An irrelevant variability normalization approach to discriminative training of multi-prototype based classifiers and its applications for online handwritten Chinese character recognition

Jun Du a,*, Qiang Huo b

a National Engineering Laboratory for Speech and Language Information Processing (NEL-SLIP), University of Science and Technology of China, No. 96, JinZhai Road, Hefei, Anhui PR China
b Microsoft Research Asia, 13/F, Building 2, No. 5 Danling Street, Haidian District, Beijing, PR China

A R T I C L E   I N F O

Article history:
Received 4 January 2014
Received in revised form 19 May 2014
Accepted 17 June 2014
Available online 24 June 2014

Keywords:
Irrelevant variability normalization
Sample separation margin
Minimum classification error
Rprop
Discriminative training
Online handwritten Chinese character recognition

A B S T R A C T

This paper presents an irrelevant variability normalization (IVN) approach to jointly discriminative training of feature transforms and multi-prototype based classifier for recognition of online handwritten Chinese characters. A sample separation margin based minimum classification error criterion is adopted in IVN-based training, while an Rprop algorithm is used for optimizing the objective function. For the IVN approach based on piecewise linear transforms, the corresponding recognizer can be made both compact and efficient by using a two-level fast-match tree whose internal nodes coincide with the labels of feature transforms. Furthermore, the IVN system using weighted sum of linear transforms outperforms that based on piecewise linear transforms. The effectiveness of the proposed approach is first confirmed using an in-house developed online Chinese handwriting corpus with a vocabulary of 9306 characters, and then further verified on a standard benchmark database for an online handwritten character recognition task with a vocabulary of 3755 characters.

© 2014 Elsevier Ltd. All rights reserved.

1. Introduction

Using online handwritten Chinese character recognition as an input mode on a portable device has been becoming increasingly popular. Good solutions have been developed to build product engines for online handwritten Chinese character recognition (e.g., [1–3]). In spite of many successful applications, however, the problem of diversified training data and/or possible mismatch between training and testing conditions has not been addressed explicitly in the above solutions. Also the results of the recent Chinese handwriting recognition competition [4,5] reveal the challenge of both isolated character recognition and handwriting text recognition. In this study, we adopt a so-called irrelevant variability normalization (IVN) [6] based training strategy to tackle the above problem. IVN is a general concept for pattern recognition problems, which was first proposed in speech recognition area [6]. In a specific pattern recognition problem, there are many irrelevant variabilities in the contents to be recognized, which lead to the degradation of recognition performance in real applications. For example, in speech recognition area, those variabilities could be speaker variability (or accent), background noises, etc. The writing style in online handwriting recognition, the font style, lighting condition, background noises, perspective distortions in the image for optical character recognition (OCR) can be also considered as irrelevant variabilities. The main idea of IVN is to normalize all those variabilities explicitly or implicitly either in the feature space or model space for both training stage and recognition stage.

In [7], a so-called speaker adaptive training (SAT) approach was proposed to normalize speaker variability in training hidden Markov models (HMMs) for automatic speech recognition (ASR). The concept of SAT training was generalized to deal with any variabilities irrelevant to phonetic classification in [6], therefore a term of IVN training was coined, where as an illustrative example, the IVN training was used to improve learning HMM state tying from data based on phonetic decision-tree. Since then, many variants of IVN training methods have been tried in ASR area. For example, IVN-based training of feature transforms and HMMs based on maximum likelihood [8] and discriminative training [9] has been verified to be effective for large vocabulary continuous speech recognition (LVCSR). A region-dependent feature transform (RDT) approach proposed in [10,11], which was a weighted sum of linear transforms using the Gaussian posterior as the weight coefficient, was yet another example of IVN training. Only recently, the concept of IVN training was tried in the area of handwriting recognition. For example, in [12], writer adaptive training (WAT) using constrained maximum likelihood linear regression (CMLLR)
based feature transform was studied for an HMM-based Arabic handwriting recognition task. RDT-based approach in [10] was also applied to HMM-based off-line handwriting recognition in [14]. More recently, a pattern field classification approach with style normalized transformation was proposed in [15] and demonstrated to be effective for several pattern recognition applications, including handwritten Chinese character recognition.

In our recent work [16], we study the problem of IVN-based training for online handwritten Chinese character recognition. One of the state-of-the-art techniques to build a Chinese handwriting recognizer is to use a so-called sample separation margin (SSM) based minimum classification error (MCE) criterion [17,18], which is similar to the generalized learning vector quantization (GLVQ) approach in [35], to train a prototype-based classifier as reported in [1]. In spite of the large vocabulary of Chinese characters, such a classifier can be made both compact (e.g., [19]) and efficient (e.g., [20]) in the recognition stage. In [16], we propose an approach to IVN-based joint training of feature transforms and prototype-based classifier parameters by using the SSM-MCE criterion and demonstrate its effectiveness for Chinese handwriting recognition as an illustrative example. An Rprop algorithm ([21,22]) is used to optimize the objective function. Furthermore, the IVN-trained recognizer can be made both compact and efficient by using a two-level fast-match tree [20] whose internal nodes coincide with the labels of feature transforms. In this paper, we extend the above work in the following ways: (1) the weighted sum of linear transforms, similar to the region dependent function. Furthermore, the IVN-trained recognizer can be made both compact and efficient by using a so-called sampled separation margin (SSM) as follows [17]:

\[g_\text{IVN}(\mathbf{y}; \mathbf{Y}_i) = -\min_k \|\mathbf{y} - \mathbf{m}_k\|^2 \]

(1)

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

\[r(\mathbf{y}; \mathbf{A}) = \arg\max_i g(\mathbf{y}; \lambda_i). \]

(2)

In the training stage, given a set of training feature vectors \( \mathcal{Y} = \{\mathbf{y}_r \in \mathbb{R}^D | r = 1, \ldots, K\} \), first we initialize \( \mathbf{A} \) by LBG clustering [26]. Then \( \mathbf{A} \) can be re-estimated by minimizing the following SSM-MCE objective function:

\[l(\mathbf{y}; \mathbf{A}) = \frac{1}{K} \sum_{k=1}^{g} \frac{1}{1 + \exp(-ad(\mathbf{y}_r; \mathbf{A}) + \beta)} \]

(3)

where \( a, \beta \) are two control parameters, and \( d(\mathbf{y}_r; \mathbf{A}) \) is a misclassification measure defined by using a so-called sample separation margin (SSM) as follows [17]:

\[d(\mathbf{y}_r; \mathbf{A}) = \frac{\|\mathbf{y}_r - \mathbf{m}_k\|^2}{2\|\mathbf{m}_k - \mathbf{m}_{k'}\|^2} \]

(4)

where

\[\hat{k} = \arg\min_k \|\mathbf{y}_r - \mathbf{m}_k\|^2 \]

(5)

\[q = \arg\max_{i \in \lambda_i} g(\mathbf{y}_r; \lambda_i) \]

(6)

\[\mathbf{t}_k = \arg\min_k \|\mathbf{y}_r - \mathbf{m}_k\|^2 \]

(7)

and \( \lambda_i \) is the hypothesis space for the \( r \)th sample, excluding the true label \( p \).

To optimize the objective function in Eq. (3), the same implementation of Rprop algorithm as described in [1] is adopted here.

3. IVN-based SSM-MCE joint training

3.1. Feature transformation

In this study, the concept of IVN is implemented by using feature transformation. Two feature transforms are explored, namely piecewise linear transforms (PLT) and weighted sum of linear transforms (WSLT). For PLT based IVN training, the following feature transformation is used:

\[\mathbf{x}_i = \mathcal{F}_1(\mathbf{y}_r; \Theta) = \mathbf{A}_r \mathbf{y}_r + \mathbf{b}_r \]

(8)

where \( \mathbf{y}_r \) and \( \mathbf{x}_r \) are the \( D \)-dimensional input and transformed feature vectors, respectively; and \( \mathbf{e}_r \) is the transform label for the \( r \)th sample. Let’s use \( \mathcal{O} = (\mathbf{A}_r, \mathbf{b}_r) | r = 1, \ldots, E \) to denote the set of transform parameters of \( E \) linear transforms, where \( \mathbf{A}_r \) is a \( D \times D \) nonsingular matrix and \( \mathbf{b}_r \) is a \( D \)-dimensional bias vector. As for WSLT based IVN training, the corresponding transform is defined as

\[\mathbf{x}_i = \mathcal{F}_2(\mathbf{y}_r; \Theta) = \sum_{e=1}^{E} w_e^r (\mathbf{A}_e \mathbf{y}_r + \mathbf{b}_e) \]

(9)

with the constraints

\[\sum_{e=1}^{E} w^r_e = 1, \]

(10)

2. SSM-MCE training of a multi-prototype based classifier

Suppose our classifier can recognize \( M \) character classes denoted as \( \{C_i | i = 1, \ldots, M\} \). For a multi-prototype based classifier, each class \( C_i \) is represented by \( K_i \) prototypes, \( \lambda_i = \{\mathbf{m}_k \in \mathbb{R}^D | k = 1, \ldots, K_i\} \), where \( \mathbf{m}_k \) is the \( k \)th prototype of the \( i \)th class. Let’s use \( \Lambda = \{\lambda_i\} \) to denote the set of prototypes. In the classification stage, a feature vector \( \mathbf{y} \in \mathbb{R}^D \) is first extracted. Then \( \mathbf{y} \) is compared with each of the \( M \) classes by evaluating a Euclidean distance based discriminant function for each class \( C_i \) as follows:

\[g_i(\mathbf{y}; \lambda_i) = -\min_k \|\mathbf{y} - \mathbf{m}_k\|^2 \]

(11)

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

\[r(\mathbf{y}; \mathbf{A}) = \arg\max_i g(\mathbf{y}; \lambda_i). \]

(12)

In the training stage, given a set of training feature vectors \( \mathcal{Y} = \{\mathbf{y}_r \in \mathbb{R}^D | r = 1, \ldots, K\} \), first we initialize \( \mathbf{A} \) by LBG clustering [26]. Then \( \mathbf{A} \) can be re-estimated by minimizing the following SSM-MCE objective function:

\[l(\mathbf{y}; \mathbf{A}) = \frac{1}{K} \sum_{k=1}^{g} \frac{1}{1 + \exp(-ad(\mathbf{y}_r; \mathbf{A}) + \beta)} \]

(13)

where \( a, \beta \) are two control parameters, and \( d(\mathbf{y}_r; \mathbf{A}) \) is a misclassification measure defined by using a so-called sample separation margin (SSM) as follows [17]:

\[d(\mathbf{y}_r; \mathbf{A}) = \frac{\|\mathbf{y}_r - \mathbf{m}_k\|^2}{2\|\mathbf{m}_k - \mathbf{m}_{k'}\|^2} \]

(14)

where

\[\hat{k} = \arg\min_k \|\mathbf{y}_r - \mathbf{m}_k\|^2 \]

(15)

\[q = \arg\max_{i \in \lambda_i} g(\mathbf{y}_r; \lambda_i) \]

(16)

\[\mathbf{t}_k = \arg\min_k \|\mathbf{y}_r - \mathbf{m}_k\|^2 \]

(17)

and \( \lambda_i \) is the hypothesis space for the \( r \)th sample, excluding the true label \( p \).

To optimize the objective function in Eq. (3), the same implementation of Rprop algorithm as described in [1] is adopted here.
where the $w_r^e$ is the weight coefficient of the $e$th linear transform for the $r$th sample $y_r$.

Hopefully the transformed feature vector $x_r$ in Eq. (8) or Eq. (9) has less irrelevant information to the content to be recognized and finally results in a compact classifier. In the next subsection, we elaborate on how to determine the transform label $e_r$ in Eq. (8) and the weight coefficient $w_r^e$ in Eq. (9).

### 3.2. Cluster construction and selection

Suppose the feature space can be divided into $E$ clusters and each cluster $e$ is associated with one linear transform $(A_e, b_e)$, which is assumed to normalize the irrelevant variability of the feature vector belonging to this cluster. In this work, to construct the clusters, we divide all the prototypes of all classes into $E$ groups by using k-means clustering approach. The centroid of each cluster $c_e$ is calculated as the sample mean of the prototypes belonging to the cluster $e$. Then in both IVN-based training and recognition stage based on PLT, given the clusters and for each feature vector $y_r$, a transform label $e_r$ is assigned to the feature vector as the label of the cluster having the minimum Euclidean distance between the feature vector and the cluster centroid:

$$ e_r = \arg\min_e \| y_r - c_e \|^2. $$

For WSLT, given the above clusters, the weight coefficient $w_r^e$ in Eq. (9) can be calculated using the softmax function

$$ w_r^e = \frac{\exp \{-\tau \| y_r - c_e \|^2 \}}{\sum_{e=1}^{E} \exp \{-\tau \| y_r - c_e \|^2 \}} $$

and $\tau$ is estimated from the training set:

$$ \tau = \frac{\tau_0}{\sum_{r=1}^{R} \min_e \| y_r - c_e \|^2}. $$

where $\tau_0$ is a scaling factor. Another way to calculate $w_r^e$ can refer to [10], where the Gaussian posterior probability is used as the weight coefficient for the RDT approach. First a Gaussian mixture model (GMM) with $E$ mixture components can be built on top of the $E$ clusters by using Expectation-Maximization (EM) algorithm:

$$ p(y) = \sum_{e=1}^{E} \omega_e N(y; \mu_e, \Sigma_e) $$

where $\mu_e$, $\Sigma_e$, and $\omega_e$ are the mean vector, diagonal covariance matrix and mixture weight of the $e$th Gaussian component. Then $w_r^e$ is set as the Gaussian posterior probability:

$$ w_r^e = \frac{\omega_e N(y_r; \mu_e, \Sigma_e)}{\sum_{e=1}^{E} \omega_e N(y_r; \mu_e, \Sigma_e)}. $$

---

**Fig. 1.** Overall flow of system development.
3.3. Training procedure

The IVN-based SSM-MCE objective function is defined as follows:

\[
l(y; \Lambda, \Theta) = \frac{1}{R} \sum_{i=1}^{R} \frac{1}{1 + \exp[-\alpha d(y_i; \Lambda, \Theta) + \beta]} \tag{16}
\]

where

\[
d(y_i; \Lambda, \Theta) = -\frac{g_p(x_i; \lambda_p) + g_q(x_i; \lambda_q)}{2 \cdot ||\Theta_i - \Theta_j||} \tag{17}
\]

In the above equations, \( x_i \) is defined in Eq. (8) or Eq. (9), where the corresponding feature transforms can be determined as described in Section 3.2. The following method of alternating variables can then be used to jointly estimate \( \Theta \) and \( \Lambda \) by minimizing the above objective function:

Step 1 : Initialization
First, the classifier parameters \( \Lambda \) are initialized by using SSM-MCE training described in Section 2. The transform parameters \( \Theta \) are initialized as \( \Theta = 0 \) and \( \Lambda = 1 \).

Step 2 : Estimating the feature transform parameters \( \Theta \) by fixing the classifier parameters \( \Lambda \)
Given the fixed classifier parameters \( \Lambda \), the SSM-MCE objective function \( l(y; \Lambda, \Theta) \) can be optimized by using an Rprop algorithm with \( N_I \) iterations as described in Appendix A.

Step 3 : Estimating the classifier parameters \( \Lambda \) by fixing the feature transform parameters \( \Theta \)
Given the updated transform parameters \( \Theta \) obtained in Step 2, we first transform each training feature vector \( y \) by using Eq. (8) or Eq. (9). Then an Rprop algorithm with \( N_C \) iterations is performed as described in Appendix A to re-estimate classifier parameters \( \Lambda \) by minimizing the objective function \( l(y; \Lambda, \Theta) \).

Step 4 : Repeat Step 2 and Step 3 \( N_{IVN} \) times.
In the above training procedure, the control parameters \( N_I \), \( N_C \), and \( N_{IVN} \) are set empirically.

4. Fast match technique

Our fast-match technique is based on a two-level tree [20]. To construct the tree, \( G \) clusters are first generated as described in Section 3.2. Each cluster has a bucket consisting of character classes with their prototypes belonging to the cluster. Each training feature vector will then be classified into the cluster with the minimum Euclidean distance between the feature vector and the cluster centroid. The character class of the training sample will then be used to jointly estimate \( \Theta \) and \( \Lambda \) by minimizing the above objective function:

- Compare the input feature vector with each cluster centroid and sort the result in ascending order of the Euclidean distances, which can be considered as the first-level recognition;
- If PLT based IVN training is performed, the feature transform associated with the first cluster is applied to the input feature vector; Otherwise, skip this step;
- Merge all character classes in the top \( N_b \) buckets and use them to perform the second-level recognition as usual.

In the above procedure, a technique known as the partial based elimination has been used to speed up the process of identifying the “Top \( N \)” candidates.

5. Experiments and results

5.1. Experimental setup

The experiments are first conducted on an in-house corpus for the task of recognizing isolated online handwritten characters with a vocabulary of 9306 character classes including 9143 Chinese characters, 62 alphanumeric characters, 101 punctuation marks and symbols. As shown in Table 1, this vocabulary mainly consists of 3755 GB2312-L1 characters, 3008 GB2312-L2 characters, 2380 CJK characters, and other 163 characters, which is used in the product engine for Chinese handwriting recognition of Microsoft. For training, we used about 1000 samples averaged per character class. Also from Table 1, the regular-style training samples are collected for all characters while the cursive-style training samples are only covered for GB2312-L1 and GB2312-L2. Because there are much more regular-style samples than cursive ones, even for GB2312-L1 and GB2312-L2, the re-sampling of training samples is performed as in [18]. We enlarge the proportion of cursive-style samples by using one more duplicate for the most commonly used GB2312-L1 characters, which is listed in Table 1. Three testing sets are used for evaluation: (1) Regular-1: 97,221 samples from 6903 character classes which are written in regular style; (2) Regular-2: 84,549 samples from 2355 uncommon character classes in regular style; (3) Cursive: 383,064 samples from 3755 frequently used character classes written in cursive style. For feature extraction, a 512-dimensional raw feature vector is extracted as described in [27], which is followed by LDA (linear discriminant analysis) transformation to obtain a lower dimensional feature vector. 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 and IVN-based SSM-MCE joint training, the control parameters are set as described in [1] and [28,29]. Other control parameters are set as: \( D = 80, E = G = 128, N_I = 10, N_C = 10, N_{IVN} = 5, \tau_0 = 20 \).

To evaluate on a standard benchmark, we also verify our approach on the public database released by the Institute of Automation of Chinese Academy of Sciences (CASIA) [30]. The feature datasets for evaluating isolated online handwritten Chinese character recognition are used which can be downloaded via [31]. The detailed information of the datasets can be found in Table 2. By combining OLHDB21.0 and OLHDB21.1 datasets, there are totally 2,154,582 samples in the training set and 538,601 samples in the testing set. The raw feature of the online handwritten character sample is a 512-dimensional vector using the 8-direction histogram of original trajectory direction combined with pseudo 2D bi-moment normalization [32,30]. Then each feature vector is transformed by Box-Cox transformation [33], followed by LDA transformation to obtain a 160-dimensional feature vector. To perform Rprop-based SSM-MCE training and IVN-based SSM-MCE joint training, only the parameter \( \alpha \) should be set to 1 due to

<table>
<thead>
<tr>
<th>Vocabulary</th>
<th>GB2312-L1</th>
<th>GB2312-L2</th>
<th>CJK</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>Writing style</td>
<td>Regular</td>
<td>Cursive</td>
<td>Regular</td>
<td>Cursive</td>
</tr>
<tr>
<td># of character</td>
<td>3755</td>
<td>3755</td>
<td>3008</td>
<td>3008</td>
</tr>
<tr>
<td># of duplicates</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
to the dynamic range of the new feature. All the other parameters are the same as those in the experiments on in-house corpus.

To handle large-scale training data, the tools for LBG clustering, SSM-MCE training, and IVN-based SSM-MCE joint training with the Rprop algorithm have been implemented based upon MSR Asia’s MPI-based machine learning platform [34]. 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.

5.2. Experimental results on in-house Corpus

Table 3 summarizes a performance (recognition accuracies in %) comparison of baseline systems and IVN-trained systems using piecewise linear transforms on three testing sets under different settings of the number of prototypes and the number of buckets searched in fast-match tree. The footprint (in MB) and runtime latency (normalized by Baseline(2,1) without fast-match) of the corresponding recognizers are also compared.

![Table 3](https://via.placeholder.com/150)

<table>
<thead>
<tr>
<th>Method(A,B)</th>
<th>Nb</th>
<th>Regular-1</th>
<th>Regular-2</th>
<th>Cursive</th>
<th>Footprint</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline(2,1)</td>
<td>N/A</td>
<td>96.53</td>
<td>94.84</td>
<td>91.53</td>
<td>4.10</td>
<td>1</td>
</tr>
<tr>
<td>Baseline(4,2)</td>
<td>N/A</td>
<td>96.88</td>
<td>94.93</td>
<td>92.88</td>
<td>8.08</td>
<td>1.92</td>
</tr>
<tr>
<td>Baseline(8,4)</td>
<td>N/A</td>
<td>97.09</td>
<td>94.66</td>
<td>92.74</td>
<td>16.00</td>
<td>3.66</td>
</tr>
<tr>
<td>IVN(2,1)</td>
<td>N/A</td>
<td>96.89</td>
<td>95.32</td>
<td>92.18</td>
<td>7.30</td>
<td>1.06</td>
</tr>
<tr>
<td>IVN(4,2)</td>
<td>N/A</td>
<td>97.19</td>
<td>95.54</td>
<td>92.88</td>
<td>11.28</td>
<td>1.99</td>
</tr>
<tr>
<td>IVN(8,4)</td>
<td>N/A</td>
<td>97.38</td>
<td>95.32</td>
<td>93.23</td>
<td>19.2</td>
<td>3.75</td>
</tr>
<tr>
<td>Baseline(2,1)</td>
<td>5</td>
<td>96.52</td>
<td>94.84</td>
<td>91.53</td>
<td>4.44</td>
<td>0.52</td>
</tr>
<tr>
<td>Baseline(4,2)</td>
<td>5</td>
<td>96.88</td>
<td>94.92</td>
<td>92.28</td>
<td>8.41</td>
<td>0.89</td>
</tr>
<tr>
<td>Baseline(8,4)</td>
<td>5</td>
<td>97.09</td>
<td>94.65</td>
<td>92.74</td>
<td>16.33</td>
<td>1.58</td>
</tr>
<tr>
<td>IVN(2,1)</td>
<td>5</td>
<td>96.89</td>
<td>95.32</td>
<td>92.19</td>
<td>7.60</td>
<td>0.53</td>
</tr>
<tr>
<td>IVN(4,2)</td>
<td>5</td>
<td>97.19</td>
<td>95.53</td>
<td>92.88</td>
<td>11.57</td>
<td>0.90</td>
</tr>
<tr>
<td>IVN(8,4)</td>
<td>5</td>
<td>97.39</td>
<td>95.31</td>
<td>93.23</td>
<td>19.49</td>
<td>1.59</td>
</tr>
<tr>
<td>Baseline(2,1)</td>
<td>3</td>
<td>96.52</td>
<td>94.80</td>
<td>91.52</td>
<td>4.44</td>
<td>0.37</td>
</tr>
<tr>
<td>Baseline(4,2)</td>
<td>3</td>
<td>96.88</td>
<td>94.89</td>
<td>92.28</td>
<td>8.41</td>
<td>0.65</td>
</tr>
<tr>
<td>Baseline(8,4)</td>
<td>3</td>
<td>97.09</td>
<td>94.62</td>
<td>92.74</td>
<td>16.33</td>
<td>1.17</td>
</tr>
<tr>
<td>IVN(2,1)</td>
<td>3</td>
<td>96.89</td>
<td>95.29</td>
<td>91.78</td>
<td>7.60</td>
<td>0.38</td>
</tr>
<tr>
<td>IVN(4,2)</td>
<td>3</td>
<td>97.19</td>
<td>95.48</td>
<td>92.87</td>
<td>11.57</td>
<td>0.66</td>
</tr>
<tr>
<td>IVN(8,4)</td>
<td>3</td>
<td>97.39</td>
<td>95.27</td>
<td>93.22</td>
<td>19.49</td>
<td>1.18</td>
</tr>
</tbody>
</table>

5.3. Experimental results on CASIA database

Table 5 shows a performance (recognition accuracies in %) comparison of two prototype-based systems under different settings of the number of prototypes on the training and testing sets. “LBC” denotes a system trained using LBG clustering while “SSM-MCE” refers to a system trained by the SSM-MCE criterion. SSM-MCE systems consistently and significantly outperform LBG systems on both training and testing sets with different numbers of prototypes.

![Table 4](https://via.placeholder.com/150)

<table>
<thead>
<tr>
<th>Method(A,B)</th>
<th>Top-N</th>
<th>Regular-1</th>
<th>Regular-2</th>
<th>Cursive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline(2,1)</td>
<td>96.52</td>
<td>94.84</td>
<td>91.53</td>
<td>4.10</td>
</tr>
<tr>
<td>Baseline(4,2)</td>
<td>96.88</td>
<td>94.93</td>
<td>92.88</td>
<td>8.08</td>
</tr>
<tr>
<td>Baseline(8,4)</td>
<td>97.09</td>
<td>94.66</td>
<td>92.74</td>
<td>16.00</td>
</tr>
<tr>
<td>IVN(2,1)</td>
<td>96.89</td>
<td>95.32</td>
<td>92.18</td>
<td>7.30</td>
</tr>
<tr>
<td>IVN(4,2)</td>
<td>97.19</td>
<td>95.54</td>
<td>92.88</td>
<td>11.28</td>
</tr>
<tr>
<td>IVN(8,4)</td>
<td>97.38</td>
<td>95.32</td>
<td>93.23</td>
<td>19.2</td>
</tr>
</tbody>
</table>

The footprint (in MB) and runtime latency (normalized by Baseline(2,1) without fast-match) of the corresponding recognizers are also compared. Here footprint is the size of the resources for the recognition engine including the classifier and the corresponding transformations while the runtime latency refers to the run time of the recognizer averaged on each character sample, which is a normalized version by the system denoted as Baseline(2,1) without fast-match. “Baseline” refers to an SSM-MCE trained system without IVN training while “IVN” refers to a system using our proposed IVN-based joint training. The second column of Table 3 indicates the top Nb buckets selected for second-level recognition in fast-match tree, where “N/A” means no fast-match is used. Three prototype configurations, namely (2,1), (4,2), and (8,4) are listed for comparison, because over-training would be observed if the number of prototypes was increased beyond (8,4) in current experiments. The runtime latency in the last column only includes the recognition time after feature extraction.

Based on those results, several observations can be made. First, IVN systems can achieve consistently significant improvements in recognition accuracy compared with the corresponding Baseline systems on all testing sets. Second, under the same prototype setting, the increased runtime latency from Baseline to IVN systems can be almost ignored, especially in cases with fast-match because the first-level recognition of fast-match tree and the cluster selection for each testing feature vector are shared completely. Third, IVN systems can still outperform Baseline systems with smaller footprints and less runtime latency, e.g., IVN(2,1) vs. Baseline(4,2) and IVN(4,2) vs. Baseline(8,4). Finally, with fast-match technique, the runtime latency of IVN systems can be reduced significantly while the footprint is only increased slightly. The tradeoff between recognition accuracy and efficiency can be made by setting different Nb. Compared with systems without using fast-match technique, Nb=5 keeps the same recognition accuracy with reduced runtime latency while Nb=3 degrades slightly recognition accuracy with a much more significant reduction of runtime latency.

Table 4 compares the “Top-N” recognition accuracies of Baseline systems and IVN-trained systems using piecewise linear transforms on three testing sets under different settings of the number of prototypes and a single setting of Nb=5 for fast match. From the Top-5 and Top-10 results, IVN systems can achieve very high recognition accuracies already.
of recognition accuracy over the LBG systems on the training set is more obvious than that on the testing set. To verify that our prototype-based classifier is state-of-the-art, we compare our results with those reported in [30]. The readers can refer to Table 8 in [30] where the classifier is discriminatively trained followed by discriminative feature extraction using a global linear transform. With the increased number of transforms, the recognition accuracies on the training set increase monotonically while the number of transforms achieving the best performance on the testing set for different number of prototypes is 128, which is set as default for the following experiments. Further increasing the number of transforms beyond 128 leads to over-training problem.

Table 7 summarizes a performance (“Top-N” recognition accuracies in %) comparison of baseline systems and different IVN-trained systems on the testing set under different settings of the number of prototypes. “Baseline” refers to the SSM-MCE trained system without IVN training. “IVN-PLT” denotes the system using the proposed IVN-based joint training via piecewise linear transforms. “IVN-WSLT-1” and “IVN-WSLT-2” are the IVN-trained systems using weighted sum of linear transforms where the weight is calculated. First, all the IVN-trained systems can yield significant performance improvements over the corresponding baseline systems on the testing set for different prototype setting. Second, compared with the performance on the in-house corpus in Table 3, IVN-PLT systems achieve similarly relative improvements of recognition accuracy on CASIA database with different feature type and dimension. Third, both IVN-WSLT-1 and IVN-WSLT-2 systems consistently outperform IVN-PLT systems which indicates the “soft” selection of linear transforms in WSLT is more powerful than the “hard” selection of linear transforms in PLT. And IVN-WSLT-1 systems achieve the best performance among all the IVN approaches. Finally, IVN-WSLT-2 systems are inferior to IVN-WSLT-1 systems which may be due to the different distance measures where Euclidean distance is used for our classifier design and weight calculation in IVN-WSLT-1 system while Mahalanobis distance is adopted for weight (Gaussian posterior) calculation in IVN-WSLT-2 system.

6. Conclusion

In this paper, we have proposed an approach to IVN-based SSM-MCE joint training of feature transforms and a prototype-based classifier and demonstrated its effectiveness for online handwritten Chinese character recognition on two large vocabulary tasks as an illustrative example. The IVN-trained recognizer using piecewise linear transforms can be made both compact and efficient by using a two-level fast-match tree whose internal nodes coincide with the labels of feature transforms. Given the consistent improvement of recognition accuracy compared with the corresponding SSM-MCE trained systems without using IVN training, even in the case of smaller footprint and runtime latency, the proposed IVN approach in this study offers a good product solution to construct a handwritten character recognizer to be deployed on mobile devices with limited memory for East Asian languages such as Chinese, Japanese, and Korean.

**Conflict of Interest**

None declared.
Appendix A. Rprop optimization procedure for classifier parameters

Step 1 : Let \( t=0 \). Calculate the derivative of \( l(\mathbf{y}; \mathbf{A}, \Theta) \) w.r.t. each \( a_{kd} \) and update the prototype parameters as follows:

\[
m_{kd}^{(t+1)} = m_{kd}^{(t)} + \Delta m_{kd}^{(t)}
\]

\[
\Delta m_{kd}^{(t)} = -\text{sign}\left( \frac{\partial l(\mathbf{y}; \mathbf{A}, \Theta)}{\partial a_{kd}} \right) \frac{\partial l(\mathbf{y}; \mathbf{A}, \Theta)}{\partial m_{kd}}
\]

where \( m_{kd} \) is the \( d \)th element of \( m_{kd} \), \( m_{kd}^{(t)} = m_{kd}, \Delta m_{kd}^{(t)} = \Delta_0 \), and

\[
\frac{\partial l(\mathbf{y}; \mathbf{A}, \Theta)}{\partial m_{kd}} = \frac{\partial \partial \mathbf{y}; \mathbf{A}, \Theta}{\partial m_{kd}^c} |_{A = \Lambda^t}.
\]

Step 2 : Let \( t = t+1 \). Define

\[
S = \frac{\partial l(\mathbf{y}; \mathbf{A}^{(t-1)}, \Theta)}{\partial m_{kd}}\bigg|_{A = \Lambda^t} \frac{\partial l(\mathbf{y}; \mathbf{A}^{(t)}, \Theta)}{\partial m_{kd}}
\]

Then, the updating formulas are

\[
\Delta_{kd}^{(t)} = \begin{cases} 
\min(\eta^+, \Delta_{kd}^{(t-1)}, \Delta_{\text{max}}) & \text{if } S > 0 \\
\max(\eta^-, \Delta_{kd}^{(t-1)}, \Delta_{\text{min}}) & \text{if } S < 0 \\
\Delta_{kd}^{(t-1)} & \text{else}
\end{cases}
\]

If \( S < 0 \),

\[
m_{kd}^{(t+1)} = m_{kd}^{(t)} + \Delta m_{kd}^{(t)}
\]

Step 3 : Repeat Step 2 \((N_\ell - 1)\) times.

In the above procedure, the relevant derivative can be calculated as follows:

\[
\frac{\partial l_r}{\partial m_{kd}} = l_r(1 - l_r) \left[ \frac{\delta(i, p) \delta(k, \hat{k})(m_{p,k} - m_{q,k}) - \delta(i, q) \delta(k, \hat{k})(m_{p,q} - m_{q,k})}{\| m_{p,k} - m_{q,k} \|^2} \right]
\]

\[
- d(\mathbf{y}; \mathbf{A}, \Theta) \left[ \frac{\delta(i, p) \delta(k, \hat{k}) - \delta(i, q) \delta(k, \hat{k})}{\| m_{p,k} - m_{q,k} \|^2} \right]
\]

where

\[
l_r = \frac{1}{1 + \exp(-\theta(\mathbf{y}; \mathbf{A}, \Theta) + \beta)}
\]

and \( \delta \) is Kronecker delta function.

Appendix B. Rprop optimization procedure for transform parameters

Step 1 : Let \( t=0 \). Calculate the derivative of \( l(\mathbf{y}; \mathbf{X}, \Theta) \) w.r.t. each \( A_{\text{edj}} \) and \( b_{\text{edj}} \), where \( A_{\text{edj}} \) is the \((d,j)\)th element of the matrix \( A_{kd} \) and \( b_{\text{edj}} \) is the \( d \)th element of the bias vector \( \mathbf{b} \). Then update the transform parameters as follows:

\[
A_{\text{edj}}^{(t+1)} = A_{\text{edj}}^{(t)} + \Delta A_{\text{edj}}^{(t)}
\]

\[
\Delta A_{\text{edj}}^{(t)} = -\text{sign}\left( \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta)}{\partial A_{\text{edj}}} \right) \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta)}{\partial A_{\text{edj}}}
\]

\[
b_{\text{edj}}^{(t+1)} = b_{\text{edj}}^{(t)} + \Delta b_{\text{edj}}^{(t)}
\]

\[
\Delta b_{\text{edj}}^{(t)} = -\text{sign}\left( \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta)}{\partial b_{\text{edj}}} \right) \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta)}{\partial b_{\text{edj}}}
\]

where \( b_{\text{edj}}^{(t)} = 0, \Delta b_{\text{edj}}^{(t)} = \Delta_0 \), and

\[
\frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta)}{\partial A_{\text{edj}}} = \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta)}{\partial A_{\text{edj}}} |_{\Theta = \theta^{(t)}}
\]

\[
\frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta)}{\partial b_{\text{edj}}} = \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta)}{\partial b_{\text{edj}}} |_{\Theta = \theta^{(t)}}
\]

Step 2 : Let \( t = t+1 \). Define

\[
S_A = \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta^{(t-1)})}{\partial A_{\text{edj}}} \bigg|_{\Theta = \theta^{(t)}} \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta^{(t)})}{\partial A_{\text{edj}}}
\]

\[
S_B = \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta^{(t-1)})}{\partial b_{\text{edj}}} \bigg|_{\Theta = \theta^{(t)}} \frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta^{(t)})}{\partial b_{\text{edj}}}
\]

Then, the updating formulas are

\[
\Delta_{\text{edj}}^{(t)} = \begin{cases} 
\min(\eta^+ \Delta_{\text{edj}}^{(t-1)}, \Delta_{\text{max}}) & \text{if } S_A > 0 \\
\max(\eta^- \Delta_{\text{edj}}^{(t-1)}, \Delta_{\text{min}}) & \text{if } S_A < 0 \\
\Delta_{\text{edj}}^{(t-1)} & \text{else}
\end{cases}
\]

\[
\Delta_{\text{edj}}^{(t)} = \begin{cases} 
\min(\eta^+ \Delta_{\text{edj}}^{(t-1)}, \Delta_{\text{max}}) & \text{if } S_B > 0 \\
\max(\eta^- \Delta_{\text{edj}}^{(t-1)}, \Delta_{\text{min}}) & \text{if } S_B < 0 \\
\Delta_{\text{edj}}^{(t-1)} & \text{else}
\end{cases}
\]

If \( S_A < 0 \),

\[
\frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta^{(t)})}{\partial A_{\text{edj}}} = 0
\]

If \( S_B < 0 \),

\[
\frac{\partial l(\mathbf{y}; \mathbf{X}, \Theta^{(t)})}{\partial b_{\text{edj}}} = 0
\]

\[
A_{\text{edj}}^{(t+1)} = A_{\text{edj}}^{(t)} + \Delta A_{\text{edj}}^{(t)}
\]

\[
b_{\text{edj}}^{(t+1)} = b_{\text{edj}}^{(t)} + \Delta b_{\text{edj}}^{(t)}
\]

Step 3 : Repeat Step 2 \((N_\ell - 1)\) times.

In the above procedure, the relevant derivatives can be calculated. For the case of piecewise linear transforms, the formulations are

\[
\frac{\partial l_r}{\partial A_{\text{edj}}} = \frac{a_{l_r}(1 - l_r)(m_{p,k} - m_{q,k}) \delta(e, i) y_{ji}}{\| m_{p,k} - m_{q,k} \|^2}
\]

where

\[
l_r = \frac{1}{1 + \exp(-\theta(\mathbf{y}; \mathbf{A}, \Theta) + \beta)}
\]
where the case of weighted sum of linear transforms, the corresponding derivatives are

\[ \frac{\partial L}{\partial b_{cd}} = \frac{aL(1 - p)(m_{q, y} - m_{p, k})\alpha(e, e)}{\|m_{p, k} - m_{q, y}\|} \]  

(42)

while for the case of weighted sum of linear transforms, the corresponding derivatives are

\[ \frac{\partial L}{\partial A_{cdj}} = \frac{aL(1 - p)(m_{q, y} - m_{p, k}(w_{x, y}))}{\|m_{p, k} - m_{q, y}\|} \]  

(43)

\[ \frac{\partial L}{\partial b_{d}} = \frac{aL(1 - p)(m_{q, y} - m_{p, k})w_{x, y}}{\|m_{g, y} - m_{q, y}\|} \]  

(44)

where

\[ l = \frac{1}{1 + \exp(-\alpha d_{q, y}; \hat{X}, \Theta) + \beta} \]  

(45)

References


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

Qiang Huo (M’95) is a Senior Researcher and Manager in Microsoft Research Asia (MSRA), Beijing, China. Prior to joining MSRA on August 1, 2007, he had been a faculty member at the Department of Computer Science, The University of Hong Kong since 1998, where he also did his Ph.D. research on speech recognition during 1991 to 1994. From 1995 to 1997, Dr. Huo worked at the ATR Interpreting Telecommunications Research Laboratories, Kyoto, Japan. In the past 25 years, he has been doing research and making contributions in the areas of speech recognition, handwriting recognition, OCR, gesture recognition, biometric-based user authentication, hardware design for speech and image processing. Dr. Huo received the B.Eng. degree from the University of Science and Technology of China (USTC), Hefei, China, in 1987, the M.Eng. degree from Zhejiang University, Hangzhou, China, in 1989, and the Ph.D. degree from the USTC in 1994, all in electrical engineering.