Parsimonious HMMs for Offline Handwritten Chinese Text Recognition

Wenchao Wang, **Jun Du** and Zi-Rui Wang
University of Science and Technology of China

ICFHR 2018, Niagara Falls, USA, Aug. 5-8, 2018

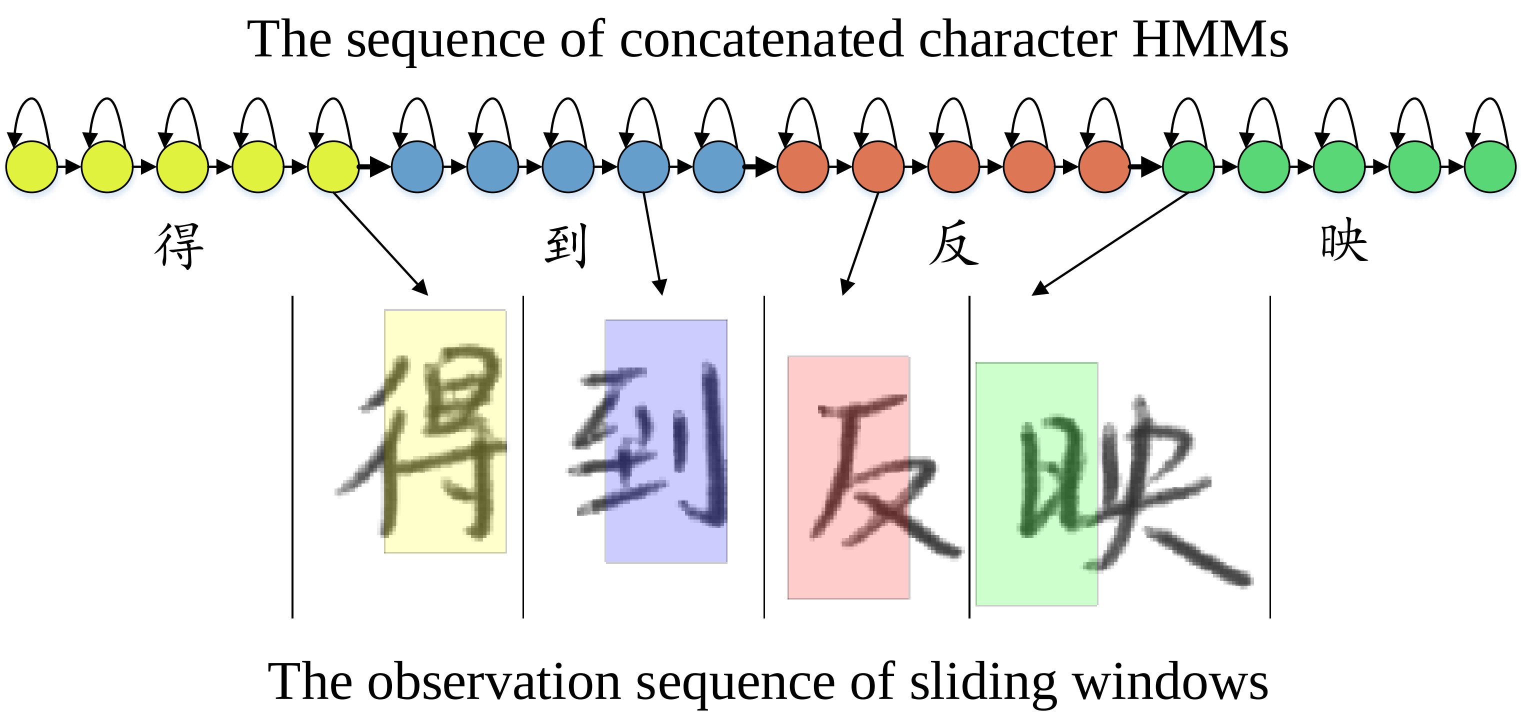
Background

- Offline handwritten Chinese text recognition (OHCTR) is challenging
  - No trajectory information in comparison to the online case
  - Large vocabulary of Chinese characters
  - Sequential recognition with the potential segmentation problem

- Approaches
  - Oversegmentation approaches
    - Character oversegmentation/classification
  - Segmentation-free approaches
    - GMM-HMM: Gaussian mixture model - hidden Markov model
    - MDLSTM-RNN: Multidimensional LSTM-RNN + CTC
    - DNN-HMM: Deep neural network – hidden Markov model

Review of HMM Approach for OHCTR

- Left-to-right HMM is adopted to represent Chinese character.
- The character HMMs are concatenated to model the text line.



Review of DNN-HMM Approach for OHCTR

- The Bayesian framework

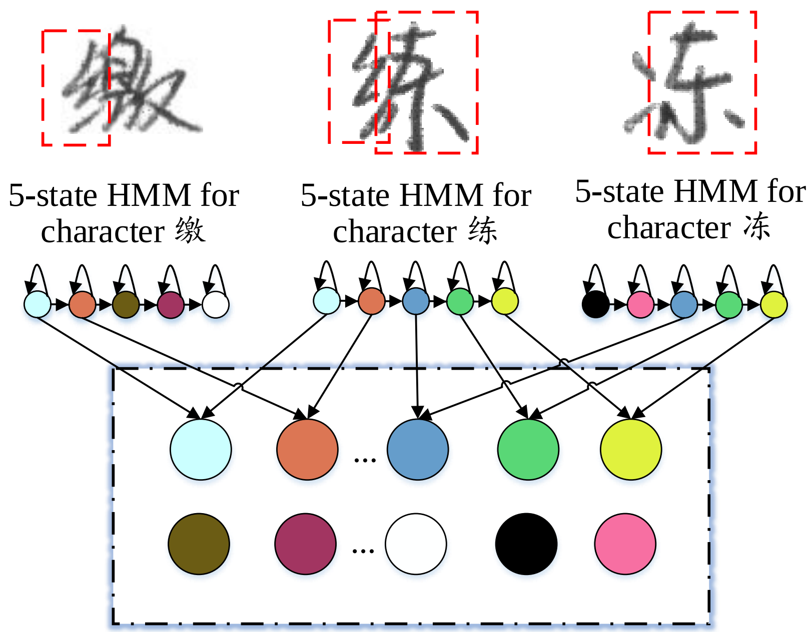
$$\hat{\mathbf{C}} = \arg\max_{\mathbf{C}} p(\mathbf{C} \mid \mathbf{X}) = \arg\max_{\mathbf{C}} \boxed{p(\mathbf{X} \mid \mathbf{C})} P(\mathbf{C})$$ $\longrightarrow$ Character modeling

$$\begin{aligned} p(\mathbf{X} \mid \mathbf{C}) &= \Sigma_S \left[p(\mathbf{X} \mid S, \mathbf{C}) p(S \mid \mathbf{C}) \right] \\ &= \Sigma_S \left[\pi(s_0) \prod_{t=1}^{T} a_{s_{t-1}s_t} \prod_{t=0}^{T} \boxed{p(\mathbf{x}_t | s_t)} \right] \end{aligned}$$ $\longrightarrow$ Output distribution

$$p(\mathbf{x}_t | s_t) = \frac{\boxed{p(s_t | \mathbf{x}_t)} p(\mathbf{x}_t)}{p(s_t)}$$ $\longrightarrow$ DNN to calculate state posterior probability

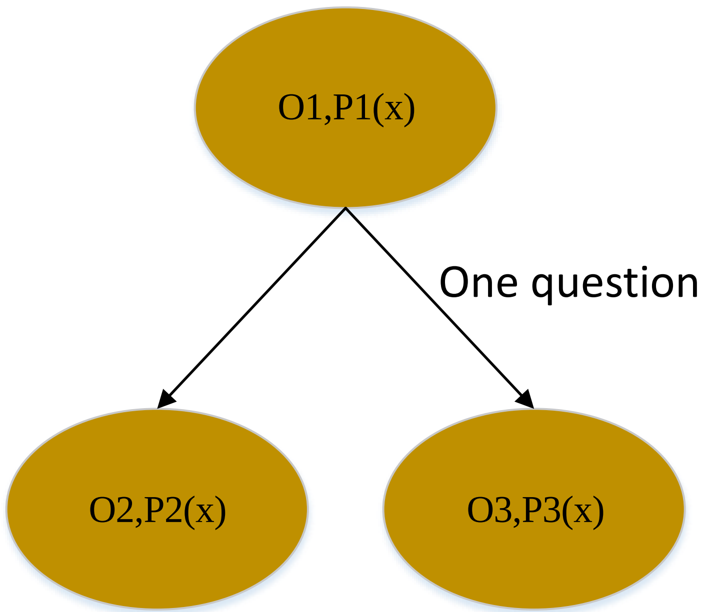
Motivation

- High demand of **memory** and **computation** from DNN output layer
- Model redundancy due to similarities among different characters
- Parsimonious HMMs to address these two problems
- Decision tree based two-step approach to generate tied-state pool



Tied-state Pool

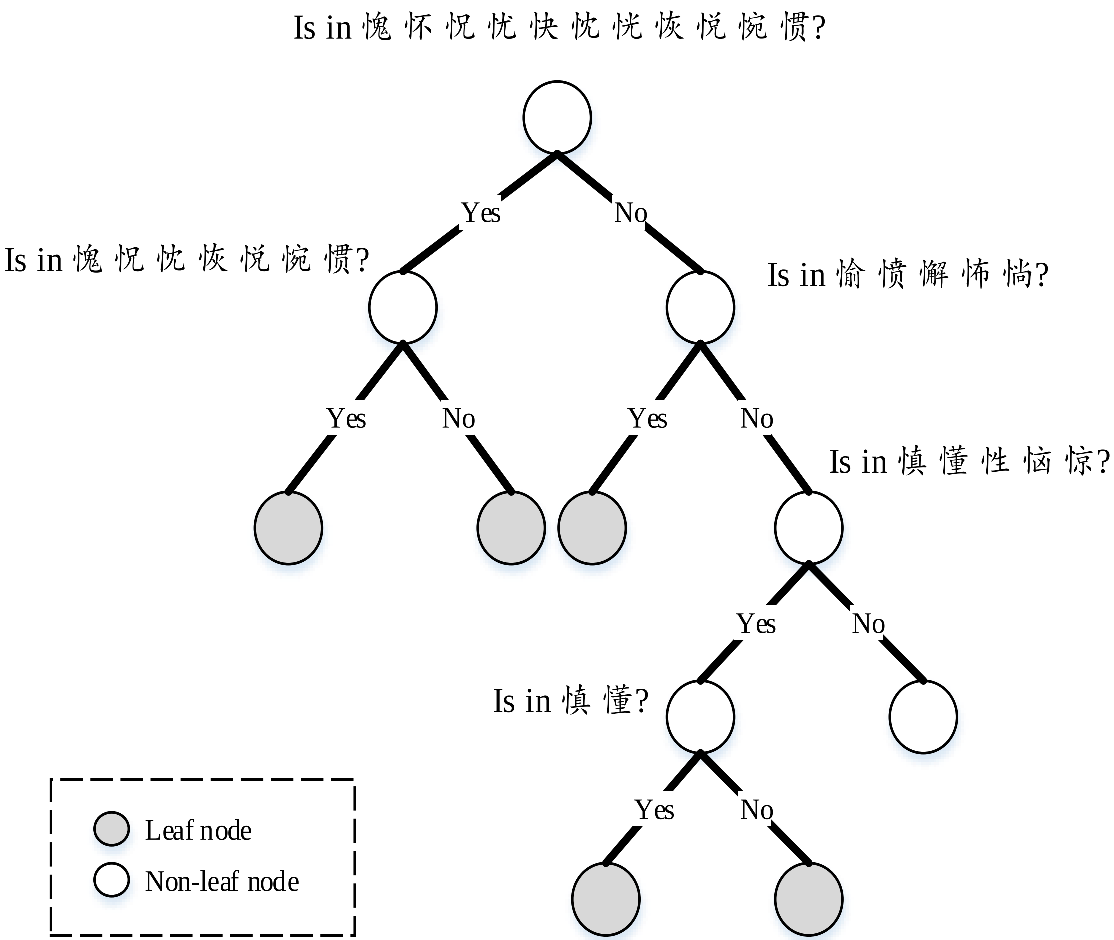
Binary Decision Tree for State Tying



$$O_1 = O_2 \bigcup O_3$$

- The parent set O_1 has a distribution $P_1(x)$, the total log-likelihood of all observations in O_1 on the distribution of $P_1(x)$ is: $L(O_1) = \sum_{x \in O_1} \log(P_1(x))$
- The child set O_2 has a distribution $P_2(x)$, the total log-likelihood of all observations in O_2 on the distribution of $P_2(x)$ is: $L(O_2) = \sum_{x \in O_2} \log(P_2(x))$
- The child set O_3 has a distribution $P_3(x)$, the total log-likelihood of all observations in O_3 on the distribution of $P_3(x)$ is: $L(O_3) = \sum_{x \in O_3} \log(P_3(x))$
- The total increase in set-conditioned log-likelihood of observations due to partitioning is: $L(O_2) + L(O_3) - L(O_1)$

Step 1: Clustering Characters with Decision Tree



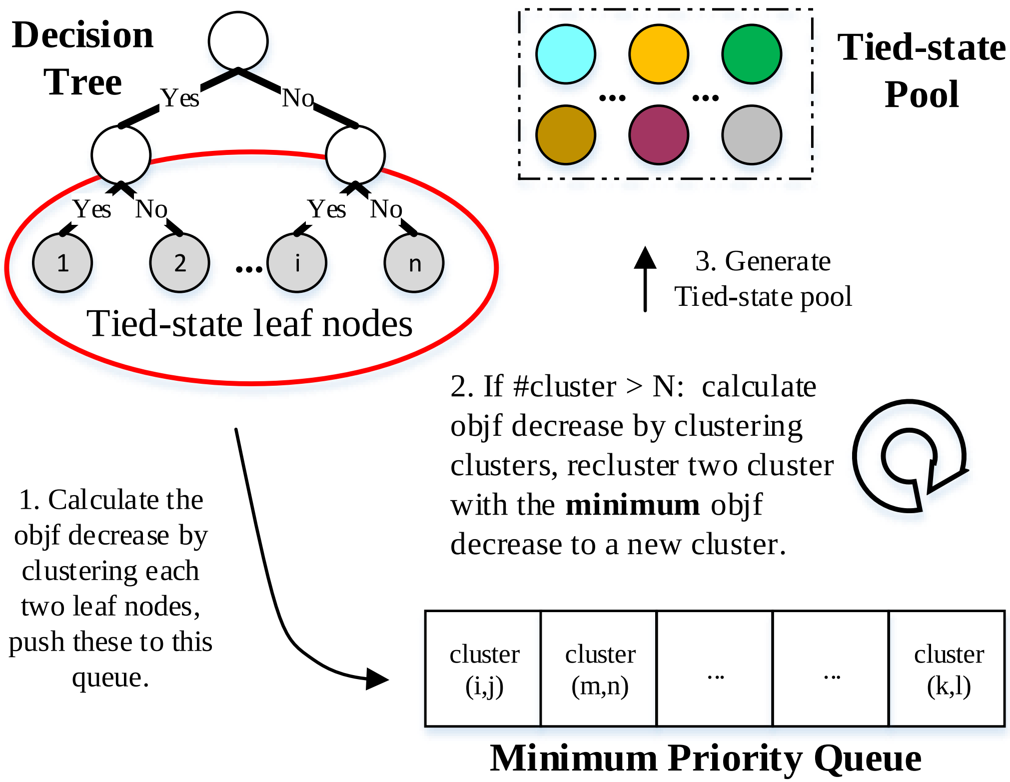
A tree fragment for tying the first state of HMM

- All states with the same HMM position are initially grouped together at the root node.
- Each node is then recursively partitioned to **maximize** the increase in expected log-likelihood with question set.

$$
\begin{aligned}
L(\mathbf{x}) &= E\left[\log \mathcal{N}(\mathbf{x};\mu,\boldsymbol{\Sigma})\right] \\
&= -\frac{1}{2}E\left[(\mathbf{x}-\mu)^{\top}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mu)+\log((2\pi)^{D}|\boldsymbol{\Sigma}|)\right] \\
&= -\frac{1}{2}[(1+\log(2\pi))D+\log|\boldsymbol{\Sigma}|]
\end{aligned}
$$

- All states in the leaves of the decision tree are tied together.

Step 2: Bottom-up Re-clustering



- In the second step, the clusters in leaf nodes obtained in the first step is re-clustered by a bottom-up procedure using sequential greedy optimization.
- The expected log-likelihood decrease by combining every two clusters is calculated.
- A minimum priority queue is maintained to re-cluster the two clusters with minimum log-likelihood decrease to a new cluster.

Training Procedure for Parsimonious HMMs

1. Training conventional GMM-HMM system
2. Calculating the first-order and second-order statistics based on state-level forced-alignment
3. Two-step algorithm:
   First-step: Building the state-tying tree
   Second-step: Re-clustering the tied-states based on the first-step
4. Parsimonious GMM-HMMs training based on the tied states
5. Parsimonious DNN-HMMs training based on the tied states

Experiments

- Training set

  CASIA-HWDB database including HWDB1.0, HWDB1.1, HWDB2.0-HWDB2.2
- Test set

  ICDAR-2013 competition set.
- Vocabulary: 3980 character classes
- GMM-HMM system
  - Each character modeled by a left-to-right HMM with 40-component GMM
  - Gradient-based features followed by PCA to obtain a 50-dimensional vector
- DNN-HMM system
  - 350-2048-2048-2048-2048-2048-2048-3980*N
- DNN-PHMM system
  - 350-2048-2048-2048-2048-2048-2048-M

HMM vs. PHMM

Table I
THE CER(%) COMPARISON OF HMM SYSTEMS WITH DIFFERENT NUMBER SETTINGS OF TIED-STATES PER CHARACTER N_s.

N_s	5	4	3	2	1
GMM-HMM	20.04	**19.94**	21.94	24.92	30.34
GMM-PHMM	-	19.41	18.83	**18.14**	18.49
DNN-HMM	**6.73**	6.80	7.11	8.21	11.09
DNN-PHMM	-	6.37	**6.31**	6.48	7.15

- Performance saturation with the increase of states for each character
- PHMM outperforming HMM with the same setting of tied-state number
- Parsimoniousness of the best PHMM compared with the best HMM
- Demonstrating the reasonability of the proposed state tying algorithm

HMM vs. PHMM

Table II
THE CER(%) COMPARISON OF HMM SYSTEMS WITH DIFFERENT NUMBER SETTINGS OF TIED-STATES PER CHARACTER N_s <1.

N_s	0.9	0.8	0.7	0.6	0.5
GMM-PHMM	18.66	19.17	19.92	21.28	22.54
DNN-PHMM	7.34	7.50	7.97	8.80	9.52

- Much more compact by setting the number of tied-states per character < 1
- DNN-PHMM (Ns=0.5, 9.52%) outperforming DNN-HMM (Ns=1, 11.09%)

Memory and Computation Costs

Table III

THE PERFORMANCE COMPARISON OF THE BEST CONFIGURED DNN-HMM AND DNN-PHMM SYSTEMS WITH DIFFERENT DNN STRUCTURES. (N_{U} AND N_{L} ARE THE NUMBERS OF HIDDEN UNITS AND LAYERS, N_{M} AND N_{T} ARE THE MODEL SIZE AND RUN-TIME LATENCY NORMALIZED BY DNN-HMM WITH N_{U}=2048 AND N_{L}=6.)

(N_{U}, N_{L})		(1024, 4)	(1024, 6)	(2048, 6)
DNN-HMM	CER	7.15	6.91	6.73
	N_{M}	0.38	0.42	1
	N_{T}	0.82	0.93	1
DNN-PHMM	CER	6.78	6.48	6.31
	N_{M}	0.25	0.27	0.74
	N_{T}	0.28	0.31	0.40

DNN-PHMM using (1024, 4) setting achieved a comparable CER with DNN-HMM using (2048, 6) setting, 75% of model size and 72% of run-time latency were reduced in DNN-PHMM compared with DNN-HMM.

State Tying Result Analysis

Tied characters	**Radical structure**	**Similar part**
喷 喻 嗅 嗡 吃 咆 哦 哨 嘈 嘲 噬 嚼	Left-right	口
客 害 容 密 寇 蜜 穷 穿 突 窃 窍 窑	Top-bottom	宀
圃 圆 囚 囤 困 围 固	Surround	囗
巨 匝 匠 匡 匣 匪 匹 医 匿 臣	Left-surround	匚
诞 巡 边 逊 辽 达 谜 迁 迂 过 近 这	Bottom-left-surround	辶
澜 阐 阑 鬲 闸 闻 闽 润	Top-surround	门
串 吊 甲 牢 帛 旱 平	Cross	丨
氛 氢 氦 氨	Top-right-surround	气

The Chinese characters with the **same or similar radicals** were easily tied using the proposed algorithm. This is the reason that the proposed DNN-PHMM with quite compact design can still maintain high recognition performance.

Thanks!