DISCRIMINATIVE TRAINING OF HMM USING MAXIMUM NORMALIZED LIKELIHOOD ALGORITHM

Konstantin Markov[*]*, Seiichi Nakagawa*

Dep. of Information & Computer Sciences,
Toyohashi University of Technology,
Toyohashi, Japan

Satoshi Nakamura

ATR Spoken Language Translation
Research Laboratories,
Kyoto, Japan

ABSTRACT

In this paper, we present the Maximum Normalized Likelihood Estimation (MNLE) algorithm and its application for discriminative training of HMMs for continuous speech recognition. The objective of this algorithm is to maximize the normalized frame likelihood of training data. Instead of gradient descent techniques usually applied for objective function optimization in other discriminative algorithms such as Minimum Classification Error (MCE) and Maximum Mutual Information (MMI), we used a modified Expectation-Maximization (EM) algorithm which greatly simplifies and speeds up the training procedure. Evaluation experiments showed better recognition rates compared to both the Maximum Likelihood (ML) training method and MCE/GPD discriminative method. In addition,the MNLE algorithm showed better generalization abilities and was faster than MCE/GPD.

1. INTRODUCTION

In recent years, discriminative training methods have attracted the attention of many researchers because they help to overcome the fundamental limitations of the traditional maximum likelihood training approach.

Widely-known and popular discriminative algorithms include Minimum Classification Error/Generalized Probabilistic Descent (MCE/GPD) and Maximum Mutual Information (MMI), which have been successfully applied for speech recognition [1, 2, 3, 4]. When applied for model parameter estimation, discriminative training generally outperforms conventional ML training when the parametric distribution function (usually Gaussian) of the models is inconsistent with the actual data distribution and when the amount of training data is limited and does not allow reliable parameter estimation.

The MCE/GPD method uses classification errors on the training data directly in its objective function. However, as pointed out by some researchers [1, 5], there are some practical difficulties associated with the number of competing classes and the shape of the loss function. In general, when all other classes except the target class are regarded as competing classes and their number is large, the MCE algorithm may become computationally expensive. In addition, although the sigmoid loss function assures good learning for samples laying near correct/incorrect boundaries, almost no learning occurs for badly misrecognized samples.

The objective of the MMI method, on the other hand, is to maximize the class *a posteriori* probability, which is not directly connected with the classification accuracy. Since the calculation of the *a posteriori* probability requires likelihoods from all classes, however, the method also becomes computationally expensive with large number of reference classes. In the MMI method, in addition, gradient descent algorithms which are slow are typically used for optimization of the objective function. As pointed in [6], the Generalized Probabilistic Descent (GPD) algorithm can also be applied in the MMI technique. GPD is an effective algorithm, although it is complicated and requires the adjustment of several free parameters. As concluded in [6], however, the gradient search used in GPD is usually slower than the common EM method, especially in a large-scale task. Another possible approach for MMI optimization is the use of the extended Baum-Welch (EBW) algorithm [4, 7]. However, the EBW convergence is highly sensitive to the value of constant D, and it has been shown that for better results this constant should be model dependent.

The above mentioned drawbacks of existing discriminative design methods prompted us to find a new, less complicated, and faster discriminative training method. We therefore developed the Maximum Normalized Likelihood Estimation algorithm and successfully applied it for the task of speaker recognition [8, 9]. The algorithm aims at increasing the separation between the most competitive classes. The objective function to be maximized uses the likelihood ratio between the target model and its "cohort" of competing models taken for each frame. For the optimization, we use the standard Expectation Maximization (EM) algorithm with some modifications. This affords us a simple and tractable re-estimation procedure.

2. MAXIMUM NORMALIZED LIKELIHOOD ESTIMATION ALGORITHM

2.1. The MNLE Objective

First, we define the normalized likelihood of an observation frame $\boldsymbol{x}$ given for the target model λ_i as:

$$Sc(\boldsymbol{x}|\lambda_i) = \frac{p(\boldsymbol{x}|\lambda_i)}{\frac{1}{B}\sum_{b=1}^{B} p(\boldsymbol{x}|\lambda_{b_i})} \tag{1}$$

where $\lambda_{b_i}(\ b_i = 1_i, \ldots, B_i)$ denotes a set of B competing models (classes) called *background* models for the target model i. If $\boldsymbol{x}$ is a part of the training data for the model i, then we can regard Eq. (1) as a measure of how good model i is on its own data with respect to the background models. The bigger this measure is, the better the separability is between the involved classes. In the same

[*]Currently with ATR Spoken Language Translation Research Laboratories.

manner, we can define the normalized likelihood of a sequence of frames $\boldsymbol{X} = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$ in the log domain as:

$$\log Sc(\boldsymbol{X}|\lambda_i) = \sum_{t=1}^{T} \log \frac{p(\boldsymbol{x}_t|\lambda_i)}{\frac{1}{B}\sum_{b=1}^{B} p(\boldsymbol{x}_t|\lambda_{b_i})} \tag{2}$$

Now, we can define our objective as to maximize $Sc(\boldsymbol{X}|\lambda_i)$ for each i which results in the following objective function:

$$\mathcal{F}(\Lambda) = \sum_{n=1}^{N}\sum_{k=1}^{K_n}\sum_{t=1}^{T_{n,k}} \log \frac{p(\boldsymbol{x}_{n,k,t}|\lambda_n)}{\frac{1}{B}\sum_{b=1}^{B} p(\boldsymbol{x}_{n,k,t}|\lambda_{n,b})} \tag{3}$$

where N is the number of classes, K_n is the number of frame sequences, $T_{n,k}$ is the number of frames in the k^{th} sequence, $\boldsymbol{X}_{n,k} = \{\boldsymbol{x}_{n,k,t}\}$ of the n^{th} class training data, and $\lambda_{b,n}(b = 1_n, \ldots, B_n)$ denotes the background classes for class n. Next, our task is to maximize $\mathcal{F}(\Lambda)$ over the collection of all model parameters $\Lambda = \{\lambda_1, \ldots, \lambda_N\}$.

Note that Eq. (3) can be re-written as:

$$\begin{aligned}\mathcal{F}(\Lambda) &= \sum_{n=1}^{N}\sum_{k=1}^{K_n}\sum_{t=1}^{T_{n,k}} \log p(\boldsymbol{x}_{n,k,t}|\lambda_n) - \\ &\quad \sum_{n=1}^{N}\sum_{k=1}^{K_n}\sum_{t=1}^{T_{n,k}} \frac{1}{B}\sum_{b=1}^{B} p(\boldsymbol{x}_{n,k,t}|\lambda_{n,b}) \\ &= \mathcal{F}_{ML}(\Lambda) - \mathcal{F}_D(\Lambda)\end{aligned} \tag{4}$$

where $\mathcal{F}_{ML}(\Lambda)$ is actually the objective of the standard maximum likelihood (ML) training. The second term $\mathcal{F}_D(\Lambda)$ is a correction term responsible for the discriminative nature of the MNLE method.

We can also express Eq. (3) in the following form:

$$\mathcal{F}(\Lambda) = \sum_{n=1}^{N} \mathcal{F}_n(\Lambda) \tag{5}$$

where

$$\mathcal{F}_n(\Lambda) = \sum_{k=1}^{K_n}\sum_{t=1}^{T_{n,k}} \log \frac{p(\boldsymbol{x}_{n,k,t}|\lambda_n)}{\frac{1}{B}\sum_{b=1}^{B} p(\boldsymbol{x}_{n,k,t}|\lambda_{n,b})} \tag{6}$$

We call $\mathcal{F}_n(\Lambda)$ an *individual objective function* for class n.

The choice of the background models in Eq. (3) greatly influences the performance of MNLE. We have every reason to believe that the target class data is most often misrecognized with some of its closest competing classes. Therefore, choosing those classes as background classes will ensure that maximizing the objective function will result in better separation between them and the target class.

The MNL objective is similar to the MMI objective when defined at the frame level. However, there are two main differences. First, in the MMI objective, the target class acts as a background class for the normalization of its own likelihood. Second, in the MNLE objective, only a small number of the most competitive classes are used as a background set.

2.2. Optimization Algorithm

Let's consider the Expectation Maximization (EM) algorithm for the optimization of the objective function of Eq. (3). Assuming that our classes are modeled by a mixture of Gaussian densities with M pdfs and taking for simplicity only one training sequence of samples, we have for the EM auxiliary function:

$$Q(\Lambda|\overline{\Lambda}) = \sum_{n=1}^{N}\sum_{t=1}^{T_n}\sum_{j=1}^{M} \frac{f(\boldsymbol{x}_{n,t}, \omega_{n,j}|\Lambda)}{f(\boldsymbol{x}_{n,t}|\Lambda)} \log f(\boldsymbol{x}_{n,t}, \omega_{n,j}|\overline{\Lambda}) \tag{7}$$

where $\omega_{n,j}$ specifies the n^{th} model's j^{th} mixture. Now, based on the normalized likelihood formulation, we have for the conditional probability density functions $f()$'s:

$$f(\boldsymbol{x}_{n,t}|\Lambda) = \frac{p(\boldsymbol{x}_{n,t}|\lambda_n)}{\frac{1}{B}\sum_{b=1}^{B} p(\boldsymbol{x}_{n,t}|\lambda_{n,b})} \tag{8}$$

$$f(\boldsymbol{x}_{n,t}, \omega_{n,j}|\Lambda) = \frac{c_{n,j} b_{n,j}(\boldsymbol{x}_{n,t})}{\frac{1}{B}\sum_{b=1}^{B} p(\boldsymbol{x}_{n,t}|\lambda_{n,b})} \tag{9}$$

$$f(\boldsymbol{x}_{n,t}, \omega_{n,j}|\overline{\Lambda}) = \frac{\overline{c}_{n,j} \overline{b}_{n,j}(\boldsymbol{x}_{nt})}{\frac{1}{B}\sum_{b=1}^{B} p(\boldsymbol{x}_{n,t}|\overline{\lambda}_{n,b})} \tag{10}$$

where $c_{n,j}$ is model n's j^{th} mixture coefficient and $b_{n,j}(\boldsymbol{x}_{n,t}) = N(\boldsymbol{x}_{n,t}; \mu_{n,j}, \Sigma_{n,j})$ is a Gaussian function. By inserting the Eqs. (8), (9) and (10) into Eq. (7), we get the final formula for the Q function.

The next step requires that derivatives of the Q function be taken with respect to the new parameters - $\overline{c}_{n,j}$, $\overline{\boldsymbol{\mu}}_{n,j}$, and $\overline{\boldsymbol{\Sigma}}_{n,j}$. Here, we present the final re-estimation equations. The detailed derivations can be found in [9].

$$\overline{c}_{n,j} = \frac{\sum_{t=1}^{T_n} P_{n,j,t} - \sum_{n'}\sum_{t=1}^{T_{n'}} P^b_{n,j,n',t}}{T_n - \sum_{n'}\sum_{t=1}^{T_{n'}} \frac{p(\boldsymbol{x}_{n',t}|\overline{\lambda}_n)}{\sum_{b=1}^{B} p(\boldsymbol{x}_{n',t}|\overline{\lambda}_{n',b})}} \tag{11}$$

$$\overline{\boldsymbol{\mu}}_{n,j} = \frac{\sum_{t=1}^{T_n} P_{n,j,t}\boldsymbol{x}_{n,t} - \sum_{n'}\sum_{t=1}^{T_{n'}} P^b_{n,j,n',t}\boldsymbol{x}_{n',t}}{\sum_{t=1}^{T_n} P_{n,j,t} - \sum_{n'}\sum_{t=1}^{T_{n'}} P^b_{n,j,n',t}} \tag{12}$$

$$\overline{\boldsymbol{\Sigma}}_{n,j} = \frac{\sum_{t=1}^{T_n} P_{n,j,t}\boldsymbol{A}_{n,j,t} - \sum_{n'}\sum_{t=1}^{T_{n'}} P^b_{n,j,n',t}\boldsymbol{A}_{n,j,n',t}}{\sum_{t=1}^{T_n} P_{n,j,t} - \sum_{n'}\sum_{t=1}^{T_{n'}} P^b_{n,j,n',t}} \tag{13}$$

where

$$\boldsymbol{A}_{n,j,t} = (\boldsymbol{x}_{n,t} - \boldsymbol{\mu}_{n,j})(\boldsymbol{x}_{n,t} - \boldsymbol{\mu}_{n,j})^t \tag{14}$$

$$\boldsymbol{A}_{n,j,n',t} = (\boldsymbol{x}_{n',t} - \boldsymbol{\mu}_{n,j})(\boldsymbol{x}_{n',t} - \boldsymbol{\mu}_{n,j})^t \tag{15}$$

and

$$P^b_{n,j,n',t} = P_b(\omega_{n,j}|\boldsymbol{x}_{n',t}) = \frac{\overline{c}_{n,j}\overline{b}_{n,j}(\boldsymbol{x}_{n',t})}{\sum_{b=1}^{B} p(\boldsymbol{x}_{n',t}|\overline{\lambda}_{n',b})} \tag{16}$$

$$P_{n,j,t} = P(\omega_{n,j}|\boldsymbol{x}_{n,t}) = \frac{c_{n,j} b_{n,j}(\boldsymbol{x}_{n,t})}{p(\boldsymbol{x}_{n,t}|\lambda_n)} \tag{17}$$

where $P(\omega_{n,j}|\boldsymbol{x}_{n,t})$ is the *a posteriori* probability of mixture $\omega_{n,j}$ given the sample $\boldsymbol{x}_{n,t}$. In all of these equations, $\sum_{n'}$ means the

summation over those models for which the current model n has been acting as a background model. It is easy to recognize that if we drop the second terms from the numerators and denominators of Eqs. (11), (12), and (13), we get the standard maximum likelihood re-estimation equations. The second terms of these equations are correction terms corresponding to the correction term of the MNLE objective function of Eq. (4). Since all of the re-estimation equations have similar structures, we can express them in the following general form:

$$\theta^{i+1} = \frac{MLE^i_{numer} - COR^i_{numer}}{MLE^i_{denom} - COR^i_{denom}} \tag{18}$$

where θ^{i+1} represents the new parameter estimate for iteration $i+1$, MLE^i_{numer} and MLE^i_{denom} are terms corresponding to maximum likelihood estimation expressions, and COR^i_{numer} and COR^i_{denom} are correction terms. When used directly, Eq. (18) causes fluctuations of the objective function and there is also a danger of the numerator and denominator terms becoming negative. In order to avoid these problems, in practice, we use a modified version of the above equation [9]:

$$\theta^{i+1} = \frac{ML^0_{numer} - \varepsilon \sum_{j=1}^{i} COR^j_{numer}}{ML^0_{denom} - \varepsilon \sum_{j=1}^{i} COR^j_{denom}} \tag{19}$$

where ML^0_{numer} and ML^0_{denom} are the values obtained after the MLE training and ε denotes a fixed learning coefficient. Our experiments have shown that Eq. (19) leads to monotonic increasing of the objective function up to some point, after which it starts to decrease.

Let's recall that the MNLE objective can be expressed as in Eq. (5). According to this equation, the MNLE objective is a sum of the individual objectives of all classes. Therefore, the maximum of the individual objectives for all classes will maximize the overall objective function. However, we cannot be sure that all individual objectives will reach their maximums at the same iteration (and this was experimentally confirmed). Therefore, it is reasonable to stop the re-estimation process for some particular model when its individual objective function comes to be maximum. With this modification, the MNL training algorithm can be summarized as follows:

Step 1 Begin with MLE trained models. Mark all of the models as to be re-estimated.

Step 2 Compute the new parameters of the models marked to be re-estimated using Eq. (19).

Step 3 Check each class' individual objective function $\mathcal{F}_n(\Lambda)$ and mark the model as not to be re-estimated further if the maximum of $\mathcal{F}_n(\Lambda)$ has been achieved.

Step 4 Repeat Step 2 and Step 3 until all models are no longer marked to be re-estimated.

3. EXPERIMENTS

We implemented the MNLE training algorithm for the task of speaker independent recognition of Japanese syllables in continuous speech. There are 114 such syllables in the Japanese language which we consider as separate classes. Each syllable is modeled by a left-to-right hidden Markov model (HMM). The segmented syllables are obtained automatically from continuously spoken utterances using Viterbi alignment.

A seed left-to-right 5-state HMM was trained by syllable-segmented data from 500 sentences of the ATR speech database and uttered by six male speakers. The observation vectors for each state were represented by one Gaussian distribution with full covariance matrix. They were further refined by maximum *a posteriori* (MAP) estimation using the same 500 sentences uttered by another 30 male speakers. After that, they were retrained by both the MCE/GPD and MNLE algorithms using the same data as for the MAP training.

For the MNLE training, we used a background (cohort) syllable set consisting of 15 syllables (classes) determined using all of the training data in advance.

The speech signal was sampled at 12kHz. The sampled data were blocked into frames of 21.33 ms duration with 8 ms frame shifts. Each frame was then analyzed using 14th order LPC analysis and 10 mel-scale cepstral coefficients were calculated from the LPC coefficients. The feature vectors we used consisted of four successive frames with a reduced dimension of 20 (from the original 40) and the first and second delta cepstral coefficients and energy [10].

As the test data, we used segmented syllables from 939 newspaper article sentences spoken by nine male speakers not used for the HMM training.

Table 1. Recognition rates of training data (%)

Training alg.	ACC.	COR.	SUB.	INS.	DEL.
MAP	61.4	77.2	20.6	15.8	2.2
MCE	62.8	78.9	19.7	16.1	1.4
MNLE	63.9	79.5	18.7	15.6	1.8

Table 2. Recognition rates of test data (%)

Training alg.	ACC.	COR.	SUB.	INS.	DEL.
MAP	50.1	68.1	29.1	18.0	2.8
MCE	48.9	66.5	30.3	17.6	3.2
MNLE	51.3	69.2	28.5	17.9	2.3

Results for the MAP (which served as the baseline), MCE/GPD, and MNLE training algorithms are presented in Table 1 and Table 2. As the results show, the MCE/GPD method is better than MAP only for the training data. For the test data, the MCE/GPD performance degrades, which suggests poor generalization on unseen data for this task. On the other hand, the MNLE algorithm outperforms the MAP and MCE/GPD methods for both the training and test data.

In Fig. 1 and Fig. 2, changes are shown of the training data accuracy recognition rates during the MCE/GPD and MNLE training respectively. As can be seen, the MNLE curves go up monotonically and are much smoother than the MCE/GPD curves. Note that the MCE/GPD method needs about 100 iterations to achieve its best performance. In contrary, the MNLE training achieves more than 50% improvement at the first several iterations and converges at the 17th iteration.

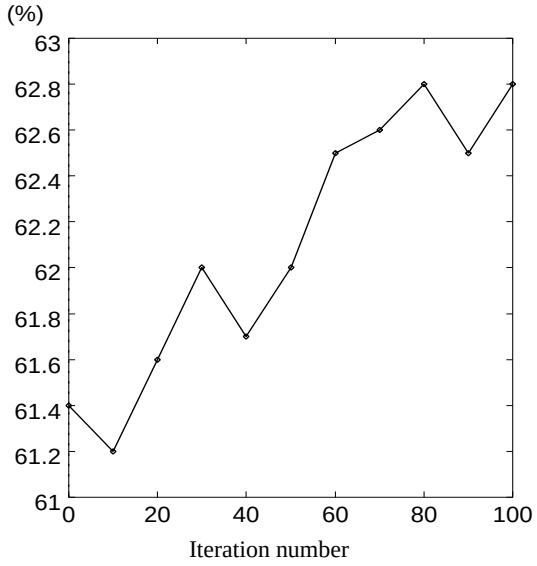

Fig. 1. Training data recognition rate during MCE/GPD training.

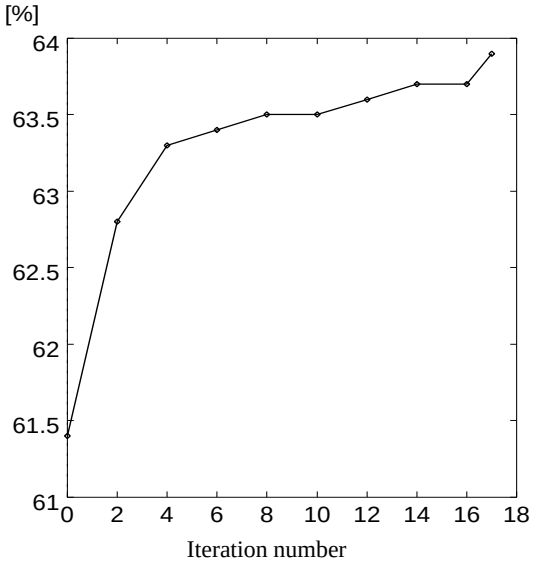

Fig. 2. Training data recognition rate during MNLE training.

4. CONCLUSION

We introduced a new discriminative training method which, in contrast to some other discriminative methods, is based on the well-known EM algorithm.

In a task of syllable recognition in continuous speech, the MNLE training algorithm showed a better performance than both the MAP and MCE/GPD methods, improving the results by 4.1% for the training data and by 2.4% for the test data. The results suggested that MNLE has good generalization abilities on unseen data. It is much faster than the MCE/GPD method since it requires a lower number of training iterations and a lower number of competing classes.

Future work will focus on refinements to this algorithm, such as applying variable and/or class dependent learning steps.

5. ACKNOWLEDGMENT

We are grateful to Mr. Kengo Hanai who performed the speech recognition experiments.

6. REFERENCES

[1] Erik McDermott and Shigeru Katagiri, "Prototype-based minimum classification error/generalized probabilistic descent training for various speech units," *Computer Speech and Language*, vol. 8, pp. 551–368, 1994.

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

[3] Yves Normandin, Regis Cardin, and Renato de Mori, "High-performance connected digit recognition using maximum mutual information estimation," *IEEE Trans. on SAP*, vol. 2, no. 2, pp. 299–311, Apr. 1994.

[4] V. Valtchev, J. Odel, P. Woodland, and S. Young, "MMIE training of large vocabulary recognition systems," *Speech Communication*, vol. 22, no. 4, pp. 303–314, Sept. 1997.

[5] C.M. Alamo, F.J. Gil, C.T. Muninlla, and H. Gomez, "Discriminative training of GMM for speaker identification," in *Proc. ICASSP'96*, 1996, pp. 89–92.

[6] Shigeru Katagiri, Biing-Hwang Juang, and Chin-Hui Lee, "Pattern recognition using a family of design algorithms based upon the generalized probabilistic descent method," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2345–2373, Nov. 1998.

[7] D. Povey and P. C. Woodland, "Frame discrimination training of HMMs for large vocabulary speech recognition," in *Proc. ICASSP*, 1999, pp. 333–336.

[8] Konstantin Petrov Markov and Seiichi Nakagawa, "Discriminative training of GMM using a modified EM algorithm for speaker recognition," in *Proc. ICSLP*, 1998, vol. 2, pp. 177–180.

[9] Konstantin Petrov Markov, *Text-independent speaker recognition based on frame level likelihood transformations*, Ph.D. thesis, Toyohashi University of Technology, Japan, January 1999.

[10] Seiivhi Nakagawa and Kazumasa Yamamoto, "Evaluation of segmental unit input HMM," in *Proc. ICASSP*, 1996, pp. 439–442.