Neural Networks 147 (2022) 42–52

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Fast writer adaptation with style extractor network for handwritten text recognition

Zi-Rui Wang [a,*], Jun Du [b]

[a] Chongqing University of Posts and Telecommunications, Chongqing, China
[b] University of Science and Technology of China, Hefei, Anhui, China

ARTICLE INFO

Article history:
Received 22 May 2021
Received in revised form 15 October 2021
Accepted 2 December 2021
Available online 9 December 2021

Keywords:
Fast writer adaptation
Style extractor network
Offline handwritten text recognition

ABSTRACT

Writing style is an abstract attribute in handwritten text. It plays an important role in recognition systems and is not easy to define explicitly. Considering the effect of writing style, a writer adaptation method is proposed to transform a writer-independent recognizer toward a particular writer. This transformation has the potential to significantly increase accuracy. In this paper, under the deep learning framework, we propose a general fast writer adaptation solution. Specifically, without depending on other complex skills, a well designed style extractor network (SEN) trained by identification loss (IDL) is introduced to explicitly extract personalized writer information. The architecture of SEN consists of a stack of convolutional layers followed by a recurrent neural network with gated recurrent units to remove semantic context and retain writer information. Then, the outputs of the GRU are further integrated into a one-dimensional vector that is adopted to represent writing style. Finally, the extracted style information is fed into the writer-independent recognizer to achieve adaptation. Validated on offline handwritten text recognition tasks, the proposed fast sentence-level adaptation achieves remarkable improvements in Chinese and English text recognition tasks. Specifically, in the HETR task, a multi-information fusion network that is equipped with a hybrid attention mechanism and that integrates visual features, context features and writing style is proposed. In addition, under the same condition (only one writer-specific text line used as adaptation data), the proposed solution, without consuming extra time, can significantly outperform the previous multiple-pass decoding method. The code is available at https://github.com/Wukong90/Handwritten-Text-Recognition.

1. Introduction

Deep learning (LeCun, Bengio, & Hinton, 2015) methods have recently made notable improvements on handwritten text recognition (Fujisawa, 2008). However, it remains challenging to build a robust text recognizer that can handle varying data of new writers effectively, due to the large variability of handwriting styles across individuals. In addition to the morphological variations within characters, writing orientation and ligatures make text recognition much more challenging than character recognition. To handle this problem, a general solution is to ensure the writer-independent recognizer has been trained with large-scale annotated data from many writers, which is time consuming to collect, label and train these data and cannot cover all diversities. However, in the test stage, the unlabeled writer-specific data are always accessed. Thus if we can fully utilize these data to extract style information for guiding the writer-independent recognizer, it can help improve the recognition performance. This process can be described as an unsupervised writer adaptation algorithm.

Writer adaptation is closely similar to speaker adaptation (Leggetter & Woodland, 1995; Saon, Soltau, Nahamoo, & Picheny, 2013) in speech recognition (Hinton, Deng, et al., 2012), which can be regarded as a special case of domain adaptation (Pan & Yang, 2009) where the test data usually have a different distribution from the training data. Depending on whether the adaptation data from each particular writer are labeled or unlabeled, the adaptation algorithms can be divided into supervised, unsupervised and semi-supervised adaptation. In most cases, there are no labeled data available in the test stage. From this perspective, unsupervised writer adaptation seems to be more difficult but practically useful. Previous unsupervised writer adaptation methods usually depend on extra complex procedures, such as multiple-pass decoding to obtain pseudo labels (Wang, Du, & Wang, 2020) for adaptation and style transfer mapping (Zhang & Liu, 2012) to transform writer-specific features into style-free space. In Zhang and Liu (2012), style transfer mapping (STM) based on least squares regression is employed to project source domain data (writer-specific space) into the target domain

* Corresponding author.
E-mail address: wangzr@cqupt.edu.cn (Z.-R. Wang).

https://doi.org/10.1016/j.neunet.2021.12.002
0893-6080/

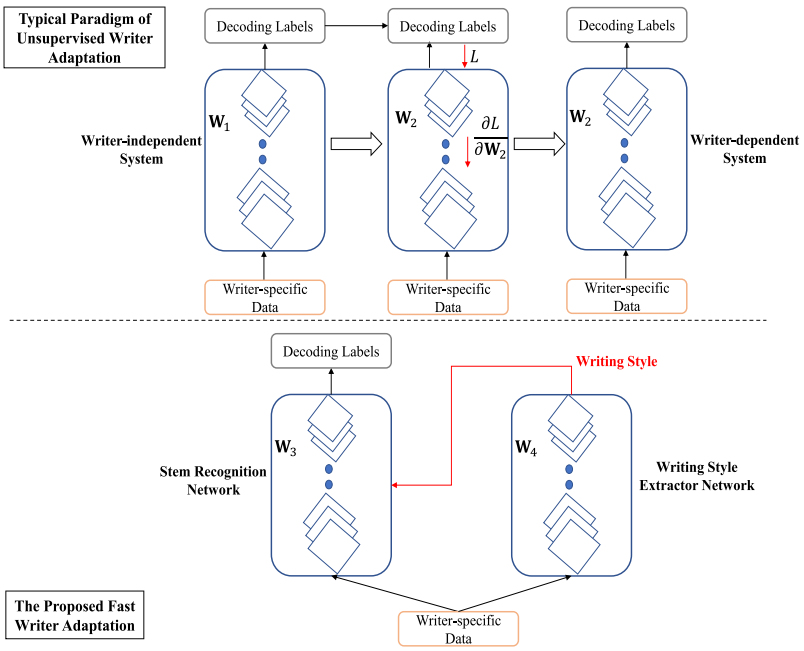

Fig. 1. Comparison of the typical paradigm of unsupervised writer adaptation and the proposed fast writer adaptation. The abbreviations $L, \mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3, \mathbf{W}_4$ denote the loss function and the weights in different neural networks, respectively.

(writer-independent space). However, due to the difficulty of defining source domain and target domain in the text recognition task, it is not easy to extend this method to text recognition tasks. In a recent work (Bhunia et al., 2021), Bhunia et al. employed model-agnostic meta-learning (MAML) (Finn, Abbeel, & Levine, 2017) to achieve the purpose of writer adaptation.

From an alternative perspective, the research on writer adaptation can be divided into feature-space and model-space approaches based on the parts to which the adaptation parameters are being applied. The former one focuses on transforming writer-dependent features into writer-independent features. In this approach, typical algorithms include incremental linear discriminant analysis (ILDA) (Huang, Ding, Jin, & Gao, 2009), incremental modified quadratic discriminant function (IMQDF) (Ding & Jin, 2010) and the already mentioned STM. The ILDA tries to search for the optimal linear projection matrix to maximize the distance between classes and minimizing the distance within the same class. The IMQDF can be regarded as the further development of the ILDA. The data of each class are assumed to be sampled from Gaussian distributions. The latter solution adjusts the writer-independent model toward the writer-dependent model by using specific writer data. The most common strategy is to re-estimate parameters of recognition systems, such as Gaussian Mixture Model-Hidden Markov Model (GMM–HMM) (Young et al., 2002) and deep neural network (DNN) (Wang, Du, Wang, Zhai, & Hu, 2018). In the presence of large amount of available data and GPU resources, DNN based methods have outperformed other models in many fields including image recognition (Deng et al., 2009), speech verification (Lin, Mak, Li, Su, & Yu, 2020), text recognition (Wu, Yin, & Liu, 2017), object detection (Carion et al., 2020) etc. Considering the flexible and changeable structure of DNNs, it is easy to design specific layers and losses to achieve model transformation.

In this paper, we propose a fast writer adaptation based on a well-designed style extractor network (SEN). Guided by identification loss (IDL), the SEN explicitly extracts style information. Then, the extracted style information is integrated into a one-dimensional vector and fed into the writer-independent recognizer (a typical recognition network) to achieve adaptation. The main features of this approach are described as follow:

- The SEN is a relatively independent module and can be embedded seamlessly into any recognition networks.
- Using the SEN to extract style information is an end-to-end approach without requiring other extra complex procedures and assumptions.
- The SEN is not related to character classes, which makes it suitable for any large-category problems, where the writer-specific data are insufficient to cover all classes.
- The adaptation depending on SEN is an unsupervised writer adaptation and can achieve sentence-level adaption.

In general, the comparison between the typical paradigm of unsupervised writer adaptation and the proposed fast writer adaptation is shown in Fig. 1. For typical paradigm, to re-optimize the parameters of the writer-dependent system well, a large amount of writer-specific data are needed. In addition, considering that the decoding time usually becomes bottleneck of the total running time, in systems that use the multiple-pass decoding methods, the time consumption increases accordingly. We validate the SEN-based adaptation on two different offline handwritten text databases in two different languages, i.e., the ICDAR2013 competition of CASIA-HWDB and the IAM. The proposed algorithm can achieve remarkable improvement. In addition, under the same condition (only one writer-specific text line used as adaptation data), the proposed solution can significantly outperform the previous multiple-pass decoding method without increased time consumption.

The contributions of this paper are described as follows:

(1) Under the deep learning framework, we propose a general writer adaptation solution, which can achieve fast sentence-level unsupervised adaptation.
(2) We propose the well-designed SEN to explicitly extract writer information. Through visualization of experimental analysis, we can observe how the extracted codes can accurately represent writing styles.
(3) We validate the capabilities of SEN on different offline handwritten text recognition tasks. The recognizer equipped with SEN can obtain obvious performance improvement and it far outperforms the multiple-pass decoding method (Wang, Du, et al., 2020) under the same comparable condition.

Table 1
The comparison of different segmentation-free methods.

Method	Solution	Network Input	Network Output
HMM	$p(\mathbf{C}\|\mathbf{X}) = \frac{p(\mathbf{X}\|\mathbf{C})p(\mathbf{C})}{p(\mathbf{X})} = \frac{[\sum_S \pi(s_0)\prod_{i=1} a_{s_{i-1}s_i}\prod_{i=0} p(\mathbf{x}_i\|s_i)]P(\mathbf{C})}{p(\mathbf{X})} = \frac{[\sum_S \pi(s_0)\prod_{i=1} a_{s_{i-1}s_i}\prod_{i=0} (p(s_i\|\mathbf{x}_i)p(\mathbf{x}_i)/p(s_i))]P(\mathbf{C})}{p(\mathbf{X})}$	Frame-level Image	State Posterior Probability $p(s_i\|\mathbf{x}_i)$
CTC	$p(\mathbf{C}\|\mathbf{X}) = \underset{U}{} = \sum_{u=1} \alpha(t,u)\beta(t,u)$	Text Image	Character Posterior Probability $p(C'_u\|\mathbf{x}_t)$ in Forward Probability $\alpha(t,u)$ and Back Probability $\beta(t,u)$
ED	$p(\mathbf{C}\|\mathbf{X}) = \prod_i p(C_u\|\mathbf{x}_i)$	Text Image	Character Posterior Probability $p(C_u\|\mathbf{x}_t)$

(4) A multi-information fusion network that is equipped with a hybrid attention mechanism and that integrates visual features, context features and writing style is proposed.

The remainder of this paper is organized as follows: Section 2 reviews the related work. Section 3 elaborates the details of the proposed method. Section 4 reports the experimental results and analyses. Finally, Section 5 concludes the paper.

2. Related works

In this section, we review related works, including handwritten text recognition, traditional writer adaptation and neural network based approaches for writer adaptation.

2.1. Handwritten text recognition

Handwritten text recognition (HTR) is a typical sequence-to-sequence problem. Offline HCTR can be formulated as a Bayesian decision problem:

$$\widehat{\mathbf{C}} = \underset{\mathbf{C}}{\operatorname{argmax}}\, p(\mathbf{C}|\mathbf{X}) \tag{1}$$

where $\mathbf{X}$ is the feature sequence of a given text line image and $\mathbf{C} = \{C_1, C_2, \ldots, C_n\}$ is the underlying n-character sequence. The key problem is how to compute the posterior probability $p(\mathbf{C}|\mathbf{X})$. The research efforts for addressing such sequence modeling tasks can be divided into four categories: oversegmentation (Wang, Wang, Yin, & Liu, 2020; Wu et al., 2017), connectionist temporal classification (CTC) (Graves, Fernández, Gomez, & Schmidhuber, 2006; Messina & Louradour, 2015; Zhang et al., 2021), hidden Markov model (HMM) (Wang et al., 2018) and encoder–decoder (ED) method (Zhang, Liang, & Jin, 2020; Zhang, Zhu, Du, & Dai, 2018). Almost all these approaches benefit from the recent progresses in deep learning. The outputs of neural networks in different modeling methods correspond to different concepts. For example, in oversegmentation approaches, the outputs of the neural network are related to segmentation identification or character classes. The outputs of the network used in HMM-based approaches are posterior probabilities of states, while the outputs of the network in CTC, ED-based approaches correspond to character classes. In general, oversegmentation and HMM-based approaches have more complex steps. On the other hand, CTC and ED-based networks are end-to-end frameworks, which are relatively easier to follow and have faster decoding speed. However, the latter two approaches are also highly dependent on large amounts of text data for training. The HMM-based approach can make full use of isolated characters and text data in datasets.

As shown in Table 1, we list the mathematical solutions of current segmentation-free methods (HMM, CTC, ED) and the corresponding inputs and outputs of recognition networks in different methods. In the HMM-based method (Wang et al., 2018), each character is modeled by an HMM and a text line can be represented by cascaded HMMs. At time $i-1$, the frame-level image $\mathbf{x}_{i-1}$ extracted from the original image by a left-to-right sliding window is assigned to one underlying state s_{i-1}. The next frame $\mathbf{x}_i$ moves to the state s_i with the transition probability $a_{s_{i-1}s_i}$. After obtaining the probability $p(c_i|\mathbf{x}_i)$ from a neural network, the formula $p(\mathbf{X}|\mathbf{C})$ is computable by using the forward–backward algorithm. The training loss in HMM based network is:

$$L_{\mathrm{HMM}}(\Theta) = -\sum_i \log p(c_i|\mathbf{x}_i, \Theta) \tag{2}$$

Θ denotes all the parameters of the weight matrices and bias vectors in the network. The frame-level minibatch training dataset with the state labels can be prepared by using the GMM–HMM (Wang et al., 2018). $P(\mathbf{C})$ is the so-called language model that can be modeled by traditional N-gram algorithm (Katz, 1987) or recurrent neural network (Mikolov, Karafiát, Burget, Cernockỳ, & Khudanpur, 2010). Since we only focus on writing style adaptation in this paper, the language model is not elaborated and not used in experiments. Considering that the main purpose is to find the most suitable $\widehat{\mathbf{C}}$, the prior probability $P(\mathbf{X})$ can be ignored. Different from the HMM, the CTC (Graves et al., 2006) adds a new class termed 'blank' and defines a sequence-to-sequence function φ to eliminate explicit segmentation. After a sentence is fed into the function φ, only one of the consecutive adjacent repeating characters is retained and the 'blank' characters are removed. Firstly, a new sentence $\mathbf{C}'$ is generated by adding 'blank' characters in the first and last locations and between each two characters of the sentence $\mathbf{C}$. Then, to effectively compute the probability of $p(\mathbf{C}'|\mathbf{X})$, the forward probability $\alpha(t,u)$ and back probability $\beta(t,u)$ are defined. The $\alpha(t,u)$ denotes the probability sum of all paths that arrive at the location u in the sentence $\mathbf{C}'$ at time t, in addition, these paths should get through all $u/2$ characters in the sentence $\mathbf{C}$. The $\beta(t,u)$ is used to compute the probability sum of the left corresponding paths that can form completed paths by combining with the paths in $\alpha(t,u)$. For the input image $\mathbf{X}$, the probability of C'_u at time t is estimated via a neural network. Thus the CTC can be regarded as a loss function of neural networks:

$$L_{\mathrm{CTC}}(\Theta) = -\log\Big(\sum_{\boldsymbol{\pi}:\varphi(\boldsymbol{\pi})=\mathbf{C}} p(\boldsymbol{\pi}|\mathbf{X}, \Theta)\Big) \tag{3}$$

The input is a completed image $\mathbf{X}$ with the sequence label $\mathbf{C}$, and $\boldsymbol{\pi}$ is the predicted character sequence under the constraint of $\varphi(\boldsymbol{\pi}) = \mathbf{C}$. Again, we did not use a language model and lexicon in our experiments. For the ED system, the network directly outputs the probability $p(C_u|\mathbf{x}_t)$.

2.2. Traditional writer adaptation

In the HTR task, most of the early works are based on the Gaussian Mixture Model-Hidden Markov Model (GMM–HMM) (Senior & Nathan, 1997). Naturally, writer adaptation can be achieved by using writer-specific data to adjust the means and variances in the Gaussian distributions. In Vinciarelli and Bengio (2002), maximum likelihood linear regression (MLLR) is employed to seek linear transformations between original parameters and re-estimated parameters. Furthermore, Brakensiek, Kosmala, and Rigoll (2001) compare different adaptation techniques of GMM–HMM for cursive German script, i.e., retraining models according to the maximum likelihood (ML) criterion using the Baum–Welch algorithm (Rabiner & Juang, 1986), the MLLR algorithm and the maximum a posterior (MAP) adaptation. Almost simultaneously, Vinciarelli and Bengio (2002) introduce similar algorithms for cursive English script recognition. Different from the above adaptation methods, Cao and Tan (2000) first perform page style clustering by using style features (such as contour slope, pen pressure, writing velocity) and then the authors build independent HMMs for each cluster. This approach can be regarded as an instance of unsupervised writer adaptation. Unlike the modeling way in the GMM–HMM system, inspired by the human perceptive psychology, segmentation based methods can achieve unsupervised adaptation or self-supervised adaptation through graphemes analysis. However, these methods only work on small datasets due to limited ability of traditional classifiers. In addition, style features introduced by artificial designs do not accurately represent complex writing variants.

2.3. Neural network based approaches for writer adaptation

The generality of neural networks indicates that they should be suitable for large-scale HTR tasks. An early attempt to use neural networks in this context is made in Frinken and Bunke (2009). In particular, a bidirectional long short-term memory (LSTM) neural network (Hochreiter & Schmidhuber, 1997) is used as a classifier for cursive word recognition in which the neural network is iteratively retrained on its own outputs of new unlabeled data. With the success of deep learning in a wide range of applications, many neural networks based writer adaptation approaches have been proposed recently. Nair et al. (2018) simply fine-tune pretrained networks on a new dataset to achieve writer adaptation. Due to large differences in writing styles between datasets, this simple operation can yield a remarkable improvement. This technique is also widely used in speaker adaptation (Huang, Lu, Lei, & Yan, 2018) and domain adaptation (Wang & Deng, 2018). Furthermore, Soullard, Swaileh, Tranouez, Paquet, and Chatelain (2019) adopt both the optical model and the language model (LM) to a particular writer. Specially, a linear interpolation of a writer-independent LM and a writer-specific LM is proposed. More recently, Zhang et al. (2019) employ a gated attention similarity unit in the network to adaptively find character-level writer-invariant features. The whole framework is an ED model. Validated on the same task of this paper, Wang, Du, et al. (2020) integrate each convolutional layer with one adaptive layer fed by a writer- dependent vector to extract the irrelevant variability in writer information to improve recognition performance. The authors first define the writer code with random initialization for each writer. Then the first-pass decoding results obtained from the writer-independent model are used to optimize these codes for the adaptation purpose. The above process can be repeated. However, this method highly depends on a large amount of writer-specific data and needs multiple-pass decoding, which is time consuming.

3. The proposed method

Given a baseline writer-independent neural network WINN($\mathbf{\Theta}$) and an SEN($\mathbf{\Gamma}$), let $\mathbf{\Theta}$ and $\mathbf{\Gamma}$ represent the weight sets of the WINN and the SEN, respectively. The writer adaptation method aims to guide the WINN toward the corresponding writer-dependent neural network WDNN($\mathbf{\Theta}$, $\mathbf{\Gamma}$, $\mathbf{\Lambda}$) by using particular writer data. Here, the symbol $\mathbf{\Lambda}$ represents the parameters connecting the WINN and the SEN. In our method, we employ the SEN to explicitly extract writer style for sentence-level fast writer adaptation. It should be noted that the SEN is a plug-and-play module that can be directly embedded into any handwritten text recognition networks. The whole training procedure has been summarized in Algorithm 1. As shown in Algorithm 2, in the testing stage, the corresponding style representation is first extracted based on the input text line. Then, the input text line and style representation are fed into the stem recognition network to obtain the decoding result. In the following subsections, we will elaborate on the details of the different networks.

Algorithm 1 The training procedure of the fast writer adaptation.

Require:
The randomly initialized parameter sets $\{\Theta, \Gamma, \Lambda\}$;
The learning rates $\epsilon_{\mathbf{WI}}$, $\epsilon_{\mathbf{SEN}}$, $\epsilon_{\mathbf{WD}}$ and loss functions $L_{\mathbf{WI}}$, $L_{\mathbf{SEN}}$, $L_{\mathbf{WD}}$ in the training stage of networks WINN, SEN and WDNN, respectively.

1: Optimize the WINN parameter set Θ by using back-propagation and stochastic gradient descent (SGD).
$\Theta = \Theta - \epsilon_{\mathbf{WI}} \frac{\partial \mathbf{L_{WI}}}{\partial \Theta}$
2: Train the SEN parameter set Γ based on the identification loss (IDL).
$\Gamma = \Gamma - \epsilon_{\mathbf{SEN}} \frac{\partial \mathbf{L_{SEN}}}{\partial \Theta}$
3: Jointly update $\{\Gamma, \Lambda\}$ during the training stage of the WDNN.
$\Gamma = \Gamma - \epsilon_{\mathbf{WD}} \frac{\partial \mathbf{L_{WD}}}{\partial \Theta}$
$\Lambda = \Lambda - \epsilon_{\mathbf{WD}} \frac{\partial \mathbf{L_{WD}}}{\partial \Theta}$
4: **return** The WDNN(Θ, Γ, Λ).

Algorithm 2 In the testing stage of the proposed system.

Require:
The text line $\mathbf{X}$;
Prepare the WDNN parameter set $\{\Theta, \Gamma, \Lambda\}$;

1: Extract the writer style representation $\mathbf{Y}$.
$\mathbf{Y} = \mathrm{NN}(\mathbf{X}|\Gamma, \Lambda)$
2: Obtain the final network output $\mathbf{O}$.
$\mathbf{O} = \mathrm{NN}(\mathbf{X}, \mathbf{Y}|\Theta)$
3: **return** Decoding result based on $\mathbf{O}$.

3.1. The backbone recognition networks

Regardless of the differences between the sequence algorithms mentioned in Section 2, the backbone recognition networks in the algorithms are similar. All of them are constructed by a stack of convolutional layers, fully-connected layers and recurrent neural units, and their training loss functions are typical cross entropy (CE). As shown in Fig. 2, three typical neural units are compared. For a one-dimensional input vector $\mathbf{I}^t$ fed into a fully-connected layer at time t, the corresponding output $\mathbf{O}^t$ is the linear transformation of $\mathbf{I}^t$ based on the weight matrix $\mathbf{W}_{\mathrm{FC}}$ and the bias vector $\mathbf{b}_{\mathrm{FC}}$, and followed by a nonlinear activation function (ReLu). Different from the fully-connected layer, the output of the recurrent unit (GRU [39] or LSTM) is not only related to the current input, but it also employs the historical information $\mathbf{h}^{t-1}$. Here, we use the generic function f instead of a specific expression.

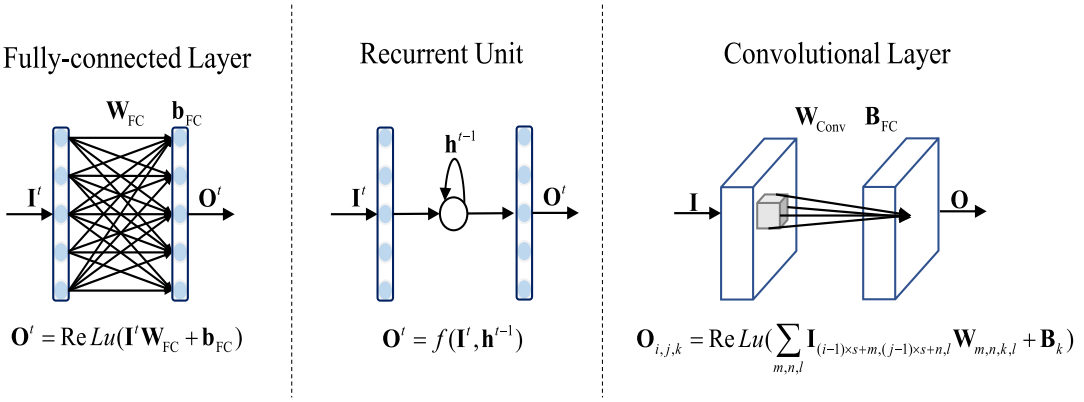

Fig. 2. Three typical neural units: fully-connected layer, recurrent unit and convolutional layer.

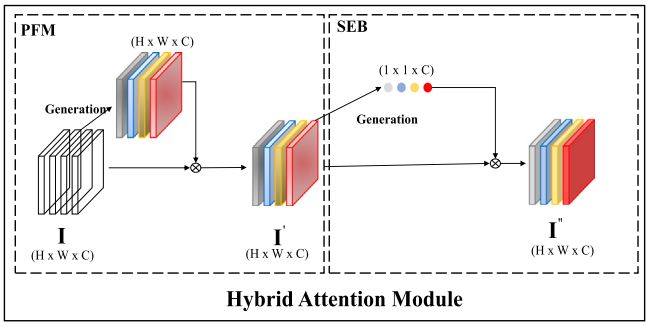

Fig. 3. The proposed hybrid attention module (HAM). The input **I** is a typical 3-D tensor in which the number of channels is C, height is H and width is W.

The transmission signal of the 2D convolutional layer is a three-dimensional tensor organized by a set of planes called feature maps. For the unit $\mathbf{I}_{i,j,k}$ in the feature map k at row i and column j, it is constrained to connect a local region $(M \times N)$ across all channels (L) in the previous layer, called the local receptive field. Two contiguous local receptive fields are usually s pixels (referred to as stride) shifted in a certain direction. Usually, all units in the same feature map k of a convolutional layer share a set of weights $\mathbf{W}_k$ and add the same bias value B_k, each computing a dot product between its weights and the local receptive field in the previous layer. In addition, convolutional layers can be followed by attention blocks to further improve network performance. In this paper, we also investigate two attention mechanisms, i.e., squeeze-and-excitation block (SEB) (Hu, Shen, & Sun, 2018) and parameter-free attention module (PFM) (Yang, Zhang, Li, & Xie, 2021). From Fig. 3, we can clearly observe that the SEB can achieve channel-wise attention by assigning to each channel a different weight. In PFM, based on neuroscience theories, 3-D attention weights are inferred for each pixel without adding new parameters to the original network. Benefitting from their observations, we combine both to form a finer attention, i.e., hybrid attention module (HAM). Intuitively, considering that the HAM consists of double attention modules, our attention module should improve the accuracy of the assigned weight of each pixel and promote the learning and recognition of the whole network.

In the offline handwritten Chinese text recognition (HCTR) task, we directly employ the CNN–PHMM system (Wang, Du, et al., 2020) as our baseline. The corresponding backbone recognition network is a simple convolutional neural network. For the offline handwritten English text recognition (HETR), we build a CRNN based on the CTC loss (Graves et al., 2006). Instead of simply combining common CNN and RNN, we equip the convolutional layers with the HAM. Further, the outputs of CNN noted as visual features are connected to the outputs of RNN (context feature) for the final prediction. The details of different backbone recognition networks are illustrated in Fig. 4.

3.2. The style extractor network

As shown in Fig. 5, on the left is the stem recognition network, which can be constructed for character modeling by using any type of network units in Fig. 2. For a given text image $\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_N$, where each $\mathbf{x}_t$ represents the feature vector at frame t. For the input feature $\mathbf{x}_t$, the corresponding network output is $\mathbf{y}_t$. The hidden layers of the stem recognition network are denoted as $\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_l, \mathbf{H}_{l+1}$. The structure of the SEN is described on the right of Fig. 5. The front part of the network is a cascade of CNN and RNN. CNN extracts deep representation of the current frame, and the extracted feature is then sent to RNN. There are two typical units in RNN, namely, LSTM and GRU. The details of LSTM and GRU are shown in Fig. 6. The LSTM consists of forget, input and output gates for maintaining its state over time to handle the problem of long-term dependencies well:

$$\mathbf{f}_t = \mathrm{sigm}(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{b}_f) \tag{4}$$

$$\mathbf{i}_t = \mathrm{sigm}(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{b}_i) \tag{5}$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_{\tilde{c}}\mathbf{x}_t + \mathbf{U}_{\tilde{c}}\mathbf{h}_{t-1} + \mathbf{b}_{\tilde{c}}) \tag{6}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \tag{7}$$

$$\mathbf{o}_t = \mathrm{sigm}(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{b}_o) \tag{8}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{9}$$

where $\mathbf{x}_t$ represents the input at time t and $\mathbf{h}_t$ is the corresponding output, **i**, **f**, **o** and **c** are the input gate, forget gate, output gate and cell vectors, respectively. The weight matrix subscripts have the meaning suggested by their names. The GRU can be regarded as a simplified version of the LSTM. It consists of an update gate and a forget gate. The update gate selects whether the hidden state is to be updated with a state candidate. The reset gate decides whether the previous hidden state is ignored. They are computed as follows:

$$\mathbf{r}_t = \mathrm{sigm}(\mathbf{W}_r\mathbf{x}_t + \mathbf{U}_r\mathbf{h}_{t-1} + \mathbf{b}_t) \tag{10}$$

$$\mathbf{z}_t = \mathrm{sigm}(\mathbf{W}_z\mathbf{x}_t + \mathbf{U}_z\mathbf{h}_{t-1} + \mathbf{b}_z) \tag{11}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}\mathbf{x}_t + \mathbf{U}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}) \tag{12}$$

$$\mathbf{h}_t = \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \tilde{\mathbf{h}}_t \tag{13}$$

where $\tilde{\mathbf{h}}_t$ is the so-called state candidate. $\mathbf{r}_t$ and $\mathbf{z}_t$ are the values of the reset gate and the update gate, respectively. We can observe that the current state candidate is controlled by the reset gate, which decides how much history information flows into the

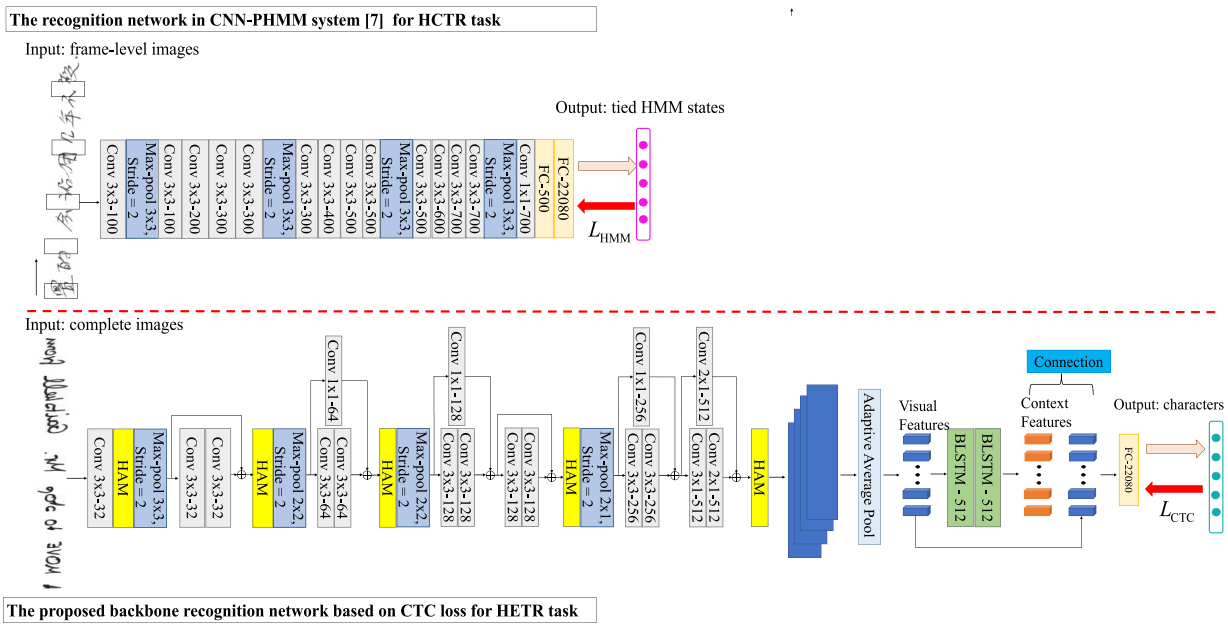

Fig. 4. The backbone recognition networks used in HCTR and HETR tasks. All convolutional layers are followed by the ReLU and the stride is 1. In addition, the BN operation is equipped for the outputs before nonlinearity in every convolutional layer.

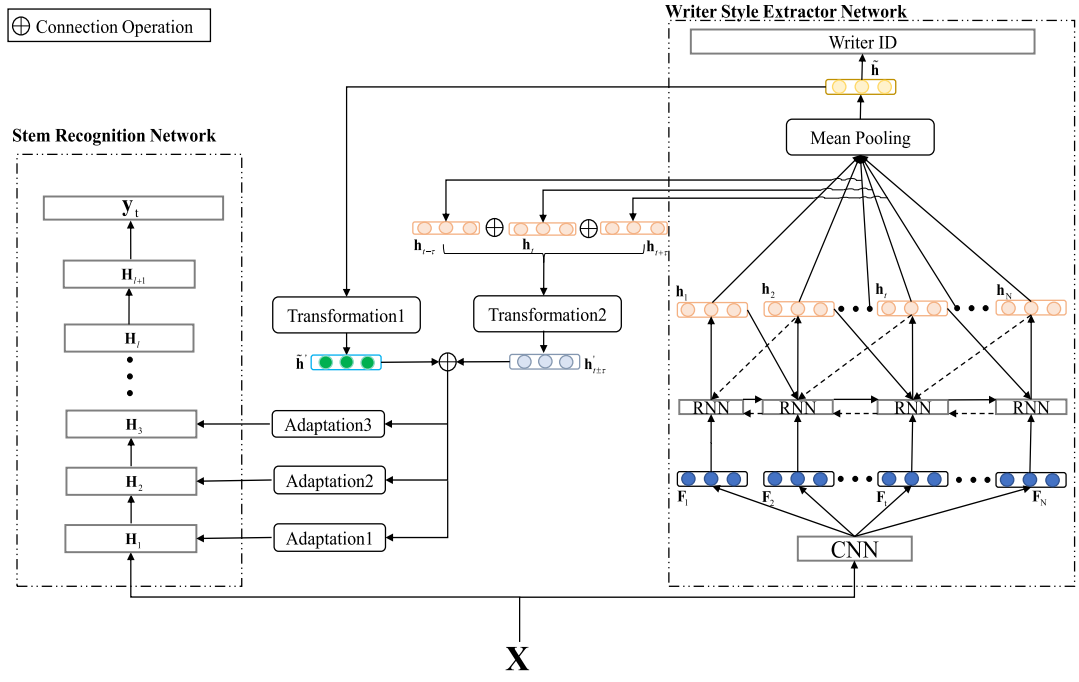

Fig. 5. The network structure of the proposed fast writer adaptation.

current state candidate. When the value of the reset gate's output approaches 0, the current state candidate only depends on the current input. It can be observed that the value of the update gate controls how much information comes from the state value of the previous moment and the current state candidate, so as to realize the long-term memory function.

By using the gating mechanism, we expect the content information of the text **X** is gradually filtered out and the writer style information can be retained. A mean pooling layer is added to obtain the average value of all time points:

$$\bar{\mathbf{h}} = \frac{1}{\mathrm{T}} \sum_{t} \mathbf{h}_t \tag{14}$$

The vector $\bar{\mathbf{h}}$ is the representation of the global style of the sentence. In the decoding stage, the system recognizes the text time by time, so we add the style information representation of the current time and its neighborhood to better guide the recognition network. In order to better fuse global style features with local style features, the extracted style information is first transformed.

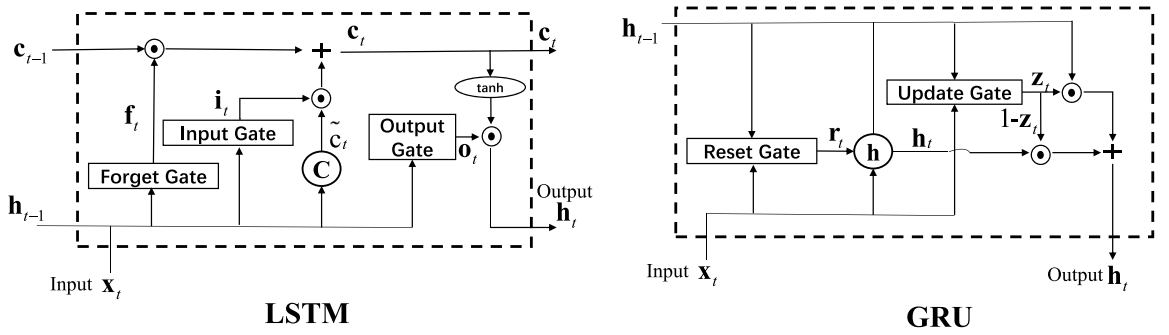

Fig. 6. The illustration of the LSTM and the GRU.

This transformation is constructed by using two fully-connected layers:

$$\bar{\mathbf{h}}' = ReLu((ReLu(\bar{\mathbf{h}}\mathbf{W}_{11} + \mathbf{b}_{11}))\mathbf{W}_{12} + \mathbf{b}_{12}) \tag{15}$$

$$\bar{\mathbf{h}}'_{t\pm\tau} = ReLu((ReLu(\mathbf{h}_{t\pm\tau}\mathbf{W}_{21} + \mathbf{b}_{21}))\mathbf{W}_{22} + \mathbf{b}_{22}) \tag{16}$$

The transformed global style representation and local style representation are then connected together:

$$\mathbf{c} = \bar{\mathbf{h}}' \oplus \mathbf{h}'_{t\pm\tau} \tag{17}$$

Then, the vector $\mathbf{c}$ is sent into the stem recognition network to achieve writer adaptation. In principle, the style information can be integrated into any type layers. To better guide the stem recognition network, the style information is projected into the feature space of the recognition network through adaptation layer:

$$\mathbf{c}' = f(\bar{\mathbf{c}}\mathbf{W}_{\text{ada}} + \mathbf{b}_{\text{ada}}) \tag{18}$$

The transformed vector is added into the convolution layer:

$$\mathbf{O}_{i,j,k} = ReLu(\sum_{m,n,l} \mathbf{I}_{(i-1)\times s+m,(j-1)\times s+n,j}\mathbf{W}_{m,n,k,l} + \mathbf{c}''_k) \tag{19}$$

$$\mathbf{c}''_k = \mathbf{B}_k + \mathbf{c}'_k \tag{20}$$

Thus, the writing style plays a role of bias vector to guide the stem recognition network. In the actual implementation, we make the dimension of the vector $\mathbf{c}'$ the same as the channel number of the corresponding convolution layer. Then, each element in the vector $\mathbf{c}'$, is expanded into a two-dimensional plane by repeating the same value for addition into channels.

In training the SEN, the loss function L_{SEN} is defined as follows:

$$L_{\text{SEN}}(\mathbf{\Gamma}) = -\sum_{u} \log p(ID_u|\mathbf{X}_u, \mathbf{\Gamma}) \tag{21}$$

For a text line u, the variable ID_u denotes the corresponding writer (identification). Therefore, the loss function L_{SEN} is noted as identification loss (IDL). After completing the training of the SEN, we jointly optimize the parameters Θ, Γ:

$$L_{\text{WD}}(\mathbf{\Gamma}, \mathbf{\Lambda}) = -\sum_{u}\sum_{t} \log p(s_t|\mathbf{x}_t, \mathbf{X}_u, \mathbf{\Gamma}, \mathbf{\Lambda}) \tag{22}$$

4. Experiments and result analysis

4.1. Datasets and metric

The proposed fast writer adaptation algorithm is validated on two tasks: offline HCTR and offline HETR. In HCTR, we use the CASIA database (Liu, Yin, Wang, & Wang, 2011). Both offline isolated handwritten Chinese character datasets (HWDB1.0, HWDB1.1 and HWDB1.2) and the training sets of offline handwritten Chinese text datasets (HWDB2.0, HWDB2.1 and HWDB2.2) are used. The details of the CASIA database are listed in Table 2. Because the offline text databases are produced by the same writers of the isolated character datasets and each person writes five pages of given texts, the total number of writers is counted according to the datasets HWDB1.0-1.2. In our experiments, there are 7,360 classes (Chinese characters, symbols, garbage) in total. The ICDAR2013 competition set including 3,432 text lines and written by 60 persons, who do not contribute to the training dataset is adopted as the evaluation set (Yin, Wang, Zhang, & Liu, 2013). In HETR, we evaluate the performance of our method on the IAM dataset (Marti & Bunke, 2002). The IAM dataset contains a total number of 9,862 text lines written by 500 writers. As shown in Table 3, it provides one training set, one testing set and two validation sets. The text lines of all datasets are mutually exclusive, thus, each writer has contributed to one set only.

Table 2
The statistics of the datasets CASIA1.0-1.2, 2.0-2.2.

Dataset	Writers	Lines	Character/class
HWDB1.0	420	–	1,680,258/3,866
HWDB1.1	300	–	1,172,907/3,755
HWDB1.2	300	–	1,041,970/3,319
HWDB2.0	419	20,495	538,868/1,222
HWDB2.1	300	17,292	429,553/2,310
HWDB2.2	300	14,443	380,993/1,331

Table 3
The partition of the dataset IAM.

Set name	Text lines	Writers
Train	6,161	283
Validation1	900	46
Validation2	940	43
Test	1,861	128
Total	9,862	500

The evaluation criterion is defined as follows:

$$\frac{N_{\text{s}} + N_{\text{i}} + N_{\text{d}}}{N} \tag{23}$$

where N is the total number of samples in the evaluation set. N_{s}, N_{i} and N_{d} denote the number of substitution errors, insertion errors and deletion errors, respectively. All experiments were implemented by using the Kaldi (Povey et al., 2011) and Pytorch (Paszke et al., 2019) toolkits. PyTorch was used as a deep learning platform in all experiments.

4.2. Experiments on CASIA

We directly employ the state-of-the-art writer-independent network and the same training strategy (Wang, Du, et al., 2020) as our stem recognition network. Because the optimal system was constructed based on the so-called parsimonious HMM,

Table 4
The structure details of the SEN. The abbreviations F, K, S, P in CNN represent the number of features, kernel size, stride length and padding size, respectively.

Notation	Layer	Configurations	Output size
Output	FC	200 × 1836	[1836,1,1]
Mean Pooling			
GRU	State Candidate	$\mathbf{W}$: 250 × 200, $\mathbf{U}$: 250 × 200	[200,1,1]
	Reset Door	$\mathbf{W}_r$: 250 × 200, $\mathbf{U}_r$: 250 × 200	[200,1,1]
	Update Door	$\mathbf{W}_z$: 250 × 200, $\mathbf{U}_z$: 250 × 200	[200,1,1]
CNN	Conv	F : 250, K : 1, S : 1, P : 0	[250,1,1]
	Max Pooling	K : 3, S : 2	[200,1,1]
	Conv	F : 200, K : 3, S : 1, P : 1	[200,4,4]
	Max Pooling	K : 3, S : 2	[150,4,4]
	Conv	F : 150, K : 3, S : 1, P : 1	[150,10,10]
	Max Pooling	K:3, S:2	[100,10,10]
	Conv	F : 100, K : 3, S : 1, P : 1	[100,22,22]
	Max Pooling	K : 3, S : 2	[50,22,22]
	Conv	F : 50, K : 3, S : 1, P : 0	[50,46,46]
Input	Frame-level image	–	[1,48,48]

each character was modeled by 3 tied HMM states on average. Therefore, the input was a normalized frame-level image of size 40 × 40 extracted from original images, and then each frame size was extended to 48 × 48 by adding the margin. The output nodes of the DCNN was 7360 × 3 = 22,080. Table 4 lists the details of the main part of the style extraction network (dashed box on the right of Fig. 5). The structure was a CNN followed by an RNN. The CNN consisted of alternately stacked convolution layers and max pooling layers. The configuration of CNN referred to the stem recognition network, i.e., the kernel size of the last convolution layer was set to 1 × 1, which was used to further integrate the information between different channels extracted by the front convolutional layers. The kernel size of other convolution layers was typical 3 × 3. The max pooling layers with a window size of 3 × 3 and a stride of 2 were used to reduce the size of the input image, so as to integrate spatial information and overcome noise interference. The number of channels increased from 50 to 250 so that the appropriate deep feature could be extracted without consuming too much computation. Because the CNN is only responsible for extracting the deep feature of the original frame-level image time by time, it does not have the function of fusing semantic information. Therefore, we added a RNN to fuse information at different time steps. In order to achieve the goal of information fusion, filtering reductant text information and retaining style information under the premise of relatively less computation, we adopted a GRU to achieve above purpose. The parameter configurations of the reset gate, update gate and state candidate in the GRU are shown in Table 4, corresponding to formulas (3)–(5), respectively. In particular, we hoped to keep accurate semantic and style information by using update and reset gates. As shown in Eq. (14), the outputs of GRU were further integrated through a mean pooling layer to obtain the vector $\widetilde{\mathbf{h}}$, which can be used to characterize the style of the sentence. In the training stage of the SEN, the vector $\widetilde{\mathbf{h}}$ was directly fed into a fully-connected layer to obtain the posterior probability of writer IDs.

Based on the above description, the pretraining of the style extraction network is based on complete images. In order to achieve a better training performance, we first randomized the total training samples, and selected 3,800,000 images as the training samples and 132,197 images as the development samples. Although text lines are written by the same writers in isolated characters dataset, the situation of text lines is richer than that of isolated characters. For example, the writing style of the text is reflected not only in isolated Chinese characters, but also in the trend of text, the compact arrangement of different characters and so on. Therefore, in the pretraining SEN, we set the writers of text and isolated characters to be different, so that the output layer had a total of 1,836 nodes that correspond to different writer IDs. We still used the SGD algorithm to train the network. The initial learning rate was set to 0.1, the momentum was 0.9 and the weight decay was 10-4. We adjusted the learning rate according to time steps. For every 4 million steps, the learning rate was multiplied by 0.92. Once the IDL loss of the network on the development set no longer changes greatly, we stopped training.

Table 5
The CER comparisons of different connection ways of global writing style and local writing style.

τ	Without G	With G	
		Connection 1	Connection 2
0	9.10	8.86	9.22
1	9.05	8.83	8.88
2	8.97	8.80	8.73
3	8.94	8.83	8.71
baseline	9.17		

After the training of the individual SEN was completed, joint optimization of the stem recognition network and the SEN was conducted according to the connection way in Fig. 5. In the stage of joint optimization, the output layer of the SEN can be directly discarded. Since both the stem recognition network and the SEN have completed pretraining, the initial learning rate should not be set too large during the joint training to avoid destroying the function of the pre-trained networks. In the experiments, we found that the initial learning rate set to 0.001 is a good choice. The parameters involved in the SGD algorithm are the same as before, that is, the momentum was 0.9 and the weight decay was 0.5. We fixed the parameters of the stem recognition network and only updated the parameters in the added transformation layers, adaptation layers and style extraction network.

4.2.1. Different connection ways of global writing style and local writing style

We first explored the influence of different connection ways of global writing style and local writing style. As shown in Table 5, there are three ways in experiments: (a). Do not use global writing style information; (b). Firstly, the global and local style information are connected. And then, the connected feature is fed into a transformation. We label this way as Connection 1; (c). According to Eqs. (15) and (16), the global and local styles are fed into two independent transformations, respectively. And then the transformed style representations are connected together and sent to the adaptation layers to guide the corresponding convolution layers (Connection 2 in Table 5). In Table 5, we not only list the recognition performance of different connection methods, but also show the influence of adding different adjacent frames in the local handwriting style representation.

First of all, we can observe that with the increase of τ in the local writing style, the recognition performance can be consistently improved, reaching saturation when the value of τ is 3. This result is reasonable, because in the HMM method, the decoding result at the current time is affected by the information of the adjacent frames. Thus the integration of the adjacent frames in an appropriate range can certainly promote the recognition performance. Secondly, from the experiments of adding or not adding global writing style information, we can observe that the recognition rate can be improved by adding global writing style information, which proves that the style information extracted by the SEN plays a writer-aware role in the backbone recognition network. If the information extracted by the SEN is highly related to the

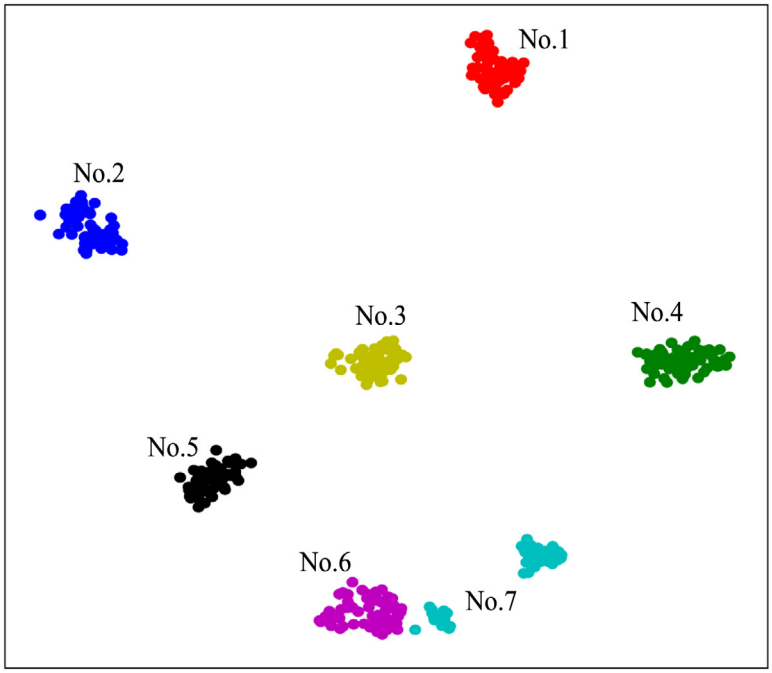

Fig. 7. The different writing styles obtained by using the t-SNE algorithm.

content of the text after joint training, then for the handwritten text lines with long content, the representation after average process is almost impossible to improve the decoding result of the current time. Finally, from the results of different connection modes (Connection1 and Connection2), 0.1% CER reduction can be obtained by changing Connection1 to Connection2. Furthermore, if we increase the joint training to two epochs, the final CER can be reduced to 8.51%, thus yielding a 7% relative CER reduction. Compared with our previous algorithms, the proposed unsupervised adaptation algorithm at the sentence level is great progress.

In order to understand the effectness of the proposed fast writer adaptation, as shown in Fig. 7, we project the style representation extracted from each text line into a 2-dimensional plane by using t-SNE algorithm (Van der Maaten & Hinton, 2008). It clearly shows that points from different writers are clustered in different locations while the representations from the same writer are gathered together. From Fig. 8, we can observe that although the text content are the same for writers No. 1 and No. 7, their corresponding representations are far away due to their styles' diversity. The writing representations obtained from No. 7 are close to the writing representations of No. 6. Especially, for these text lines No. 7-a, the style is very similar to that of No. 6. Therefore, their corresponding writing representations (the left cyan dots in Fig. 7) are also closer to the representations of No. 6.

4.2.2. Comparison between the proposed method and writer code based method

We compare the proposed fast sentence-level adaptation scheme with the system based on writer code in paper (Wang, Du, et al., 2020). The main disadvantage of the adaptation scheme based on writer code is that these codes are initialized randomly in advance and must be optimized repeatedly by using the back propagation algorithm with the decoding results. On the one hand, in order to optimize the specific writer code, we need to use enough adaptation data. On the other hand, in unsupervised adaptation, the labels of adaptation data need to be obtained from the first-pass decoding by using a writer-independent network, which leads to a large delay. As shown in Fig. 9, in the case of using fewer than 10 sentences, the recognition rate of the system is even worse than that of the baseline. In the case of using only one sentence which is comparable to the proposed adaptation scheme, the recognition performance is far less than the fast adaptation algorithm proposed in this paper (9.53% vs 8.51%). When using 40 lines as adaptation data, the CER can achieve a comparable result with the proposed fast writer adaptation. However, the time consumption of the writer code based solution includes two parts: the adaptation time and the decoding time, which linearly increases the recognition time. For systems with slow decoding speed, such as HMM, this is a heavy computational burden to bear. More importantly, in some applications, there are no enough adaptation data to use. In contrast, the proposed fast adaptation scheme decodes the current input sentence directly.

4.3. Experiments on IAM

In this task, we first built the CTC loss based recognition network (Fig. 4). The network front end was composed of four blocks, each of which was composed of several convolution layers, and the max pooling layer was inserted between the blocks. The back end consisted of two layers of bidirectional LSTM. In particular, we added the proposed HAM (Fig. 3) to the output of the convolution layer and the visual features were connected to the context features for the final prediction. Since it was optimized by the CTC loss, the input of the network was a complete image with a height of 124 and a width of 1751. The dataset does not include enough samples and the CTC training depends on a large number of text line data. Therefore, we conducted the following data augmentation: for a single picture, it was rotated randomly in the range of [-0.5,−0.3,-0.1,0.1,0.3,0.5] and cut randomly in the range of [-0.5,−0.3,-0.1,0.3,0.5]. In the training stage, the batch size was 20, the RMSProp optimization algorithm (Hinton, Srivastava, & Swersky, 2012) with 0.0005 as the initial learning rate was adopted, and the total number of training epochs was set to 400. We used the validation1 set to select the best model for testing and subsequent joint training. Referring to the structure of the backbone recognition network, we constructed the corresponding SEN, which was still composed of four blocks. Each block

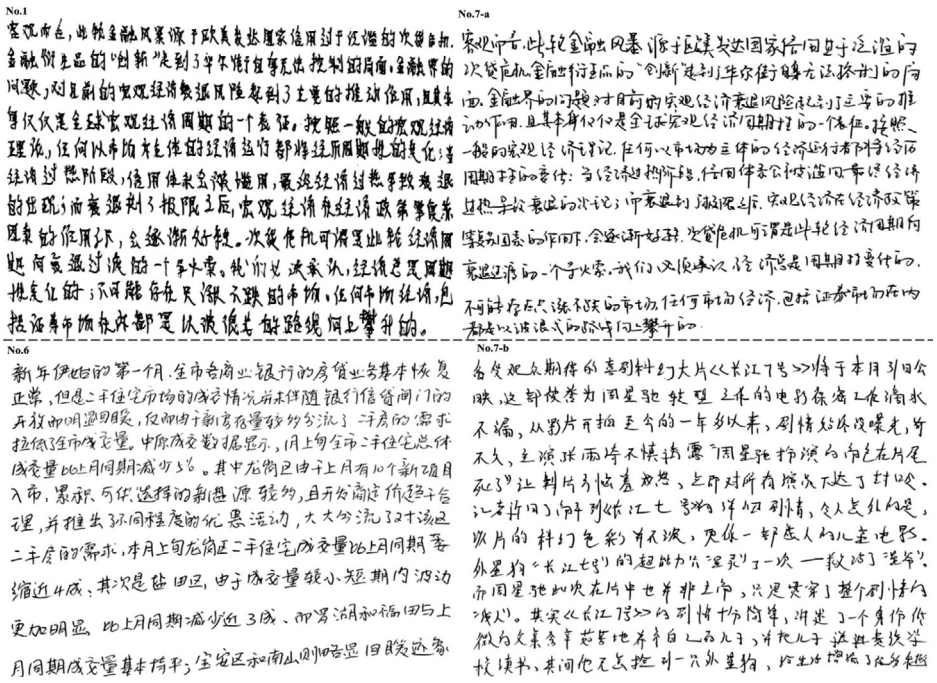

Fig. 8. Examples of text lines written by No. 1, No. 6 and No. 7.

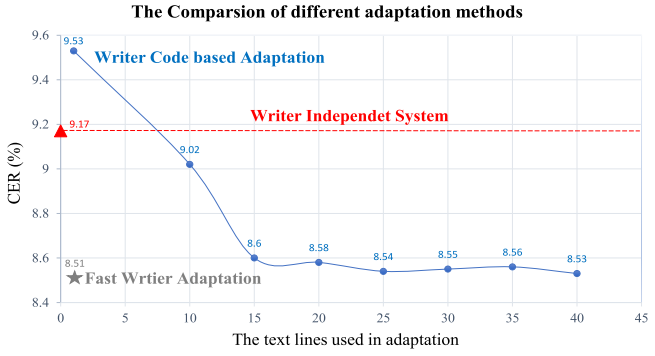

Fig. 9. Comparison of writer code based adaptation and the proposed fast adaptation.

contained only one single convolution layer, and the number of channels in different convolution layers was [16,32,64,128], respectively. The output of the last convolution layer was fed into one single bidirectional LSTM, and the final results were obtained through a fully connected layer. For the training of SEN by using the IDL, the RMSProp optimization algorithm with an initial learning rate of 0.0005 was used. The batch size was 64 and the total number of training epochs was 400. Because the writers of the training set and the validation set in IAM are different from each other, we divided 500 samples from the training set as the verification set. Similarly, we selected the best model for subsequent joint training according to the results in the validation set. In the stage of joint training, a similar training strategy was used for the recognition network.

From the results in Table 6, we can observe that the proposed fusion of visual features, writing style information and the HAM can achieve a remarkable improvement over the baseline system: the word error rates (WERs) of the validation set and the test set have decreased by more than 3 percentage points, and the character error rates (CERs) have dropped by approximately 1 percentage point. Each individual technology can also obviously improve the performance. For example, the fast writer adaptation can reduce the WERs of the validation set and the test set by approximately 1 percentage point.

Table 6
The results of different models in IAM dataset.

Model	Validation		Test	
	CER	WER	CER	WER
baseline	5.29	19.83	6.39	22.03
+HAM	4.72	18.02	5.64	20.39
+HAM +Vis	4.58	17.4	5.5	19.5
+HAM +Vis +SEN	**4.4**	**16.6**	**5.3**	**18.5**

5. Conclusion

In this paper, under the deep learning framework, we propose a general fast writer adaptation solution. A well designed style extractor network (SEN) is introduced to explicitly extract personalized writer information for guiding the stem recognition network. Validated on offline handwritten text recognition tasks, the proposed fast sentence-level adaptation achieves remarkable improvement in Chinese and English text recognition tasks. Specifically, in the HETR task, a multi-information fusion network that is equipped with a hybrid attention mechanism and that integrates visual features, context features and writing style is proposed. Moreover, the proposed fast writer adaptation far outperform the previous multiple-pass decoding method. However, the current framework does not fully utilize all writer-specific data to extract sentence-level style information during the decoding of a text line. In future work, we will consider using a memory mechanism to dynamically update and store style information to further improve the recognition results.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 62171427 and 62106031.

References

Bhunia, Ayan Kumar, Ghose, Shuvozit, Kumar, Amandeep, Chowdhury, Pinaki Nath, Sain, Aneeshan, & Song, Yi-Zhe (2012). MetaHTR: Towards writer-adaptive handwritten text recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 15830–15839).

Brakensiek, Anja, Kosmala, Andreas, & Rigoll, Gerhard (2001). Comparing adaptation techniques for on-line handwriting recognition. In *Proceedings of sixth international conference on document analysis and recognition* (pp. 486–490). IEEE.

Cao, Ruini, & Tan, Chew Lim (2000). A model of stroke extraction from chinese character images. In *Proceedings 15th international conference on pattern recognition, vol. 4* (pp. 368–371). IEEE.

Carion, Nicolas, Massa, Francisco, Synnaeve, Gabriel, Usunier, Nicolas, Kirillov, Alexander, & Zagoruyko, Sergey (2020). End-to-end object detection with transformers. In *European conference on computer vision* (pp. 213–229). Springer.

Deng, Jia, Dong, Wei, Socher, Richard, Li, Li-Jia, Li, Kai, & Fei-Fei, Li (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255). Ieee.

Ding, Kai, & Jin, Lianwen (2010). Incremental MQDF learning for writer adaptive handwriting recognition. In *2010 12th International conference on frontiers in handwriting recognition* (pp. 559–564). IEEE.

Finn, Chelsea, Abbeel, Pieter, & Levine, Sergey (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine Learning* (pp. 1126–1135). PMLR.

Frinken, Volkmar, & Bunke, Horst (2009). Evaluating retraining rules for semi-supervised learning in neural network based cursive word recognition. In *2009 10th International conference on document analysis and recognition* (pp. 31–35). IEEE.

Fujisawa, Hiromichi (2008). Forty years of research in character and document recognition—an industrial perspective. *Pattern Recognition*, *41*(8), 2435–2446.

Graves, Alex, Fernández, Santiago, Gomez, Faustino, & Schmidhuber, Jürgen (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on machine learning* (pp. 369–376).

Hinton, Geoffrey, Deng, Li, Yu, Dong, Dahl, George E, Mohamed, Abdel-rahman, Jaitly, Navdeep, et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82–97.

Hinton, Geoffrey, Srivastava, Nitish, & Swersky, Kevin (2012). Rmsprop: Divide the gradient by a running average of its recent magnitude. *Neural Networks for Machine Learning, Coursera Lecture 6e*, 13.

Hochreiter, Sepp, & Schmidhuber, Jürgen (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Hu, Jie, Shen, Li, & Sun, Gang (2018). Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7132–7141).

Huang, Zhibin, Ding, Kai, Jin, Lianwen, & Gao, Xue (2009). Writer adaptive online handwriting recognition using incremental linear discriminant analysis. In *2009 10th International conference on document analysis and recognition* (pp. 91–95). IEEE.

Huang, Zhiying, Lu, Heng, Lei, Ming, & Yan, Zhijie (2018). Linear networks based speaker adaptation for speech synthesis. In *2018 IEEE international conference on acoustics, speech and signal processing* (pp. 5319–5323). IEEE.

Katz, Slava (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, *35*(3), 400–401.

LeCun, Yann, Bengio, Yoshua, & Hinton, Geoffrey (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Leggetter, Christopher J., & Woodland, Philip C. (1995). Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, *9*(2), 171–185.

Lin, Weiwei, Mak, Man-Mai, Li, Na, Su, Dan, & Yu, Dong (2020). Multi-level deep neural network adaptation for speaker verification using MMD and consistency regularization. In *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing* (pp. 6839–6843). IEEE.

Liu, Cheng-Lin, Yin, Fei, Wang, Da-Han, & Wang, Qiu-Feng (2011). CASIA online and offline Chinese handwriting databases. In *2011 International conference on document analysis and recognition* (pp. 37–41). IEEE.

Van der Maaten, Laurens, & Hinton, Geoffrey (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(11).

Marti, U.-V., & Bunke, Horst (2002). The IAM-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, *5*(1), 39–46.

Messina, Ronaldo, & Louradour, Jerome (2015). Segmentation-free handwritten Chinese text recognition with LSTM-RNN. In *2015 13th International conference on document analysis and recognition* (pp. 171–175). IEEE.

Mikolov, Tomas, Karafiát, Martin, Burget, Lukas, Cernockỳ, Jan, & Khudanpur, Sanjeev (2010). Recurrent neural network based language model. In *Interspeech, vol. 2, no. 3* (pp. 1045–1048). Makuhari.

Nair, Rathin Radhakrishnan, Sankaran, Nishant, Kota, Bharagava Urala, Tulyakov, Sergey, Setlur, Srirangaraj, & Govindaraju, Venu (2018). Knowledge transfer using neural network based approach for handwritten text recognition. In *2018 13th IAPR international workshop on document analysis systems* (pp. 441–446). IEEE.

Pan, Sinno Jialin, & Yang, Qiang (2009). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359.

Paszke, Adam, Gross, Sam, Massa, Francisco, Lerer, Adam, Bradbury, James, Chanan, Gregory, et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, *32*, 8026–8037.

Povey, Daniel, Ghoshal, Arnab, Boulianne, Gilles, Burget, Lukas, Glembek, Ondrej, Goel, Nagendra, et al. (2011). The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on automatic speech recognition and understanding*. (CONF), IEEE Signal Processing Society.

Rabiner, Lawrence, & Juang, Biinghwang (1986). An introduction to hidden Markov models. *IEEE Assp Magazine*, *3*(1), 4–16.

Saon, George, Soltau, Hagen, Nahamoo, David, & Picheny, Michael (2013). Speaker adaptation of neural network acoustic models using i-vectors. In *2013 IEEE workshop on automatic speech recognition and understanding* (pp. 55–59). IEEE.

Senior, Andrew, & Nathan, Krishna (1997). Writer adaptation of a HMM handwriting recognition system. In *1997 IEEE international conference on acoustics, speech, and signal processing, vol. 2* (pp. 1447–1450). IEEE.

Soullard, Yann, Swaileh, Wassim, Tranouez, Pierrick, Paquet, Thierry, & Chatelain, Clément (2019). Improving text recognition using optical and language model writer adaptation. In *2019 International conference on document analysis and recognition* (pp. 1175–1180). IEEE.

Vinciarelli, Alessandro, & Bengio, Samy (2002). Writer adaptation techniques in HMM based off-line cursive script recognition. *Pattern Recognition Letters*, *23*(8), 905–916.

Wang, Mei, & Deng, Weihong (2018). Deep visual domain adaptation: A survey. *Neurocomputing*, *312*, 135–153.

Wang, Zi-Rui, Du, Jun, & Wang, Jia-Ming (2020). Writer-aware CNN for parsimonious HMM-based offline handwritten Chinese text recognition. *Pattern Recognition*, *100*, Article 107102.

Wang, Zi-Rui, Du, Jun, Wang, Wen-Chao, Zhai, Jian-Fang, & Hu, Jin-Shui (2018). A comprehensive study of hybrid neural network hidden Markov model for offline handwritten Chinese text recognition. *International Journal on Document Analysis and Recognition (IJDAR)*, *21*(4), 241–251.

Wang, Zhen-Xing, Wang, Qiu-Feng, Yin, Fei, & Liu, Cheng-Lin (2020). Weakly supervised learning for over-segmentation based handwritten Chinese text recognition. In *2020 17th International conference on frontiers in handwriting recognition* (pp. 157–162). IEEE.

Wu, Yi-Chao, Yin, Fei, & Liu, Cheng-Lin (2017). Improving handwritten Chinese text recognition using neural network language models and convolutional neural network shape models. *Pattern Recognition*, *65*, 251–264.

Yang, Lingxiao, Zhang, Ru-Yuan, Li, Lida, & Xie, Xiaohua (2021). Simam: A simple, parameter-free attention module for convolutional neural networks. In *International conference on machine learning* (pp. 11863–11874). PMLR.

Yin, Fei, Wang, Qiu-Feng, Zhang, Xu-Yao, & Liu, Cheng-Lin (2013). ICDAR 2013 Chinese handwriting recognition competition. In *2013 12th International conference on document analysis and recognition* (pp. 1464–1470). IEEE.

Young, Steve, Evermann, Gunnar, Gales, Mark, Hain, Thomas, Kershaw, Dan, Liu, Xunying, et al. (2002). *The HTK book, vol. 3. no. 175* (p. 12). Cambridge University Engineering Department.

Zhang, Hesuo, Liang, Lingyu, & Jin, Lianwen (2020). SCUT-HCCDoc: A new benchmark dataset of handwritten Chinese text in unconstrained camera-captured documents. *Pattern Recognition*, *108*, Article 107559.

Zhang, Xu-Yao, & Liu, Cheng-Lin (2012). Writer adaptation with style transfer mapping. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(7), 1773–1787.

Zhang, Yaping, Nie, Shuai, Liu, Wenju, Xu, Xing, Zhang, Dongxiang, & Shen, Heng Tao (2019). Sequence-to-sequence domain adaptation network for robust text image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 2740–2749).

Zhang, Zhao, Tang, Zemin, Wang, Yang, Zhang, Zheng, Zhan, Choujun, Zha, Zhengjun, et al. (2021). Dense residual network: Enhancing global dense feature flow for character recognition. *Neural Networks*, *139*, 77–85.

Zhang, Jianshu, Zhu, Yixing, Du, Jun, & Dai, Lirong (2018). Radical analysis network for zero-shot learning in printed Chinese character recognition. In *2018 IEEE international conference on multimedia and expo* (pp. 1–6). IEEE.