Journal Pre-proof

Writer-Aware CNN for Parsimonious HMM-Based Offline Handwritten Chinese Text Recognition

Zi-Rui Wang, Jun Du, Jia-Ming Wang

PII: S0031-3203(19)30403-0
DOI: https://doi.org/10.1016/j.patcog.2019.107102
Reference: PR 107102

To appear in: *Pattern Recognition*

Received date: 29 November 2018
Revised date: 6 September 2019
Accepted date: 31 October 2019

Please cite this article as: Zi-Rui Wang, Jun Du, Jia-Ming Wang, Writer-Aware CNN for Parsimonious HMM-Based Offline Handwritten Chinese Text Recognition, *Pattern Recognition* (2019), doi: https://doi.org/10.1016/j.patcog.2019.107102

Writer-Aware CNN for Parsimonious HMM-Based Offline Handwritten Chinese Text Recognition

Zi-Rui Wang, Jun Du✉, Jia-Ming Wang

Abstract

Recently, the hybrid convolutional neural network hidden Markov model (CNN-HMM) has been introduced for offline handwritten Chinese text recognition (HCTR) and has achieved state-of-the-art performance. However, modeling each of the large vocabulary of Chinese characters with a uniform and fixed number of hidden states requires high memory and computational costs and makes the tens of thousands of HMM state classes confusing. Another key issue of CNN-HMM for HCTR is the diversified writing style, which leads to model strain and a significant performance decline for specific writers. To address these issues, we propose a writer-aware CNN based on parsimonious HMM (WCNN-PHMM). First, PHMM is designed using a data-driven state-tying algorithm to greatly reduce the total number of HMM states, which not only yields a compact CNN by state sharing of the same or similar radicals among different Chinese characters but also improves the recognition accuracy due to the more accurate modeling of tied states and the lower confusion among them. Second, WCNN integrates each convolutional layer with one adaptive layer fed by a writer-dependent vector, namely, the writer code, to extract the irrelevant variability in writer information to improve recognition performance. The parameters of writer-adaptive layers are jointly optimized with other network parameters in the training stage, while a multiple-pass decoding strategy is adopted to learn the writer code and generate recognition results. Validated on the ICDAR 2013 competition of CASIA-HWDB database, the more compact WCNN-PHMM of a 7360-class vocabulary can achieve a relative character error rate (CER) reduction of 16.6% over the conventional CNN-HMM without considering language modeling. By adopting a powerful hybrid language model (N-gram language model and recurrent neural network language model), the CER of WCNN-PHMM is reduced to 3.17%. Moreover, the state-tying results of PHMM explicitly show the information sharing among similar characters and the confusion reduction of tied state classes. Finally, we visualize the learned writer codes and demonstrate the strong relationship with the writing styles of different writers. To the best of our knowledge, WCNN-PHMM yields the

Zi-Rui Wang, Jun Du, and Jia-Ming Wang are with the National Engineering Laboratory for Speech and Language Information Processing, University of Science and Technology of China, Hefei, Anhui, P. R. China; email: cs211@mail.ustc.edu.cn, jundu@ustc.edu.cn, jmwang66@mail.ustc.edu.cn

best results on the ICDAR 2013 competition set, demonstrating its power when enlarging the size of the character vocabulary.

Index Terms

Offline handwritten Chinese text recognition, writer-aware CNN, parsimonious HMM, state tying, adaptation, hybrid language model.

I. INTRODUCTION

The robust recognition of handwritten text lines in an unconstrained writing style plays an important role in many applications, such as machine scoring, express sorting and document recognition. Specifically, handwritten Chinese text recognition (HCTR) has been intensively studied as a popular research topic for many years [1], [2]. However, it remains a challenging problem due to the large vocabulary and the diversity of writing styles. Moreover, offline HCTR, which is the focus of this study, is more difficult than online HCTR [3], [4], as the ink trajectory information is missing.

In general, the research efforts for offline HCTR can be divided into two categories: oversegmentation-based approaches and segmentation-free approaches. The former approaches [5], [6], [7], [8] often build several modules by first including character oversegmentation, character classification, and modeling the linguistic and geometric contexts, and then incorporating them to calculate the score for path search. The recent work in [8], with the neural network language model, adopted three different CNN models to replace the conventional character classifier, segmentation and geometric models to achieve the best performance of oversegmentation-based methods on the ICDAR 2013 competition dataset [9]. By contrast, segmentation-free approaches do not need to explicitly segment text lines. One early approach to text line modeling [10] used the Gaussian mixture model hidden Markov model (GMM-HMM). Another recent approach [11] utilized multidimensional long short-term memory recurrent neural network (MDLSTM-RNN), which was inspired by well-verified LSTM-RNN approaches [12] for the recognition of handwritten western languages with a small set of character classes. The MDLSTM-RNN approach is quite flexible due to the connectionist temporal classification (CTC) technique [13], which avoids explicit segmentation. In [14], the authors employed a CNN and an LSTM neural network under the HMM framework to obtain a significant improvement over the LSTM-HMM model. In [15], the authors used separable MDLSTM-RNN (SMDLSTM-RNN) with CTC loss, instead of the traditional LSTM-CTC method. More recently, the authors in [16] proposed a novel aggregation cross-entropy loss for sequence recognition, which was shown to exhibit competitive performance for offline HCTR. In [17], we verified that combining hybrid deep CNN-HMM (DCNN-HMM) with a powerful language model could achieve the best reported results of the segmentation-free approaches on the ICDAR 2013 competition dataset.

However, the impressive results reported in recently proposed oversegmentation-based and segmentation-free approaches [8], [16], [17] highly depend on the use of strong language models (LMs) built with a large number of text corpora, which partially masks the weakness of character models and makes the comparison of character models unfair. Actually, the large vocabulary of Chinese characters and the diversified writing styles of text lines still limit the performance of deep learning methods based on character modeling. For example, in our DCNN-HMM work [17], the number of output nodes in DCNN, i.e., the total state class number, was 19900 by modeling 3980 characters with a 5-state HMM for each. Obviously, a further increase of the vocabulary size could potentially lead to a data sparsity problem and high computation and memory costs, which makes the training of CNNs become difficult. Moreover, similar radicals among different Chinese characters should be shared by the same states to reduce ambiguity in the decoding stage. Another key issue is that free-style writing usually causes a mismatch between the distributions of the training and testing datasets, which significantly degrades the recognition accuracy of certain writers.

To address these two main problems, we propose a novel writer-aware CNN based on parsimonious HMM (WCNN-PHMM). First, PHMM is designed using a data-driven state-tying algorithm to freely compress the total number of HMM states. The binary decision tree with a data-driven question set is adopted to represent one fixed-position HMM state of all character classes. In this way, it can not only yield a compact CNN by state sharing of the same or similar radicals among different Chinese characters but also improve the recognition accuracy due to the more accurate modeling of tied states and the lower confusion among them. Second, WCNN embeds one linear adaptive layer fed by a writer-dependent vector (namely, the writer code) into each convolutional layer, which extracts the irrelevant variability of writer information to improve recognition performance. In the training stage, all writer codes and the parameters of the adaptation layers are initialized randomly and then jointly optimized with other network parameters using the writer-specific data. In the recognition stage, with the initial recognition results from the first-pass decoding with the writer-independent CNN-PHMM model, an unsupervised adaptation is performed to generate the writer code for the subsequent decoding of WCNN-PHMM. Furthermore, in order to overcome the data sparseness problem of traditional N-gram LM (NLM) [18], similar to [8], we introduce a recurrent neural network LM (RNNLM) [19] to form a hybrid LM (HLM).

The main contributions of this study can be summarized as follows:

- The new structure WCNN-PHMM is presented to tackle two key issues for offline HCTR: the large vocabulary and the diversity of writing styles.
- A general adaptive training approach is proposed to integrate with any type of CNNs to create writer-aware models. To the best of our knowledge, this paper is the first study of writer adaptation for offline HCTR.
- The fast and compact design of PHMM via state tying improves the recognition accuracy. More importantly, compared with other segmentation-free approaches, PHMM can yield even better recognition accuracy when enlarging the size of the character vocabulary by fully leveraging more training data and class information sharing.
- The effectiveness of WCNN-PHMM is visually illustrated by the analyses of the state-tying results and the learned writer codes.
- The proposed WCNN-PHMM demonstrates the best reported character error rate (CER) (8.42%) for a 7360-class vocabulary on the ICDAR 2013 competition set without using language models. By adopting a powerful HLM, the CER of WCNN-PHMM can be further reduced to 3.17%.

The remainder of this paper is organized as follows. Section II introduces related work. Section III gives an overview of the proposed framework. Section IV elaborates on the details of WCNN-PHMM. Section V reports the experimental results and analyses. Finally, Section VI concludes.

II. Related Work

In this section, we describe related work, including the basic principles for mainstream approaches of offline HCTR, model compression and writer adaptation.

A. Basic principles for offline HCTR

Offline HCTR can be formulated as the Bayesian decision problem:

$$\begin{aligned}\hat{\mathbf{C}} &= \arg\max_{\mathbf{C}} p(\mathbf{C}|\mathbf{X}) \\ &= \arg\max_{\mathbf{C}} p(\mathbf{X}|\mathbf{C})p(\mathbf{C})\end{aligned} \tag{1}$$

where $\mathbf{X}$ is the feature sequence of a given text line image and $\mathbf{C} = \{C_1, C_2, ..., C_n\}$ is the underlying n-character sequence. In oversegmentation-based approaches [6], the posterior probability $p(\mathbf{C}|\mathbf{X})$ can be computed by searching the optimal segmentation path and the corresponding posterior probability of the character sequence by combining the character classifier, the segmentation model and the geometric/language model. Regarding segmentation-free approaches, the CTC-based and HMM-based approaches are two mainstream frameworks. In the CTC-based approach [15], a special character blank class and a defined many-to-one mapping function are introduced to directly compute $p(\mathbf{C}|\mathbf{X})$ with the forward-backward algorithm [13]. For the HMM-based approach [17], $p(\mathbf{C}|\mathbf{X})$ can be reformulated as the conditional probability $p(\mathbf{X}|\mathbf{C})$ and the prior probability $p(\mathbf{C})$. More details will be provided in Section III.

B. Model compression

The state tying can be regarded as belonging to a more general field, i.e., model compression [20]. With the emergence of deep learning [21], many studies have focused on building compact and fast CNNs for practicability. Regarding the reduction in the number of parameters and the computation complexity of convolutional layers, research efforts can be divided roughly into low-rank decomposition [22], pruning [23], quantization [24] and compact network design [25]. Aside from these methods, a key issue with CNN-HMM-based offline HCTR [17] is the large vocabulary problem, which leads to tens of thousands of output nodes (corresponding to HMM states) in CNN architecture. This heavy overhead in the output layer of the CNN not only requires high memory and computation costs but also yields more confusion among state classes and CNN training difficulties. To handle this problem, inspired by the early work in speech recognition [26], [27], we introduce state tying via decision trees to freely compress the output layer of the CNN model. Considering the particularity of HCTR and the difficulty of defining an effective question set for the Chinese language, in our previous work [31], we successfully invented a data-driven state-tying approach for a huge set of HMMs representing Chinese characters and achieved promising recognition performance. It should be noted that, if we simply reduce the

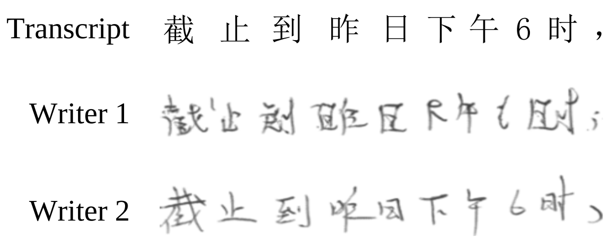


Fig. 1. Handwritten examples of different writers with the same transcript.

state number for each character, the recognition accuracy will decline dramatically due to the lack of resolution for text line modeling [17].

C. *Writer adaptation*

Writer adaptation is similar to other topics, such as transfer learning [33] and speaker adaptation [32], where the distribution of test data is different from that of training data [36]. In offline HCTR, as shown in Fig. 1, the writing styles could be quite different, which makes the recognition accuracy of unseen writers unpredictable. In comparison to handwritten Chinese character recognition (HCCR), aside from the morphological variations within characters, writing orientation and ligatures make HCTR much more challenging. In general, there are two mainstream methodologies to achieve writer adaptation. The one type is to adopt writer-specific data to guide writer-independent classifier toward the new distribution of the particular writer, the other is to extract writer-independent features for classifier. More specifically, this process might be supervised, semisupervised or unsupervised, depending on whether the adaptation writer-specific data are labeled. Usually, unsupervised adaptation needs to reuse the test data. Besides, it depends on adequate writer data. In some applications such as the machine scoring of essays [35], the recognition rate is the most important factor to be considered and there are enough specific writer data available to adopt adaptation techniques for improving the recognition rate. Moreover, the research on writer adaptation could be divided into feature-space and model-space approaches based on the part on which the adaptation parameters are working [37]. To the best of our knowledge, for Chinese handwriting recognition, almost all efforts of writer adaptation focus on the HCCR task. One such method uses a linear feature transformation to adapt the writing styles via discriminative linear regression (DLR) [38], [39], which is verified to be effective when incorporated with a prototype-based classifier and an NN-based classifier. Another representative

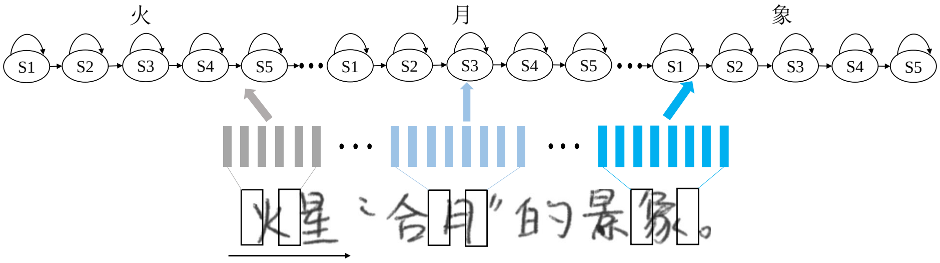


Fig. 2. Illustration of text line modeled by cascading character HMMs.

method introduces style transfer mapping (STM) [36] for learning a linear transformation to project writer-specific data onto a style-free space. As a flexible adaptation method, STM can work on the outputs of both fully connected layers [40], [41] and convolutional layers [56]. A recent study [42] uses adversarial learning [42] to transform writer-dependent features into writer-independent features under the guidance of printed data. However, there are very few studies for the writer adaptation of the more challenging HCTR problem. Inspired by [43], [44], in [45] we propose an unsupervised writer adaptation strategy for DNN-HMM-based HCTR.

This study is comprehensively extended from our previous conference papers [31], [45] with the following new contributions: 1) the proposed PHMM is introduced with more technical details and verified for a more promising CNN-HMM, rather than the DNN-HMM in [31]; 2) we present a novel unsupervised adaptation strategy with writer codes and adaptation layers to guide the convolutional layers in CNN-HMM, rather than using the fully connected layers in DNN-HMM [45]; 3) WCNN-PHMM perfectly combines the two techniques to yield a compact and high-performance model; 4) instead of the NLM, the HLM is used to further improve performance; and 5) all experiments are redesigned to verify the effectiveness of WCNN-PHMM, and detailed analyses are described to give the readers a deep understanding of our approach.

III. System Overview

Our system follows the basic HMM framework [17] in which the handwritten text line is modeled by a series of cascading HMMs, each representing one character, as illustrated in Fig. 2. The mathematic principle of HMM can be represented by rewriting the formula $p(\mathbf{X}|\mathbf{C})p(\mathbf{C})$ in

Eq. (1):

$$
\begin{aligned}
p(\mathbf{X}|\mathbf{C})p(\mathbf{C}) &= \sum_{S}\left[\pi(s_0)\prod_{t=1}^{T}a_{s_{t-1}s_t}\prod_{t=0}^{T}p(\mathbf{x}_t|s_t)\right]\prod_{i=1}^{n}p(C_i|C_{i-1},C_{i-2},...,C_1) \quad (2)\\
&= \sum_{S}\left[\pi(s_0)\prod_{t=1}^{T}a_{s_{t-1}s_t}\prod_{t=0}^{T}\frac{p(s_t|\mathbf{x}_t)p(\mathbf{x}_t)}{p(s_t)}\right]\prod_{i=1}^{n}p(C_i|C_{i-1},C_{i-2},...,C_1) \quad (3)
\end{aligned}
$$

where $\mathbf{X} = \{\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T\}$ is a $(T+1)$-frame observation sequence of one text line image. $p(\mathbf{X}|\mathbf{C})$, which can be called the character model, is the conditional probability of $\mathbf{X}$ given $\mathbf{C}$ corresponding to a sequence of HMMs with the corresponding hidden state sequence $S = \{s_0, s_1, s_2, ..., s_T\}$. Each HMM with a set of states represents one character class. With HMMs, the $p(\mathbf{X}|\mathbf{C})$ can be decomposed in the frame level: $\pi(s_0)$ is the initial state probability, $a_{s_{t-1}s_t}$ is the state transition probability from frame $t-1$ to t, $p(\mathbf{x}_t|s_t)$ is the output probability of $\mathbf{x}_t$ given s_t, $p(s_t)$ is the prior probability of state s_t estimated from the training set, $p(s_t|\mathbf{x}_t)$ is the posterior probability of state s_t given $\mathbf{x}_t$, and $p(\mathbf{x}_t)$ is independent of the character sequence. As mentioned in [17], GMM can be used to calculate $p(\mathbf{x}_t|s_t)$ in Eq. (2) for the GMM-HMM system, while DNN/CNN can be adopted to compute $p(s_t|\mathbf{x}_t)$ in Eq. (3) for the DNN-HMM/CNN-HMM system.

Meanwhile, $p(\mathbf{C})$, namely the language model, is the probability of an n-character sequence $\mathbf{C} = \{C_1, C_2, ..., C_n\}$ and can be decomposed as $\prod_{i=1}^{n} p(C_i|C_{i-1}, C_{i-2}, ..., C_1)$. However, as the number of these values V^i for even a moderate vocabulary size V is too large to be accurately estimated. The so-called N-gram LM can not realistically depend on all $i-1$ conditioning histories $C_1, C_2, ..., C_{i-1}$ to compute the term $p(C_i|C_{i-1}, C_{i-2}, ..., C_1)$. Obviously, a higher order N leads to a more powerful language model which can significantly improve the recognition accuracy. In this work, the SRILM toolkit [46] is employed to generate a 5-gram LM. To further enhance the ability of the LM, we linearly interpolate a standard NLM with an RNNLM to form an HLM.

In the training stage, we first build the conventional GMM-HMM system as in [17]. Then, the state-tying GMM-HMM system (GMM-PHMM) can be generated using the proposed decision-tree algorithm to greatly reduce the total number of states, i.e., the dimension of the CNN output layer. Meanwhile, state-level forced-alignment is conducted to obtain frame-level labels for the subsequent CNN cross-entropy training. After the conventional CNN is trained, a series of adaptation layers with the writer codes as the input are appended in parallel to form the

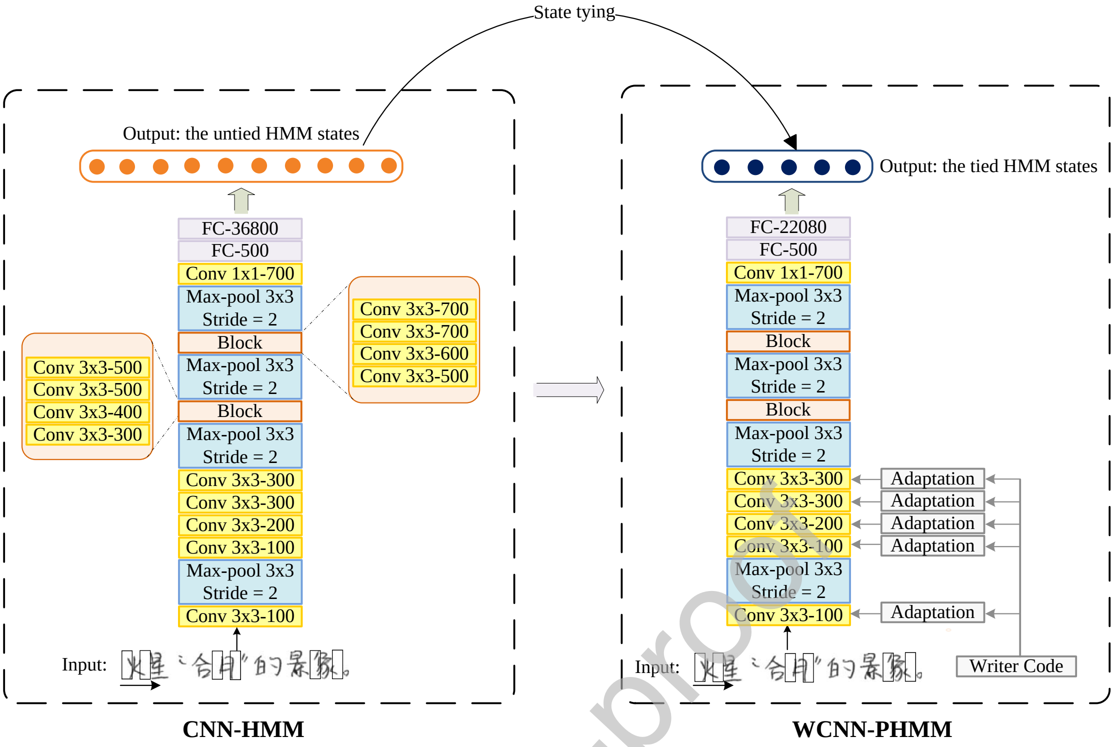


Fig. 3. Comparison between the conventional CNN-HMM and the proposed WCNN-PHMM.

WCNN. With writer-specific training data, the writer codes and the parameters of the adaptation layers for WCNN are jointly optimized.

In the testing stage, with the initial recognition results from the first-pass decoding using CNN-PHMM, the codes of unknown writers are learned from random initialization via WCNN for the second-pass decoding. This process could be iteratively conducted for multipass decoding to refine the recognition results and the writer codes.

IV. WCNN-PHMM

Fig. 3 illustrates two main innovations of our proposed WCNN-PHMM architecture over the conventional CNN-HMM in [17], namely, the compact design of the output layer and writer-aware convolutional layers. In the following subsections, we elaborate three basic components of WCNN-PHMM: convolutional neural network, state tying for PHMM, and writer code-based adaptive training for WCNN. In order to help readers understand clearly, in Table I, we first describe acronyms that are frequently used in this paper. For example, according to Table I, the

TABLE I

ACRONYM DESCRIPTION

Acronym	Description
CNN	Convolutional neural network
WCNN	Writer-aware convolutional neural network
TCNN	Tied-state convolutional neural netwotk
HMM	Hidden Markov model
PHMM	Parsimonious hidden Markov model
CER	Character error rate

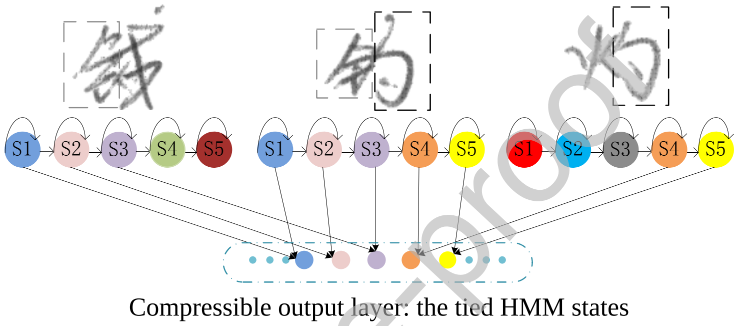


Fig. 4. Illustration of tied state design for CNN output layer.

system WCNN-PHMM means characters are modeled by the PHMM where the WCNN is used to compute the posterior probabilities of tied-states.

A. Convolutional neural network

As shown in Fig. 3, CNN [47] successively consists of stacked convolutional layers (Conv) optionally followed by spatial pooling, one or more fully connected layer (FC) and a softmax layer. For the convolutional and pooling layers, each layer is a three-dimensional tensor organized by a set of planes called feature maps, while the fully connected layer and the softmax layer are the same as those in the conventional DNN. Inspired by the locally sensitive, orientation-selective neurons in the visual system of cats [48], each unit in a feature map is constrained to connect a local region in the previous layer, which is called the local receptive field. Two contiguous local receptive fields are usually s pixels (referred as stride) shifted in a certain direction. Usually, all units in the same feature map of a convolutional layer share a set of weights, each computing a dot product between its weights and the local receptive field in the previous layer and then followed by batch normalization (BN) [49] and a nonlinear activation function. Meanwhile, the units in a pooling layer perform a spatial average or max operation for their local receptive field to reduce spatial resolution and noise interference. Accordingly, the key information for identifying the pattern is retained. We formalize operations in a convolutional layer as:

$$\boldsymbol{O}_{i,j,k} = f(\mathrm{BN}(\sum_{m,n,l} \boldsymbol{I}_{(i-1)\times s+m,(j-1)\times s+n,l} \boldsymbol{W}_{m,n,k,l} + \boldsymbol{B}_k)) \tag{4}$$

where $\boldsymbol{I}_{i,j,k}$ is the value of the input unit in feature map k at row i and column j while $\boldsymbol{O}_{i,j,k}$ corresponds to the output unit, $\boldsymbol{W}_{m,n,k,l}$ is the connection weight between a unit in feature map k of the output and a unit in channel l of the input, with an offset of m rows and n columns between the output unit and the input unit. $\boldsymbol{B}_k$ is the k-th value of bias vector $\boldsymbol{B}$ for all units in the feature map k. BN is used to handle the change of the distribution in each layer by simply normalizing the input of layers [49], which can yield an obvious improvement in the HCTR task [17]. f is a nonlinear function, i.e., ReLU [50], used in this study.

B. State tying for PHMM

Fig. 4 illustrates the main motivation of our proposed algorithm to tie HMM states, namely, fully utilizing the partial similarities of characters (e.g., radicals). State tying is completed using a binary decision tree in which the question for each node of the tree is automatically generated by a data-driven algorithm. If each character is represented by a 5-state HMM, then 5 trees are built, with each representing one positioned HMM state to cluster all character classes. Suppose S is the set of HMM states in one nonleaf node of a tree and $L(\mathrm{S})$ is the log-likelihood of S generating the training dataset with F frames. Then, by the attached question q, which is selected from an automatically generated question set, this node with S is split into two children nodes, namely, a left node with a subset S_l and a right node with a subset S_r, to maximize the log-likelihood increase with respect to q in the current node:

$$\Delta L = L(\mathrm{S}_\mathrm{l}(q)) + L(\mathrm{S}_\mathrm{r}(q)) - L(\mathrm{S}) \tag{5}$$

where $L(\mathrm{S})$, $L(\mathrm{S}_\mathrm{l}(q))$ and $L(\mathrm{S}_\mathrm{r}(q))$, are log-likelihoods of the state set in the current node, its left node and its right node, respectively. Based on the assumptions that all tied states in S share a common mean $\boldsymbol{\mu}(\mathbf{S})$ and variance $\boldsymbol{\Sigma}(\mathbf{S})$, and the tying states does not change the frame/state alignment, a reasonable approximation of $L(\mathbf{S})$ via Gaussian output distribution $\mathcal{N}$ is given by:

$$\begin{aligned} L(\mathbf{S}) &= \sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f) \ln \mathcal{N}(\boldsymbol{o}_f; \boldsymbol{\mu}(\mathbf{S}), \boldsymbol{\Sigma}(\mathbf{S})) \\ &= -\frac{1}{2}\sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f)[D\ln(2\pi) + \ln|\boldsymbol{\Sigma}(\mathbf{S})| + D_M^2(\boldsymbol{o}_f)] \end{aligned} \tag{6}$$

where $D_M(\boldsymbol{o}_f)$ is the Mahalanobis distance:

$$D_M(\boldsymbol{o}_f) = \sqrt{(\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))^\top (\boldsymbol{\Sigma}(\mathbf{S}))^{-1} (\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))}. \tag{7}$$

In Eq. (6), $\gamma_s(\boldsymbol{o}_f)$ is the posterior probability of the D-dimensional feature vector $\boldsymbol{o}_f$ at the f-th frame that is generated by state s. $\boldsymbol{\mu}(\mathbf{S})$ and $\boldsymbol{\Sigma}(\mathbf{S})$ can be estimated as:

$$\mu(\mathbf{S}) = \frac{\sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f)\boldsymbol{o}_f}{\sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f)} \tag{8}$$

$$\boldsymbol{\Sigma}(\mathbf{S}) = \frac{\sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f)(\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))(\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))^\top}{\sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f)}. \tag{9}$$

Using Eq. (9), we can have the following derivation for the last item in Eq. (6):

$$\begin{aligned} &\sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f) D_M^2(\boldsymbol{o}_f) \\ =&\sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f) \mathrm{Tr}\{(\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))^\top (\boldsymbol{\Sigma}(\mathbf{S}))^{-1} (\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))\} \\ =&\sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f) \mathrm{Tr}\{(\boldsymbol{\Sigma}(\mathbf{S}))^{-1} (\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))(\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))^\top\} \\ =&\mathrm{Tr}\{(\boldsymbol{\Sigma}(\mathbf{S}))^{-1} \sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f)(\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))(\boldsymbol{o}_f - \boldsymbol{\mu}(\mathbf{S}))^\top\} \\ =&\mathrm{Tr}\{(\boldsymbol{\Sigma}(\mathbf{S}))^{-1}\boldsymbol{\Sigma}(\mathbf{S})\} \sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f) = D\sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f) \end{aligned} \tag{10}$$

where Tr$\{\cdot\}$ denotes the trace of a square matrix. If we further define the notation:

$$\gamma(\mathbf{S}) = \sum_{f=1}^{F}\sum_{s\in\mathbf{S}} \gamma_s(\boldsymbol{o}_f) \tag{11}$$

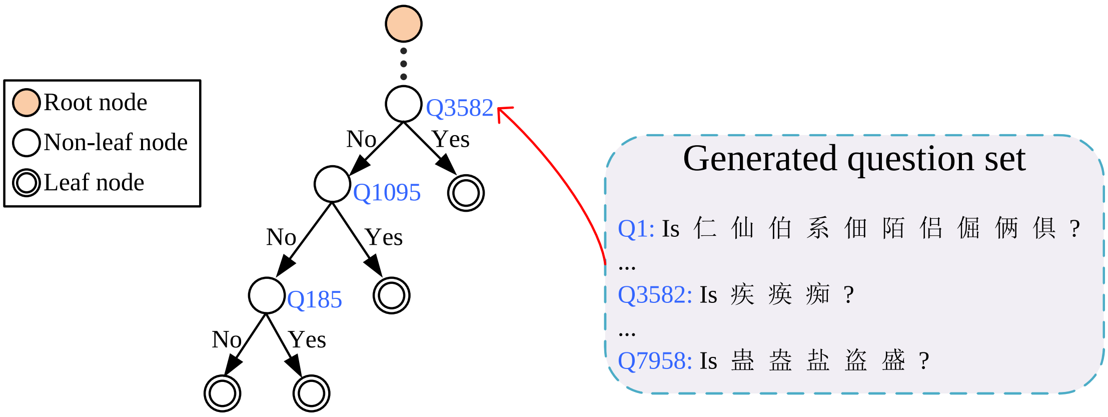


Fig. 5. Fraction of a generated tree for the first state of a 5-state HMM.

Then, Eq. (6) can be rewritten as:

$$L(\mathbf{S}) \quad = \quad -\frac{1}{2}\gamma(\mathbf{S})[\ln|\mathbf{\Sigma}(\mathbf{S})|+D+D\ln(2\pi)] \tag{12}$$

Thus, the log-likelihood $L(\mathbf{S})$ depends only on the pooled state occupancy $\gamma(\mathbf{S})$ and the pooled state variance $\mathbf{\Sigma}(\mathbf{S})$. Both could be calculated from the saved parameters of state occupancy counts, means, and variances for all HMM states during the preceding Baum-Welch re-estimation.

Initially, all corresponding states are placed in the root node of a tree. Then, the above algorithm is conducted in a top-down manner to build this binary tree until reaching to a fixed threshold. Finally, a merge operation of leaf nodes is conducted using a minimum priority queue in a bottom-up manner by computing the log-likelihood decrease to reach the target tied-state number.

To generate the question set, all feature frames of characters are placed in the root node of a binary decision tree and then a k-means ($k = 2$) algorithm is used to find an optimal partition, which aims to maximize the log-likelihood of frames under the assumption of a single Gaussian distribution. This procedure is conducted in a top-down manner until each node only contains one character class. One question of a nonleaf node can be obtained from all reachable leaves of this node. All questions form our question set for the state tying. There are 5 trees in total, as each character is modeled by a 5-state HMM. In Fig. 5, a fraction of a generated tree for the first state is illustrated.

In Table II, we summarize the differences of state tying between HCTR and speech recognition (SR). First, the original signal in HCTR is two-dimension image and the signal is one-dimension

TABLE II

THE DIFFERENCES OF STATE TYING IN HCTR AND SR

	HCTR	SR
Original Signal	Two dimension	One dimension
Object	The states of characters being in the same position	The states of tri-phones with the same central phone
Motivation	Existing similar radicals among characters	Data sparseness problem of tri-phone
Categories	Tens of thousands	Hundreds
Question Set	Data driven	Date driven or Artificial rules

speech in SR. Second, the motivation of state tying in HCTR is to overcome the difficulty of training and decoding in CNN-HMM due to many similar radicals among tens of thousands of characters while the state tying in SR is introduced for the data sparseness problem of tri-phone. Third, considering the ways of modeling in HCTR, we only tie the states of characters being in the same position to capture similar radicals more accurately. For SR, the state tying is usually conducted on the states of tri-phones with the same central phone. Finally, for HCTR, the question set used in state tying totally depends on the character based features while the question set in SR can be predefined artificially according to pronunciation characteristics.

C. Adaptive training for WCNN based on writer code

As shown in Fig. 3, the conventional CNN used for offline HCTR does not explicitly incorporate the writer information in both training and testing stages. However, the writing style could play an essential role in the final CER as an irrelevant variability to recognize the character class. Accordingly, a learnable vector (writer code) is introduced to represent the writer style of each writer. If we consider the CNN architecture to integrate both feature extraction and classifier implicitly, then the proposed ingenious design of WCNN in Fig. 3 seems like a joint feature and model adaptive training strategy.

To guide the CNN with writer information, two key components, i.e, writer codes and adaptation layers, are randomly initialized and can be optimized using the back-propagation algorithm. The code of the r-th writer is a G-dimensional vector $\boldsymbol{V}^r$ directly connected with all adaptation layers. The p-th adaptation layer can be represented by a $K \times G$ matrix $\boldsymbol{A}^p$. The writer code is fed into the adaptation layer and transformed into a new vector $\boldsymbol{Q}^{r,p}$:

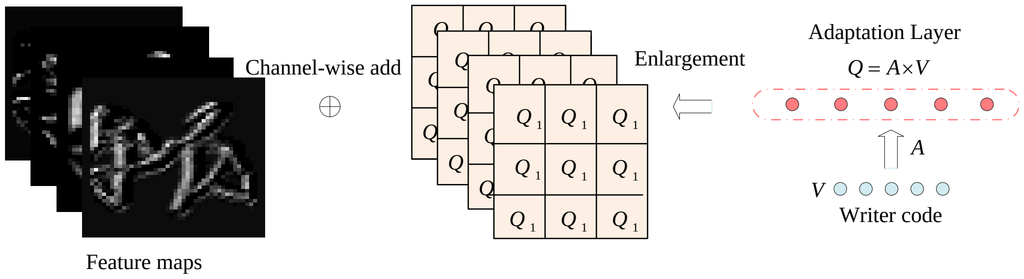


Fig. 6. Illustration of convolutional layer with writer code in WCNN.

$$\boldsymbol{Q}^{r,p} = \boldsymbol{A}^p \boldsymbol{V}^r. \tag{13}$$

With the writer information $\boldsymbol{Q}$, the corresponding p-th convolutional layer of WCNN can be reformulated as:

$$\boldsymbol{O}^{r,p}_{i,j,k} = f(\mathrm{BN}(\boldsymbol{M}^p_{i,j,k} + \boldsymbol{Q}^{r,p}_k)) \tag{14}$$

where

$$\boldsymbol{M}^p_{i,j,k} = \sum_{l,m,n} \boldsymbol{I}^p_{(i-1)\times s+m,(j-1)\times s+n,l} \boldsymbol{W}^p_{m,n,k,l} + \boldsymbol{B}^p_k. \tag{15}$$

In Eqs. (14-15), $\boldsymbol{I}^p_{i,j,k}$, $\boldsymbol{O}^p_{i,j,k}$, $\boldsymbol{W}^p_{m,n,k,l}$, and $\boldsymbol{B}^p_k$ are the corresponding items like in Eq. (4) for the p-th convolutional layer. The writer information $\boldsymbol{Q}^{r,p}_k$, which is the k-th value of bias vector $\boldsymbol{Q}^{r,p}$, is newly added as a bias to build writer-aware convolutional layers. The key innovation of the WCNN architecture is illustrated in Fig. 6.

Suppose we use P adaptation layers with the parameter set $\boldsymbol{A} = \{\boldsymbol{A}^p | p = 1, ..., P\}$. In the training stage, a well-trained CNN-HMM or CNN-PHMM system is first used to initialize WCNN with the writer-independent parameter set $\boldsymbol{\Lambda}$. Assume we have R writers in the training dataset with the corresponding writer code set $\boldsymbol{V} = \{\boldsymbol{V}^r | r = 1, ..., R\}$. Then, the cross-entropy criterion is minimized with respect to writer-aware parameter set $\{\boldsymbol{A}, \boldsymbol{V}\}$ in WCNN:

$$E(\boldsymbol{A}, \boldsymbol{V}) = -\sum_{t=1}^{N_B} \log p(s_t | \boldsymbol{X}_t, \boldsymbol{\Lambda}, \boldsymbol{A}, \boldsymbol{V}) \tag{16}$$

where the WCNN output $p(s_t | \boldsymbol{X}_t, \boldsymbol{\Lambda}, \boldsymbol{A}, \boldsymbol{V})$ is the posterior probability of the reference state s_t given the input image $\boldsymbol{X}_t$ within the sliding window. N_B is the minibatch size using stochastic gradient decent algorithm. In our implementation, we process the text lines one by one. Thus,

N_B equals the number of frames of each text line. Please note that, for each frame $\boldsymbol{X}_t$, the input parallel writer code vector is selected from $\boldsymbol{V}$ with the writer-aware information. With the random initialization, we jointly update $\{\boldsymbol{A}, \boldsymbol{V}\}$ using backpropagation and SGD:

$$\begin{aligned}\boldsymbol{A}^p &\leftarrow \boldsymbol{A}^p - \varepsilon^{\text{tr}}\frac{\partial E(\boldsymbol{A}, \boldsymbol{V})}{\partial \boldsymbol{A}^p}\\ \boldsymbol{V}^r &\leftarrow \boldsymbol{V}^r - \varepsilon^{\text{tr}}\frac{\partial E(\boldsymbol{A}, \boldsymbol{V})}{\partial \boldsymbol{V}^r}\end{aligned} \tag{17}$$

where ε^{tr} is the step size in the training stage, which is initially set to 0.001 and decreased by a factor of 0.8 after updating with 5 million frames. We summarize the training procedure of WCNN in Algorithm 1.

Algorithm 1 The training procedure of WCNN.

```
Input:
    The writer-independent parameter set $\boldsymbol{\Lambda}$ is generated using conventional CNN-HMM/CNN-PHMM systems;
    Randomly initialize the writer-aware parameter set $\{\boldsymbol{A}, \boldsymbol{V}\}$;
    Prepare the minibatch level training dataset with the state label and writer information in each frame,
1: Randomly select one minibatch and set the input writer code of each frame using writer information and $\boldsymbol{V}$.
2: Calculate all required derivatives using backpropagation.
3: Update the adaptation layer parameters and writer codes $\{\boldsymbol{A}, \boldsymbol{V}\}$ using Eq. (17).
4: Go to step 1 until the convergence condition is satisfied.
Output: The parameter set of WCNN $\{\boldsymbol{\Lambda}, \boldsymbol{A}, \boldsymbol{V}\}$
```

In the recognition stage, for the data of an unknown writer, a multipass decoding is conducted. In the first-pass decoding, we use only CNN-HMM/CNN-PHMM with the parameter set $\boldsymbol{\Lambda}$ to generate the recognition results that are adopted as the state labels for updating the writer code vector of this unknown writer in the next pass. In the second pass, we perform the adaptation by minimizing the cross-entropy criterion with respect to the writer code $\boldsymbol{V}^{\text{U}}$:

$$E'(\boldsymbol{V}^{\text{U}}) = -\sum_{t=1}^{N'_B} \log p(s_t^{\text{U}}|\boldsymbol{X}_t^{\text{U}}, \boldsymbol{\Lambda}, \boldsymbol{A}, \boldsymbol{V}^{\text{U}}). \tag{18}$$

Similar to Eq. (16), $\boldsymbol{X}_t^{\text{U}}$ is the t-th input frame of an unknown writer, while s_t^{U} is its corresponding state label from the first-pass recognition. The batch size N'_B refers to the number of frames of

Algorithm 2 The adaptation/recognition procedure of WCNN.

```
Input:
   Prepare the WCNN parameter set $\{\mathbf{\Lambda}, \boldsymbol{A}\}$;
   Prepare the minibatch level dataset of an unknown writer;
   Randomly initialize the corresponding writer code $\boldsymbol{V}^{\mathrm{U}}$,
1: Generate the state labels via first-pass decoding using $\mathbf{\Lambda}$.
2: Perform the adaptation to refine $\boldsymbol{V}^{\mathrm{U}}$ using Eq. (19).
3: Conduct decoding using $\{\mathbf{\Lambda}, \boldsymbol{A}, \boldsymbol{V}^{\mathrm{U}}\}$ of WCNN.
4: Go to step 2 for alternative adaptation and recognition until a specified number of multipass decoding is reached.
Output: The writer code $\boldsymbol{V}^{\mathrm{U}}$ and recognition results
```

each text line. Please note that we do not use $\boldsymbol{V}^{\mathrm{U}}$ from the training stage and randomly initialize the code $\boldsymbol{V}^{\mathrm{U}}$ of the unknown writer. Accordingly, we can update $\boldsymbol{V}^{\mathrm{U}}$ as:

$$\boldsymbol{V}^{\mathrm{U}} \leftarrow \boldsymbol{V}^{\mathrm{U}} - \varepsilon^{\mathrm{ts}} \frac{\partial E'(\boldsymbol{V}^{\mathrm{U}})}{\partial \boldsymbol{V}^{\mathrm{U}}} \tag{19}$$

where $\varepsilon^{\mathrm{ts}}$ is the step size in the testing stage, which is set to 0.001. Then, we conduct a second-pass decoding using $\{\mathbf{\Lambda}, \boldsymbol{A}, \boldsymbol{V}^{\mathrm{U}}\}$ of WCNN. This adaptation and recognition processes could be alternatively and iteratively conducted until a specified number of multipass decoding is reached. We summarize the adaptation/recognition procedure of WCNN in Algorithm 2.

D. Hybrid language model

The HLM is linear interpolation of a traditional NLM and an RNNLM. Considering all calculations in Eq. (2) are performed in the logarithmic domain, the HLM is represented as:

$$\log p_{\mathrm{HLM}}(\mathbf{C}) = \omega \log p_{\mathrm{NLM}}(\mathbf{C}) + (1-\omega) \log p_{\mathrm{RNNLM}}(\mathbf{C}) \tag{20}$$

where the $p_{\mathrm{NLM}}(\mathbf{C})$ means the probability of an n-character sequence $\mathbf{C} = \{C_1, C_2, ..., C_n\}$ is computed based on NLM while the value of $p_{\mathrm{RNNLM}}(\mathbf{C})$ is obtained from RNNLM. ω is a hyperparameter to adjust the ratio between NLM and RNNLM. In the RNNLM, a simple RNN with three layers including input layer, hidden layer and output layer is used. At time step i, the input vectors consist of a 1-of-V coding $\boldsymbol{R}_i$ that represents the previous word C_{i-1}, and the previous hidden layer output $\boldsymbol{H}_{i-1}$. The output of the hidden layer is computed as:

$$\boldsymbol{H}_i = f(\boldsymbol{W}_{H,V}\boldsymbol{R}_i + \boldsymbol{W}_{H,H}\boldsymbol{H}_{i-1}) \tag{21}$$

TABLE III

THE INFORMATION OF THE CASIA-HWDB DATABASES.

#	Class	Writer	Text Line	Character Sample
HWDB1.0	3,837	420	-	1,592,978
HWDB1.1	3,834	300	-	1,145,074
HWDB2.0	1,222	419	20,495	540,468
HWDB2.1	2,310	300	17,292	429,926
HWDB2.2	1,331	300	14,443	383,153

where $\boldsymbol{W}_{H,V}$ and $\boldsymbol{W}_{H,H}$ are learnable matrices of size $H \times V$ and $H \times H$, respectively. The activation function f is sigmoid. In the output layer, using the history information $\boldsymbol{H}_i$, the probabilities of the predicted characters at time step i are estimated:

$$\boldsymbol{P}_i = g(\boldsymbol{W}_{V,H}\boldsymbol{H}_i) \tag{22}$$

g is the softmax function and $\boldsymbol{W}_{V,H}$ is a $V \times H$ learnable matrix. Naturally, for a predicted character C_i at time step i, we have the following equation:

$$p_{\text{RNNLM}}(C_i|C_{i-1}, C_{i-2}, ..., C_1) = \boldsymbol{P}_i(C_i). \tag{23}$$

Finally,

$$p_{\text{RNNLM}}(\mathbf{C}) = \prod_{i=1}^{n} p_{\text{RNNLM}}(C_i|C_{i-1}, C_{i-2}, ..., C_1) = \prod_{i=1}^{n} \boldsymbol{P}_i(C_i). \tag{24}$$

In this work, the dimension of the hidden layer is set to 300, the ω is 0.5 and the weights $\{\boldsymbol{W}_{H,V}, \boldsymbol{W}_{H,H}, \boldsymbol{W}_{V,H}\}$ in the RNNLM are optimized by using the truncated BPTT [52].

V. EXPERIMENTS

We designed a set of experiments to validate and explain the effectiveness of the proposed method for offline HCTR. All experiments were implemented with Kaldi [29] and Pytorch [30] toolkits using NVIDIA GeForce GTX 1080Ti GPUs.

A. Dataset and metrics

We conducted the experiments on a widely used database for HCTR released by the Institute of Automation of Chinese Academy of Sciences (CASIA) [53], [54]. To train the character models, both offline isolated handwritten Chinese character datasets (HWDB1.0 and HWDB1.1) and the training sets of offline handwritten Chinese text datasets (HWDB2.0, HWDB2.1, and HWDB2.2) were used. The detailed information, including the number of classes, writers, lines, and characters for each dataset, are shown in Table III. In total, 3,980 classes (Chinese characters, symbols, garbage) were formed with 4,091,599 samples. To train the language model, the training sets of offline handwritten Chinese text of HWDB2.0-2.2 and the news data downloaded from Internet are used. All the news data have been checked to exclude the text of the test set. The whole corpus contains approximately ten million characters. The ICDAR 2013 competition set with 60 writers unseen to the training dataset was adopted as the evaluation set [9]. The CER was computed as:

$$\text{CER} = \frac{N_\text{s} + N_\text{i} + N_\text{d}}{N} \tag{25}$$

where N is total number of character samples in the evaluation set. N_s, N_i and N_d denote the number of substitution errors, insertion errors and deletion errors, respectively. Firstly, to focus on character modeling, we did not use additional language models.

B. Experiments on state tying of PHMM

1) Comparison between CNN-HMM and CNN-PHMM: We first compared CNN-HMM with CNN-PHMM according to the best configuration in our previous work [17], i.e., there were 16 weight layers (14 Conv and 2 FC layers) and the number of channels increased from 100 to 700. The image patch of each frame was passed through a stack of 3×3 convolutional layers. After the last max pooling layer, a 1×1 convolutional layer was used to increase the nonlinearity of the net without more computation and memory than the other larger receptive fields. All convolutional layers were followed by the ReLU and the stride was 1, while the stride of all max pooling layers was 2 with a 3×3 window. The BN operation was equipped for the outputs before nonlinearity in every convolutional layer. The minibatch size was 1,000, the momentum was 0.9 and the weight decay was 0.0001. The learning rate was initially set to 0.01 and decreased by 0.92 after every 4,000 batches. Three epochs were conducted. All other parameters, such as frame length, frame shift, feature extraction for GMM-HMM, and parameters of GMM-HMM, were the same as those used in [17].

For CNN-HMM, we list the results of different settings of states per HMM in Table IV. The observation consistent with [17] was that the CER increased greatly from 5 states to 1 state due to the lack of adequate resolution. Notably, the number of output nodes of CNN was 3,980×5

TABLE IV
CER (%) COMPARISON BETWEEN CNN-HMM AND CNN-PHMM BASED ON DIFFERENT SETTINGS OF AVERAGE STATES PER HMM.

# of states per HMM	5	4	3	2	1
CNN-HMM	10.02	10.11	10.77	11.71	13.85
CNN-PHMM	10.02	9.44	9.54	9.91	11.61

TABLE V
PRACTICAL ISSUE COMPARISON OF DIFFERENT SETTINGS OF AVERAGE STATES PER HMM FOR THE CORRESPONDING CNN-PHMM SYSTEM IN TABLE IV. N_{M} AND N_{T} REPRESENT THE MODEL SIZE AND RUN-TIME LATENCY, RESPECTIVELY, WHICH ARE NORMALIZED BY THOSE OF CNN-HMM SYSTEM WITH 5 STATES PER HMM.

# of states per HMM	5	4	3	2	1
N_{M}	1	0.96	0.87	0.83	0.77
N_{T}	1	0.91	0.72	0.63	0.57

(19,900) for 5-state HMM, while the number of output nodes was 3,980 for 1-state HMM, which means that, the more states for each character, the more challenging it is to train CNN. Based on the optimal settings of the 5-state CNN-HMM system, we conducted the state-tying algorithm of our PHMM to reduce the average number of states per HMM. Interestingly, the performance of CNN-PHMM could improve when the average number of states equaled 3 or 4; however, if we kept reducing this number to 2 or 1, the performance declined. These observations implied that there was a tradeoff between the model resolution and the parameter redundancy. Moreover, the CER of CNN-PHMM was much lower than the CER of CNN-HMM for the same average state number, which indicated that CNN-PHMM achieved more reasonable state assignment among all character HMMs than CNN-HMM. Another advantage of CNN-PHMM is its more compact CNN output layer, which helps compress the CNN and accelerate the decoding process, as shown in Table V. Finally, for CNN-PHMM, an average 3 states was used as the default for the subsequent experiments, which not only achieved a much lower CER than the best configured CNN-HMM with 5 states but also yielded a much smaller model size and a faster decoding speed.

2) Analysis of state tying: In Fig. 7, we list representative examples of tied Chinese characters from positioned states 1 to 5 in our CNN-PHMM system. It was quite intuitive and reasonable that most of the tied Chinese characters shared the same or similar radicals although the state-

State	Characters	Similar radical
1	仁 仕 仙 估 佃 侣	亻
	樹 椎 槐 槛 栓 桅 梳	木
	圈 囚 园 困 围	囗
2	疥 疹 痒 痔 痹 瘁 瘴	疒
	赴 赵 赶 起 趁 超 趋	走
	财 败 贬 购	贝
3	砂 炒 纱 妙 抄 秒	少
	闯 闰 闺 润	门
	仑 仓 沦 沧	仑
4	氦 氨 氮 氯	气
	试 式 武	弋
	胳 骆 铬 赂 略 烙 路 咯 洛	各
5	邮 邯 邵 郡 都	阝
	砍 坎 饮 炊 吹 欢	欠
	炬 距 矩 拒 柜	巨

Fig. 7. Examples of tied Chinese characters with similar radicals.

tying process was purely data driven with diversified writing styles. This result could explain why there was a large amount of parameter redundancy in the conventional untied CNN-HMM model. We also give partial results of the data-driven question set in Fig. 8. In total, there were 7,938 questions generated. It could be observed that the related characters in one question were similar, which demonstrated the effectiveness of the k-means clustering algorithm. Overall, the proposed state-tying method has two advantages. First, because the total number of states corresponds to the size of the CNN output layer, having fewer categories will make CNN training easier and speed up the recognizer. Second, reducing parameter redundancy can potentially increase the number of training samples for the tied states from different characters.

For further analysis, we draw the learning curves during training for conventional CNN and tied-state CNN (TCNN) in Fig. 9. Obviously, the learning curve of TCNN was always below that of CNN. More interestingly, the gap between the two curves significantly increased in the beginning stage and then decreased to a relatively stable value as an increasing amount of training data was used. We believe that the compact design of the CNN output layer not only made the CNN model easier to train and more effective to classify but also fully utilized the training data by state tying.

Generated question set	
Number	Question
1	Is 仁 仙 伯 佃 陌 侣 倔 俩 俱 ?
2	Is 漳 潭 滓 谭 淖 ?
3	Is 马 鸣 呼 哗 哼 嘎 鸣 啤 ?
4	Is 肩 雇 扁 庸 ?
5	Is 植 栏 桂 桓 检 杜 枉 柱 ?
6	Is 客 害 宾 寄 寒 案 牢 穷 突 窖 宇 守 ?
7	Is 奖 桨 浆 裴 ?
8	Is 义 艾 又 叉 ?
9	Is 昭 眨 眯 睡 睦 睬 睹 瞅 瞎 瞒 旺 ?
…	…
7958	Is 蛊 盎 盐 盗 盛 ?

Fig. 8. Partial results of generated question set for tree-based state tying.

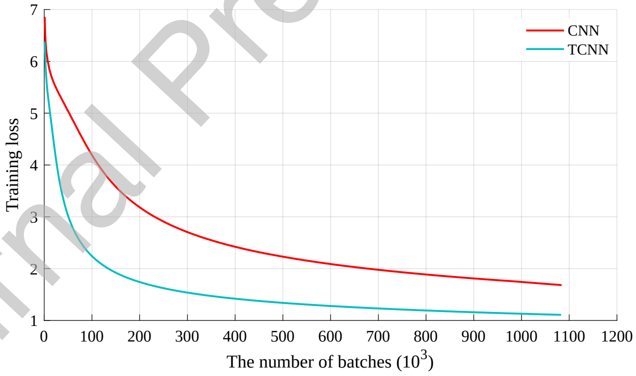


Fig. 9. Training loss comparison between CNN and TCNN.

C. Experiments on writer adaptive training for WCNN

1) The configuration of WCNN: As shown in Fig. 4, there are two key factors for writer-adaptive training: the number of adaptation layers P and the dimension of writer code G. The increase in the number of adaptation layers linking to the convolutional layers goes from input layer to output layer. Table VI compares different settings of adaptation layer number P and writer code dimension G in WCNN-PHMM. P=0 denotes the CNN-PHMM system without

TABLE VI

CER (%) COMPARISON OF DIFFERENT SETTINGS OF ADAPTATION LAYER NUMBER P AND WRITER CODE DIMENSION G IN WCNN-PHMM.

G	200							100	400
P	0	1	2	3	4	5	6	5	5
CER	9.54	9.29	9.17	9.04	8.99	8.96	8.96	9.05	9.02

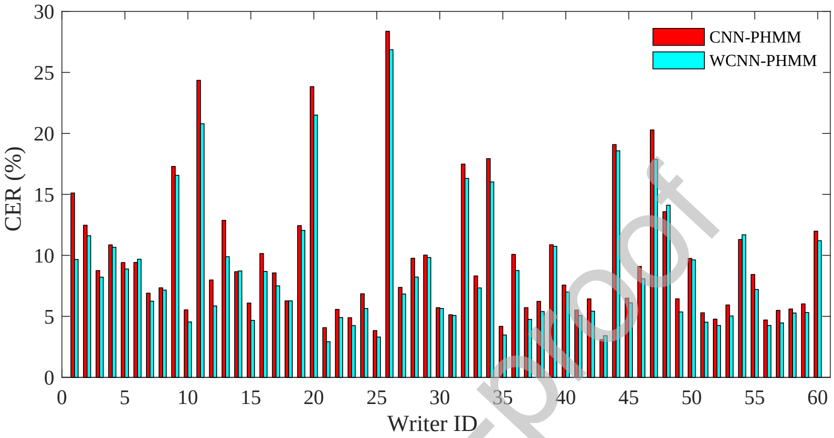


Fig. 10. CER (%) comparison between WCNN-PHMM and CNN-PHMM for each writer of the competition set.

writer adaptive training. Please note that second-pass decoding was adopted as a default for WCNN-PHMM. When the writer code dimension was fixed as 200, the CER decreased from 9.54% to 8.96% with P increasing from 0 to 5. The performance was saturated when more than 5 adaptation layers were used due to the limited adaptation data. Another interesting observation is that the performance of WCNN-PHMM was not sensitive to writer code dimension, with a good tradeoff of G=200. Thus, we use the configuration of P=5 and G=200 in the following experiments.

To further demonstrate the effectiveness of writer adaptive training, we make a CER comparison between WCNN-PHMM and CNN-PHMM for each writer in Fig. 10. Consistent improvements could be obtained for most of the 60 writers, and there were only 5 exceptions (No. 6, No. 14, No. 43, No. 48, No. 54). Especially for those writers with relatively high CERs, significant gains could be achieved, e.g., the CER was reduced from 15.11% to 9.66% for writer No. 1, with a relative CER reduction of 36.1%.

2) WCNN with/without state tying: In section V-C1, we illustrated that WCNN could yield additional gains over CNN on top of PHMM using state tying. In this section, as shown in Fig. 11, we compare the relative CER reduction (%) in WCNN over CNN with/without state tying for different settings of text lines on the competition set. For the CNN-HMM system without state tying, the best configured 5-state HMM in Table IV was used. In the competition set, the number of text lines for each writer ranged from 44 to 82. Overall, using all handwritten text lines of one writer for unsupervised adaptation, the CERs could be reduced from 10.02% to 9.55% (CNN-HMM vs. WCNN-HMM) and from 9.54% to 8.96% (CNN-PHMM vs. WCNN-PHMM). Those stable performance gains indicated that the proposed writer-adaptive training method was effective for systems with/without state tying (PHMM/HMM). Regarding the performance with respect to the amount of adaptation data, we observed that only 15 handwritten text lines for each writer on average could start to improve the recognition accuracy for unsupervised adaptation. When the number of text lines was reduced to 10, the relative CER reduction was limited, i.e., 0.5% and 1.1% for WCNN-PHMM and WCNN-HMM, respectively. Furthermore, when we continued to reduce the number of text lines to 5, the CERs increased compared with respective baselines. More interestingly, with increased adaptation data, the CER reduction in WCNN over CNN for the PHMM system with state tying became more significant than that for the HMM system without state tying, which implies that, as more handwritten data are collected from one writer, the proposed unsupervised adaptation via WCNN-PHMM can recognize handwritten text lines from this writer with more accuracy. Thus, the proposed WCNN-PHMM is a perfect demonstration of a compact model with adaptive capability.

3) Multiple-pass decoding of WCNN-PHMM: The basic intuition in the adaptation stage is better targets can promote the learning of the writer code and so produce beneficial feedback on the decoding results. By using the results of second-pass decoding based on WCNN-PHMM to generate better targets for the learning of the test writer codes, a third-pass decoding is conducted to get our final results. As shown in Table VII, the multiple-pass decoding can improve the recognition results (from 8.96% to 8.64%), which demonstrates that our intuition is right. We also list the run time comparison for different pass numbers. In order to make a fair comparison, all experiments here were evaluated on the same machine and we normalized the decoding time of first-pass to 1. The relative time consumption of n-pass (n=2,3) included two parts: the adaptation time and the decoding time. Although we could obtain a remarkable improvement via adaptation, the time consumption was linearly increased with the number of decoding passes.

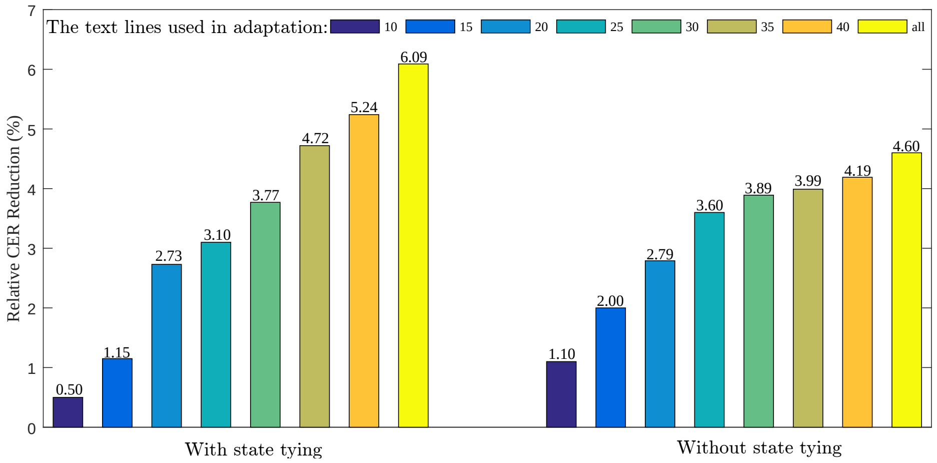


Fig. 11. The relative CER reduction (%) of WCNN over CNN with/without state tying for different settings of text lines on the competition set.

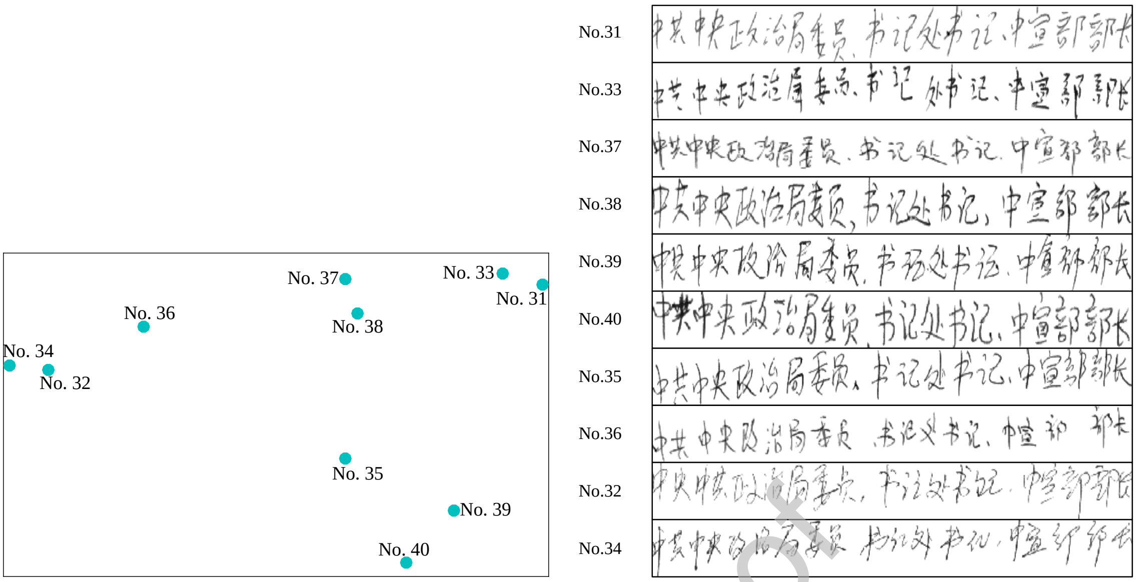
TABLE VII

CER (%) AND TIME CONSUMPTION COMPARISONS OF MULTIPLE-PASS DECODING OF WCNN-PHMM SYSTEM.

Multiple-pass Decoding	CER (%)	Decoding Time	Adaptation Time
First-pass (CNN-PHMM)	9.54	1.00	0.00
Second-pass	8.96	1.98	0.47
Third-pass	**8.64**	2.95	0.93

To address this problem, the acceleration of CNN and fast adaptation will be investigated in our future work.

4) Visualization analysis for writer code: To better understand why adaptation based on the writer code improves recognition performance, we adopted the t-SNE [55] technique to visualize the generated writer codes by reducing its dimension to 2. In Fig. 12(a), the distribution of several writer codes with the same transcripts on the competition set is shown. Correspondingly, we list their handwriting in Fig. 12(b). Interestingly, the distance between different writers in Fig. 12(a) was a strong indicator of the similarity of the writing styles of different writers. For example, all the distances of ID pairs (31, 33), (32, 34), and (39, 40) were small, while the corresponding writing styles for those pairs were quite similar, as observed from the handwritten text lines, which demonstrates that the learned writer code indeed carries the writer information.



(a) The t-SNE visualization of several writer codes.

(b) The corresponding handwriting examples of different writers.

Fig. 12. Visualization analysis of several writer codes on the competition set.

D. Comparison of different language models

Table VIII shows CER comparison of different language models. First, to demonstrate the scalability of our approach, we also conducted the corresponding 7360-class vocabulary experiments for different HMM systems. Please note that all the classes and writer data in HWDB1.0-HWDB1.2 were used in the 7360-class experiments rather than the subset listed in Table III that includes 3980-class experiments. Thus, the output layer sizes of CNN in the CNN-HMM system and WCNN in the WCNN-PHMM system were 36800 and 22080 for the 7360-class experiments, respectively, as illustrated in Fig. 3. Although the confusion among the 7360 classes is higher, the CER of the 7360-class CNN-HMM was slightly increased from 10.02% to 10.1%, thus demonstrating the robustness of the HMM system. A surprising observation was that the CER of the 7360-class CNN-PHMM was remarkably reduced from 9.54% in the 3980-class CNN-PHMM to 9.17%, which might be due to the larger amount of training data used for 7360-class being better utilized and shared among different classes (compared with the 3980-class case) due to the use of our state-tying algorithm. Correspondingly, the recognition performance of WCNN-PHMM was also improved from the 3980-class case to the 7360-class case, i.e, 8.60%, 8.42% for the second-pass decoding and the third-pass decoding, respectively.

TABLE VIII

CER (%) COMPARISON OF DIFFERENT LANGUAGE MODELS.

Method	Vocabulary	Without LM	NLM	HLM
CNN-HMM	3980	10.02	3.72	3.54
	7360	10.1	3.82	3.58
CNN-PHMM	3980	9.54	3.57	3.44
	7360	9.17	3.52	3.35
WCNN-PHMM	3980	8.64	3.39	3.27
	7360	8.42	3.33	3.17

Second, by adding a language model, a great improvement could be obtained for all the systems. Besides, compared with the NLM, all systems that use the HLM performed better, e.g, a relative CER reduction of 6.3%, 4.8% and 4.8% could be obtained in the 7000-class CNN-HMM, CNN-PHMM and WCNN-PHMM, respectively. It is reasonable that a weak character model could benefit more from a powerful language model.

E. Overall comparison and error analysis

Table IX shows an overall comparison of our proposed method and other state-of-the-art methods without/with a language model on the ICDAR 2013 competition set. we list the state-of-the-art oversegmentation method heterogeneous CNN [7], CNNs-RNNLM [8] and the segmentation-free method SMDLSTM-CTC [15], CNN-ACE [16] in Table IX for comparison. With the same configuration of vocabulary size (4 more garbage classes adopted in our HMM system), the proposed WCNN-PHMM yielded the best performance whether a language model was employed or not. Moreover, as shown in Table VIII, by using a powerful language model (HLM), the CNN-HMM, CNN-PHMM with one-pass decoding still could outperform the other methods.

For error analysis, we provide two examples in Fig. 13. In the left part of the figure, the conventional CNN-HMM misrecognized the first character of the text line, while CNN-PHMM generated the correct result. A reasonable explanation is that the left radical of the character in the brown box became easier to recognized because state tying could potentially learn the parameters better than the radical with more shared training samples from other characters. In the right of the figure, CNN-PHMM made a substitution error (red), while WCNN-PHMM could correct this mistake. Arguably, even humans could confuse this handwritten character in isolation without any prior knowledge. However, by learning the writing style of this particular writer

TABLE IX

PERFORMANCE COMPARISON OF OUR PROPOSED METHOD AND OTHER STATE-OF-THE-ARTS METHODS WITHOUT/WITH LANGUAGE MODELS ON THE 2013 ICDAR COMPETITION SET.

Method	Vocabulary	Without LM	With LM
WCNN-PHMM	3980	8.64	3.27
	7360	8.42	3.17
Wu *et al.* [15]	2672	9.98	7.39
	7356	13.36	9.62
Wang *et al.* [7]	7356	11.21	5.98
Wu *et al.* [8]	7356	-	3.80
Xie *et al.* [16]	7357	8.75	3.78

using the writer code, our WCNN-PHMM could correctly recognize it. Besides, the HMM-based approaches can assign each image frame to a certain state belonging to a character. Once the process of recognition is completed, the segmentation information between different characters can be naturally found. Fig. 14 shows the segmentation results of different HMM-based systems, i.e. CNN-HMM, CNN-PHMM and WCNN-PHMM. The red lines were the boundaries of different characters. For many characters such as the characters within the green dotted boxes, the CNN-PHMM and WCNN-PHMM provided more accurate boundaries than the CNN-HMM. For characters within the blue dotted boxes, we observed that the WCNN-PHMM could still give the right boundaries while the CNN-PHMM and CNN-HMM failed.

Finally, in Figs. 15(a) and 15(b), we explain and analyze the scores of the reference states of the underlying characters from the CNN outputs for CNN-HMM, CNN-PHMM, and WCNN-PHMM. Fig. 15(a) shows the comparison of the state posterior probability (SPP) of the frames for the reference character class in the brown box of Fig. 13. CNN-PHMM consistently generated higher SPPs than CNN-HMM for all frames of the sequence. Similarly, in Fig. 15(b), corresponding to the character class in the red box of Fig. 13, WCNN-PHMM always yielded higher SPPs than CNN-PHMM.

VI. CONCLUSION

In this study, we propose a novel WCNN-PHMM architecture for offline handwritten Chinese text recognition to handle two key issues: the large vocabulary of Chinese characters and the diversity of writing styles. By combining parsimonious HMM based on state tying and unsupervised adaptation based on writer code, our new approach demonstrates its superiority to

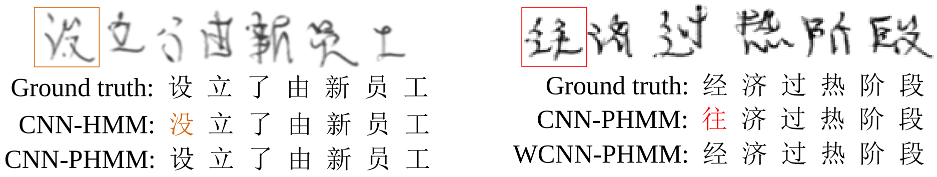

Ground truth: 设 立 了 由 新 员 工
CNN-HMM: 没 立 了 由 新 员 工
CNN-PHMM: 设 立 了 由 新 员 工
Ground truth: 经 济 过 热 阶 段
CNN-PHMM: 往 济 过 热 阶 段
WCNN-PHMM: 经 济 过 热 阶 段

Fig. 13. Two examples of recognition results for different HMM systems.

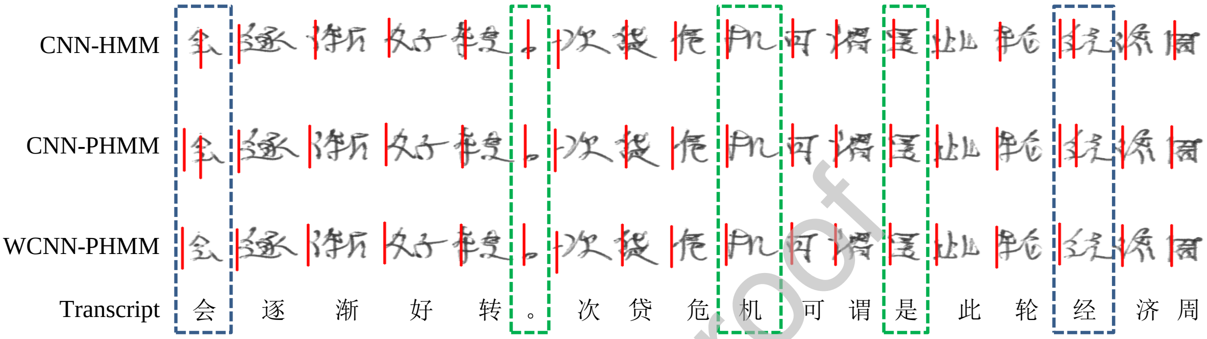
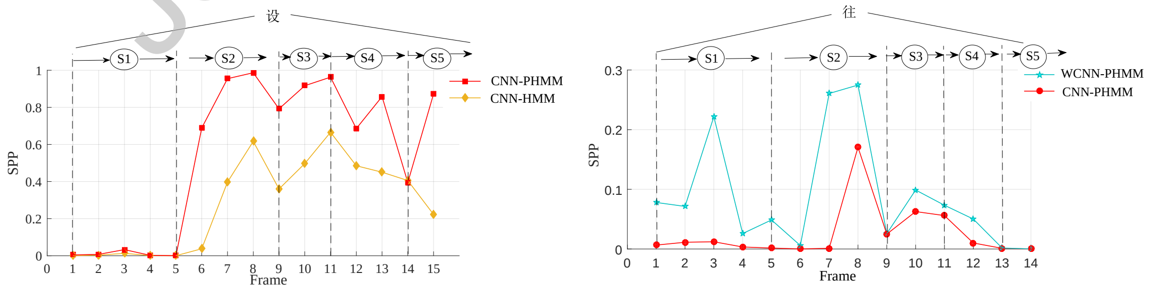


Fig. 14. Comparison of segmentation results of different HMM systems.

other state-of-the-art approaches according to both experimental results and analysis. However, current code-based adaptation simply depends on the backpropagation of network, which means adequate data is important. Besides, the 1-D HMM can not provide up-and-down information of characters. For future work, we will investigate the meta-learning to reduce dependence on data in adaptation and a more advanced way by using 2D-HMM to achieve recognition and segmentation. Furthermore, we will aim to accelerate the CNN to reduce decoding time.



(a) Comparison of reference state posterior probability of the frames for CNN-HMM and CNN-PHMM.

(b) Comparison of reference state posterior probability (SPP) of the frames for CNN-PHMM and WCNN-PHMM.

Fig. 15. Comparison of reference state posterior probability (SPP) for different HMM systems.

ACKNOWLEDGMENTS

This work was supported in part by the National Key R&D Program of China under contract No. 2017YFB1002202, the National Natural Science Foundation of China under Grant Nos. 61671422 and U1613211, the Key Science and Technology Project of Anhui Province under Grant No. 17030901005, and the MOE-Microsoft Key Laboratory of USTC.

REFERENCES

[1] H. Fujisawa, "Forty years of research in character and document recognition–an industrial perspective," *Pattern Recognition*, Vol. 41, No. 8, pp.2435-2446, 2008.

[2] C.-L. Liu and L. Yue, "Advances in chinese document and text processing," *World Scientific*, Vol. 2, 20018.

[3] Z.-C. Xie, Z.-H. Sun, L.-W. Jin, H. Ni and T. Lyons, "Learning spatial-semantic context with fully convolutional recurrent network for online handwritten Chinese text recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 40, No. 8, pp.1903-1917, 2017.

[4] X.-D. Zhou, D.-H. Wang, F. Tian, C.-L. Liu and M. Nakagawa, "Handwritten Chinese/Japanese text recognition using semi-Markov conditional random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 35, No. 10, pp.2413-2426, 2013.

[5] N.-X. Li and L.-W. Jin, "A Bayesian-based probabilistic model for unconstrained handwritten offline Chinese text line recognition," *Proc. IEEE SMC*, 2010, pp. 3664-3668.

[6] Q.-F. Wang, F. Yin, and C.-L. Liu, "Handwritten Chinese text recognition by integrating multiple contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 34, No. 8, pp.1469-1481, 2012.

[7] S. Wang, L. Chen, L. Xu, W. Fan, J. Sun, and S. Naoi, "Deep Knowledge Training and Heterogeneous CNN for Handwritten Chinese Text Recognition," *Proc. ICFHR*, 2016, pp.84-89.

[8] Y.-C. Wu, F. Yin, and C.-L. Liu, "Improving handwritten Chinese text recognition using neural network language models and convolutional neural network shape models," *Pattern Recognition*, Vol. 65, pp.251-264, 2017.

[9] F. Yin, Q.-F. Wang, X.-Y. Zhang, and C.-L. Liu, "ICDAR 2013 Chinese handwriting recognition competition," *Proc. ICDAR*, 2013, pp.1464-1470.

[10] T.-H. Su, T.-W. Zhang, D.-J. Guan, and H.-J. Huang, "Off-line recognition of realistic Chinese handwriting using segmentation-free strategy," *Pattern Recognition*, Vol. 42, No. 1, pp.167-182, 2009.

[11] R. Messina and J. Louradour, "Segmentation-free handwritten Chinese text recognition with LSTM-RNN," *Proc. ICDAR*, 2015, pp.171-175.

[12] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for improved unconstrained handwriting recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 31, No. 5, pp.855-868, 2009.

[13] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," *Proc. ICML*, 2006, pp. 369-376.

[14] D. Suryani, P. Doetsch, and H. Ney, "On the benefits of convolutional neural network combinations in offline handwriting recognition," *Proc. ICFHR*, 2016.

[15] Y.-C. Wu, F. Yin, Z. Chen, and C.-L. Liu, "Handwritten Chinese text recognition using separable multi-dimensional recurrent neural network," *Proc. ICDAR*, 2017, pp.79-84.

[16] Z.-C. Xie, Y.-X. Huang, Y.-Z. Zhu, L.-W. Jin, Y.-L. Liu and L.-L. Xie, "Aggregation cross-entropy for sequence recognition," *Proc. CVPR*, 2019, pp.6538-6547.

[17] Z.-R. Wang, J. Du, W.-C. Wang, J.-F. Zhai, and J.-S. Hu, “ A comprehensive study of hybrid neural network hidden Markov model for offline handwritten Chinese text recognition,” *IJDAR*, Accpted.

[18] S. Katz, “Estimation of probabilities from sparse data for the language model component of a speech recognizer,” *IEEE Trans. Acoust. Speech Signal Processs.*, Vol. 35, No. 3, pp.400-401, 1987.

[19] T. Mikolov, M. Karafit, L. Burget, J. ernock, and S. Khudanpur, “Recurrent neural network based language model,” *Proc. INTERSPEECH*, 2010, pp.1045-1048.

[20] C. Bucilua, R. Caruana and A. Niculescu-Mizil, “Model compression,” *Proc. KDD*, 2006.

[21] L.-C. Yan, Y.-S. Bengio, and G. Hinton “Deep learning,” *Nature*,Vol. 521, No. 7553, pp. 426, 2015.

[22] X. Zhang, J, Zou, K.-M. He and J. Sun “Accelerating very deep convolutional networks for classification and detection,” *PAMI*,Vol. 38, No. 10, pp. 1943-1955, 2016.

[23] Y.-H. He, X.-Y. Zhang and J. Sun “Channel pruning for accelerating very deep neural networks,” *Proc. ICCV*, 2017.

[24] C. Leng, H. Li, S.-H. Zhu and R. Jin, “Extremely low bit neural network: Squeeze the last bit out with admm,” *arXiv:1707.09870*, 2017.

[25] X.-Y. Zhang, X.-Y. Zhou, M.-X. Lin and J. Sun, “Shufflenet: An extremely efficient convolutional neural network for mobile devices,” *arXiv:1707.01083*, 2017.

[26] S.-J. Young, J.-J. Odell and P.-C. Woodland ‘Tree-based state tying for high accuracy acoustic modelling,” *Proc. workshop on Human Language Technology*, pp. 307-312, 1994.

[27] S. Young *et al.*, The HTK Book (Revised for HTK version 3.4.1), Cambridge University, 2009.

[28] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, “Deep neural networks for acoustic modeling in speech recognition,” *IEEE Signal Processing Magazine*, Vol. 29, No. 6, pp.82-97, 2012.

[29] D. Povey, A. Ghoshal, et al., “The kaldi speech recognition toolkit,” *Proc. ASRU*, 2011.

[30] P. Adam, et al., “Automatic differentiation in pytorch,” *NIPS-W*, 2017.

[31] W.-C. Wang, J. Du and Z.-R. Wang, “Parsimonious HMMs for offline handwritten Chinese text recognition,” *Proc. ICFHR*, 2018.

[32] C.-J. Leggetter and P.-C. Woodland, “Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models,” *Proc. Computer speech and language*, pp. 171-185, 1995.

[33] Sinno, J.-L pan and Q. Yang “A survey on transfer learning,” *Proc. IEEE Transactions on knowledge and data engineering*, Vol. 22, No. 10, pp. 1345-1359, 2010.

[34] M.-L. Yu, P.C.-K. Kwok, C.-H. Leung and K.-W. Tse, “Segmentation and recognition of Chinese bank check amounts,” *IJDAR*, Vol. 3, pp.207-217, 2001.

[35] B. Bridgeman, C. Trapani and Y. Attali, “Comparison of human and machine scoring of essays: Differences by gender, ethnicity, and country,” *Applied Measurement in Education*, Vol. 25, pp.27-40, 2012.

[36] X.-Y. Zhang and C.-L. Liu, “Writer adaptation with style transfer mapping,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 7, pp. 1773-1787, 2013.

[37] J. Du and Q. Huo, “A discriminative linear regression approach to adaptation of multi-prototype based classifiers and its applications for Chinese OCR,” *Pattern Recognition*, Vol. 46, No. 8, pp. 2313-2322, 2013.

[38] J. Du, J.-S. Hu, B. Zhu, S. Wei, and L.-R. Dai, “Writer adaptation using bottleneck features and discriminative linear regression for online handwritten Chinese character recognition,” *Proc. ICFHR*, 2014, pp. 311-316.

[39] J. Du, J.-F. Zhai, J.-S. Hu, B. Zhu, S. Wei, and L.-R. Dai, “Writer adaptive feature extraction based on convolutional neural networks for online handwritten Chinese character recognition,” *Proc. ICDAR*, 2015, pp.841-845.

[40] H.-M. Yang, X.-Y. Zhang, F. Yin, Z.-B. Luo and C.-L. Liu, “Unsupervised adaptation of neural networks for Chinese handwriting recognition,” *Proc. ICFHR*, pp. 512-517, 2016.

[41] X.-Y. Zhang, Y. Bengio, and C.-L. Liu, “Online and offline handwritten chinese character recognition: A comprehensive study and new benchmark” *Pattern Recognition*, Vol. 38, pp.348-360, 2016.

[42] I.-J. Goodfellow, J.-P. Abadie, M. Mirza, B. Xu, D.-W. Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Proc. NIPS*, pp.2672-2680, 2014.

[43] O. Abdel-Hamid and H. Jiang, “Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code,” *Proc. ICASSP*, 2013, pp.7942-7946.

[44] S.-F. Xue, O. Abdel-Hamid, H. Jiang, L.-R. Dai, and Q.-F. Liu, “Fast adaptation of deep neural network based on discriminant codes for speech recognition,” *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, Vol. 22, No. 12, pp.1713-1725, 2014.

[45] Zi-Rui Wang and Jun Du, “Writer Code Based Adaptation of Deep Neural Network for Offline Handwritten Chinese Text Recognition,” *Proc. ICFHR*, 2016, pp.311-316.

[46] A. Stolcke, “SRILM: an extensible language modeling toolkit,” *Proc. ICSLP*, 2002, pp.901-904.

[47] L. Yann, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, Vol. 11, pp.2278-2324, 1986.

[48] D. H. Hubel and T. N. Wiesel, “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex,” *Journal of Physiology*, Vol. 160, pp.106-154, 1962.

[49] S. Ioffe and C. Szegedy, “Batch normalization: accelerating deep network training by reducing internal covariate shift,” *arXiv:1502.03167*, 2015.

[50] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” *Proc. NIPS*, 2012, pp.1097-1105.

[51] Y.-P. Zhang, S. Liang, S. Nie, W.-J. Liu and S.-Y. Peng, “Robust offline handwritten character recognition through exploring writer-independent features under the guidance of printed data,” *Pattern Recognition letters*, Vol. 106, pp.20-26, 2018.

[52] T. Mikolov, S. Kombrink, L. Burget, J.H. ernock, and S. Khudanpur, “Extensions of recurrent neural network language model,” *Proc. ICASSP*, 2011, pp.5528-5531.

[53] C.-L. Liu, F. Yin, D.-H. Wang, and Q.-F. Wang, “CASIA online and offline Chinese handwriting databases,” *Proc. ICDAR*, 2011, pp.37-41.

[54] C.-L. Liu, F. Yin, D.-H. Wang, and Q.-F. Wang, “Online and offline handwritten Chinese character recognition: benchmarking on new databases,” *Pattern Recognition*, Vol. 46, No. 1, pp.155-162, 2013.

[55] L. van der Maaten, and G. Hinton, “Visualizing data using t-SNE,” *Machine Learning*, Vol. 9, pp.2579-2605, 2008.

[56] H.-M Yang, X.-Y Zhang, F. Yin, J. Sun and C.-L. Liu, “Deep transfer mapping for unsupervised writer adaptation,” *Proc. ICFHR*, 2018.

Zi-Rui Wang received a B.Eng. degree from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC), in 2015. He is currently a Ph.D. candidate at USTC. His current research area includes deep learning, handwritten Chinese text recognition and medical image processing.

Jun Du received B.Eng. and Ph.D. degrees from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC), in 2004 and 2009, respectively. From 2004 to 2009, he was with the iFlytek Speech Lab of USTC. During the above period, he worked as an Intern twice for 9 months at Microsoft Research Asia (MSRA), Beijing. In 2007, he also worked as a Research Assistant for 6 months in the Department of Computer Science at the University of Hong Kong. From July 2009 to June 2010, he worked at iFlytek Research on speech recognition. From July 2010 to January 2013, he joined MSRA as an Associate Researcher, working on handwriting recognition, OCR and speech recognition. Since February 2013, he has been with the National Engineering Laboratory for Speech and Language Information Processing (NEL-SLIP) of USTC.

Jia-Ming Wang received a B.Eng. degree from Anhui University in 2018. He is currently a Master’s degree candidate of USTC. His current research area is handwritten mathematical expression recognition.