A Study of Designing Compact Recognizers of Handwritten Chinese Characters Using Multiple-Prototype Based Classifiers *

Yongqiang Wang and Qiang Huo
Microsoft Research Asia, Beijing, China
(Emails: yqwang@cs.hku.hk, qianghuo@microsoft.com)

Abstract

We present a study of designing compact recognizers of handwritten Chinese characters using multiple-prototype based classifiers. A modified Quickprop algorithm is proposed to optimize a sample-separation-margin based minimum classification error objective function. Split vector quantization technique is used to compress classifier parameters. Benchmark results are reported for classifiers with different footprints trained from about 10 million samples on a recognition task with a vocabulary of 9282 character classes which include 9119 Chinese characters, 62 alphanumeric characters, 101 punctuation marks and symbols.

1. Introduction

In the past decade, it has been demonstrated by several research groups that multiple-prototype (MP) based classifiers trained with minimum classification error (MCE) criteria [8] and compressed with split vector quantization (VQ) technique [5] can offer a practical solution to recognizing both printed (e.g., [7]) and handwritten (e.g., [4]) Chinese characters with a good tradeoff among three design factors, namely recognition accuracy, footprint, and recognition time. Recently, a new sample-separation-margin (SSM) based MCE criterion was proposed for training MP-based classifiers in [6]. Comparative experiments were conducted for classifiers trained with 3 different MCE criteria from about 700K samples on a handwriting recognition task with a vocabulary of 2965 Japanese Kanji characters. Experimental results demonstrate that SSM-based MCE training achieves significant character recognition error rate reduction compared with two traditional MCE training criteria. In order to verify whether this conclusion is true for other tasks as well, in this study new experiments are conducted on a recognition task with a much larger vocabulary and much more training and testing samples. To improve the throughput of experiments and take advantage of the computational capability offered by the cluster computing infrastructure we can access, a batch-mode Quickprop algorithm [2] is adopted for MCE training while in [6], a sequential gradient descent algorithm (a.k.a. generalized probabilistic descent, GPD [8]) was used. It is observed that MCE-trained classifiers with a traditional Quickprop implementation as described in [10] achieve slightly worse recognition accuracy than that of GPD. We therefore proposed a modified Quickprop algorithm which can bridge the above performance gap. Furthermore, split VQ technique is used to compress classifier parameters so that compact recognizers with different footprints can be constructed. The purpose of this paper is to report research findings and benchmark results we obtained in the above study.

The rest of this paper is organized as follows. A brief introduction of MP-based classifier and different MCE training criteria are presented in Section 2. The modified Quickprop algorithm is described in Section 3. The classifier compression procedure is given in Section 4. Experimental results are reported in Section 5. Finally, the paper is concluded in Section 6.

2. Classifiers and MCE Training Criteria

Suppose our classifier is expected to distinguish M character classes denoted as $\{C_i|i = 1, \cdots, M\}$. We are given a set of training feature vectors $\mathcal{X} = \{\boldsymbol{x}_r \in \mathcal{R}^D|r = 1, \cdots, R\}$ together with their labels $\mathcal{I} = \{i_r|r = 1, \cdots, R\}$. For MP-based classifiers, each class C_i is represented by K_i prototypes, $\boldsymbol{\lambda}_i = \{\boldsymbol{m}_{ik} \in \mathcal{R}^D|k = 1, \cdots, K_i\}$, where $\boldsymbol{m}_{ik}$ is the kth prototype of the ith class. We use $\boldsymbol{\Lambda} = \{\boldsymbol{\lambda}_i\}$ to denote the set of prototypes. In classification stage, a feature vector $\boldsymbol{x} \in \mathcal{R}^D$ is first extracted. Then $\boldsymbol{x}$ is compared with

*This work has been done when the first author was an intern at Microsoft Research Asia, Beijing, China.

1051-4651/10 $26.00 © 2010 IEEE
DOI 10.1109/ICPR.2010.1138

each of the M class models and a discriminant function using Euclidean distance as the dissimilarity measure is computed for each class C_i as follows:

$$g_i(\boldsymbol{x};\boldsymbol{\lambda}_i) = -\min_k \|\boldsymbol{x}-\boldsymbol{m}_{ik}\|^2 . \quad (1)$$

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

$$r(\boldsymbol{x};\boldsymbol{\Lambda}) = \arg\max_i g_i(\boldsymbol{x};\boldsymbol{\lambda}_i) . \quad (2)$$

In training stage, classifier parameters $\boldsymbol{\Lambda}$ can be estimated by minimizing the following MCE objective function [8]:

$$\ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}) = \frac{1}{R}\sum_{r=1}^{R}\frac{1}{1+\exp[-\alpha d(\boldsymbol{x}_r,i_r;\boldsymbol{\Lambda})+\beta]} \quad (3)$$

where $d(\boldsymbol{x}_r,i_r;\boldsymbol{\Lambda})$ is a so-called misclassification measure, and α,β are two control parameters. As discussed in [6], any of the following three misclassification measures can be plugged into Eq. (3) to form a specific MCE criterion:

$$d(\boldsymbol{x}_r,i_r;\boldsymbol{\Lambda}) = -g_{i_r}(\boldsymbol{x}_r;\boldsymbol{\lambda}_{i_r}) + g_q(\boldsymbol{x}_r;\boldsymbol{\lambda}_q) \quad (4)$$

$$d(\boldsymbol{x}_r,i_r;\boldsymbol{\Lambda}) = \frac{g_{i_r}(\boldsymbol{x}_r;\boldsymbol{\lambda}_{i_r}) - g_q(\boldsymbol{x}_r;\boldsymbol{\lambda}_q)}{g_{i_r}(\boldsymbol{x}_r;\boldsymbol{\lambda}_{i_r}) + g_q(\boldsymbol{x}_r;\boldsymbol{\lambda}_q)} \quad (5)$$

$$d(\boldsymbol{x}_r,i_r;\boldsymbol{\Lambda}) = \frac{-g_{i_r}(\boldsymbol{x}_r;\boldsymbol{\lambda}_{i_r}) + g_q(\boldsymbol{x}_r;\boldsymbol{\lambda}_q)}{\|\boldsymbol{m}_{i_r\hat{k}} - \boldsymbol{m}_{q\overline{k}}\|} \quad (6)$$

where

$$\begin{aligned}\hat{k} &= \arg\min_k \|\boldsymbol{x}_r - \boldsymbol{m}_{i_rk}\|^2,\\ q &= \arg\max_{i\in\mathcal{M}_r} g_i(\boldsymbol{x}_r;\boldsymbol{\lambda}_i),\\ \overline{k} &= \arg\min_k \|\boldsymbol{x}_r - \boldsymbol{m}_{qk}\|^2 ,\end{aligned}$$

and $\mathcal{M}_r$ is the hypothesis space for the rth sample, excluding the true label i_r. Following the convention in [6], the resultant MCE criteria are referred to as MCE1, MCE2, and MCE3, respectively.

3. Optimization Procedure

In this study, we propose to use the following modified Quickprop procedure to optimize the objective function in Eq. (3):

Step 1: Let $t = 1$. Calculate the derivative of $\ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda})$ w.r.t. each m_{ikd} and update it by

$$m_{ikd}^{(t+1)} = m_{ikd}^{(t)} - \varepsilon_0 \frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}} ,$$

where m_{ikd} is the d-th element of $\boldsymbol{m}_{ik}$, $\frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}} \triangleq \frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda})}{\partial m_{ikd}}\big|_{\boldsymbol{\Lambda}=\boldsymbol{\Lambda}^{(t)}}$, and ε_0 is an initial learning rate set empirically.

Step 2: Let $t \leftarrow t+1$. Calculate the approximate second derivative of $\ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda})$ w.r.t. each m_{ikd} as follows:

$$\frac{\partial^2 \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}^2} \approx \frac{\frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}} - \frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t-1)})}{\partial m_{ikd}}}{m_{ikd}^{(t)} - m_{ikd}^{(t-1)}} . \quad (7)$$

Step 3: Calculate update step differently depending on the following cases:

- If $\frac{\partial^2 \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}^2} > 0$ and the sign of gradient $\frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}}$ differs from that of $\frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t-1)})}{\partial m_{ikd}}$, then the following Newton step is used:

$$\delta_t m_{ikd} = -\frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}} \Big/ \frac{\partial^2 \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}^2} , \quad (8)$$

where $\delta_t m_{ikd}$ denotes the update step of m_{ikd}.

- If $\frac{\partial^2 \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}^2} > 0$ and $\frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}}$ and $\frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t-1)})}{\partial m_{ikd}}$ have the same sign, the following modified Newton step is used:

$$\delta_t m_{ikd} = -\left(1/\frac{\partial^2 \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}^2} + \varepsilon_t\right)\frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}} \quad (9)$$

with ε_t being a learning rate set as $\varepsilon_t = \varepsilon_0(1 - t/T_Q)$, where T_Q is the total number of iterations to be performed.

- If $\frac{\partial^2 \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}^2} < 0$ or the magnitude of $\delta_t m_{ikd}$ is too small, backoff to gradient descent by setting the update step as follows:

$$\delta_t m_{ikd} = -\varepsilon_t \frac{\partial \ell(\mathcal{X},\mathcal{I};\boldsymbol{\Lambda}^{(t)})}{\partial m_{ikd}} . \quad (10)$$

Step 4: If $|\delta_t m_{ikd}| > \text{limit} \times |\delta_{t-1} m_{ikd}|$, set $\delta_t m_{ikd} = \text{sign}(\delta_t m_{ikd}) \times \text{limit} \times |\delta_{t-1} m_{ikd}|$ to limit the absolute update step size, where limit is a control parameter and set as 1.75 in our experiments.

Step 5: Update m_{ikd} by $m_{ikd}^{(t+1)} \leftarrow m_{ikd}^{(t)} + \delta_t m_{ikd}$.

Step 6: Repeat Step 2 to Step 5 $T_Q - 1$ times.

Step 7: Repeat Step 1 to Step 6 $T_R - 1$ times.

Due to space limitation, the formulas of relevant derivative calculation are omitted. Because the above procedure works in batch mode, it can be easily parallelized, for example, by using multiple computers to calculate the derivative in Step 1. Our procedure is similar to the

Quickprop implementation described in [10] except for adding Step 7 whose rationale will be explained in the following.

Quickprop approximates the Hessian matrix by assuming it is a diagonal matrix, i.e.,

$$\boldsymbol{H} = \frac{\partial \ell^2(\mathcal{X}, \mathcal{I}; \boldsymbol{\Lambda})}{\partial \boldsymbol{\Lambda} \partial \boldsymbol{\Lambda}^T} \approx \mathrm{diag}(\boldsymbol{H}) \,, \qquad (11)$$

where $\mathrm{diag}(\boldsymbol{H})$ means setting all the off-diagonal elements in $\boldsymbol{H}$ to zero. Therefore, when $\boldsymbol{\Lambda}^{(t)}$ is approaching to the minimum point, the step size calculated by Eq. (8) is quite small, compared with the step size computed by Newton's method whose convergence rate is quadratic near the minimum point, since

$$\|\delta \boldsymbol{\Lambda}\| = \|\mathrm{diag}(\boldsymbol{H})\|^{-1} \|\boldsymbol{g}\| \leq \|\boldsymbol{H}\|^{-1} \|\boldsymbol{g}\| \,,$$

where $\boldsymbol{g} = \frac{\partial \ell(\mathcal{X}, \mathcal{I}; \boldsymbol{\Lambda})}{\partial \boldsymbol{\Lambda}}$. The inequality holds because for semi-positive definite matrix $\boldsymbol{H}$, $\|\mathrm{diag}(\boldsymbol{H})\| \geq \|\boldsymbol{H}\|$ (Hadamard's inequality). The underestimation problem becomes more severe when the dimension of $\boldsymbol{H}$, which equals to the number of parameters in $\boldsymbol{\Lambda}$, is quite large (In our case there are about 1 millon parameters in $\boldsymbol{\Lambda}$). In fact, we observed that the element of Hessian matrix estimated by Eq. (7) is often unreasonably large, therefore making the updating step size calculated by Eq. (8) almost zero. Moreover, for some m_{ikd}'s, once the updating size is nearly zero, the subsequent iterations will not affect m_{ikd} either because of the limits of updating step size imposed in Step 4. Our modification may be the simplest method to alleviate this problem, which is to periodically restart the procedure from gradient descent, i.e., back to Step 1 every T_Q traditional Quickprop iterations. How to solve this problem more elegantly is still an open question, which will be a topic for future research.

4. Compression of Classifier Parameters

As in [7, 4], we use the split VQ technique [5] to compress the MP-based classifier parameters as follows. Each prototype $\boldsymbol{m}_{ik} \in \mathcal{R}^D$ is first split into S sub-vectors, where the dimension of the sth sub-vector, $\boldsymbol{m}_{ik}^{(s)}$, is D_s, i.e., $D = \sum_{s=1}^{S} D_s$. Then for each set of sub-vectors $\{\boldsymbol{m}_{ik}^{(s)} | i = 1, \cdots, M; k = 1, \cdots, K_i\}$, the LBG clustering algorithm [9] is used to generate a codebook with 256 codewords using Euclidean distance as the distortion measure. Each codeword is a D_s-dimensional vector, whose elements are represented by D_s 4-byte floating-point numbers, while the index for each codeword is a single-byte unsigned integer which explains why the codebook size is 256. In total, it takes $S \times \sum_{i=1}^{M} K_i$ bytes to store the indices and $4 \times D \times 256$ bytes to store the codebooks.

Table 1. Comparison of recognition accuracies (in %) on "JPN" task of MP-based classifiers trained by using MCE3 criterion with different optimization methods: GPD vs. Quickprop.

Optimization Methods	$K_i = 2$	$K_i = 4$
GPD	98.36	98.35
Quickprop ($T_R = 1$)	98.20	98.22
Quickprop ($T_R = 2$)	98.25	98.27
Quickprop ($T_R = 3$)	98.31	98.31

5. Experiments and Results

5.1 Experiments on a Medium-Scale Task

To debug the code of new Quickprop implementation, the first set of experiments are conducted on a handwriting recognition task with a vocabulary of 2965 Japanese Kanji characters. The experimental setup is exactly the same as in [6]. This task is referred to as "JPN" hereinafter. Two optimization methods, GPD and Quickprop, are compared using MCE3 criterion. Starting from the LBG-trained prototype parameters, GPD and Quickprop algorithms are used to optimize classifier parameters $\boldsymbol{\Lambda}$. The control parameters for Quickprop experiments are set as follows: $\alpha = 7$, $\beta = 0$, $\varepsilon_0 = 0.1$, and $T_Q = 20$. Results are summarized in Table 1. It is observed that periodically restarting the Quickprop procedure from gradient descent improves the recognition accuracy. After three cycles ($T_R = 3$), the recognition accuracy achieved by Quickprop is comparable to that achieved by GPD. Since most of the computation is spent on the calculation of the loss function and its gradient, which can be parallelized for Quickprop, the Quickprop experiments can be completed in much shorter time than its GPD counterparts: for example, the training time can be reduced from days for GPD run on a single CPU to hours for Quickprop run on 40 CPUs, namely a speedup of 20 to 30 times in this case.

5.2 Experiments on a Large-Scale Task

The remaining experiments are conducted on a much larger scale task of recognizing isolated online handwritten characters. The recognition vocabulary consists of 9282 character classes which include 9119 Chinese characters, 62 alphanumeric characters, 101 punctuation marks and symbols. An in-house developed corpus is used. The training set contains 9,795,394 character samples. The testing set contains 614,369 handwriting samples, which is further divided into two subsets according to the writing style (cursive or regular) of each

Table 2. Comparison of recognition accuracies (in %) on two different test sets of "CHS" task of MP-based classifiers trained by three different MCE criteria.

MCE Criterion	Regular	Cursive
MCE1	97.8	92.4
MCE2	97.8	92.1
MCE3	97.9	92.7

testing sample as follows: 1) **Cursive**: 431,546 samples from 3,863 character classes; 2) **Regular**: 182,823 samples from 6,763 character classes. It is observed that there are much more regular-style training samples than the cursive ones. We therefore selectively duplicated some cursive-style training samples to avoid the possible bias of the MCE-trained classifiers to work well only for regular-style samples. After this *resampling*, the total number of training samples is about 12 millon. This task is referred to as "CHS" hereinafter. As for feature extraction, a 512-dimensional raw feature vector $\boldsymbol{y}$ is first extracted from each handwriting sample by using the procedure described in [1]. Then we use an LDA transformation matrix estimated from the training data to transform $\boldsymbol{y}$ into a new 128-dimensional feature vector $\boldsymbol{x}$. The number of prototypes K_i is set as 2 for 3,755 most frequently used Chinese characters and 1 for the rest of character classes.

In the second set of experiments, we use the proposed modified Quickprop procedure to optimize three different objective functions, namely MCE1, MCE2 and MCE3 respectively. The control parameters are set as follows: α is 7, 20 and 1.5 for MCE1, MCE2 and MCE3 respectively; β is 0; T_Q is 20 and T_R is 3; $\varepsilon_0 = 0.1$. Experimental results are summarized in Table 2. It is observed that MCE3 criterion performs consistently better than the other two criteria.

Finally, in the third set of experiments, we take the MCE3-trained MP model in the second set of experiments to do model compression as described in Section 4. Table 3 compares the accuracy-footprint tradeoff of 4 classifiers, where "128×1" and "64×2" mean that the sub-vector dimension in split VQ is 1 and 2 respectively, "$32 \times 2 + 16 \times 4$" means that the sub-vector dimension is 2 for the first 64 dimensions and 4 for the remaining 64 dimensions. After compression, only a minor drop in recognition accuracies is observed, but footprints of the corresponding classifiers can be reduced dramatically.

6. Conclusion

It is clear from the above experimental results that MP-based classifiers trained by using the SSM-based MCE criterion (i.e., MCE3) with the proposed modified Quickprop procedure can achieve promising recognition accuracies, while such classifiers can be made very compact by using split VQ compression. Combined with the fast-match technique as described in [3], an efficient and compact handwriting recognizer can be constructed. Such a recognizer can be deployed alone or combined further with other compact classifiers to offer a good handwriting recognition solution on mobile devices with limited memory for East Asian languages such as Chinese, Japanese, and Korean.

Table 3. Comparison of recognition accuracies (in %, on two different test sets of "CHS" task) and footprint (in MB) of several MP-based classifiers.

Setup	Regular	Cursive	Footprint
No Compression	97.9	92.7	6.37
128×1	98.0	92.4	1.72
64×1	97.7	92.2	0.92
$32 \times 2 + 16 \times 4$	97.6	92.0	0.72

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] S. E. Fahlman, "An empirical study of learning speed in back-propragation networks," Technical Report CMU-CS-88-162, Carnegine Mellon University, 1988.

[3] 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.

[4] Y. Ge and Q. Huo, "A comparative study of several modeling approaches for large vocabulary offline recognition of handwritten Chinese characters," *Proc. ICPR-2002*, pp.III-85-88.

[5] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression,* Springer, 1991.

[6] 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*.

[7] Q. Huo, Y. Ge and Z.-D. Feng, "High performance Chinese OCR based on Gabor features, discriminative feature extraction and model training," *Proc. ICASSP-2001*, pp.III-1517-1520.

[8] B.-H. Juang and S. Katagiri, "Discriminative learning for minimum error classification," *IEEE Trans. on Signal Processing*, vol. 40, no. 12, pp.3043-3054, 1992.

[9] Y. Linde, A. Buzo, R. Gray, "An algorithm for vector quantizer design", *IEEE Trans. on Communications,* vol. 28, pp. 84-94, 1980.

[10] E. McDermott, T. J. Hazen, J. Le Roux, A. Nakamura and S. Katagiri, "Discriminative training for large vocabulary speech recognition using minimum classification error," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 15, no. 1, pp. 203-223, 2007.