FAST DISCRIMINATIVE TRAINING FOR SEQUENTIAL OBSERVATIONS WITH APPLICATION TO SPEAKER IDENTIFICATION

Qi Li

225 Runnymede Parkway
New Providence, NJ 07974, USA
qili@ieee.org; www.lilabs.com

Biing-Hwang Juang

School of Electrical & Computer Engineering
Georgia Institute of Technology
juang@ece.gatech.edu

ABSTRACT

This paper presents a fast discriminative training algorithm for sequences of observations. It considers a sequence of feature vectors as one single composite token in training or testing. In contrast to the traditional EM algorithm, this algorithm is derived from a discriminative objective, aiming at directly minimizing the recognition error. Compared to the gradient-descent algorithms for discriminative training, this algorithm invokes a mild assumption which leads to closed-form formulas for re-estimation, rather than relying on gradient search, without sacrificing the algorithmic rigor. As such, it is in general much faster than a descent based algorithm and does not need to determine the learning rate or step size. Our experiment shows that the proposed algorithm reduces error rate by 14.65, 66.46, and 100.00% for 1, 5, and 10 seconds of testing data respectively, in a speaker identification application.

1. INTRODUCTION

Pattern recognition is one of the core techniques for computer applications. It constructs mathematical pattern classifiers using pre-collected training data. Current approaches to pattern classifier design fall into two categories: (1) the distribution-estimation approach based on Bayses' decision theory; and (2) the discriminative training approach based on minimizing the classification error rate [1]. Since the second approach aims directly at minimizing the error rate, it usually provides better performance compared to the more traditional distribution estimation approach. In term of training algorithms, the first approach uses the EM algorithm for maximum likelihood estimation of the data distributions. It is usually very efficient even though it is an iterative procedure. For speech recognition, it often takes only a few iterations for the solution to converge. The second approach uses gradient-descent algorithms, which usually need more care in numeric treatments, need to properly determine the learning rate or step size for parameter update, and take more iterations during optimization. To address the training problem in the second approach, we proposed a new discriminative training algorithm called the fast MER estimation in [2], where the results showed that the new algorithm can provide better performance with much faster training compared to a conventional gradient-descent based algorithm for discriminative training.

In our previous paper [2], the fast MER algorithm is a fixed-dimensional algorithm, i.e. we assume that one training or testing token is one observation or a single feature vector. It is for general pattern recognition applications. In this paper, we extended the algorithm to a sequence-based algorithm, i.e. we assume that one training or testing token is an observation sequence or a set of feature vectors. A natural application for this extended algorithm is speaker identification, which is important today for security applications. We then verify the proposed algorithm on the performance of a speaker identification task.

2. OBJECTIVE FUNCTION

In an M-class classification problem, we are asked to make a decision to identify a sequence of observations, $\mathbf{X}$, as 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 reference 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 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:

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

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 minimum error rate (MER) [3] in an ideal setup. The *a posteriori* probability $P(C_i|\mathbf{X})$ is often modelled 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:

$$J = \frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{N_m}\frac{p(\mathbf{X}_{m,n}|\lambda_m)P_m}{p(\mathbf{X}_{m,n})} \tag{3}$$

where $\mathbf{X}_{m,n}$ is the n'th training token from class m, M is the total number of classes, N_m is the total number of tokens for class m, and P_m is the corresponding prior probability. Let us assume: (1) one token, $\mathbf{X}_{m,n}$, consists of a sequence of observations (or frames): $\mathbf{X}_{m,n} = \{\mathbf{x}_{m,n,q}\}_{q=1}^{Q_n}$, where Q_n is the total number of the observations or frames in the n'th token; and (2) the observations are independent, identically distributed (i.i.d.). Thus, probability or likelihood $p(\mathbf{X}_{m,n}|\lambda_m)$ is calculated as: $p(\mathbf{X}_{m,n}|\lambda_m) = \prod_{q=1}^{Q_n} p(\mathbf{x}_{m,n,q}|\lambda_m)$. Note that the MAP objective can be rewritten for the AAP probability as:

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

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

$$d_{m,n} = \log p(\mathbf{X}_{m,n}|\lambda_m)P_m - \log\sum_{j\neq m} p(\mathbf{X}_{m,n}|\lambda_j)P_j \tag{5}$$

represents a log probability ratio between the true class m and competing classes $j \neq m$. The sigmoid function provides different weights to different training data. For those data that are hardly ambiguous in their classification, the weight is close to 1 (i.e., decisively wrong) or 0 (i.e., decisively correct); for those data near the classification boundary, the weighting is in-between. 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. The value needs to be pre-selected for different tasks as in other discriminative training algorithms.

For numeric consistency, we introduce a weighting scalar L_m into (5). Thus, we have $d_{m,n} = \log p(\mathbf{X}_{m,n}|\lambda_m)P_m - L_m \log\sum_{j\neq m} p(\mathbf{X}_{m,n}|\lambda_j)P_j$, where $0 < L_m \leq 1$. For simplicity, we denote $L_m = L$. Intuitively, L represents the weighting between 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 $\tilde{J} = J$.

3. SEQUENCE-BASED ESTIMATION FORMULAS

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

$$p(\mathbf{x}_{m,n,q}|\lambda_m) = \sum_{i=1}^{I} c_{m,i}p(\mathbf{x}_{m,n,q}|\lambda_{m,i}) \tag{6}$$

where $p(\mathbf{x}_{m,n,q}|\lambda_m)$ is a mixture density, $p(\mathbf{x}_{m,n,q}|\lambda_{m,i})$ is a component density, $c_{m,i}$ is the mixing parameter 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 is 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 an Gaussian kernel.

Let $\nabla_{\theta_{m,i}} J$ be the gradient of J with respect to $\theta_{m,i} \subset \lambda_{m,i}$. Vanishing the gradient to maximize J, we have:

$$\begin{aligned}\nabla_{\theta_{m,i}}\tilde{J} = &\sum_{n=1}^{N_m}\sum_{q=1}^{Q_n}\Omega_{m,i}(\mathbf{x}_{m,n,q})\nabla_{\theta_{m,i}}\log p(\mathbf{x}_{m,n,q}|\lambda_{m,i})\\ &-L\sum_{j\neq m}\sum_{\bar{n}=1}^{N_j}\sum_{\bar{q}=1}^{Q_n}\bar{\Omega}_{j,i}(\mathbf{x}_{j,\bar{n},\bar{q}})\nabla_{\theta_{m,i}}\log p(\mathbf{x}_{j,\bar{n},\bar{q}}|\lambda_{m,i})\\ &= 0\end{aligned} \tag{7}$$

$$\Omega_{m,i}(\mathbf{x}_{m,n,q}) = \ell_{m,n}(1-\ell_{m,n})\frac{c_{m,i}p(\mathbf{x}_{m,n,q}|\lambda_{m,i})}{p(\mathbf{x}_{m,n,q}|\lambda_m)} \tag{8}$$

$$\bar{\Omega}_{j,i}(\mathbf{x}_{j,\bar{n},\bar{q}}) = \ell_{j,\bar{n}}(1-\ell_{j,\bar{n},\bar{q}})\frac{c_{m,i}p(\mathbf{x}_{j,\bar{n},\bar{q}}|\lambda_{m,i})P_m}{\sum_{k\neq j}p(\mathbf{x}_{j,\bar{n},\bar{q}}|\lambda_k)P_k} \tag{9}$$

where ℓ is computed using (5) 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 (7), we assume that $\Omega_{m,i}$ and $\bar{\Omega}_{j,i}$ can be approximated as constants.

1) Estimation of Covariance Matrices: For a Gaussian component, we have

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

where $\mu_{m,i}$ and $\Sigma_{m,i}$ are the mean vector and covariance matrix of the i'th component of the m'th GMM. d is the dimension of observation vectors, and T represents the vector or matrix transpose.

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

$$\nabla_{\Sigma_{m,i}} \log p(\mathbf{x}_{m,n,q}|\lambda_{m,i}) = -\frac{1}{2}\Sigma_{m,i}^{-1} + \frac{1}{2}\Sigma_{m,i}^{-1}(\mathbf{x}_{m,n,q} - \mu_{m,i})(\mathbf{x}_{m,n,q} - \mu_{m,i})^T\Sigma_{m,i}^{-1} \tag{10}$$

where ∇_Σ is defined as a matrix operator $\nabla_\Sigma \equiv \left[\frac{\partial}{\partial s_{i,j}}\right]_{i,j=1}^{d}$, where $s_{i,j}$ is an entry of matrix Σ, and d is the dimension number of observation vectors. Bringing (10) into (7) and rearranging the terms, we have:

$$\Sigma_{m,i} = \frac{\mathbf{A} - L\mathbf{B}}{D} \tag{11}$$

$$\mathbf{A} = \sum_{n=1}^{N_m}\sum_{q=1}^{Q_n}\Omega_{m,i}(\mathbf{x}_{m,n,q})(\mathbf{x}_{m,n,q}-\mu_{m,i})(\mathbf{x}_{m,n,q}-\mu_{m,i})^T \tag{12}$$

$$\mathbf{B} = \sum_{j\neq m}\sum_{\bar{n}=1}^{N_j}\sum_{\bar{q}=1}^{Q_n}\bar{\Omega}_{j,i}(\mathbf{x}_{j,\bar{n},\bar{q}})(\mathbf{x}_{j,\bar{n},\bar{q}}-\mu_{m,i})(\mathbf{x}_{j,\bar{n},\bar{q}}-\mu_{m,i})^T \tag{13}$$

$$D = \sum_{n=1}^{N_m}\sum_{q=1}^{Q_n}\Omega_{m,i}(\mathbf{x}_{m,n,q}) - L\sum_{j\neq m}\sum_{\bar{n}=1}^{N_j}\sum_{\bar{q}=1}^{Q_{\bar{n}}}\bar{\Omega}_{j,i}(\mathbf{x}_{j,\bar{n},\bar{q}}). \tag{14}$$

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. We have proved these claims in [4]. L can then be determined as:

$$L < \min\left\{\frac{\tilde{a}_k}{\tilde{b}_k}\right\}_{k=1}^{d}, \tag{15}$$

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 $D(L) > 0$ and $0 < L \leq 1$. Thus, for the i'th 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 $L_m \leq \min\{L_{m,1}, L_{m,2}, \ldots, L_{m,I}\}$. In numerical computation, we need an exact number of L; therefore, we have

$$L_m = \eta \min\{L_{m,1}, L_{m,2}, \ldots, L_{m,I}\}, \tag{16}$$

where $0 < \eta \leq 1$ is a pre-selected constant and it is easier to determine compared to the learning rate in gradient-descent algorithms.

3) Estimation of Mean Vectors: We take the derivative of (10) with respect to vector $\mu_{m,i}$:

$$\nabla_{\mu_{m,i}} \log p(\mathbf{x}_{m,n,q}|\lambda_{m,i}) = \Sigma_{m,i}^{-1}(\mathbf{x}_{m,n,q} - \mu_{m,i}) \tag{17}$$

where ∇_μ is defined as a vector operator $\nabla_\mu \equiv \left[\frac{\partial}{\partial \nu_i}\right]_{i=1}^{d}$, where ν_i is an entry of vector μ, and d is the dimension number of observation vectors. Bringing (17) into (7) and rearranging the terms, we obtain the solution for mean vectors:

$$\mu_{m,i} = \frac{\mathbf{E} - L\mathbf{F}}{D} \tag{18}$$

$$\mathbf{E} = \sum_{n=1}^{N_m}\sum_{q=1}^{Q_n}\Omega_{m,i}(\mathbf{x}_{m,n,q})\mathbf{x}_{m,n,q} \tag{19}$$

$$\mathbf{F} = \sum_{j\neq m}\sum_{\bar{n}=1}^{N_j}\sum_{\bar{q}=1}^{Q_{\bar{n}}}\bar{\Omega}_{j,i}(\mathbf{x}_{j,\bar{n},\bar{q}})\mathbf{x}_{j,\bar{n},\bar{q}} \tag{20}$$

and D is defined in (14). 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 for the mixture parameters $c_{m,i}$ subject to $\sum_i^I c_{m,i} = 1$. Introducing Lagrangian multipliers γ_m, we have

$$\hat{J} = \tilde{J} + \sum_{m=1}^{M}\gamma_m\left(\sum_{i=1}^{I_m} c_{m,i} - 1\right). \tag{21}$$

Taking the first derivative and vanishing it for maximization, we get

$$\frac{\partial \tilde{J}}{\partial c_{m,i}}\frac{1}{c_{m,i}}D + \gamma_m = 0. \tag{22}$$

Rearranging the terms, we then have

$$c_{m,i} = -\frac{1}{\gamma_m}D. \tag{23}$$

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

$$\gamma_m = -(G - LH) \tag{24}$$

$$G = \sum_{n=1}^{N_m}\sum_{q=1}^{Q_n}\sum_{i=1}^{I_m}\Omega_{m,i}(c_i, \mathbf{x}_{m,n,q}) \tag{25}$$

$$H = \sum_{j\neq m}\sum_{\bar{n}=1}^{N_j}\sum_{\bar{q}=1}^{Q_{\bar{n}}}\sum_{i=1}^{I_j}\bar{\Omega}_{j,i}(c_i, \mathbf{x}_{j,\bar{n}}). \tag{26}$$

Bring (24) into (23), we have

$$c_{m,i} = \frac{D}{G - LH}. \tag{27}$$

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

1. $\nabla^2_{\theta_{m,i}} J < 0$. This is to ensure an 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 (7) are approximately constant so as to support the closed-form re-estimation formulas. While theoretical proof of this assumption is not available, we validate the algorithm through experiments and use it in applications.

4. TRAINING PROCEDURE

The training procedure of the proposed fast MER 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 (8) and (9), and compute $\mathbf{A}$, $\mathbf{B}$, and D using (12), (13), and (14);

3. Determine the weighting scalar L by (16);

4. For every mixture component i, compute $\Sigma_{m,i}$ $\mu_{m,i}$, and $c_{m,i}$ using (11), (18), and (27);

5. Evaluate the performance using (4) and (5) for model m. If the performance is improved, save the best model parameters;

6. Repeat Step 2 to 5 for the required number of iterations for model m;

7. Use the saved model for class m and repeat the above procedure for all untrained models.

8. Output the saved models for testing and applications.

5. EXPERIMENTS ON SPEAKER IDENTIFICATION

We applied the proposed string-based MER estimation to a text-independent speaker identification task. There were 11 speakers in a group. Each speaker had 60 seconds of training data and 30 - 40 seconds of testing data . The speakers were randomly picked up from the 2000 NIST Speaker Recognition Evaluation database. The speech data were first converted into 12-dimensional (12-D) Mel-frequency cepstral coefficients through a 30 ms window shifted every 10 ms. Thus, for every 10 ms, we have one 12-D MFCC feature frame. The silence frames were then removed by a batch-mode endpoint detection algorithm [5]. The testing performance is evaluated based on segments of 1, 5, and 10 seconds of testing speech. The speech segment was constructed by moving a window of the length of 10, 50, or 100 frames at every frame on the testing data. A detailed introduction to speaker identification and typical ML estimation approach can be found in [6].

Table 1. 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
String-Based MER (Proposed)	MLE5 + MER1	26.81	2.21	0.00
Error Reduction		14.65	66.46	100.00

We first constructed GMM with 8-mixture components for every speaker using the ML estimation. Each GMM was then further trained discriminatively using the proposed sequence-based MER estimation. During test, for every segment, we computed the likelihood scores of all trained GMM's. The speaker with the highest score was labelled as the owner of the segment.

The experimental results are listed in Table 1. For 1, 5, and 10 seconds of testing data, the proposed string-based MER algorithm made 14.56%, 66.46%, and 100.00% relative error rate reduction compared to the ML estimation.

6. CONCLUSIONS

We extended the frame-based, fast MER estimation to a sequence-based one. It showed significant improvement in text-independent speaker identification application compared the traditional ML estimation. It will be straightforward to extend this string-based MER algorithm to train hidden Markov models for speech recognition.

7. REFERENCES

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

[2] Q. Li and B.-H. Juang, "A new algorithm for fast discriminative training," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Orlando, FL, May 2002.

[3] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification, Second Edition*, John & Wiley, New York, 2001.

[4] Q. Li and B.-H. Juang, "A new discriminative training algorithm for pattern recognition," *IEEE Trans. on Speech and Audio Processing*, to be submitted.

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

[6] D. Reynolds and R. C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. on Speech and Audio Processing*, vol. 3, no. 1, pp. 72–83, 1993.