ISCA Archive
http://www.isca-speech.org/archive

International Symposium on Chinese Spoken Language Processing (ISCSLP 2004)
Hong Kong
December 15-18, 2004

MCE-BASED TRAINING OF SUBSPACE DISTRIBUTION CLUSTERING HMM

Xiao-Bing Li, Li-Rong Dai, Ren-Hua Wang

USTC iFly Speech Lab
University of Science and Technology of China, Hefei, Anhui, 230027
lixiaobing@ustc.edu, {lrdai, rhw}@ustc.edu.cn

ABSTRACT

For resource-limited platforms, Subspace Distribution Clustering Hidden Markov Model (SDCHMM) is better than Continuous Density Hidden Markov Model (CDHMM) for its smaller storage and lower computations while maintaining a decent recognition performance. But the normal SDCHMM obtaining method doesn't ensure the optimality in classifier design. In order to obtain an optimal classifier, a new SDCHMM training algorithm that adjusts the parameters of SDCHMM according to Minimum Classification Error (MCE) criterion is proposed in this paper. Our experimental results on TiDigits and RM tasks show the MCE-based SDCHMM training algorithm provides 15-80% Word Error Rate Reduction (WERR) compared with the normal SDCHMM that is converted from CDHMM.

1. INTRODUCTION

Though in many applications CDHMM has the best recognition performance, the required computation and memory resource are unacceptable for some resource-limited systems. In [1][2], SDCHMM is proposed to solve this problem. SDCHMM splits the full feature space into disjoint subspaces and uses a small set of subspace Gaussian distribution prototypes to approximate the original full-space distribution. While one speech frame comes in, the log likelihood of these subspace distributions can be pre-computed just once. And then the state log likelihood can be summed by the pre-computed subspace distribution log likelihoods. All of these bring great reduction in storage and computations with a small degradation of recognition accuracy. The storage and arithmetic operations can be reduced to 6-15% and 30-60%, respectively, compared to the case when original CDHMM is used [2].

Usually SDCHMM is converted from CDHMM by splitting the full-space and clustering the subspace distributions. The converting procedure destroys the integrity of the full feature space and reduces the model resolution. This leads to that the system cannot obtain its optimal performance. In order to remedy this problem in some degree, we propose to use MCE criterion [3][4] to optimize the parameters of SDCHMM, for as we know, MCE can adjust the classification parameters to achieve the classifier's minimum recognition error. We tested our new MCE-based training algorithm on TiDigits and RM tasks, and as we expected, obvious system performance improvement was obtained.

The rest of the paper is organized as follows. The next section describes the concept of SDCHMM and its converting method from CDHMM. Section 3 presents our derivation of MCE-trained formula for SDCHMM parameters. The experimental results are given in section 4. This is followed by a discussion of MCE-trained SDCHMM and MLE-trained SDCHMM in section 5. Finally, section 6 summarizes our work.

2. SUBSPACE DISTRIBUTION CLUSTERING HMM

For CDHMM, the state output probability is given by:

$$b_j(\vec{o}_t) = \sum_{m=1}^{M} c_{jm} b_{jm}(\vec{o}_t) \tag{1}$$

where c_{jm} is the mixture weight, $b_{jm}(\vec{o}_t)$ is the Gaussian mixture distribution with diagonal covariance matrix as variance vector.

Split the full feature space into K streams (subspaces), which are orthogonal and together span the original D dimensional space. Each stream has D_k (k = 1,…,K) dimensions. With diagonal covariance, (1) becomes:

$$b_j(\vec{o}_t) = \sum_{m=1}^{M} c_{jm} \left(\prod_{k=1}^{K} b_{jmk}(\vec{o}_{tk}) \right) \tag{2}$$

where $b_{jmk}(\vec{o}_{tk})$ is the k^{th} stream Gaussian distribution. We cluster the Gaussians of each stream into a small set of P_k prototypes, denoted as:

$$b_{pk}(\vec{o}_{tk}) = \frac{1}{(2\pi)^{\frac{D_k}{2}} \prod_{l=1}^{D_k} \sigma_{pkl}} \exp\left(-\frac{1}{2} \sum_{l=1}^{D_k} \frac{(o_{tkl} - \mu_{pkl})^2}{\sigma_{pkl}^2} \right)$$

$$where \;\; p = 1, \cdots, P_k, \; k = 1, \cdots, K$$

And use the closest prototype $b_{idx(jm,k)}(\vec{o}_{tk})$ to approximate the original distribution $b_{jmk}(\vec{o}_{tk})$. That is:

$$b_{jmk}(\vec{o}_{tk}) \approx b_{idx(jm,k)}(\vec{o}_{tk}) \tag{3}$$

where *idx*(*jm*, *k*) is the prototype index for the k^{th} stream Gaussian distribution of the m^{th} mixture in state *j*. Therefore (2) becomes:

$$b_j(\vec{o}_t) \approx \sum_{m=1}^{M} c_{jm} \left(\prod_{k=1}^{K} b_{idx(jm,k)}(\vec{o}_{tk}) \right) \tag{4}$$

Such HMM model is called as SDCHMM. It is composed of the initial state probabilities, the state transition probabilities, the mixture weights, the prototypes of each stream and the prototype indexes. It also can be seen as a special HMM tying scheme [1].

From the above we know that there are two steps to convert CDHMM to SDCHMM: the splitting-step and the clustering-step.

For the splitting-step, a stream definition is used to split the original full feature space into disjoint subspaces, that is, to divide the mixture Gaussian distributions of CDHMM into disjoint subspace Gaussian distributions. For the conventional 39-dimension MFCC features (12 static MFCCs, log energy, and their first and second order time derivatives), 1-, 4-, 13-, 20- and 39-stream definitions [1] are often used.

For the clustering-step, many clustering algorithms can be used to cluster the distributions of each stream into a small set of prototypes. We use Modified *k*-Means clustering algorithm [2] to perform the clustering.

3. MCE-BASED SDCHMM TRAINING

MCE adjusts the classification parameters to achieve minimum recognition error. Here, we use the MCE criterion to optimize the parameters of SDCHMM.

Let Λ denote the SDCHMM parameters. In Continuous Speech Recognition (CSR), string-model-based discriminant function [3][4] is used. For an input speech utterance, the feature sequence is $O = \{\vec{o}_1, \cdots, \vec{o}_T\}$. Let S_i (i = 1,…,N) denote the top N best competing strings, the corresponding discriminant function is given by:

$$g(O, S_i, \Lambda) = \log f(O, Q_{S_i}, S_i \mid \Lambda) \tag{5}$$

And for the correct string S_{lex}, the discriminant function is:

$$g(O, S_{lex}, \Lambda) = \log f(O, Q_{S_{lex}}, S_{lex} \mid \Lambda) \tag{6}$$

where Q_{S_i} ($Q_{S_{lex}}$) is the optimal state sequencce of the word string S_i (S_{lex}). The misclassification measure is defined as:

$$d(O, \Lambda) = -g(O, S_{lex}, \Lambda) + \log\left\{ \frac{1}{N-1} \sum_{k=1, S_k \neq S_{lex}}^{N} \exp\left(g(O, S_k, \Lambda)\eta\right) \right\}^{1/\eta} \tag{7}$$

where η is a positive number. Then embed it into the sigmoid function: $l(O, \Phi) = 1 / (1 + e^{-\gamma d(O, \Phi) + \theta})$. Normally we set $\gamma \geq 1$, and $\theta = 0$.

The goal of MCE is to minimize the expected loss $L(\Phi) = E_X[l(O, \Phi)]$. This can be solved by the Generalized Probabilistic Descent (GPD) algorithm as:

$$\Lambda_{n+1} = \Lambda_n - \varepsilon_n U \left. \frac{\partial l(O_n, \Lambda)}{\partial \Lambda} \right|_{\Lambda = \Lambda_n} \tag{8}$$

where U is a positive definite matrix, ε_n is the learning step size. The chain rule of differential calculus is used to adjust Λ.

For SDCHMM, we should use the following transformations to maintain the parameter constrains during gradient calculation:

$$a_{ij} \rightarrow \tilde{a}_{ij} \quad where \quad a_{ij} = \frac{\exp(\tilde{a}_{ij})}{\sum_k \exp(\tilde{a}_{ik})} \tag{9}$$

$$c_{jm} \rightarrow \tilde{c}_{jm} \quad where \quad c_{jm} = \frac{\exp(\tilde{c}_{jm})}{\sum_k \exp(\tilde{c}_{jk})} \tag{10}$$

$$\mu_{pkl} \rightarrow \tilde{\mu}_{pkl} = \frac{\mu_{pkl}}{\sigma_{pkl}} \tag{11}$$

$$\sigma_{pkl} \rightarrow \tilde{\sigma}_{pkl} = \log \sigma_{pkl} \tag{12}$$

For the mean and variance, from (8), we have:

$$\tilde{\mu}_{pkl}(n+1) = \tilde{\mu}_{pkl}(n) - \varepsilon_n \left. \frac{\partial l(O_n, \Lambda)}{\partial \tilde{\mu}_{pkl}} \right|_{\Lambda = \Lambda_n} \tag{13}$$

$$\tilde{\sigma}_{pkl}(n+1) = \tilde{\sigma}_{pkl}(n) - \varepsilon_n \left. \frac{\partial l(O_n, \Lambda)}{\partial \tilde{\sigma}_{pkl}} \right|_{\Lambda = \Lambda_n} \tag{14}$$

The gradient calculation is described in detail as follows.

$$\frac{\partial l(O, \Lambda)}{\partial \Lambda} = \frac{\partial l(O, \Lambda)}{\partial d(O, \Lambda)} \frac{\partial d(O, \Lambda)}{\partial \Lambda} \tag{15}$$

$$\frac{\partial l(O, \Lambda)}{\partial d(O, \Lambda)} = \gamma l(O, \Lambda)[1 - l(O, \Lambda)] \tag{16}$$

$$\frac{\partial d(O, \Lambda)}{\partial \Lambda} = -\frac{\partial g(O, S_{lex}, \Lambda)}{\partial \Lambda} + \sum_{i=1, S_i \neq S_{lex}}^{N} \left[\frac{\exp[g(O, S_i, \Lambda)\eta]}{\sum_{j=1, S_j \neq S_{lex}}^{N} \exp[g(O, S_j, \Lambda)\eta]} \frac{\partial g(O, S_i, \Lambda)}{\partial \Lambda} \right] \tag{17}$$

Let S denotes S_i or S_{lex}, and

$$f(O, Q_S, S \mid \Lambda) = \pi_{q_0} \prod_{t=1}^{T} a_{q_{t-1} q_t} b_{q_t}(\vec{o}_t)$$

we have:

$$\frac{\partial g(O,S,\Lambda)}{\partial \tilde{\mu}_{pkl}} = \sum_{t=1}^{T} \delta(q_t - j) b_j^{-1}(\vec{o}_t) \frac{\partial b_j(\vec{o}_t)}{\partial \tilde{\mu}_{pkl}}$$

$$= \sum_{t=1}^{T} \delta(q_t - j) \sum_{m=1}^{M} \delta(idx(jm,k) - pk) \gamma_{jm}(\vec{o}_t) \left(\frac{o_{tkl} - \mu_{pkl}}{\sigma_{pkl}} \right) \quad (18)$$

$$\frac{\partial g(O,S,\Lambda)}{\partial \tilde{\sigma}_{pkl}} = \sum_{t=1}^{T} \delta(q_t - j) b_j^{-1}(\vec{o}_t) \frac{\partial b_j(\vec{o}_t)}{\partial \tilde{\sigma}_{pkl}}$$

$$= \sum_{t=1}^{T} \delta(q_t - j) \sum_{m=1}^{M} \delta(idx(jm,k) - pk) \gamma_{jm}(\vec{o}_t) \left(\left(\frac{o_{tkl} - \mu_{pkl}}{\sigma_{pkl}} \right)^2 - 1 \right) \quad (19)$$

where $\delta(\bullet)$ denotes the Kronecker delta function, and

$$\gamma_{jm}(\vec{o}_t) = c_{jm} \left(\prod_{k=1}^{K} b_{idx(jm,k)}(\vec{o}_{tk}) \right) b_j^{-1}(\vec{o}_t) \quad (20)$$

is the posteriori probability weight of the m^{th} Gaussian mixture in state j.

Finally, use the inverse transform of formula (11) and (12), we can get the updated mean and variance. The updating formulations for the state transition probabilities and the mixture weights can be derived similarly.

4. EXPERIMENTAL RESULTS

We implemented the MCE-based training method on TiDigits and Resource Management (RM) tasks. The conventional 39-dimension MFCC features were used for both tasks. 1-, 4-, 13-, 20- and 39-stream definitions were used. And we use modified *k*-Means clustering algorithm to cluster the Gaussian distributions of each stream.

As we know, compared with the original CDHMM, converted SDCHMM provides a significant reduction in computation load [1][2]. Our MCE-based training method has this advantage too since it is a slight modified approach based on the converted SDCHMM. So we don't list the results of the recognition speed here. The comparisons of the system performance between the converted SDCHMM and the MCE-trained SDCHMM are given in Word Error Rate (WER).

4.1. TiDigits task

TiDigits is a speaker independent, connected digit utterances database [5]. The speech signal was recorded from various regions of the United States. It contains 12,549 strings for training and 12,547 strings for testing. The digits string has a random length from 1 to 7. The model we used is a 10-state, whole-word based HMM model. A 3-state silence and a 1-state short pause models were added. The number of mixtures for each state is two. The WER of baseline CDHMM is 1.81%. The results of the converted SDCHMM with 16 and 32 prototypes per stream and then the MCE-trained SDCHMM were shown in figure 1. We can see that the system performance is degraded severely when converting CDHMM to SDCHMM, especially in low streams. By using MCE criterion to adjust the parameters, as we see, the system performance was significantly improved, and 44-80% WERR over the converted SDCHMM was observed with different stream definitions.

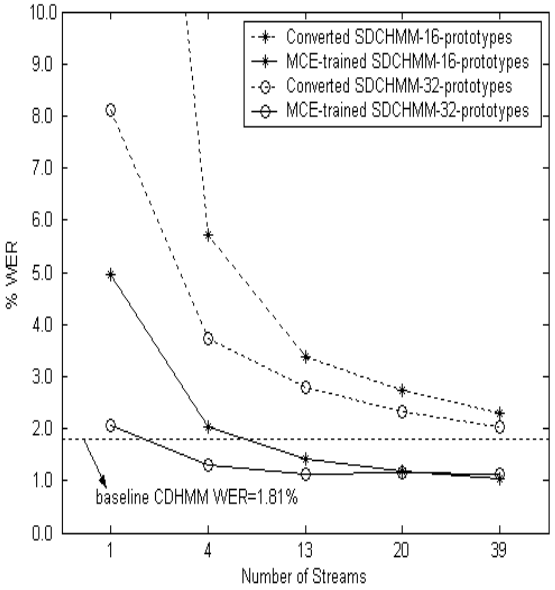

Figure 1: % WER of different SDCHMM on TiDigits task with various stream definitions

4.2. RM task

In order to examine our method further, we tested it on a more complex task: the DARPA 991-word RM task [6]. In this task the standard SI-109 training data set (3,990 utterances) was used. The CMU 48 phone set was used to create the context-independent (CI) models, which with 3 states per phone and 12 mixtures per state. A single pronunciation for each word was derived from the dictionary used by Lee [7]. Figure 2 shows the results on the Feb89 test set. The standard word-pair grammar with an average perplexity of 60 was used for decoding. For the MLE-trained CDHMM, there are 1,740 Gaussian distributions in total, and with a WER of 7.18%. For SDCHMM, we only use 32 and 64 prototypes per stream. Thus, as we see, converting CDHMM to SDCHMM with such few parameters results in severely increment of the recognition error. But through the adjustment with MCE criterion, this is improved to a great degree. Compared to the converted SDCHMM, this new training method provides 15-46% WERR.

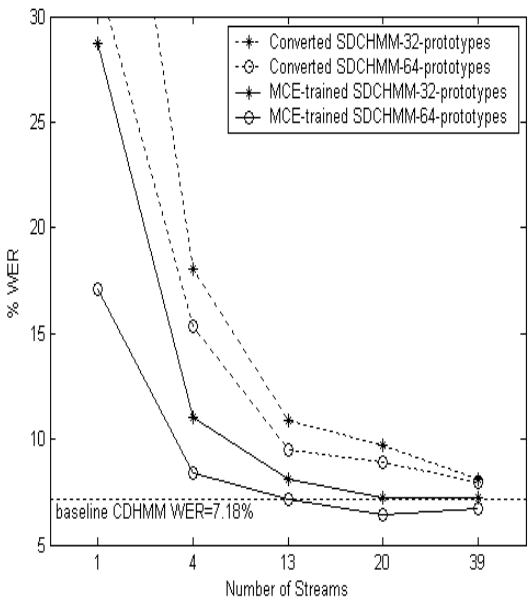

Figure 2: % WER of different SDCHMM on RM task with various stream definitions

5. DISCUSSION

In our experiments, SDCHMM is converted from CDHMM. [1] reported a direct SDCHMM training algorithm: Training SDCHMM parameters with the Maximum Likelihood Estimation (MLE), which is called MLE-based SDCHMM training. Though with much fewer parameters, we can use less data to train MLE-based SDCHMM. The performance of MLE-trained SDCHMM isn't obviously better than that of converted SDCHMM [1]. We didn't do the experiment to compare MLE-trained SDCHMM and our MCE-trained SDCHMM, since our work focuses on the system classification performance improvement instead of how to train using less data.

Though MLE is the optimal distribution estimation method, it is suboptimal to the classifier's design [3]. In contrast, MCE criterion can achieve the optimality of the classifier. Thus, it can be expected that to the same initial SDCHMM, the MCE-trained SDCHMM is more accurate than the MLE-trained one. It can also be expected that if we use MCE criterion to adjust the MLE-trained SDCHMM, we can obtain clearly WERR as we observed in the experiments presented above when the converted SDCHMM is adjusted with MCE criterion. And we'll give our experimental evidence of these expectations in the future.

6. CONCLUSIONS

In this paper, we proposed to optimize the parameters of SDCHMM according to MCE criterion to achieve the minimum recognition error. We implemented our new approach on two tasks, and achieved significant improvement. On TiDigits task, with different stream definitions, MCE-trained SDCHMM provides 44-80% WERR to the converted SDCHMM. 15-46% WERR was observed still on the even more complex RM task.

We carried out our approach with the SDCHMM converted from context-independent CDHMM, and in the near future, we'll test the approach on the SDCHMM converted from context-dependent CDHMM. Excepting this, we'll compare the MCE-trained SDCHMM with the MLE-trained SDCHMM, as mentioned in section 5.

7. REFERENCES

[1] B. Mak, *Towards A Compact Speech Recognizer: Subspace Distribution Clustering Hidden Markov Model*, Ph.D. thesis, Department of Computer Science and Engineering, Oregon Graduate Institute of Science and Technology, April 1998.

[2] E. Bocchieri, and B. Mak, "Subspace Distribution Clustering Hidden Markov Model," *IEEE Trans. Speech and Audio Proc.*, Vol. 9, No. 3: pp. 264-275, March 2001.

[3] B.H. Juang, W. Chou, and C.H. Lee, "Minimum Classification Error Rate Methods for Speech Recognition," *IEEE Trans. Speech and Audio Proc.*, Vol. 5, No. 3: pp. 257-265, May 1997.

[4] W. Chou, "Discriminant-Function-Based Minimum Recognition Error Rate Pattern-Recognition Approach to Speech Recognition," *Proceedings of IEEE*, Vol. 88, No. 8: pp. 1201-1223, August 2000.

[5] R.G. Leonard, "A Database for Speaker-Independent Digit Recognition," *Proc. ICASSP*, Vol. 9: pp. 328-331, March 1984.

[6] P. Price, W.M. Fisher, J. Bernstein, and D.S. Pallett, "The DARPA 1000-word Resource Management Database for Continuous Speech Recognition," *Proc. ICASSP*, Vol. 1: pp. 651-654, April 1988.

[7] K.F. Lee, *Automatic Speech Recognition: The Development of the SPHINX System*, Kluwer Academic, 1989.