Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Pattern Recognition Letters xxx (2007) xxx–xxx
Pattern Recognition Letters
www.elsevier.com/locate/patrec

A new look at discriminative training for hidden Markov models

Xiaodong He *, Li Deng *

Microsoft Research, 1 Microsoft Way, Redmond, WA 98052, United States

Abstract

Discriminative training for hidden Markov models (HMMs) has been a central theme in speech recognition research for many years. One most popular technique is minimum classification error (MCE) training, with the objective function closely related to the empirical error rate and with the optimization method based traditionally on gradient descent. In this paper, we provide a new look at the MCE technique in two ways. First, we develop a non-trivial framework in which the MCE objective function is re-formulated as a rational function for multiple sentence-level training tokens. Second, using this novel re-formulation, we develop a new optimization method for discriminatively estimating HMM parameters based on growth transformation or extended Baum–Welch algorithm. Technical details are given for the use of lattices as a rich representation of competing candidates for the MCE training.

Keywords: Hidden Markov model; Discriminative learning; Minimum classification error; Extended Baum–Welch algorithm; Growth transformation

1. Introduction

Hidden Markov models (HMMs) have been a well established framework for a variety of pattern recognition applications, including, most prominently, speech recognition applications (Rabiner and Juang, 1993; Bahl et al., 1987; Deng and O'Shaughnessy, 2003). One most attractive feature of the HMM framework is that its parameters can be learned automatically from the training data. In early days of HMMs, the parameters were learned by the maximum likelihood (ML) criterion based on the EM algorithm (e.g., Bahl et al., 1987; Rabiner and Juang, 1993). Improvement of parameter learning beyond ML has been pursued for many years (Brown, 1987; Chou, 2003; Deng et al., 2005; Deng et al., 2005; Gopalakrishnan et al., 1991; He and Chou, 2003; Juang and Katagiri, 1992; Juang et al., 1997; Macherey et al., 2005; McDermott et al., in press; Normandin, 1991; Povey and Woodland, 2002; Povey et al., 2003; Povey, 2004; Povey et al., 2004; Rathinavalu and Deng, 1998; Schluter et al., 2001), based on the concept of discrimination against classes, in contrast to maximizing likelihood of each individual class. The reason behind discriminative training is that complete knowledge of speech data distributions is lacking and training data is always limited. It is not until recently that discriminative training has shown uniform success in speech recognition over virtually all tasks, including especially large tasks (e.g., Woodland and Povey, 2000; Povey, 2004).

Among several types of discriminative training for HMMs, one prominent type is minimum classification error (MCE) training (Chou, 2003; Juang and Katagiri, 1992; Juang et al., 1997; He and Chou, 2003; Macherey et al., 2005; McDermott et al., in press; Roux and McDermott, 2005; Rathinavalu and Deng, 1998). The essence of MCE is to define the objective function for optimization that is closely related to the empirical classification errors. This is more desirable than other types of discriminative training that are less closely related to the classification errors. The conventional MCE has been based on the sequential gradient-descent based technique, named generalized probabilistic descent (GPD), which optimizes the objective function as a highly complex function of the HMM parameters.

* Corresponding authors. Tel.: +1 425 706 4939 (Xiaodong He); +1 425 706 2719 (Li Deng); fax: +1 425 936 7329.
E-mail addresses: xiaohe@microsoft.com (X. He), deng@microsoft.com (L. Deng).

0167-8655/$ - see front matter
doi:10.1016/j.patrec.2006.11.022

Another significant advance in discriminative training is the development and application of a special type of optimization technique, called growth transformation (GT) or extended Baum–Welch (EBW) algorithm when it is used for HMM parameter estimation. GT is an iterative optimization scheme where if the parameter set Λ is subject to a transformation $\Lambda = T(\Lambda')$, then the objective function "grows" in its value $O(\Lambda) > O(\Lambda')$ unless $\Lambda = \Lambda'$. In (Gopalakrishnan et al., 1991), GT/EBW was developed for rational functions such as the mutual information as the optimization criterion. Maximization of mutual information (MMI) as a form of discriminative criterion for the discrete HMM was described in (Gopalakrishnan et al., 1991). This has been extended to the continuous-density HMM in (Normandin, 1991; Gunawardana and Byrne, 2001). The significance of GT/EBW lies in its effectiveness and closed-form parameter updating for large-scale optimization problems with difficult objective functions. Compared with the gradient based techniques which often require special and delicate care for tuning the parameter-dependent learning rate, GT/EBW avoids such requirements and with the closed-form updating formula it is generally faster in reaching algorithm convergence.

Mutual information is naturally in the form of a rational function and MMI is obviously suited to GT/EBW optimization. However, as a discriminative criterion, it is only indirectly related to classification errors. On the other hand, MCE as a discriminative criterion is closely related to classification errors, but it is not naturally in the form of a rational function when there are multiple utterance tokens in the training data. Hence, it has been a tradition to use the gradient-descent techniques (GPD) for optimizing the MCE criterion (Chou, 2003; Juang et al., 1997; McDermott et al., in press; Rathinavalu and Deng, 1998). In this paper, we break this long-held tradition and take a fresh look at the MCE. This new analysis and formulation of the MCE covers two main issues. First we re-examine the MCE criterion. Second the results of the re-examination permit the use of the new GT/EBW optimization technique for optimizing the MCE criterion with respect to the HMM parameters.

The organization of this paper is as follows. In Section 2, an overview of the traditional MCE is provided. Then, in Section 3, we reformulate the MCE criterion (with multiple training tokens) into a rational functional form. We provide a rigorous proof by induction for the correctness of the rational functional form. Given this non-trivial reformulation, in Section 4, we present in detail a novel GT/EBW based optimization technique for estimating the parameters of the Gaussian HMMs. In Section 5, the lattice-based MCE training is described, and a summary is given in Section 6.

2. Overview of minimum classification error (MCE) training

We denote by Λ the parameter set of the generative model expressed in terms of a joint statistical distribution

$$p_\Lambda(X,S) = p_\Lambda(X|S)P(S), \tag{1}$$

on the observation training data sequence X and on the corresponding label sequence S, where we assume the parameters in the "language model" $P(S)$ are not subject to optimization. We use $r = 1,\ldots,R$ as the index for "token" (e.g., a single sentence or utterance) in the training data, and each token consists of a "string" of an observation data sequence: $X_r = x_{r,1},\ldots,x_{r,Tr}$, with the corresponding label (e.g., word) sequence: $S_r = w_{r,1},\ldots,w_{r,Nr}$. That is, S_r denotes correct label sequence for token r. Further, we use s_r to denote all possible label sequences for the rth token, including the correct label sequence S_r and all other incorrect label sequences.

MCE learning was originally introduced for multiple-category classification problems where the smoothed error rate is minimized for isolated "tokens" (Juang and Katagiri, 1992). It was later generalized to minimize the smoothed "sentence token" or string-level error rate (Juang et al., 1997; Chou, 2003), which is known as "embedded MCE".

The MCE objective function is defined first based on a set of class discriminant functions and a special type of loss function. Then the model is estimated to minimize the expected loss that is closely related to the recognition error rate of the classifier.

In embedded MCE training, a set of discriminant functions is first defined based on the correct string S_r and the N most confusable competing strings, $s_{r,1},\ldots,s_{r,N}$. Define the top N best competing strings as

$$\begin{aligned} s_{r,1} &= \arg\max_{s_r: s_r \neq S_r}\{\log p_\Lambda(X_r, s_r)\}, \\ s_{r,i} &= \arg\max_{s_r: s_r \neq S_r, s_r \neq s_{r,1},\ldots,s_{r,i-1}}\{\log p_\Lambda(X_r, s_r)\} \quad i = 2,\ldots,N. \end{aligned}$$

Then, the discriminant functions for the correct string and the N competing strings take the form of

$$g_{s_r}(X_r;\Lambda) = \log p_\Lambda(X_r, s_r), \quad s_r \in \{S_r, s_{r,1},\ldots,s_{r,N}\}.$$

And the decision rule for the recognizer or classifier is the one that for the observation data sequence, X_r,

$$C(X_r) = s_r^* \quad \text{if } s_r^* = \arg\max_{s_r} g_{s_r}(X_r;\Lambda).$$

Next, a misclassification measure in MCE is defined. For the general N-best MCE training, the following misclassification measure has been widely used (Juang et al., 1997):

$$d_r(X_r,\Lambda) = -\log p_\Lambda(X_r, S_r) + \log\left\{\frac{1}{N}\sum_{s_r, s_r \neq S_r} \exp\left[\eta \log p_\Lambda(X_r, s_r)\right]\right\}^{\frac{1}{\eta}}. \tag{2}$$

This misclassification measure function emulates the decision rule, i.e., $d_r(X_r,\Lambda) \geqslant 0$ implies misclassification and $d_r(X_r,\Lambda) < 0$ implies a correct classification. The second term in (2) is a soft-max function, which counts the scores of all N competitive candidates. It can be looked as an average over the scores of competitive candidates weighted based on their individual significance. Moreover, this misclassification measure can be closely approximated by the following simpler form:

$$d_r(X_r, \Lambda) = -\log p_\Lambda(X_r, S_r) + \log \sum_{s_r, s_r \neq S_r} w(s_r) \cdot p_\Lambda(X_r, s_r), \tag{3}$$

where $w(s_r)$ is a non-negative weighting factor for competitive string s_r. Note that the sum of $w(s_r)$ is not necessarily equal to one.

Finally, to define the objective function of MCE, a loss function for a single sentence token or string X_r is established, as originally proposed in (Juang and Katagiri, 1992; Juang et al., 1997), in the following form:

$$l_r(d_r(X_r, \Lambda)) = \frac{1}{1 + e^{-\alpha d_r(X_r, \Lambda) + \beta}} = \frac{1}{1 + e^{-d_r(X_r, \Lambda)}}, \tag{4}$$

where we assume $\alpha = 1$, $\beta = 0$ for simplicity in exposition without loss of generality. This loss function emulates the zero-one recognition error count function, i.e., when $d_r(X_r, \Lambda)$ is larger than zero, which implies an incorrect recognition, the loss function approaches to one, which essentially becomes a recognition error count.

With the misclassification measure in the form of (3), the loss function for the N-best version of MCE becomes:

$$\begin{aligned} l_r(d_r(X_r, \Lambda)) &= \frac{\sum_{s_r, s_r \neq S_r} w(s_r) p_\Lambda(X_r, s_r)}{\sum_{s_r, s_r \neq S_r} w(s_r) p_\Lambda(X_r, s_r) + p_\Lambda(X_r, S_r)} \\ &= \frac{\sum_{s_r, s_r \neq S_r} w(s_r) p_\Lambda(X_r, s_r)}{\sum_{s_r} w(s_r) p_\Lambda(X_r, s_r)}. \end{aligned} \tag{5}$$

The last step is obtained after the assignment of $w(S_r) \equiv 1$ for the correct string S_r.

Given the loss function for one sentence token r in (5), the empirical loss function over the whole training set with all R training tokens becomes:

$$L(\Lambda) = \sum_{r=1}^{R} l_r(d_r(X_r, \Lambda)). \tag{6}$$

Therefore, (6) is closely related to the empirical recognition error rate and is the objective function to minimize in MCE. The traditional MCE methods minimize the loss function via the technique of probabilistic gradient descent or GPD, which we refer the readers to an excellent review in (Chou, 2003).

3. A new look at MCE – optimization criterion

We now take a new look at MCE in terms of its optimization criterion as expressed in (6). Minimizing the overall loss function of $L(\Lambda)$ in (6) is to the same as maximizing the following equivalent objective function:

$$\begin{aligned} O(\Lambda) = R - L(\Lambda) &= \sum_{r=1}^{R} \left[1 - \frac{\sum_{s_r, s_r \neq S_r} w(s_r) p_\Lambda(X_r, s_r)}{\sum_{s_r} w(s_r) p_\Lambda(X_r, s_r)} \right] \\ &= \sum_{r=1}^{R} \frac{w(S_r) p_\Lambda(X_r, S_r)}{\sum_{s_r} w(s_r) p_\Lambda(X_r, s_r)}. \end{aligned} \tag{7}$$

Importantly, (7) is a sum of rational functions rather than a rational function in itself, and hence it would not be directly amenable to GT/EBW for its optimization. The difficulty of formulating a rational function and the desire of moving away from gradient descent have been discussed in (Povey, 2004). In this section, we directly tackle this difficulty and re-formulate the MCE objective function of (7) as a true rational function of the following specific form:

$$O(\Lambda) = \frac{\sum_{s_1 \ldots s_R} w(s_1 \ldots s_R) p_\Lambda(X_1 \ldots X_R, s_1 \ldots s_R) C(s_1 \ldots s_R)}{\sum_{s_1 \ldots s_R} w(s_1 \ldots s_R) p_\Lambda(X_1 \ldots X_R, s_1 \ldots s_R)}, \tag{8}$$

where $w(s_1 \ldots s_R) = \prod_{r=1}^{R} w(s_r)$and $C(s_1 \ldots s_R) = \sum_{r=1}^{R} C(s_r)$, $C(s_r) = \delta(s_r, S_r)$. Here, $\delta(s_r, S_r)$ is the Kronecker delta function that equals one if $s_r = S_r$, and zero otherwise. Note that $w(s_1, \ldots, s_R)$ and $C(s_1, \ldots, s_R)$ are quantities not relevant to Λ. In (8), $X = X_1, \ldots, X_R$ denotes the collection of all observation data sequences in all R training tokens, and $p_\Lambda(X_1, \ldots, X_R, s_1, \ldots, s_R)$ is the joint distribution for all training data and their corresponding label sequence assignments $s = s_1, \ldots, s_R$.

We now provide a rigorous proof that (7) and (8) are equivalent. We use the induction method for the proof in the following two steps.

(1) We prove the equivalence of (7) and (8) when there are two training utterances, or $R = 2$, as follows. Starting from (7), we have:

$$\begin{aligned} O(\Lambda) &= \frac{w(S_1) p_\Lambda(X_1, S_1)}{\sum_{s_1} w(s_1) p_\Lambda(X_1, s_1)} + \frac{w(S_2) p_\Lambda(X_2, S_2)}{\sum_{s_2} w(s_2) p_\Lambda(X_2, s_2)} \\ &= \frac{\sum_{s_1} w(s_1) p_\Lambda(X_1, s_1) \delta(s_1, S_1)}{\sum_{s_1} w(s_1) p_\Lambda(X_1, s_1)} \\ &\quad + \frac{\sum_{s_2} w(s_2) p_\Lambda(X_2, s_2) \delta(s_2, S_2)}{\sum_{s_2} w(s_2) p_\Lambda(X_2, s_2)} \\ &= \frac{\sum_{s_1} \sum_{s_2} w(s_1) w(s_2) p_\Lambda(X_1, s_1) p_\Lambda(X_2, s_2) [\delta(s_1, S_1) + \delta(s_2, S_2)]}{\sum_{s_1} \sum_{s_2} w(s_1) w(s_2) p_\Lambda(X_1, s_1) p_\Lambda(X_2, s_2)} \\ &= \frac{\sum_{s_1 s_2} w(s_1 s_2) p_\Lambda(X_1, X_2, s_1, s_2) [C(s_1 s_2)]}{\sum_{s_1 s_2} w(s_1 s_2) p_\Lambda(X_1, X_2, s_1, s_2)}. \end{aligned} \tag{9}$$

The last step used the common assumption that the training tokens are independent of each other. Clearly (9), is in the same form of (8) when $R = 2$.

(2) After assuming the equivalence of (7) and (8) for $R = R_0$, we now prove the equivalence for $R = R_0 + 1$ as follows. Again, starting from (7) for $R = R_0 + 1$,we have,

$$
\begin{aligned}
\sum_{r=1}^{R_0+1}\frac{w(S_r)p_\Lambda(X_r,S_r)}{\sum_{s_r}w(s_r)p_\Lambda(X_r,s_r)} &= \sum_{r=1}^{R_0}\frac{w(S_r)p_\Lambda(X_r,S_r)}{\sum_{s_r}w(s_r)p_\Lambda(X_r,s_r)}+\frac{w(S_{R_0+1})p_\Lambda(X_{R_0+1},S_{R_0+1})}{\sum_{s_{R_0+1}}w(s_{R_0+1})p_\Lambda(X_{R_0+1},s_{R_0+1})}\\
&= \frac{\sum_{s_1\ldots s_{R_0}}w(s_1\ldots s_{R_0})p_\Lambda(X_1\ldots X_{R_0},s_1\ldots s_{R_0})C(s_1\ldots s_{R_0})}{\sum_{s_1\ldots s_{R_0}}w(s_1\ldots s_{R_0})p_\Lambda(X_1\ldots X_{R_0},s_1\ldots s_{R_0})}\\
&\quad+\frac{\sum_{s_{R_0+1}}w(s_{R_0+1})p_\Lambda(X_{R_0+1},s_{R_0+1})\delta(s_{R_0+1},S_{R_0+1})}{\sum_{s_{R_0+1}}w(s_{R_0+1})p_\Lambda(X_{R_0+1},s_{R_0+1})}\\
&= \frac{\sum_{s_1\ldots s_{R_0}}\sum_{s_{R_0+1}}w(s_{R_0+1})p_\Lambda(X_{R_0+1},s_{R_0+1})w(s_1\ldots s_{R_0})p_\Lambda(X_1\ldots X_{R_0},s_1\ldots s_{R_0})[C(s_1\ldots s_{R_0})+\delta(s_{R_0+1},S_{R_0+1})]}{\sum_{s_1\ldots s_{R_0}}\sum_{s_{R_0+1}}w(s_{R_0+1})p_\Lambda(X_{R_0+1},s_{R_0+1})w(s_1\ldots s_{R_0})p_\Lambda(X_1\ldots X_{R_0},s_1\ldots s_{R_0})}\\
&= \frac{\sum_{s_1\ldots s_{R_0+1}}w(s_1\ldots s_{R_0+1})p_\Lambda(X_1\ldots X_{R_0+1},s_1\ldots s_{R_0+1})C(s_1\ldots s_{R_0+1})}{\sum_{s_1\ldots s_{R_0+1}}w(s_1\ldots s_{R_0+1})p_\Lambda(X_1\ldots X_{R_0+1},s_1\ldots s_{R_0+1})},
\end{aligned}
\tag{10}
$$

that is, (8) is valid for $R = R_0 + 1$. This completes the proof by induction.

The significance of the rational functional form of the MCE criterion is that it enables the use of the GT/EBW optimization method for discriminative training of the HMM parameters, which we elaborate below.

4. A new look at MCE – optimization method

4.1. Introduction to the growth-transformation optimization technique

GT/EBW technique was developed for optimization of a rational function. Gopalakrishnan et al. (1991) proposed the GT/EBW based MMI estimation for the discrete HMM, and the method was extended for MMI estimation of the continuous-density HMM (CDHMM) in (Normandin, 1991). Later Gunawardana and Byrne (2001) give an alternative method for MMI estimation of CDHMM, and its validity is proved in (Axelrod et al. (in press)). In following sections, we will present a similar method for optimization of the re-formulated MCE objective function.

Let $G(\Lambda)$ and $H(\Lambda)$ be two real valued functions on the parameter set Λ, and let the denominator function $H(\Lambda)$ be positive valued. Construct the objective function as the ratio of them to form the rational function of

$$O(\Lambda) = \frac{G(\Lambda)}{H(\Lambda)}. \tag{11}$$

An example of this rational function is the objective function for the MCE criterion, where

$$G(\Lambda) = \sum_s w(s)p_\Lambda(X,s)C(s), \quad \text{and}$$
$$H(\Lambda) = \sum_s w(s)p_\Lambda(X,s), \tag{12}$$

and we use $s = s_1,\ldots,s_R$ to denote the label sequences for all R training tokens, and use $X = X_1,\ldots,X_R$, to denote the observation data sequences for all R training tokens.

As in (Gopalakrishnan et al., 1991), for the objective function with the form of (11), the GT-based optimization algorithm constructs the auxiliary function of

$$F(\Lambda;\Lambda') = G(\Lambda) - O(\Lambda')H(\Lambda) + D, \tag{13}$$

where D is a quantity independent of the parameter set Λ, and Λ' denotes the parameter set obtained from the immediately previous iteration of the algorithm.

The GT algorithm starts by initializing the parameter set as, say, Λ'. (This is often accomplished by the ML training using, for instance, EM or Baum–Welch algorithm for HMMs.) Then, updating of the parameter set from Λ' to Λ proceeds by maximizing the auxiliary function $F(\Lambda;\Lambda')$, and the process iterates until convergence is reached. Maximizing the auxiliary function $F(\Lambda;\Lambda')$ is often easier than maximizing the original rational function $O(\Lambda)$. It is easy to prove (Gopalakrishnan et al., 1991) that as long as D is a quantity not relevant to the parameter set Λ, an increase of $F(\Lambda;\Lambda')$ guarantees an increase of $O(\Lambda)$.

We now define another auxiliary function from the previous auxiliary function $F(\Lambda;\Lambda') = F(\theta)$ defined in (13). This new function is:

$$V(\theta;\theta') = \sum_q \int_\chi f(\chi,q,\theta')\log f(\chi,q,\theta)\mathrm{d}\chi, \tag{14}$$

where the positive, real valued function $f(x,q,\theta) > 0$ is defined by

$$F(\theta) = \sum_q \int_\chi f(\chi,q,\theta)\mathrm{d}\chi \tag{15}$$

and where q is a discrete variable (e.g., a state sequence in an HMM).

Then we have:

$$
\begin{aligned}
&\log F(\theta) - \log F(\theta') \\
&\quad = \log \frac{F(\theta)}{F(\theta')} = \log \sum_q \int_\chi \frac{f(\chi,q,\theta')}{F(\theta')} \frac{f(\chi,q,\theta)}{f(\chi,q,\theta')} \mathrm{d}\chi \\
&\quad \geqslant \sum_q \int_\chi \frac{f(\chi,q,\theta')}{F(\theta')} \log \frac{f(\chi,q,\theta)}{f(\chi,q,\theta')} \mathrm{d}\chi \\
&\quad = \frac{1}{F(\theta')} \left[\sum_q \int_\chi f(\chi,q,\theta') \log f(\chi,q,\theta) \mathrm{d}\chi \right. \\
&\qquad \left. - \sum_q \int_\chi f(\chi,q,\theta') \log f(\chi,q,\theta') \mathrm{d}\chi \right] \\
&\quad = \frac{1}{F(\theta')} [V(\theta;\theta') - V(\theta';\theta')]. \qquad (16)
\end{aligned}
$$

The inequality above is due to Jensen's inequality (Jensen, 1906) applied to the concave log function. The result of (16) says that an increase in the auxiliary function $V(\theta;\theta')$ guarantees an increase in $\log F(\theta)$. Since logarithm is a monotonically increasing function, this also guarantees an increase of $F(\theta)$ and hence the original objective function $O(\Lambda)$. The technique that "transforms" the parameters from Λ' to Λ so as to increase or "grow" the values of the auxiliary functions and hence the value of the original objective function is called the growth-transformation (GT) technique (Gopalakrishnan et al., 1991). We now apply this GT technique to the Gaussian HMM with the MCE optimization criterion formulated in (8).

4.2. Application to Gaussian HMM

Substituting (12) into (13), we obtain the auxiliary function

$$
\begin{aligned}
F(\Lambda;\Lambda') &= \textstyle\sum_s w(s) p_\Lambda(X,s) C(s) \\
&\quad - O(\Lambda') \sum_s w(s) p_\Lambda(X,s) + D \\
&= \sum_s w(s) p_\Lambda(X,s)[C(s) - O(\Lambda')] + D \\
&= \sum_q \sum_s w(s) p_\Lambda(X,q,s)[C(s) - O(\Lambda')] + D \qquad (17)
\end{aligned}
$$

where q is the HMM state sequence, and $s = s_1, \ldots, s_R$ is the label sequence (e.g., the word or phone sequence) for all R training tokens (including both correct or incorrect label sequences).

Follow the method used in (Gunawardana and Byrne, 2001), we can re-formulate $F(\Lambda;\Lambda')$ as follows. Let Λ consist of mean and variance parameters in the HMM. Since (q,s) is irrelevant with Λ, we have $p(X,q,s|\Lambda) = p(X|q,\Lambda)P(q,s)$, and hence

$$
\begin{aligned}
F(\Lambda;\Lambda') &= \sum_q \left[\sum_s w(s) P(q,s)[C(s) - O(\Lambda')] \right] p_\Lambda(X|q) + D \\
&= \sum_q \int_\chi [\Gamma(\Lambda') + d(q)] p_\Lambda(\chi|q) \mathrm{d}\chi, \qquad (18)
\end{aligned}
$$

where $\Gamma(\Lambda') = \delta(\chi,X) \sum_s w(s) P(q,s)[C(s) - O(\Lambda')]$, and $D = \sum_q \mathrm{d}(q)$ is a quantity independent of the parameter set Λ. This quantity should be sufficiently large to guarantee that the integrant of (15) be positive, or $\Gamma(\Lambda') + d(q) > 0$ (note $p_\Lambda(\chi|q)$ in (18) is non-negative).

We now desire to construct the auxiliary function of (14) based on the auxiliary function (18). To achieve this, we first identify from (18) that

$$
f(\chi,q,\Lambda) = [\Gamma(\Lambda') + d(q)] p(\chi|q,\Lambda),
$$

according to (15). Then, using (14), we have

$$
\begin{aligned}
V(\Lambda;\Lambda') &= \sum_q \int_\chi [\Gamma(\Lambda') + d(q)] p_{\Lambda'}(\chi|q) \log\{[\Gamma(\Lambda') + d(q)] p_\Lambda(\chi|q)\} \mathrm{d}\chi \\
&= \sum_q \int_\chi [\Gamma(\Lambda') + d(q)] p_{\Lambda'}(\chi|q) \log p_\Lambda(\chi|q) \mathrm{d}\chi + K \\
&= \sum_q [\sum_s w(s) p_{\Lambda'}(X,q,s)(C(s) - O(\Lambda'))] \log p_\Lambda(X|q) \\
&\quad + \sum_q d(q) \int_\chi p_{\Lambda'}(\chi|q) \log p_\Lambda(\chi|q) \mathrm{d}\chi + K. \qquad (19)
\end{aligned}
$$

Ignoring optimization-independent quantity K in (19), and dividing $V(\Lambda;\Lambda')$ by another optimization-independent quantity $p_{\Lambda'}(X)$, we obtain an equivalent auxiliary function of

$$
\begin{aligned}
U(\Lambda;\Lambda') &= \sum_q \left[\sum_s w(s) p_{\Lambda'}(s|X) p_{\Lambda'}(q|X,s)(C(s) - O(\Lambda')) \right] \log p_\Lambda(X|q) \\
&\quad + \sum_q d'(q) \int_\chi p_{\Lambda'}(\chi|q) \log p_\Lambda(\chi|q) \mathrm{d}\chi \qquad (20)
\end{aligned}
$$

where

$$
d'(q) = d(q)/p_{\Lambda'}(X). \qquad (21)
$$

Note $X = X_1, \ldots, X_R$, is a large aggregate of all training data with R independent sentence tokens, and for each token $X_r = x_{r,1}, \ldots, x_{r,Tr}$, the observation vector x_r, depends only on the state at time t. This enables decomposition of $\log p_\Lambda(X|q)$ and then drastic simplification of both terms in (20). To pursue the simplification, we define

$$
\gamma_{i,r,s_r}(t) = p_{\Lambda'}(q_{r,t} = i|X_r, s_r), \qquad (22)
$$

as the occupation probability of state i at time t, given the label sequence s_r and observation sequence X_r. Note (22) can be efficiently computed by the standard forward–backward algorithm (Rabiner and Juang, 1993). We also define

$$
d(r,t,i) = \sum_{q,q_{r,t}=i} d'(q). \qquad (23)
$$

Then, after a series of algebraic steps, Eq. (20) can be simplified to:

Please cite this article in press as: He, X., Deng, L., A new look at discriminative training for hidden Markov models, Pattern Recogn. Lett. (2007), doi:10.1016/j.patrec.2006.11.022

$$
\begin{aligned}
U(\Lambda;\Lambda') = {} & \sum_{r=1}^{R}\sum_{t=1}^{T_r}\sum_{i=1}^{I}\sum_{s} w(s)p_{\Lambda'}(s|X)(C(s) \\
& - O(\Lambda'))\gamma_{i,r,s_r}(t)\log p_{\Lambda}(x_{r,t}|q_{r,t}=i) \\
& + \sum_{r=1}^{R}\sum_{t=1}^{T_r}\sum_{i=1}^{I} d(r,t,i) \\
& \times \int_{\chi_{r,t}} p_{\Lambda'}(\chi_{r,t}|q_{r,t}=i)\log p_{\Lambda}(\chi_{r,t}|q_{r,t}=i)\mathrm{d}\chi_{r,t}.
\end{aligned}
\tag{24}
$$

We now proceed to maximize (24) with respect to the Gaussian HMM's parameters, mean vectors and covariance matrices $\Lambda = \{\mu_i, \Sigma_i\}$, $i = 1, 2, \ldots, I$, in the following state-conditioned Gaussian distribution:

$$
p_{\Lambda}(x|q=i) \propto \frac{1}{|\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)\right].
$$

We set $\frac{\partial U(\Lambda;\Lambda')}{\partial \Lambda} = 0$ and solve for Λ given the model parameters $\Lambda' = \{\mu'_i, \Sigma'_i\}$ from the previous iteration of the GT/EBW. For the mean and covariance, respectively, in the Gaussian at the HMM's state i, we set

$$
\frac{\partial U(\Lambda;\Lambda')}{\partial \mu_i} = 0; \quad \text{and} \quad \frac{\partial U(\Lambda;\Lambda')}{\partial \Sigma_i} = 0.
$$

This eventually gives the "growth transformation" formulas of:

$$
\mu_i = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T_r}\Delta\gamma(i,r,t)x_t + D_i\mu'_i}{\sum_{r=1}^{R}\sum_{t=1}^{T_r}\Delta\gamma(i,r,t) + D_i}
\tag{25}
$$

and

$$
\Sigma_i = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T_r}[\Delta\gamma(i,r,t)(x_t-\mu_i)(x_t-\mu_i)^T] + D_i\Sigma'_i + D_i(\mu_i-\mu'_i)(\mu_i-\mu'_i)^T}{\sum_{r=1}^{R}\sum_{t=1}^{T_r}\Delta\gamma(i,r,t) + D_i},
\tag{26}
$$

where we use the new definitions of

$$
D_i = \sum_{r=1}^{R}\sum_{t=1}^{T_r} d(r,t,i),
\tag{27}
$$

$$
\Delta\gamma(i,r,t) = \sum_{s} w(s)p_{\Lambda'}(s|X)(C(s) - O(\Lambda'))\gamma_{i,r,s_r}(t).
\tag{28}
$$

And we leave the detailed derivations leading to (25) and (26) to the interested readers.

4.3. Computing $\Delta\gamma(i,r,t)$

The major computational steps in the above GT re-estimation formulas are the computation of $\Delta\gamma(i,r,t)$ in (28), which involves summation over all possible label sequences $s = s_1, \ldots, s_R$. The number of training tokens (sentence strings), R, is usually very large. Hence, summation over s needs to be decomposed and simplified.

Denote $s' = s_1, \ldots, s_{r-1}$, $s'' = s_{r+1}, \ldots, s_R$, $X' = X_1, \ldots, X_{r-1}$, $X'' = X_{r+1}, \ldots, X_R$. Then, from (28), we obtain

$$
\begin{aligned}
\Delta\gamma(i,r,t) = {} & \sum_{s'}\sum_{s_r}\sum_{s''} w(s',s_r,s'')p_{\Lambda'}(s',s_r,s''|X',X_r,X'') \\
& \times (C(s',s_r,s'') - O(\Lambda'))\gamma_{i,r,s_r}(t) \\
= {} & \sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r) \\
& \times \underbrace{\left[\sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')(C(s',s_r,s'') - O(\Lambda'))\right]}_{\text{Term I}}\gamma_{i,r,s_r}(t).
\end{aligned}
\tag{29}
$$

Using $C(s',s_r,s'') = C(s_r) + C(s',s'')$, Term I in (29) can be simplified to

$$
\begin{aligned}
\text{Term I} = {} & \sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')(C(s',s_r,s'') - O(\Lambda')) \\
= {} & \sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')C(s_r) \\
& + \sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')C(s',s'') \\
& - O(\Lambda')\sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'').
\end{aligned}
$$

And using

$$
\begin{aligned}
O(\Lambda') = {} & \sum_{r=1}^{R}\frac{w(S_r)p_{\Lambda'}(X_r,S_r)}{\sum_{s_r} w(s_r)p_{\Lambda'}(X_r,s_r)} \\
= {} & \frac{w(S_r)p_{\Lambda'}(X_r,S_r)}{\sum_{s_r} w(s_r)p_{\Lambda'}(X_r,s_r)} + \sum_{i=1,i\neq r}^{R}\frac{w(S_i)p_{\Lambda'}(X_i,S_i)}{\sum_{s_i} w(s_i)p_{\Lambda'}(X_i,s_i)} \\
= {} & \frac{w(S_r)p_{\Lambda'}(X_r,S_r)}{\sum_{s_r} w(s_r)p_{\Lambda'}(X_r,s_r)} \\
& + \frac{\sum_{s',s''} w(s',s'')p_{\Lambda'}(s',s'',X',X'')C(s',s'')}{\sum_{s',s''} w(s',s'')p_{\Lambda'}(s',s'',X',X'')} \\
= {} & \frac{w(S_r)p_{\Lambda'}(S_r|X_r)}{\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)} \\
& + \frac{\sum_{s',s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')C(s',s'')}{\sum_{s',s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')},
\end{aligned}
$$

we obtain:

$$
\begin{aligned}
\text{Term I} = {} & \sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')C(s_r) \\
& + \sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')C(s',s'') \\
& - \frac{w(S_r)p_{\Lambda'}(S_r|X_r)}{\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)}\sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'') \\
& - \sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')C(s',s'') \\
= {} & \underbrace{\sum_{s'}\sum_{s''} w(s',s'')p_{\Lambda'}(s',s''|X',X'')}_{\text{Term II}} \\
& \times \left[C(s_r) - \frac{w(S_r)p_{\Lambda'}(S_r|X_r)}{\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)}\right].
\end{aligned}
$$

The quantity above denoted by Term II can be simplified to:

$$\text{Term II} = \frac{\sum_{s'}\sum_{s_r}\sum_{s''} w(s', s_r, s'')p_{\Lambda'}(s', s_r, s''|X', X_r, X'')}{\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)}$$
$$= \frac{Q(\Lambda')}{\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)}$$

where we define

$$Q(\Lambda') = \sum_{s} w(s)p_{\Lambda'}(s|X)$$
$$= \sum_{s_1}\cdots\sum_{s_R} w(s_1)\cdots w(s_R)\cdot p_{\Lambda'}(s_1|X_1)\cdots p_{\Lambda'}(s_R|X_R)$$
$$= \prod_{r=1}^{R}\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r).$$

Substituting the above terms back to, we obtain:

$$\Delta\gamma(i,r,t) = \sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)\frac{Q(\Lambda')}{\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)} \times \left[C(s_r) - \frac{w(S_r)p_{\Lambda'}(S_r|X_r)}{\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)}\right]\gamma_{i,r,s_r}(t). \quad (30)$$

For the 1-best MCE where $w(s) \equiv 1$, $Q(\Lambda') = 1$, takes a simpler form of:

$$\Delta\gamma(i,r,t) = \sum_{s_r} p_{\Lambda'}(s_r|X_r)[C(s_r) - p_{\Lambda'}(S_r|X_r)]\gamma_{i,r,s_r}(t). \quad (31)$$

4.4. Considerations for setting empirical constant D_i

In the GT re-estimation formulas (25) and (26), the value of constant D_i is empirically set and it determines the stability and convergence rate of the algorithm. We now discuss the basis for setting this constant in practice. From (27), (23), and (21), we have

$$D_i = \sum_{r=1}^{R}\sum_{t=1}^{T_r} d(r,t,i) = \sum_{r=1}^{R}\sum_{t=1}^{T_r}\sum_{q,q_{r,t}=i} d'(q)$$
$$= \frac{1}{p_{\Lambda'}(X)}\sum_{r=1}^{R}\sum_{t=1}^{T_r}\sum_{q,q_{r,t}=i} d(q). \quad (32)$$

According to Jensen Inequality, the theoretical basis for setting D_i is the requirement described in (18) that $d(q)$ be sufficiently large to ensure that $d(q) > -\Gamma(\Lambda')$.

This gives

$$D_i > \frac{1}{p(X|\Lambda')}\sum_{r=1}^{R}\sum_{t=1}^{T_r}\sum_{q,q_{r,t}=i} -\Gamma(\Lambda'). \quad (33)$$

For the continuous density HMM case, however,$\Gamma(\Lambda')$ is unbounded since $\delta(\chi, X)$ is unbounded at the center point $\chi = X$, and D_i needs to approach to infinite. To address this issue, follow the similar derivation as in (Axelrod et al. (in press)), it can be proved that with a large enough but bounded D_i, the function $V(\Lambda; \Lambda')$ at (19) is still a valid auxiliary function of the objective function $O(\Lambda)$, i.e., increasing the value of $V(\Lambda; \Lambda')$ will guarantee increase of the value of $F(\Lambda; \Lambda')$, and so as to guarantee increase of the value of the objective function $O(\Lambda)$.

In implementation, we are more interested in the practical setting of D_i, which is usually determined empirically for fast convergence. The value of $D_i$which we have found practically effective for 1-best MCE is

$$D_i = E\cdot\sum_{r=1}^{R} p_{\Lambda'}(S_r|X_r)\sum_{s_r} p_{\Lambda'}(s_r|X_r)\sum_{t}\gamma_{i,r,s_r}(t), \quad (34)$$

where E is a factor controlling the learning rate. The larger the E is, the slower the learning rate becomes, and E is usually a factor between one and four for 1-best MCE. Extending (34) to N-best MCE, we have,

$$D_i = E\cdot\sum_{r=1}^{R}\frac{Q(\Lambda')}{\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)}\frac{w(S_r)p_{\Lambda'}(S_r|X_r)}{\sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)} \times \sum_{s_r} w(s_r)p_{\Lambda'}(s_r|X_r)\sum_{t}\gamma_{i,r,s_r}(t). \quad (35)$$

5. Use of Lattice for representing competitive candidates in MCE training

In the above novel development of the MCE training technique, N-best lists are used to represent the competing candidates for discriminative learning. In many speech recognition tasks, in order to make the competing candidates sufficiently rich, N in the N-best lists needs to be very large (e.g., in the order of millions). This will create computational difficulties. To overcome such difficulties, we can use a lattice to serve as a compressed form of a very large N-best list, which has been shown to be successful and critical in MMI and MPE learning (Woodland and Povey, 2000; Povey, 2004). The previous work that discussed the use of lattices for MCE training was reported in (Schluter et al., 2001), where the misclassification measure takes the approximate form of

$$d_r(X_r, \Lambda) = -\log p_{\Lambda}^{\alpha}(X_r, S_r) + \log\sum_{s_r, s_r\neq S_r} p_{\Lambda}^{\alpha}(X_r, s_r), \quad (36)$$

to the misclassification measure of (2). This is a special case of our approximate form of (3) where $w(s_r) \equiv 1$ for all strings including the incorrect (competing) strings $\{s_r: s_r \neq S_r\}$, and the correct string S_r.

In the above special case and with $\alpha = 1$, our earlier results in (30) and (35) become simplified to

$$\Delta\gamma(i,r,t) = \sum_{s_r} p_{\Lambda'}(s_r|X_r)[C(s_r) - p_{\Lambda'}(S_r|X_r)]\gamma_{i,r,s_r}(t), \quad (37)$$

$$D_i = E\cdot\sum_{r=1}^{R} p_{\Lambda'}(S_r|X_r)\sum_{t}\sum_{s_r} p_{\Lambda'}(s_r|X_r)\gamma_{i,r,s_r}(t). \quad (38)$$

In this section, instead of computing $\Delta\gamma(i,r,t)$ of (37) and in (38) a brute-force manner by summing a huge number of strings of s_r for the very large N-best list as represented by a lattice, we use an approximation that makes the computation of (37) and (38) practically feasible. This then gives a solution for lattice-based MCE parameter estimation after using the computed results of (37) and (38) in the estimation formulas (25) and (26). This solution differs markedly from that reported in (Schluter et al., 2001). Specifically, our solution does not require removing the correct word string S_r from the lattice. In contrast, removal of S_r in the lattice is required by the solution provided in (Schluter et al., 2001), which is more difficult to implement in practice than our solution. In addition, our solution has the theoretical appeal of guaranteed algorithm convergence since it is derived based on GT/EBW for a rational function. The solution provided in (Schluter et al., 2001) does not have such convergence guarantee.

To compute (37), we first use $C(s_r) = \delta(s_r, S_r)$ for MCE to rewrite (37) as

$$\begin{aligned}\Delta\gamma(i,r,t) &= \sum_{s_r} p_{\Lambda'}(s_r|X_r,)C(s_r)\gamma_{i,r,s_r}(t) \\ &\quad - \sum_{s_r} p_{\Lambda'}(s_r|X_r)p_{\Lambda'}(S_r|X_r)\gamma_{i,r,s_r}(t) \\ &= p_{\Lambda'}(S_r|X_r)\gamma_{i,r,S_r}(t) - p_{\Lambda'}(S_r|X_r) \\ &\quad \times \underbrace{\sum\nolimits_{s_r} p_{\Lambda'}(s_r|X_r)\gamma_{i,r,s_r}(t)}_{\Upsilon}. \end{aligned} \tag{39}$$

In (39), since the correct string S_r is known, $\gamma_{i,r,S_r}(t)$ can be computed straightforwardly by the standard forward–backward algorithm for the HMM (Rabiner and Juang, 1993). The main computation thus lies in

$$\Upsilon = \sum_{s_r} p_{\Lambda'}(s_r|X_r)\gamma_{i,r,s_r}(t) \tag{40}$$

and

$$p_{\Lambda'}(S_r|X_r) = \frac{p_{\Lambda'}(S_r, X_r)}{p_{\Lambda'}(X_r)}. \tag{41}$$

for $\Delta\gamma(i,r,t)$ of (37), as well as for D_i in (38).

To efficiently compute (40) and (41) for the lattice representation of strings of s_r, we need to make a mild approximation. A lattice is a compact representation of a large list of strings. It is an acyclic directed graph consisting of a number of nodes and a set of directed arcs each connecting two nodes. A typical arc is denoted as q, and an arc corresponds to a substring (e.g., a word in a sentence). Two time stamps, b_q and e_q, are associated with each arc, providing an estimate of the segment boundaries for the substring. For a time slice t within the arc segment q, we have $b_q \leqslant t \leqslant e_q$.

We will show below that (40) and (41) can both be computed efficiently by a forward–backward algorithm on the lattice after the mild assumption that HMM state sequences are independent across arcs, that is,

$$\gamma_{i,r,q}(t) \approx \gamma_{i,r,s_r}(t) \quad \text{when} \quad b_q \leqslant t \leqslant e_q \quad and\ q \in s_r. \tag{42}$$

This approximation says that within the segment of arc q, its occupancy posterior probability $\gamma_{i,r,s_r}(t) = p(q_{r,t} = i|X_r, s_r, \Lambda')$ given the observation sequence X_r for the sentence-level string s_r that passes arc q approximates the posterior probability $\gamma_{i,r,q}(t)$ for arc q alone. The justification of approximation (42) is that the state sequence within arc $q$should be roughly independent of other arcs. This was called "exact-matching" approximation in (Povey, 2004). To see this, let s_r be composed of three sub-strings: s'_r, q, s''_r. Then the right hand side of (42) can be analyzed to be

$$\begin{aligned}\gamma_{i,r,s_r}(t : b_q \leqslant t \leqslant e_q) &= p_{\Lambda'}(q_{r,t:b_q\leqslant t\leqslant e_q} = i|X_r, s_r) \\ &= p_{\Lambda'}\left(q_{r,t:b_q\leqslant t\leqslant e_q} = i|X_r, s'_r, q, s''_r\right) \\ &\approx p_{\Lambda'}(q_{r,t:b_q\leqslant t\leqslant e_q} = i|X_r, q)\end{aligned}$$

which is the left hand side of (42).

The essence of approximation (42) is to decouple the dependency on the local arc q from the entire string s_r. This enables drastic simplification of the computation in (40) and (41), which we discuss below.

As we discussed earlier, the principal computation burden in (40) is the huge number (N) of summation terms for s_r for the equivalent N-best list of a lattice. Using approximation of (42), we can drastically reduce the computation by the following simplification:

$$\begin{aligned}\Upsilon &\approx \sum\nolimits_{s_r} p_{\Lambda'}(s_r|X_r)\gamma_{i,r,q}(t) \\ &= \sum_{q:t\in[b_q,e_q]} \gamma_{i,r,q}(t) \cdot \sum_{s_r:q\in s_r} p_{\Lambda'}(s_r|X_r) \\ &= \sum_{q:t\in[b_q,e_q]} \gamma_{i,r,q}(t) \cdot p_{\Lambda'}(q|X_r) \\ &= \sum_{q:t\in[b_q,e_q]} \gamma_{i,r,q}(t) \cdot \frac{p_{\Lambda'}(q,X_r)}{p_{\Lambda'}(X_r)}.\end{aligned} \tag{43}$$

The key quantities in (43) can be efficiently computed as follows (See the derivation in Appendix I):

$$p_{\Lambda'}(q, X_r) = \alpha(q)\beta(q); \tag{44}$$

$$p_{\Lambda'}(X_r) = \sum_{q:q\in\ \{\text{ending arcs}\}} p_{\Lambda'}(q, X_r) = \sum_{q:q\in\ \{\text{ending arcs}\}} \alpha(q) \tag{45}$$

where the "forward" and "backward" probabilities are defined by

$$\alpha(q) = \sum_{p(\text{preceding } q)} p_{\Lambda'}(p, q, X'_r(q), X_r(q)) = p_{\Lambda'}(q, X'_r(q), X_r(q)); \tag{46}$$

$$\beta(q) = \sum_{v(\text{succeeding } q)} p_{\Lambda'}(v, X''_r(q)|q) = p_{\Lambda'}(X''_r(q)|q). \tag{47}$$

In (46), $X'_r(q)$ denotes the rth training token's observation sequence preceding arc q, i.e., during $1 \leqslant t < b_q$. $X_r(q)$ is the observation sequence bounded by arc q with $b_q \leqslant t \leqslant e_q$. $X''_r(q)$ in (47) denotes the observation sequence succeeding arc q, or during$e_q < t \leqslant T_r$.

For each arc q in the lattice, $\alpha(q)$ and $\beta(q)$ can be computed by the following efficient forward and backward recursions, respectively (See the derivation in Appendix II):

$$\alpha(q) = \sum_{p(\text{preceding } q)} p_{\Lambda'}(q|p) p_{\Lambda'}(X_r(q)|q)\alpha(p) \tag{48}$$

and

$$\beta(q) = \sum_{v(\text{succeeding } q)} p_{\Lambda'}(v|q) p_{\Lambda'}(X_r(v)|v)\beta(v), \tag{49}$$

where $\alpha(q)$ is initialized at the starting arc q_0 by $\alpha(q_0) = \pi(q_0) p_{\Lambda'}(X_r(q_0)|q_0)$, and $\beta(q)$ initialized at the ending arc q_E by $\beta(q_E) = 1$.

Using forward probability $\alpha(q)$, we can efficiently compute (41) as follows. Since $p(X_r|\Lambda') = \sum_{q:q\in\{\text{ending arcs}\}}\alpha(q)$ and $p'_\Lambda(S_r, X_r) = p'_\Lambda(X_r|S_r)p(S_r)$, we have

$$p(S_r|X_r, \Lambda') = \frac{p(X_r|S_r, \Lambda')p(S_r)}{\sum\limits_{q:q\in\{\text{ending arcs}\}} \alpha(q)}. \tag{50}$$

6. Summary and discussion

HMMs are continuing to play a central role in speech recognition research and technology deployment, where training techniques for the HMM parameters have been a critical determinant for the speech recognition accuracy and user satisfaction level. While discriminative training for HMMs, typified by the MCE technique, has been pursued with a relatively long history, it is not until recently that the traditional gradient-based MCE optimization technique has been questioned (Macherey et al., 2005; He and Chou, 2003). In this paper, we provide a fresh look at the MCE technique not only from the perspective of the optimization technique, but also of the objective function. The key technical contribution of this paper is the establishment of a non-trivial framework in which the MCE objective function is re-formulated as a rational function for multiple sentence-level training tokens. And we show that the *N*-best representation of the competitive candidates in MCE training amounts to a special weighting function in the newly formulated MCE objective function. As a consequence of this re-formulation, we most naturally derive the new optimization method for discriminatively estimating HMM parameters based on GT/EBW. This method has been successfully implemented in a speech recognition system, and the positive experimental results can be found in (He et al., 2006).

In addition to the usual treatment of MCE training using the *N*-best paradigm, in this paper, we also provide further, more difficult technical detail for the use of lattices as a richer representation of competing candidates. This treatment can be considered as a technical guide for implementing MCE training in large-scale speech recognition systems. We are currently experimenting with this approach.

Appendix I. Derivation of Eqs. (44) and (45)

Given the "forward" and "backward" probabilities defined as (46) and (47), as well as $X'_r(q)$, $X_r(q)$ and $X''_r(q)$ defined in Section 5, a derivation of (44) and (45) is provided below.

Derivation of (44):

$$\begin{aligned}
p_{\Lambda'}(q, X_r) &= \sum_{p(\text{preceding } q)} \sum_{v(\text{succeeding } q)} p_{\Lambda'}\left(p, q, v, X'_r(q), X_r(q), X''_r(q)\right) \\
&= \sum_{p(\text{preceding } q)} \sum_{v(\text{succeeding } q)} p_{\Lambda'}\left(p, q, X'_r(q), X_r(q)\right) p_{\Lambda'} \\
&\quad \times \left(v, X''_r(q)|p, q, X'_r(q), X_r(q)\right) \\
&= \sum_{p(\text{preceding } q)} p_{\Lambda'}\left(p, q, X'_r(q), X_r(q)\right) \\
&\quad \times \sum_{v(\text{succeeding } q)} p_{\Lambda'}\left(v, X''_r(q)|q\right) = \alpha(q)\beta(q).
\end{aligned}$$

Derivation of (45):

$$\begin{aligned}
p_{\Lambda'}(X_r) &= \sum_{q:q\in\{\text{ending arcs}\}} p_{\Lambda'}(q, X_r) \\
&= \sum_{q:q\in\{\text{ending arcs}\}} p_{\Lambda'}(q, X'_r(q), X_r(q)) \\
&= \sum_{q:q\in\{\text{ending arcs}\}} \alpha(q).
\end{aligned}$$

Appendix II. Derivation of (48) and (49) for the forward–backward computation

Given the "forward" and "backward" probabilities defined as (46) and (47), as well as $X'_r(q)$, $X_r(q)$ and $X''_r(q)$ defined in Section 5, a derivation of (48) and (49) is provided here.

Forward computation for $\alpha(q)$:

$$\begin{aligned}
\alpha(q) &= \sum_{p(\text{preceding } q)} p_{\Lambda'}\left(p, q, X'_r(q), X_r(q)\right) \\
&= \sum_{p(\text{preceding } q)} p_{\Lambda'}\left(p, q, X'_r(p), X_r(p), X_r(q)\right) \\
&= \sum_{p(\text{preceding } q)} p_{\Lambda'}\left(q, X_r(q)|p, X'_r(p), X_r(p)\right) p_{\Lambda'}\left(p, X'_r(p), X_r(p)\right) \\
&= \sum_{p(\text{preceding } q)} p_{\Lambda'}(q|p) p_{\Lambda'}(X_r(q)|q) p_{\Lambda'}\left(p, X'_r(p), X_r(p)\right) \\
&= \sum_{p(\text{preceding } q)} p_{\Lambda'}(q|p) p_{\Lambda'}(X_r(q)|q)\alpha(p).
\end{aligned}$$

Backward computation for $\beta(q)$:

$$\begin{aligned}
\beta(q) &= \sum_{v(\text{succeeding } q)} p_{\Lambda'}(v, X''_r(q)|q) \\
&= \sum_{v(\text{succeeding } q)} p_{\Lambda'}(v, X_r(v), X''_r(v)|q) \\
&= \sum_{v(\text{succeeding } q)} p_{\Lambda'}(v, X_r(v)|q) p_{\Lambda'}(X''_r(v)|q, v, X_r(v))
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{v(\text{succeeding } q)} p_{\Lambda'}(v|q) p_{\Lambda'}(X_r(v)|q, v) p_{\Lambda'}(X''_r(v)|v) \\
&= \sum_{v(\text{succeeding } q)} p_{\Lambda'}(v|q) p_{\Lambda'}(X_r(v)|v) \beta(v)
\end{aligned}
$$

References

Axelrod, S., Goel, V., Gopinath, R., Olsen, P., Visweswariah, K., in press. Discriminative Estimation of Subspace Constrained Gaussian Mixture Models for Speech Recognition, IEEE Trans. Audio Speech Language Process., http://ieeexplore.ieee.org/iel5/10376/32978/101109TASL 2006872617.pdf.

Bahl, L., Jelinek, F., Mercer, R., 1987. A Maximum likelihood approach to continuous speech recognition. IEEE Trans. Pattern Anal. Mach. Intell. PAMI-5, 179–190.

Brown, P., 1987. The Acoustic Modeling Problem in Automatic Speech Recognition, Ph.D. thesis, Carnegie Mellon University.

Chou, W., 2003. Minimum classification error approach in pattern recognition. In: Chou, W., Juang, B.-H. (Eds.), Pattern Recognition in Speech and Language Processing. CRC Press, pp. 1–49.

Deng, L., Wu, J., Droppo, J., Acero, A., 2005. Analysis and comparison of two feature extraction/compensation algorithms. IEEE Signal Process. Lett. 12 (6), 477–480.

Deng, L., Yu, D., Acero, A., 2005. A generative modeling framework for structured hidden speech dynamics. In: Proc. of Neural Information Processing System (NIPS) Workshop, Whistler, BC, Canada, December 2005.

Deng, L., O'Shaughnessy, D., 2003. SPEECH PROCESSING – A Dynamic and Optimization-Oriented Approach. Marcel Dekker Inc., New York, NY, USA.

Gopalakrishnan, P., Kanevsky, D., Nadas, A., Nahamoo, D., 1991. An inequality for rational functions with applications to some statistical estimation problems. IEEE Trans. Inf. Theory. 37, 107–113.

Gunawardana, A., Byrne, W., 2001. Discriminative speaker adaptation with conditional maximum likelihood linear regression. In: Proc. EUROSPEECH.

He, X., Chou, W., 2003. Minimum classification error linear regression for acoustic model adaptation of continuous density HMMs. In: Proc. ICASSP.

He, X., Deng, L., Chou, W., 2006. A novel learning method for hidden Markov models in speech and audio processing. In: Proc. IEEE Workshop on Multimedia Signal Processing, Victoria, BC.

Jensen, J.L.W.V., 1906. Sur les fonctions convexes et les inegalites entre les valeurs moyennes. Acta Math., 175–193.

Juang, B.-H., Katagiri, S., 1992. Discriminative learning for minimum error classification. IEEE Trans. Signal Process. 40 (12), 3043–3054.

Juang, B.-H., Chou, W., Lee, C.-H., 1997. Minimum classification error rate methods for speech recognition. IEEE Trans. Speech Audio Process. 5.

Macherey, W., Haferkamp, L., Schluter, R., Ney, H., 2005. Investigations on error minimizing training criteria for discriminative training in automatic speech Recognition. In: Proc. Interspeech, Lisbon, Portugal, pp. 2133–2136.

McDermott, E., Hazen, T., Roux, J., Nakamura, A., Katagiri, S., in press. Discriminative training for large vocabulary speech recognition using minimum classification error. IEEE Trans. Audio Speech Language Process., http://www.kecl.ntt.co.jp/icl/signal/erik/index-j.htm.

Normandin, Y., 1991. Hidden Markov Models, Maximum Mutual Information Estimation, and the Speech Recognition Problem, Ph.D. dissertation, McGill University, Montreal.

Povey, D., 2004. Discriminative Training for Large Vocabulary Speech Recognition, Ph.D. thesis, Cambridge University, Cambridge, UK.

Povey, D., Gales, M.J.F., Kim, D.Y., Woodland, P.C., 2003. MMI-MAP and MPE-MAP for acoustic model adaptation, In: Proc. Eurospeech.

Povey, D., Kingsbury, B., Mangu, L., Saon, G., Soltau, H., Zweig, G., 2004. fMPE: Discriminatively trained features for speech recognition. In: Proc. DARPA EARS RT-04 Workshop, November 7–10, Palisades, NY, Paper No. 35.

Povey, D., Woodland, P.C., 2002. Minimum phone error and I-Smoothing for improved discriminative training. In: Proc. ICASSP.

Rabiner, L., Juang, B.-H., 1993. Fundamentals of Speech Recognition. Prentice Hall, Englewood Cliffs, New Jersey.

Rathinavalu, C., Deng, L., 1998. Speech trajectory discrimination using the minimum classification error learning. IEEE Trans. Speech Audio Process. 6 (6), 505–515.

Roux, J., McDermott, E., 2005. Optimization for discriminative training. In: Proc. INTERSPEECH.

Schluter, R., Macherey, W., Muller, B., Ney, H., 2001. Comparison of discriminative training criteria and optimization methods for speech recognition. Speech Commun. 34, 287–310.

Woodland, P.C., Povey, D., 2000. Large scale discriminative training for speech recognition. In: Proc. ITRW ASR, ISCA.

Please cite this article in press as: He, X., Deng, L., A new look at discriminative training for hidden Markov models, Pattern Recogn. Lett. (2007), doi:10.1016/j.patrec.2006.11.022