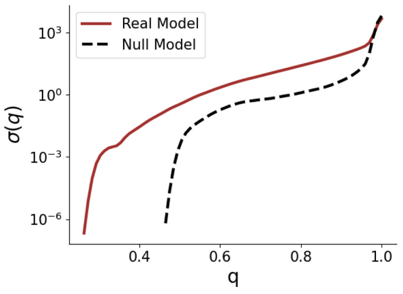
As shown in Fig. 3b, the lower $\mu(q)$ value of the null model implies a more uniform category distribution compared to the actual system. Similarly, as shown in Fig. 3a, the higher $H(q)$ value of the null model indicates less variation and greater uniformity in QWCT between adjacent nodes. In summary, for the city of London, the null model exhibits a more uniform category distribution compared to the real system.



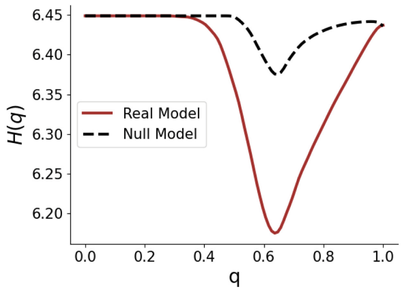
(a) Distribution of normalized QWCT values $\tilde{Q}_i(q)$ in London



(b) Distribution of spatial heterogeneity $\mu(q)$ for real and null model in London



(c) Distribution of spatial variance $\sigma(q)$ for real and null model in London



(d) Distribution of spatial diversity $H(q)$ for real and null model in London

Fig. 3. a The spatial distribution of normalized QWCT values in London. **bcd** The values of spatial heterogeneity $\mu(q)$, spatial variance $\sigma(q)$ and spatial diversity $H(q)$ for the real model(solid red line) and the corresponding null model(black dashed line), with the independent variable being the proportion q of categories visited(Color figure online).

From the analysis of category distribution, spatial heterogeneity $\Delta\mu$ is directly related to the size of clusters with similar racial compositions. Higher spatial variance $\Delta\sigma$ is associated with a more imbalanced distribution of categories in space. For cities and races, this implies that the convergence time varies greatly depending on the starting node. A high spatial diversity ΔH value suggests that convergence times in adjacent areas are relatively balanced, with no significant differences. Conversely, a low ΔH value indicates significant differences in convergence times between neighboring nodes, influenced by clusters with similar racial distributions. In general, smaller clusters result in shorter convergence times because walkers require less time to leave them and access neighboring nodes. Conversely, walkers need more time to exit larger clusters and find other races.

We computed the time statistics generated by quantum walk and random walk on the London area and its null model, resulting in three spatial metric indicators $\Delta\mu$, $\Delta\sigma$ and ΔH, as shown in Table. 2. We observed that the three metrics generated by the random walk method are all larger than those generated by the quantum walk. In reality, adjacent areas in London often organize into small clusters with very similar racial distributions[19], so its spatial heterogeneity metric $\Delta\mu$ should be smaller. The characteristic of quantum walk, where convergence time is not heavily dependent on the starting node, results in a smaller spatial variance metric $\Delta\sigma$. However, there is no significant difference in spatial diversity metric ΔH between them. Therefore, we conclude that the performance and accuracy of quantum walk are superior to those of random walk.

Table 2. $\Delta\mu$, $\Delta\sigma$, ΔH of time statistics generated by quantum walks and random walks in the London area.

	$\Delta\mu$	$\Delta\sigma$	ΔH
Quantum Walk	3.812	590.936	0.0642
Random Walk	5.532	750.476	0.0643

6 Conclusion

This paper proposes a method to quantify racial segregation between cities using quantum walks. We perform quantum walks on graphs and define the expected number of steps taken by the walker to visit different categories as Quantum Walk Convergence Time. Through virtual experiments, we demonstrate that higher QWCT values indicate more severe social segregation. By computing the average deviation from the corresponding null models, $\Delta\mu$, $\Delta\sigma$, and ΔH are suitable for comparing the spatial heterogeneity of the same variable in different systems, regardless of size, shape, or the number of different categories in the system.

We compare the results of quantum walk with those of traditional random walks and find that quantum walks can cover nodes of different categories faster. This is because particles in quantum walks can exist in superposition, enabling them to reach the target location more quickly, thus having a faster "diffusion" rate. In contrast, in random walks, walkers can only choose one path from all possible paths.Through measurements in the city of London, we find that QWCT generated by quantum walks provides a more accurate assessment of racial segregation. Additionally, we observe that the JS divergence between the real system and the null model is not significantly different, indicating that attribute invariance has minimal impact on the time statistics generated by quantum walks.

Due to the complexity involved in the concept of quantum walks and the extensive computational requirements, this study primarily focuses on two main concepts: quantum state superposition and unitary evolution. Therefore, further enhancement of the quantum walk method is needed for future research.

In summary, the QWCT shows promising results in quantifying racial segregation. While this study specifically investigates the relationship between QWCT and racial segregation, QWCT can also be applied to quantify the spatial distribution of other categories, such as socioeconomic status, educational level, and more.

Acknowledgement. This work is supported by the Natural Science Foundation of the Jiangsu Higher Educational Institution of China (Grant No. 24KJB510049), and Research Development Fund (RDF-23-01-044) at Xi'an Jiaotong-Liverpool University.

References

1. Massey, D.S., Denton, N.A.: The dimensions of residential segregation. Soc. Forces **67**(2), 281–315 (1988)
2. Wong, D.W.: Spatial indices of segregation. Urban studies **30**(3), 559–572 (1993)
3. Morgan, B.S.: A temporal perspective on the properties of the index of dissimilarity. Environ Plan A **15**(3), 379–389 (1983)
4. Ambainis, A.: Quantum walks and their algorithmic applications. International Journal of Quantum Information **1**(04), 507–518 (2003)
5. Kendon, V.: Quantum walks on general graphs. International Journal of Quantum Information **4**(05), 791–805 (2006)
6. Sousa, S., Nicosia, V.: Quantifying ethnic segregation in cities through random walks. Nat. Commun. **13**(1), 5809 (2022)
7. Venegas-Andraca, S.E.: Quantum walks: a comprehensive review. Quantum Inf. Process. **11**(5), 1015–1106 (2012)
8. Rossi, L., Torsello, A., Hancock, E.R., Wilson, R.C.: Characterizing graph symmetries through quantum jensen-shannon divergence. Physical Review E-Statistical, Nonlinear, and Soft Matter Physics **88**(3), 032806 (2013)
9. Farhi, E., Gutmann, S.: Quantum computation and decision trees. Phys. Rev. A **58**(2), 915 (1998)
10. Kadian, K., Garhwal, S., Kumar, A.: Quantum walk and its application domains: A systematic review. Computer Science Review **41**, 100419 (2021)
11. Mülken, O., Blumen, A.: Continuous-time quantum walks: Models for coherent transport on complex networks. Phys. Rep. **502**(2–3), 37–87 (2011)
12. Shannon, C.E.: A mathematical theory of communication. The Bell system technical journal **27**(3), 379–423 (1948)

13. Hardy, O.J.: Testing the spatial phylogenetic structure of local communities: statistical performances of different null models and test statistics on a locally neutral community. J. Ecol. **96**(5), 914–926 (2008)
14. Menéndez, M., Pardo, J., Pardo, L., Pardo, M.: The jensen-shannon divergence. J. Franklin Inst. **334**(2), 307–318 (1997)
15. Hershey, J.R., Olsen, P.A.: Approximating the kullback leibler divergence between gaussian mixture models. In: 2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07. vol. 4, pp. IV–317. IEEE (2007)
16. UK Data Service Census Support: Office for National Statistics (2011), https://borders.ukdataservice.ac.uk/, Last accessed 8 September 2023
17. US Census Bureau: Tiger Line Shapefiles (2010), http://www2.census.gov/geo/tiger/TIGER2010DP1/, Last accessed 8 September 2023
18. Barros, J., Feitosa, F.F.: Uneven geographies: Exploring the sensitivity of spatial indices of residential segregation. Environment and Planning B: Urban Analytics and City Science **45**(6), 1073–1089 (2018)
19. Catney, G.: The complex geographies of ethnic residential segregation: Using spatial and local measures to explore scale-dependency and spatial relationships. Trans. Inst. Br. Geogr. **43**(1), 137–152 (2018)

Clustering-based Image-Text Graph Matching for Domain Generalization

Nokyung Park[1], Daewon Chae[1], Jeongyong Shim[3], Sangpil Kim[2], Eun-Sol Kim[4](✉), and Jinkyu Kim[1]

[1] Department of Computer Science and Engineering, Korea University, Seoul, South Korea
jinkyukim@korea.ac.kr
[2] Department of Artificial Intelligence, Korea University, Seoul, South Korea
[3] Department of Artificial Intelligence Application, Hanyang University, Seoul, South Korea
[4] Department of Computer Science, Hanyang University, Seoul, South Korea
eunsolkim@hanyang.ac.kr

Abstract. Learning domain-invariant visual representations is important to train a model that can generalize well to unseen target task domains. Recent works demonstrate that text descriptions contain high-level class-discriminative information and such auxiliary semantic cues can be used as effective pivot embedding for domain generalization problems. However, they use pivot embedding in a global manner (i.e., aligning an image embedding with sentence-level text embedding), which does not fully utilize the semantic cues of given text description. In this work, we advocate for the use of local alignment between image regions and corresponding textual descriptions to get domain-invariant features. To this end, we first represent image and text inputs as graphs. We then cluster nodes within these graphs and match the graph-based image node features to the nodes of textual graphs. This matching process is conducted both globally and locally, tightly aligning visual and textual semantic sub-structures. We experiment with large-scale public datasets, such as CUB-DG and DomainBed, and our model achieves matched or better state-of-the-art performance on these datasets. The code is available at: https://github.com/noparkee/Graph-Clustering-based-DG

Keywords: Domain Generalization · Multimodal Learning

1 Introduction

How can humans effectively comprehend visual concepts despite variations in backgrounds, textures, and artistic styles? If it is impossible to collect sufficient examples of various combinations of domains, can current machine learning methods found on the i.i.d. assumption achieve robust generalization performance across domains? In this paper, we consider the domain generalization

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78192-6_26.

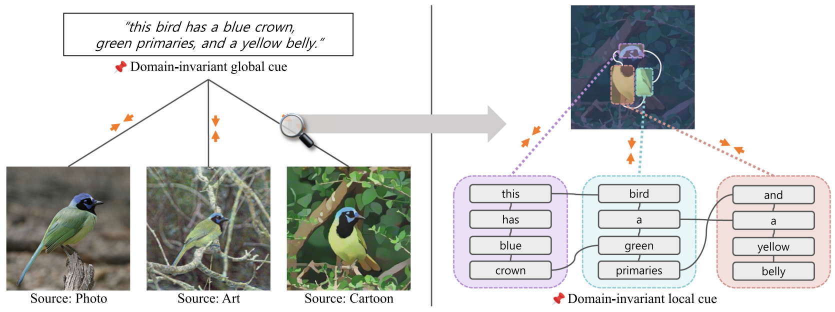

"this bird has a blue crown, green primaries, and a yellow belly."
Domain-invariant global cue
Source: Photo
Source: Art
Source: Cartoon
this
has
blue
crown
bird
a
green
primaries
and
a
yellow
belly
Domain-invariant local cue

Fig. 1. Our model learns domain-invariant visual representations by matching images and text descriptions at both global and local levels. Images and texts are represented as clustering-based graphs, encouraging the model to learn domain-invariant local semantic cues (e.g., "a yellow belly" and "green primaries").

problem on image datasets and introduce a novel clustering-based image-text graph matching to tackle the problem.

Domain generalization aims to improve a model's generalization ability for unseen task domains. Previous research has explored various approaches to address this challenge, including minimizing domain discrepancies in the visual feature space [19,35], augmenting data to cover various domains [18,40], and utilizing ensemble learning [8,22]. Notably, recent work such as GVRT [28] suggests leveraging natural language descriptions (e.g., "this bird has a blue crown, green primaries, and a yellow belly") to infuse visual encoders with domain-invariant semantic cues, i.e., a visual encoder is optimized to produce an embedding that aligns well with the corresponding text embedding. While promising, optimizing a model with such global alignment often leads to suboptimal results, as these models may lack diverse attribute focus and occasionally attend to irrelevant regions for the class (e.g., see Figure 6a).

To address these limitations, as shown in Figure 1, we focus on local matching, wherein image regions are matched with corresponding textual descriptions (e.g., an image region of blue crown and "blue crown" in a sentence). This approach involves representing the text descriptions with graphs and aligning the embedding of images and text by matching the graphs.

As shown in Figure 2, the suggested method consists of three parts: (i) a graph-based visual encoder, (ii) a graph-based textual encoder, and (iii) a graph-based alignment. Based on the graph-based representations, we aim to learn the domain-invariant features by grounding the graph-based image features into textual graphs, as the textual graphs contain explicitly verbalized knowledge from humans' typical reasoning. To solve the language grounding with structural information, we suggest a new method that clusters the graph node features then matches those clusters. By matching the multimodal graphs while clustering each node's features, our suggested method can get robust domain-invariant features representing multilevel semantic alignment.

Experimental results with two popular benchmark datasets, CUB-DG [28] and DomainBed [14], show the pivotal role of multimodal structural representations. Quantitatively, our suggested method achieves a new state-of-the-art performance, especially by increasing generalization ability on the most difficult domain *paint* of CUB-DG dataset. With robust qualitative visualization results, we argue that our model learns domain-invariant features across various feature resolutions by locally and globally aligning with textual graphs.

Our contributions can be summarized as follows. (1) We propose the first approach using graph representations for both image and text inputs for the DG problem. (2) We suggest a novel method that clusters and matches node features to align two multimodal graphs. (3) We achieve a new state-of-the-art DG performance on the CUB-DG dataset and DomainBed benchmark.

2 Related Work

Domain Generalization. Domain generalization aims to enhance a model's ability to generalize to unseen target domains with different data distributions compared to the source domains. The main idea of domain generalization is to learn domain-invariant features from multiple source domains. Various methods have been proposed to resolve this problem (i) by reducing domain discrepancies in the feature space [19,35], (ii) by implementing data augmentation [18,40], and (iii) by utilizing ensemble learning [8]. (iv) Other studies have proposed using auxiliary semantic cues to facilitate learning domain-invariant features [2,9,33].

Recently, GVRT [28] successfully leverages textual descriptions for models to learn domain-invariant visual representations by aligning them with verbalized (domain-invariant and class-discriminative) knowledge from humans' typical reasoning (e.g., given a text "this bird is black with an orange spot on its wing"). GVRT improves the model's generalization power by leveraging visual and textual inputs together and simply matching global representations. However, our focus extends beyond this, emphasizing the alignment of locally-aware high-order semantic relations via graph structures.

Graph Neural Network. Along with the huge success of neural networks in computer vision and natural language processing domains, new methodologies to deal with irregular structural inputs have been recently suggested. Various graph-based neural network algorithms are recommended for learning representations from structural inputs like molecular graphs, social networks, and meshes. According to the ways of representing graph data, attention-based methods (e.g., MoNet [6]), convolution-based methods (e.g., GCN [20]), and message-passing methods (e.g., MPNN [13]) can be applied to graph representation learning. Those graph neural network methods have achieved great performance on graph-related tasks, such as node classification [4], link prediction [41], and graph classification problems [42], by leveraging the non-euclidean data manifolds to get informative representations. Recently, the applications of graph neural networks have been extended to image and text domains [26]. By representing the image and text inputs as graphs, it becomes possible to consider the irregular and high-order correlations between tokens. In this paper, we suggest representing the multimodal inputs as graphs and matching the semantic correspondences between the multimodal inputs using graph neural networks to get the domain-invariant features.

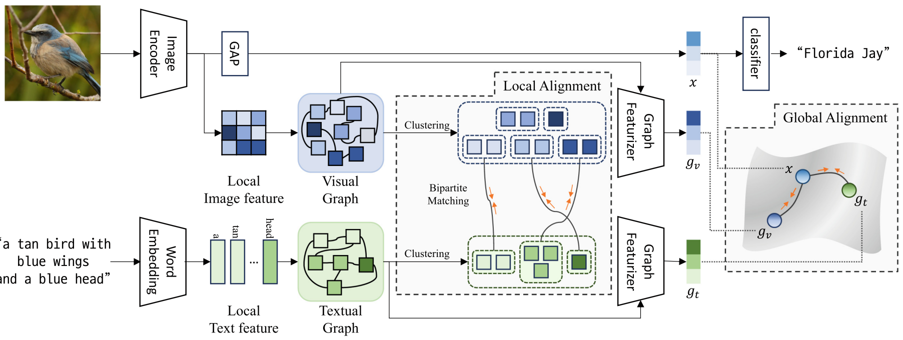


Fig. 2. An overview of our proposed method. We introduce multimodal graphs (visual and textual) that align with each other locally and globally, yielding domain-invariant visual features that are well-aligned with humans' explicitly verbalized knowledge.

3 Method

Given a distribution over multiple (or single) source domains $\{\mathcal{S}_1, \mathcal{S}_2, \dots\} \in \mathcal{S}$, the domain generalization (DG) problem considers the following classical stochastic optimization, in which we minimize the data-dependent generalization upper bound of the expected task loss [34]:

$$\underset{\theta}{\text{minimize}} \sup_{\mathcal{T}:\mathcal{D}(\mathcal{S},\mathcal{T})\leq\rho} \mathbb{E}_{\mathcal{T}}\left[\mathcal{L}(\theta;\mathcal{S})\right] \tag{1}$$

where we consider unseen target domains $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, \dots\}$ and the discrepancy between $\mathcal{S}$ and $\mathcal{T}$ is bounded by an arbitrary bound ρ, i.e. $D(\mathcal{S}, \mathcal{T}) \leq \rho$. We consider image classification scenarios and define the task-specific loss $\mathcal{L}$ function by the cross-entropy loss. Extracting domain-invariant representations from an image is key for training a model to generalize well to unseen domains.

Inspired by recent work by [28], we also want to improve the model's generalization power by leveraging visual and textual inputs together. Our model learns to extract (domain-invariant) visual representations that are well-aligned with explicitly verbalized knowledge from humans' typical reasoning. Unlike from [28] in that we focus more on aligning locally-aware high-order semantic relations via graphs instead of simply aligning global representations.

Figure 2 illustrates the architecture of our model, which is composed of three primary components: (i) a Graph-based Visual Encoder, (ii) a Graph-based Textual Encoder, and (iii) Graph-based Alignment for Learning Domain-Invariant Features. In (i), local latent representations (from a backbone network) are represented as a graph structure. Each local latent vector becomes a node, creating edges based on pairwise similarity in the embedding space (see Section 3.1). In (ii), we build a textual graph given a natural language description about a specific class (e.g., if the image corresponds to the *Florida Jay* class, the description could be "a tan bird with blue wings and a blue head."). Each word embedding forms a node, creating edges based on embedding-level similarity (refer Section 3.2). In (iii), We regularize multi-modal encoders for locally aligned representations by minimizing graph-level distances between visual and textual data. Applying clustering-based graph matching makes our model generalizable by learning human-compatible visual cues. (see Section 3.3).

3.1 Graph-based Visual Encoder

Global Visual Feature Extraction. Following standards in the domain generalization task, we use the pretrained ResNet50 [16] on ImageNet dataset [11] as a backbone. Our backbone takes an image $\mathcal{I}$ as an input and produces a d-dimensional global visual representation $\mathbf{x}_g \in \mathbb{R}^d$. This global representation $\mathbf{x}_g$ is trained to predict its classification label $\mathbf{y}$ with a linear layer, yielding the per-class softmax probabilities $\hat{\mathbf{y}}$. Both the backbone and the classifier are trained by a classification loss $\mathcal{L}_c$ as follows:

$$\mathcal{L}_c(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_i y_i \text{log}(\hat{y}_i) \tag{2}$$

where $\mathbf{y} \in \mathbb{R}^{|C|}$ represents the ground-truth one-hot vector, and $|C|$ denotes the size of the ground-truth class set. The model minimizes the loss function $\mathcal{L}_c$, but, unfortunately, this optimization often results in the model becoming semantically shallow. Thus model would not generalize well in environments different from those in which they were trained. In our work, we aim to regularize our model to understand relations between visual cues and use those relations for the final verdict, thus making it more generalizable. We want to achieve such a regularization effect through utterances from human verbalized reasoning.

Locally-aware Visual Graph Construction. We first construct a graph with visual representations to achieve the above mentioned goal. Formally, given M number of d-dimensional local visual representations $\mathbf{x}_l \in \{\mathbf{x}_{l,1}, \mathbf{x}_{l,2}, \dots, \mathbf{x}_{l,M}\}$ extracted from intermediate layers of the backbone (before global average pooling layer), we consider these representations as a set of unordered nodes. Note that each representation vector $\mathbf{x}_{l,i} \in \mathbb{R}^d$ for $i \in [1, M]$ corresponds to a certain grid over an input image $\mathcal{I}$. Inspired by the recent work [15], we construct a graph such that each node $\mathbf{x}_{l,i}$ has an edge with the other K_v nearest neighbors (Visual Graph in Figure 2). We use the widely-used L_2 distance to measure pairwise node similarity. In summary, our visual graph $\mathcal{G}_v = (\mathcal{V}_v, \mathcal{E}_v)$, where $\mathcal{V}_v$ denotes the node set consisting of M nodes (representing each visual feature of the local grid). The edge set $\mathcal{E}_v$ is comprised of the connections between nodes, with each node connected to its K_v nearest neighbors. Detailed explanations are provided in the supplementary material.

Graph-based Visual Representation. Given the visual graph $\mathcal{G}_v$, we further apply two layers of graph convolution network (GCN) [20], each followed by a linear layer, BatchNorm [17] layer and ReLU [1] activation. Subsequently, we employ dropout layer (only during training) and average readout operation to learn relational knowledge between local visual representations. Formally, we use a GCN-based function $f_{\text{GCN}}(\mathcal{G}_v)$ to obtain a final d_g-dimensional locally-aware visual graph representation $\mathbf{g}_v \in \mathbb{R}^{d_g}$, $\mathbf{g}_v = f_{\text{GCN}}(\mathcal{G}_v)$. Note that, we also add an additional classifier that takes the $\mathbf{g}_v$ as an input to create a graph that better captures the characteristics of the class. We provide detailed explanations in the supplementary material.

3.2 Graph-based Textual Encoder

Word-level Textual Graph Construction. Our graph-based visual representation $\mathbf{g}_v$ encodes relational knowledge via a graph structure between representations of local visual features $\mathbf{x}_l$. We empirically observe that such graph-based representation's sole use is still insufficient for models to learn domain-invariant and human-compatible visual cues. Thus, to regularize our visual encoders to be aligned with human knowledge, we build a textual graph from a natural language description of each image, followed by aligning both visual and textual graphs. A sequence of L (at maximum) words is first tokenized and encoded with a standard word-level (learnable) embedding layer, producing d_t-dimensional embedding vectors $\mathbf{t} \in \{\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_L\}$ where $\mathbf{t}_i \in \mathbb{R}^{d_t}$. Similar to our Visual Graph $\mathcal{G}_v$, we consider these word embeddings as an unordered set. We then construct a graph such that each node $\mathbf{t}_i$ has an edge with the other K_t nearest neighbors (textual graph in Figure 2). We use L_2 distance to measure pairwise node similarity. Finally, we create a textual graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$, where $\mathcal{V}_t$ denotes the node set comprising L nodes, each representing a textual feature of word embedding, and $\mathcal{E}_t$ signifies the edge set.

Graph-based Textual Representation. Given the textual graph $\mathcal{G}_t$, we apply the same architecture (but not shared) to obtain textual graph representation $\mathbf{g}_t \in \mathbb{R}^{d_g}$. I.e. we apply another GCN-based function $f_{\text{GCN}}(\mathcal{G}_t)$ to learn relational knowledge between word embeddings, $\mathbf{g}_t = f_{\text{GCN}}(\mathcal{G}_t)$.

3.3 Graph-based Alignment for Learning Domain-Invariant Features

We apply the following two graph-alignment approaches: (i) Graph-based Global Alignment and (ii) Local Alignment through Clustering-based Fine-grained Graph Matching, which comprises clustering and matching steps.

Global Graph Alignment. We assume that text descriptions inherently contain class-discriminative semantic cues. Thus, our model can learn domain-invariant features with aid of textual information. A standard approach to aligning different representations is minimizing the Euclidean distance. We employ this alignment technique to graph features as follows:

$$\mathcal{L}_{\text{global}} = ||f_{\text{proj},x}(\mathbf{x}_g) - f_{\text{proj},v}(\mathbf{g}_v)||_2 + ||f_{\text{proj},x}(\mathbf{x}_g) - f_{\text{proj},t}(\mathbf{g}_t)||_2 \quad (3)$$

where we use a linear layer to project each feature (i.e. $\mathbf{x}_g$, $\mathbf{g}_v$, and $\mathbf{g}_t$) such that these three projected features are pulled together. Note that $f_{\text{proj},x}$, $f_{\text{proj},v}$, and $f_{\text{proj},t}$ represent a projection layer. Importantly, as we use only a force to pull latent representations together, the training dynamics may become unstable, causing a representation collapse. To avoid this, like the approach in [28], we add an auxiliary classifier which is trained with the standard cross-entropy loss takes $f_{\text{proj},x}(\mathbf{x}_g)$ as an input to prevent a mode collapse, outputting the per-class softmax probability.

Clustering Graph Nodes. In addition to global alignment, initially, we tried to match the nodes in the visual graph with textual graph to align the locally-aware semantic relations. However, simply aligning nodes from two different graphs may not work as these nodes have different representations (i.e. a visual feature of a local image region vs. a word-level representation). Therefore, we present clustering-based local graph matching, which applies a node clustering algorithm to ensure that the two graphs have the same level of semantic representation and then performs graph matching. We define user-specified parameters $N_v(\leq M)$ and $N_t(\leq L)$ to the number of clusters for our visual and textual graphs, respectively. Note that we set $N_v \geq N_t$ since images may contain visual contents (e.g. backgrounds) that are not generally described in the text.

Our approach to constructing a graph is based on measuring node similarity, which can result in a well-defined semantic structure in the graph. Therefore, we choose a modularity-based method for graph clustering that can reflect this semantic structure while remaining stable. Specifically, we use a deep learning-based modularity measurement method [36]. Our model first encodes the cluster assignment matrix using the features of the graph nodes. Then, we calculate the modularity using this matrix, which measures the quality of the clustering. We train the model to maximize the modularity while also constraining it with collapse regularization to prevent trivial solutions such as assigning all nodes to the same cluster. We formulate it as follows:

$$\mathcal{L}_d = -\frac{1}{2m}\operatorname{Tr}\left(\mathbf{C}^{\mathrm{T}}\mathbf{B}\mathbf{C}\right) + \frac{\sqrt{k}}{n}\left\|\sum_i \mathbf{C}_i^{\mathrm{T}}\right\|_F - 1 \tag{4}$$

where $\mathbf{C}$ is the cluster assignment matrix calculated with our graph feature, and $\mathbf{B}$ is the modularity matrix calculated with the adjacency matrix. m, n, and k represent the number of edges, the number of nodes, and the number of clusters, respectively. The first term refers to modularity, which is measured using the assignment matrix and the modularity matrix, while the second term represents the collapse regularization term. So, our model can cluster semantically similar nodes together, allowing us to proceed with the matching process.

When applying node clustering to the visual graph $\mathcal{G}_v$, the cluster assignment matrix $\mathbf{C}$ has dimensions of $\mathbb{R}^{M\times N_v}$ (or $\mathbb{R}^{L\times N_t}$ when applied to the textual graph). Each element in matrix $\mathbf{C}$ at the i-th row and j-th column represents the softmax probability that the i-th local feature (or the i-th node) belongs to the j-th cluster. In this context, cluster feature is obtained through the following equation: $\mathcal{C}_v = f_{\text{proj},x}(\text{SeLU}((\mathbf{C}/N_v)^{\mathrm{T}}\mathbf{x}_l))$, where SeLU [21] serves as one of the activation functions. Note that when dealing with the textual graph $\mathcal{G}_t$, N_v and $\mathbf{x}_l$ are replaced by N_t and $\mathbf{t}$, respectively.

Clustering-based Graph Matching. Inspired by previous work [7], we use the set-based loss, i.e. the bipartite matching loss, between two disjoint sets of clusters: (i) a set of clusters $\mathcal{C}_v \in \{\mathcal{C}_v^1, \mathcal{C}_v^2, \ldots, \mathcal{C}_v^{N_v}\}$ of the visual graph $\mathcal{G}_v$ and (ii) a set of clusters $\mathcal{C}_t \in \{\mathcal{C}_t^1, \mathcal{C}_t^2, \ldots, \mathcal{C}_t^{N_t}\}$ from the textual graph $\mathcal{G}_t$. We minimize the following pair-wise matching loss:

$$\mathcal{L}_p = \frac{1}{N_t}\sum_{i=1}^{N_t} ||\mathcal{C}_v^{\mu_i} - \mathcal{C}_t^i||_2 \tag{5}$$

where $\mu_i \in \{1, 2, \ldots, N_v\}$ is the cluster index of $\mathcal{C}_v$ which matches to i in $\mathcal{C}_t$, producing the smallest total Euclidean distance by bipartite matching.

As the pair-wise matching loss pulls positive pairs together, negative pairs to add a repulsive force may need to prevent representation collapse. Thus, we also use a hinge loss based on $\mathcal{C}_v^i$ and $\mathcal{C'}_t^j$ (where $i \in [1, N_v]$ and $j \in [1, N_t]$), considering them as a negative pair if they are clusters for different input images. Thus, the matched distance L_p should be smaller than any other pairs between $\mathcal{C}_v^j$ and $\mathcal{C'}_t^i$ (or $\mathcal{C'}_v^j$ and $\mathcal{C}_t^i$). We formulate it as a hinge loss as follows:

$$\mathcal{L}_h = \max(0, \mathcal{L}_p - \text{MinDst}(\mathcal{C'}_v, \mathcal{C}_t) + \epsilon) + \max(0, \mathcal{L}_p - \text{MinDst}(\mathcal{C}_v, \mathcal{C'}_t) + \epsilon) \tag{6}$$

where $\text{MinDst}(\mathcal{C}_v, \mathcal{C'}_t)$ and $\text{MinDst}(\mathcal{C'}_v, \mathcal{C}_t)$ represents the minimum pair-wise matching loss similar to $\mathcal{L}_p$, but is applied between two disjoint sets of clusters originating from different inputs within a mini-batch. We compute it across all sample pairs in a mini-batch and use the average as the final loss value:

$$\mathcal{L}_{\text{local}} = \frac{1}{B}\sum_{b}(\lambda_d\mathcal{L}_d + \lambda_h\mathcal{L}_h + \lambda_{\text{aux}}\mathcal{L}_{\text{aux}}) \tag{7}$$

where we set the size of a mini-batch to B and λ_p, λ_h, and λ_d adjustable hyperparameters that control the weight of each loss term. In our model, values of 1, 0.1, and 0.1 are used for λ_d, λ_h, and λ_{aux}, respectively. Note that, similar to our global alignment module, we also add an auxiliary classifier that takes the average-pooled representation of visual clusters matched with textual clusters, denoted as $\frac{1}{N_t}\sum_{i=1}^{N_t}\mathcal{C}_v^{\mu_i}$, as an input. This classifier outputs the per-class softmax probability and is trained using the standard cross-entropy loss $\mathcal{L}_{\text{aux}}$.

Loss Function. Ultimately, we train our model end-to-end by minimizing the following loss L:

$$\mathcal{L} = \mathcal{L}_c + \mathcal{L}_{\text{global}} + \mathcal{L}_{\text{local}} \tag{8}$$

4 Experiments

4.1 Setup

Implementation Details. Same as previous domain generalization approaches, we also use ResNet50 [16], pre-trained on ImageNet [11], as our backbone, yielding a 2,048-dimensional visual representation from the last layer. Our model is trained end-to-end for 5,000 training steps using Adam optimizer with a learning rate of 5e-5. For training, we use standard image augmentation techniques such as random cropping, horizontal flipping, color jittering, grayscale conversion, and normalization. Our implementation is based on DomainBed [14], which is a unified domain generalization testbed, and our code will be publicly available upon publication. More details, including information that varies depending on the dataset, are available in supplementary material.

Datasets. To demonstrate our model's effectiveness, we first use the CUB-DG dataset (for fine-grained image classification task), which is extended from the CUB dataset [39] for the domain generalization task. This dataset contains 11,768 images for 200 classes of North American bird species. Each image has 10 text descriptions describing the content in detail, e.g., "this bird is black with an orange spot on its wing." Each image is manipulated to create the following four domains: Photo, Cartoon, Art, and Painting. We follow the common evaluation protocol and use the CUB-DG dataset's official split (the train and validation set has 5,994 samples, while the test set has 5,794 samples).

Further, we also evaluate our model on DomainBed [14], which contains the following five multi-domain DG datasets: VLCS [12], PACS [25], OfficeHome [38], TerraIncognita [3], and DomainNet [23]. Among these, we would emphasize that PACS [25] dataset is useful for our experiments as (i) it provides a bigger domain shift than existing photo-only benchmarks, and (ii) it needs to exploit local information to learn discriminative subtle visual features. We follow the standard evaluation protocol used in [14]. For datasets that do not provide text inputs, we use both (1) textual class definitions from the Oxford dictionary similar to GVRT [28] and (2) descriptions generated by InstructBLIP [10] with the prompt "write a detailed description about the image." (refer supplementary material).

4.2 Performance Comparison

Table 1. The out-of-distribution classification accuracies (in %) on CUB-DG (top) and PACS (bottom) datasets based on the standard leave-one-out multi-source DG task setting. We compare ours with other existing DG approaches. (we provide full tables in supplementary material). *Abbr.* I: Image, T: Text.

Algorithms	Modality	Target Domain (Data: CUB-DG [28])				Avg. ↑
		Photo	Cartoon	Art	Paint	
MIRO [9]	I	68.2	59.1	46.5	38.2	53.0
SD [31]	I	71.3	62.2	50.8	34.8	54.7
CORAL [35]	I	72.2	63.5	50.3	35.8	55.4
CCFP [24]	I	70.0	61.5	52.1	<u>40.4</u>	56.0
GVRT [28]	I+T	<u>74.6</u>	<u>64.2</u>	<u>52.2</u>	37.0	57.0
Ours	I+T	**75.4**	**65.5**	**54.0**	**41.4**	**59.1**
Algorithms	Modality	Target Domain (Data: PACS [25])				Avg. ↑
		Art Painting	Cartoon	Photo	Sketch	
SelfReg [19]	I	<u>87.9 ± 1.0</u>	79.4 ± 1.4	96.8 ± 0.7	78.3 ± 1.2	85.6
CORAL [35]	I	**88.3** ± **0.2**	80.0 ± 0.5	97.5 ± 0.3	78.8 ± 1.3	86.2
mDSDI [5]	I	87.7 ± 0.4	80.4 ± 0.7	**98.1** ± **0.3**	78.4 ± 1.2	86.2
SagNet [29]	I	87.4 ± 1.0	80.7 ± 0.6	97.1 ± 0.1	80.0 ± 0.4	86.3
CCFP [24]	I	87.5 ± 0.1	<u>81.3 ± 0.3</u>	96.4 ± 0.3	**81.4** ± **0.8**	86.6
Ours	I+T	<u>87.9 ± 0.7</u>	**81.4** ± **0.1**	<u>98.0 ± 0.1</u>	<u>80.5 ± 1.1</u>	**87.0**

As shown in Table 1, we compared the out-of-distribution classification accuracies on the following two datasets: CUB-DG (top) and PACS (bottom) datasets. We compare ours with other existing state-of-the-art domain generalization approaches, SagNet [29], MIRO [9], SD [31], CORAL [35], GVRT [28], SelfReg [19], mDSDI [5], and CCFP [24]. Due to space constraints, we only report top-6 results (we provide full tables in supplementary material).

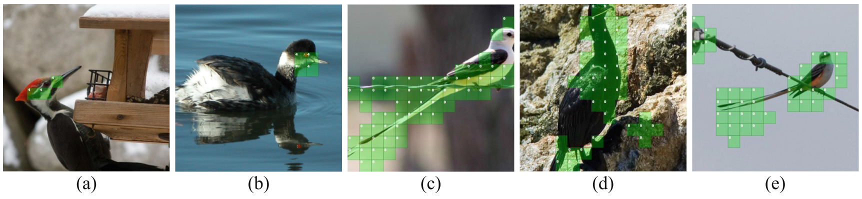
As shown in Table 1 (top), our proposed method clearly outperforms the other domain generalization techniques on the CUB-DG dataset in all target domains with a significant gain. In terms of the average image classification accuracy, ours shows 59.1%, which is 2.1% higher than GVRT [28] (which uses the same image and text inputs) and 3.1% higher than image-only approach, CCFP [24]. Similar trends are also observed in our experiment on the large-scale PACS [25] dataset. As shown in Table 1 (bottom), our model also outperforms the other approaches, i.e., ours shows 87.0% that is 0.4% higher than CCFP [24]-based SOTA approach and 1.9% higher than GVRT [28]. These confirm that our graph-based approach is effective in aligning visual and textual encoders for fine-grained image classification tasks, improving the visual encoder's generalization power to unseen target domains.

4.3 Few-shot DG Performance Comparison

Conventional DG approaches often assume that a sufficient number of images is available for all classes and domains enough to learn domain-invariant class-discriminative features. However, this may be practically challenging in real-world scenarios. We emphasize that our method, which leverages textual descriptions as pivotal information, can benefit learning domain-invariant features in the few-shot setting. In Table 2, our model significantly outperforms the other approaches on PACS [25] and VLCS [12] datasets. Note that we use randomly chosen five images (per class in each domain) as an input to train all models (i.e., 5-shot DG). Except for the number of training images, we generally follow the standard protocol of DomainBed for evaluation.

Table 2. Few-shot DG performance comparison.

Algorithm (Data: PACS [25])	Avg.	Algorithm (Data: VLCS [12])	Avg.
mDSDI [5]	63.5	mDSDI [5]	68.5
CORAL [35]	64.6	GVRT [28]	69.4
MIRO [9]	65.5	MIRO [9]	69.6
GVRT [28]	68.7	CORAL [35]	71.1
Ours	**70.7**	Ours	**71.4**



(a) This **medium sized** bird **has a red crown**, black **superciliary and** white **cheek patch**.

(b) This bird **has** wings that are black **and has orange eyes**.

(c) This bird **has a** white head **and chest** that **slowly turns in** to **a peach color near his feet, and has extremely long tail** feathers that are **longer than** his body.

(d) This bird **has** wings that are black **and has a long neck**.

(e) This bird **has** wings that are brown with very **long tailfeathers**.

Fig. 3. Examples of the matched image region (in visual graph clusters) and texts (in textual graph clusters).

4.4 Analysis on CUB-DG Dataset

Analysis of Graph Clusters and Their Matchings. In Figure 3, we provide examples of a matched pair of image regions and a set of words. For example, in (a), a region around the bird's head is matched with a textual graph cluster that contains words including "orange eyes". We observe that our model reasonably matches image features with class-discriminative texts, e.g., red crown, cheek patch, extremely long tail, and long neck.

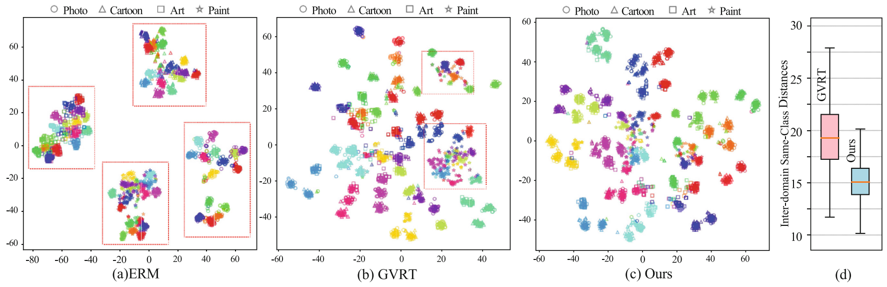


Fig. 4. Visualizations by t-SNE for (a)ERM [37], (b)GVRT [28], and (c)Ours on CUB-DG. Points are color-coded differently by its class and has different shapes according to its domain. (d)We also compare inter-domain same-class distances.

As shown in Figure 4, we provide t-SNE [27] visualization of (a) ERM [37], (b) GVRT [28], and (c) Ours to visualize their embedding space on CUB-DG dataset. We use different marker styles (for target domains) and different colors (for classes). An ideal model would show that visual features of the same class but different domains are gathered together. Ours clearly outperform ERM, which has scattered points per domain and better than GVRT in that features of the same class but different domains are more clustered (see red boxes). In Figure 4 (d), we provide box plots for GVRT and ours, showing that our model produces lower same-class inter-domain distances than GVRT. Note that we provide detailed t-SNE visualizations in supplementary material. Figure 5 further shows the nearest examples in the unseen target domain to the given image (e.g., "dog"). In contrast to MIRO and GVRT, which often provide examples of different classes (e.g., "person"), ours consistently provide examples of the same class. This is consistent with our t-SNE analysis. We provide more examples in in supplementary material.

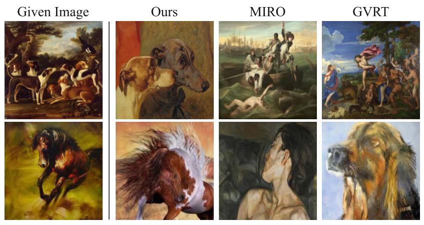


Fig. 5. Exemplars of the nearest examples from PACS dataset (in the unseen target domain) to the given image (e.g., "dog" and "horse").

GradCAM Visualization. In Figure 6, we use GradCAM [32] to visualize image regions where the model focuses on for the final verdict. We observe that our model generally focuses on multiple class-discriminative features, giving the benefits of more robust and generalizable recognition performance. More specifically, in the first image of Figure 6b, our model focuses on the head region, a relevant area for the classification task, in contrast to the GVRT model which attends to a region less relevant to class information (Figure 6a).

Furthermore, in the second image of the GVRT scenario, the model concentrates solely on the belly region, while our model exhibits a more diverse focus on attributes, encompassing both the belly and beak. Unlike GVRT, which focuses solely on global alignment, our model incorporates local alignment through graphs. This approach enables our model to capture diverse attributes, increasing its ability to generalize effectively.

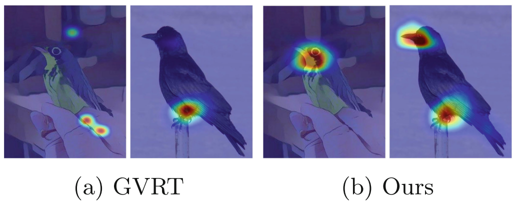


Fig. 6. GradCAM [32] visualizations to evaluate where the model sees.

Table 3. Ablation studies to compare variants of our model. Data: CUB-DG.

Global Alignment	Local Alignment	Visual Graph	Textual Graph	Target Domain				Avg.
				Photo	Cartoon	Art	Paint	
-	-	✓	✓	65.1	52.5	38.2	29.0	46.2
✓	-	-	-	69.5	57.1	44.2	30.2	50.2
✓	-	✓	-	70.3	57.0	48.1	33.5	52.2
✓	-	-	✓	75.0	64.4	53.0	34.7	56.8
-	✓	✓	✓	71.4	57.6	46.6	37.2	53.2
✓	✓	✓	✓	**75.4**	**65.5**	**54.0**	**41.4**	**59.1**

Ablation Studies. In Table 3, we conduct an ablation study to demonstrate the effect of main modules: (i) a global alignment, (ii) a local alignment, (iii) a visual graph, and (iv) a textual graph. Our study demonstrates that (1) a global alignment, which aligns graph-level features together, effectively improves accuracies, especially in photo, cartoon, and art domains. (2) Adding local alignment, which aligns graphs via the clustering-based matching algorithm, improves all domains while using both alignments outperforms the alternatives. (3) Either using a visual or textual graph alone improves model generalization, but the gain is marginal with the visual graph alone. (4) The gain is maximized by using both graphs, which indicates that a graph structure effectively transfers text knowledge to train a generalizable visual encoder.

Table 4. The test accuracies (in %) on the DomainBed datasets in the multi-source DG task setting. We compare ours with other existing DG approaches (we provide full tables in supplementary material). †: utilizing texts from a dictionary, ‡: incorporating texts from InstructBLIP. *Abbr.* I: Image, T: Text.

Algorithm	Modality	Dataset					Avg.
		VLCS	PACS	OfficeHome	TerraIncognita	DomainNet	
CORAL [35]	I	78.8 ± 0.6	86.2 ± 0.3	68.7 ± 0.3	47.6 ± 1.0	41.5 ± 0.1	64.6
CCFP [24]	I	78.9 ± 0.3	86.6 ± 0.2	68.9 ± 0.1	48.6 ± 0.4	41.2 ± 0.0	64.8
mDSDI [5]	I	79.0 ± 0.3	86.2 ± 0.2	69.2 ± 0.4	48.1 ± 1.4	42.8 ± 0.1	65.1
GVRT (PTE) [28]	I+T	79.0 ± 0.2	85.1 ± 0.3	70.1 ± 0.1	48.0 ± 0.2	44.1 ± 0.1	65.2
MIRO [9]	I	79.0 ± 0.0	85.4 ± 0.4	70.5 ± 0.4	50.4 ± 1.1	44.3 ± 0.2	**65.9**
Ours†	I+T	78.3 ± 0.4	85.7 ± 0.1	70.1 ± 0.1	49.5 ± 0.9	43.7 ± 0.0	65.5
Ours‡	I+T	78.6 ± 0.3	87.0 ± 0.4	70.4 ± 0.2	49.2 ± 0.5	44.2 ± 0.0	**65.9**

Table 5. Performance comparison between variants of our model with different matching techniques: bipartite matching and greedy matching. Data: CUB-DG.

Bipartite Matching	Greedy Matching	Time Complexity	Target Domain				Avg.
			Photo	Cartoon	Art	Paint	
-	✓	$O(V^2)$	75.3	64.7	**54.8**	36.6	57.8
✓	-	$O(V^3)$	**75.4**	**65.5**	54.0	**41.4**	**59.1**

Complexity of Graph Matching. In our matching algorithm design, we explored two approaches: bipartite matching and greedy matching. Bipartite matching establishes one-to-one correspondences between clusters, minimizing pairwise distances, while greedy matching allows many-to-one associations based on spatial proximity. Bipartite matching operates with a time complexity of $O(V^3)$, where V denotes the number of vertices, while greedy matching operates in $O(V^2)$. Despite the higher time complexity of bipartite matching, our experimental results (refer to Table 5) demonstrate its superior performance over greedy matching. This may be attributed to the constraints imposed on one-to-one matching, which result in a dispersed effect on models attempting to optimize cluster pairs on a global scale. Moreover, given that V does not exceed 5 in our method, we opted for bipartite matching due to its enhanced performance in our specific context.

4.5 Performance on DomainBed Benchmark

We evaluate our model with a large-scale DomainBed [14] datasets. We use the following five multi-domain datasets, including VLCS [12], PACS [25], OfficeHome [38], TerraIncognita [3], and DomainNet [30], comparing ours with 19 domain generalization algorithms. Due to space constraints, we only report top-6 results (see supplementary material for full table). The reported score represents averaged results obtained from three independent runs using randomly chosen hyperparameters. We observe in Table 4 that our proposed method shows matched or better state-of-the-art performance, where it ranks 1st (tied) in average performance.

5 Conclusion

We propose a novel domain generalization method that encodes domain-invariant visual representations. To this end, we use a textual description to utilize verbalized (domain-invariant) knowledge from humans' typical reasoning. To align these, we use a clustering-based graph-matching algorithm based on visual and textual graphs built upon images and texts, respectively. We evaluate our model with state-of-the-art domain generalization approaches on CUB-DG and DomainBed datasets, achieving SOTA performance.

Acknowledgements. This work was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2021R1A6A1A13044830, 20%) and supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2022-II220043, Adaptive Personality for Intelligent Agents, 30%, IITP-2024-RS-2024-00397085, Leading Generative AI Human Resources Development, 30%, RS-2202-II220264, Comprehensive Video Understanding and Generation with Knowledge-based Deep Logic Neural Network, 20%). Daewon Chae was supported by Hyundai Motor Chung Mong-Koo Foundation.

References

1. Agarap, A.F.: Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375 (2018)
2. Bai, H., Sun, R., Hong, L., Zhou, F., Ye, N., Ye, H.J., Chan, S.H.G., Li, Z.: Decaug: Out-of-distribution generalization via decomposed feature representation and semantic augmentation. In: Proceedings of the AAAI Conference on Artificial Intelligence (2021)
3. Beery, S., Van Horn, G., Perona, P.: Recognition in terra incognita. In: Proceedings of the European conference on computer vision (ECCV) (2018)
4. Bhagat, S., Cormode, G., Muthukrishnan, S.: Node classification in social networks. CoRR (2011)
5. Bui, M.H., Tran, T., Tran, A., Phung, D.: Exploiting domain-specific features to enhance domain generalization. Advances in Neural Information Processing Systems (2021)
6. Burgess, C.P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M.M., Lerchner, A.: Monet: Unsupervised scene decomposition and representation. CoRR (2019)
7. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-End Object Detection with Transformers. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12346, pp. 213–229. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_13

8. Cha, J., Chun, S., Lee, K., Cho, H.C., Park, S., Lee, Y., Park, S.: Swad: Domain generalization by seeking flat minima. Advances in Neural Information Processing Systems (2021)
9. Cha, J., Lee, K., Park, S., Chun, S.: Domain generalization by mutual-information regularization with pre-trained models. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIII (2022)
10. Dai, W., Li, J., Li, D., Tiong, A.M.H., Zhao, J., Wang, W., Li, B., Fung, P., Hoi, S.: Instructblip: Towards general-purpose vision-language models with instruction tuning (2023)
11. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition (2009)
12. Fang, C., Xu, Y., Rockmore, D.N.: Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2013)
13. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. CoRR **abs/1704.01212** (2017)
14. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. arXiv preprint arXiv:2007.01434 (2020)
15. Han, K., Wang, Y., Guo, J., Tang, Y., Wu, E.: Vision gnn: An image is worth graph of nodes. Advances in Neural Information Processing Systems (2022)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2016)
17. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning (2015)
18. Kang, J., Lee, S., Kim, N., Kwak, S.: Style neophile: Constantly seeking novel styles for domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2022)
19. Kim, D., Yoo, Y., Park, S., Kim, J., Lee, J.: Selfreg: Self-supervised contrastive regularization for domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021)
20. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
21. Klambauer, G., Unterthiner, T., Mayr, A., Hochreiter, S.: Self-normalizing neural networks. Advances in neural information processing systems (2017)
22. Lee, K., Kim, S., Kwak, S.: Cross-domain ensemble distillation for domain generalization. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXV (2022)
23. Leventidis, A., Di Rocco, L., Gatterbauer, W., Miller, R.J., Riedewald, M.: Domainnet: Homograph detection for data lake disambiguation. arXiv preprint arXiv:2103.09940 (2021)
24. Li, C., Zhang, D., Huang, W., Zhang, J.: Cross contrasting feature perturbation for domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1327–1337 (2023)
25. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.: Deeper, broader and artier domain generalization. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)

26. Liu, C., Mao, Z., Zhang, T., Xie, H., Wang, B., Zhang, Y.: Graph structured network for image-text matching. CoRR (2020)
27. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research (2008)
28. Min, S., Park, N., Kim, S., Park, S., Kim, J.: Grounding visual representations with texts for domain generalization. In: European Conference on Computer Vision. pp. 37–53. Springer (2022)
29. Nam, H., et al.: Reducing domain gap by reducing style bias. In: CVPR (2021)
30. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2019)
31. Pezeshki, M., Kaba, S.O., Bengio, Y., Courville, A., Precup, D., Lajoie, G.: Gradient starvation: A learning proclivity in neural networks. arXiv preprint arXiv:2011.09468 (2020)
32. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision (2017)
33. Shahtalebi, S., Zhu, Z., Rudzicz, F.: Out-of-distribution failure through the lens of labeling mechanisms: An information theoretic approach. In: ICML 2022: Workshop on Spurious Correlations, Invariance and Stability (2022)
34. Sinha, A., Namkoong, H., Volpi, R., Duchi, J.: Certifying some distributional robustness with principled adversarial training. ICLR (2017)
35. Sun, B., Saenko, K.: Deep coral: Correlation alignment for deep domain adaptation. In: Proceedings of the European Conference on Computer Vision (ECCV) (2016)
36. Tsitsulin, A., Palowitch, J., Perozzi, B., Müller, E.: Graph clustering with graph neural networks. arXiv preprint arXiv:2006.16904 (2020)
37. Vapnik, V.N.: An overview of statistical learning theory. IEEE transactions on neural networks (1999)
38. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2017)
39. Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., Perona, P.: Caltech-UCSD Birds 200. Tech. rep, California Institute of Technology (2010)
40. Xu, Q., Zhang, R., Zhang, Y., Wang, Y., Tian, Q.: A fourier-based framework for domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021)
41. Zhang, M., Chen, Y.: Link prediction based on graph neural networks. CoRR (2018)
42. Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. Proceedings of the AAAI Conference on Artificial Intelligence (2018)

Knowledge from Large-Scale Protein Contact Prediction Models Can Be Transferred to the Data-Scarce RNA Contact Prediction Task

Yiren Jian[1](✉), Chongyang Gao[2], Chen Zeng[3], Yunjie Zhao[4], and Soroush Vosoughi[1]

[1] Dartmouth College, Hanover, NH 03755, USA
yiren.jian.gr@dartmouth.edu
[2] Northwestern University, Evanston, IL 60208, USA
[3] The George Washington University, Washington, DC 20052, USA
[4] Central China Normal University, Wuhan 430079, China

Abstract. RNA, whose functionality is largely determined by its structure, plays an important role in many biological activities. The prediction of pairwise structural proximity between each nucleotide of an RNA sequence can characterize the structural information of the RNA. Historically, this problem has been tackled by machine learning models using expert-engineered features and trained on scarce labeled datasets. Here, we find that the knowledge learned by a protein-coevolution Transformer-based language model can be transferred to the RNA contact prediction task. As protein datasets are orders of magnitude larger than those for RNA contact prediction, our findings and the subsequent framework greatly reduce the data scarcity bottleneck. Experiments confirm that RNA contact prediction through transfer learning using a publicly available protein language-model is greatly improved. *Our findings indicate that the learned structural patterns of proteins can be transferred to RNAs, opening up potential new avenues for research. The code and data (for inference and training) are available at https://github.com/yiren-jian/CoT-RNA-Transfer.*

Keywords: Transfer Learning · RNA Contact Prediction · Biological Modeling

1 Introduction

Proteins and RNAs are critical to many biological processes such as coding, regulation, and expression [9,10,13,30,31]. Understanding their structures is key to deciphering their functionalities. While experimental methods like X-ray diffraction [32], nuclear magnetic resonance (NMR) [5], and Cryogenic electron microscopy (Cryo-EM) [11] can determine 3D structures, it remains challenging

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15310, pp. 407–423, 2025.
https://doi.org/10.1007/978-3-031-78192-6_27

for structurally flexible molecules, e.g., RNAs [24]. Consequently, the Protein Data Bank has limited RNA structures cataloged [3].

In response, many computational tools for 3D structure prediction of biological molecules have been developed in the last decade [19,20,40]. Recently, deep neural networks [6,18,22] have revolutionized 3D protein structure prediction, partly due to their large size and training datasets. However, this progress has not been paralleled for RNAs, mainly due to the scarcity of RNA datasets. Current RNA datasets are significantly smaller than protein datasets, with well-curated datasets containing less than 100 RNAs [42] and models trained on fewer than 300 RNA structures [33]. These small datasets are insufficient for training large deep neural networks.

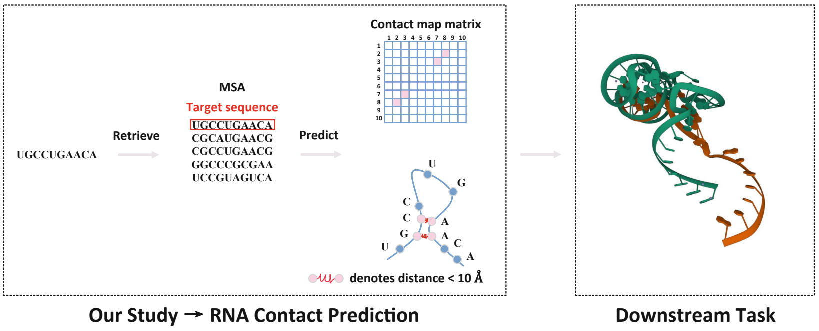


Fig. 1. Our study is focused on RNA contact prediction, i.e., predicting the contact map matrix for an RNA sequence. The contact map indicates the proximity between each nucleotide, with those closer than a threshold (10 Å) being deemed in contact. Correct predictions of the contact map can benefit downstream tasks, e.g., by acting as constraints for filtering 3D RNA structure predictions.

In the absence of powerful 3D structure prediction models, certain structural properties of RNAs can be determined through RNA Contact Prediction [16]. The contact predictions can be used as an intermediary step to facilitate the prediction of 3D structures or directly for downstream tasks that rely on RNA structural information. For an RNA sequence of length L, this task aims at predicting a $L \times L$ symmetric binary matrix (called contact map) where a value of 1 at position (i, j) indicates that the i^{th} and j^{th} nucleotides are in contact[1] with each other to each other in 3D space. The predicted contact maps capture structural constraints, which can be used for downstream tasks, such as refining RNA 3D prediction tools [37] (see Fig. 1 for an overview of the task). Note that for a target RNA sequence, the input for an RNA contact prediction model is an RNA multiple sequence alignment (MSA), which corresponds to the target RNA sequence stacked with known homologous sequences.

[1] Contact is defined as distances smaller than a specific threshold. Following prior works, this is set by a hard distance threshold of 10 Å.

The first RNA Contact Prediction attempt was Direct Coupling Analysis (DCA), with variants like mfDCA [25], mpDCA [38], plmDCA [8], bmDCA [26], and PSICOV [17]. These self-supervised methods infer contact maps using maximum likelihood estimation without labeled datasets. Recently, supervised RNA contact prediction methods leverage additional knowledge from RNA analytic tools for more informative features. These small models are necessitated by limited training examples. In contrast, abundant protein data allowed for training a large Transformer-based deep neural network, Co-evolution Transformer (CoT), for protein contact prediction [43]. CoT was trained on 90K curated protein structures, compared to ≤ 100 RNA structures.

Since we lack the data to train such a model for RNA contact prediction from scratch, we investigate the possibility of re-using and tuning the learned parameters of a pre-trained protein language-model (such as CoT) to create an RNA contact prediction model, a process referred to as transfer learning. Inspired by recent breakthroughs in unified vision-language models [2,21,36] and transfer learning across text and visual domains [15,23], which have demonstrated the effectiveness of transferring knowledge between related modalities, such as leveraging the structural abilities learned from code and music to enhance language models [27], we propose that bio-molecule contact patterns learned by the CoT protein Transformer network could be transferred to improve RNA contact prediction performance.

Similar to RNA contact prediction models, CoT takes protein MSAs as input. The input to CoT is represented using English characters, with each amino acid represented by a unique English character. CoT then utilizes the attention mechanism of Transformers [35] to learn the contacts, analogous to how Transformer-based language models, such as GPT-3 [4], learn dependencies between words in a given text. Though at the surface level, RNA and protein sequence data are comprised of different building blocks (nucleotides for RNAs and amino acids for proteins), we speculate that they share deeper similarities concerning their contact patterns, analogous to two languages with different lexicons but a similar syntax. Hence, it may be possible to transfer knowledge about contact patterns from one to another, analogous to cross-lingual transfer in Transformer-based language models [12].

We investigate our hypothesis by adapting the pre-trained CoT to our RNA dataset and using the adapted representations to train a convolutional network (ConvNet) for RNA contact prediction (see Fig. 2 for an overview of our method). Our explorations show that this simple method, which does not rely on any additional pre-processing or feature engineering and can detect true contacts missed by prior works. In addition to improving RNA contact prediction by using knowledge from a pre-trained protein language-model, ***more importantly, our study serves as a strong proof of concept for the possibility of transfer learning between the proteins and RNAs.***

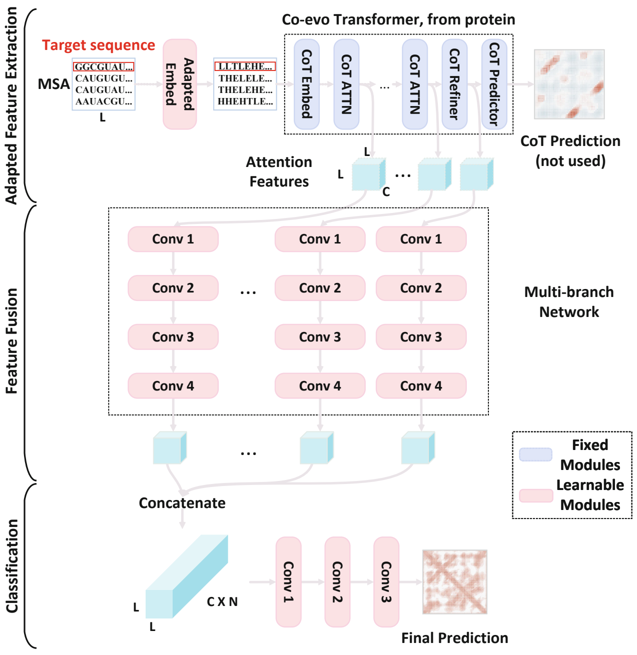


Fig. 2. Overview of our three-stage method (from top to bottom). *Adapted Feature Extraction*: First, a projection layer is used to translate the RNA MSA sequences into protein language (e.g., from nucleotide "AUCG" to amino acids "HETL"). Then, we leverage a fixed large-scale pre-trained protein contact prediction transformer model (called Co-evolution Transformer model (CoT)) to extract attentive (i.e., contribution) features at different layers. *Feature Fusion*: Features from different layers are processed by separate convolution blocks before being concatenated. *Classification*: The aggregated features are sent into a standard Convolutional Network (ConvNet) classifier with three layers of convolution.

2 Background and Related Works

Unsupervised Contact Prediction Based on the Co-Evolution Hypothesis. The *co-evolution hypothesis* is the basis of many contact prediction methods (for both proteins and RNA). The hypothesis suggests that spatially proximate pairs of amino acids or nucleotides tend to co-evolve to maintain their structure and function [38]. DCA methods are all purely unsupervised, based on the counting frequency of the residues in MSA, and can be applied to proteins and RNA sequences.

Supervised Contact Prediction. Given $\sim$ 180K known protein structures in PDB, A 20M-parameters attention-based Transformer model for end-to-end prediction of protein contacts based on MSA [43]. The attention mechanism of the model, called the Co-evolution Transformer (CoT), is specifically designed to model co-evolution by considering the outer product of representations of two positions.

Such a model is only successful given a large labeled dataset of known MSA to contact mappings.

Training such a large model for RNA is unfortunately not practical as there are currently no such large datasets available. The largest of such data is at least 2–3 orders of magnitude smaller than what is available for proteins. To overcome this bottleneck, most resort to feature engineering to train smaller models. For instance, a recent work [33] combines DCA outputs (or similarly, covariance matrices) with other features (such as predicted secondary structures, solvent surface areas, etc.) extracted from different RNA analysis tools to train relatively small convolution networks, using only hundreds of labeled data point. Then, CoCoNet [42] shows that the output of DCA by itself is sufficient for training such models and that oftentimes expensive additional feature extraction is not needed. Finally, a recent work [34] investigates self-supervised training with regression for RNA contact prediction.

Transfer Learning. We show in this paper that the learned knowledge of a pre-trained protein contact prediction model can be effectively used for RNA contact prediction, not only removing the need for additional feature engineering and extraction but also vastly outperforming CoCoNet. Our proposed method is built upon the concept of "Transfer Learning" [7], which assumes that knowledge learned from one task is beneficial to other related tasks. Transfer learning has enabled the adaption of large pre-trained deep neural networks to new tasks with a limited number of labeled examples. This is typically done by training newly initialized layers at the end of the pre-trained network (which tends to be task-specific) using the small dataset while keeping the other layers frozen (which preserves the learned knowledge from the previous task). Only the relatively small set of parameters in the final layers will be updated, which will adapt the network to the new task.

A key challenge of RNA contact prediction is the small dataset size, which prohibits us from learning a deep model from scratch. We hypothesize (and later verify) that knowledge could be effectively transferred from a pre-trained protein contact Transformer to RNA contact prediction, enabling us to train high-performing RNA contact prediction models without the need for additional labeled or feature engineering, both of which can be prohibitively expensive. Analogies can be drawn between our approach and research done on the cross-lingual transfer of language models [12] that adapt a pre-trained model to a new language by learning its syntax while retaining the semantic knowledge in the pre-trained model; here we are adapting a biological model pre-trained on the "language of proteins" to the "language of RNAs".

Our model design incorporates Transformer blocks followed by convolutional networks [1,39,44]. While this framework is not innovative in terms of its architecture, we are the first to apply it to the RNA contact prediction task. This novel application demonstrates significant improvements and addresses the critical challenge of data scarcity in RNA contact prediction by leveraging knowledge transferred from protein datasets.

3 Methods and Setups

3.1 Protein-To-RNA Transferred Contact Prediction Model

In this section, we provide details of our model's architecture, input, and output. An overview of our approach is visualized in Fig. 2.

MSA as Input. Our RNA contact prediction model relies on the CoT model, which takes protein MSA as the input. Thus, we need to adapt or map the RNA language, which is comprised of nucleotides, to the protein language, which is comprised of amino acids. Specifically, suppose our target RNA MSA has M aligned sequences, each with the length of L nucleotides. Then, the RNA MSA can be represented as a $M \times L$ matrix, with each element being "A", "U", "C"', "G", "-", where "-" denotes a gap in the alignment. As the CoT embedding layer recognizes only symbols corresponding to amino acids and not nucleotides, we assign each type of nucleotide in the RNA MSA to an amino acid symbol. For example, we could take a random translation from "A", "U", "C"', "G" to "H", "E", "T"', "L", to get the following translation: "A" (Adenine) $\rightarrow$ "H" (Histdine), "U" (Uracil) $\rightarrow$ "E" (Glutamic Acid), "C" (Cytosine) $\rightarrow$ "T" (Threonine), "G" (Guanine) $\rightarrow$ "L" (Leucine).

As we show in our experiments, a random translation between nucleotide and amino acid symbols would be sufficient for adapting the protein contact prediction model, CoT, to RNA contact prediction. However, as discussed in Sect. 4.2, smarter translations, which can be manually devised or learned, could result in a better performance for our adapted RNA contacted prediction model.

The Learnable Model. The CoT model has six consecutive attention blocks and one refinement block, each outputting a $L \times L \times C$ attentive feature map, where C is a hyper-parameter corresponding to the number of features being learned by the model. In the original implementation of the protein CoT, C is set to 96, and the output of each attention block (i.e., the feature map for that block) is fed into the next block (see the "Adapted Feature Extraction" row in Fig. 2).

For our RNA contact prediction, we further attach four layers of 2D convolution (Conv2d) modules to the intermediate feature map outputs for each of the seven attention blocks described above (see the "Feature Fusion" row in Fig. 2). We concatenate the output of the Conv2d modules for each of the seven attention blocks into one $L \times L \times (C \times 7)$ tensor and finally pass it to a classifier module with 3 Conv2d layers for contact prediction (see the "classification" row in Fig. 2). The output of our model has shape $L \times L \times 37$, i.e., the distance between pairs of nucleotides is divided into 37 bins. Our model is trained using standard cross-entropy loss with the bins as labels. The summed probability value of the bins for a distance less than 10Å is used as the final contact prediction.

The complete architecture of our model is visualized in Fig. 2.

3.2 Dataset

We use a publicly-available well-curated RNA dataset [42]. We use the provided data split for training, validation, and testing. `RNA_DATASET` is used for training (and validation) and `RNA_TESTSET` is for testing. We removed 3 RNAs (RF02540, RF01998 and RF02012) whose sequences are too long for CoT. In total, we have 56 RNAs for training and validation and 23 RNAs for testing, all from different RNA families. We set the maximum number of homology sequences in MSA to be 200, based on the limits of our GPU memory. This constraint can be alleviated if large GPUs are available.

To avoid overfitting, we split the dataset into training and validation sets, allowing us to select models that are less prone to overfitting. We provide the validation splits in Table 1. Most experiments are carried out using Validation Set 1. Additionally, we average the results of the four different runs using all validation splits and report the means and standard deviations in Table 2.

Table 1. Validation set partitions.

Validation Set 1	Validation Set 2	Validation Set 3	Validation Set 4
RF01510	RF01826	RF00442_1	RF00921
RF01689	RF01831_1	RF00458	RF01051
RF01725	RF01852	RF00504	RF01054
RF01734	RF01854	RF00606_1	RF01300
RF01750	RF01982	RF01750	RF01415
RF01763	RF02001_2	RF01763	RF01510
RF01767	RF02266	RF01767	RF01689
RF01786	RF02447	RF01786	RF01725
RF01807	RF02553	RF01807	RF01734

3.3 Baseline Methods

We compare our method to several representative MSA-based methods: (1) Unsupervised methods: mfDCA and plmDCA, using the implementations from `pydca` [41], and PSICOV [17], and PLMC [14] (2) Supervised method: CoCoNet.

Note that all these baselines are variants of DCA or are based on DCA (the current trend in studying RNA contacts). Our proposed method does not rely on DCA and approaches the problem from a different angle through the transfer learning of learned knowledge from a pre-trained protein contact prediction model.

3.4 Training Details

We use the Adam optimizer and cosine anneal learning rate scheduler with an initial learning rate of $1e^{-3}$. We train on an RTX-A6000 GPU using PyTorch-1.8 and CUDA-11 and search the hyper-parameters for the total training epochs among $\{100, 300, 500\}$ and batch sizes among $\{4, 8, 12, 16\}$. The code is available at https://github.com/yiren-jian/CoT-RNA-Transfer.

We randomly divide the 56 RNAs reserved for training into 47 RNAs for training and 9 RNAs for validation and use the given 23 RNAs in the test dataset for testing. The best-validated model during the training is used for testing.

3.5 Evaluation Metrics

Following the standard protocol of prior works [16,33], we evaluate the precision on each RNA sequence of length L with top-L predictions of each method (PPV_L). We also report results for $\text{PPV}_{0.3/0.5L}$.

4 Results

Unless specified otherwise, the results presented in this manuscript employ translation nucleotides to amino acids (AUCG $\rightarrow$ HETL) as detailed in Sect. 3.1.

4.1 Main Results

We first compare our method to those only using MSA as input to evaluate the contribution of the pre-trained protein Transformer to RNA contact prediction. Unsupervised algorithms like mfDCA, plmDCA, PSICOV, and PLMC are based on covariance analysis. CoCoNet uses DCA output as input and ground truth contact maps to train a supervised ConvNet classifier. We compare six CoCoNet configurations [42].

Table 2 shows supervised models significantly outperform unsupervised baselines. Our transfer learning-based model outperforms the best CoCoNet configuration by an absolute of 5.0, 7.4, and 7.8 for PPVL, PPV0.5L, and $\text{PPV}_{0.3L}$, respectively.

CoCoNet is designed to be shallow, with a few parameters to learn, given the very limited number of available RNA contact prediction training data and features (from DCA). In contrast, by transfer learning of CoT, a large pre-trained protein contact prediction model, our model is much larger and deeper and can learn more diverse features through the multi-layer attentions of CoT contain.

While CoCoNet takes DCA as input which is a tensor of $1 \times L \times L$ (L being the RNA sequence length), our method, leveraging multi-layer CoT, has diverse attentive features of shape $(7 \times 96) \times L \times L$ (There are 7 layers in CoT and each layer outputs 96 channels/features). These diverse features allow us to learn deeper and larger models that can better generalize. In Sect. 4.2, we further investigate this, showing that our model indeed benefits from increasing the number of parameters in the transfer modules.

Table 2. Comparison of different RNA contact prediction methods based on MSA. §: Using the publicly released parameters [42], which are trained using our training and validation sets. ¶: Using prior knowledge of Watson-Crick pairs is used. †: Our models trained using only the training set, selected based on the best validation and evaluated on the testing set. ‡: We repeat the experiments four times using different random training and validation splits and report the mean and standard deviation of the results on the test dataset.

Method	PPV_L	$PPV_{0.5L}$	$PPV_{0.3L}$
mfDCA	34.1	46.7	57.4
plmDCA	30.6	43.2	57.8
PSICOV	32.1	43.8	57.8
PLMC	33.5	45.9	57.4
CoCoNet§ $_{(3\times3)}$	61.6	67.7	69.1
CoCoNet§ $_{(5\times5)}$	61.8	65.2	67.8
CoCoNet§ $_{(7\times7)}$	62.4	66.6	69.2
CoCoNet§¶ $_{(3\times3)\times2}$	67.1	71.6	72.3
CoCoNet§¶ $_{(5\times5)\times2}$	67.5	71.9	75.0
CoCoNet§¶ $_{(7\times7)\times2}$	68.5	73.2	75.2
CoT-RNA† (Ours)	**73.5**	**80.6**	**83.0**
CoT-RNA‡ (average)	**72.1** $_{\pm1.1}$	**79.2** $_{\pm1.7}$	**81.9** $_{\pm2.2}$

4.2 Ablation Studies

Here, we examine the design choices of our model for transfer learning by exploring different configurations.

Common Transfer Learning Strategies. We investigate three common transfer learning strategies (see Fig. 3 for an overview) and show that they are not well-suited for this task:

Using CoT directly (CoT directly). By adapting the embedding to map RNA nucleotides to protein amino acids, the pre-trained CoT can directly output a prediction of a distance map for RNA contact prediction without any model modifications.

Fine-tuning the classification block of CoT (CoT cls fine-tuned). A typical approach in transfer learning is to fine-tune the last few layers of a model. CoT has six attention blocks followed by a final ResNet block for prediction. We attach a new classification block while keeping others fixed.

Fine-tuning the entire CoT end-to-end (CoT end-to-end). Another common protocol for transfer learning is to fine-tune the entire pre-trained model end-to-end. We update all parameters in the pre-trained protein CoT by the RNA training set.

Table 3 shows the performance of these methods on our dataset. With a PPV_L of 30.4, the direct use of CoT (CoT directly) without any learning is

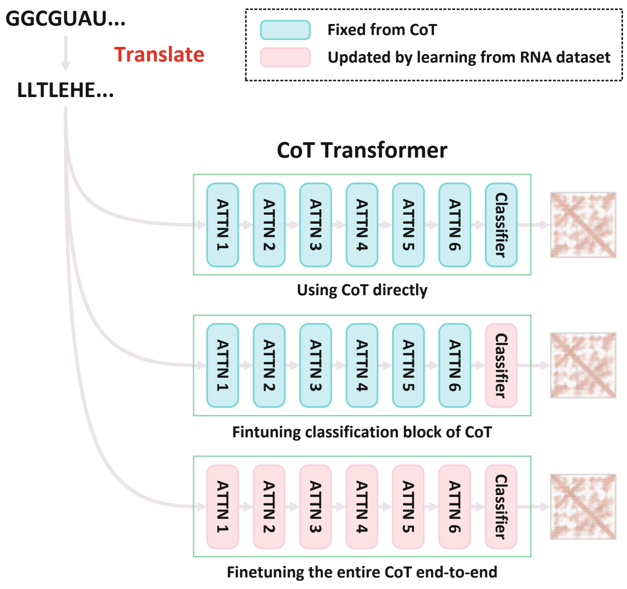


Fig. 3. Common baselines for transferring protein CoT to RNA contact prediction.

shown to be inefficient. This suggests that learned protein knowledge by itself cannot be successfully transferred to RNA tasks without some fine-tuning. The results for transfer learning through fine-tuning the classification block of CoT (CoT cls fine-tuned) are considerably better, being competitive with the mfDCA baseline. These results suggest that tuning the attention features in the last layer of CoT enables the transfer of knowledge to the RNA tasks to some extent. However, as this configuration ignores the attention features from the other layers, it performs significantly worse than our method, suggesting that these features also play an important role in contact prediction and need to be tuned for RNA contact prediction. Finally, the end-to-end model, which updates all the parameters in CoT (CoT end-to-end), performs similarly to the last variant. Though this model does not ignore any part of the CoT, it requires the tuning of ∼20M parameters. With only 56 RNA training points, the model is likely to over-fit.

From these experiments, we can conclude that the effective transfer of protein CoT to the RNA contact prediction task requires (1) adapting some of the parameters of CoT to the new task, (2) leveraging the multi-layer attention features from CoT, and (3) making the number of learnable parameters "small" and proportional to the size of the training set for the new task. These findings lead to our final model design described in Sect. 3.1 that leverages multi-layer attention features from CoT and learns an appropriate number of parameters for our small training set.

Table 3. Common transfer learning strategies applied to CoT.

Method	PPV_L	$\text{PPV}_{0.5L}$	$\text{PPV}_{0.3L}$
mfDCA (baseline)	34.1	46.7	57.4
CoT directly	30.4	33.1	34.0
CoT cls fine-tuned	38.3	41.6	41.2
CoT end-to-end	36.2	43.2	46.6
Ours	**73.5**	**80.6**	**83.0**

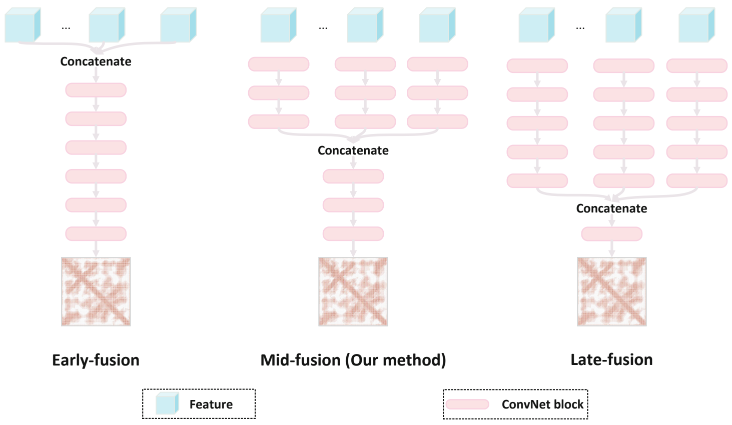


Fig. 4. Different feature fusion strategies. Our final model (shown in Fig. 2) uses the *mid-fusion* design.

We examine various strategies for combining attention features from different CoT layers/blocks. Our model, as shown in Fig. 2, employs multi-branch networks (each with 4 ConvNet layers) followed by a shared 3-layer ConvNet classification block. Each branch network separately processes the attention features of CoT at each layer, before being fused and passed into the classification block (termed *mid-fusion* design). Other designs include *early-fusion* and *late-fusion*. Early-fusion concatenates all CoT features from different layers and processes them using a single shared network. Late-fusion has separate branch networks for each CoT layer's features before being merged at the very end, followed by a single classification layer. Figure 4 provides a schematic diagram of these three fusion strategies.

To make the comparison of these three designs fair, we modify the number of channels in each layer so that the three models have a similar number of parameters. As shown in Table 4, while all design choices work well, *mid-fusion* has the best performance. It is possible that features from different layers contain different types of information that may need to be processed by different "expert" models (i.e., ConvNet branches in our model), making an early-fusion model inefficient. In both *mid-fusion* and *late-fusion*, each branch network will process

Table 4. Comparison of different feature fusion designs. We modify the number of channels in each layer so that all three models have a similar number of parameters for fair comparison.

Method	PPV_L	$\mathrm{PPV}_{0.5L}$	$\mathrm{PPV}_{0.3L}$
early-fusion	71.8	79.5	82.4
mid-fusion (Ours)	**73.5**	**80.6**	**83.0**
late-fusion	72.0	77.7	79.4

the attention features of each layer separately, with *late-fusion* having a relatively smaller classification head. The overall better performance of *mid-fusion* suggests that a good design choice is to have a balanced distribution of parameters into the branch networks and the classification head.

Different Model Sizes. As discussed in Sect. 4.1, the deep-CoCoNet variant of CoCoNet under-performs compared to the shallower original CoCoNet, likely due to the limited expressiveness of input DCA features which are single channel with a shape of $[1 \times L \times L]$. In contrast, here we demonstrate that our transferred CoT model allows for learning deeper networks, with its performance improving as we increase the parameters in the transfer modules.

We create larger and smaller versions of our model by increasing and decreasing the number of channels in each layer, respectively. As shown in Table 5, the larger models outperform smaller ones, possibly due to the expressiveness of the CoT features from the 7 different attention blocks.

Table 5. Comparison of our model with different sizes. While maintaining the network structures, we vary the number of channels in each layer so that we end up with models with different numbers of parameters.

Method	PPV_L	$\mathrm{PPV}_{0.5L}$	$\mathrm{PPV}_{0.3L}$
Ours (small)	71.1	76.0	79.0
Ours	73.5	80.6	83.0
Ours (large)	**78.3**	**82.3**	**86.1**

Table 6. Results of different translations/translations from nucleotides to amino acids of our transferred CoT. Bold corresponds to the best-performing translation; underline corresponds to the main translation used in the experiments.

Method	PPV_L	$\mathrm{PPV}_{0.5L}$	$\mathrm{PPV}_{0.3L}$
AUCG $\rightarrow$ ACDE	68.6	76.9	82.5
AUCG $\rightarrow$ HETL	$\underline{73.5}$	$\underline{80.6}$	$\underline{83.0}$
AUCG $\rightarrow$ RDSY	76.1	81.4	83.4
AUCG $\rightarrow$ KDNY	**77.3**	**84.5**	**88.6**

Table 7. esults of different translations/mappings from nucleotides to amino acids of our transferred CoT. **A.** We experiment with an additional set of mappings. While performances vary, they are all competitive over the baseline CoCoNet. **B.** We explore different mappings using permutations of KDNY, which is shown to a good mapping. **C.** We explore different mappings using permutations of RDSY, which is shown to a good mapping. **D.** We explore different mappings using permutations of ACDE, which is shown to be less competitive.

Method	PPV_L	$\text{PPV}_{0.5L}$	$\text{PPV}_{0.3L}$
A. Different RNA-to-protein mappings			
AUCG $\rightarrow$ AVIL	66.9	76.3	80.8
AUCG $\rightarrow$ ILMF	67.0	74.0	78.0
AUCG $\rightarrow$ YDKS	68.6	77.6	82.3
AUCG $\rightarrow$ RDSC	72.3	81.7	82.7
AUCG $\rightarrow$ KDSU	73.9	80.4	81.4
AUCG $\rightarrow$ HETG	75.1	81.2	84.4
AUCG $\rightarrow$ HDTU	76.2	83.2	86.8
AUCG $\rightarrow$ KEQG	76.9	80.9	81.4
AUCG $\rightarrow$ KDTI	76.9	83.6	85.8
AUCG $\rightarrow$ KDQG	77.3	84.8	87.7
AUCG $\rightarrow$ KDNU	78.2	86.2	88.0
AUCG $\rightarrow$ RSKY	78.6	84.3	87.7
AUCG $\rightarrow$ SDYK	79.8	85.5	87.7
B. Different permutations of KDNY			
AUCG $\rightarrow$ KYND	74.0	79.8	83.6
AUCG $\rightarrow$ DNYK	77.0	82.7	85.7
AUCG $\rightarrow$ NYKD	78.6	83.7	84.5
AUCG $\rightarrow$ YNDK	80.6	85.2	86.2
C. Different permutations of RDSY			
AUCG $\rightarrow$ RYSD	77.2	82.3	85.9
AUCG $\rightarrow$ YRDS	78.9	86.4	90.7
AUCG $\rightarrow$ DRSY	80.3	85.4	87.1
AUCG $\rightarrow$ SDYR	81.4	87.2	88.6
D. Different permutations of ACED			
AUCG $\rightarrow$ EDAC	64.8	70.3	71.7
AUCG $\rightarrow$ CAED	65.3	72.8	76.0
AUCG $\rightarrow$ DECA	66.2	76.7	82.7
AUCG $\rightarrow$ ACDE	68.6	76.9	82.5

Protein to RNA Translation Variations. We have used a random translation from RNA nucleotides to protein amino acids (e.g., "AUCG" to "HETL") in our experiments. Here, we study the effects of different translations on our model's performance.

The 20 amino acids can be categorized into four groups: (1) electrically charged, (2) polar uncharged, (3) hydrophobic, and (4) special cases. Randomly selecting one from each group generally works well (e.g., "AUCG" → "RDSY" and "AUCG" → "KDNY" in Table 6), indicating our framework's robustness to translation choices.

We also test possibly one of the worst translations, "AUCG" → ACDE", as it may generate unlikely amino acid chains (e.g., a string of negatively charged residues, as "D" and "E" are negatively charged) and hence CoT will have had limited exposure to such sequences during its pre-training. Though we see a relative performance drop, the results are still comparable to CoCoNet.

A learnable 4×20 nucleotide-to-amino acid embedding could yield better results but faces implementation challenges, such as requiring powerful GPUs and adapting the original CoT model's separate binary executable embedding layer to the PyTorch framework.

We provide additional experiments using different protein/RNA mappings in Table 7. Given the approximately 20^4 distinct nucleotide-to-amino acid translations, exhaustively examining all combinations is not feasible. Instead, we evaluate our method using an alternative set of randomly chosen translations. As demonstrated in Table 7, our method exhibits considerable robustness with respect to the selection of different translations. While selecting different amino acids (e.g., DNYK or ACDE) may lead to varying performance, permutations of a set of amino acids generally yield similar results. For example, AUCG → DNYK, KYND, NYKD, YNDK all perform well, whereas AUCG → ACDE, CAED, DECA, EDAC underperform. Based on these experiments, we hypothesize that the transferable contact patterns from the protein transformer CoT to RNA tasks may not rely on an exact translation. It is likely that the rich attention information learned in CoT is preserved across specific sets of amino acids.

5 Discussion and Conclusion

We demonstrate the effectiveness of transferring CoT, a pre-trained protein Transformer model for contact prediction, to the RNA contact prediction task using a small curated RNA dataset. Unlike hybrid methods, our approach does not use additional features extracted by RNA analysis tools (e.g., RNAcontact). Incorporating CoT features and RNA features (extracted by tools like RNAcontact) could potentially improve our method's performance.

Note that even though our method uses a Transformer-based architecture that models attention between every element in a sequence, the maximum sequence length is typically limited to a few hundred elements due to computational constraints. Additionally, we sample only 200 homologous sequences from the multiple sequence alignment (MSA) of an RNA as input. This sampling process may result in the loss of co-evolutionary information, potentially limiting the learning capacity of CoT. Furthermore, instead of utilizing a "manual translation" of CoT with a translation of AUCG → HETL, a "soft-learnable translation" from RNA to proteins (a 4×20 matrix) could potentially yield improved results. Nevertheless, as discussed in Sect. 4.2, implementing such an approach currently faces several engineering challenges.

Our findings shed light on a compelling representation transfer problem in computational structural biology; specifically, we investigate if structural patterns learned from large-scale protein datasets can be transferred to data-scarce RNA problems, particularly for structural contact predictions. Our results indicate that protein-to-RNA transfer learning can improve RNA model performance, suggesting that other pre-trained protein Transformers, such as MSA-Transformer [29] and ESM [28], could potentially be transferred to RNA for other downstream tasks.

References

1. Aitazaz, T., Tubaishat, A., Al-Obeidat, F., Shah, B., Zia, T., Tariq, A.: Transfer learning for histopathology images: an empirical study. Neural Comput. Appl. **35**(11), 7963–7974 (2023)
2. Bao, H., Wang, W., Dong, L., Liu, Q., Mohammed, O.K., Aggarwal, K., Som, S., Piao, S., Wei, F.: Vlmo: Unified vision-language pre-training with mixture-of-modality-experts. Adv. Neural. Inf. Process. Syst. **35**, 32897–32912 (2022)
3. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The protein data bank. Nucleic Acids Res. **28**(1), 235–242 (2000)
4. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Adv. Neural. Inf. Process. Syst. **33**, 1877–1901 (2020)
5. Cavalli, A., Salvatella, X., Dobson, C.M., Vendruscolo, M.: Protein structure determination from nmr chemical shifts. Proc. Natl. Acad. Sci. **104**(23), 9615–9620 (2007)
6. Dauparas, J., Anishchenko, I., Bennett, N., Bai, H., Ragotte, R.J., Milles, L.F., Wicky, B.I., Courbet, A., de Haas, R.J., Bethel, N., et al.: Robust deep learning-based protein sequence design using proteinmpnn. Science **378**(6615), 49–56 (2022)
7. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. In: International conference on machine learning. pp. 647–655. PMLR (2014)
8. Ekeberg, M., Lövkvist, C., Lan, Y., Weigt, M., Aurell, E.: Improved contact prediction in proteins: using pseudolikelihoods to infer potts models. Phys. Rev. E **87**(1), 012707 (2013)
9. Esteller, M.: Non-coding rnas in human disease. Nat. Rev. Genet. **12**(12), 861–874 (2011)
10. Fire, A., Xu, S., Montgomery, M.K., Kostas, S.A., Driver, S.E., Mello, C.C.: Potent and specific genetic interference by double-stranded rna in caenorhabditis elegans. Nature **391**(6669), 806–811 (1998)
11. Glaeser, R.M.: How good can cryo-em become? Nat. Methods **13**(1), 28–32 (2016)
12. Gogoulou, E., Ekgren, A., Isbister, T., Sahlgren, M.: Cross-lingual transfer of monolingual models. In: Proceedings of the Thirteenth Language Resources and Evaluation Conference. pp. 948–955. European Language Resources Association, Marseille, France (Jun 2022)

13. Goodarzi, H., Liu, X., Nguyen, H.C., Zhang, S., Fish, L., Tavazoie, S.F.: Endogenous trna-derived fragments suppress breast cancer progression via ybx1 displacement. Cell **161**(4), 790–802 (2015)
14. Hopf, T.A., Ingraham, J.B., Poelwijk, F.J., Schärfe, C.P., Springer, M., Sander, C., Marks, D.S.: Mutation effects predicted from sequence co-variation. Nat. Biotechnol. **35**(2), 128–135 (2017)
15. Jian, Y., Gao, C., Vosoughi, S.: Non-linguistic supervision for contrastive learning of sentence embeddings. In: Advances in Neural Information Processing Systems (2022)
16. Jian, Y., Wang, X., Qiu, J., Wang, H., Liu, Z., Zhao, Y., Zeng, C.: Direct: Rna contact predictions by integrating structural patterns. BMC Bioinformatics **20**(1), 1–12 (2019)
17. Jones, D.T., Buchan, D.W., Cozzetto, D., Pontil, M.: Psicov: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. Bioinformatics **28**(2), 184–190 (2012)
18. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al.: Highly accurate protein structure prediction with alphafold. Nature **596**(7873), 583–589 (2021)
19. Källberg, M., Wang, H., Wang, S., Peng, J., Wang, Z., Lu, H., Xu, J.: Template-based protein structure modeling using the raptorx web server. Nat. Protoc. **7**(8), 1511–1522 (2012)
20. Leaver-Fay, A., Tyka, M., Lewis, S.M., Lange, O.F., Thompson, J., Jacak, R., Kaufman, K.W., Renfrew, P.D., Smith, C.A., Sheffler, W., et al.: Rosetta3: an object-oriented software suite for the simulation and design of macromolecules. In: Methods in enzymology, vol. 487, pp. 545–574. Elsevier (2011)
21. Li, J., Selvaraju, R., Gotmare, A., Joty, S., Xiong, C., Hoi, S.C.H.: Align before fuse: Vision and language representation learning with momentum distillation. Adv. Neural. Inf. Process. Syst. **34**, 9694–9705 (2021)
22. Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al.: Evolutionary-scale prediction of atomic level protein structure with a language model. bioRxiv (2022)
23. Lu, K., Grover, A., Abbeel, P., Mordatch, I.: Pretrained transformers as universal computation engines. arXiv preprint arXiv:2103.05247 (2021)
24. Ma, H., Jia, X., Zhang, K., Su, Z.: Cryo-em advances in rna structure determination. Signal Transduct. Target. Ther. **7**(1), 1–6 (2022)
25. Morcos, F., Pagnani, A., Lunt, B., Bertolino, A., Marks, D.S., Sander, C., Zecchina, R., Onuchic, J.N., Hwa, T., Weigt, M.: Direct-coupling analysis of residue coevolution captures native contacts across many protein families. Proc. Natl. Acad. Sci. **108**(49), E1293–E1301 (2011)
26. Muntoni, A.P., Pagnani, A., Weigt, M., Zamponi, F.: adabmdca: adaptive boltzmann machine learning for biological sequences. BMC Bioinformatics **22**(1), 1–19 (2021)
27. Papadimitriou, I., Jurafsky, D.: Learning music helps you read: Using transfer to study linguistic structure in language models. In: EMNLP. pp. 6829–6839 (01 2020). https://doi.org/10.18653/v1/2020.emnlp-main.554
28. Rao, R., Meier, J., Sercu, T., Ovchinnikov, S., Rives, A.: Transformer protein language models are unsupervised structure learners. In: International Conference on Learning Representations (2021)
29. Rao, R.M., Liu, J., Verkuil, R., Meier, J., Canny, J., Abbeel, P., Sercu, T., Rives, A.: Msa transformer. In: International Conference on Machine Learning. pp. 8844–8856. PMLR (2021)

30. Sharma, U., Conine, C.C., Shea, J.M., Boskovic, A., Derr, A.G., Bing, X.Y., Belleannee, C., Kucukural, A., Serra, R.W., Sun, F., et al.: Biogenesis and function of trna fragments during sperm maturation and fertilization in mammals. Science **351**(6271), 391–396 (2016)
31. Shi, M., Lin, X.D., Tian, J.H., Chen, L.J., Chen, X., Li, C.X., Qin, X.C., Li, J., Cao, J.P., Eden, J.S., et al.: Redefining the invertebrate rna virosphere. Nature **540**(7634), 539–543 (2016)
32. Stubbs, G., Warren, S., Holmes, K.: Structure of rna and rna binding site in tobacco mosaic virus from 4-å map calculated from x-ray fibre diagrams. Nature **267**(5608), 216–221 (1977)
33. Sun, S., Wang, W., Peng, Z., Yang, J.: Rna inter-nucleotide 3d closeness prediction by deep residual neural networks. Bioinformatics **37**(8), 1093–1098 (2021)
34. Taubert, O., von der Lehr, F., Bazarova, A., Faber, C., Knechtges, P., Weiel, M., Debus, C., Coquelin, D., Basermann, A., Streit, A., et al.: Rna contact prediction by data efficient deep learning. Communications Biology **6**(1), 913 (2023)
35. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
36. Wang, J., Yang, Z., Hu, X., Li, L., Lin, K., Gan, Z., Liu, Z., Liu, C., Wang, L.: GIT: A generative image-to-text transformer for vision and language. Transactions on Machine Learning Research (2022), https://openreview.net/forum?id=b4tMhpN0JC
37. Wang, J., Wang, J., Huang, Y., Xiao, Y.: 3drna v2. 0: An updated web server for rna 3d structure prediction. International Journal of Molecular Sciences **20**(17), 4116 (2019)
38. Weigt, M., White, R.A., Szurmant, H., Hoch, J.A., Hwa, T.: Identification of direct residue contacts in protein-protein interaction by message passing. Proc. Natl. Acad. Sci. **106**(1), 67–72 (2009)
39. Yan, H., Li, Z., Li, W., Wang, C., Wu, M., Zhang, C.: Contnet: Why not use convolution and transformer at the same time? arXiv preprint arXiv:2104.13497 (2021)
40. Yang, J., Anishchenko, I., Park, H., Peng, Z., Ovchinnikov, S., Baker, D.: Improved protein structure prediction using predicted interresidue orientations. Proc. Natl. Acad. Sci. **117**(3), 1496–1503 (2020)
41. Zerihun, M.B., Pucci, F., Peter, E.K., Schug, A.: pydca v1. 0: a comprehensive software for direct coupling analysis of rna and protein sequences. Bioinformatics **36**(7), 2264–2265 (2020)
42. Zerihun, M.B., Pucci, F., Schug, A.: Coconet–boosting rna contact prediction by convolutional neural networks. Nucleic Acids Res. **49**(22), 12661–12672 (2021)
43. Zhang, H., Ju, F., Zhu, J., He, L., Shao, B., Zheng, N., Liu, T.Y.: Co-evolution transformer for protein contact prediction. Adv. Neural. Inf. Process. Syst. **34**, 14252–14263 (2021)
44. Zhou, H.Y., Lu, C., Yang, S., Yu, Y.: Convnets vs. transformers: Whose visual representations are more transferable? In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2230–2238 (2021)

Improving Multi-label Recognition using Class Co-Occurrence Probabilities

Samyak Rawlekar[1](✉), Shubhang Bhatnagar[1], Vishnuvardhan Pogunulu Srinivasulu[2], and Narendra Ahuja[1]

[1] University of Illinois Urbana-Champaign, Urbana, IL, USA
{samyakr2,sb56,n-ahuja}@illinois.edu
[2] Vizzhy, Austin, USA
vishnu@vizzhy.com

Abstract. Multi-label Recognition (MLR) involves the identification of multiple objects within an image. To address the additional complexity of this problem, recent works have leveraged information from vision-language models (VLMs) trained on large text-images datasets for the task. These methods learn an independent classifier for each object (class), overlooking correlations in their occurrences. Such co-occurrences can be captured from the training data as conditional probabilities between a pair of classes. We propose a framework to extend the independent classifiers by incorporating the co-occurrence information for object pairs to improve the performance of independent classifiers. We use a Graph Convolutional Network (GCN) to enforce the conditional probabilities between classes, by refining the initial estimates derived from image and text sources obtained using VLMs. We validate our method on four MLR datasets, where our approach outperforms all state-of-the-art methods.

Keywords: Multi-label Recognition · Graph Convolution Networks · Vision-Language Models

1 Introduction

Multi-label recognition (MLR) involves identifying each of the multiple classes from which objects are present in an image. It has many applications such as identifying all different: diseases evident in a chest x-ray [22], products in a query image for e-commerce [5], and food items in a plate for diet monitoring systems [2,36]. MLR is more challenging than classifying images having a single object [15,18] because an image may contain combinatorially large mix of classes,

S. Rawlekar and S. Bhatnagar—Equal contribution.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78192-6_28.

for which learning would require exponentially larger number of images ($O(2^N)$ images for N classes) than for Single label recognition (SLR). The objects may also occur in different layouts so recognition either requires object segmentation and recognition of each segmented object independently, or recognizing the object mix from the features measured over the entire image.

Several approaches [9,41] have taken the latter approach, namely recognition from image level features. Further, these approaches also reduce complexity by recognizing the presence of each object in an image independent of the others that may also be present. Training here amounts to learning independent classifiers for each object, which are then used to detect the corresponding objects, both using image level features. Apart from not segmenting the image and features into those for different objects, these methods also neglect the evidence for the presence of an object provided by the context of other objects. In practice, many objects are in sets, making their occurrences interdependent. Using independent classifiers neglects the mutual information present, which, if used, could enhance the performance of individual classifiers. This is particularly important in view of the already larger amount of data needed for MLR, and the relatively small sizes (vs SLR) of available annotated MLR datasets (because multiple classes present in an image require more annotation effort than required for SL images).

To mitigate the paucity of labeled data, recent MLR approaches [17,41] have focused on adapting large Vision Language Models (VLMs) e.g. CLIP [39], ALIGN [24], OpenCLIP [23] for the task. Instead of finetuning the VLM on small MLR datasets, these approaches [17,41] rely on learning text prompts associated with each class; the prompts associated with a class are learned by maximizing/minimizing the similarities of their embeddings with those of the embeddings extracted from images containing objects of the class. The additional information captured by the prompts constrains possible matches between the data and the desired labels, in turn limiting the number of parameters needed to be learned. This helps mitigate overfitting on small datasets. At test time, whether a class is present in a given image is ascertained by measuring the similarity of the image embeddings with embeddings extracted from previously learned positive/negative prompts for the class. However, the learning of prompts here is done independently for each class, again neglecting the class co-occurrences.

We propose a two-stage framework that leverages the knowledge of VLMs but injects the co-occurrence information into the models. We obtain an initial estimate of the evidence logits for each class in a subimage in terms of the match between the embeddings of the subimage and those of the positive and negative text prompts associated with the class. Results from the subimages are aggregated to obtain logits for all classes across the image. These independently extracted subimage logits are refined to enforce prior knowledge of the joint probabilities of pairs of classes present in different subimages. Given that a class is present in a training image, the conditional probabilities of other classes being also present are measured from the frequencies of observing those classes in the image. Since the prior probability of a class is known from the VLM output, we can combine it with the conditional probabilities of other classes to obtain joint probabilities of class pairs. Enhancement of the outputs of the independent classifiers by using conditional probabilities is carried out through a graph convolutional network (GCN). GCN thus utilizes the conditional probabilities to improve upon the outputs of the independently obtained class estimates using the vision-language model. As commonly done, we compensate for the differences in the frequencies of different classes occurring in the training images by reweighting the estimated probabilities of different classes to remove the class bias in the training data.

We test our approach on four benchmarks: MS-COCO-small (5% of the training set), PASCAL VOC, FoodSeg103, and UNIMIB-2016. The first two are commonly used for MLR, whereas we have two additional datasets that have been used in other contexts [10,44]. The number of images in each of these datasets is small compared to even many SLR datasets, although MLR calls for larger datasets. This makes MLR here even harder. Our experiments show that the use of inter-class influence in the second stage of our framework significantly improves performance over the state-of-the-art methods, which detect each class independently. As expected, the advantage is higher for classes for which VLM yields low accuracy but which frequently co-occur with other classes for which VLM accuracy is higher. We also show that our loss re-weighing greatly improves the performance on datasets where there is a significant class bias.

Our contributions:

– We propose a two-stage framework to adapt VLMs for MLR with limited annotated data, by enhancing the VLM based independent class estimates obtained in the first stage, with conditional probability priors extracted from the dataset, using a GCN.
– We validate our algorithm quantitatively using mean average precision (mAP) on four MLR datasets. Our method surpasses the previous SOTA MLR approaches by more than 2% (COCO-14-small), 0.4% (VOC-2007), 3.9% (FoodSeg103) and 11% (UNIMIB2016).

2 Related Works

2.1 Multi-Label Recognition

Multi-label recognition is an important, well-studied problem in computer vision, with a wide range of approaches being proposed to tackle it [32]. An important line of work has focused on learning disjoint binary classifiers for identifying each class of objects in an image [9,11,31,37]. These approaches require large labeled datasets for training. They make no use of information that can be derived from modeling the co-occurrence of different classes, which is especially important when only a small amount of annotated data is available for training.

Other works have proposed to use recurrent neural networks (RNNs) to model label dependencies in an image [30,42,48]. Specifically, they cast MLR into a sequence prediction problem, using beam search to find a sequence of objects having the highest likelihood of being present in the image. Like our method, these methods also model label dependencies, but do so implicitly via the hidden state of RNNs. This requires vast amounts of labeled data to learn them. Recent approaches for MLR have proposed the use of VLMs to improve MLR performance when only a limited amount of labeled data is available.

2.2 Vision-Language Models for MLR

VLMs learn representations that are transferable to a wide range of downstream tasks such as classification [21,43,47,51], retrieval [3,26], and segmentation[45,49,50] by aligning hundreds of millions of image-text(prompts) pairs. Such approaches commonly focus on learning prompts suitable for these downstream tasks [53]. [41] adapts VLMs for MLR, proposing to learn a pair of prompts associated with the presence/ absence of each class. Text embeddings extracted from the learned prompts are used to gather local evidence from the image features extracted by the VLM, which is then aggregated and combined. On the other hand, SCPNet [17] is another VLM based approach for MLR that uses a GCN to learn prompt embeddings, where the GCN helps model priors derived from class name similarities (derived from CLIP's embedding space). They augment the training with a self supervised contrastive loss. Both these methods learn independent classifiers, not modeling any co-occurrence information. In contrast, our method learns inter-dependent classifiers by using a GCN to model conditional probabilities (co-occurrence of actual training dataset), enhancing the VLM's initial independent prediction.

2.3 Long tailed Learning

The distribution of frequency with which objects belong to a class in real-world images often follows a long-tailed distribution[13,25,34,35]. Networks trained for multi-class classification on such data tend to perform poorly on the tail classes which have less data available. Several approaches have been proposed to mitigate this issue including data augmentation (augment the tail classes) [46], data re-sampling (sample images to obtain balanced distribution) [6,19,33,52], adjusting classifier margins (classification thresholds vary for every class)[4,27] and loss re-weighing [14,38] being popular. We use loss re-weighing to mitigate the effect of label imbalance on our method when it is trained to model label conditional probabilities.

3 Method

Suppose in a given set of images, $\mathcal{D} = \{\mathbf{x}_i\}, i \in \{1 \dots |\mathcal{D}|\}$, every image $\mathbf{x_i}$ may contain objects from up to N classes. The image is thus associated with N labels $\mathbf{y_i} \in \{0, 1\}^N$ where y_i^j denotes the presence or absence (1 or 0) of the j-th class in the image. Then the MLR problem requires identification of all labels associated with any input image.

Our approach in this paper uses a VLM g_{ϕ}, parameterized using weights ϕ (a pair of encoders $g_{\phi,img}$, $g_{\phi,text}$). VLMs are pretrained to align image and textual features over large datasets to learn features suited for various tasks/domains. As mentioned in Sec. 1, these models associate a pair of positive and negative text prompts $\{\mathbf{t_{j,+}}, \mathbf{t_{j,-}}\}$ with each class j (complete set denoted by ψ). A text encoder $g_{\phi,text}$ extracts text embeddings from each of these prompts, gives them to an image-text feature aggregation head p, which matches visual features extracted from different subimages with the text embeddings, and combines them to obtain an initial set of logits for each class in the image. We then use a GCN f_{θ} with weights θ to refine the logits output by p, by leveraging the statistical co-occurrence of classes observed in the training dataset. An overview of our proposed method is given in Figure 1.

The following subsections present the various parts of our method.

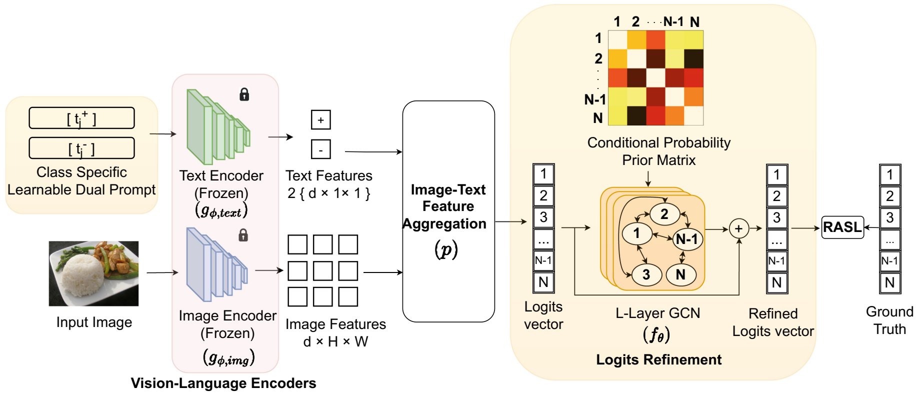

[t_j^+]
[t_j^-]
Class Specific Learnable Dual Prompt
Text Encoder (Frozen) ($g_{\phi,text}$)
Text Features 2 { d × 1× 1 }
Input Image
Image Encoder (Frozen) ($g_{\phi,img}$)
Image Features d × H × W
Vision-Language Encoders
Image-Text Feature Aggregation (p)
Logits vector
1 2 ··· N-1 N
Conditional Probability Prior Matrix
L-Layer GCN (f_θ)
Refined Logits vector
RASL
Ground Truth
Logits Refinement

Fig. 1. Method Overview: Given an image with multiple objects, we extract image features and text features from the subimages using a vision-language model (CLIP). An image-text feature aggregation module (Sec. 3.1) combines these features to identify all classes present in the image as a union of the classes present in the subimages, giving an initial set of image level class logits. These logits are passed to a GCN, that uses conditional probabilities between classes to refine these initial predictions (Sec. 3.2). We train this framework while reweighting the loss generated by classes to address any class imbalance in the training data using a Reweighted Asymmetric Loss (RASL), a weighted version of the familiar ASL.

3.1 Initial Logits Estimation

We use a VLM as a feature extractor and initial classifier for our method. The VLM's image encoder does spatial pooling of windows in the final layer to obtain a single d dimensional feature vector for a single-label image x_i. This is not suitable for MLR case as spatial pooling operation combines the features of multiple objects in different regions of the image, with overall features being dominated by those extracted from a single object. We remove the pooling layer of the image encoder, using it to get features $g_{\phi,img}(\mathbf{x_i}) = \mathbf{z_i}$ (of shape $d \times H \times W$) for a given image x_i, hence preserving information from the individual windows.

Our image-text feature aggregation head p is similar to [41]. For each class j, it learns a pair of text prompts $\{\mathbf{t_{j,+}}, \mathbf{t_{j,-}}\}$, which are projected to d-dimensional embeddings $\mathbf{r_{j,+}}, \mathbf{r_{j,-}}$ using $g_{\phi,text}$. Cosine similarity of the d-dimensional image features at a particular point (h, w) with $\mathbf{r_{j,+}}$ indicates the presence of the class, while similarity with $\mathbf{r_{j,-}}$ indicates its absence. These similarities are aggregated and used by p to give logits $p(\mathbf{z_i})$ for the image.

3.2 Refining Logits using Conditional Prior

We refine the logits $p(\mathbf{z_i})$ using a GCN, which ensures conformity with the conditional probability priors extracted from the training dataset.

Estimating Label Conditional Probability Prior: To derive the label prior from the dataset, we first calculate the label co-occurrence matrix $C_{N \times N} = (c_{mn})$ over training dataset $\mathcal{D}$. Each entry c_{mn} is given by

$$c_{mn} = \sum_{i=0}^{|\mathcal{D}|} y_i^m \times y_i^n \tag{1}$$

The c_{mn} denotes the number of times that objects belonging to classes m and n occur together in an image. Using this, we calculate an estimate of conditional probability matrix $A_{N \times N} = (a_{mn})$, where each entry gives an estimate for the probability $P(y_i^n = 1 | y_i^m = 1)$:

$$a_{mn} = \hat{P}(y_i^n = 1 | y_i^m = 1) = \frac{c_{mn}}{c_{mm}} \tag{2}$$

GCN for Refinement: We refine the logits $p_\psi(\mathbf{z_i})$ using the GCN f_θ which uses the conditional probability matrix A to define the connection weights between its N nodes. Specifically, given that f_θ has L layers, each layer calculates:

$$H^l = \rho(AH^{l-1}W^l) \tag{3}$$

where H^{l-1} is the output vector of the previous layer and ρ is a non linearity (Leaky ReLU). H^0 is defined as the input logits $p_\psi(\mathbf{z_i})$. W^l are learnable weights for each layer. Using a GCN layer ensures that the logits are refined using information from only those nodes used for computing the logits, reducing the number of parameters learned while also taking advantage of the conditional probability estimates. After passing through multiple layers of the GCN, we get the updated predictions $f_\theta(p_\psi(\mathbf{z_i}))$. We add the initial logits to the updated logits to obtain our refined logits prediction $p_\psi(\mathbf{z_i}) + f_\theta(p_\psi(\mathbf{z_i}))$

3.3 Training

We train the image-text feature aggregation module p and the GCN f_θ, while freezing the VLM g_ϕ. We adopt the widely used Asymmetric Loss (ASL) [40], a modified version of the focal loss, to train our network for MLR.

ASL [40] addresses the inherent imbalance in MLR caused by the prevalence of negative examples compared to positive ones in training images. Similar to focal loss [28], ASL underweighs the loss term due to negative examples. However, it does so using two focusing parameters (γ_+ and γ_-) instead of one (γ) used by focal loss. However, ASL does not address the issue of sample imbalance, caused by some classes having fewer examples in the dataset. Towards this, we add a loss re-weighting term (α) to ASL. Our re-weighed ASL (RASL) is defined as:

$$\mathcal{L}_{RASL}(\hat{y}_i^j) = \begin{cases} \alpha_j \left(1 - \hat{y}_i^j\right)^{\gamma_+} \log\left(\hat{y}_i^j\right) & \text{when } y_i^j = 1 \\ \alpha_j \left(\hat{y}_{i,\delta}^j\right)^{\gamma_-} \log\left(1 - \hat{y}_{i,\delta}^j\right) & \text{when } y_i^j = 0 \end{cases} \tag{4}$$

where $\hat{y}_i^j$ represents the corresponding prediction associated with label y_i^j; $\hat{y}_{i,\delta}^j = \max(\hat{y} - \delta, 0)$, with δ representing the shifting parameter defined in ASL; and α_j is the re-weighting parameter for class j, defined as:

$$\alpha = \left(\frac{a_{jj}}{\sum_{j=1}^{N} a_{jj}}\right)^{-1} \tag{5}$$

Then the total loss over the dataset $|\mathcal{D}|$ is given by:

$$\mathcal{L}_{total} = \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{N} \mathcal{L}_{RASL}(\hat{y}_i^j) \tag{6}$$

4 Experiments

In this section, we discuss the datasets used, implementation details of our approach, evaluation metrics and provide a thorough analysis of our approach.

4.1 Datasets

We evaluate our method on four different MLR benchmarks: MS-COCO 2014-small and PASCAL VOC 2007, which are widely used MLR benchmarks, as well as FoodSeg103 and UNIMIB 2016, which are smaller MLR datasets suitable for testing in the low data regime. Details of these datasets are given below:
MS-COCO 2014-small: MS-COCO [29] is another popular MLR dataset and consists of 82,081 training images and 40,504 validation images with objects belonging to 80 classes. To evaluate our methods performance in the low data

Table 1. Comparison of results obtained by our method and the state-of-the-art baselines, on four MLR datasets in the low data regime: FoodSeg103, UNIMIB 2016, COCO-small (5% of COCO's training data) and VOC-2007. Our approach achieves the best performance on all metrics: per-class and overall average precisions (CP and OP), recalls (CR and OR), F1 scores (CF1 and OF1), and mean average precision (mAP). * indicates methods that fine-tune the complete backbone network.

Dataset	Method	CP	CR	CF1	OP	OR	OF1	mAP
COCO-small [29]	DualCoOp[41]	53.3	73.5	59.8	47.1	79.5	59.2	70.2
	SCPNet[17]	51.9	70.3	59.7	47.2	78.9	59.1	69.3
	Ours	**54.1**	**74.3**	**62.6**	**47.7**	**82.6**	**60.5**	**72.6**
VOC [20]	SSGRL*[8]	-	-	-	-	-	-	93.4
	GCN-ML*[9]	-	-	-	-	-	-	94
	KGGR*[7]	-	-	-	-	-	-	93.6
	DualCoOp[41]	81.1	93.3	86.5	83.5	94.1	88.5	94.0
	SCPNet[17]	68.9	91.6	76.8	68.5	93.5	79.1	87.4
	Ours	**81.1**	**94.1**	**87.1**	**83.6**	**94.5**	**88.6**	**94.4**
FoodSeg103 [44]	DualCoOp[41]	44.9	52.7	46.9	59.2	69.2	63.8	49.0
	SCPNet[17]	39.4	54.4	43.2	61.4	67.8	64.4	48.8
	Ours w/o reweigh	44.8	55.0	48.0	58.6	70.2	63.9	51.3
	Ours	**47.1**	**55.1**	**50.8**	**63.7**	**69.9**	**66.7**	**52.9**
UNIMIB [10]	DualCoOp[41]	46.9	54.7	48.4	69.0	79.0	73.7	58.1
	SCPNet[17]	50.5	52.9	49.9	69.6	78.4	73.8	60.0
	Ours w/o reweigh	52.6	59.6	53.8	73.5	83.3	78.1	64.4
	Ours	**66.8**	**65.8**	**64.2**	**80.9**	**86.1**	**83.4**	**72.2**

regime, we use MS-COCO 2014-small, which is a small, randomly selected subset comprising 5% of MS-COCO 2014 which amounts to 4014 images. During testing, we use the complete validation set.

PASCAL VOC 2007: VOC [20] is a widely used outdoor scene MLR dataset consisting of 9,963 images from 20 classes. We follow the standard trainval set for training and use the test set for testing.

FoodSeg103: FoodSeg103 [44] serves as a benchmark dataset for food segmentation and multi-label food recognition. It consists of 4983 training images and 2135 test images, with a total of 32,097 food instances belonging to 103 different food classes. The number of images per class follows a long-tail distribution typical of real-world datasets. We use the standard train-test data split.

UNIMIB 2016: UNIMIB [10] is another multi-label food recognition dataset. It consists of 1027 images with 3616 food instances spanning 73 classes. Similar to FoodSeg103, UNIMIB also follows a long-tail distribution typical of real-world datasets. We comply with the official train-test split.

4.2 Implementation Details

In our experiments, we use CLIP (Contrastive Language-Image Pre-Training) [39] as the VLM. Consistent with recent works that use VLMs for MLR [1,17,41], we select ResNet-101 as the visual encoder and standard transformer within CLIP as the text encoder. Both encoders are kept frozen during our experiments, and we train the GCN and learnable prompts. Following [9,17,41], we resize the images to 448×448 for COCO and VOC datasets and to 224×224 for UNIMIB and FoodSeg103. Similar to previous works [17,41] we apply Cutout [16] and RandAugment [12] to augment training images. We use a 3-layer GCN network for all our experiments. We use SGD for optimizing parameters with an initial learning rate of 0.002, which is reduced by cosine annealing. We train for 50 epochs and use a batch size of 32. We set the loss hyperparameters in Eq. 4 as $\gamma_- = 3$, $\gamma_+ = 1$ and $\delta = 0.05$. We conduct all experiments on a single RTX A4000 GPU.

4.3 Evaluation Metrics

To evaluate the performance of our approach on the four MLR datasets, we use standard metrics, also used by previous MLR approaches [7–9]. The metrics include the commonly used mean average precision (mAP) as well as class and overall precisions (CP and OP), recalls (CR and OR), and F1 scores (CF1 and OF1). mAP is obtained by calculating the mean of individual average precision (AP) values over all classes. For each class, AP is computed as the area under the Precision-Recall curve.

4.4 Results

We primarily compare our approach with DualCoOp [41] and SCPNet [17], as they are the only other MLR baselines that use VLMs. Making them SOTA VLM-based methods in limited data settings across all four standard benchmarks discussed earlier. As seen in Table 1, our method outperforms DualCoOp by 0.4% and SCPNet by 7.0% mAP on the VOC-2007. On COCO-small, our method outperforms DualCoOp by 2.4% and SCPNet by 3.3% mAP. On the FoodSeg103 dataset, our approach significantly improves upon DualCoOp by 3.9% and SCPNet by 4.1% mAP. In the UNIMIB dataset, our method achieves substantial performance gains of 14.1% over DualCoOp and 12.2% mAP over SCPNet.

Furthermore, for VOC, we extend our comparison to approaches that do not use VLMs and instead rely on complete fine-tuning[7–9]. These approaches also use a ResNet-101 backbone similar to our visual encoder, but the backbone is initialized with weights pre-trained on ImageNet instead [15]. Our method also outperforms these methods. More detailed comparison can be found in Table. 1. Additionally, we provide a comparison of our method with zero-shot MLR using VLMs in Sec 1. of the supplementary material.

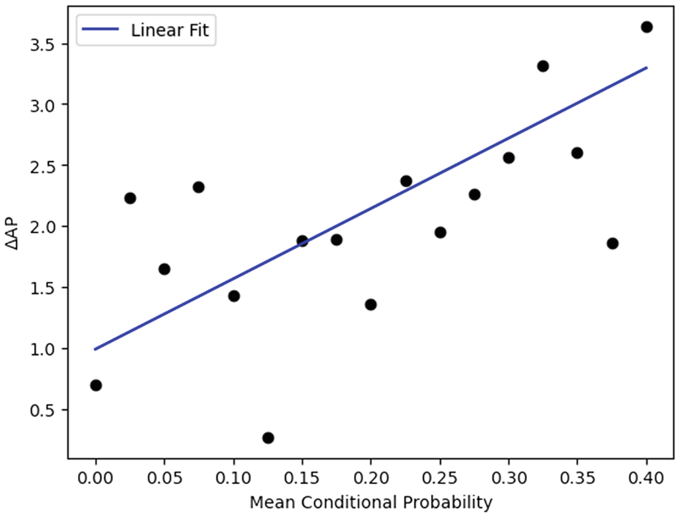


Fig. 2. Improvement in average precision (ΔAP) of a class obtained by refining VLM-based initial logits to incorporate the information provided by conditional probabilities, shown as a function of the mean conditional probability of most co-occurring three classes.

4.5 Impact of the Strength of Conditional Probability on Performance

In this section, we determine the impact of the strength of conditional probability of a pair of classes on MLR performance on the COCO-small dataset. Specifically, we observe how the improvement in average precision of a class of objects (ΔAP) brought by our method varies with the average conditional probability of the class paired with the top three other classes it co-occurs the most with. Note that we choose to average the top three values of conditional probabilities of the class because the COCO dataset typically contains an average of around three objects per image. The improvement in (ΔAP) for a class = AP achieved by logits after refinement - AP achieved by the raw VLM logits before refinement.

We visualize the variation in ΔAP with the avg. conditional probability in Figure 2, where we observe an increasing trend of ΔAP with increase in the average of the top-3 conditional probabilities for a given class. Note that for ease of visualization of the scatterplot, we use bins of size 0.02 to group together classes having similar average conditional probabilities. The points represent the average ΔAP value of all classes within the respective bin.

This implies that classes having stronger conditional probabilities with other classes benefit more from our approach of refining logits using conditional probabilities, as is intuitively expected.

4.6 Performance on Classes that are Difficult to Recognize

In this section, we empirically explore the impact of our approach on classes that are difficult to recognize when using image features exclusively. For concreteness, we focus our analysis on the 10 classes in FoodSeg103 and UNIMIB datasets on which the previous state-of-the-art approach (DualCoop[41]) performs (in terms of CF1) the worst.

Table 2 compares the performance of our method on these classes with the previous SOTA DualCoOp[41] and SCPNet[17]. We see that our method significantly improves the performance of these methods, which relies solely on VLMs without modeling any conditional probabilities. Specifically, for DualCoOp, we observe a growth of 22.3% in CP, 33.9% in CR and 34.8% for CF1 on UNIMIB2016, and 15.6% in CP, 7.2% in CR and 11.89% in CF on FoodSeg103. For SCPNet, we see gains of 27.1% CP, 25.2% CR, and 26.5% CF1 on UNIMIB2016, and 15.8% CP, 5.8% CR, and 12.4% CF1 on FoodSeg103.

This underscores the importance of the information obtained by modeling joint class probabilities in recognizing classes of objects that are difficult to recognize from image features alone.

Table 2. A comparison of the average performance of our approach with the previous state-of-the-art VLM-based method DualCoOp[41] amd [17] on classes that are difficult to recognize using only visual features (having 10 lowest CF1 values on the FoodSeg103[44] and UNIMIB[10]). Our approach significantly improves MLR performance on such classes due to its use of information derived from class conditional probabilities.

	UNIMIB			FoodSeg103		
Methods	CP	CR	CF1	CP	CR	CF1
DualCoOp	25.4	26.2	24.3	13.7	19.7	16.5
SCPNet	30.5	34.8	32.5	12.9	21.1	16.0
Ours w/o reweigh	41.9	57.5	44.9	14.8	22.5	18.7
Ours	**57.6**	**60.0**	**59.1**	**28.7**	**26.9**	**28.4**

4.7 Effect of Loss Reweighing

In this subsection, we investigate the effects of our loss reweighing strategy for UNIMIB2016 and FoodSeg103, which exhibit significant class imbalance. We analyze its impact on performance on (1) All classes as a whole and (2) Classes that are difficult to recognize using only visual features, and hence have a greater reliance on information obtained from conditional probabilities.

(1) **All classes as a whole:** As observed in Table 1, loss re-weighing improves the performance of our method by 1.6% and 7.8% in mAP on FoodSeg103 and UNIMIB2016, respectively.

This demonstrates the utility of using loss re-weighing when training a method modeling the joint probability distribution of classes as opposed.

(2) **Classes that are difficult to recognize using only visual features:** As seen in Table 2, loss re-weighing also significantly boosts the performance of our method on these classes. It improves CP, CR and CF1 by 13.9%, 4.4% and 9.7%, respectively on the FoodSeg103 dataset and by 15.7%, 4.5% and 14.2%, respectively on the UNIMIB2016 dataset. Note that these increases are higher than corresponding gains observed for all classes in the dataset, signifying that loss re-weighing delivers larger benefits to classes more reliant on conditional probabilities for recognition (as opposed to those that derive more information from initial logits estimated from logits)

5 Conclusions

In this paper, we present a novel two-stage framework for multi-label recognition when only a small number of annotated images are available. Our framework builds on recent methods that make use of VLMs to counter this paucity of labeled data but overlook information derived from co-occurrence of object pairs. Our framework refines the logit predictions made by VLMs adapted for multi-label recognition by leveraging known conditional probabilities of class pairs derived from the training data distribution. Specifically, we use a graph convolutional network to enrich the logits predicted by the VLM with information from conditional probabilities of classes. Our method outperforms all state-of-the-art approaches on 4 MLR benchmarks: COCO-14-small, VOC 2007, FoodSeg103 and UNIMIB2016 in a low data regime, demonstrating the utility of modeling class co-occurrence in such cases.

6 Limitations

(1) If the independent classifiers learned by state-of-the-art approaches (relying on only visual information, not modeling the conditional probability of class pairs) are strong, and characterized by a high average precision (AP) of each class, our method would yield lower improvements. However, in practice, many MLR datasets are not very large, with independent classifiers learned from them being relatively weak. MLR on such datasets is likely to benefit significantly from our method.

(2) As shown in Figure. 2, the advantage provided by our method is higher when the conditional probability of pairs of classes co-occurring in an image is higher. For images that consist of objects which are rarely found together, our method provides very little added benefit over independent classifiers.

Acknowledgement. We thank Kamila Abdiyeva for her insightful feedback on the manuscript. The support of the Office of Naval Research under grant N00014-20-1-2444, of USDA National Institute of Food and Agriculture under grant 2020-67021-32799/1024178 and Vizzhy.com are gratefully acknowledged.

References

1. Abdelfattah, R., Guo, Q., Li, X., Wang, X., Wang, S.: Cdul: Clip-driven unsupervised learning for multi-label image classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1348–1357 (2023)
2. Anthimopoulos, M.M., Gianola, L., Scarnato, L., Diem, P., Mougiakakou, S.G.: A food recognition system for diabetic patients based on an optimized bag-of-features model. IEEE J. Biomed. Health Inform. **18**(4), 1261–1271 (2014)
3. Bhatnagar, S., Ahuja, N.: Piecewise-linear manifolds for deep metric learning. In: Conference on Parsimony and Learning. pp. 269–281. PMLR (2024)
4. Cao, K., Wei, C., Gaidon, A., Arechiga, N., Ma, T.: Learning imbalanced datasets with label-distribution-aware margin loss. Advances in neural information processing systems **32** (2019)
5. Chang, W.C., Jiang, D., Yu, H.F., Teo, C.H., Zhang, J., Zhong, K., Kolluri, K., Hu, Q., Shandilya, N., Ievgrafov, V., et al.: Extreme multi-label learning for semantic matching in product search. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining. pp. 2643–2651 (2021)
6. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research **16**, 321–357 (2002)
7. Chen, T., Lin, L., Chen, R., Hui, X., Wu, H.: Knowledge-guided multi-label few-shot learning for general image recognition. IEEE Trans. Pattern Anal. Mach. Intell. **44**(3), 1371–1384 (2020)
8. Chen, T., Xu, M., Hui, X., Wu, H., Lin, L.: Learning semantic-specific graph representation for multi-label image recognition. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 522–531 (2019)
9. Chen, Z.M., Wei, X.S., Wang, P., Guo, Y.: Multi-label image recognition with graph convolutional networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5177–5186 (2019)
10. Ciocca, G., Napoletano, P., Schettini, R.: Food recognition: a new dataset, experiments, and results. IEEE J. Biomed. Health Inform. **21**(3), 588–598 (2016)
11. Cole, E., Mac Aodha, O., Lorieul, T., Perona, P., Morris, D., Jojic, N.: Multi-label learning from single positive labels. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 933–942 (2021)
12. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. pp. 702–703 (2020)
13. Cui, Y., Jia, M., Lin, T.Y., Song, Y., Belongie, S.: Class-balanced loss based on effective number of samples. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9268–9277 (2019)
14. Cui, Y., Jia, M., Lin, T.Y., Song, Y., Belongie, S.: Class-balanced loss based on effective number of samples. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9268–9277 (2019)
15. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
16. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017)

17. Ding, Z., Wang, A., Chen, H., Zhang, Q., Liu, P., Bao, Y., Yan, W., Han, J.: Exploring structured semantic prior for multi label recognition with incomplete labels. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3398–3407 (2023)
18. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
19. Estabrooks, A., Jo, T., Japkowicz, N.: A multiple resampling method for learning from imbalanced data sets. Comput. Intell. **20**(1), 18–36 (2004)
20. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. Int. J. Comput. Vision **88**, 303–338 (2010)
21. Gao, P., Geng, S., Zhang, R., Ma, T., Fang, R., Zhang, Y., Li, H., Qiao, Y.: Clip-adapter: Better vision-language models with feature adapters. Int. J. Comput. Vision **132**(2), 581–595 (2024)
22. Huang, H., Rawlekar, S., Chopra, S., Deniz, C.M.: Radiology reports improve visual representations learned from radiographs. In: Medical Imaging with Deep Learning. pp. 1385–1405. PMLR (2024)
23. Ilharco, G., Wortsman, M., Wightman, R., Gordon, C., Carlini, N., Taori, R., Dave, A., Shankar, V., Namkoong, H., Miller, J., Hajishirzi, H., Farhadi, A., Schmidt, L.: Openclip (Jul 2021). https://doi.org/10.5281/zenodo.5143773, https://doi.org/10.5281/zenodo.5143773, if you use this software, please cite it as below
24. Jia, C., Yang, Y., Xia, Y., Chen, Y.T., Parekh, Z., Pham, H., Le, Q., Sung, Y.H., Li, Z., Duerig, T.: Scaling up visual and vision-language representation learning with noisy text supervision. In: International conference on machine learning. pp. 4904–4916. PMLR (2021)
25. Kang, B., Li, Y., Xie, S., Yuan, Z., Feng, J.: Exploring balanced feature spaces for representation learning. In: International Conference on Learning Representations (2020)
26. Karthik, S., Roth, K., Mancini, M., Akata, Z.: Vision-by-language for training-free compositional image retrieval. arXiv preprint arXiv:2310.09291 (2023)
27. Khan, S., Hayat, M., Zamir, S.W., Shen, J., Shao, L.: Striking the right balance with uncertainty. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 103–112 (2019)
28. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
29. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common Objects in Context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
30. Liu, F., Xiang, T., Hospedales, T.M., Yang, W., Sun, C.: Semantic regularisation for recurrent image annotation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2872–2880 (2017)
31. Liu, W., Tsang, I.: On the optimality of classifier chain for multi-label classification. Advances in Neural Information Processing Systems **28** (2015)
32. Liu, W., Wang, H., Shen, X., Tsang, I.W.: The emerging trends of multi-label learning. IEEE Trans. Pattern Anal. Mach. Intell. **44**(11), 7955–7974 (2021)
33. Liu, X.Y., Wu, J., Zhou, Z.H.: Exploratory undersampling for class-imbalance learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **39**(2), 539–550 (2008)

34. Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., Yu, S.X.: Large-scale long-tailed recognition in an open world. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2537–2546 (2019)
35. Menon, A.K., Jayasumana, S., Rawat, A.S., Jain, H., Veit, A., Kumar, S.: Long-tail learning via logit adjustment. arXiv preprint arXiv:2007.07314 (2020)
36. Meyers, A., Johnston, N., Rathod, V., Korattikara, A., Gorban, A., Silberman, N., Guadarrama, S., Papandreou, G., Huang, J., Murphy, K.P.: Im2calories: towards an automated mobile vision food diary. In: Proceedings of the IEEE international conference on computer vision. pp. 1233–1241 (2015)
37. Misra, I., Lawrence Zitnick, C., Mitchell, M., Girshick, R.: Seeing through the human reporting bias: Visual classifiers from noisy human-centric labels. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2930–2939 (2016)
38. Park, S., Lim, J., Jeon, Y., Choi, J.Y.: Influence-balanced loss for imbalanced visual classification. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 735–744 (2021)
39. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
40. Ridnik, T., Ben-Baruch, E., Zamir, N., Noy, A., Friedman, I., Protter, M., Zelnik-Manor, L.: Asymmetric loss for multi-label classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 82–91 (2021)
41. Sun, X., Hu, P., Saenko, K.: Dualcoop: Fast adaptation to multi-label recognition with limited annotations. Adv. Neural. Inf. Process. Syst. **35**, 30569–30582 (2022)
42. Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., Xu, W.: Cnn-rnn: A unified framework for multi-label image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2285–2294 (2016)
43. Wortsman, M., Ilharco, G., Kim, J.W., Li, M., Kornblith, S., Roelofs, R., Lopes, R.G., Hajishirzi, H., Farhadi, A., Namkoong, H., et al.: Robust fine-tuning of zero-shot models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 7959–7971 (2022)
44. Wu, X., Fu, X., Liu, Y., Lim, E.P., Hoi, S.C., Sun, Q.: A large-scale benchmark for food image segmentation. In: Proceedings of the 29th ACM international conference on multimedia. pp. 506–515 (2021)
45. Xu, M., Zhang, Z., Wei, F., Lin, Y., Cao, Y., Hu, H., Bai, X.: A simple baseline for open-vocabulary semantic segmentation with pre-trained vision-language model. In: European Conference on Computer Vision. pp. 736–753. Springer (2022)
46. Yang, J., Price, B., Cohen, S., Yang, M.H.: Context driven scene parsing with attention to rare classes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3294–3301 (2014)
47. Yao, Y., Zhang, A., Zhang, Z., Liu, Z., Chua, T.S., Sun, M.: Cpt: Colorful prompt tuning for pre-trained vision-language models. AI Open **5**, 30–38 (2024)
48. Yazici, V.O., Gonzalez-Garcia, A., Ramisa, A., Twardowski, B., Weijer, J.v.d.: Orderless recurrent models for multi-label classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13440–13449 (2020)
49. Zhang, H., Li, F., Ahuja, N.: Open-nerf: Towards open vocabulary nerf decomposition. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 3456–3465 (2024)

50. Zhang, H., Li, F., Qi, L., Yang, M.H., Ahuja, N.: Csl: Class-agnostic structure-constrained learning for segmentation including the unseen. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 7078–7086 (2024)
51. Zhang, R., Zhang, W., Fang, R., Gao, P., Li, K., Dai, J., Qiao, Y., Li, H.: Tip-adapter: Training-free adaption of clip for few-shot classification. In: European conference on computer vision. pp. 493–510. Springer (2022)
52. Zhang, Z., Pfister, T.: Learning fast sample re-weighting without reward data. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 725–734 (2021)
53. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Learning to prompt for vision-language models. Int. J. Comput. Vision **130**(9), 2337–2348 (2022)

Author Index