IJDAR (2017) 20:69–78
DOI 10.1007/s10032-017-0280-z

ORIGINAL PAPER

Writer adaptation via deeply learned features for online Chinese handwriting recognition

Jun Du[1] · **Jian-Fang Zhai**[1] · **Jin-Shui Hu**[2]

Received: 28 October 2015 / Revised: 6 October 2016 / Accepted: 9 January 2017 / Published online: 19 January 2017

Abstract This paper proposes a novel framework of writer adaptation based on deeply learned features for online handwritten Chinese character recognition. Our motivation is to further boost the state-of-the-art deep learning-based recognizer by using writer adaptation techniques. First, to perform an effective and flexible writer adaptation, we propose a tandem architecture design for the feature extraction and classification. Specifically, a deep neural network (DNN) or convolutional neural network (CNN) is adopted to extract the deeply learned features which are used to build a discriminatively trained prototype-based classifier initialized by Linde–Buzo–Gray clustering techniques. In this way, the feature extractor can fully utilize the useful information of a DNN or CNN. Meanwhile, the prototype-based classifier could be designed more compact and efficient as a practical solution. Second, the writer adaption is performed via a linear transformation of the deeply learned features which is optimized with a sample separation margin-based minimum classification error criterion. Furthermore, we improve the generalization capability of the previously proposed discriminative linear regression approach for writer adaptation by using the linear interpolation of two transformations and adaptation data perturbation. The experiments on the tasks of both the CASIA-OLHWDB benchmark and an in-house corpus with a vocabulary of 20,936 characters demonstrate the effectiveness of our proposed approach.

Keywords Writer adaptation · Deep neural network · Convolutional neural network · Handwriting recognition

✉ Jun Du
jundu@ustc.edu.cn

Jian-Fang Zhai
jfzhai@iflytek.com

Jin-Shui Hu
jshu@iflytek.com

[1] University of Science and Technology of China, Hefei, Anhui, People's Republic of China

[2] iFlytek Research, Hefei, Anhui, People's Republic of China

1 Introduction

In the mobile internet era, online handwritten Chinese character recognition as an input mode has become increasingly popular on portable devices (e.g., Smartphone, Tablet). Previously, there were several solutions that could be developed to build the online Chinese handwriting recognizer (e.g., [1,2]). But in real applications, the user experience is not always satisfactory due to the different writing styles, especially the cursive and continuous writing styles. To address this problem, writer adaptation is one solution which aims to improve the recognition performance and user experience of a single writer by using the corresponding data samples in a two-pass recognition manner via an *unsupervised adaptation* strategy, or by using a small amount of labeled adaptation data samples collected from the target writer via a *supervised adaptation* strategy. For this study, we focus on the latter.

For the past several decades, many writer adaptation approaches have been proposed. Several representative adaptation techniques for different models are reviewed as follows. For example, a writer adaptive online character recognizer was designed via a time delay neural network where the last layer is adopted as a linear optimal hyperplane classifier for adapting to new writing styles in [3]. In [4], a radial basis function (RBF) network was used as a so-called output adaptation module on top of standard neural networks. In [5], writer adaptation via maximum likelihood linear regres-

sion (MLLR) or maximum a posteriori (MAP) criterion was conducted for a hidden Markov model (HMM)-based recognition system for cursive German script. Several strategies in [6] were proposed for adaptation of a prototype-based classifier, including adding new prototypes, reshaping existing prototypes, and inactivating poorly performing prototypes. In [7], a support vector machine (SVM)-based classifier with a biased regularization was adopted for personalization.

Most of the abovementioned adaptation approaches on top of different classifiers are applied for recognition of the western languages where there are only a small number of character classes (e.g., dozens of characters for English). However, for online handwritten Chinese character recognition, it is more challenging due to the large vocabulary task with thousands of character classes. By considering the memory footprint, runtime latency, and also the recognition accuracy, one widely used product solution is the prototype-based classifier as reported in [2] initialized by Linde–Buzo–Gray (LBG, the acronyms for the family names of three authors Yoseph Linde, Andres Buzo and Robert M. Gray) clustering techniques [8] and further optimized via a so-called sample separation margin-based minimum classification error (SSM-MCE) criterion [9], which can be made both compact [10] and efficient [11] in the recognition stage. Based on the prototype-based classifier, one type of the adaptation methods is to use a linear feature transformation for adapting the writing styles via different criteria, e.g., style transfer mapping (STM) with a least regularized weighted squared error criterion [12,13], or discriminative linear regression (DLR) with SSM-MCE criterion [14,15].

Recently, deep learning technologies have been widely used for pattern recognition problems, especially in speech and computer vision areas [16,17]. For both online and offline handwritten Chinese character recognition, the convolutional neural network (CNN) originally proposed in [18] as a classifier makes a new milestone in terms of recognition accuracy [19,20]. In the 2013 Chinese handwriting recognition competition task, the system from University of Warwick [21] using a deep CNN achieved the best performance on the online character recognition task, while the system from the Dalle Molle Institute for Artificial Intelligence (IDSIA) using multi-column deep CNNs [22] and the system from Fujitsu R&D Center using multiple CNNs [23] reported the two best results for the offline character recognition task.

So the motivation of this study is to further improve the recognition accuracy by writer adaptation on top of the high-performance deep learning-based classifier and make it a practical solution [15,24]. First, our proposed recognizer can be considered as a hybrid version of deep learning-based and prototype-based recognizers in which deep neural network (DNN) with the first several hidden layers or CNN with the convolutional layers is adopted as a highly nonlinear and discriminative feature extractor, while the last fully connected layer is replaced by a prototype-based classifier. The use of a DNN or CNN as a feature extractor is motivated by the concept of deep feature representation which is one widely accepted explanation to the success of deep learning techniques in many applications. Our design can maintain the high recognition accuracy of a deep learning-based classifier, while many well-established techniques associated with the prototype-based classifier can be adopted to make it both compact and efficient as a practical solution. According to [25], a DNN-based feature extractor plus prototype-based classifier can be designed to be more compact and efficient than the prototype-based classifier with the traditional feature extraction. In this work, the CNN-based feature extractor is adopted and compared, which is often more time-consuming than the DNN-based feature extractor, but with higher recognition performance. Second, we adopt the STM and DLR approaches for writer adaptation and propose an improved DLR approach by using a simple regularization method and adaptation data perturbation. Our experiments conducted on both the CASIA-OLHWDB benchmark [26], and the data collected from mobile internet users with a vocabulary of 20,936 characters show the effectiveness of our proposed approach. Overall, the main contribution of this study is the design of a novel writer adaptive recognizer incorporating with the deep learning techniques, specifically by using a CNN-based feature extractor and an improved writer adaptation approach via a simple regularization and data perturbation strategy.

The remainder of the paper is organized as follows. In Sect. 2, the prototype-based classifier is introduced. In Sect. 3, different feature extractors are presented. In Sect. 4, the writer adaptation approach based on linear transformation is described. In Sect. 5, we report experimental results. Finally, we conclude the paper in Sect. 6.

2 Prototype-based classifiers

A system overview of our prototype-based recognizer is illustrated in Fig. 1. In the training stage, first the feature vectors $\mathcal{X} = \{\mathbf{x}_r \in \mathcal{R}^D | r = 1, \ldots, R\}$ are extracted from the training samples by different approaches which will be described in the next section. Then a multi-prototype-based classifier which can recognize M character classes $\{C_i | i = 1, \ldots, M\}$ is constructed by using the LBG clustering algorithm [8]. Each class C_i is represented by a set of K_i prototypes $\lambda_i = \{\mathbf{m}_{ik} \in \mathcal{R}^D | k = 1, \ldots, K_i\}$, where $\mathbf{m}_{ik}$ is the kth prototype of the ith class with D dimension. Let us use $\Lambda = \{\lambda_i\}$ to denote the set of prototypes for all classes, which is refined by minimizing the following SSM-MCE objective function:

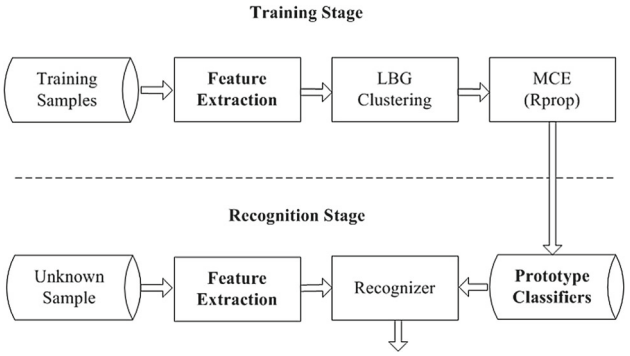

Fig. 1 Overall development flow of the recognizer

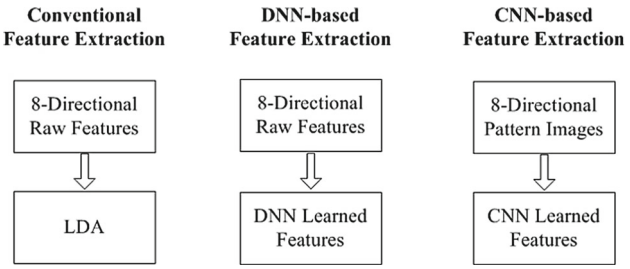

Fig. 2 Comparison of three feature extractions

$$l(\mathcal{X};\Lambda)=\frac{1}{R}\sum_{r=1}^{R}\frac{1}{1+\exp[-\alpha d(\mathbf{x}_r;\Lambda)+\beta]} \tag{1}$$

where α, β are two control parameters, and $d(\mathbf{x}_r;\Lambda)$ is a misclassification measure defined by a so-called sample separation margin (SSM) [24]. To optimize this objective function with the parameter set Λ, the Rprop algorithm is adopted as described in [2]. At the recognition stage, with the feature vector extracted from the unknown sample, the recognition is performed based on a prototype-based classifier.

3 Feature extraction

In this work, three feature extraction approaches are compared, as shown in Fig. 2. The first one is the conventional feature extraction for Chinese handwriting, namely the 8-directional raw feature extraction followed by linear discriminant analysis (LDA) transformation for dimension reduction [27]. The second one is the DNN-based feature extraction which also uses 8-directional raw features as the input of the following DNN for feature learning. The third one is the CNN-based feature extraction including the generation of 8-directional pattern images which are the intermediate products of the 8-directional raw features [27] and the CNN-based feature learning. The three systems corresponding to those features are denoted as LDA-MP, DNN-MP, CNN-MP, respectively. Next, the details of DNN-based and CNN-based feature learning are described.

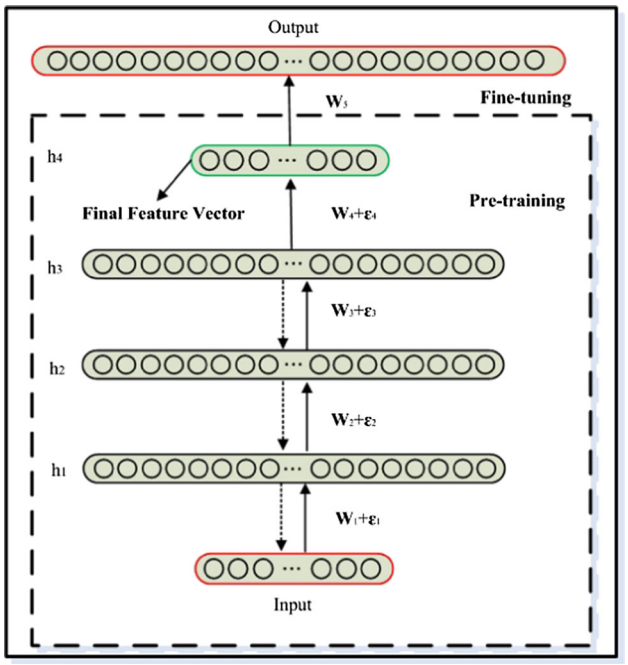

Fig. 3 DNN-based feature learning

3.1 DNN-based feature learning

As shown in Fig. 3, DNN learned features [28,29] can be generated from a DNN where one of the internal layers (bottleneck layer) has a small number of hidden units, relative to the size of the other layers. The bottleneck layer creates a constriction in the network that forces the information pertinent to classification into a low-dimensional representation. In this work, bottleneck features are created from a deep neural network trained to predict character classes. The inputs to the hidden units of the bottleneck layer are used as features for a prototype-based classifier. These bottleneck features represent a nonlinear and discriminative transformation of the input features. The training procedure of this DNN consists of generative pre-training via a restricted Boltzmann machine (RBM) [30] layer-by-layer and supervised fine-tuning using the cross-entropy criterion [25]. According to [25], the design of the DNN-MP recognizer can be even more compact and efficient (with less runtime latency) than the LDA-MP recognizer meanwhile with a higher recognition accuracy.

3.2 CNN-based feature learning

We adopt a modified CNN architecture with alternating convolutional and max-pooling layers proposed in [21], which is inspired by [31,32]. As shown in Fig. 4, the CNN consists of

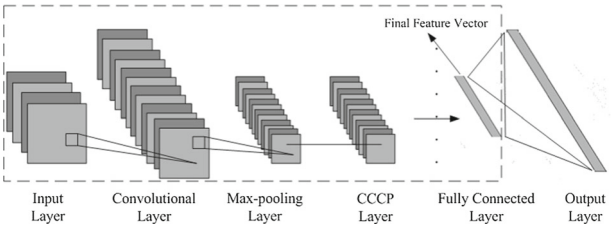

Fig. 4 CNN-based feature learning

one input layer, one fully connected layer, one output layer, and several groups of convolutional layers, max-pooling layers, and cascaded cross channel parametric (CCCP) layers. According to [33], one CCCP layer is equivalent to a convolutional layer with a 1×1 convolution kernel, which can make the CNN-based recognizer much more compact and efficient while maintaining the recognition accuracy. Eight-directional pattern images plus one image with original ink trajectories are collected for the input layer, which is one alternative to the signature representation mentioned in [21]. Each node in the output layer corresponds to one character class. All layers of the CNN model are trained using the backpropagation algorithm. For error propagation and weight adaptation in fully connected, convolutional, and max-pooling layers we follow the standard procedure. After CNN training, only the parameters before the fully connected layer are left to extract the final feature vector which is fed to the prototype-based classifier described in the next section. So the CNN-based feature extractor which is irrelevant to the number of the character classes could be designed very compact, e.g., with a size of <2 mega bytes for the CNN architecture used in the experiments conducted on the in-house corpus.

4 Writer adaptation

Given a set of labeled adaptation data $\mathcal{Y} = \{\mathbf{y}_r \in \mathcal{R}^D | r = 1, \ldots, R'\}$ collected from a single writer, the writer adaptation is performed via a linear feature transformation of the feature vector (LDA/DNN/CNN-based feature):

$$\mathbf{x}_r = \mathcal{F}(\mathbf{y}_r; \Theta) = \mathbf{A}\mathbf{y}_r + \mathbf{b} \tag{2}$$

where $\Theta = \{\mathbf{A}, \mathbf{b}\}$ denotes the set of transform parameters. $\mathbf{A}$ is a $D \times D$ nonsingular matrix and $\mathbf{b}$ is a D-dimensional bias vector. $\mathbf{y}_r$ and $\mathbf{x}_r$ are the rth D-dimensional input and transformed feature vectors, respectively. In this study, we focus on the supervised writer adaptation as in most cases the labels of adaptation samples could be provided by the users when they select the true label from the recognition candidates displayed on the interface of the handwriting application. In the recognition stage, the estimated parameters $\{\mathbf{A}, \mathbf{b}\}$ are used to transform the feature vector of each unknown sample first, which is then fed to the classifier for recognition.

4.1 Style transfer mapping

One popular approach is the style transfer mapping proposed in [13]. In STM, first we define the source point set as the set of feature vectors of adaptation samples (i.e., $\mathcal{Y}$), and the target point set as the set of the corresponding prototypes with the minimum Euclidean distances to those features vectors. Then a style transfer matrix $\mathbf{A}^{\mathrm{STM}}$ could be found to solve the following optimization problem with a least regularized squared error criterion:

$$\min_{\mathbf{A}^{\mathrm{STM}}} \sum_{r=1}^{R'} \|\mathbf{A}^{\mathrm{STM}}\mathbf{y}_r - \mathbf{t}_r\|_2^2 + \beta_1 \|\mathbf{A}^{\mathrm{STM}} - \mathbf{I}\|_2^2 \tag{3}$$

where $\mathbf{t}_r$ is the target point set as the corresponding prototype with the minimum Euclidean distances to the input feature vector. The hyperparameter β_1 can be set according to [12]. In [13], an extended formulation with a bias vector is also derived. In our experiments, the STM approach in [12] is used for performance comparison. With a closed-form solution of the above problem for $\mathbf{A}^{\mathrm{STM}}$, STM-based approach is quite effective when there is only a small amount of adaptation data.

4.2 Discriminative linear regression

But the recognition performance is easily saturated with increased adaptation data. The reason might be the objective function of STM, namely the least regularized weighted squared error criterion, only considers the fitting between the transformed features and the target features, not discriminating different character classes from the aspect of pattern classification. Another adaptation approach is discriminative linear regression, verified to be effective for handwriting recognition [15], especially with a large amount of adaptation data. To learn the corresponding transformation $\Theta^{\mathrm{DLR}} = \{\mathbf{A}^{\mathrm{DLR}}, \mathbf{b}^{\mathrm{DLR}}\}$, the same SSM-MCE objective function as the criterion of the classifier training is adopted:

$$l(\mathcal{Y}; \Lambda, \Theta^{\mathrm{DLR}}) = \frac{1}{R'} \sum_{r=1}^{R'} \frac{1}{1 + \exp[-\alpha d(\mathbf{y}_r; \Lambda, \Theta^{\mathrm{DLR}}) + \beta]} \tag{4}$$

where

$$d(\mathbf{y}_r; \Lambda, \Theta^{\mathrm{DLR}}) = \frac{-g_p(\mathbf{x}_r; \lambda_p) + g_q(\mathbf{x}_r; \lambda_q)}{2 \parallel \mathbf{m}_{p\hat{k}} - \mathbf{m}_{q\bar{k}} \parallel} . \tag{5}$$

$\mathbf{x}_r$ in Eq. (5) is defined in Eq. (2). The optimization procedure for Θ^{DLR} is the same as in [14,24]. From the viewpoint of classification measure, SSM-MCE seems a more reasonable objective function to learn the feature transform compared with STM, which is also confirmed by our experiments.

4.3 Improved DLR

To improve the regularization of DLR-based adaptation and fully tap its potential with large adaptation data, two strategies are presented. The first one is a simple linear interpolation between STM-based and DLR-based transformations as a regularization to DLR:

$$\mathbf{A}^{\mathrm{IDLR}} = \beta_2 \mathbf{A}^{\mathrm{DLR}} + (1 - \beta_2)\mathbf{A}^{\mathrm{STM}} \tag{6}$$

where the factor β_2 is adaptive with the number of adaptation data R' and can be empirically set as:

$$\beta_2 = 0.5 + 0.1 * \log_2 \frac{R'}{N_{\mathrm{T}}} . \tag{7}$$

N_{T} is a threshold for the number of adaptation data R'. Eq. (7) only holds for β_2 in the range [0,1]. If $\beta_2 < 0$, then it will be set to 0, which implies that we only use the STM-based transformation with a very few amount of adaptation data. If $\beta_2 > 1$, then it will be set to 1, which indicates that only the DLR-based transformation is adopted with quite a large amount of adaptation data.

The second strategy is a simple perturbation, which can synthesize more adaptation data by using deformations of handwritten characters. This is motivated by the recent success of using distorted samples to improve the generalization performance of DNN-based classifier [34] and other classifiers [35]. We believe that the perturbation could also work for writer adaptation by carefully designing the deformations. In this work, the distorted operations including rotation, shearing, and scaling are randomly performed on the original adaptation samples.

5 Experiments and results

The experiments are first conducted on an in-house corpus for the task of recognizing isolated online handwritten Chinese characters with a vocabulary of 20,936 character classes. For training, we use in total 25,560,960 character samples, more than 1000 samples per character class. The training data are collected from a large amount of real users with different mobile devices, which is quite different from the in-house corpus used in our recent work [15]. As for the adaptation and test data, the samples of 200 new users collected across several months are used, including total 1,849,643 samples within 4799 classes. For each writer, one half of the samples is randomly selected as adaptation data to learn the feature transformation parameters, while the other half is used as test data.

To evaluate on a standard benchmark, we also verify our approach with a CNN-based feature extractor on the public database released by the Institute of Automation of the Chinese Academy of Sciences (CASIA) [26]. By combining all of the OLHWDB1.0 and the training set of the OLHWDB1.1 datasets, there are totally 2,468,624 samples of 3755 character classes for training of the CNN and the prototype-based classifier. We use all 60 writers from the competition data [20] and randomly divide them into two halves for each writer, one half as the adaptation set and the other half as the test set.

5.1 Parameter seining and development

For the experiments on the in-house corpus, a 392-dimensional 8-directional raw feature vector is extracted which is then transformed to a 100-dimensional feature vector by LDA. The DNN architecture is 392-1024-1024-1024-100-20936, where three hidden layers are used with 1024 nodes for each, and the input/output size is 392/20936, respectively. The dimension of the DNN-based feature vector is 100, corresponding to the size of bottleneck layer. The CNN architecture is $48 \times 48 \times 9$-100C3-MP2-50CCCP-150C2-MP2-75CCCP-200C2-MP2-100CCCP-250C2-MP2-100CCCP-100N-20936, where xCy is a convolutional layer with x maps and $y \times y$ filters, MPy is a max-pooling layer with $y \times y$ pooling size, xCCCP is a CCCP layer with x maps, and xN is a fully connected layer with x neurons. This configuration denotes that the input layer is $48 \times 48 \times 9$ pattern images, the dimension of the final feature vector from the fully connected layer is 100, and the output layer is 20,936 character classes. We use the Caffe toolkit to implement the CNN, and the total training time of this CNN is about 53 hours. It should be noted that this compact CNN feature extractor with a small amount of parameters in each layer excluding the input layer can still maintain the high recognition accuracy of the CNN architecture with much more parameters in the corresponding layers. For Rprop-based SSM-MCE training and adaptation, the setting of the control parameters is as in [2,14]. For IDLR, N_{T} is set to 2048 and we use 50-fold distorted samples for perturbation. The number of prototypes for each character class is non-uniformly set as 4 for 2864 commonly used characters and 1 for 18,072 uncommonly used characters. To handle the large-scale training data, the computations of the LBG clustering, SSM-MCE training, and adaptation with the Rprop algorithm are parallelized on the CPU cluster, while DNN/CNN training is performed on GPUs.

For the experiments on the CASIA database, the architecture of the CNN is $96 \times 96 \times 9$-150C3-MP2-300C2-MP2-450C2-MP2-600C2-MP2-700C2-900N-3755. The learning rate is initially set to 0.02 and decayed with an exponential factor of 0.999995 in each epoch. The strategy of weight initialization is "xavier" [36], which is an approach to uniform sampling in the interval adaptive to the size of input layer. The number of epochs is 630000. The dropout strategy is only used for the last four layers (including the fully connected layer) with the corresponding probabilities 0.1, 0.2, 0.3, 0.4, respectively. The sample distortion is not used. For the prototype-based classifier, four prototypes are used for all classes. In STM, the hyperparameter $\tilde{\beta}$ in [12] is set to 2. In DLR and IDLR, α and β are set to 20 and 0, respectively. Please note that the adaptation data perturbation is only used in IDLR. All the other parameters are the same as those in the experiments on the in-house corpus. Finally, to help the reader to implement the CNN, a recipe for building the DeepCNet [21,37], which won the task 3 of 2013 Chinese handwriting recognition competition, is given. Our implementation using the Caffe toolkit mostly refers to this recipe.

It should be emphasized that the parameter settings, e.g., DNN or CNN architecture, might vary for different tasks in terms of vocabulary size, the amount and diversity of training data, etc. For example, the initial learning rate of CNN, one important factor to control the learning convergence, is usually tuned on a small cross-validation set randomly held out from the training set. In the DNN/CNN training, the parameter setting is really like an art. But fortunately, with the development of deep learning in many applications (speech recognition, image recognition, handwriting recognition, etc.) and the corresponding toolkits, there are many papers or documents to help you to rapidly design a good neural network for a specified task. In our study, the design of DNN/CNN architectures mainly refers to the previous work such as [16,17,19,22,25,31,33,36,37].

5.2 Experiments on the in-house corpus

Table 1 gives a performance comparison of different feature extraction systems on the test set of all 200 writers. The systems "LDA-MP," "DNN-MP," "CNN-MP" use the same multi-prototype-based classifiers with different approaches to extract the same 100-dimensional feature vector. We can observe that without writer adaptation CNN-based feature extraction significantly outperforms the other two feature extractors, with relative character error rate (CER) reductions of 34 and 21% over the LDA-based and DNN-based feature extraction, respectively. The reason why "CNN-MP" is better than "DNN-MP" might be that more input information is provided by the CNN than the DNN (392 vs. $48 \times 48 \times 9$) and CNN can handle it well via the convolutional operations.

Table 1 Performance (CER in %) comparison of different feature extraction systems on the test set

	LDA-MP	DNN-MP	CNN-MP
Baseline	8.84	7.35	5.83
STM	6.86	6.06	4.5

Table 2 Performance (CER in %) comparison of CNN and CNN-MP recognizers on the test set

System	CNN-MP	CNN
CER	5.83	5.63

Table 3 Performance (CER in %) comparison of different adaptation approaches on the test set using 6000 adaptation samples

	STM	DLR	IDLR-LI	IDLR-DP	IDLR
CER	4.43	3.91	3.83	3.76	3.68

For the adaptation using STM, 1000 samples for each writer are used for learning the transformation. Large performance gains are consistently achieved by STM adaptation over the baseline for all systems. With STM adaptation "CNN-MP" is still the best system, yielding relative CER reductions of 34% and 26% over the "LDA-MP" and "DNN-MP" approaches, respectively. By considering this, all the subsequent adaptation experiments are conducted on top of the "CNN-MP" system.

Table 2 shows a performance comparison of two CNN-based systems averaged on all 200 writers of the test set without writer adaptation. The "CNN-MP" system uses the multi-prototype-based classifier with the CNN-based feature extraction proposed in this work. The "CNN" system is a conventional purely CNN-based classifier. Our proposed "CNN-MP" system is slightly worse than the "CNN" system, but can be designed to be more compact and is efficient as a practical solution. Furthermore, the "CNN-MP" system with writer adaptation can significantly outperform the best "CNN" system, which is demonstrated in the following experiments.

Table 3 compares different adaptation approaches on the test set using 6000 adaptation samples. The baseline system without adaptation is "CNN-MP" in Table 2. "IDLR-LI" and "IDLR-DP" refer to the IDLR systems using only linear interpolation or data perturbation, respectively. Obviously, both "IDLR-LI" and "IDLR-DP" can generate performance gains over DLR demonstrating the effectiveness of each strategy. Also an additional gain can be achieved by combining two strategies in the final IDLR system.

Table 4 lists a performance comparison of three adaptation approaches with different amounts of adaptation data

Table 4 Performance (CER in %) comparison of three adaptation approaches with different amounts of adaptation data

# of adaptation samples	200	500	1000	6000
STM	4.8	4.6	4.5	4.43
DLR	4.77	4.5	4.31	3.91
IDLR	4.7	4.42	4.18	3.68

Table 5 Comparison of different classifiers with CNN-based feature extraction in terms of model size and latency

	LC	MPC	MPC-Opt
Model size (MB)	8.0	11.3	2.1
Runtime latency (ms)	4.2	6.1	2.0

averaged on 200 writers of the test set. The baseline system without adaptation is "CNN-MP" in Table 2. Four configurations with different numbers of adaptation samples are compared, namely 200, 500, 1000, and 6000. First, on top of the high-performance baseline with a 5.83% CER, the STM-based writer adaptation with more than 500 adaptation samples can achieve more than a 20% relative error rate reduction. Second, with the increased amount of adaptation data, the performance of the STM-based approach is quickly saturated, while the error rates of both DLR and IDLR are still significantly decreased. The superiority of DLR/IDLR over STM can be explained as the objective function of the DLR/IDLR-based adaptation, namely the SSM-MCE criterion which is exactly the same as that for the prototype-based classifier training, is closer to the recognition measure than the least regularized weighted squared error criterion used in the STM-based adaptation. Also the discriminative criterion will be more powerful with more adaptation data. We can imagine that the performance gap between IDLR and STM will be larger with more adaptation data, which results in a better user experience. Finally, the effectiveness of IDLR can be verified by the observation that the relative performance gain of IDLR from DLR is consistently comparable to that between DLR and STM for different amounts of adaptation data. Overall, a 16.9% relative character error rate reduction is yielded by IDLR over STM with 6000 adaptation samples.

Figure 5 gives a performance comparison of different adaptation approaches for each test set of 20 selected writers sorted by the baseline recognition performance. For each writer, no more than 6000 adaptation samples are used. In general, the similar observations as in Table 4 for each writer can be made with the same order sorted by character error rate, namely Baseline > STM > DLR > IDLR. There is only one exception on writer No.13 where the performance of DLR is the same as STM. Although the baseline recognition error rate varies from 17.37 to 1.27% for different writers,

Table 6 Performance (CER in %) comparison of different adaptation approaches for CASIA competition data of each writer

ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Baseline	6.66	2.72	3.94	8.33	1.76	1.97	3.96	3.51	6.04	1.55	4.28	1.39	2.40	5.46	2.08	4.58	3.25	0.59	2.56	12.59
STM	5.70	2.62	3.99	8.28	1.70	1.76	3.42	2.61	5.61	1.55	3.58	1.17	1.97	5.52	1.55	4.26	2.77	0.59	2.45	11.57
DLR	5.91	2.14	3.99	7.90	1.76	2.08	3.53	2.56	5.71	1.50	3.58	1.07	1.86	4.98	1.44	4.37	2.99	0.75	2.34	11.40
IDLR	5.59	2.35	3.78	7.96	1.65	1.86	3.37	2.40	5.23	1.50	3.53	1.01	2.08	5.03	1.49	3.94	2.72	0.59	2.45	11.05
	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
Baseline	0.91	1.33	2.13	2.08	2.56	2.08	1.49	3.14	2.56	1.49	1.60	3.25	2.77	22.09	1.60	4.06	4.95	1.65	4.21	3.04
STM	0.85	1.17	1.87	1.92	2.40	1.70	1.33	2.61	2.03	1.39	1.33	2.98	2.40	20.39	1.60	3.52	4.74	1.60	3.57	2.56
DLR	0.75	1.22	1.92	1.81	2.61	1.97	1.28	2.61	2.08	1.07	1.23	2.88	2.61	19.72	1.65	3.20	4.69	1.44	3.41	2.02
IDLR	0.75	1.01	1.92	1.81	2.40	1.65	1.44	2.50	1.98	1.33	1.33	2.82	2.34	19.34	1.44	3.20	4.58	1.60	3.62	2.29
	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
Baseline	1.39	4.86	0.96	4.15	0.91	1.33	11.22	3.53	1.23	5.22	0.96	2.93	2.29	1.76	2.56	0.91	0.80	1.12	1.34	3.78
STM	1.33	4.06	1.01	3.78	0.96	1.39	9.72	3.69	1.17	5.01	0.80	2.66	2.13	1.66	2.24	0.80	0.69	1.06	1.22	3.41
DLR	1.23	4.27	1.06	4.10	0.85	1.49	8.92	3.59	1.28	4.69	0.91	2.66	1.86	1.55	2.45	0.80	0.75	0.96	1.06	3.36
IDLR	1.39	4.01	0.91	3.67	0.96	1.39	9.03	3.59	1.12	5.01	0.80	2.56	1.92	1.55	2.13	0.80	0.64	1.06	1.28	3.25

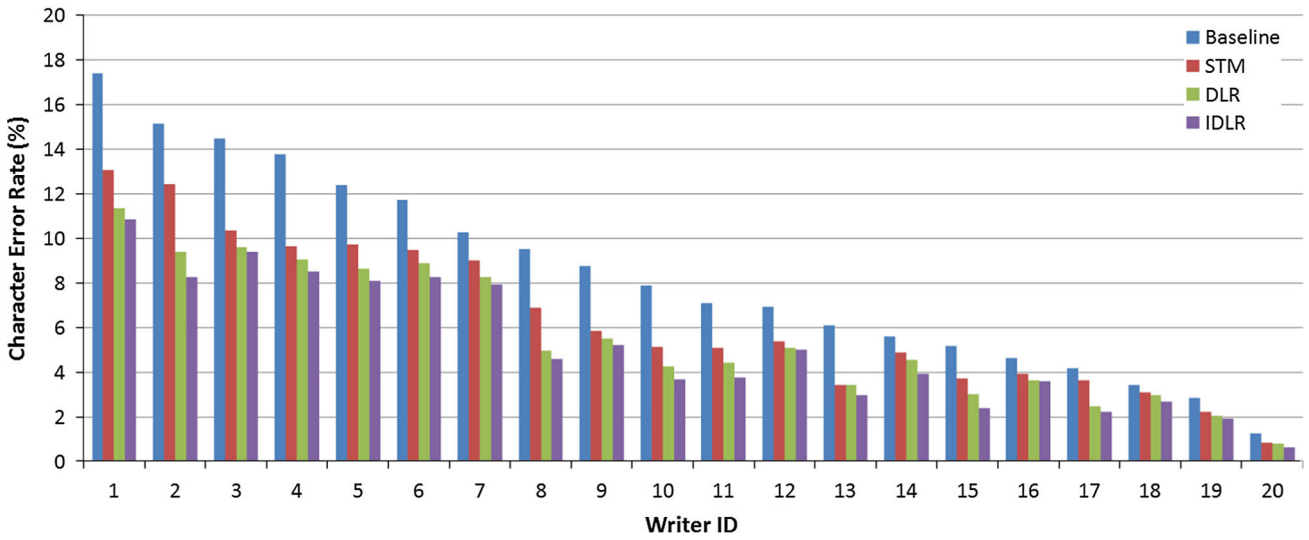

Fig. 5 Performance (sorted by Baseline CER in %) comparison of different adaptation approaches for each test set of 20 writers selected from all 200 new users of the in-house corpus

the IDLR-based adaptation achieves consistently the best recognition performance, with the relative character error rate reductions of 54% at most and 22% at least over the baseline system.

Finally, we compare different classifiers with CNN-based feature extraction in terms of model size and runtime latency (per character) as shown in Table 5. LC denotes the linear classifier represented by the last hidden layer of the CNN (100*20,936), while MPC refers to the multi-prototype-based classifier. MPC-Opt is an optimized version of MPC by using the well-established split vector quantization (VQ) [10] and fast match tree [11] techniques. In split VQ, 50 sub-vectors are used with two dimensions for each. The codebook size is 256. For the fast match tree, 2048 buckets are used. With this configuration, MPC-Opt can achieve exactly the same recognition accuracy as MPC on the test set. From Table 5, although the model size and runtime latency of MPC is comparable or slightly more than LC, MPC-Opt needs much smaller resources and is more efficient than both.

5.3 Experiments on the CASIA-OLHWDB benchmark

In Table 6, the performance comparison of different adaptation approaches for CASIA competition data of each writer is given. The baseline system with an average CER of 3.34% among 60 writers is denoted as CNN-MP, which is namely the CNN-based feature extraction plus multi-prototype-based classifier. Most of the STM results are better than those of the baseline system, with a relative CER reduction of 9.0% in average. The DLR results are mixed compared with the STM results, while most of the IDLR results are better than the STM or DLR results, yielding a relative CER reduction of 12.9% over the baseline in average. Obviously, the relative gains from all adaptation approaches on the CASIA-OLHWDB dataset are smaller than those in Fig. 5. The possible reasons for those different observations of writer adaptation are summarized as: (1) The baseline performance of the CASIA-OLHWDB data is higher than that of real user data in average, (2) the writer adaptation on the CASIA-OLHWDB task is more difficult than that on real user data as there is no character class overlap between adaptation and test set (only one sample for each character), while for real applications this overlap is permissible which implies that more similar writing styles of test data can be learned from adaptation data, (3) the amount of adaptation data for each writer in the CASIA-OLHWDB task (no more than 2000 samples) is less than that of real user data (6000 samples at most).

Finally, in Fig. 6, we make an interesting comparison between two models trained with the user data of the in-house corpus (denoted as UD) and the CASIA-OLHWDB database (denoted as CASIA) for the writer adaptation on the same real users as in Fig. 5. To perform a fair comparison between these, only the samples of 3755 character classes are used for adaptation and test set to guarantee that there is no sample out of vocabulary for the two models. First, we could observe that without adaptation the UD model achieves much better recognition performance than the CASIA model on the real user data. In comparison to the results in Table 6, it is clear that there are huge differences of recognition accuracy between two test sets using the same CASIA model. By considering that a larger CNN and a higher dimensional feature vector for the prototype-based classifier is adopted for the CASIA model, we could conclude that the training data from diversified real users (e.g., used to train the UD model) are crucial to the recognition accuracy in real

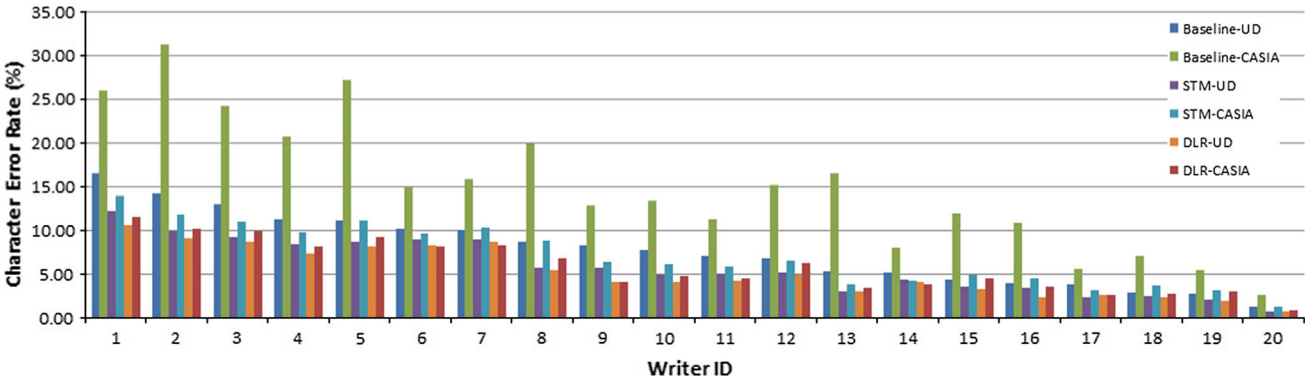

Fig. 6 Performance (sorted by Baseline-UD CER in %) comparison of two models (trained using UD and CASIA data) with different adaptation approaches for real user data of each writer. (*UD* user data of the in-house corpus, *CASIA* CASIA-OLHWDB database)

applications. Second, the STM-CASIA system almost yields 50% of relative CER reductions over the Baseline-CASIA system for most of the writers, which are more significant than those achieved by the UD model. However, after STM-based adaptation, the UD model still outperforms the CASIA model (with only one exception on No. 14), but the performance gap is largely reduced, which is more significant for DLR-based adaptation. The best recognition performances (mostly $<10\%$ CERs) are consistently obtained by the DLR-based adaptation on top of both UD and CASIA models. This observation implies that with writer adaptation, the recognition performance might be not highly dependent on the baseline model. It is really a good news for the developers without the diversified training data at hands. With a weak baseline model, it is believed that the Chinese handwriting recognizer can be progressively boosted with the proposed writer adaptation approach. With more frequent uses of the recognizer by the customers, the higher recognition accuracy can be expected. Finally, for the improvements of STM- and DLR-based adaptation over the two baseline models, similar observations could be made as in Fig. 5.

6 Conclusion

In this work, we investigate the writer adaptation for the prototype-based recognizers using different feature extractions. The experiments are performed on real user data including training, adaptation, and test sets. CNN-MP achieves the best recognition accuracy over LDA-MP and DNN-MP with/without writer adaptation. As for different adaptation approaches, STM-based adaptation is quite effective with a small amount of adaptation data, while DLR approach can significantly improve the recognition accuracy over STM approach with more adaptation data. Furthermore, our proposed IDLR approach by using distorted samples and a simple regularization can yield consistently performance gains over the DLR approach.

Acknowledgements This work was supported in part by the National Natural Science Foundation of China under Grant No. 61671422.

References

1. Liu, C.-L., Sako, H., Fujisawa, H.: Discriminative learning quadratic discriminant function for handwriting recognition. IEEE Trans. Neural Netw. **15**(2), 430–444 (2004)
2. Du, J., Huo, Q.: Designing compact classifiers for rotation-free recognition of large vocabulary online handwritten Chinese characters. In: Proceedings of ICASSP-2012, pp. 1721–1724 (2012)
3. Matić, N., Guyon, I., Denker, J., Vapnik, V.: Writer adaptation for on-line handwritten character recognition. In: Proceedings of ICDAR-1993, pp. 187–191 (1993)
4. Platt, J.C., Matić, N.P.: A constructive RBF network for writer adaptation. In: Proceedings of NIPS-1997 (1997)
5. Brakensiek, A., Kosmala, A., Rigoll, G.: Comparing adaptation techniques for on-line handwriting recognition. In: Proceedings of ICDAR-2001, pp. 486–490 (2001)
6. Vuori, V., Korkeakoulu, T.: Adaptive methods for online recognition of isolated handwritten characters. PhD thesis, Helsinki University of Technology (2002)
7. Kienzle, W., Chellapilla, K.: Personalized handwriting recognition via biased regularization. In: Proceedings of ICML-2006 (2006)
8. Linde, Y., Buzo, A., Gray, R.: An algorithm for vector quantizer design. IEEE Trans. Commun. **28**(1), 84–95 (1980)
9. He, T., Huo, Q.: A study of a new misclassification measure for minimum classification error training of prototype-based pattern classifiers. In: Proceedings of ICPR-2008 (2008)
10. Wang, Y.-Q., Huo, Q.: A study of designing compact recognizers of handwritten Chinese characters using multiple-prototype based classifiers. In: Proceedings of ICPR-2010, pp. 1872–1875 (2010)
11. Feng, Z.-D., Huo, Q.: Confidence guided progressive search and fast match techniques for high performance Chinese/English OCR. In: Proceedings of ICPR-2002, vol. III, pp. 89–92 (2002)
12. Zhang, X.-Y., Liu, C.-L.: Style transfer matrix learning for writer adaptation. In: Proceedings of CVPR-2011, pp. 393–400 (2011)
13. Zhang, X.-Y., Liu, C.-L.: Writer adaptation with style transfer mapping. IEEE Trans. Pattern Anal. Mach. Intell. **35**(7), 1773–1787 (2013)
14. Du, J., Huo, Q.: A discriminative linear regression approach to adaptation of multi-prototype based classifiers and its applications for Chinese OCR. Pattern Recognit. **46**(8), 2313–2322 (2013)
15. Du, J., Hu, J.-S., Zhu, B., Wei, S., Dai, L.-R.: Writer adaptation using bottleneck features and discriminative linear regression for

online handwritten Chinese character recognition. In: Proceedings of ICFHR-2014 (2014)
16. Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., Kingsbury, B.: Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Process. Mag. **29**(6), 82–97 (2012)
17. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Li, F.-F.: ImageNet large scale visual recognition challenge. IJCV **115**(3), 211–252 (2015)
18. Yann, L., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **11**, 2278–2324 (1986)
19. Ciresan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J.: Convolutional neural network committees for handwritten character classification. In: Proceedings of ICDAR-2011 (2011)
20. Yin, F., Wang, Q.-F., Zhang, X.-Y., Liu, C.-L.: ICDAR 2013 Chinese handwriting recognition competition. In: Proceedings of ICDAR-2013, pp. 1464–1470 (2013)
21. Graham, B.: Sparse arrays of signatures for online character recognition. Technical Report, University of Warwick (2013)
22. Ciresan, D.C., Schmidhuber, J.: Multi-column deep neural networks for offline handwritten Chinese character classification. arXiv:1309.0261 (2013)
23. Wu, C., Fan, W., He, Y., Sun, J., Naoi, S.: Handwritten character recognition by alternately trained relaxation convolutional neural network. In: Proceedings of ICFHR-2014 (2014)
24. Du, J., Zhai, J.-F., Hu, J.-S., Zhu, B., Wei, S., Dai, L.-R.: Writer adaptive feature extraction based on convolutional neural networks for online handwritten Chinese character recognition. In: Proceedings of ICDAR-2015 (2015)
25. Du, J., Hu, J.-S., Zhu, B., Wei, S., Dai, L.-R.: A study of designing compact classifiers using deep neural networks for online handwritten Chinese character recognition. In: Proceedings of ICPR-2014 (2014)
26. Liu, C.-L., Yin, F., Wang, D.-H., Wang, Q.-F.: Online and offline handwritten Chinese character recognition: benchmarking on new databases. Pattern Recognit. **46**(1), 155–162 (2013)
27. Bai, Z.-L., Huo, Q.: A study on the use of 8-directional features for online handwritten Chinese character recognition. In: Proceedings of ICDAR-2005, pp. 262–266 (2005)
28. Grézl, F., Karafiát, M., Kontár, S., Ćernocký, J.: Probabilistic and bottle-neck features for LVCSR of meetings. In: Proceedings of ICASSP-2007, pp. 757–761 (2007)
29. Yu, D., Seltzer, M. L.: Improved bottleneck features using pretrained deep neural networks. In: Proceedings of INTERSPEECH-2011, pp. 237–240 (2011)
30. Hinton, G., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
31. Ciresan, D.C., Meier, U., Masci, J., Gambardella, L.M., Schmidhuber, J.: Flexible, high performance convolutional neural networks for image classification. In: Proceedings of IJCAI-2011, pp. 1237–1242 (2011)
32. Scherer, D., Müller, A., Behnke, S.: Evaluation of pooling operations in convolutional architectures for object recognition In: Proceedings of International Conference on Artificial Neural Networks (2010)
33. Lin, M., Chen, Q., Yan, S.-C.: Network in network. arXiv:1312.4400v3 (2014)
34. Ciresan, D., Meier, U., Schmidhuber, J.: Multi-column deep neural networks for image classification In: Proceedings of CVPR-2012, pp. 3643–3649 (2012)
35. Yin, F., Zhou, M.-K., Wang, Q.-F., Liu, C.-L.: Style consistent perturbation for handwritten Chinese character recognition In: Proceedings of ICDAR-2013, pp. 1051–1055 (2013)
36. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks In: Proceedings of AISTATS-2010, pp. 249–256 (2010)
37. http://www2.warwick.ac.uk/fac/sci/statistics/staff/academic-research/graham/deepcnet.zip