Discriminative Learning Quadratic Discriminant Function for Handwriting Recognition

Cheng-Lin Liu, *Member, IEEE*, Hiroshi Sako, *Senior Member, IEEE*, and Hiromichi Fujisawa, *Fellow, IEEE*

***Abstract*—In character string recognition integrating segmentation and classification, high classification accuracy and resistance to noncharacters are desired to the underlying classifier. In a previous evaluation study, the modified quadratic discriminant function (MQDF) proposed by Kimura *et al.* was shown to be superior in noncharacter resistance but inferior in classification accuracy to neural networks. This paper proposes a discriminative learning algorithm to optimize the parameters of MQDF with aim to improve the classification accuracy while preserving the superior noncharacter resistance. We refer to the resulting classifier as discriminative learning QDF (DLQDF). The parameters of DLQDF adhere to the structure of MQDF under the Gaussian density assumption and are optimized under the minimum classification error (MCE) criterion. The promise of DLQDF is justified in handwritten digit recognition and numeral string recognition, where the performance of DLQDF is comparable to or superior to that of neural classifiers. The results are also competitive to the best ones reported in the literature.**

***Index Terms*—Discriminative learning quadratic discriminant function (DLQDF), handwritten digit recognition, minimum classification error (MCE), noncharacter resistance, numeral string recognition, pattern classification.**

I. Introduction

STATISTICAL techniques and neural networks are widely used for classification in various pattern recognition problems [1]. Statistical classifiers include linear discriminant function (LDF), quadratic discriminant function (QDF), Parzen window classifier, nearest-neighbor (1-NN) and k-NN rules, etc. The QDF is obtained under the assumption of multivariate Gaussian density for each class. The modified QDF (MQDF) proposed by Kimura *et al.* [2] aims to improve the computation efficiency and classification performance of QDF via eigenvalue smoothing. Alternatively, the regularized discriminant analysis (RDA) of Friedman [3] improves the performance of QDF by covariance matrix interpolation. Hoffbeck and Landgrebe extended the method of RDA to optimize the interpolation coefficients [4].

The parameters of MQDF are estimated via maximum likelihood (ML) estimation of covariance matrices followed by K-L transform. The MQDF is different from the QDF in that the eigenvalues of minor axes are set to a constant. The motivation behind this is to smooth the parameters for compensating for the estimation error on finite sample size. This strategy has been pursued further by some researchers [5]–[7]. In these methods, however, the parameters are determined class by class wherein the classification error is not considered. In contrast, neural classifiers, such as the multilayer perceptron (MLP) and the radial basis function (RBF) classifier, are trained in discriminative learning, where the parameters are adjusted to separate the samples of different classes.

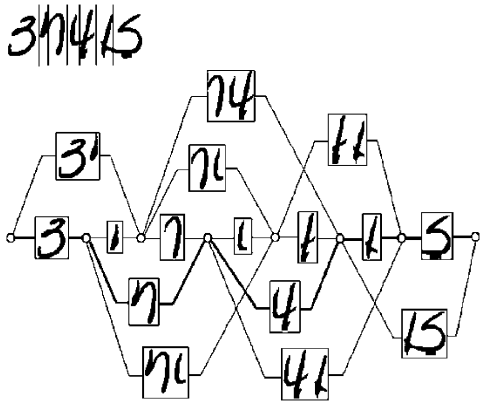

Fig. 1. Segmentation candidate lattice of character string image. Each node represents a separation point and each edge represents a candidate pattern. The path corresponding to the most plausible segmentation is denoted with thick line.

In a performance evaluation study of classifiers in handwritten character recognition, the MQDF was shown to be inferior to neural classifiers in classification accuracy [8]. However, it was shown to be superior in the resistance to noncharacters even though it was not trained with noncharacter data. The noncharacter resistance is important for character string recognition integrating segmentation and classification [9]–[11], where noncharacter patterns are generated in trial segmentation and should be rejected. Fig. 1 shows a segmentation candidate lattice, where we can see that some candidate patterns are noncharacters. The noncharacter resistance of MQDF originates from the assumption of Gaussian density, under which noncharacter patterns generally have low density and setting a threshold, the class decision regions are closed. On the contrary, the decision regions of most neural networks are open such that patterns distinct from training samples may be classified to hypothesized classes with high confidence. The QDF and MQDF have been successfully used in numeral string recognition by, e.g., Saifullah *et al.* [10] and Kimura *et al.* [12]. When using neural classifiers in string recognition, the external verification of candidate patterns is necessary because of the susceptibility to noncharacters [13]–[16].

In this paper, we propose a discriminative learning quadratic discriminant function (DLQDF), whose structure is identical

Manuscript received April 19, 2002; revised November 3, 2003.

The authors are with the Central Research Laboratory, Hitachi, Ltd. Tokyo 185-8601, Japan (e-mail: liucl@crl.hitachi.co.jp; sakou@crl.hitachi.co.jp; fujisawa@crl.hitachi.co.jp).

Digital Object Identifier 10.1109/TNN.2004.824263

to MQDF but the parameters are optimized in discriminative learning. The motivation of DLQDF is to improve the classification accuracy of MQDF and meanwhile preserve the superior noncharacter resistance. As opposed to ML estimation, the parameters of DLQDF are determined in discriminative learning aimed to minimize the classification error. During discriminative learning, the DLQDF adheres to the structure of MQDF under the Gaussian density assumption, and the parameters (mean vectors, eigenvalues and eigenvectors) are optimized under the minimum classification error (MCE) criterion [17], [18]. This can be viewed as a hybrid of informative learning and discriminative learning and can overcome the insufficiencies of both [19]. On large training sample size, discriminative learning generally gives higher classification accuracy than ML estimation. On the other hand, the adherence to Gaussian density assumption renders the classifier resistant to noncharacters.

The MCE criterion was proposed to reduce the classification error of neural networks and learning vector quantization (LVQ) [17]. Unlike the minimum square error (MSE) criterion, the MCE criterion does not depend on any target function and can apply to arbitrary parametric models and discriminant functions. It has been widely used in speech recognition for optimizing the parameters of hidden Markov models (HMM's) to overcome the model deviation from data distribution [18], [20], [21]. In addition, the promise of MCE training was justified in discriminative feature extraction [22], discriminative metric design [23], and other applications including character recognition [24], [25]. In this paper, MCE training is used to optimize the parameters of MQDF, which, as a parametric model under the Gaussian density assumption, is by no means a precise description of actual data.

Some previous works have contributed to the discriminative learning of quadratic classifiers. Fukumoto *et al.* used a learning vector quantization (LVQ) algorithm to optimize the mean vectors of quadratic distance functions [26]. Watanabe and Katagiri have applied the MCE method to the eigenvector learning of subspace method [27]. Kurosawa discussed theoretically the discriminative learning of general quadratic classifiers [28] but the effect was not justified experimentally. Zhang *et al.* proposed a Gaussian mixture classifier with mean vectors and diagonal covariance matrices on transformed space optimized in discriminative learning [29]. On the other hand, Yau and Manry mapped the QDF into a higher-order single-layer neural network and trained the connecting weights by MSE learning [30]. This neural network is actually a polynomial classifier (PC), which was treated in depth by Schürmann *et al.* [31], [32]. In fact, because the connecting weights are not constrained, the decision regions of PC are not closed.

The proposed DLQDF is a fully discriminative version of MQDF, which is unique in that strong constraints are imposed onto the parameters under the Gaussian density assumption. To test the performance of DLQDF, we conducted experiments of handwritten digit recognition and numeral string recognition. The experimental results justified the promise of DLQDF in classification accuracy and noncharacter resistance. Using the DLQDF as the classification unit in a numeral string recognition system, we have achieved competitive results on the string images of NIST Special Database 19 (SD19). The preliminary results of DLQDF have been reported in [33], [34]. We have modified the learning algorithm and extended the experiments since then, however.

The rest of this paper is organized as follows. Section II reviews the MQDF; Section III describes the learning algorithm for DLQDF; Sections IV and V presents the experimental results of handwritten digit recognition and numeral string recognition, respectively. Concluding remarks are provided in Section VI.

II. MQDF

In this section we briefly review the MQDF proposed by Kimura *et al.* [2]. Let us start with the Bayesian decision rule, which classifies the input pattern to the class of maximum a posteriori (MAP) probability out of M classes. Representing a pattern with a feature vector $\mathbf{x} = (x_1, \ldots, x_d)^T$, the a posteriori probability is computed by Bayes rule:

$$P(\omega_i|\mathbf{x}) = \frac{P(\omega_i)p(\mathbf{x}|\omega_i)}{p(\mathbf{x})}, \qquad i = 1, \ldots, M \tag{1}$$

where $P(\omega_i)$ is the a priori probability of class ω_i, $p(\mathbf{x}|\omega_i)$ is the class probability density function (pdf) and $p(\mathbf{x})$ is the mixture density function. Since $p(\mathbf{x})$ is independent of class label, the nominator of (1) can be used as the discriminant function for classification:

$$g(\mathbf{x}, \omega_i) = P(\omega_i)p(\mathbf{x}|\omega_i). \tag{2}$$

The Bayesian classifier is reduced to LDF or QDF under the Gaussian density assumption with varying restrictions. Assume the pdf of each class is multivariate Gaussian

$$p(\mathbf{x}|\omega_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left[-\frac{(\mathbf{x}-\mu_i)^T\Sigma_i^{-1}(\mathbf{x}-\mu_i)}{2}\right] \tag{3}$$

where μ_i and Σ_i denote the mean vector and the covariance matrix of class ω_i, respectively. Inserting (3) into (2), taking the negative logarithm and omitting the common terms under equal a priori probabilities, the QDF is obtained as

$$g_0(\mathbf{x}, \omega_i) = (\mathbf{x}-\mu_i)^T\Sigma_i^{-1}(\mathbf{x}-\mu_i) + \log|\Sigma_i|. \tag{4}$$

The QDF is actually a distance metric in the sense that the class of minimum distance is assigned to the input pattern.

By K-L transform, the covariance matrix can be diagonalized as

$$\Sigma_i = \Phi_i\Lambda_i\Phi_i^T \tag{5}$$

where $\Lambda = \mathrm{diag}[\lambda_{i1}, \ldots, \lambda_{id}]$ with λ_{ij}, $j = 1, \ldots, d$, being the eigenvalues (ordered in decreasing order) of Σ_i, and $\Phi_i = [\phi_{i1}, \ldots, \phi_{id}]$ with ϕ_{ij}, $j = 1, \ldots, d$, being the ordered eigenvectors. Φ_i is ortho-normal (unitary) such that $\Phi_i^T\Phi_i = I$.

According to (5), the QDF can be rewritten in the form of eigenvectors and eigenvalues:

$$g_0(\mathbf{x},\omega_i) = \left[\Phi_i^T(\mathbf{x}-\mu_i)\right]^T \Lambda_i^{-1}\Phi_i^T(\mathbf{x}-\mu_i) + \log|\Lambda_i| = \sum_{j=1}^{d}\frac{1}{\lambda_{ij}}\left[\phi_{ij}^T(\mathbf{x}-\mu_i)\right]^2 + \sum_{j=1}^{d}\log\lambda_{ij}. \quad (6)$$

Replacing the minor eigenvalues with a constant δ_i, the modified quadratic discriminant function (MQDF2[1]) [2] is obtained as

$$\begin{aligned} g_2(\mathbf{x},\omega_i) &= \sum_{j=1}^{k}\frac{1}{\lambda_{ij}}\left[\phi_{ij}^T(\mathbf{x}-\mu_i)\right]^2 + \sum_{j=k+1}^{d}\frac{1}{\delta_i}\left[\phi_{ij}^T(\mathbf{x}-\mu_i)\right]^2 + \sum_{j=1}^{k}\log\lambda_{ij} + (d-k)\log\delta_i \\ &= \sum_{j=1}^{k}\frac{1}{\lambda_{ij}}\left[\phi_{ij}^T(\mathbf{x}-\mu_i)\right]^2 + \frac{1}{\delta_i}r_i(\mathbf{x}) + \sum_{j=1}^{k}\log\lambda_{ij} + (d-k)\log\delta_i \end{aligned} \quad (7)$$

where k denotes the number of principal axes and $r_i(\mathbf{x})$ is the residual of subspace projection:

$$r_i(\mathbf{x}) = \|\mathbf{x}-\mu_i\|^2 - \sum_{j=1}^{k}\left[(\mathbf{x}-\mu_i)^T\phi_{ij}\right]^2. \quad (8)$$

The (7) utilizes the invariance of Euclidean distance:

$$d_E(\mathbf{x},\omega_i) = \|\mathbf{x}-\mu_i\|^2 = \sum_{j=1}^{d}\left[\phi_{ij}^T(\mathbf{x}-\mu_i)\right]^2. \quad (9)$$

The advantage of MQDF2 is multifold. First, it overcomes the bias of minor eigenvalues (which are underestimated on small sample size) such that the classification performance can be improved. Second, for computing the MQDF2, only the principal eigenvectors and eigenvalues are to be stored so that the memory space is reduced. Third, the computation effort is largely saved because the projections to minor axes are not computed.

The parameters of MQDF2 are estimated as follows. The mean vector and covariance matrix of a class are estimated from the sample data of this class. Then the eigenvectors and eigenvalues of covariance matrix are computed by K-L transform. The parameter δ_i can be set to a class-independent constant as proposed by Kimura *et al.* [2]. This setting performs fairly well in practice. Moghaddam and Pentland [35] showed that under ML, the class-dependent δ_i is the average of minor eigenvalues:

$$\delta_i = \frac{\mathrm{tr}(\Sigma_i) - \sum_{j=1}^{k}\lambda_{ij}}{d-k} = \frac{1}{d-k}\sum_{j=k+1}^{d}\lambda_{ij} \quad (10)$$

where $\mathrm{tr}(\Sigma_i)$ denotes the trace of covariance matrix.

[1]MQDF2 refers to one of the two forms of MQDF proposed by Kimura [2].

We have also combined the principle of RDA into MQDF2 by interpolating the covariance matrices and then replacing the minor eigenvalues with the average in the complement subspace. We called the MQDF2 combined with RDA as MQDF3 [8]. By RDA, the covariance matrix is interpolated with an identity matrix by

$$\hat{\Sigma}_i = (1-\gamma)\Sigma_i + \gamma\sigma_i^2 I \quad (11)$$

where $\sigma_i^2 = (1/d)\mathrm{tr}(\Sigma_i)$, and $0 < \gamma < 1$.

III. DLQDF

As the QDF, the MQDF2 assumes that the underlying density function is Gaussian. Therefore, it inevitably suffers from the estimation error from small sample size and the deviation of density model itself. To improve the classification performance, we optimize the parameters (mean vectors, eigenvalues and eigenvectors) of MQDF by discriminative learning under the MCE criterion and refer to the resulting classifier as discriminative learning QDF (DLQDF). The DLQDF is expected to yield lower generalization error than the MQDF. Meanwhile, since the structure of MQDF under the Gaussian density assumption is preserved in DLQDF, the decision regions of DLQDF remain closed.

The initial parameters of DLQDF are inherited from the MQDF2 of the same structure. Then in discriminative learning, the parameters are optimized under the MCE criterion. Actually, as in (7), the MQDF2 has the same form as the QDF except that the diagonalized covariance matrix is modified to

$$\tilde{\Lambda}_i = \mathrm{diag}[\lambda_{i1},\ldots,\lambda_{ik},\delta_i,\ldots,\delta_i]. \quad (12)$$

In the DLQDF, all the class parameters are adjustable in MCE training. The mean vectors can move freely in the feature space, yet the adjustment of eigenvalues and eigenvectors is confined such that the property of covariance matrix (symmetry and positive definiteness) is satisfied. In the form $\Sigma_i = \Phi_i\Lambda_i\Phi_i^T$, when the eigenvalues (diagonal elements of Λ_i) remain positive, the covariance matrix is guaranteed to be symmetric and positively definite. Further, the equivalence between (4) and (6) requires $|\Sigma_i| = |\Lambda_i|$ and the modification of (6) to (7) requires (9). Both the two conditions are implied by the ortho-normality of eigenvectors: $\Phi_i^T\Phi_i = I$. Therefore, in adjusting the parameters of DLQDF, if the eigenvalues remain positive and the eigenvectors hold ortho-normal, the discriminant function adheres to the Gaussian density assumption.

Note that the ortho-normality of Φ_i is not a necessary condition of (6) being a Gaussian discrimiannt function because $\sum_{j=1}^{d}(1/\lambda_{ij})\left[\phi_{ij}^T(\mathbf{x}-\mu_i)\right]^2$ is positive no matter whether Φ_i is unitary or not, though the covariance matrix of the corresponding Gaussian density function may not equal to $\Phi_i\Lambda_i\Phi_i^T$ and the a priori probabilities are not necessarily equal.[2] For the discriminant function (7), however, the positive definiteness of covariance matrix requires $r_i(\mathbf{x}) \geq 0$. This is generally satisfied if the "eigenvectors" are nearly orthogonal and their norms are not greater than one.[3]

[2]Thanks to a reviewer for commenting this.

[3]In adjusting the parameters, if the ortho-normality constraint is not imposed onto the eigenvectors, we still refer to them as "eigenvectors" for simplicity.

In the following we describe the learning algorithm of DLQDF. For convenience of illustration, the DLQDF is rewritten in the form

$$\begin{aligned} d_Q(\mathbf{x},\omega_i) &= g_2(\mathbf{x},\omega_i) \\ &= \sum_{j=1}^{k}\frac{1}{\lambda_{ij}}p_{ij}^2+\frac{1}{\delta_i}\left[d_E(\mathbf{x},\omega_i)-\sum_{j=1}^{k}p_{ij}^2\right] \\ &\quad +\sum_{j=1}^{k}\log\lambda_{ij}+(d-k)\log\delta_i \end{aligned} \tag{13}$$

where $p_{ij}=\phi_{ij}^T(\mathbf{x}-\mu_i)$ denotes the projection onto the eigenvector ϕ_{ij}.

The MCE criterion of Juang *et al.* [17], [18] is illustrated as follows. On a training pattern, a loss function is computed to approximate the classification error and on a training dataset, the empirical loss is minimized by gradient descent to optimize the classifier parameters. Let the discriminant function of class ω_i equals the negative of quadratic distance:

$$g_i(\mathbf{x})=-d_Q(\mathbf{x},\omega_i) \tag{14}$$

following [18], the misclassification measure of a pattern from class ω_c is given by

$$h_c(\mathbf{x})=-g_c(\mathbf{x})+\log\left[\frac{1}{M-1}\sum_{i\neq c}e^{\eta g_i(\mathbf{x})}\right]^{1/\eta} \tag{15}$$

where η is a positive number. When η approaches infinity, the misclassification measure becomes

$$h_c(\mathbf{x})=-g_c(\mathbf{x})+g_r(\mathbf{x}) \tag{16}$$

where $g_r(\mathbf{x})$ is the discriminant function of the closest rival class:

$$g_r(\mathbf{x})=\max_{i\neq c}g_i(\mathbf{x}). \tag{17}$$

The simplification of misclassification measure by setting $\eta\rightarrow\infty$ is helpful to speed up the learning process by stochastic gradient descent[4] [36], where only the parameters involved in the loss function are updated on a training pattern. Finally,

$$h_c(\mathbf{x})=d_Q(\mathbf{x},\omega_c)-d_Q(\mathbf{x},\omega_r). \tag{18}$$

The loss of misclassification is computed by

$$l_c(\mathbf{x})=l_c(h_c)=\frac{1}{1+e^{-\xi h_c}}. \tag{19}$$

On a training dataset $\{(\mathbf{x}^n,c^n)|n=1,2,\ldots,N\}$ (where c^n is the class label of pattern $\mathbf{x}^n$), the empirical loss is computed by

$$L_0=\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{M}l_i(\mathbf{x}^n)I(\mathbf{x}^n\in\omega_i) \tag{20}$$

where $I(\cdot)$ is an indicator function which takes value 1 when the condition in the parentheses is satisfied, otherwise takes value 0.

In discriminative learning, the parameters of DLQDF are adjusted to minimize the classification error on the training dataset. Certainly, the parameters will deviate from the ML estimates (initial parameters inherited from MQDF2). Since the ML estimate does not give optimal classification performance due to the imprecision of Gaussian density model, the discriminative learning will also lead to lower generalization error if trained with large sample size. However, the excessive deviation of parameters from ML estimate may bring about negative effects. Though imprecise, the ML estimate describes fairly well the distribution of training data and provides good resistance to noncharacters. Hence, excessive deviation of parameters from the ML estimate may deteriorate the noncharacter resistance. On the other hand, the freedom of parameters is connected to the classifier capacity and may influence the generalization performance [37]. To alleviate the negative effects, we confine the parameter adjustment by minimizing a regularized objective:

$$L_1=\frac{1}{N}\sum_{n=1}^{N}[l_c(\mathbf{x}^n)+\alpha d_Q(\mathbf{x}^n,\omega_c)] \tag{21}$$

where $d_Q(\mathbf{x}^n,\omega_c)$ is the quadratic distance between the input pattern and the genuine class and α is the regularization coefficient. Since the minimization of in-class quadratic distances leads to ML estimate, by regularization, the parameters are attracted to the ML estimate and adjusted in the vicinity of ML estimate. We will show in experiments that the regularization benefits both generalization performance and noncharacter resistance.

The parameters of DLQDF are updated by stochastic gradient descent:

$$\begin{cases} \theta_c(t+1)=\theta_c(t)-\epsilon(t)\left[\frac{\partial l_c(\mathbf{x})}{\partial\theta_c}+\alpha\frac{\partial d_Q(\mathbf{x},\omega_c)}{\partial\theta_c}\right] \\ \theta_r(t+1)=\theta_r(t)-\epsilon(t)\frac{\partial l_c(\mathbf{x})}{\partial\theta_r} \end{cases} \tag{22}$$

where θ_i stands for the parameters of class ω_i. The partial derivatives of $l_c(\mathbf{x})$ with respect to the parameters are computed by

$$\begin{cases} \frac{\partial l_c(\mathbf{x})}{\partial\theta_c}=\xi l_c(1-l_c)\frac{\partial d_Q(\mathbf{x},\omega_c)}{\partial\theta_c} \\ \frac{\partial l_c(\mathbf{x})}{\partial\theta_r}=-\xi l_c(1-l_c)\frac{\partial d_Q(\mathbf{x},\omega_r)}{\partial\theta_r}. \end{cases} \tag{23}$$

To guarantee the positiveness of eigenvalues in discriminative learning, we utilize the variable transform

$$\begin{cases} \lambda_{ij}=e^{\sigma_{ij}} \\ \delta_i=e^{\tau_i}. \end{cases} \tag{24}$$

The transformed variables $\sigma_{ij}=\log\lambda_{ij}$ and $\tau_{ij}=\log\delta_i$ are updated in gradient descent.

Now the quadratic distance has the form

$$\begin{aligned} d_Q(\mathbf{x},\omega_i) &= \sum_{j=1}^{k}e^{-\sigma_{ij}}p_{ij}^2 \\ &\quad +e^{-\tau_i}\left[d_E(\mathbf{x},\omega_i)-\sum_{j=1}^{k}p_{ij}^2\right] \\ &\quad +\sum_{j=1}^{k}\sigma_{ij}+(d-k)\tau_i. \end{aligned} \tag{25}$$

[4]Stochastic gradient descent is also referred to as generalized probabilistic descent (GPD) [20].

To guarantee the ortho-normality of eigenvectors, we embed the Gram-Schmidt ortho-normalization (GSO) operation in gradient descent such that the eigenvectors of each class are ortho-normalized whenever the parameters are updated on a training pattern. GSO has been used in the discriminative learning of subspace method [27]. For the ordered eigenvectors ϕ_{ij}, $j = 1, \ldots, k$ of class ω_i, in GSO the first eigenvector is simply normalized:

$$\tilde{\phi}_{i1} = \frac{\phi_{i1}}{\|\phi_{i1}\|}. \tag{26}$$

The succeeding ones are ortho-normalized by

$$\begin{cases} \hat{\phi}_{ij} = \phi_{ij} - \sum_{l=1}^{j-1} \left(\phi_{ij}^T \tilde{\phi}_{il}\right) \tilde{\phi}_{il} \\ \tilde{\phi}_{ij} = \frac{\hat{\phi}_{ij}}{\|\hat{\phi}_{ij}\|}. \end{cases} \tag{27}$$

The ortho-normalized eigenvectors are used in the processing of subsequent training patterns.

Based on (22) and (23), the parameter updating is reduced to the computation of partial derivatives of quadratic distance: see (28), shown at the bottom of the page.

In the implementation of MCE training, the learning rate $\epsilon(t)$ decreases progressively and the hardness parameter ξ increases progressively, both in a prespecified schedule. For more details of stochastic gradient descent (GPD), the readers are referred to [18], [20], [36]. To achieve good convergence, the initial values of ϵ and ξ should be carefully selected. In the following are some heuristics for selecting these hyper-parameters.

According to (19), to control the value of loss function in a moderate range between 0 and 1, the value of ξ should be inversely proportional to the magnitude of $h_c(\mathbf{x})$. On initializing the parameters of DLQDF from MQDF2, the magnitude of $h_c(\mathbf{x})$ is estimated to be the average of the abstract difference of quadratic distance between the genuine class and the closest runner-up on the training samples. In MCE training, the value of ξ increases progressively such that the classification loss approaches hard 0–1 decision.

The learning rate ϵ determines the step size of parameter move in parameter space. For good convergence, the step size should be related to the eigenvalues of the Hessian matrix, and so, it is dependent on specific parameters. Assume the Hessian matrix is diagonal, the learning rate for each parameter is inversely proportional to the second-order partial derivative. We set three learning rates ϵ_1, ϵ_2, and ϵ_3, respectively, for three groups of parameters: eigenvalues (including δ_i), mean vectors, and eigenvectors. The second-order derivatives in each group have similar magnitudes. The expectations of second-order derivatives are given in Appendix. Based on them, we set $\epsilon_1 = \epsilon_3$ and $\epsilon_2 = var \cdot \epsilon_1$, where var is the average in-class variance of training samples. The initial value of ϵ_1 is set to a very small number (10^{-4}).

In the regularized loss function (21), to make $\alpha d_Q(\mathbf{x}^n, \omega_c)$ have similar magnitude with $l_c(\mathbf{x})$, α should be inversely proportional to the average magnitude of $d_Q(\mathbf{x}^n, \omega_c)$, which is averaged over the in-class training samples on the initial parameters (ML estimate). Denote the average in-class quadratic distance by D_Q, α is set to $\alpha = \tilde{\alpha}/D_Q$, where $\tilde{\alpha}$ is a moderate number between 0 and 1.

Care should be taken when regularization is adopted ($\alpha > 0$). From (13), we can see that the in-class quadratic distance can be decreased not only by moving the parameters to the ML estimate, but also by contracting the projections. If the eigenvectors are not constrained in MCE training, the projections can be contracted via decreasing the norm of eigenvectors. This will lead to a trivial solution that gives no benefit to generalization and noncharacter resistance. Therefore, regularization must be accompanied with GSO or eigenvector normalization (to unit norm).

IV. Handwritten Digit Recognition

To justify the promise of DLQDF for classification and noncharacter resistance in handwriting recognition, we conducted experiments of handwritten digit recognition and numeral string recognition. This section presents the results of handwritten digit recognition.

A. Experiment Database

In our previous evaluation study of handwritten digit recognition [8], we have compiled an experiment database from the NIST Special Database 19 (SD19) [38]. The training dataset was composed of the digit patterns of 600 writers (writers no. 0–399 and no. 2100–2299), and the test dataset was composed of the digit patterns of 400 writers (writers no. 500–699 and no. 2400–2599). The total numbers of patterns of training dataset and test dataset are 66 214 and 45,398, respectively. To test the rejection ability to noncharacters, we generated synthesized noncharacter data by splitting and merging digit images, simulating the trial segmentation in handwriting recognition. We generated 16 000 noncharacter samples for training and 10 000

$$\begin{cases} \frac{\partial d_Q(\mathbf{x},\omega_i)}{\partial \tau_i} = -e^{-\tau_i}\left[d_E(\mathbf{x},\omega_i) - \sum_{j=1}^{k} p_{ij}^2\right] + d - k, \\ \frac{\partial d_Q(\mathbf{x},\omega_i)}{\partial \sigma_{ij}} = -e^{-\sigma_{ij}} p_{ij}^2 + 1, & j = 1, \ldots, k, \\ \frac{\partial d_Q(\mathbf{x},\omega_i)}{\partial \mu_{il}} = 2\sum_{j=1}^{k}(e^{-\tau_i} - e^{-\sigma_{ij}}) p_{ij}\phi_{ijl} - 2e^{-\tau_i}(x_l - \mu_{il}), & l = 1, \ldots, d, \\ \frac{\partial d_Q(\mathbf{x},\omega_i)}{\partial \phi_{ijl}} = 2(e^{-\sigma_{ij}} - e^{-\tau_i}) p_{ij}(x_l - \mu_{il}), & j = 1, \ldots, k, l = 1, \ldots, d. \end{cases} \tag{28}$$

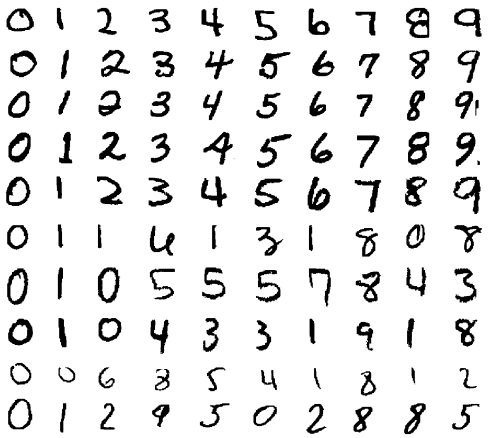

Fig. 2. Example patterns of test digit data.

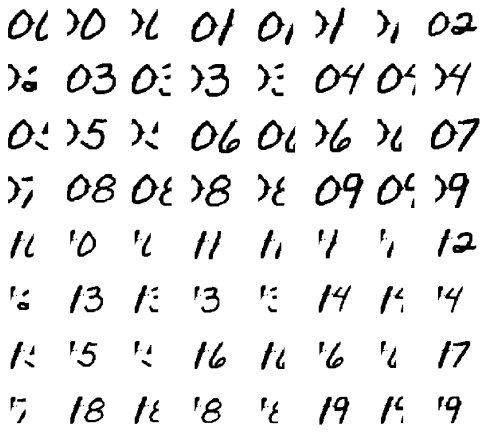

Fig. 3. Example patterns of type I noncharacter data.

noncharacter samples for testing. Using the noncharacter samples in neural network training was shown to largely improve the noncharacter resistance.

The synthesized noncharacter data was called type I noncharacter data. We further used some English letter images from NIST SD19 as type II noncharacter data for testing. The type II noncharacter dataset has 8,800 patterns with 200 from each of 44 classes (all English letters except for "IOZiloqz"). For this time, we exclude some more letters ("DGSbgs") that resemble numerals to obtain a new dataset containing 7,600 patterns. Some examples of test digit data and type I noncharacter data are shown in Figs. 2 and 3, respectively.

For recognition, each pattern (either digit or noncharacter sample) is represented in a feature vector of 100 measurements, extracted in sequential operations of size normalization, contour direction assignment and measurement blurring [39]. The size of normalized image was 35×35 and on each of 4 orientation planes, 5×5 blurring masks were located uniformly to compute 25 measurements. The blurring mask was a Gaussian filter with the variance parameter determined from the sampling theorem. For more details, the readers are referred to [8] and [39]. In the evaluation study, the polynomial classifier (PC) was shown to give the highest accuracy in classifying the test data, while the MQDF performed best in noncharacter rejection even though it was not trained with noncharacter data.

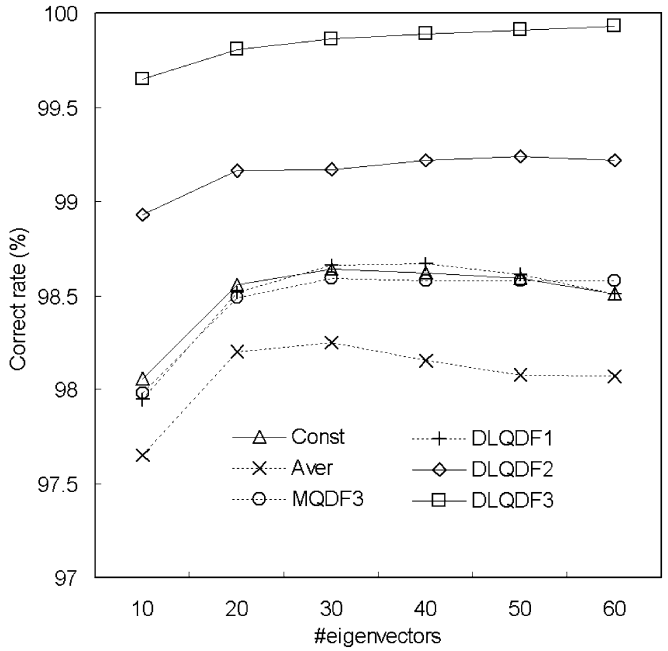

Fig. 4. Accuracies of DLQDFs on the training dataset.*Const* refers to the MQDF2 with class independent constant δ_i, and *Aver* refers to the MQDF2 with δ_i being the average of minor eigenvalues.

B. Effect of Discriminative Learning

To investigate the effect of discriminative learning on classification accuracy, we tested different versions of DLQDFs that update different groups of parameters: DLQDF1 updates the eigenvalues and δ_i; DLQDF2 updates the eigenvalues, δ_i, and mean vectors; and DLQDF3 updates all the parameters. The DLQDFs inherit initial parameters from the MQDF2 with δ_i being the average of minor eigenvalues. In these experiments, GSO was not embedded and the regularization term of in-class distance was not applied, i.e., $\alpha = 0$. Our results show that even if the "eigenvectors" are not constrained to be ortho-normal, the DLQDF gives superior classification performance. Also, it was observed that without GSO, the deviation of eigenvectors from ortho-normality is very small. The results of DLQDF with GSO and regularization will be shown later.

With variable number of eigenvectors, the classification accuracies on the training dataset are shown in Fig. 4. The results of MQDF2 and MQDF3 are also given for comparison. The parameter δ_i of MQDF2 is set to the average of minor eigenvalues (*Aver*) or a class independent constant (*Const*). The interpolation coefficient of MQDF3 is set to $\gamma = 0.2$. We can see that via discriminative learning of parameters, the DLQDFs classify the training data more accurately than MQDF2 and MQDF3. The improvement from MQDF2 (*Aver*) to DLQDF1 is evident, and the accuracy increases in turn from DLQDF1 to DLQDF2 and then to DLQDF3.

The classification accuracies on the test dataset are shown in Fig. 5. From the results, we can see that via discriminative updating of eigenvalues, the accuracy of DLQDF1 approaches that of MQDF3 and MQDF2 (*Const*) though the eigenvalues of DLQDF1 are initialized from MQDF2 (*Aver*). This indicates that the discriminative learning of eigenvalues is effective to improve the classification accuracy. By updating both the eigen-

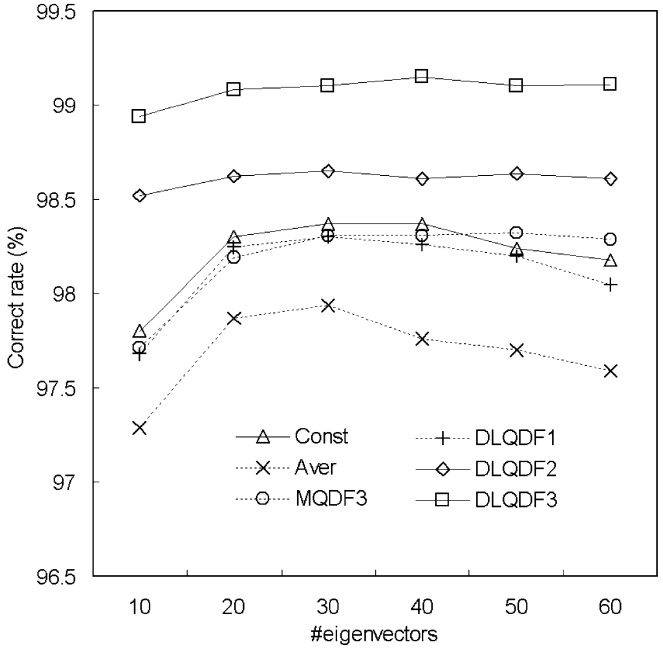

Fig. 5. Accuracies of DLQDFs on the test dataset.

values and mean vectors, the accuracy of DLQDF2 is consistently higher than that of MQDF3 and MQDF2 (*Const*). Further, by updating all the parameters, the improvement of accuracy from DLQDF2 to DLQDF3 is significant.

C. Effect of Regularization

In this subsection, we present the results of MCE training with GSO, regularization of in-class distance in loss function (21), and regularization in initialization (11) (i.e., initializing DLQDF from MQDF3). We also compare the effects of GSO and eigenvector normalization (NORM) only. The regularization of in-class distance is accompanied with GSO or NORM. The regularization coefficients, $\tilde{\alpha}$ and γ, were set to 0.1 and 0.2. The classification accuracies on the test dataset are listed in Table I, where the first row shows the results of DLQDF without normalization, and the rightmost column shows the average ortho-normalization error of eigenvectors, which is calculated by

$$e(\Phi) = \frac{1}{\frac{M \cdot k(k+1)}{2}} \sum_{i=1}^{M} \left[\sum_{j=1}^{k} \left(||\phi_{ij}||^2 - 1 \right)^2 + \sum_{j=1}^{k-1} \sum_{l=j+1}^{k} (\phi_{ij}^T \phi_{il})^2 \right]$$

at $k = 50$. We can see that even when GSO is not embedded in MCE training, the ortho-normalization error of eigenvectors is very small. This is because the parameters of DLQDF move in the vicinity of initial parameters and do not deviate much from the ML estimate.

Comparing the results of Table I, we can see that both GSO and eigenvector normalization deteriorate the classification performance, while the covariance matrix interpolation ($\gamma > 0$) does not affect the performance. However, when accompanied with GSO, the regularization of in-class distance ($\tilde{\alpha} > 0$) improves the generalization accuracy considerably. This justifies that attracting the parameters to ML estimate by regularization is benefitial to the generalization performance.

TABLE I
ACCURACIES OF DLQDF (%) WITH AND WITHOUT REGULARIZATION

Regu	coefficient	$k = 20$	30	40	50	$e(\Phi)$
w/o	w/o	99.08	99.10	99.15	99.10	5.02×10^{-4}
GSO	$\tilde{\alpha} = 0$	99.07	99.10	99.10	99.06	0
	$\tilde{\alpha} = 0.1$	99.13	99.17	99.14	99.17	0
	$\tilde{\alpha} = 0.2$	99.15	**99.19**	99.16	99.16	0
NORM	$\tilde{\alpha} = 0$	99.05	99.09	99.09	99.06	4.45×10^{-4}
	$\tilde{\alpha} = 0.1$	99.09	99.11	99.10	99.08	4.41×10^{-4}
	$\tilde{\alpha} = 0.2$	99.08	99.09	99.06	99.05	4.70×10^{-4}
Interpo	$\gamma = 0.1$	99.10	99.11	99.14	99.11	6.75×10^{-4}
	$\gamma = 0.2$	99.09	99.10	99.11	99.11	8.02×10^{-4}

TABLE II
CLASSIFICATION ACCURACIES (%) OF NEURAL CLASSIFIERS

Classifier	MSE	MCE	Enhanced
MLP	**98.93**	98.89	98.90
PC	99.11	99.06	**99.12**

D. Comparison With Neural Classifiers

We compare the performance of DLQDF with two neural classifiers, MLP (multilayer perceptron) and PC (polynomial classifier) [31], [32], on the same training dataset and test dataset. The MLP has one hidden layer of 300 units, trained with the BP (back-propagation) algorithm [40]. The PC uses the linear and binomial terms of 70 principal components as well as the projection residual as the inputs of single-layer classification units. We have tried variable sizes of MLP and PC and the setting of 300 hidden units for MLP and 70 principal components for PC nearly achieved the best generalization accuracy. Each neural classifier has three versions: MSE training, MCE training, and MSE training with noncharacter data (enhanced version, called EMLP or EPC). When trained with noncharacter data, the noncharacter samples are added to the training digit data and for each noncharacter sample, all the target outputs are set to zero. This strategy has been shown previously to largely improve the noncharacter resistance of neural classifiers [8], [41], [42].

The classification accuracies of neural classifiers are listed in Table II. We can see that for either MLP or PC, the three versions yield similar accuracies on the test data. In this case, MCE training does not give higher generalization accuracy than MSE training. We have applied the MCE training to many different classifiers and found that MCE training yields higher accuracy than MSE training only for simple classifiers such as the single-layer perceptron. From our viewpoint, the advantage of MCE training mainly lies in the applicability to arbitrary discriminant functions while MSE training is limited to the classifiers with 0–1 target functions.

It is shown that the highest accuracies of MLP (98.93%) and PC (99.12%) are lower than that of DLQDF (99.19%). This jus-

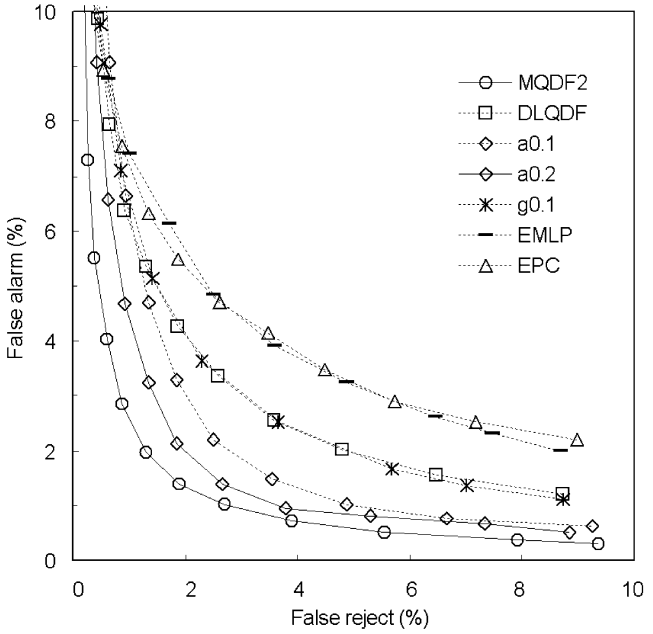

Fig. 6. Rejection performance of type I noncharacters.

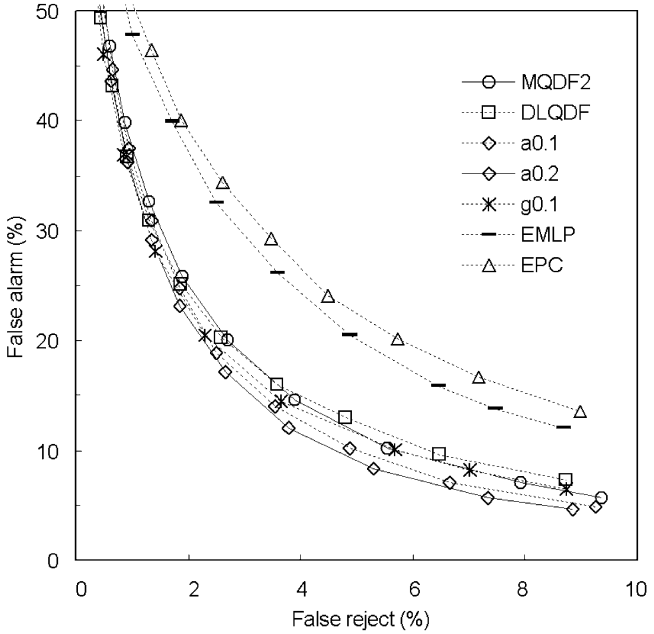

Fig. 7. Rejection performance of type II noncharacters.

tifies the promise of discriminative learning in parameter optimization of MQDF. Without discriminative learning, the highest accuracy of MQDF2 (*Const*) is 98.37% (when using 30 eigenvectors for each class). Thus, the DLQDF reduces the error rate of MQDF2 by the factor of 50 percent.

E. Noncharacter Rejection

We compare the noncharacter rejection performance of MQDF and DLQDF with that of neural classifiers trained with noncharacter data (EMLP and EPC). The rejection performance is measured in the tradeoff between the false reject of digit patterns and the false acceptance of noncharacters, and the rejection decision is made according to the output of top rank class compared to variable thresholds.

The false reject-acceptance tradeoffs for type I noncharacters and type II noncharacters are shown in Figs. 6 and 7, respectively. In the plane of coordinates, the tradeoff curve that is closer to the origin indicates better rejection behavior. In the figures, "MQDF2" refers to the MQDF2 with class-independent constant δ_i, "DLQDF" refers to the DLQDF without regularization, "a0.1" and "a0.2" refer to in-class distance regularization with $\tilde{\alpha} = 0.1$ and $\tilde{\alpha} = 0.2$, respectively, and "g0.1" refers to $\gamma = 0.1$ in initialization. Either the MQDF2 or the DLQDF uses 30 eigenvectors for each class.

We can see that on both two types of noncharacter data, the MQDF2 exhibits superior rejection behavior, whereas the performance of EMLP and EPC are inferior though they were trained with noncharacter samples. On type I noncharacters, the rejection behavior of DLQDFs is intermediate between MQDF2 and EMLP, and the performance is improved evidently by in-class distance regularization. On type II noncharacters, the rejection behavior of DLQDFs is close to that of MQDF2. The DLQDF gives promising noncharacter rejection performance because the parameters adhere to the structure of MQDF under the Gaussian density assumption, particularly when the parameters are attracted to ML estimate by in-class distance regularization.

V. Numeral String Recognition

To compare the performance of classifiers in handwriting recognition, we have built a baseline numeral string recognition system integrating segmentation and classification [34]. The DLQDF and neural classifiers are evaluated by using them as the classification unit in the integrated string recognition system. The performance of string recognition was tested on the numeral string images extracted from the handwriting sample form (HSF) images of NIST SD19.

A. Overview of String Recognition System

Handwritten character string recognition finds wide applications including automatic mail sorting, bankcheck processing, form processing, etc. This problem is challenging due to the variability of writing styles, irregularity of character spacing, and touching of characters. Since the characters cannot be reliably segmented before classification, the hypothesis-verification (integrated segmentation and recognition, ISR) or holistic recognition strategy is adopted. The holistic recognition is only applicable to limited vocabulary, whereas the ISR can be flexibly used with any linguistic context and geometric layout. Though the speech recognition methods such as HMM's are getting success in handwriting recognition, the problem of handwriting recognition is different in that the shape information in handwriting is very rich such that segmentation-based recognition is successful.

In ISR, a character classifier is used to classify and verify the candidate patterns generated in trial segmentation. The classifier can be trained at character-level or string-level.

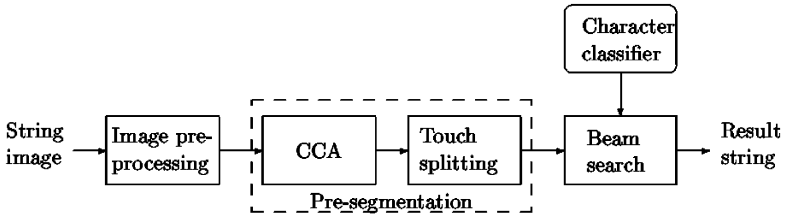

Fig. 8. Diagram of numeral string recognition system.

This paper adopts the former scheme, i.e., character-level training with isolated samples. The string-level training is receiving interests in recent years and was shown to yield higher string recognition accuracy than character-level training [25], [43]–[45]. However, the advantage of character-level training lies in that very large dataset of isolated character samples are available for training, and the trained classifier can be used flexibly in various geometric and linguistic contexts. For a specified context, the character-level trained classifier can be boosted up in string-level training. On the other hand, by appropriately integrating the information from multiple sources (layout, character likeliness, classification score, linguistics, etc.), character-level training can yield competitive string recognition performance.

The intention of our experiments is not to reach the best string recognition performance, but to compare the performance of character classifiers in ISR, so we only utilize the character shape information in classification, while the geometric features (character size, position, inter-character relationship, etc.) and linguistics are not considered. Further, in trial segmentation (presegmentation), we use a simple procedure for splitting touching characters.

The diagram of the numeral string recognition system is shown in Fig. 8. Considering that some string images have underlines, in image preprocessing, the underline is detected according to horizontal projection profile and removed. The presegmentation module has two steps: connected component analysis (CCA) and touching pattern splitting. In CCA, very small components are removed and the connected components that heavily overlap in horizontal direction are merged because they are likely to compose the same character. After overlapping component merging, the components (not necessarily connected now) with large width (relative to the estimated string height) or width/height ratio are considered as potential touching patterns.

For splitting of touching patterns, the upper and lower profile curves are analyzed to generate candidate cuts. As shown in Fig. 9, the corner points of profile curves are found by polygonal approximation [46]. Upward corners in upper profile and downward corners in lower profile correspond to potential cut positions. If an upper cut and a lower cut are close to each other, they are merged into one new cut at the position of narrower image column. After touching pattern splitting, the string image is represented as a sequence of primitive image segments. Fig. 10 shows some examples of image segment sequences. In string recognition, consecutive segments are combined to form candidate character patterns subject to constraints of size and position. The maximum number of segments composing a candidate pattern is set to four and the size and position are loosely constrained such that nearly all the legal characters are composed. The candidate patterns in a string image form a candidate lattice like that in Fig. 1.

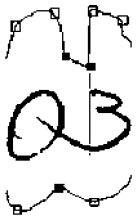

Fig. 9. Touching pattern splitting by profile curve analysis.

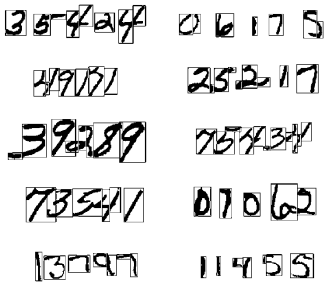

Fig. 10. Primitive image segments after presegmentation. Each segment is enclosed with a rectangular box.

In the candidate lattice, each path from the start node to the end node (goal) corresponds to a candidate segmentation of the string image. The optimal path in sense of maximum score or minimum cost gives the segmentation-recognition result, which is found by beam search. A segmentation path is composed of a sequence of pattern-class pairs. Because linguistic information is not available in numeral string recognition, it is reasonable to assume that the constituent patterns are independent. Accordingly, the path can be scored in the product of class conditional probabilities

$$p(\mathbf{x}_1,\ldots,\mathbf{x}_n|C_{i_1},\ldots,C_{i_n}) = \prod_{j=1}^{n} p(\mathbf{x}_j|C_{i_j}) \qquad (29)$$

where $\mathbf{x}_j$, $j = 1,\ldots,n$, denote the constituent patterns, and C_{i_j}, $j = 1,\ldots,n$, denote the corresponding class labels. In practice, however, the class probabilities are not output by clas-

sifiers. Hence, we instead evaluate a path in respect of the accumulated cost

$$D(\mathbf{x}_1,\ldots,\mathbf{x}_n,C_{i_1},\ldots,C_{i_n})=\sum_{j=1}^{n}d(\mathbf{x}_j,C_{i_j}) \tag{30}$$

where $d(\mathbf{x}_j,C_{i_j})$ denotes the cost/dissimilarity/distance of classifying candidate pattern $\mathbf{x}_j$ to class C_{i_j}. For neural networks, the class cost can be taken as the negative of class outputs.

To achieve high string recognition performance, it is hoped that the correct segmentation path (composed of legal character patterns) have minimum cost while other paths have high costs. Hence, it is desirable to the underlying classifier that each character pattern is given low dissimilarity to the correct class and high dissimilarities to other classes (classification accuracy); while for a noncharacter candidate pattern, all the character classes should be given high dissimilarities (noncharacter resistance).

Since the string length of test string image is unknown and the accumulated cost is biased to short strings, we instead use the normalized cost with respect to the string length for path search:

$$ND(\mathbf{x}_1,\ldots,\mathbf{x}_n,C_{i_1},\ldots,C_{i_n})=\frac{1}{n}\sum_{j=1}^{n}d(\mathbf{x}_j,C_{i_j}). \tag{31}$$

To find the optimal path, we previously used dynamic programming (DP) search and obtained fairly good results [34]. Since the normalized cost at an intermediate node depends on all the preceding nodes, DP search does not guarantee to find the optimal path, and beam search works better. By beam search, at an intermediate node, the partial paths from start to the preceding nodes are extended to the current node and a number of extended partial paths of high scores are retained for further extension. At the goal node, the path of maximum score (minimum normalized cost) gives the segmentation-recognition result.

The string recognition result is considered correct if the class string corresponding to the minimum cost path is identical to the ground truth label. To reject the result strings of low score (high cost) can improve the reliability of string recognition. In our previous work, rejection was made using a local threshold, whereby a segmentation path is abandoned whenever the distance of a constituent pattern exceeds a threshold [34]. To improve the reject-error tradeoff, we now consider the difference of normalized cost between the two best result strings. The result string is rejected if the difference of cost is smaller than a threshold. Variable thresholds are tried to reduce the string error rate to a specified level. The two result strings either have different segmentation paths, or correspond to the same segmentation path but a constituent pattern is associated with two different classes.

B. Numeral String Recognition Results

The numeral string images of NIST SD19 have been tested by many researchers [13], [14], [16], [47]. In NIST SD19, the number of HSF pages is very large and different researchers rarely selected the same partition of samples in training and testing. For example, Ha *et al.* [13] and Kim *et al.* [14] tested on selected samples of writers no. 1800–1899 and no. 2000–2099, whereas Oliveira *et al.* [16] tested on selected samples of hsf-7 (writers no. 3600–4099). The page and field numbers that they selected are not available.

TABLE III
CORRECT RATES OF NIST 3-DIGIT STRING IMAGES

Classifier	Train	No reject	1% error	0.5% error
MQDF2	Const	94.51	88.21	81.37
DLQDF	w/o	95.73	90.58	86.99
	a0.1	**96.48**	92.28	**89.97**
	a0.2	**96.48**	**92.95**	89.09
MLP	MSE	82.52	39.23	29.95
	MCE	69.99	15.99	10.70
	Enhanced	95.12	88.82	86.79
PC	MSE	74.80	14.30	6.30
	MCE	75.54	11.65	4.13
	Enhanced	96.14	92.28	87.80

TABLE IV
CORRECT RATES OF NIST 6-DIGIT STRING IMAGES

Classifier	Train	No reject	1% error	0.5% error
MQDF2	Const	93.75	75.39	63.70
DLQDF	w/o	95.45	83.96	77.50
	a0.1	95.51	90.62	88.17
	a0.2	95.58	90.35	**88.38**
MLP	MSE	63.70	14.14	11.35
	MCE	37.32	2.52	2.04
	Enhanced	94.56	87.42	83.21
PC	MSE	55.61	7.00	3.33
	MCE	55.61	7.75	3.13
	Enhanced	**96.13**	**91.16**	86.13

In our experiments, we extracted numeral string images from the HSF pages of 300 writers, no. 1800–2099. Among the 300 page images, 4 pages (no. 1965, 1977, 2027, and 2067) with faded ink were excluded. The numeral fields with string length 2–6 were extracted. We tested the numeral string recognition performance on 3-digit and 6-digit string images. From the 1,480 3-digit images and 1,480 6-digit images, we further excluded some samples that mismatch the ground-truth labels. As the result, we have 1,476 3-digit samples and 1,471 6-digit samples in testing.

The classifiers tested in numeral string recognition are MQDF2 (*Const*), DLQDF (without regularization), DLQDF-a0.1 ($\tilde{\alpha}=0.1$), and DLQDF-a0.2 ($\tilde{\alpha}=0.2$). Either the MQDF2 or the DLQDF has 30 eigenvectors for each class. For comparison, the MLP and PC trained by MSE and MCE, and the enhanced versions (EMLP and EPC) are tested as well. For these classifiers, the correct rates of numeral string recognition at no reject, 1% error rate, and 0.5% error rate are given in Tables III and IV, for 3-digit strings and 6-digit strings, respectively.

As shown in Table III, the highest recognition rates of 3-digit strings are given by DLQDF-a0.1 and DLQDF-a0.2, which evidently outperform the MQDF2 and the DLQDF without regularization. When trained with noncharacter samples, the enhanced versions of MLP and PC also perform fairly well. Particularly, the performance of EPC is comparable to that of DLQDF with regularization. In Table IV, the highest recognition rate of 6-digit strings is given by the EPC, yet the DLQDF-a0.1 and DLQDF-a0.2 perform comparably well.

Fig. 11. Examples of numeral string recognition error. The blank columns in the images correspond to candidate cuts, and the segmented characters are enclosed with boxes. "C" denotes correct recognition.

The MLP and PC trained without noncharacter samples, by either MSE or MCE, yield very low correct rates in numeral string recognition. It is evident that the low string recognition rate is due to the inferior noncharacter resistance, since the classification accuracies of MLP and PC trained by either MSE or MCE are very high. On the contrary, the classification accuracy of MQDF2 on isolated digits (98.37%) is evidently lower than those of MLP (98.93%) and PC (99.12%), yet MQDF2 still perform well in numeral string recognition because of the inherent noncharacter resistance resulting from the Gaussian density assumption. The DLQDF outperforms the MQDF2 because it gives higher classification accuracy and preserves the superior noncharacter resistance of MQDF2.

It is noteworthy that though on isolated noncharacter samples, the rejection behavior of EPC is inferior to that of DLQDF, in numeral string recognition, its performance is comparable to that of DLQDF with in-class distance regularization. This may be explained in two ways. First, the difference of noncharacter rejection behavior shown in Figs. 6 and 7 is not significant compared to the difference between neural classifier trained with and without noncharacter samples. Second, the evaluation of noncharacter rejection on the artificial noncharacter samples is not fully relevant to the noncharacter resistance in string recognition though it makes sense to certain extent.

The performance of neural classifiers in string recognition can be improved by training with realistic noncharacter samples collected in string segmentation or by string-level training. Nevertheless, the DLQDF was trained totally without noncharacter samples yet yield competitive string recognition performance. In the future, we should consider training DLQDF with noncharacter samples or at string-level as well.

Fig. 11 shows some string recognition errors given by MQDF2, DLQDF-a0.1 and EPC. Compared to DLQDF and EPC, many of the mis-recognized strings by MQDF2 are due

TABLE V
COMPARISON OF CORRECT RATES OF NIST STRING IMAGES

Method	strlen	#strings	No reject	1% error	0.5% error
Ha et al. [13]	3	986	92.7	79.5	70.5
	6	982	90.3	75.5	66.5
Kim et al. [14]	3	956	97.0	N/A	N/A
	6	961	96.7	N/A	N/A
Oliveira et al. [16]	3	2,385	95.38	90.60	89.84
	6	2,167	93.12	85.68	84.03
DLQDF-a0.1	3	1,476	96.48	92.28	89.97
	6	1,471	95.51	90.62	88.17
EPC	3	1,476	96.14	92.28	87.80
	6	1,471	96.13	91.16	86.13

to the misclassification of correctly segmented characters. The string images of Fig. 11 are mis-recognized by at least one of the three classifiers. The upper part shows the samples that three classifiers give the same segmentation. The differing portion of classification results is shown in parentheses. The lower part shows the samples that three classifiers give different segmentation paths, and the classification results are shown separately. There are two types of string mis-recognition: mis-segmentation and misclassification. The misclassification of correctly segmented patterns is due to image degradation or shape confusion between classes. The mis-segmentation has two reasons: failure of touching pattern splitting and false acceptance of noncharacters.

Our experiments of numeral string recognition are aimed to compare the performance of classifiers, so we completely ignored the geometric features of candidate patterns, which are utilized in most character string recognition systems to evaluate character likeliness and string likelihood (e.g., [16], [48]). Incorporating these features into our system will definitely reduce segmentation errors. Moreover, the recognition performance can be improved by refined preprocessing and presegmentation of string images. For example, the slant normalization can reduce character overlapping and regulate character shape [49], and sophisticated splitting techniques (e.g., [50]) help to precisely locate the cut points of touching patterns.

Though we did not intend to reach the best string recognition performance, the results obtained in our experiments are competitive to the best ones reported in the literature. In experiments on NIST string images, Ha *et al.* [13], Kim *et al.* [14], and Oliveira *et al.* [16] have reported high recognition rates compared to other works. Their recognition rates on 3-digit and 6-digit strings are collected in Table V, with comparison to our results of DLQDF-a0.1 and EPC. It is fair to compare our results with those of [13] and [14] because the string images were extracted from the page images of writers no. 1800–2099. Kim *et al.* [14] reported the highest correct rates but the rates with rejection are not available.

In both [13] and [14], some string samples were excluded selectively (we suppose the excluded samples are hard to recognize). To compare the recognition rates, we should account for the inclusion rate of samples. Out of 200 page images (each page has 5 samples for each of string length 2–6), the inclusion rate of [13] and [14] are $(986 + 982)/(10 \times 200) = 0.984$ and $(956+961)/(10\times200) = 0.9585$, respectively. Out of 300 page images, our inclusion rate is $(1{,}476 + 1{,}471)/(10 \times 300) = 0.9823$. Hence, it is fair to compare our results with those of [13], while the string samples of [14] are easier on average. Overall, our results of numeral string recognition are competitive to the best ones reported previously despite that the previous methods utilized geometric features in verification while we did not.

VI. CONCLUSION

This paper proposed a discriminative learning quadratic discriminant function (DLQDF) for handwriting recognition. The structure of DLQDF adheres to the Gaussian density assumption while the parameters are optimized in discriminative learning under the MCE criterion. The nature of hybrid informative and discriminative learning renders the classifier resistant to noncharacters and accurate in classification. The promise of DLQDF was justified in experiments of handwritten digit recognition and numeral string recognition, wherein the recognition performance of DLQDF is competitive to that of neural classifiers (MLP and PC) and the best results reported in the literature. In addition to handwriting recognition, we suppose that DLQDF is applicable to general pattern classification problems where the class density is approximately unimodal and to pattern sequence recognition integrating segmentation and recognition.

Though we did not intend to reach the best performance of numeral string recognition, our results of segmentation-based recognition using accurate and noncharacter resistant classifiers are promising. The best results were given by DLQDF and the PC trained with noncharacter samples. The noncharacter resistance of DLQDF is inherent from the Gaussian density assumption while that of PC is obtained in noncharacter training. The comparison of MQDF2 and DLQDF justifies the importance of classification accuracy in string recognition, while the comparison of neural classifiers trained with and without noncharacter samples justifies the effect of noncharacter resistance.

Some strategies are under consideration for further improving the string recognition performance. Since the MCE criterion does not depend on the forms of discriminant functions, it is readily applicable to string-level training of classifiers, either DLQDF or neural classifiers. On the other hand, character-level training is flexible for varying layout contexts and the fusion of contextual information can significantly improve the segmentation-recognition performance. The string recognition performance can also be benefited from multiple classifier combination since different classifiers give different recognition errors. Moreover, a high performance string recognition system entails elaborate image preprocessing and presegmentation procedures.

APPENDIX

The second-order partial derivatives of loss function (19) with respect to class parameters are computed as follows. The derivatives are computed with respect to the parameters of two classes:

the genuine class of input pattern and the closest runner-up. The derivatives with respect to the parameters of rival class have similar forms and magnitudes to those of genuine class, so we will give the derivatives of genuine class only.

Let θ_c stands for a parameter of genuine class ω_c, the second-order derivative is computed by

$$\begin{aligned}\frac{\partial^2 l_c(\mathbf{x})}{\partial\theta_c^2} &= \frac{\partial}{\partial\theta_c}\left[\xi l_c(1-l_c)\frac{\partial d_Q(\mathbf{x},\omega_c)}{\partial\theta_c}\right]\\ &= \xi\Bigg[(1-2l_c)\frac{\partial l_c}{\partial\theta_c}\frac{\partial d_Q(\mathbf{x},\omega_c)}{\partial\theta_c}\\ &\qquad + l_c(1-l_c)\frac{\partial^2 d_Q(\mathbf{x},\omega_c)}{\partial\theta_c^2}\Bigg]\\ &= \xi^2 l_c(1-l_c)(1-2l_c)\left[\frac{\partial d_Q(\mathbf{x},\omega_c)}{\partial\theta_c}\right]^2\\ &\quad + \xi l_c(1-l_c)\frac{\partial^2 d_Q(\mathbf{x},\omega_c)}{\partial\theta_c^2}\end{aligned} \tag{32}$$

which signifies that $(\partial^2 l_c(\mathbf{x})/\partial\theta_c^2)$ is the weighted sum of $[\partial d_Q(\mathbf{x},\omega_c)/\partial\theta_c]^2$ and $(\partial^2 d_Q(\mathbf{x},\omega_c)/\partial\theta_c^2)$. While $(\partial d_Q(\mathbf{x},\omega_c)/\partial\theta_c)$ is computed as in (28), the second-order derivatives of $d_Q(\mathbf{x},\omega_c)$ with respective to the eigenvalues, mean vectors, and eigenvectors are computed by (33), shown at the bottom of the page.

The expected magnitude of partial derivatives can be approximated on the initial parameters of DLQDF (ML estimates). Denote the average variance of class ω_c by var, it holds true that

$$var = \frac{1}{N_c}\sum_{\mathbf{x}\in\omega_c}\|\mathbf{x}-\mu_c\|^2 = \sum_{j=1}^{d}\lambda_{cj}$$

and

$$\lambda_{cj} = \frac{1}{N_c}\sum_{\mathbf{x}\in\omega_c}\left[\phi_{cj}^T(\mathbf{x}-\mu_c)\right]^2 = \frac{1}{N_c}\sum_{\mathbf{x}\in\omega_c}p_{cj}^2$$

where N_c denotes the number of samples of class ω_c. It is shown in the above that $p_{cj}^2, \lambda_{cj}, \delta_c, \|\mathbf{x}-\mu_c\|^2$ and the variance have the same scale. The average value of ϕ_{cjl}^2 can be taken as $E[\phi_{cjl}^2] = 1/d$ because $\sum_{l=1}^{d}\phi_{cjl}^2 = 1$. According to these relations and (33), the expectations of second-order partial derivatives are estimated on training samples shown in (34) at the bottom of the page. Similarly, we can show that

$$\begin{cases} E\left\{\left[\frac{\partial d_Q(\mathbf{x},\omega_i)}{\partial\sigma_{cj}}\right]^2\right\} \propto 1, & j=1,\ldots,k\\ E\left\{\left[\frac{\partial d_Q(\mathbf{x},\omega_i)}{\partial\mu_{cl}}\right]^2\right\} \propto \frac{1}{\delta_c}, & l=1,\ldots,d\\ E\left\{\left[\frac{\partial d_Q(\mathbf{x},\omega_i)}{\partial\phi_{cjl}}\right]^2\right\} \propto 1, & j=1,\ldots,k, l=1,\ldots,d. \end{cases} \tag{35}$$

Combining (34) and (35) and considering that $\delta_c \propto var$, we have

$$\begin{cases} E\left[\frac{\partial^2 l_c(\mathbf{x})}{\partial\sigma_{cj}^2}\right] \propto 1, & j=1,\ldots,k\\ E\left[\frac{\partial^2 l_c(\mathbf{x})}{\partial\mu_{cl}^2}\right] \propto \frac{1}{var}, & l=1,\ldots,d\\ E\left[\frac{\partial^2 l_c(\mathbf{x})}{\partial\phi_{cjl}^2}\right] \propto 1, & j=1,\ldots,k, l=1,\ldots,d. \end{cases} \tag{36}$$

The expectations of second-order derivatives are useful for setting the learning steps of discriminative learning as in Section III.

ACKNOWLEDGMENT

The authors are grateful to anonymous reviewers for their invaluable comments and suggestions.

$$\begin{cases} \frac{\partial^2 d_Q(\mathbf{x},\omega_c)}{\partial\sigma_{cj}^2} = e^{-\sigma_{cj}}p_{cj}^2 = \frac{p_{cj}^2}{\lambda_{cj}}, & j=1,\ldots,k,\\ \frac{\partial^2 d_Q(\mathbf{x},\omega_c)}{\partial\mu_{cl}^2} = 2\sum_{j=1}^{k}(e^{-\sigma_{cj}}-e^{-\tau_c})\phi_{cjl}^2 + 2e^{-\tau_c} = 2\sum_{j=1}^{k}\left(\frac{1}{\lambda_{cj}}-\frac{1}{\delta_c}\right)\phi_{cjl}^2 + \frac{2}{\delta_c}, & l=1,\ldots,d,\\ \frac{\partial^2 d_Q(\mathbf{x},\omega_c)}{\partial\phi_{cjl}^2} = 2\sum_{j=1}^{k}(e^{-\sigma_{cj}}-e^{-\tau_c})(x_l-\mu_{cl})^2 = 2\sum_{j=1}^{k}\left(\frac{1}{\lambda_{cj}}-\frac{1}{\delta_c}\right)(x_l-\mu_{cl})^2, & j=1,\ldots,k, l=1,\ldots,d. \end{cases} \tag{33}$$

$$\begin{cases} E\left[\frac{\partial^2 d_Q(\mathbf{x},\omega_c)}{\partial\sigma_{cj}^2}\right] = \frac{E[p_{cj}^2]}{\lambda_{cj}} = 1, & j=1,\ldots,k,\\ E\left[\frac{\partial^2 d_Q(\mathbf{x},\omega_c)}{\partial\mu_{cl}^2}\right] = 2\sum_{j=1}^{k}(\frac{1}{\lambda_{cj}}-\frac{1}{\delta_c})E[\phi_{cjl}^2] + \frac{2}{\delta_c} \propto \frac{1}{\delta_c}, & l=1,\ldots,d,\\ E\left[\frac{\partial^2 d_Q(\mathbf{x},\omega_c)}{\partial\phi_{cjl}^2}\right] = 2\sum_{j=1}^{k}(\frac{1}{\lambda_{cj}}-\frac{1}{\delta_c})E[(x_l-\mu_{cl})^2] \propto 1, & j=1,\ldots,k, l=1,\ldots,d. \end{cases} \tag{34}$$

REFERENCES

[1] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 22, pp. 4–37, Jan. 2000.

[2] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake, "Modified quadratic discriminant functions and the application to chinese character recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, pp. 149–153, Jan. 1987.

[3] J. H. Friedman, "Regularized discriminant analysis," *J. Am. Statist. Ass.*, vol. 84, no. 405, pp. 165–175, 1989.

[4] J. P. Hoffbeck and D. A. Landgrebe, "Covariance matrix estimation and classification with limited training data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, pp. 763–767, July 1996.

[5] M. Sakai, M. Yoneda, and H. Hase, "A new robust quadratic discriminant function," in *Proc. 14th Int. Conf. Pattern Recognition*, Brisbane, 1998, pp. 99–102.

[6] T. Kawatani, "Handwritten kanji recognition with determinant normalized quadratic discriminant function," in *Proc. 15th Int. Conf. Pattern Recognition*, vol. 2, Barcelona, 2000, pp. 343–346.

[7] M. Iwamura, S. Omachi, and H. Aso, "A modification of eigenvalues to compensate estimation errors of eigenvectors," in *Proc. 15th Int. Conf. Pattern Recognition*, vol. 2, Barcelona, 2000, pp. 378–381.

[8] C.-L. Liu, H. Sako, and H. Fujisawa, "Performance evaluation of pattern classifiers for handwritten character recognition," *Int. J. Document Analysis and Recognition*, vol. 4, no. 3, pp. 191–204, 2002.

[9] H. Fujisawa, Y. Nakano, and K. Kurino, "Segmentation methods for character recognition: From segmentation to document structure analysis," *Proc. IEEE*, vol. 80, pp. 1079–1092, July 1992.

[10] Y. Saifullah and M. T. Manry, "Classification-based segmentation of ZIP codes," *IEEE Trans. System Man Cybernet.*, vol. 23, no. 5, pp. 1437–1443, 1993.

[11] R. G. Casey and E. Lecolinet, "A survey of methods and strategies in character segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, pp. 690–706, July 1996.

[12] F. Kimura, Y. Miyake, and M. Sridhar, "Handwritten ZIP code recognition using lexicon free word recognition algorithm," in *Proc. 3rd Int. Conf. Document Analysis and Recognition*, Montreal, 1995, pp. 906–910.

[13] T. M. Ha, J. Zimmermann, and H. Bunke, "Off-line handwritten numeral string recognition by combining segmentation-based and segmentation-free methods," *Pattern Recognition*, vol. 31, no. 3, pp. 257–272, 1998.

[14] K. K. Kim, Y. K. Chung, J. H. Kim, and C. Y. Suen, "Recognition of uncontrained handwritten numeral strings using decision value generator," in *Proc. 6th Int. Conf. Document Analysis and Recognition*, Seattle, 2001, pp. 14–17.

[15] J. Zhou, A. Krzyzak, and C. Y. Suen, "Verification—A method of enhancing the recognizers of isolated and touching handwritten numerals," *Pattern Recognition*, vol. 35, no. 5, pp. 1179–1189, 2002.

[16] L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "Automatic recognition of handwritten numeral strings: A recognition and verification strategy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, pp. 1438–1454, Nov. 2002.

[17] B.-H. Juang and S. Katagiri, "Discriminative learning for minimum error classification," *IEEE Trans. Signal Processing*, vol. 40, pp. 3043–3054, Dec. 1992.

[18] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech and Audio Processing*, vol. 5, pp. 257–265, May 1997.

[19] Y. D. Rubenstein and T. Hastie, "Discriminative vs informative learning," in *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining*, CA, 1997, pp. 49–53.

[20] S. Katagiri, B.-H. Juang, and C.-H. Lee, "Pattern recognition using a family of design algorithms based upon the generalized probabilistic descent method," *Proc. IEEE*, vol. 86, pp. 2345–2375, 1998.

[21] W. Chou, "Discriminant-function-based minimum recognition error pattern-recognition approach to speech recognition," *Proc. IEEE*, vol. 88, pp. 1201–1223, Aug. 2000.

[22] A. Biem, S. Katagiri, and B.-H. Juang, "Pattern recognition using discriminative feature extraction," *IEEE Trans. Signal Processing*, vol. 45, pp. 500–504, Feb. 1997.

[23] H. Watanabe, T. Yamaguchi, and S. Katagiri, "Discriminative metric design for robust pattern recognition," *IEEE Trans. Signal Processing*, vol. 45, pp. 1655–2662, Nov. 1997.

[24] Q. Huo, Y. Ge, and Z.-D. Feng, "High performance Chinese OCR based on Gabor features, discriminative feature extraction and model training," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 3, Utah, USA, 2001, pp. 1517–1520.

[25] A. E. Biem, "Minimum classification error training of hidden Markov models for handwriting recognition," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 3, Utah, USA, 2001, pp. 1529–1532.

[26] T. Fukumoto, T. Wakabayashi, F. Kimura, and Y. Miyake, "Accuracy improvement of handwritten character recognition by GLVQ," in *Proc. 7th Int. Workshop Frontiers of Handwriting Recognition*, Amsterdam, The Netherlands, 2000, pp. 271–280.

[27] H. Watanabe and S. Katagiri, "Subspace method for minimum error pattern recognition," *IEICE Trans. Information Syst.*, vol. E80-D, no. 12, pp. 1095–1104, 1997.

[28] Y. Kurosawa, "Probabilistic descent method applied to similarity and distance measure of quadratic form for pattern recognition technical report of IEICE," (in Japanese), PRMU97-181 (1997-12).

[29] R. Zhang, X. Ding, and J. Zhang, "Offline handwritten character recognition based on discriminative training of orthogonal Gaussian mixture model," in *Proc. 6th Int. Conf. Document Analysis and Recognition*, Seattle, WA, 2001, pp. 221–225.

[30] H.-C. Yau and M. T. Manry, "Iterative improvement of a Gaussian classifier," *Neural Networks*, vol. 3, no. 4, pp. 437–443, 1990.

[31] J. Schürmann, *Pattern Classification: A Unified View of Statistical and Neural Approaches*. West Sussex, U.K.: Wiley Interscience, 1996.

[32] U. Kreßel and J. Schürmann, "Pattern classification techniques based on function approximation," in *Handbook of Character Recognition and Document Image Analysis*, H. Bunke and P. S. P Wang, Eds. Singapore: World Scientific, 1997, pp. 49–78.

[33] C.-L. Liu, H. Sako, and H. Fujisawa, "Learning quadratic discriminant function for handwritten character recognition," in *Proc. 16th Int. Conf. Pattern Recognition*, vol. 4, Quebec, Canada, 2002, pp. 44–47.

[34] ——, "Integrated segmentation and recognition of handwritten numerals: Comparison of classification algorithms," in *Proc. 8th Int. Workshop Frontiers of Handwriting Recognition*, Ontario, Canada, 2002, pp. 303–308.

[35] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, pp. 696–710, July 1997.

[36] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Stat.*, vol. 22, pp. 400–407, 1951.

[37] V. Vapnik, *The Nature of Statistical Learning Theory*. New Work: Springer-Verlag, 1995.

[38] P. J. Grother. (1995) NIST Special Database 19: Handprinted Forms and Characters Database, Technical Rep.

[39] C.-L. Liu, Y.-J. Liu, and R.-W. Dai, "Preprocessing and statistical/structural feature extraction for handwritten numeral recognition," in *Progress of Handwriting Recognition*, A. Downton and S. Impedovo, Eds. Singapore: World Scientific, 1997, pp. 161–168.

[40] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagation errors," *Nature*, vol. 323, no. 9, pp. 533–536, 1986.

[41] J. Bromley and J. S. Denker, "Improving rejection performance on handwritten digits by training with rubbish," *Neural Computation*, vol. 5, pp. 367–370, 1993.

[42] L. Yaeger, R. Lyon, and B. Webb, "Effective training of a neural network character classifier for word recognition," in *Advances in Neural Information Processing Systems 9*, M. Mozer, M. Jordan, and T. Petsche, Eds. Cambridge, MA: MIT Press, 1997.

[43] Y. LeCun, L. Bottou, Y. Bengyo, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, pp. 2278–2324, Nov. 1998.

[44] W.-T. Chen and P. Gader, "Word level discriminative training for handwritten word recognition," in *Proc. 7th Int. Workshop Frontiers of Handwriting Recognition*, Amsterdam, The Netherlands, 2000, pp. 393–402.

[45] Y. H. Tay, P.-M. Lallican, M. Khalid, S. Knerr, and C. Viard-Gaudin, "An analytical handwritten word recognition system with word-level discriminant training," in *Proc. 6th Int. Conf. Document Analysis and Recognition*, Seattle, WA, 2001, pp. 726–730.

[46] U. Ramer, "An iterative procedure for the polygonal approximation of plane closed curves," *Computer Graphics and Image Processing*, vol. 1, pp. 244–256, 1972.

[47] S.-W. Lee and S.-Y. Kim, "Integrated segmentation and recognition of handwritten numerals with cascade neural network," *IEEE Trans. Syst. Man Cybernet. C*, vol. 29, pp. 285–290, May 1999.

[48] P. D. Gader, M. Mohamed, and J.-H. Chiang, "Handwritten word recognition with character and inter-character neural networks," *IEEE Trans. System Man Cybernet. B.*, vol. 27, no. 1, pp. 158–164, 1997.

[49] A. d. S. Britto Jr, R. Sabourin, E. Lethelier, F. Bortolozzi, and C. Y. Suen, "Improvement in handwritten numeral string recognition by slant normalization and contextual information," in *Proc. 7th Int. Workshop Frontiers of Handwriting Recognition*, Amsterdam, The Netherlands, 2000, pp. 323–332.

[50] Y.-K. Chen and J.-F. Wang, "Segmentation of single- or multiple-touching handwritten numeral string using background and foreground analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 1304–1317, Nov. 2000.

Cheng-Lin Liu (A'00–M'03) received the B.S. degree in electronic engineering from Wuhan University, Wuhan, China, the M.E. degree in electronic engineering from Beijing Polytechnic University, Beijing, China, and the Ph.D. degree in pattern recognition and artificial intelligence from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 1989, 1992, and 1995, respectively.

From March 1995 to October 1996, he was a Postdoctoral Fellow at the Korea Advanced Institute of Science and Technology (KAIST), Taejon, Korea, and from November 1997 to March 1999, at the Tokyo University of Agriculture and Technology. Afterwards, he became a Research Staff Member at the Central Research Laboratory, Hitachi, Ltd., where he is now a Senior Researcher. His research interests include pattern recognition, artificial intelligence, image processing,neural networks, machine learning, and especially the applications to character recognition and document processing.

Hiroshi Sako (M'88–SM'03) received the B.E. and M.E. degrees in mechanical engineering from Waseda University, Tokyo, Japan, in 1975 and 1977, respectively, and the Dr. Eng. degree in computer science from the University of Tokyo, in 1992.

From 1977 to 1991, he worked in the field of industrial machine vision at the Central Research Laboratory of Hitachi, Ltd., Tokyo, Japan (HCRL). From 1992 to 1995, he was a Senior Research Scientist at Hitachi Dublin Laboratory, Ireland,where he did research in facial and hand gesture recognition. Since 1996, he has been with the HCRL, where he directs a research group of image recognition and character recognition where he is currently a Chief Researcher. Since 1998, he has also been a Visiting Professor at the Japan Advanced Institute of Science and Technology, Hokuriku Postgraduate University, and a Visiting Lecturer at Hosei University, Hosei, Japan, since 2003.

Dr. Sako is a Member of the Institute of Electronics, Information, and Communication Engineers (IEICE) of Japan, the Japanese Society of Artificial Intelligence (JSAI) and the Information Processing Society of Japan (IPSJ). He was a recipient of the 1988 Best Paper Award from the Institute of Electronics, Information, and Communication Engineers (IEICE) of Japan, and an Industrial Paper Award from the 12th ICPR, Jerusalem, Israel, in 1994.

Hiromichi Fujisawa (M'75–SM'00–F'02) received the B.E., M.E., and Doctor of Engineering degrees in electrical engineering from Waseda University, Tokyo, Japan, in 1969, 1971, and 1975, respectively.

In 1974, he joined Central Research Laboratory, Hitachi, Ltd., Tokyo, Japan, where he is currently a Corporate Chief Scientist. At Central Research Laboratory, he has engaged in research and development work on character recognition and document understanding including mail-piece address recognition and forms processing, and document retrieval. From 1980 through 1981, he was a visiting scientist at the Computer Science Department, Carnegie Mellon University, Pittsburgh, PA. Besides working at Hitachi, he has been a visiting lecturer at Waseda University from 1985 to 1997, and at Kogakuin University, Tokyo, Japan, from 1998 to the present.

Dr. Fujisawa is a Fellow of the International Association for Pattern Recognition (IAPR), the Institute for Electronics, Information and Communication Engineers, Japan, and a Member of the Association for Computing Machinery (ACM), American Association for Artificial Intelligence (AAAI), and the Information Processing Society of Japan.