Study of a Fast Discriminative Training Algorithm for Pattern Recognition

Qi (Peter) Li, *Senior Member, IEEE*, and Biing-Hwang Juang, *Fellow, IEEE*

***Abstract*—Discriminative training refers to an approach to pattern recognition based on direct minimization of a cost function commensurate with the performance of the recognition system. This is in contrast to the procedure of probability distribution estimation as conventionally required in Bayes' formulation of the statistical pattern recognition problem. Currently, most discriminative training algorithms for nonlinear classifier designs are based on gradient-descent (GD) methods for cost minimization. These algorithms are easy to derive and effective in practice, but are slow in training speed and have difficulty selecting the learning rates. To address the problem, we present our study on a fast discriminative training algorithm. The algorithm initializes the parameters by the expectation–maximization (EM) algorithm, and then uses a set of closed-form formulas derived in this paper to further optimize a proposed objective of minimizing error rate. Experiments in speech applications show that the algorithm provides better recognition accuracy in a fewer iterations than the EM algorithm and a neural network trained by hundreds of GD iterations. Although some convergent properties need further research, the proposed objective and derived formulas can benefit further study of the problem.**

***Index Terms*—Classification, discriminative training, EM algorithm, neural networks, pattern recognition.**

I. Introduction

PATTERN recognition, one of the core techniques in computer applications, has played an important role in image and speech processing, communications, economics, biometrics, security, and biomedical research, to name a few. The mathematical construct of a pattern classifier can be linear, such as a single-layer perceptron, or nonlinear, such as a multilayer perceptron (MLP), a Gaussian mixture model (GMM), or a hidden Markov model (HMM) if the event to be recognized involves a sequentially varying phenomenon. A linear classifier uses a (set of) hyper-plan(s) to partition the data space [1], [2]. A nonlinear classifier uses nonlinear kernels to model the data distribution or the *a posteriori probability*, and may be better matched to the statistical behavior of the data than a linear classifier [3]. Parameters that define the classifier need to be "trained" or optimized based on a given set of known data.

General methodology for optimizing the classifier parameters falls into two broad classes: Distribution estimation and discriminative training. The distribution–estimation approach to classifier training is based on Bayes' decision theory, which suggests estimation of data distribution as the first and most imperative step in the design of a classifier. The most commonly used criterion for distribution estimation is maximum likelihood (ML) [3]. For those complex distributions used in many nonlinear classifiers, the re-estimation (or, equivalently, the EM) algorithm for ML estimation [4] is, in general, very efficient because, while it is a hill-climbing algorithm, it guarantees a net gain in the optimization objective at every iteration, leading to uniform, rather than stochastic, convergence to a fixed-point solution. The principle of discriminative training is to directly minimize a cost function commensurate with the performance of the pattern classification or recognition system. Cost functions proposed for discriminative training include the conventional squared-error used in the backpropagation algorithm [5]–[7], the minimum classification error (MCE) criterion in the generalized probabilistic descent (GPD) algorithm [8], [9], the maximum mutual information (MMI) criterion [10]–[13], and other versions [14]–[16]. Variations of the gradient-descent (GD) algorithm [8], [17], [18] as well as simulated and deterministic annealing algorithms [19] are used for optimizing these objectives. In general, these optimization algorithms converge slowly, particularly for large-scale or real-time problems such as automatic speech and speaker recognition, which employ HMMs with hundreds of GMMs and tens of thousands of parameters, hindering their applications.

In this paper, we present our study results in a fast and effective discriminative training algorithm for a class of nonlinear classifiers consisting of GMMs. The GMM is generally enough to model any distribution, and can be implemented as an artificial neural network (ANN) if needed. GMMs have been used in most speech and speaker recognition systems as the basic parameter model structure. We define a *generalized minimum error rate* (GMER) as the objective for discriminative training, and thus name the algorithm *fast GMER estimation*. It is a batch-mode approach using an approximate closed-form solution for optimization. The model parameters are first initialized by ML estimation, and then, the model is trained to optimize the GMER objective in a few iterations. The proposed procedure is similar to the re-estimation algorithm, but the aim is at minimum error rates (MERs) rather than ML. We demonstrate through experiments that the algorithm produces optimization results as fast as the re-estimation algorithm in terms of computational efficiency, and better than the ML estimation in terms of recognition accuracy. We also provide discussions and analysis on the convergent problem in this study.

II. A Fast Training Algorithm

In this section, we first define an objective, then derive the estimation formulas, and finally discuss the necessary and sufficient conditions for the optimization problem.

Manuscript received April 28, 2005; revised February 9, 2006.

Q. Li was with Bell Labs, Lucent Technologies. He is now with Li Creative Technologies, Inc., Florham Park, NJ 07932 USA (e-mail: qili@ieee.org).

B.-H. Juang was with AVAYA Labs Research. He is now with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: juang@ece.gatech.edu).

Color versions of Figs. 1–7 are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2006.875992

A. Objective for Generalized MER Estimation

In an M-class classification problem, we are asked to make a decision, to identify an observation $\mathbf{x}$, as a member of a class, say, C_i. The true identity of $\mathbf{x}$, say C_j, is not known, except in the design or training phase, in which observations of known identity are used as references for parameter optimization. We denote event α_i as the action of identifying an observation as class i. The decision is correct if $i = j$; otherwise, it is incorrect. It is natural to seek a decision rule that minimizes the probability of error, or, empirically, the error rate, which entails a zero-one loss function

$$\mathcal{L}(\alpha_i|C_j) = \begin{cases} 0, & i = j, \\ 1, & i \neq j \end{cases} \qquad i, j = 1, \ldots, M \,. \tag{1}$$

It assigns no loss to a correct decision, and assigns a unit loss to an error. The probabilistic risk of α_i corresponding to this loss function is

$$R(\alpha_i|\mathbf{x}) = \sum_{j=1}^{M} \mathcal{L}(\alpha_i|C_j) P(C_j|\mathbf{x}) = 1 - P(C_i|\mathbf{x}) \tag{2}$$

where $P(C_i|\mathbf{x})$ is the *a posteriori* probability that $\mathbf{x}$ belongs to C_i. To minimize the probability of error, one should, therefore, maximize the *a posteriori* probability $P(C_i|\mathbf{x})$. This is the basis of Bayes' maximum *a posteriori* (MAP) decision theory, and is also referred to as MER [3]. The *a posteriori* probability $P(C_i|\mathbf{x})$ is often modeled as $P_{\lambda_i}(C_i|\mathbf{x})$, a function defined by a set of parameters λ_i. Since the parameter set λ_i has a one-to-one correspondence with C_i, we write $P_{\lambda_i}(C_i|\mathbf{x}) = P(\lambda_i|\mathbf{x})$ and other similar expressions without ambiguity.

For training, we further define an "aggregate" *a posteriori* (AAP) probability for the set of design samples $\{\mathbf{x}_{m,n}; n = 1, 2, \ldots, N_m, m = 1, 2, \ldots, M\}$

$$\begin{aligned} J &= \frac{1}{M} \sum_{m=1}^{M} \sum_{n=1}^{N_m} P(\lambda_m|\mathbf{x}_{m,n}) \\ &= \frac{1}{M} \sum_{m=1}^{M} \sum_{n=1}^{N_m} \frac{p(\mathbf{x}_{m,n}|\lambda_m) P_m}{p(\mathbf{x}_{m,n})} \end{aligned} \tag{3}$$

where $\mathbf{x}_{m,n}$ is the nth training token from class m, N_m is the total number of tokens for class m, and P_m is the corresponding prior probability, respectively.

The aforementioned AAP objective for MAP or MERs can be further extended to a more general and flexible objective. We name it the GMER objective

$$\max \tilde{J} = \max \frac{1}{M} \sum_{m=1}^{M} \sum_{n=1}^{N_m} \ell(d_{m,n}) = \max \frac{1}{M} \sum_{m=1}^{M} \sum_{n=1}^{N_m} \ell_{m,n} \tag{4}$$

where

$$\ell(d_{m,n}) = \frac{1}{1 + e^{-\alpha d_{m,n}}} \tag{5}$$

is a sigmoid function, and

$$d_{m,n} = \log p(\mathbf{x}_{m,n}|\lambda_m) P_m - \log \sum_{j \neq m} p(\mathbf{x}_{m,n}|\lambda_j) P_j \tag{6}$$

represents a log probability ratio between the true class m and competing classes $j \neq m$. The sigmoid function can provide different weighting effects to different training data. For the data that has been well classified, weighting is close to 1 or 0; for the data near the classification boundary, weighting is near 0.5. The slope of the sigmoid function is controlled by the parameter α, where $\alpha > 0$. Thus, the values of α can affect the training performance and convergence. By adjusting the value for different recognition tasks, the GMER objective can provide better performance than the MER objective.

To ensure that the estimated covariance are positive–definite, we introduce a weighting scalar L_m into (6), thus

$$d_{m,n} = \log p(\mathbf{x}_{m,n}|\lambda_m) P_m - L_m \log \sum_{j \neq m} p(\mathbf{x}_{m,n}|\lambda_j) P_j \tag{7}$$

where $0 < L_m \leq 1$. For simplicity, we denote $L_m = L$. Intuitively, L represents the weighting between the true class m and competing classes $j \neq m$. When $L < 1$, it means that the true class is more important than the competing classes. When $L = 1$, it means the true class and competing classes are equally important. The range of the values of L can be determined during estimation. When $L = 1$ and $\alpha = 1$, we have $\tilde{J} = J$.

Although derived from different approaches, the GMER objective in (4) is equivalent to the empirical loss criterion defined in MCE [8]. Here, we use the GMER objective to facilitate the following derivation.

For testing, based on the Bayes decision rule to minimize risk and minimize average probability of error, for a given observation $\mathbf{x}$, we should select the action or class i that maximizes the posterior probability

$$i = \arg \max_{1 \leq m \leq M} \{P(\lambda_m|\mathbf{x})\} \,. \tag{8}$$

Since $P(\mathbf{x})$ in (3) is the same for all classes, we have

$$i = \arg \max_{1 \leq m \leq M} \{P(\mathbf{x}|\lambda_m) P_m\} \,. \tag{9}$$

Thus, although the training procedure can be different, the decision procedures are still the same for both ML estimation and discriminative estimations. The decision boundary between classes i and j is the line that satisfies the condition of $P(\mathbf{x}|\lambda_i) P_i = P(\mathbf{x}|\lambda_j) P_j$.

B. Derivation of Estimation Formulas

We now apply this formulation to a classifier design employing, specifically, the GMM as the conditional probability density function (pdf)

$$p(\mathbf{x}_{m,n}|\lambda_m) = \sum_{i=1}^{I} c_{m,i} p(\mathbf{x}_{m,n}|\lambda_{m,i}) \tag{10}$$

where $p(\mathbf{x}_{m,n}|\lambda_m)$ is mixture density, $p(\mathbf{x}_{m,n}|\lambda_{m,i})$ is component density, $c_{m,i}$ is mixing parameters subject to $\sum_i^I c_{m,i} = 1$, and I is the number of mixture components that constitute the conditional probability density. The parameters for the component density are a subset of the parameters of the mixture density, i.e., $\lambda_{m,i} \subset \lambda_m$. In most applications, the component density is defined as a Gaussian kernel

$$p(\mathbf{x}_{m,n}|\lambda_{m,i}) = \frac{1}{(2\pi)^{d/2}|\Sigma_{m,i}|^{1/2}} \times \exp\left(-\frac{1}{2}(\mathbf{x}_{m,n} - \mu_{m,i})^T \Sigma_{m,i}^{-1} (\mathbf{x}_{m,n} - \mu_{m,i})\right) \tag{11}$$

where $\mu_{m,i}$ and $\Sigma_{m,i}$ are, respectively, the mean vector and covariance matrix of the ith component of the mth GMM, d is the dimension of observation vectors, and T represents the vector or matrix transpose.

Let $\nabla_{\theta_{m,i}} J$ be the gradient of J with respect to $\theta_{m,i} \subset \lambda_{m,i}$. Making the gradient vanish for maximizing J, we have

$$\nabla_{\theta_{m,i}} \tilde{J} = \sum_{n=1}^{N_m} \omega_{m,i}(\mathbf{x}_{m,n}) \nabla_{\theta_{m,i}} \log p(\mathbf{x}_{m,n}|\lambda_{m,i}) - L \sum_{j \neq m} \sum_{\bar{n}=1}^{N_j} \bar{\omega}_{j,i}(\mathbf{x}_{j,\bar{n}}) \nabla_{\theta_{m,i}} \log p(\mathbf{x}_{j,\bar{n}}|\lambda_{m,i}) = 0 \tag{12}$$

where

$$\omega_{m,i}(\mathbf{x}_{m,n}) = \ell_{m,n}(1 - \ell_{m,n}) \frac{c_{m,i} p(\mathbf{x}_{m,n}|\lambda_{m,i}) P_m}{p(\mathbf{x}_{m,n}|\lambda_m) P_m} \tag{13}$$

$$\bar{\omega}_{j,i}(\mathbf{x}_{j,\bar{n}}) = \ell_{j,\bar{n}}(1 - \ell_{j,\bar{n}}) \frac{c_{m,i} p(\mathbf{x}_{j,\bar{n}}|\lambda_{m,i}) P_m}{\sum_{k \neq j} p(\mathbf{x}_{j,\bar{n}}|\lambda_k) P_k} \tag{14}$$

where ℓ is computed using (6) to represent the (unregulated) error rate. (This is deliberately set to separate, in concept, the influence of a token from the relative importance of various parameters of the classifier upon the performance of the classifier.) To find the solution to (12), we assume that $\omega_{m,i}$ and $\bar{\omega}_{j,i}$ can be approximated as constants around $\theta_{m,i}$. Discussion regarding this assumption is in Section II-C.

1) Estimation of Covariance Matrices: From the Gaussian component in (10), we have

$$\log p(\mathbf{x}_{m,n}|\lambda_{m,i}) = -\log\left[(2\pi)^{d/2}|\Sigma_{m,i}|^{1/2}\right] - \frac{1}{2}(\mathbf{x}_{m,n} - \mu_{m,i})^T \Sigma_{m,i}^{-1} (\mathbf{x}_{m,n} - \mu_{m,i}).$$

For optimization of the covariance matrix, we take the derivative with respect to matrix $\Sigma_{m,i}$

$$\nabla_{\Sigma_{m,i}} \log p(\mathbf{x}_{m,n}|\lambda_{m,i}) = -\frac{1}{2}\Sigma_{m,i}^{-1} + \frac{1}{2}\Sigma_{m,i}^{-1}(\mathbf{x}_{m,n} - \mu_{m,i})(\mathbf{x}_{m,n} - \mu_{m,i})^T \Sigma_{m,i}^{-1} \tag{15}$$

where ∇_Σ is defined as a matrix operator

$$\nabla_\Sigma \equiv \left[\frac{\partial}{\partial s_{i,j}}\right]_{i,j=1}^{d} \tag{16}$$

where $s_{i,j}$ is an entry of matrix Σ, and d is the dimension number of observation vectors. Bringing (15) into (12) and rearranging the terms, we have

$$\Sigma_{m,i} = \frac{\mathbf{A} - L\mathbf{B}}{D} \tag{17}$$

where

$$\mathbf{A} = \sum_{n=1}^{N_m} \omega_{m,i}(\mathbf{x}_{m,n})(\mathbf{x}_{m,n} - \mu_{m,i})(\mathbf{x}_{m,n} - \mu_{m,i})^T \tag{18}$$

$$\mathbf{B} = \sum_{j \neq m} \sum_{\bar{n}=1}^{N_j} \bar{\omega}_{j,i}(\mathbf{x}_{j,\bar{n}})(\mathbf{x}_{j,\bar{n}} - \mu_{m,i})(\mathbf{x}_{j,\bar{n}} - \mu_{m,i})^T \tag{19}$$

and

$$D = \sum_{n=1}^{N_m} \omega_{m,i}(\mathbf{x}_{m,n}) - L \sum_{j \neq m} \sum_{\bar{n}=1}^{N_j} \bar{\omega}_{j,i}(\mathbf{x}_{j,\bar{n}}). \tag{20}$$

Both $\mathbf{A}$ and $\mathbf{B}$ are matrices and D is a scalar. For simplicity, we ignore subscripts m, i for $\mathbf{A}$, $\mathbf{B}$, and D.

2) Determination of Weighting Scalar: The estimated covariance matrix $\Sigma_{m,i}$ must be positive–definite. We use this requirement to determine the upper bound of the weighting scalar L.

Using the eigenvectors of $\mathbf{A}^{-1}\mathbf{B}$, we can construct an orthogonal matrix $\mathbf{U}$, such that 1) $\mathbf{A} - L\mathbf{B} = \mathbf{U}^T(\tilde{\mathbf{A}} - L\tilde{\mathbf{B}})\mathbf{U}$, where both $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$ are diagonal, and 2) both $\mathbf{A} - L\mathbf{B}$ and $\tilde{\mathbf{A}} - L\tilde{\mathbf{B}}$ have the same eigenvalues. These claims have been proved in Theorems 1 and 2 in the Appendix. L can then be determined as

$$L < \min\left\{\frac{\tilde{a}_k}{\tilde{b}_k}\right\}_{k=1}^{d} \tag{21}$$

where $\tilde{a}_i > 0$ and $\tilde{b}_i > 0$ are the diagonal entries of $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$, respectively. L also needs to satisfy

$$D(L) > 0 \quad \text{and} \quad 0 < L \leq 1. \tag{22}$$

Thus, for the ith mixture component of model m, we can determine $L_{m,i}$. If model m has I mixtures, we need to determine

one L_m to satisfy all mixture components in the model. Therefore, the upper bound of L_m is

$$L_m \leq \min\{L_{m,1}, L_{m,2}, \ldots, L_{m,i}\}. \tag{23}$$

In numerical computation, we need an exact number of L; therefore, we have

$$L_m = \eta \min\{L_{m,1}, L_{m,2}, \ldots, L_{m,i}\} \tag{24}$$

where $0 < \eta \leq 1$ is a preselected constant, and it is much easier to determine compared to the learning rate in GD algorithms.

3) Estimation of Mean Vectors: We take the derivative of (15) with respect to vector $\mu_{m,i}$

$$\nabla_{\mu_{m,i}} \log p(\mathbf{x}_{m,n}|\lambda_{m,i}) = \Sigma_{m,i}^{-1}(\mathbf{x}_{m,n} - \mu_{m,i}) \tag{25}$$

where ∇_μ is defined as a vector operator

$$\nabla_\mu \equiv \left[\frac{\partial}{\partial \nu_i}\right]_{i=1}^{d} \tag{26}$$

where ν_i is an entry of vector μ and d is the dimension number of observation vectors. Bringing (25) into (12) and rearranging the terms, we obtain the solution for mean vectors

$$\mu_{m,i} = \frac{\mathbf{E} - L\mathbf{F}}{D} \tag{27}$$

where

$$\mathbf{E} = \sum_{n=1}^{N_m} \omega_{m,i}(\mathbf{x}_{m,n})\mathbf{x}_{m,n} \tag{28}$$

$$\mathbf{F} = \sum_{j\neq m} \sum_{\bar{n}=1}^{N_j} \bar{\omega}_{j,i}(\mathbf{x}_{j,\bar{n}})\mathbf{x}_{j,\bar{n}} \tag{29}$$

and D is defined in (20). Again, for simplicity, we ignore subscripts m, i for $\mathbf{E}$, $\mathbf{F}$, and D. We note that the both $\mathbf{E}$ and $\mathbf{F}$ are vectors, and scalar L has been determined when estimating $\Sigma_{m,i}$.

4) Estimation of Mixture Parameters: The last step is to compute the mixture parameters $c_{m,i}$ subject to $\sum_i^I c_{m,i} = 1$. Introducing Lagrangian multipliers γ_m, we have

$$\hat{J} = \tilde{J} + \sum_{m=1}^{M} \gamma_m \left(\sum_{i=1}^{I_m} c_{m,i} - 1\right). \tag{30}$$

Taking the first derivative and making it vanish for maximization, we have

$$\frac{\partial \tilde{J}}{\partial c_{m,i}} = \frac{1}{c_{m,i}} D + \gamma_m = 0. \tag{31}$$

Rearranging the terms, we have

$$c_{m,i} = -\frac{1}{\gamma_m} D. \tag{32}$$

Summing over $c_{m,i}$, for $i = 1 \ldots I$, we can solve γ_m as

$$\gamma_m = -(G - LH) \tag{33}$$

where

$$G = \sum_{n=1}^{N_m} \sum_{i=1}^{I_m} \omega_{m,i}(c_i, \mathbf{x}_{m,n}) \tag{34}$$

and

$$H = \sum_{j\neq m} \sum_{\bar{n}=1}^{N_j} \sum_{i=1}^{I_j} \bar{\omega}_{j,i}(c_i, \mathbf{x}_{j,\bar{n}}). \tag{35}$$

Bringing (33) into (32), we have

$$c_{m,i} = \frac{D}{G - LH}. \tag{36}$$

C. Discussion on Sufficient Conditions

So far, we have discussed the necessary conditions for optimization, i.e., $\nabla_{\theta_{m,i}} J = 0$. In theory, we also need to meet the following sufficient conditions:

1) $\nabla^2_{\theta_{m,i}} J < 0$, in order to ensure a maximum solution;
2) $|\nabla_{\theta_{m,i}} \omega_{m,i}| \approx 0$ and $|\nabla_{\theta_{m,i}} \bar{\omega}_{j,i}| \approx 0$ around $\theta_{m,i}$.

This is to ensure that $\omega_{m,i}(\theta_{m,i})$ and $\bar{\omega}_{j,i}(\theta_{m,i})$ in (12) are approximately constant; therefore, the independent assumption is sound.

The first-order derivatives of $\omega_{m,i}$ and $\bar{\omega}_{m,i}$ are as follows:

$$\begin{aligned}\nabla_{\theta_{m,i}} \omega_{m,i}(\mathbf{x}_{m,n}) = {} & \ell_{m,n}(1-\ell_{m,n}) \frac{c_{m,i} p(\mathbf{x}_{m,n}|\lambda_{m,i})}{p(\mathbf{x}_{m,n}|\lambda_m)^2} \\ & \times [(1-2\ell_{m,n}) c_{m,i} p(\mathbf{x}_{m,n}|\lambda_{m,i}) P_m \\ & + p(\mathbf{x}_{m,n}|\lambda_m) - c_{m,i} p(\mathbf{x}_{m,n}|\lambda_{m,i})] \\ & \times \nabla_{\theta_{m,i}} \log p(\mathbf{x}_{m,n}|\lambda_{m,i})\end{aligned} \tag{37}$$

and

$$\begin{aligned}&\nabla_{\theta_{m,i}} \bar{\omega}_{j,\bar{n}}(\mathbf{x}_{j,\bar{n}}) \\ &\quad = \ell_{j,\bar{n}}(1-\ell_{j,\bar{n}}) c_{m,i} p(\mathbf{x}_{j,\bar{n}}|\lambda_{m,i}) P_m \\ &\qquad \times \left\{ \frac{(1-2\ell_{j,\bar{n}})}{p(\mathbf{x}_{j,\bar{n}}|\lambda_m)} \right. \\ &\qquad\quad \left. + \frac{\sum_{k\neq j} p(\mathbf{x}_{j,\bar{n}}|\lambda_k) P_k - c_{m,i} p(\mathbf{x}_{j,\bar{n}}|\lambda_{m,i}) P_m}{\left[\sum_{k\neq j} p(\mathbf{x}_{j,\bar{n}}|\lambda_k) P_k\right]^2} \right\} \\ &\qquad \times \nabla_{\theta_{m,i}} \log p(\mathbf{x}_{j,\bar{n}}|\lambda_{m,i}).\end{aligned} \tag{38}$$

When the conditions in (37) and (38) cannot be satisfied in the beginning, an iteration procedure is then necessary to approach the condition approximately. The goal is to have the values of both equations converge to zeros through iterations by optimizing the model parameters and selecting the weighting parameter L. This needs further analysis in convergent theory.

In practice, since we have to compute $\ell_{m,n}$ of the objective function anyway while computing (13) and (14), we can evaluate the objective function directly for the purpose of evaluating convergence. If the value of the objective function is improved, we keep the new set of parameters; otherwise, we do not save them and just go to the next iteration.

D. Practical Training Procedure

The practical training procedure for the proposed fast GMER estimation can be summarized as follows:

1) initialize all models parameters for all classes by ML estimation;
2) for every mixture component i in model m, compute $\omega_{m,i}$ and $\bar{\omega}_{m,i}$ using (13) and (14), and compute $\mathbf{A}$, $\mathbf{B}$, and D, using (18)–(20);
3) determine the weighting scalar L by (24);
4) for every mixture component i, compute $\Sigma_{m,i}$, $\mu_{m,i}$, and $c_{m,i}$ using (17), (27), and (36);
5) evaluate the performance using (4) and (6) for model m; if the performance is improved, save the best model parameters;
6) repeat steps 2)–6) for the required number of iterations for model m;
7) use the saved model for class m and repeat the aforementioned procedure for all untrained models;
8) output the saved models for testing and applications.

E. Extension to Sequentially Observed Feature Vectors

In the aforementioned derivations, one observation or training token $\mathbf{x}$ represents one feature vector, and a decision is made based on the single feature vector. The result can be applied directly to a class of the applications of involving "fixed dimensional" pattern recognition, such as target recognition, and image recognition. In many applications, however, one observation or one token may consists of a sequence of observations, and a decision is made based on the sequence of feature vectors, such as in speech and speaker recognition, and also in video recognition. For example, in speech recognition, one spoken phoneme can be represented by several feature vectors, and a short sentence can have over one hundred feature vectors. For such applications, decisions are usually made on a sequence of continually observed (extracted) feature vectors, and we have to make corresponding changes in the objective and estimation formulas accordingly.

For the nth observation of class m with a sequence of feature vectors, the observation (token) can be presented as

$$\mathbf{X}_{m,n} = \{\mathbf{x}_{m,n,q}\}_{q=1}^{Q_n} \tag{39}$$

where the nth token has a sequence of Q_n feature vectors. To deal with this kind of a problem, one usually assumes that the variable to represent the vectors are independent, identically-distributed (i.i.d); therefore, the probability or likelihood $p(\mathbf{X}_{m,n}|\lambda_m)$ can be calculated as

$$p(\mathbf{X}_{m,n}|\lambda_m) = \prod_{q=1}^{Q_n} p(\mathbf{x}_{m,n,q}|\lambda_m). \tag{40}$$

Thus, the GMER objective can be rewritten as

$$\max \tilde{J} = \frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{N_m} \ell(d_{m,n}) = \frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{N_m} \ell_{m,n} \tag{41}$$

where $\ell(d_{m,n}) = 1/(1+e^{-\alpha d_{m,n}})$ is a sigmoid function, and

$$d_{m,n} = \log p(\mathbf{X}_{m,n}|\lambda_m)P_m - \log \sum_{j\neq m} p(\mathbf{X}_{m,n}|\lambda_j)P_j \tag{42}$$

represents a log probability ratio between the true class m and the competing classes $j \neq m$. Using the same method demonstrated previously, readers can derive a set of re-estimation formulas, or refer to the results in [20]. We ignore the derivations here since the procedures and results are very similar to the previous derivations.

Once models are trained, given the nth observation: A sequence of feature vectors $\mathbf{X}_n$, the decision on the observation can be made as

$$i = \arg \max_{1\leq m\leq M} \{P(\mathbf{X}_n|\lambda_m)P_m\} \tag{43}$$

$$= \arg \max_{1\leq m\leq M} \left\{P_m \prod_{q=1}^{Q_n} p(\mathbf{x}_{n,q}|\lambda_m)\right\}. \tag{44}$$

In practice, the aforementioned computation is often conducted as log likelihood

$$i = \arg \max_{1\leq m\leq M} \left\{\log P_m + \sum_{q=1}^{Q_n} \log p(\mathbf{x}_{n,q}|\lambda_m)\right\}. \tag{45}$$

We note that if the variable to represent the vectors is not i.i.d. but a nonstationary process, HMM [21] should be used as the model structure, and the aforementioned algorithm can be further extended to train HMMs. Actually, the sequential algorithm is equivalent to train one HMM state. Details of HMM training are out of the scope of this paper.

III. Experiments

The proposed fast GMER estimation has been applied to several pattern recognition experiments. We first use an illustrative example to present the concept of discriminative training,

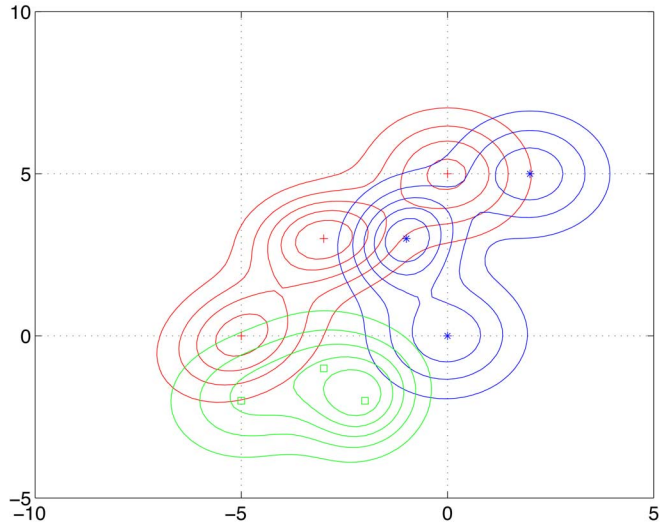

Fig. 1. Contours of the *pdf*'s of three-mixture GMMs: The models are used to generate three classes of training data.

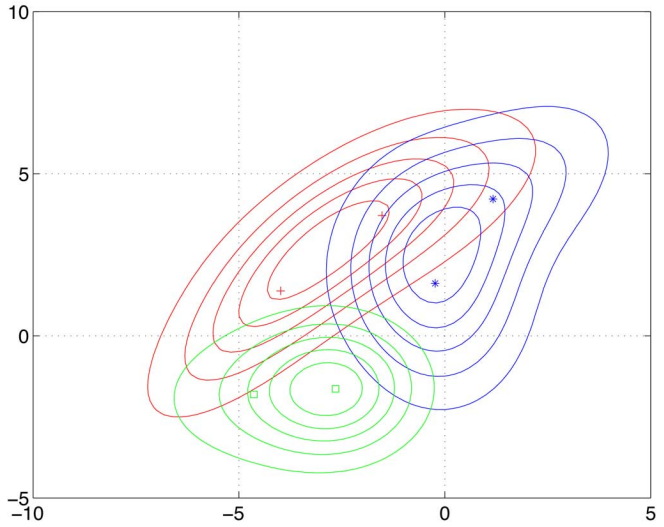

Fig. 2. Contours of the *pdf*'s of two-mixture GMMs: The models are from ML estimation using four iterations.

followed by two applications on English vowel recognition and speaker identification.

A. *Illustrative Example*

In this example, we artificially generated three classes of two-dimensional (2–D) data. The distributions were of Gaussian-mixture types, with three components in each class. Each token was of two dimensions. For each class, 1500 tokens were drawn from each of the three components; therefore, there were 4500 tokens in total. The contours of the ideal distributions of classes 1, 2, and 3 are shown in Fig. 1, where the means are represented as $+$, $*$, and boxes, respectively.

To simulate real applications, we assumed that the number of mixture components is unknown. Therefore, we assumed the GMMs, which need to be trained, have two mixture components with full covariance matrices for each class. In the first step, ML estimation was applied to initialize the GMMs with four iterations based on the training data drawn from the ideal models. The contours that represent the *pdf*'s of each of the GMMs after ML estimation are plotted in Fig. 2.

We then used the proposed fast GMER estimation to further train new GMMs with two iterations based on the parameters estimated by ML estimation. The contours representing the *pdf*'s of each of the new GMMs after the GMER estimation are plotted in Fig. 3. All the contours in Figs. 2 and 3 are plotted on the same scale.

By comparing Figs. 2 and 3, we can observe that GMER training significantly reduced the overlaps amount three classes. The decision boundaries of the three cases were plotted in Fig. 4. After GMER training, the boundaries from ML estimation shifted toward the decision boundaries from the ideal models. We note that both the ML and GMER models were trained from a limited set of training data drawn from the ideal model, and the shifted areas are with high data density. This illustrated how GMER training improves classification accuracies.

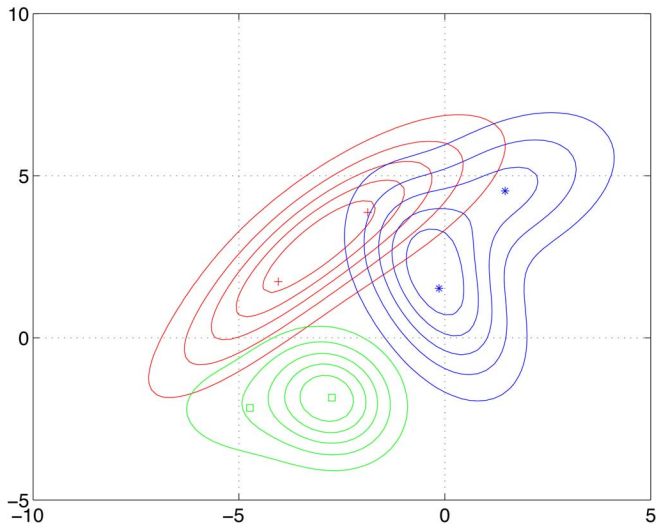

Fig. 3. Contours of the *pdf*'s of two-mixture GMMs: The models are from the proposed fast GMER estimation with two iterations on top of the ML estimation results. The overlaps among the three classes are significantly reduced.

The testing data with 4500 tokens for each class were obtained using the same methods as the training data. The ML estimation provided an accuracy of 76.07% and 75.97% for training and testing datasets, respectively, while the proposed GMER estimation improved the accuracy to 76.26% and 76.70% after two iterations. The control parameters were set to $\eta = 0.5$ and $\alpha = 0.01$. If we use the same model that generated the training data to do the testing, the ideal performances are 77.19% and 77.02% for training and testing. These ideal cases are the ceilings of this example. To evaluate the behaviors of the GMER estimation, we plot the training and testing data on each of the

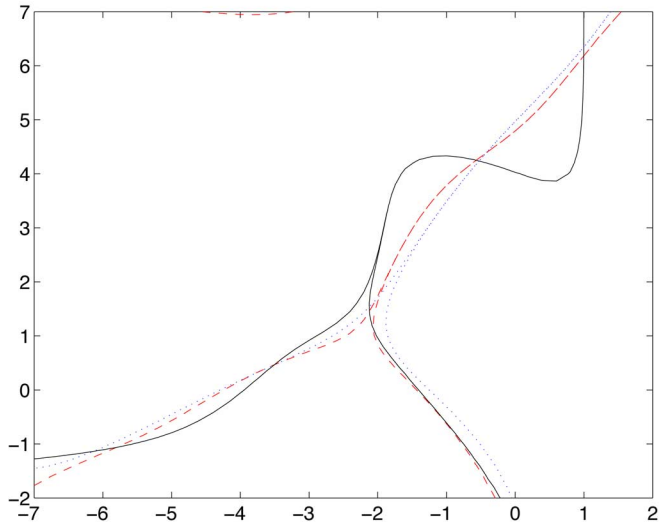

Fig. 4. Enlarged decision boundaries for the ideal three-mixture models (solid line), two-mixture ML models (dashed line), and two-mixture GMER models (dash-dotted line): After GMER training, the boundary of ML estimation shifted toward the decision boundary of the ideal models. This illustrates how GMER training improves decision accuracies.

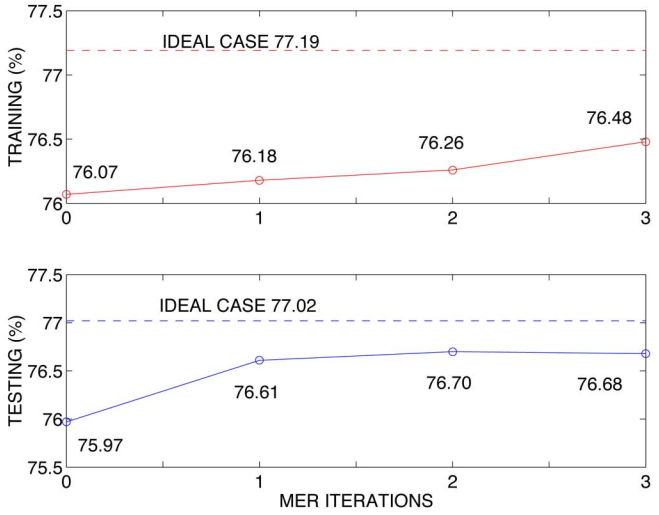

Fig. 5. Performance improvement versus iterations using the GMER estimation: The initial performances were from the ML estimation with four iterations.

iterations in Fig. 5. On testing, the relative improvement of the GMER estimation against the ceiling is significant.

B. English Vowel Recognition

In this experiment, GMMs were applied as classifiers to recognize English vowels based on single-feature vectors. The feature data were generated from a telephone speech database. First, the speech was sampled at an 8-KHz sampling rate; then, the fast Fourier transform (FFT) was applied to the speech data over a 30-ms window shifted every 10 ms through the recorded speech. Each FFT spectrum was processed by an auditory-based algorithm [22], and then converted to 12 cepstral coefficients through discrete cosine transform. The average speech energy of the 30-ms window was also included in the

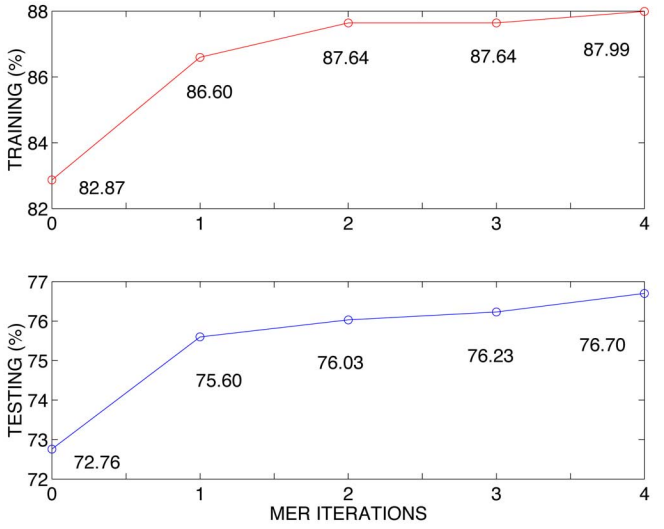

Fig. 6. English vowel classification: Performances of the proposed GMER algorithm in four iterations. The accuracy is measured in terms of the percentage of correctly-recognized feature vectors.

feature set. These 13-dimensional (13-D) feature vectors were further augmented by a set of 13-feature coefficients of the first derivative calculated over a 5-frame window, plus another set of 13 coefficients of the second derivative calculated over a 9-frame window. Thus, for every 10 ms, we had a frame of 39-D feature vectors for classification. This is the same feature we used for speech recognition. The feature vectors corresponding to each vowel were partitioned into two datasets for training and testing. Depending on the availability, each vowel has about 800 feature vectors for training and 100 feature vector for testing.

We used one GMM with eight mixtures and diagonal covariance matrices to represent each vowel. The GMMs were first initialized by ML estimation; each model was trained in 4 EM iterations using the data that belong to the corresponding class. In the next step, the GMMs were further trained discriminatively using the proposed GMER estimation in four iterations. For model m, the parameters corresponding to the best accuracy on the training dataset were saved as λ_m. We then proceeded to train model λ_{m+1} in four 4 iterations, and so on. The results of various numbers of iterations are plotted in Fig. 6.

For comparison, a GD method was also evaluated, where the model parameter θ was updated as

$$\theta_{t+1} = \theta_t + \epsilon \nabla_{\theta_{m,i}} J|_{\theta_t} \tag{46}$$

where t denotes the tth iteration, $\epsilon > 0$ represents the learning rate, and $\nabla_{\theta_{m,i}} J|_{\theta_t}$ was computed by a formula similar to (12). We note that this GD algorithm is very similar to GPD training. The transforms defined in [23] were employed in this experiment to ensure the positive–definite of the variance matrices and to meet other constraints. The models were updated using the same procedure as aforementioned.

For comparison and validation, we further employed MLP neural networks to solve the same problem. Instead of eight

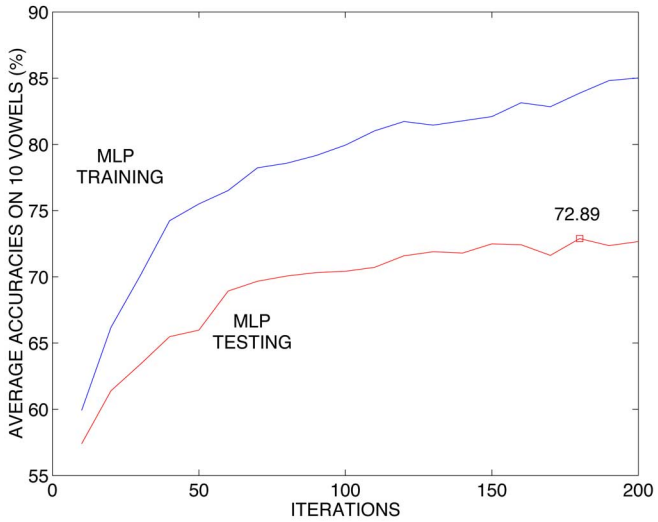

Fig. 7. Average accuracy of MLP networks in the same vowel classification problem: The MLP networks were trained by the conjugate-gradient training method. After 200 iterations, the best average accuracy on the testing dataset was 72.89%, which is worse than the result of 76.70% from the proposed GMER algorithm.

TABLE I
PERFORMANCE COMPARISON OF ENGLISH VOWEL RECOGNITION

Algorithms	Iterations	Average Accuracy (%)
MLP Neural Net	MLP200	72.89
MLE	MLE4	72.76
MLE + GMER/GD	MLE4 + GD4	73.06
MLE + GMER/Proposed	MLE4 + GMER4	76.70

mixtures, we used eight hidden nodes. Therefore, the total number of parameters of each model for each class was almost the same, except that the nonlinear function was changed from a Gaussian to a sigmoid function. For training, we compared several fast training algorithms [17], such as conjugate gradients, batch gradient descent, and quasi-Newton algorithms. We plotted the results from the conjugate algorithm because it is faster than others for this task. As shown in Fig. 7, after 200 iterations, the best result from the testing dataset was 72.89%, which was still lower than 76.70% obtained by the proposed algorithm in four ML plus four MER iterations.

The final comparison among MLP, MLE, GD, and proposed GMER algorithms is listed in Table I. The MLP training was better than MLE because it is a discriminative training algorithm. The GD algorithm based on the GMER objective provided better performance than MLP with much less iteration. The proposed GMER algorithm provided the best performance while using only four iterations from the MLE-trained models.

C. Speaker Identification

In this experiment, we used a text-independent speaker identification task to evaluate the proposed GMER algorithm on sequentially-observed feature vectors as discussed in Section II-E. The experiment included 11 speakers. Given a sequence of feature vectors extracted from a speakers voice, the task was to identify the true speaker from the group of 11 speakers. Each speaker had 60 s of training data and 30–40 s of testing data with a sampling rate of 8 KHz. These speakers were randomly picked from the 2000 National Institute of Standards and Technology (NIST) Speaker Recognition Evaluation Database. The speech data were first converted into 12-dimensional (12-D) Mel-frequency cepstral coefficients (MFCC) feature through a 30-ms window shifted every 10 ms [24]. Thus, for every 10 ms, we had one 12-D MFCC feature vector. The silence frames were then removed by a batch-mode endpoint detection algorithm [25]. The testing performance was evaluated based on segments of 1, 5, and 10 s of testing speech. The speech segment was constructed by moving a window of the length of 10, 50, or 100 vectors at every feature vector on the testing data collected sequentially. A detailed introduction to speaker identification and typical ML estimation approach can be found in [24].

TABLE II
SPEAKER IDENTIFICATION ERROR RATES ON DIFFERENT ALGORITHMS AND TESTING DATA

Algorithms	Iterations	Test Length		
		1 sec	5 sec	10 sec
ML Estimation	5	31.41%	6.59%	1.39%
GMER (Propossed)	MLE5 + MER1	26.81%	2.21%	0.00%
Relative Error Reduction		14.65%	66.46%	100.00%

We first constructed GMMs with eight-mixture components for every speaker using the ML estimation. Each GMM was then further trained discriminatively using the sequential GMER estimation discussed in Section II-E. During the test, for every segment, we computed the likelihood scores of all trained GMMs in the selected test length. The speaker with the highest score was labeled as the owner of the segment.

The experimental results are listed in Table II. For 1, 5, and 10 s of testing data, the proposed sequential GMER algorithm had 14.56%, 66.46%, and 100.00% relative error rate reductions, respectively, compared to ML estimation, which is the most popular algorithm in speaker identification.

IV. CONCLUSION AND DISCUSSION

In this study, we first defined a discriminative training objective called GMER, and then derived the formulas for a fast training algorithm for GMER using iterations. The EM algorithm was used to initialize the parameters. Our experimental results indicated that the fast GMER algorithm is efficient and effective.

Based on an approximation, the algorithm attempts to iteratively improve the parameter optimization for classifier and recognition solutions. Although the algorithm does not guarantee analytical convergence at each iteration, we empirically demonstrate that the proposed algorithm can train a classifier or recognizer in only a few iterations, much faster than GD-based methods, while also providing better recognition accuracy due to the generalization of the principle of error minimization.

We initialized the discussions of the convergent problem, but we have to note that the problem has not been thoroughly analyzed yet, and further study is still necessary. Essentially, it is a new kind of problem, and the traditional methods for convergent analysis, such as contraction mapping [26] and the general convergence theorem [27], cannot be applied here directly. We have

to look for new approaches to analyze the convergent properties of the proposed algorithm. Nevertheless, following our initial analysis, the experimental results were promising and showed that the proposed algorithm does converge to a local maximum for the overall performances, although it does not guarantee convergence on every step. This indicates that the assumptions of this study are reasonable, and our initial analysis can lead to further studies on the convergence theory.

The GMER estimation presented in this paper has two sets of parameter re-estimation formulas. One is for classification based on single vectors, i.e., a decision is made based on the probability of every feature vector. Another one is for pattern recognition based on a sequence of feature vectors, i.e., a decision is made based on the likelihood of a sequence of feature vectors.

The goal of our further study is to investigate a new mathematical method to examine the convergent properties and to extend the new GMER estimation method to HMM for speech recognition and other dynamic pattern recognition problems.

APPENDIX

Theorem 1: Two symmetric matrices $\mathbf{A}$ and $\mathbf{B}$ can be diagonalized simultaneously as

$$\mathbf{U}^T\mathbf{A}\mathbf{U} = I \tag{47}$$

and

$$\mathbf{U}^T\mathbf{B}\mathbf{U} = \Lambda \tag{48}$$

where Λ and $\mathbf{U}$ are the eigenvalue and eigenvector matrices of $\mathbf{A}^{-1}\mathbf{B}$, i.e., $\mathbf{A}^{-1}\mathbf{B} = \mathbf{U}\Lambda\mathbf{U}^{-1}$.

Proof: See [28, p. 32]. ■

Theorem 2: If $\mathbf{A}-L\mathbf{B} = \mathbf{U}^T(\tilde{\mathbf{A}}-L\tilde{\mathbf{B}})\mathbf{U}$, then $\mathbf{A}-L\mathbf{B}$ and $\tilde{\mathbf{A}}-L\tilde{\mathbf{B}}$ have the same eigenvalues with the same multiplicities.

Proof: The proof is straightforward. Let $\Psi = \mathbf{A}-L\mathbf{B}$ and $\Phi = \tilde{\mathbf{A}} - L\tilde{\mathbf{B}}$. The eigenvalues of Ψ are the roots of

$$\begin{aligned}\det(\Phi - \lambda\mathbf{I}) &= \det\left(\mathbf{U}^T\Psi\mathbf{U} - \lambda\mathbf{I}\right)\\ &= \det\left(\mathbf{U}^{-1}(\Psi - \lambda\mathbf{I})\mathbf{U}\right)\\ &= \det\mathbf{U}^{-1}\det(\Psi - \lambda\mathbf{I})\det\mathbf{U}\\ &= \det(\Psi - \lambda\mathbf{I})\end{aligned} \tag{49}$$

where, λ is eigenvalue and $\mathbf{U}$ is an orthogonal matrix, so $\mathbf{U}^{-1} = \mathbf{U}^T$. Since Φ and Ψ have the same characteristic polynomial, they have the same eigenvalues. ■

REFERENCES

[1] Q. Li and D. W. Tufts, "Principal feature classification," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 155–160, Jan. 1997.

[2] J. Peltonen and S. Kaski, "Discriminative components of data," *IEEE Trans. Neural Netw.*, vol. 16, no. 1, pp. 68–83, Jan. 2005.

[3] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2001.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Stat. Soc.*, vol. 39, no. 1, pp. 1–38, 1977.

[5] P. J. Werbos, *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting*. New York: Wiley, 1994.

[6] X. Yu, M. O. Efe, and O. Kaynak, "A general backpropagation algorithm for feedforward neural networks learning," *IEEE Trans. Neural Netw.*, vol. 13, no. 1, pp. 251–254, Jan. 2002.

[7] W. Wu, G. Feng, Z. Li, and Y. Xu, "Deterministic convergence of an online gradient method for BP networks," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 533–540, May 2005.

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

[9] W. Chou, "Discriminant-function-based minimum recognition error rate pattern-recognition approach to speech recognition," *Proc. IEEE*, vol. 88, no. 8, pp. 1201–1222, Aug. 2000.

[10] L. R. Bahl, P. F. Brown, P. V. de Souza, and R. L. Mercer, "Maximum mutual information estimation of hidden Markov model parameters for speech recognition," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Tokyo, Japan, 1986, pp. 49–52.

[11] P. S. Gopalakrishnan, D. Kanevsky, A. Nadas, and D. Nahamoo, "An inequality for rational functions with applications to some statistical estimation problems," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 107–113, Jan. 1991.

[12] Y. Normandin, R. Cardin, and R. D. Mori, "High-performance connected digit recognition using maximum mutual information estimation," *IEEE Trans. Speech Audio Process.*, vol. 2, no. 4, pp. 299–311, Apr. 1994.

[13] A. Ben-Yishai and D. Burshtein, "A discriminative training algorithm for hidden Markov models," *IEEE Trans. Speech Audio Process.*, vol. 12, no. 3, pp. 204–217, May 2004.

[14] K. Markov and S. Nakagawa, "Discriminative training of GMM using a modified EM algorithm for speaker recognition," in *Proc. Int. Conf. Spoken Language Processing (ICSLP)*, 1998, pp. 177–180.

[15] K. Markov, S. Nakagawa, and S. Nakamura, "Discriminative training of HMM using maximum normalized likelihood algorithm," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, 2001, pp. 497–500.

[16] I. Mora-Jimenez and J. Cid-Sueiro, "A universal learning rule that minimize well-formed cost functinos," *IEEE Trans. Neural Netw.*, vol. 16, no. 4, pp. 810–820, Jul. 2005.

[17] C. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford, 1995.

[18] M. Robinson, M. R. Azimi-Sadjadi, and J. Salazar, "Multi-aspect target discrimination using hidden Markov models and neural networks," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 447–459, Mar. 2005.

[19] S. Kirkpatrick, J. C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[20] Q. Li and B.-H. Juang, "Fast discriminative training for sequential observations with application to speaker identification," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Hong Kong, Apr. 2003.

[21] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[22] Q. Li, F. K. Soong, and S. Olivier, "An auditory system-based feature for robust speech recognition," in *Proc. 7th Euro. Conf. Speech Communication Technology*, Denmark, Sep. 2001, pp. 619–622.

[23] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 3, pp. 257–265, May 1997.

[24] D. Reynolds and R. C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. Speech Audio Process.*, vol. 3, no. 1, pp. 72–83, Jan. 1995.

[25] Q. Li, J. Zheng, A. Tsai, and Q. Zhou, "Robust endpoint detection and energy normalization for real-time speech and speaker recognition," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 3, pp. 146–157, Mar. 2002.

[26] D. R. Smart, *Fixed Point Theory*. Cambridge, U.K.: Cambridge Univ. Press, 1974.

[27] W. I. Zangwill, *Nonlinear Programming a Unified Approach*. Englewood Cliffs, NJ: Prentice-Hall, 1969.

[28] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd ed. New York: Academic, 1990.

Qi (Peter) Li (S'87–M'88–SM'01) received the Ph.D. degree in electrical engineering from the University of Rhode Island, Kingston, in 1995.

From 1988 to 1994, he was with F.M. Engineering and Research, Norwood, MA, where he engaged in pattern recognition, real-time systems, and networks. In 1991, he attended Harvard University to study statistical theory and methods. From 1995 to 2002, he was with Bell Laboratories, Lucent Technologies, Murray Hill, NJ, as a Member of Technical Staff in the Multimedia Communications Research Lab, where his research was in speech and speaker recognition, biometric authentication, front-end signal processing, speech modeling, and multimedia systems; his research results were implemented in Lucent products. In 2002, he established Li Creative Technologies (LcT), Inc., Florham Park, NJ. LcT is a high-tech company in R&D for signal processing, multimedia applications, and homeland security. LcT has been awarded several research contracts by the U.S. government agencies, DARPA, NAVY, ARMY, and MDA, respectively. The company is currently conducting research in acoustic-signal processing and microphone array for handheld devices and computers, speaker recognition, new features for speech processing, data-driven prognostics, dynamic pattern recognition, etc. The company is also developing commercial products based on its new research results. In 2006, LcT received the Best Consumer Technology/Electronics Company Award issued by the New Jersey Technology Council at *NJ Venture Capital Conference*. He currently holds several issued patents and has filed more than one dozen patents. He has published extensively in peer-reviewed journals and conferences.

Dr. Li has been active as a reviewer for several journals, IEEE publications, and conferences. He has also been a Local Chair for the IEEE Workshop on Automatic Identification and a committee member for several IEEE conferences and workshops. He received the best paper award, an achievement award, and several Bell Labs patent awards. He is listed in *Who's Who in America* (Millennium and 2001 Editions) and *Who's Who in Executives and Professionals* (2004 Edition). In 2004, he received the *Success Award* issued by the New Jersey Small Business Development Center.

Biing-Hwang Juang (S'79–M'80–SM'87–F'91) received the Ph.D. degree in electrical engineering and computer science from the University of California, Santa Barbara, in 1981.

He worked at Speech Communications Research Laboratory (SCRL) and Signal Technology, Inc. (STI) on a number of government-sponsored research projects. Notable accomplishments during the period include development of vector quantization for voice applications, voice coders at extremely low bit rates, 800 bps and around 300 bps, and robust vocoders for use in satellite communications. He was also a co-Principal Investigator for the project on co-channel separation of speech signals. He subsequently joined the Acoustics Research Department of Bell Laboratories, working in the area of speech enhancement, coding and recognition. He became Department Head/Director of Acoustics and Speech Research at Bell Labs, in 1996, and Director of Multimedia Technologies Research at Avaya Labs (a spinoff of Bell Labs), in 2001. In the past few years, he and his group developed a speech server for applications such as AT&T's advanced 800 calls and the Moviefone, the Perceptual Audio Coder (PAC) for digital audio broadcasting in North America (in both terrestrial and satellite systems), and a world-first realtime full-duplex hands-free stereo teleconferencing system. He joined the Georgia Institute of Technology as Motorola Foundation Chair Professor in the School of Electrical and Computer Engineering, in 2002. He is also an Eminent Scholar of Georgia Research Alliance of the state of Georgia. He has published extensively, including the book *Fundamentals of Speech Recognition* (Prentice-Hall: NJ) coauthored with L.R. Rabiner, and holds about 20 patents.

Prof. Juang has served as Editor-in-Chief for the IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, and has held a number of positions in the IEEE Signal Processing Society, including the current Chair of its Fellow Reference Committee. He is currently on the IEEE Proceedings Editorial Board. He has received a number of technical awards, notably among which are several Best Paper awards in the area of speech communications and processing, the Technical Achievement Award from the Signal Processing Society, and the IEEE Third Millennium Medal. He is a Fellow of Bell Laboratories, and a member of the National Academy of Engineering of the United States.