Modeling Inverse Covariance Matrices By Expansion Of Tied Basis Matrices For Online Handwritten Chinese Character Recognition *

Yongqiang Wang^{1,2} and Qiang Huo¹ ¹Microsoft Research Asia, Beijing, China ²Department of Computer Science, The University of Hong Kong, Hong Kong, China (Emails: yqwang@cs.hku.hk, qianghuo@microsoft.com)

Abstract

The state-of-the-art modified quadratic discriminant function (MQDF) based approach for online handwritten Chinese character recognition (HCCR) assumes that the feature vectors of each character class can be modeled by a Gaussian distribution with a mean vector and a full covariance matrix. In order to achieve a high recognition accuracy, enough number of leading eigenvectors of the covariance matrix have to be retained in MQDF. This paper presents a new approach to modeling each inverse covariance matrix by basis expansion, where expansion coefficients are character-dependent while a common set of basis matrices are shared by all the character classes. Consequently, our approach can achieve a much better accuracy-memory tradeoff. The usefulness of the proposed approach to designing compact HCCR systems has been confirmed and demonstrated by comparative experiments on popular Nakayosi and Kuchibue Japanese character databases.

Keywords: online handwriting recognition, pattern classification, covariance modeling, MQDF.

1. Introduction

By now, it has been confirmed by several research groups that a state-of-the-art performance can be achieved for online handwritten Chinese character recognition (HCCR) by using so-called 8-directional features [3, 4, 5] and modified quadratic discriminant function (MQDF) [11] to construct a character classifier. Although some online HCCR products for mobile phones (e.g., [7]) have used multiple-prototype-based (MP-based) classifier (e.g. [10]) due to its light memory requirement [8], researchers have been exploring different ways of reducing the memory requirement of MQDF-based classifiers, hoping to deploy it in handheld devices with limited memory. A recent interesting experimental study was reported in [13] which was a straight-froward application of the relevant model compression techniques originally described in [10, 8, 9]. In this paper, we explore another possibility for designing compact HCCR systems based on the concept of *structured covariance modeling*.

In the past several years, much progress has been made to improve covariance modeling for Gaussian-mixture continuous-density hidden Markov model (CDHMM) based automatic speech recognition (ASR). For example, in [16, 17], a so-called extended maximum likelihood linear transformation (EMLLT) model was proposed, where the inverse covariance (a.k.a *precision*) matrix of each Gaussian is constrained to be in a subspace of the space of symmetric matrices spanned by a set of rank-one matrices shared by all the Gaussians. In [1], a precision constrained Gaussian (PCG) model was proposed as an extension of EMLLT model by relaxing the basis from symmetric rankone matrices to symmetric full-rank matrices, although a term called SPAM (subspace precision and mean) model was used to refer to this modeling technique. An independent work of PCG model was also reported in [19]. More general SPAM models, where separate constraints can be imposed on the precisions and means, were developed later ([2] and references therein). All of the above approaches can be viewed as special cases of a general model called subspace constrained Gaussian (SCG) mixture model as discussed in [2]. A similar attempt was also made in [18] to discuss the above ideas in a more general perspective. Encouraged by the promising results for different ASR applications, in this paper, we adopt one of the above mentioned modeling techniques, namely PCG model and study its effectiveness for HCCR with a hope of identifying a good approach to designing a compact Chinese handwriting recognizer.

The rest of the paper is organized as follows. In section 2, we briefly describe our baseline MQDF-based online HCCR system. In section 3, we present our proposed PCG model (referred to as PCGM hereinafter) and its maximum likelihood (ML) training procedure for online HCCR. In section 4, we evaluate and compare the perfor-

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

mance of our PCGM approach with that of the state-ofthe-art MQDF-based approach. The paper is concluded in section 5 with a brief discussion on future works.

2. MQDF-Based Baseline System

Given a handwriting sample, our baseline system first extracts a *D*-dimensional raw feature vector *z* using the procedures described in [3, 4, 5]. To reduce the computational complexity and storage requirement, *z* will be transformed into a lower dimensional feature space using a $D \times d$ transformation matrix *W*, i.e., $x = W^T z$. This transformation matrix *W* can be obtained by linear discriminant analysis (LDA) (e.g., [10]).

Let's use $\{C_j | j = 1, \dots, M\}$ to denote the set of M character classes, and $\mathcal{X} = \{x_{ji} | j = 1, \dots, M; i = 1, \dots, n_j\}$ to denote the set of training feature vectors. Assume that feature vectors from the same character class C_j can be modeled by a Gaussian distribution with a mean vector μ_j and a full covariance matrix Σ_j . By setting the (k + 1)-th to d-th eigenvalues of Σ_j as a class-dependent constant δ_j , a so-called modified quadratic discriminant function (MQDF) can be defined as follows [11]:

$$g_{j}(x;\Theta_{j}) \triangleq -\frac{1}{2} \{ \sum_{l=1}^{k} \log \rho_{jl} + (d-k) \log \delta_{j} + (\frac{1}{\rho_{jl}} - \frac{1}{\delta_{j}}) p_{jl}^{2} + \frac{1}{\delta_{j}} d_{E}^{2}(x,\mu_{j}) \} (1)$$

where $\Theta_j = \{\mu_j, \{\rho_{jl}\}_{l=1}^k, \{v_{jl}\}_{l=1}^k, \delta_j\}, \rho_{jl}$ is the *l*-th leading eigenvalue, v_{jl} is the corresponding eigenvector of Σ_j , $p_{jl} = (x - \mu_j)^T v_{jl}$, $d_E(x, \mu_j)$ is the Euclidean distance between μ_j and x, k and δ_j are two control parameters. In practice, setting δ_j as the average of (k+1)-th to *d*-th eigenvalues works well.

Although other alternatives exist (e.g., [12]), we used the penalized (or regularized) ML training for estimating MQDF parameters Θ_j : μ_j is simply taken as the sample mean $\overline{\mu}_j$ of the training feature vectors from class C_j , while Σ_j is estimated as follows (e.g., [12]):

$$\widehat{\Sigma}_{j} = (1 - \gamma)\overline{\Sigma}_{j} + \gamma \frac{1}{d} \operatorname{tr}(\overline{\Sigma}_{j})I$$
(2)

where $\overline{\Sigma}_j$ is the sample covariance matrix of the training feature vectors from class C_j , I is an identity matrix, and γ is a control parameter which can be optimized via crossvalidation by using a development set.

In recognition stage, an unknown feature vector x will be classified as the class with the maximum discriminant function value as follows:

$$x \in C_j$$
 if $j = \arg\max_l g_l(x; \Theta_l)$. (3)

This is known as the *maximum discriminant decision rule* for pattern recognition.

3. PCG Model and ML Training Procedure

3.1 Precision Constrained Gaussian Model

In our precision constrained Gaussian (PCG) model, we assume that feature vectors of each character class C_j follow a Gaussian distribution, i.e., $p(x|C_j) = \mathcal{N}(x; \mu_j, \Sigma_j)$, where mean μ_j has no constraint imposed, while precision matrix $P_j = \Sigma_j^{-1}$ lies in a K-dimensional subspace spanned by a set of basis matrices $\Psi = \{S_k | k = 1, \dots, K\}$ which are shared by all the character classes. Consequently, the precision matrix P_j can be written as

$$P_j \stackrel{\triangle}{=} \sum_{k=1}^{K} \lambda_k^j S_k \tag{4}$$

where λ_k^j 's are class-dependent basis coefficients and Kis a control parameter [1]. It is noted that the basis matrices S_k 's are symmetric and not required to be positive definite, but P_j 's are required to be symmetric and positive definite. Therefore, the set of PCG model parameters, $\Theta = \{\Theta_{tied}, \Theta_{untied}\}$, consists of a subset of tied parameters $\Theta_{tied} = \Psi$ and a subset of untied parameters $\Theta_{untied} = \{\mu_j, \Lambda_j; j = 1, \dots, M\}$, where $\Lambda_j = (\lambda_1^j, \dots, \lambda_K^j)^T$. The total number of parameters of our PCG models is Kd(d+1)/2 + M(K+d), which is much smaller than that of MQDF models, i.e., M(k+1)(d+1), if K is small compared with both M and d(d+1).

In recognition stage, the following log likelihood function for unknown feature vector x is used as discriminant function

$$g_j(x;\Theta) \stackrel{\scriptscriptstyle \Delta}{=} \frac{1}{2} \log \det(\frac{P_j}{2\pi}) - \frac{1}{2} (x - \mu_j)^T P_j(x - \mu_j) \ . \tag{5}$$

The same *maximum discriminant decision rule* as in Eq. (3) can then be used for character classification. The computational complexity can be reduced if we evaluate the R.H.S. of Eq. (5) as follows:

$$g_j(x;\Theta) = b_j + x^T l_j + \sum_{k=1}^K \lambda_k^j f_k$$

where

$$b_j = \log \det(\frac{P_j}{2\pi}) - \frac{1}{2}\mu_j^T P_j \mu_j ,$$

$$l_j = P_j \mu_j$$

which can be pre-computed and cached, and the "quadratic feature" $f_k = -\frac{1}{2}x^T S_k x$ only need be computed once for each feature vector x because it can be shared for all Gaussians.

In the following subsection, we describe in detail an ML training procedure for estimating PCG model parameters from training data. Although we do benefit and borrow certain ideas from previously published works, the overall training procedure is different from any of the ML training procedures reported in literature.

3.2. ML Training Procedure

Given the set of training samples \mathcal{X} , the objective function of ML training is defined as the following log likelihood function of the PCG model parameters Θ :

$$\mathcal{L}(\Theta|\mathcal{X}) = \sum_{j=1}^{M} \sum_{i=1}^{n_j} \log p(x_{ji}|C_j, \Theta)$$
$$= \sum_{j=1}^{M} n_j \{\log \det(P_j) - \operatorname{tr}(\overline{\Sigma}_j P_j) - (\mu_j - \overline{\mu}_j)^T P_j(\mu_j - \overline{\mu}_j)\}.$$
(6)

The ML training problem then becomes the following constrained optimization problem:

$$\Theta^* = \arg \max_{\Theta} \mathcal{L}(\Theta | \mathcal{X})$$
(7)

piect to $\forall i$, $\sum_{k=1}^{K} \lambda_{i}^{j} S_{k} \succeq 0$.

subject to $\forall j$, $\sum_{k=1} \lambda'_k S_k \succ 0$.

It is easy to derive that the optimal μ_j^* is just the sample mean $\overline{\mu}_j$, and the rest of parameters can be optimized by solving the following new optimization problem:

$$(\Psi^*, \Lambda^*) = \arg \max_{P_j \succ 0} \mathcal{L}(\Lambda; \Psi)$$
(8)

where

$$\mathcal{L}(\Lambda; \Psi) = \sum_{j=1}^{M} n_j [\log \det(P_j) - \operatorname{tr}(\overline{\Sigma}_j P_j)] \qquad (9)$$

and $\Lambda = \{\Lambda_j; j = 1, \cdots, M\}.$

An overall ML training procedure to solve the above problem is summarized in **Algorithm 1**, and the details of three main components of the algorithm are described in the following three subsections, respectively.

3.2.1. Initialization

To ensure that our overall procedure works, we need to first specify initial basis matrices $\Psi^{(0)}$ and basis coefficients $\Lambda^{(0)}$ such that every precision matrix is positive definite. The following procedure, which is a slightly modified version of the relevant approach described originally in [19], works well in practice:

Step 1: Normalize covariance matrices

First each sample covariance matrix $\overline{\Sigma}_j$ is normalized as follows:

$$\Phi_j = (\det \overline{\Sigma}_j)^{\frac{1}{d}} (\overline{\Sigma}_j)^{-1}$$

It is noted that in [19], $\overline{\Sigma}_j$ is normalized as $\Phi_j = (\det \overline{\Sigma}_j)(\overline{\Sigma}_j)^{-1}$, which does not work well in our HCCR application here.

Algorithm 1: Overall ML Training Procedure

Input:

A set of training samples \mathcal{X} .

Output:

 $\{\mu_j, \lambda_k^j, S_k\}$ which optimize the objective function in Eq. (6).

Step 1: Initialization

Estimate $\{\mu_j\}$;

Initialize basis matrices Ψ and basis coefficients Λ (section 3.2.1).

Step 2: Alternate Optimization of Λ and Ψ

for $t = 0, \cdots, T$ do

Optimization for basis coefficients Λ (section 3.2.2):

$$\Lambda^{(t+1)} = \arg\max_{P_j \succ 0} \mathcal{L}(\Lambda; \Psi^{(t)}); \qquad (10)$$

Optimization for basis matrices Ψ (section 3.2.3):

$$\Psi^{(t+1)} = \arg \max_{P_j \succ 0} \mathcal{L}(\Lambda^{(t+1)}; \Psi) .$$
 (11)

Step 3: Output Parameters

Step 2: *Initialize basis matrices* Ψ

- Arbitrarily choose K symmetric matrices from $\{\Phi_j\}$ as the initial basis matrices S_1, \dots, S_k .
- Assign each Φ_j to a cluster with $S_{c(j)}$ as its centroid:

$$c(j) = \arg\min_{k} d(S_k, \Phi_j)$$

where

$$d(S_k, \Phi_j) = \operatorname{tr}(\Phi_j^{-1}S_k) + \operatorname{tr}(\Phi_j S_k^{-1})$$

is used as the distortion function.

- Re-calculate the centroid for each cluster as the average of all the Φ_j 's allocated to this cluster. Consequently, each S_k is positive definite.
- Repeat the above two sub-steps until some stopping criteria are satisfied.

Step 3: Initialize basis coefficients Λ

Each basis coefficient is initialized as follows:

$$\lambda_k^j \leftarrow \operatorname{tr}(\Phi_j S_k)$$

Because both Φ_j and S_k are positive definite, $\lambda_k^j > 0$. Consequently, every precision matrix P_j is symmetric and positive definite.

3.2.2. **Optimizing Untied Parameters** Λ

For the optimization problem in Eq. (10), once the set of basis matrices Ψ is fixed, different sets of basis coefficients Λ_j for different character classes are independent. Therefore, the original optimization problem can be further divided into M sub-problems, each amounts to finding an optimal Λ_i^* to maximize the following objective function:

$$\mathcal{L}_j(\Lambda_j) = \log \det(P_j) - \operatorname{tr}(\overline{\Sigma}_j P_j)$$
(12)

while maintaining the positive definiteness of P_j .

Because of the concavity of the function $\log \det(\cdot)$ and the linearity of the tr(\cdot) function, the Hessian of the above objective function $\mathcal{L}_j(\cdot)$ is always negative definite, provided P_i is positive definite [6]. We propose to use Newton's method with line search [15] to solve the above constrained optimization problem. Because the Hessian matrix of objective function is negative definite everywhere, our algorithm is guaranteed to converge to the global optimum Λ^* from any arbitrary initial $\Lambda^{(0)}$. The detailed procedure is described as follows:

Step 1: Calculate gradient and Hessian matrix

$$\nabla \mathcal{L}_j = (\operatorname{tr}(\Xi_j S_1), \cdots, \operatorname{tr}(\Xi_j S_K))^T \\ H_{pq} = -\operatorname{tr}(S_p P_j^{-1} S_q P_j^{-1})$$

,

where $\Xi_j = P_j^{-1} - \overline{\Sigma}_j$, H_{pq} is the (p, q)-th element of the Hessian matrix $H = \nabla^2 \mathcal{L}_j(\Lambda_j)$.

Step 2: Calculate search direction

Given the gradient and Hessian matrix at current position, the search direction for Λ_j is

$$\Delta \Lambda_j = -H^{-1} \nabla \mathcal{L}_j$$

where $\Delta \Lambda_j = (\Delta \lambda_1^j, \cdots, \Delta \lambda_K^j)^T$. Given this direction, the update direction of P_j can be obtained as

$$R_j = \sum_{k=1}^{K} \Delta \lambda_k^j S_k \; .$$

Step 3: Line search

Given the search direction $\Delta \Lambda_j$, a line search module is invoked to find an optimal step size α , such that

$$\phi_j(\alpha) = \mathcal{L}_j(\Lambda_j + \alpha \Delta \Lambda_j) - \mathcal{L}_j(\Lambda_j)$$

= $\log \frac{\det(P_j + \alpha R_j)}{\det P_j} - \alpha \operatorname{tr}(R_j \overline{\Sigma}_j)$

is maximized. We can evaluate efficiently the function $\phi_i(\alpha)$ and its first/second order derivative by using the fact

$$\log \frac{\det(P_j + \alpha R_j)}{\det P_j} = \sum_{p=1}^d \log(1 + \alpha w_p^j)$$

where w_p^j is the *p*-th eigenvalue of $P_i^{-\frac{1}{2}} R_j P_i^{-\frac{1}{2}}$. At the same time, the positive definiteness constraints can also be checked efficiently by using the following fact: if $1 + \alpha w_p^j > 0$ for all p and $P_j \succ 0, P_j + \alpha R_j$ will also be positive definite. It is also deserved to point out that $\phi_i''(\alpha) < 0$ for all $\alpha \in \operatorname{dom} \phi_j$, where $\operatorname{dom} \phi_j = [0, \alpha_{max})$, and

$$\alpha_{max} = \begin{cases} +\infty, & \text{if } w_p^j \ge 0 \text{ , for all } p \text{ ,} \\ \max_p \frac{-1}{w_p^j} & \text{otherwise }. \end{cases}$$

The procedure to optimize $\phi_i(\alpha)$ is as follows:

• If $\alpha_{max} = +\infty$, it can be shown that $\lim_{\alpha \to +\infty} \phi'_j(\alpha) < 0.$ Combining with the continuity of $\phi'_i(\alpha)$ and $\phi'_i(0) > 0$ (since the search direction is an ascent direction), we can deduce that there must exist an $\alpha^* \in (0, +\infty)$ such that $\phi'_i(\alpha^*) = 0$. Such an optimal α^* can be found as follows:

Step 3.a:
$$\alpha_0 \leftarrow 0, t \leftarrow 0$$
.

- **Step 3.b:** $\alpha_{t+1} \leftarrow \alpha_t \frac{\phi'_j(\alpha_t)}{\phi''_i(\alpha_t)}$. If $\alpha_{t+1} < 0$, arbitrarily choose α_{t+1} from $(0, \alpha_t)$.¹
- **Step 3.c:** $t \leftarrow t + 1$; goto step 3.b until $\|\phi'_i(\alpha_t)\| \leq \epsilon$ for some small $\epsilon > 0$.
- If α_{max} < $+\infty$, it can be shown that $\lim_{\alpha \to \alpha_{max}-} \phi_j'(\alpha) < 0.$ Following the same argument of the case where α_{max} is infinite, it can be seen that there exists an $\alpha^* \in$ $(0, \alpha_{max})$ such that $\phi'_i(\alpha) = 0$. Similarly, this optimal point can be found as follows:

Step 3.a:
$$\alpha_0 \leftarrow 0, t \leftarrow 0$$
.
Step 3.b: $\alpha_{t+1} \leftarrow \alpha_t - \frac{\phi'_j(\alpha_t)}{\phi''_j(\alpha_t)}$. If $\alpha_{t+1} < 0$,
arbitrarily choose α_{t+1} from $(0, \alpha_t)$. If
 $\alpha_{t+1} > \alpha_{max}^2$, arbitrarily choose α_{t+1}
from (α_t, α_{max}) .

Δ

Step 3.c: $t \leftarrow t + 1$; goto step 3.b until $\|\phi'_i(\alpha_t)\| \leq \epsilon$ for some small $\epsilon > 0$.

Step 4: Update untied parameters

$$\Lambda_i \leftarrow \Lambda_i + \alpha^* \Delta \Lambda_i$$

where α^* is the optimal α found by line search module.

Step 5: Repeat Steps 1 - 4 N_{untied} times.

¹This makes sense, since $\alpha_{t+1} < 0$ implies that $\phi'_i(\alpha_t) < 0$ (because $\alpha_t > 0$ and $\phi_i''(\alpha_t) < 0$). Combining with the fact $\phi_i'(0) > 0$, we can deduce that there exists an $\alpha^* \in (0, \alpha_t)$ such that $\phi'_i(\alpha^*) = 0$.

²This implies that $\phi'_i(\alpha_t) > 0$. Combing with the fact $\lim_{\alpha \to \alpha_{max}} \phi'_j(\alpha) < 0, \text{ there must exist an } \alpha^* \in (\alpha_t, \alpha_{max}) \text{ such}$ that $\phi'_i(\alpha^*) = 0.$

3.2.3. Optimizing Tied Parameters Ψ

Although other options exist (e.g., [19, 2]), we use Polak-Ribiere Conjugate Gradient (PR-CG) method [15] to solve the optimization problem in Eq. (11). The detailed procedure is described as follows:

Step 1: $t \leftarrow 0$.

Step 2: Calculate gradient

$$\begin{split} \mathcal{G}_t &\triangleq & ((\nabla_{S_1} \mathcal{L})^T, \cdots, (\nabla_{S_K} \mathcal{L})^T)^T \\ &= & \sum_{j=1}^M n_j (\lambda_1^j (\operatorname{vec} \Xi_j)^T, \cdots, \lambda_K^j (\operatorname{vec} \Xi_j)^T)^T \end{split}$$

where $\Xi_j = P_j^{-1} - \overline{\Sigma}_j$, vec is an operator on symmetric matrices defined as vector containing the elements of the upper triangular portion with the diagonal scaled by $\frac{1}{\sqrt{2}}$, i.e.,

$$\operatorname{vec}(X) = (\frac{X_{11}}{\sqrt{2}}, X_{12}, \frac{X_{22}}{\sqrt{2}}, X_{13}, \cdots, \frac{X_{dd}}{\sqrt{2}})^T$$
.

Step 3: Calculate search direction using PR-CG

Using Polak-Ribiere conjugate gradient method, an ascent search direction S_t can be found by using G_t and G_{t-1} , where S_t is defined as follows:

$$\mathcal{S}_t = ((\operatorname{vec}\Delta S_1)^T, \cdots, (\operatorname{vec}\Delta S_K)^T)^T,$$

and ΔS_k is the update direction of S_k .

Step 4: Line search

Using the similar strategy as we described in section 3.2.2, the line search module is invoked to find an optimal α^* , i.e,

$$\alpha_t^* = \arg \max_{P_j \succ 0} \mathcal{L}(\Lambda; \Psi_t + \alpha \Delta \Psi_t) - \mathcal{L}(\Lambda; \Psi_t) ,$$

where $\Delta \Psi_t$ is the update direction of Ψ_t .

Step 5: *Update tied parameters*

$$\Psi_{t+1} \leftarrow \Psi_t + \alpha_t^* \Delta \Psi_t \; .$$

Step 6: $t \leftarrow t+1$.

Step 7: Repeat Steps 2 - 6 N_{tied} times.

4. Experiments and Results

4.1. Experimental Setup

In order to evaluate the capability and limitation of the proposed PCG model for online HCCR, we conduct a series of experiments on the task of the recognition of isolated online handwritten characters with a vocabulary of 2965 level-1 Kanji characters in JIS standard. The popular Nakayosi and Kuchibue Japanese character databases [14] are used. The Nakayosi database consists of about 1.7 million character samples from 163 writers, and the Kuchibue database contains about 1.4 million character samples from 120 writers. We select randomly about 92% samples from the Nakayosi database to form the training data set, 75% samples from the Kuchibue database to form the testing data set, while the remaining samples from both databases are used to form a development set for tuning control parameters. By this partition, there are 704,650 samples in the training set, 229,398 in the development set, and 506,848 in the testing set, respectively.

As for feature extraction, a 512-dimensional raw feature vector z is first extracted from each handwriting sample by using the procedure described in [4]. Then we use the LDA transformation matrix W estimated from the training data to transform z into a new feature vector xof dimension 192 (e.g. [10]). All of our experiments are conducted on these 192-dimensional feature vectors. For MQDF-based classifier, control parameters k is chosen according to the available memory resource while γ is set to fine-tune the performance on development set. The performance of MQDF-based classifiers become saturated when k > 50. For PCG model, training basis coefficients using Newton's method with line search is quite effective: the objective function converges in 3 or 4 iterations. Optimizing basis matrices is relatively slow. In our implementation, we perform 20-50 iterations to optimize basis matrices. The number of overall training iterations (i.e. Tin step 2 of algorithm 1) is set to 10.

4.2. Experimental Results

Table 1 gives a comparison of recognition accuracies (in %) of PCGM and MQDF approaches along with the corresponding memory requirements for storing the respective model parameters. In estimating the "worstcase" memory requirement, we assume that a 4-byte floating point number is used to represent each model parameter and advanced model compression techniques such as those described in [10, 8, 9, 13] are not used. In Table 1, PCGM(K) means that the precision matrices are constrained to lie in a K-dimensional space (c.f. Eq. (4)) while MQDF(k) means that k leading eigenvalues/eigenvectors are used in Eq. (1).

Several observations can be made from Table 1: 1) When the memory resource is very stringent, the performance of MQDF-based classifier degraded severely (e.g., MQDF(1/2/3)), while PCGM achieves relatively high accuracy, say 98.19% using only 4.8MB memory. This fact suggests that PCGM could be very useful in designing compact handwriting recognizers; 2) Given gradually increased memory resource, the PCGM performance approaches quickly to the performance limit of MQDFbased classifier while consuming much less memory. This may suggest that most of the variability of precision matri**Table 1**. Comparison of recognition accuracies (in %) of PCGM and MQDF along with the corresponding memory requirements (in MB) for storing the respective model parameters.

Methods	Test Accuracy	Memory (MB)
PCGM(32)	98.19	4.80
PCGM(64)	98.46	7.41
PCGM(128)	98.65	12.66
PCGM(160)	98.68	15.29
MQDF(1)	97.17	4.37
MQDF(2)	97.52	6.55
MQDF(3)	97.84	8.73
MQDF(10)	98.47	24.01
MQDF(20)	98.62	45.84
MQDF(50)	98.71	111.33

ces can be captured by a set of well-trained basis matrices.

5. Conclusion and Future Work

In this paper, we present a new approach to designing compact handwritten Chinese character recognizers using PCG model. From the above experimental results and discussions, it is clear that PCGM-based approach offers great flexibilities in striking for a good tradeoff between the recognition accuracy and memory requirement, therefore could be a good candidate for designing a compact online HCCR system. Ongoing and future works include

- Re-do the above experiments on a much larger Chinese handwriting corpus with a much larger vocabulary;
- Study minimum classification error (MCE) training to further improve the performance of PCGM-based classifier;
- Compare the performance of "product-level" implementations of the most promising approaches with some known good "tricks" implemented.

It is our hope that a good solution can be identified for implementing a high-performance online Chinese handwriting recognizer on mobile platforms after the completion of the above studies. We will report the results elsewhere once they become available.

References

- S. Axelord, R. Gopinath, and P. Olsen, "Modeling with a subspace constraint on inverse covariance matrices," in *Proc. ICSLP-2002*, pp.2177-2180.
- [2] S. Axelord, V. Goel, R. A. Gopinath, P. A. Olsen, and K. Visweswariah, "Subspace constrained Gaussian mixture models for speech recognition," *IEEE Trans. on Speech* and Audio Processing, vol. 13, no. 6, pp.1144-1160, 2005.

- [3] Z.-L. Bai and Q. Huo, "A study on the use of 8-directional features for online handwritten Chinese character recognition," in *Proc. ICDAR-2005*, pp.262-266.
- [4] Z.-L. Bai and Q. Huo, "An improved approach to extracting 8-directional features for online handwritten Chinese character recognition," unpublished manuscript, 2005.
- [5] Z.-L. Bai and Q. Huo, "A study of nonlinear shape normalization for online handwritten Chinese character recognition: dot density vs. line density equalization," in *Proc. ICPR-2006*, pp.921-924.
- [6] S. P. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
- [7] Y. Ge, F.-J. Guo, L.-X. Zhen, and Q.-S. Chen, "Online Chinese character recognition system with handwritten Pinyin input," in *Proc. ICDAR-2005*, pp.1265-1269.
- [8] Y. Ge and Q. Huo, "A comparative study of several modeling approaches for large vocabulary offline recognition of handwritten Chinese characters," in *Proc. ICPR-2002*, pp.III-85-88.
- [9] Y. Ge and Q. Huo, "A study on the use of CDHMM for large vocabulary offline recognition of handwritten Chinese characters," in *Proc. 8th IWFHR (Eigth International Workshop on Frontiers in Handwriting Recognition)*, 2002, pp.334-338.
- [10] Q. Huo, Y. Ge, and Z.-D. Feng, "High performance Chinese OCR based on Gabor features, discriminative feature extraction and model training," in *Proc. ICASSP-2001*, May 2001, pp.III-1517-1520.
- [11] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake, "Modified quadratic discriminant functions and the application to Chinese character recognition," *IEEE Trans. on PAMI*, vol. 9, no. 1, pp.149-153, 1987.
- [12] C.-L. Liu, H. Sako, and H. Fujisawa, "Discriminative learning quadratic discriminant function for handwriting recognition," *IEEE Trans. on Neural Networks*, vol. 15, no. 2, pp.430-444, 2004.
- [13] T. Long and L.-W. Jin, "Building compact MQDF classifier for off-line handwritten Chinese characters by subspace distribution sharing," in *Proc. ICDAR-2007*, pp.909-913.
- [14] M. Nakagawa and K. Matsumoto, "Collection of on-line handwritten Japanese character pattern databases and their analysis," *International Journal on Document Analysis* and Recognition, vol. 7, pp.69-81, 2004.
- [15] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer, New York, 1999.
- [16] P. A. Olsen and R. A. Gopinath, "Modeling inverse covaraince matrices by basis expansion," in *Proc. ICASSP*-2002, pp. 945-948.
- [17] P. A. Olsen and R. A. Gopinath, "Modeling inverse covariance matrices by basis expansion," *IEEE Trans. on Speech and Audio Processing*, vol. 12, no. 1, pp.37-46, 2004.
- [18] K. C. Sim and M. J. F. Gales, "Basis superposition precision matrix modeling for large vocabulary continuous speech recognition," in *Proc. ICASSP-2004*, pp.801-804.
- [19] V. Vanhoucke and A. Sankar, "Mixtures of inverse covarainces," *IEEE Trans. on Speech and Audio Processing*, vol.12, no. 3, pp.250-264, 2004.