DESIGNING COMPACT CLASSIFIERS FOR ROTATION-FREE RECOGNITION OF LARGE VOCABULARY ONLINE HANDWRITTEN CHINESE CHARACTERS

Jun Du, Qiang Huo, Kai Chen

Microsoft Research Asia, Beijing, P. R. China

{jundu,qianghuo}@microsoft.com, chenkai.cn@hotmail.com

ABSTRACT

We present a study of designing compact multiple-prototype based classifiers for rotation-free recognition of online handwritten Chinese characters. Several versions of Rprop algorithms are adopted to optimize a sample-separation-margin based minimum classification error objective function. Split vector quantization technique is used to compress classifier parameters and a fast-match tree is used for efficient recognition. A new preprocessing technique is proposed to achieve rotation-free recognition capability. Promising benchmark results are reported on an online handwritten character recognition task with a vocabulary of 27,720 characters.

Index Terms: handwritten Chinese character recognition, rotation-free, Rprop, MCE

1. INTRODUCTION

With the fast development of mobile internet, online handwritten Chinese character recognition on a Smartphone becomes increasingly popular. To deliver a compelling user experience, a recognizer has to be designed to have a small footprint, run efficiently on a Smartphone, achieve high recognition accuracy, yet be robust to different writing styles of millions of users. In [13], we have presented a solution to achieve most of the above goals except for robustness to certain types of distortions, e.g., rotation. However, a new national standard in China [4] requires that a Chinese handwriting recognition software for text input is able to recognize a character written with a rotational distortion of up to $\pm 45^{\circ}$. In literature, many research work have been reported to improve the robustness of a handwritten Chinese character recognizer to rotational distortions. One type of methods is to add a preprocessing step to transform distorted character(s) to a normal position in recognition stage (e.g., [10, 11, 9, 6]). Reasonable improvement has been reported on the recognition of lines of characters (e.g., [10, 11]) or Chinese words (e.g., [9]), where informative clues for estimating the character orientations are available. For an isolated Chinese character, a character-structure-guided approach to estimating the possible orientations is proposed in [6]. Another type of methods is to perform rotation normalization, i.e., transforming each character to a relatively stable position in both training and recognition stages. In [7], an efficient normalization method, namely "hanging character" method, is proposed and demonstrated to be more effective than most of existing methods. The main idea of the "hanging character" method is to rotate each character sample in such a way to make the direction from the starting point of a character to the centroid of the character upright. By definition, this approach is not robust to writing orders of the strokes.

Inspired by the work in [7], in this study, we propose a new preprocessing technique to achieve rotation-free recognition capability which is more efficient yet insensitive to the stroke orders. As a follow-up study of [13], several versions of Rprop algorithms [12, 8] are adopted to optimize a sample-separation-margin (SSM) based minimum classification error (MCE) objective function [5] because the setting of control parameters for Rprop is much easier than the Quickprop algorithm used in [13]. Benchmark results are reported on an online handwritten character recognition task with a vocabulary of 27,720 characters including 27,533 Chinese characters in GB18030 standard [3], where a fast-match tree is used for efficient recognition.

The remainder of the paper is organized as follows: In Section 2, we present what's new. In Section 3, we report experimental results. Finally we conclude the paper in Section 4.

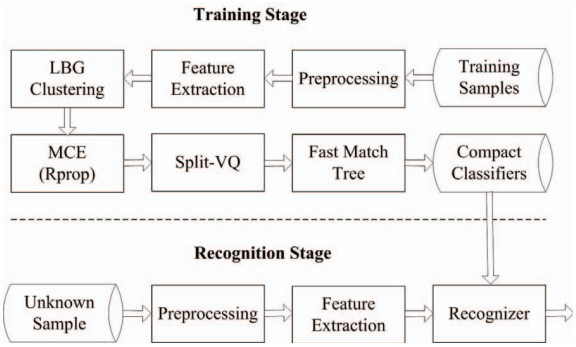

Fig. 1. Overall development flow and architecture.

2. OUR APPROACH

An overall system development flow and architecture is illustrated in Fig. 1. In training stage, each training sample is first transformed by a preprocessing module to achieve the goal of rotation-free recognition. Then a 512-dimensional raw feature vector is extracted [1], which is followed by LDA transformation to obtain a lower dimensional feature vector. After that, a multi-prototype based classifier is constructed by using LBG clustering algorithm, which is then refined by SSM-MCE training with an Rprop algorithm. As in [13], split vector quantization (VQ) technique is used to compress the classifier parameters. Finally, a two-level fast-match tree is constructed as described in [2], which is used at run-time to speed up recognition. Modules in recognition stage are self-explained. In the following subsections, we elaborate on the modules of preprocessing and Rprop-based SSM-MCE training.

2.1. Preprocessing for Rotation-Free Recognition

For an online handwritten character sample with L strokes, let's represent its k^{th} stroke as a sequence of points $\mathbf{P}^k = (P_1^k, P_2^k, ..., P_n^k, ..., P_{N_k}^k)$, where $P_n^k = (x_n^k, y_n^k)$ is the coordinates of the n^{th} point for the k^{th} stroke. Then, two key points can be defined as $\overline{S} = \sum_{k=1}^{L} P_1^k$ and $\overline{E} = \sum_{k=1}^{L} P_{N_k}^k$ which can be considered as the summation of starting points and ending points of all strokes, respectively. Let's use $(x_{\overline{S}}, y_{\overline{S}})$ and $(x_{\overline{E}}, y_{\overline{E}})$ to denote the coordinates of $\overline{S}$ and $\overline{E}$, respectively. Then our transformation is to make the direction from $\overline{S}$ to $\overline{E}$ upright as follows:

$$(x_n^k)_{\text{new}} = x_n^k \sin\theta - y_n^k \cos\theta \quad (1)$$

$$(y_n^k)_{\text{new}} = y_n^k \sin\theta + x_n^k \cos\theta \quad (2)$$

where θ represents the direction from $\overline{S}$ to $\overline{E}$, and satisfies the following equations:

$$\cos\theta = \frac{x_{\overline{E}} - x_{\overline{S}}}{\sqrt{(x_{\overline{E}} - x_{\overline{S}})^2 + (y_{\overline{E}} - y_{\overline{S}})^2}} \quad (3)$$

$$\sin\theta = \frac{y_{\overline{E}} - y_{\overline{S}}}{\sqrt{(x_{\overline{E}} - x_{\overline{S}})^2 + (y_{\overline{E}} - y_{\overline{S}})^2}} \ . \quad (4)$$

Obviously, the above transformation is rotation invariant, therefore a character classifier trained on normalized samples with the above transformation can achieve rotation-free recognition if an unknown character sample is also normalized in the same way. Our experimental results confirm that such a simple method can give very promising results. Compared with the "hanging character" method in [7], our method is by definition insensitive to stroke order, yet computationally more efficient.

2.2. Rprop Optimization for SSM-MCE Training

Suppose our classifier can recognize M character classes denoted as $\{C_i | i = 1, ..., M\}$. We are given a set of training feature vectors $\mathcal{X} = \{\mathbf{x}_r \in \mathcal{R}^D | r = 1, ..., R\}$ together with their labels $\mathcal{L} = \{i_r | r = 1, ...R\}$. For a multi-prototype based classifier, each class C_i is represented by K_i prototypes, $\lambda_i = \{\mathbf{m}_{ik} \in \mathcal{R}^D | k = 1, ..., K_i\}$, where $\mathbf{m}_{ik}$ is the k^{th} prototype of the i^{th} class. Let's use $\Lambda = \{\lambda_i\}$ to denote the set of prototypes. In classification stage, a feature vector $\mathbf{x} \in \mathcal{R}^D$ is first extracted. Then $\mathbf{x}$ is compared with each of the M classes by evaluating a Euclidean distance based discriminant function for each class C_i as follows

$$g_i(\mathbf{x}; \lambda_i) = -\min_k \| \mathbf{x} - \mathbf{m}_{ik} \|^2 \ . \quad (5)$$

The class with the maximum discriminant function score is chosen as the recognized class $r(\mathbf{x}; \mathbf{\Lambda})$, i.e.,

$$r(\mathbf{x}; \mathbf{\Lambda}) = \arg\max_i g_i(\mathbf{x}; \lambda_i) \ . \quad (6)$$

In the training stage, $\mathbf{\Lambda}$ can be estimated by minimizing the following MCE objective function:

$$l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}) = \frac{1}{R} \sum_{r=1}^{R} \frac{1}{1 + \exp[-\alpha d(\mathbf{x}_r, i_r; \mathbf{\Lambda}) + \beta]} \quad (7)$$

where α, β are two control parameters, and $d(\mathbf{x}_r, i_r; \mathbf{\Lambda})$ is a misclassification measure defined by using a so-called sample separation margin (SSM) as follows [5]:

$$d(\mathbf{x}_r, i_r; \mathbf{\Lambda}) = \frac{-g_{i_r}(\mathbf{x}_r; \lambda_{i_r}) + g_q(\mathbf{x}_r; \lambda_q)}{\| \mathbf{m}_{i_r \hat{k}} - \mathbf{m}_{q\bar{k}} \|} \quad (8)$$

where

$$\hat{k} = \arg\min_k \| \mathbf{x}_r - \mathbf{m}_{i_r k} \|^2 \quad (9)$$

$$q = \arg\max_{i \in \mathcal{M}_r} g_i(\mathbf{x}_r; \lambda_i) \quad (10)$$

$$\bar{k} = \arg\min_k \| \mathbf{x}_r - \mathbf{m}_{qk} \|^2 \quad (11)$$

and $\mathcal{M}_r$ is the hypothesis space for the r^{th} sample, excluding the true label i_r.

In previous work [13], a modified Quickprop procedure is used for SSM-MCE training. In this work, 4 improved versions of Rprop algorithms described in [8] are adopted to optimize the objective function in Eq. (7). Our Rprop based SSM-MCE training procedure is as follows:

Step 1: Let $t = 0$. η^+ and η^- ($0 < \eta^- < 1 < \eta^+$) are the increase factor and decrease factor, respectively. Δ_0 is the initial step-size. Δ_{max} and Δ_{min} are the upper limit and lower limit of step-size, respectively. Calculate the derivative of $l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda})$ w.r.t. each m_{ikd} and update the prototype parameters as follows:

$$m_{ikd}^{(t+1)} = m_{ikd}^{(t)} - \text{sign}\left(\frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)})}{\partial m_{ikd}}\right) \Delta_{ikd}^{(t)} \quad (12)$$

$$\Delta m_{ikd}^{(t)} = -\text{sign}\left(\frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)})}{\partial m_{ikd}}\right) \Delta_{ikd}^{(t)} \quad (13)$$

where m_{ikd} is the d^{th} element of $\mathbf{m}_{ik}$, $m_{ikd}^{(t)} = m_{ikd}$, $\Delta_{ikd}^{(t)} = \Delta_0$, and

$$\frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)})}{\partial m_{ikd}} \triangleq \left.\frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda})}{\partial m_{ikd}}\right|_{\mathbf{\Lambda} = \mathbf{\Lambda}^{(t)}} \ . \quad (14)$$

Step 2: Let $t = t + 1$. Define

$$S = \frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t-1)})}{\partial m_{ikd}} \cdot \frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)})}{\partial m_{ikd}} \ . \quad (15)$$

According to [8], there are four modified versions of original Rprop [12], namely Rprop-, Rprop+, iRprop-, iRprop+, which can significantly improve the learning speed.

- Rprop-:

$$\Delta_{ikd}^{(t)} = \begin{cases} \min\left(\eta^+ \Delta_{ikd}^{(t-1)}, \Delta_{\text{max}}\right) & \text{if } S > 0 \\ \max\left(\eta^- \Delta_{ikd}^{(t-1)}, \Delta_{\text{min}}\right) & \text{if } S < 0 \\ \Delta_{ikd}^{(t-1)} & \text{else} \end{cases} \quad (16)$$

$$m_{ikd}^{(t+1)} = m_{ikd}^{(t)} - \text{sign}\left(\frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)})}{\partial m_{ikd}}\right) \Delta_{ikd}^{(t)} \quad (17)$$

- Rprop+:
  If $S > 0$

$$\Delta_{ikd}^{(t)} = \min\left(\eta^+ \Delta_{ikd}^{(t-1)}, \Delta_{\text{max}}\right) \quad (18)$$

$$\Delta m_{ikd}^{(t)} = -\text{sign}\left(\frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)})}{\partial m_{ikd}}\right) \Delta_{ikd}^{(t)} \quad (19)$$

$$m_{ikd}^{(t+1)} = m_{ikd}^{(t)} + \Delta m_{ikd}^{(t)} \quad (20)$$

if $S < 0$

$$\Delta_{ikd}^{(t)} = \max\left(\eta^{-}\Delta_{ikd}^{(t-1)}, \Delta_{\min}\right) \quad (21)$$

$$m_{ikd}^{(t+1)} = m_{ikd}^{(t)} - \Delta m_{ikd}^{(t-1)} \quad (22)$$

$$\frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)})}{\partial m_{ikd}} = 0 \quad (23)$$

else

$$\Delta_{ikd}^{(t)} = \Delta_{ikd}^{(t-1)} \quad (24)$$

$$\Delta m_{ikd}^{(t)} = -\text{sign}\left(\frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)})}{\partial m_{ikd}}\right)\Delta_{ikd}^{(t)} \quad (25)$$

$$m_{ikd}^{(t+1)} = m_{ikd}^{(t)} + \Delta m_{ikd}^{(t)} \ . \quad (26)$$

- iRprop-: Almost the same as Rprop-, we only need to add one statement between Eq. (16) and Eq. (17):

$$\text{If } S < 0, \quad \frac{\partial l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)})}{\partial m_{ikd}} = 0 \ . \quad (27)$$

- iRprop+: Almost the same as Rprop+, we only need to modify Eq. (22) as follows:

$$\text{If } l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t)}) > l(\mathcal{X}, \mathcal{L}; \mathbf{\Lambda}^{(t-1)}),$$
$$m_{ikd}^{(t+1)} = m_{ikd}^{(t)} - \Delta m_{ikd}^{(t-1)} \ . \quad (28)$$

Step 3: Repeat Step 2 $(T_R - 1)$ times.

In the above procedure, the derivative of objective function can be calculated as follows:

$$\frac{\partial l_r}{\partial \mathbf{m}_{i_r\hat{k}}} =$$

$$\alpha l_r(1-l_r)\left[-\frac{\mathbf{x}_r - \mathbf{m}_{i_r\hat{k}}}{\| \mathbf{m}_{i_r\hat{k}} - \mathbf{m}_{q\bar{k}} \|} - d(\mathbf{x}_r, i_r; \mathbf{\Lambda})\frac{\mathbf{m}_{i_r\hat{k}} - \mathbf{m}_{q\bar{k}}}{\| \mathbf{m}_{i_r\hat{k}} - \mathbf{m}_{q\bar{k}} \|^2}\right]$$

$$\frac{\partial l_r}{\partial \mathbf{m}_{q\bar{k}}} =$$

$$\alpha l_r(1-l_r)\left[\frac{\mathbf{x}_r - \mathbf{m}_{q\bar{k}}}{\| \mathbf{m}_{i_r\hat{k}} - \mathbf{m}_{q\bar{k}} \|} - d(\mathbf{x}_r, i_r; \mathbf{\Lambda})\frac{\mathbf{m}_{q\bar{k}} - \mathbf{m}_{i_r\hat{k}}}{\| \mathbf{m}_{q\bar{k}} - \mathbf{m}_{i_r\hat{k}} \|^2}\right]$$

where

$$l_r = \frac{1}{1 + \exp[-\alpha d(\mathbf{x}_r, i_r; \mathbf{\Lambda}) + \beta]} \ . \quad (29)$$

To handle large-scale training data, the tools for LBG clustering and the SSM-based MCE training with both Quickprop and Rprop algorithms have been implemented based upon MSR Asia's MPI-based machine learning platform [14]. This platform was developed on top of Microsoft Windows HPC Server, and optimized for various machine learning algorithms. With this high-performance parallel computing platform, experiments can be run very efficiently for large-scale tasks.

3. EXPERIMENTS AND RESULTS

3.1. Experimental Setup

The experiments are conducted on two tasks of recognizing isolated online handwritten characters with the following two vocabularies:

- **Task-1**: 9,306 character classes including 9,143 Chinese characters, 62 alphanumeric characters, 101 punctuation marks and symbols;
- **Task-2**: 27,720 character classes including 27,533 Chinese characters in GB18030 standard, 62 alphanumeric characters, 8 handwriting gestures, 117 punctuation marks and symbols.

For training we used about 1,000 samples per character in the first vocabulary, and 60 samples per character for those rarely used characters. Three testing sets are used for evaluation: 1) **Regular-1**: 97,221 samples from 6,903 character classes which are written in regular style; 2) **Regular-2**: 84,549 samples from 2,355 uncommon character classes in regular style; 3) **Cursive**: 51,248 samples from 3,324 character classes written in cursive style.

As for the number of prototypes for each character, we use A prototypes for 3755 most frequently used Chinese characters and B prototypes for the rest of character classes. For Rprop-based SSM-MCE training, the control parameters are set as follows: $\alpha = 7$; $\beta = 0$; $T_R = 100$; $\Delta_0 = 0.05$; $\Delta_{\max} = 50$; $\Delta_{\min} = 0$; $\eta^{+} = 1.2$; $\eta^{-} = 0.5$. It is noted that all the control parameters related to Rprop are set empirically as suggested in [8] without tuning. For split VQ compression, the sub-vector dimension is 1. For the fast-match tree, the number of buckets is set to 256 and the top 5 buckets are selected for evaluation.

3.2. Experimental Results

In the first set of experiments, we study the effectiveness of Rprop optimization for SSM-MCE training. Table 1 summarizes a performance (recognition accuracies in %) comparison of Quickprop and four Rprop modifications on three test sets without preprocessing modules on Task-1. The control parameters for Quickprop have to be tuned carefully. Several observations can be made. First, all improved Rprop versions outperform Quickprop. Second, among four modifications of Rprop, iRprop- achieves consistently the best results on three test sets, therefore is adopted for the remaining experiments. It is noted that the dimension of feature vector after LDA transformation is 128 (i.e., $D = 128$) in this set of experiments, but $D = 80$ for all the other experiments to construct more compact classifiers.

In the second set of experiments, we study the effectiveness of preprocessing for rotation-free recognition. Table 2 compares the performance of baseline system without preprocessing and two rotation-free systems using preprocessing for rotation normalization. Although the baseline system achieves better performance, it is not robust to character rotation. Both rotation-free systems achieve very promising recognition accuracies. Compared with "hanging character" method, our method yields significant improvement on test sets of regular style and comparable results on the test set of cursive style. When the number of prototypes is increased, recognition accuracies are improved in most cases, especially for the rotation-free systems. This is reasonable because the additional variabilities introduced by rotation transformation can be handled by using more prototypes.

To verify how the proposed approach scales, a set of experiments are conducted on Task-2. Table 3 summarizes "Top-N" recognition accuracies and footprint comparison of baseline system without preprocessing and two rotation-free systems on three test sets. Similar

Table 1. Performance (recognition accuracies in %) comparison of Quickprop [13] and four Rprop modifications on three test sets without preprocessing for rotation-free recognition ($A = 2$, $B = 1$, $D = 128$).

	Regular-1	Regular-2	Cursive
Quickprop	97.01	96.08	88.98
Rprop-	97.30	96.22	89.73
Rprop+	97.27	96.22	89.72
iRprop-	97.32	96.24	89.81
iRprop+	97.26	96.20	89.64

Table 2. Performance (recognition accuracies in %) comparison of baseline system without preprocessing and two rotation-free systems using "hanging character" method [7] and our method on three test sets on Task-1 ($D = 80$).

	Regular-1	Regular-2	Cursive
$A = 2$, $B = 1$			
Baseline	96.52	94.84	88.74
Hanging	93.07	89.31	83.15
Our Method	94.02	91.25	83.21
$A = 4$, $B = 4$			
Baseline	97.16	94.75	89.49
Hanging	95.04	90.91	84.98
Our Method	95.63	92.39	85.32

observations can be made as in Table 2. For the baseline system, although a larger vocabulary is used, the recognition accuracy is only slightly decreased compared with those in Table 2. The baseline system with $A = 2$ and $B = 1$ gives a good tradeoff between accuracy and footprint for recognizing samples without rotational distortions. As for the rotation-free systems, our method consistently outperforms "hanging character" method, especially on Regular-2 test set. Our system with $A = 4$ and $B = 4$ is a pretty good rotation-free recognizer. From the Top-5 and Top-10 results, the above systems can deliver a quite compelling user experience already.

4. CONCLUSION

From the above experimental results, we conclude that the techniques presented in this paper can be readily used to construct a large vocabulary online handwritten Chinese character recognizer which has a small footprint, runs efficiently on a Smartphone, achieves high recognition accuracy, yet be robust to rotational distortions, therefore can be deployed in products. To the best of our knowledge, this represents the first study reporting experimental results of a recognizer which can recognize 27,533 Chinese characters in GB18030 standard.

5. REFERENCES

[1] Z.-L Bai and Q. Huo, "A study on the use of 8-directional features for online handwritten Chinese character recognition," *Proc. ICDAR-2005*, pp.262-266.

[2] Z.-D. Feng and Q. Huo, "Confidence guided progressive search and fast match techniques for high performance Chinese/English OCR," *Proc. ICPR-2002*, pp.III-89-92.

Table 3. Performance ("Top-N" recognition accuracies in %) and footprint comparison of baseline system without preprocessing and two rotation-free systems on three test sets on Task-2 ($D = 80$).

		Regular-1	Regular-2	Cursive
$A = 2$, $B = 1$, Footprint: 3.34 MB				
Baseline	Top-1	96.36	94.38	88.16
	Top-5	99.44	99.30	97.03
	Top-10	99.64	99.65	98.18
$A = 4$, $B = 4$, Footprint: 8.80 MB				
Baseline	Top-1	96.92	94.71	89.21
	Top-5	99.67	99.49	97.78
	Top-10	99.84	99.75	98.75
Hanging	Top-1	94.57	88.97	84.69
	Top-5	98.91	97.43	95.31
	Top-10	99.34	98.30	97.03
Our Method	Top-1	95.35	92.23	84.86
	Top-5	99.10	98.70	95.38
	Top-10	99.51	99.30	97.22

[3] GB18030: The New Chinese Encoding Standard, http://www.gb18030.com/

[4] GB/T18790-2010: Requirements and Test Procedure of Online Handwriting Chinese Character Recognition System.

[5] T. He and Q. Huo, "A study of a new misclassification measure for minimum classification error training of prototype-based pattern classifiers," *Proc. ICPR-2008*.

[6] T. He and Q. Huo, "A character-structure-guided approach to estimating possible orientations of a rotated isolated online handwritten Chinese character," *Proc. ICDAR-2009*, pp.536-540.

[7] S. Huang, *A Study On Recognition For Rotated Isolated Online Handwritten Chinese Character*, Master Thesis (in Chinese and supervised by Professor Lianwen Jin), South China University of Technology, China, 2010.

[8] C. Igel and M. Hüsken, "Improving the Rprop learning algorithm," *Proc. 2nd Int. Symp. on Neural Computation*, 2000, pp.115-121.

[9] T. Long and L.-W. Jin, "A novel orientation free method for online unconstrained cursive handwritten Chinese word recognition," *Proc. ICPR-2008*.

[10] M. Nakagawa and M. Onuma, "On-line handwritten Japanese text recognition free from constraints on line direction and character orientation," *Proc. ICDAR-2003*, pp.519-523.

[11] M. Nakagawa and M. Onuma, "A model of on-line handwritten Japanese text recognition free from line direction and writing format constraints," *IEICE Trans. on Information and Systems*, Vol. E88-E, pp.1815-1822, 2005.

[12] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: the RPROP algorithm," *Proc. ICNN-1993*, pp.586-591.

[13] Y.-Q. Wang and Q. Huo, "A study of designing compact recognizers of handwritten Chinese characters using multiple-prototype based classifiers," *Proc. ICPR-2010*, pp.1872-1875.

[14] Z.-J. Yan, T. Gao, and Q. Huo, "Designing an MPI-based parallel and distributed machine learning platform on large-scale HPC clusters," submitted to *IWSML-2012*.