OPTIMAL SPLITTING OF HMM GAUSSIAN MIXTURE COMPONENTS WITH MMIE TRAINING

Yves Normandin

Centre de recherche informatique de Montréal (CRIM)
1801 McGill College Avenue, Suite 800, Montréal, Québec, Canada H3A 2N4
Yves.Normandin@crim.ca

ABSTRACT

A novel approach to splitting Gaussian mixture components based on the use of MMIE training is proposed. The idea is to increase acoustic resolution only in those distributions where discrimination problems are identified. Problem mixture components are determined by looking at each mixture weight counter; a large positive counter value indicates both that the component often tends not to be recognized correctly (i.e., is not part of the best path when it should be) and that there is sufficient training data to split the component. Results in a connected digit recognition experiment on the TIDIGITS corpus indicate that much better results can be obtained with such MMIE trained digit models than with MLE trained models that use several times more mixture components.

1. INTRODUCTION

The use of mixtures of Gaussian densities in HMM output distributions has become increasingly popular in the past few years. In the framework of maximum likelihood estimation (MLE) of HMM parameters, Gaussian mixture distributions are often seen as the current best way of approximating, as closely as possible, the "true" underlying distributions.

In practice, however, efficient use of Gaussian mixture distributions is a delicate balancing act between two conflicting requirements. On the one hand, it is important that good enough acoustic resolution be provided by the set of Gaussian components (or kernels). In other words the number of such components should be large enough both to provide adequate coverage of the relevant feature space and to model the fine structure of the underlying distribution. On the other hand, the number of such components should be small enough to ensure that there are enough training data to estimate both the mixture weights and the parameters of the Gaussian densities (mean vector and covariance matrix).

In most state-of-the-art systems, the way out of this dilemma is through the concept of parameter sharing. Examples include "tied mixtures" HMMs [2], in which all output distributions share the same sets of mixture components, "phonetically-tied mixtures" [4] where the same set of mixture components is shared by allophone models of the same phone, "genonic HMMs" [3] where a clustering procedure determines which output distributions will share the same sets of mixture components, or "fully continuous HMMs" [6] in which each output distribution uses different sets of mixture components.

In this paper, we are specifically interested in fully continuous HMMs, which easily allow the number of mixture components to be adjusted on a distribution by distribution basis. For example, a large amount of training data for a distribution would allow a large number of mixture components to be reliably estimated, thereby improving the modeling accuracy for that distribution. This is in contrast to using a fixed number of mixture components per output distribution, as is often done in practice. Using this principle, there are a number of ways in which the number of mixture components per distribution can be determined. For example, it can be determined *a priori*, based on the amount of training data available for each distribution. It can also be slowly increased until recognition results on a validation test set start decreasing (which may be due to insufficient training data for estimating the parameters of each individual mixture component).

There is, however, something unsatisfactory about this whole approach in that a large number of mixture components may end up finely modeling the "interior region" of underlying distributions. This doesn't seem to be very useful insofar as the goal is to improve discrimination between different classes (phonetic or otherwise) in the models. Moreover, it may result in a large number of relatively useless mixture components in the models, whose parameters must nonetheless be properly estimated. This paper proposes one possible solution to this problem based on MMIE training.

2. INTERPRETING MMIE TRAINING

Let's assume we have a set $\mathbf{Y}$ of N training utterances (i.e., sequences of acoustic feature vectors) $\mathbf{Y} \equiv \{Y_n, n = 1, ..., N\}$, with corresponding transcriptions $\mathbf{W} \equiv \{W_n, n = 1, ..., N\}$ (typically words or word sequences). The goal of MMIE training is to maximize the following objective function [1, 5]:

$$P_\Theta(\mathbf{W}|\mathbf{Y}) = \prod_{n=1}^{N} P_\Theta(W_n|Y_n) = \prod_{n=1}^{N} \frac{P_\Theta(Y_n|W_n)P(W_n)}{\sum_{W'} P_\Theta(Y_n|W')P(W')}, \quad (1)$$

0-7803-2431-5/95 $4.00

where $P_\Theta(Y_n|W')$ is the probability that an HMM-based model corresponding to the word sequence W' produced Y_n, $P(W')$ is a probability given by some stochastic language model, and Θ is the set of all HMM parameters to be estimated.

An interpretation of (1) is that MMIE training attempts to maximize discrimination between the correct word sequence and any other competing hypothesis. We usually represent these competing hypotheses using a model M_g defined as:

$$P_\Theta(Y_n|M_g) = \sum_{W'} P_\Theta(Y_n|W')P(W'). \quad (2)$$

That is, M_g is a (typically looped) model containing a path corresponding to every possible word sequence W' in the application, with language model probability $P(W')$. The interest in defining such a model is that in practice, MMIE training then becomes equivalent to doing, for each training utterance Y_n, a Baum-Welch pass using a model built from the transcription W_n which adds to the HMM counts, and a Baum-Welch pass using M_g which subtracts from the counts.

To see this, let us define $\gamma_b(n,t)$ as the *a posteriori* probability that y_{nt}, the t-th frame of the n-th training utterance, was generated by b. Notice that $\gamma_b(n,t)$ is typically used in the Baum-Welch training procedure to reestimate the parameters of b. For example, if b were a diagonal Gaussian density, its mean would be reestimated as:

$$\hat{\mu}_b = \frac{\sum_{n=1}^{N}\sum_t \gamma_b(n,t)y_{nt}}{\sum_{n=1}^{N}\sum_t \gamma_b(n,t)} \quad (3)$$

Whether we use a gradient descent or a reestimation formula to optimize (1), $\gamma_b(n,t)$ will be required. For gradient descent, the derivative $\partial \log P_\Theta(\mathbf{W}|\mathbf{Y})/\partial\theta$ (where θ is an HMM parameter) can be expressed as:

$$\frac{\partial \log P_\Theta(\mathbf{W}|\mathbf{Y})}{\partial\theta} = \sum_{n=1}^{N}\sum_t\sum_b (\gamma_b(n,t) - \gamma_{b,g}(n,t))\frac{1}{b(y_{nt})}\frac{\partial b(y_{nt})}{\partial\theta}, \quad (4)$$

where $\gamma_{b,g}(n,t)$ is the equivalent of $\gamma_b(n,t)$, but with the model M_g used instead of the model built from the transcription (M_{W_n}). Similarly, reestimation formulas [5] can be expressed as:

$$\hat{\mu}_b = \frac{\sum_{n=1}^{N}\sum_t (\gamma_b(n,t) - \gamma_{b,g}(n,t))\, y_{nt} + D\mu_b}{\sum_{n=1}^{N}\sum_t (\gamma_b(n,t) - \gamma_{b,g}(n,t)) + D}, \quad (5)$$

$$\hat{\sigma}_b^2 = \frac{\sum_{n=1}^{N}\sum_t (\gamma_b(n,t) - \gamma_{b,g}(n,t))\, y_{nt}^2 + D\left(\sigma_b^2 + \mu_b^2\right)}{\sum_{n=1}^{N}\sum_t (\gamma_b(n,t) - \gamma_{b,g}(n,t)) + D} - \hat{\mu}_b^2, \quad (6)$$

where D is used to adjust the convergence rate (the larger the value of D, the slower the convergence). It is in fact possible to show that equations (5) and (6) are very similar to a gradient descent, with the main difference that the step size is parameter dependent in that it is proportional to σ_b for $\hat{\mu}_b$ and approximately proportional to σ_b^3 for $\hat{\sigma}_b^2$. We used these reestimation formulas in all our experiments.

3. MMIE-BASED SPLITTING OF MIXTURE COMPONENTS

The reasoning behind MMIE-based splitting of mixture components goes as follows. All paths in M_{W_n}, the model built from the transcription, will also exist in M_g. The difference, however, is that in M_g these paths will compete against paths from all other possible word sequences. In order to understand this better, we can consider the following two extreme cases.

If $P_\Theta(Y_n|W_n) \gg P_\Theta(Y_n|W)$, for $W \neq W_n$, then M_g will be dominated by the paths corresponding to the correct transcription W_n with the result that $\gamma_b(n,t) \approx \gamma_{b,g}(n,t)$ will be true for all distributions. In other words, whether we use gradient descent or reestimation, nearly the same amount will be added to and subtracted from the same counts, with negligible effect on the ultimate value of these counts. If, on the other hand, $P_\Theta(Y_n|W_n) \ll P_\Theta(Y_n|W)$, for some $W \neq W_n$, then some counts in the model for W_n will be incremented while other counts in other models will be decremented.

This is the basis of MMIE-based splitting of Gaussian mixture components. If, after a MMIE training iteration, the mixture weight count for a given mixture component is large and positive, this means that the count was often incremented using the correct transcription, but there were many cases where a different path (i.e., not using this component) was more probable in M_g. This clearly represents a discrimination problem, which we have elected to solve by splitting the mixture component, in order to improve acoustic resolution in its vicinity.

The training algorithm can be summarized as follows:

1. For every training utterance, increment HMM counts with the forward-backward algorithm using the correct transcription and decrement them using model M_g.
2. Find the mixture component with the largest positive mixture weight count.
3. Split all mixture components whose mixture weight count is greater than a certain fraction of the maximum count found above (in our case we used 0.2). In order to separate the resulting mixture components, only one of them will be reestimated.
4. Reestimate the mean and variance parameters of all mixture components, as well as the mixture weights.

4. EXPERIMENTAL RESULTS

4.1. Connected digit corpus

Experiments were performed in the context of a connected digit recognition experiment using the adult portion of the TIDIGITS corpus, from the CD-ROM release. The corpus vocabulary is made of the digits '1' to '9', plus 'oh' and 'zero', for a total of 11 words. Each speaker contributed to

the corpus with two repetitions of each digit in isolation and 55 digit strings, evenly distributed into lengths 2, 3, 4, 5 and 7. This makes a total of 77 digit strings, or 253 digits per speaker. Each string is stored in a separate signal file, with some silence (or background noise) preceding and following the speech signal. The corpus contains 225 speakers (111 men, 114 women). Approximately half the speakers have been assigned to the training set; the remaining half make up the testing set.

4.2. Initial models

The system's front-end computes a 39-feature parameter vector every 10 ms. The speech features used are: 12 mel scaled FFT based cepstral coefficients, along with their first and second derivatives, as well as the log-energy, along with its first and second derivatives. Experiments were done with word models with a left-right topology.

A set of gender-independent digit models using a single Gaussian density per state was first trained with 4 iterations of embedded MLE training (after proper bootstrapping on the word segmentation). These will be used as initial models for both the standard MLE splitting procedure and the proposed MMIE-based splitting procedure.

4.3. MLE trained mixture Gaussian models

Starting from the single Gaussian models, models with a larger number of mixture components were trained using a standard MLE-based splitting procedure. These models will be used as comparison against the MMIE trained models. In the training procedure, the number of mixture components per distribution is increased by splitting all components whose mixture weight count is greater than 0.2 times the largest count for the given distribution. Splitting is done by shifting the original mean vector by ± 0.2 times the standard deviation, while keeping the variance vector fixed, followed by Baum-Welch reestimation. In practice, all components were split, therefore resulting in models with 2, 4, and 8 mixture components per distribution. The recognition performance on the train and test sets is shown in Table 1, where the number in brackets indicates the number of components.

4.4. MMIE trained mixture Gaussian models

Again starting from the models with 1 component per distribution, several iterations of MMIE-split training, as described above, were performed. Figure 1 shows the performance on the train and test sets as a function of the iteration number. After 10 iterations, the average number of mixture components per distribution is under 2.5, but the performance is much better than that of the MLE models with 4, or even 8 components per distribution.

In order to compare the proposed approach with the more standard approach of using MMIE training to optimize MLE-trained mixture Gaussian models, we performed two additional experiments. Starting with MLE models with, respectively, 2 and 4 mixture components, 10 standard MMIE training iterations were performed to optimize their performance. The result was that the performance obtained was still significantly worse than that of the models trained with MMIE-based optimal splitting.

In fact, a revealing illustration of the technique's effectiveness can be seen in Figure 2, which shows the convergence of $\log P_\Theta(\mathbf{W}|\mathbf{Y})$ for the three MMIE experiments (optimal splitting, MMIE with 2 components, MMIE with 4 components).

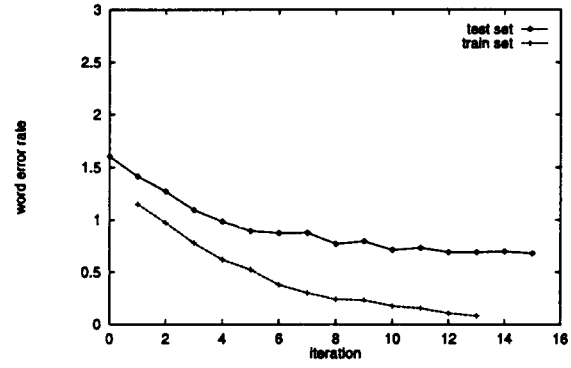

Figure 1: Error rate as a function of the MMIE training iteration.

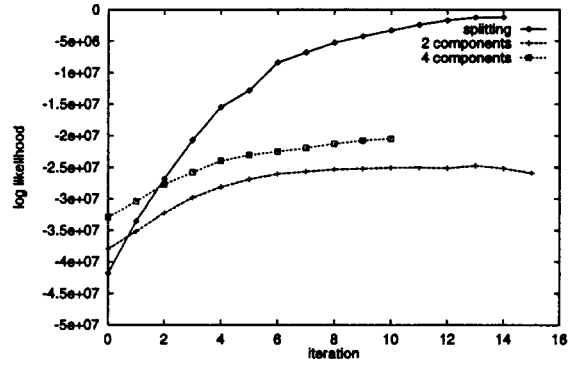

Figure 2: Value of the objective function $\log P_\Theta(\mathbf{W}|\mathbf{Y})$ as a function of the iteration number.

Table 2 shows the number of mixture components for each HMM state after 10 iterations of MMIE-based splitting. It is interesting to see that the number of mixture components per state is anything but uniform. Although it is not easy to draw general conclusions from the way components are distributed, it seems that models for problem digits such as 'oh' ended up with many more components than the others.

As a final comment, it is also important to realize that not only is the number of components important, but so is their location in feature space. In particular, discriminative training techniques will tend to place the mixture components on the border between confusable classes, rather than inside the classes. In order to demonstrate this, we started with the models obtained after 10 iterations of MMIE-based

splitting and we performed 3 MLE iterations. Figure 3 shows the recognition performance after each of these iterations. The effect is quite dramatic: any gain that was obtained through the MMIE splitting/training procedure is completely lost. As seen in Table 1, the performance is somewhere between that of the MLE models with 2 and 4 mixture components, as if MMIE had never been used.

	train set		test set	
	W.Err	W.Corr	W.Err	W.Corr
MLE (1)	1.15	98.9	1.60	98.4
MLE (2)	-	-	1.44	98.6
MLE (4)	-	-	1.22	98.8
MLE (8)	-	-	1.00	99.0
MMIE (2)	-	-	1.14	99.0
MMIE (4)	-	-	0.98	99.2
MMIE-split (2.5)	0.17	99.8	0.71	99.4
MLE (2.5)	1.03	99.0	1.31	98.7

Table 1: Summary of results.

model	number of mixture components per state
one	1 1 2 2 2 3 2 2 2 2 2 1 1 1 2 1 1 1
two	2 2 1 1 1 1 1 1 1 2 1 1 1 2 1 1 1 2
three	3 2 2 1 2 2 2 3 3 3 4 3 7 3 4 2 2 2
four	1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 2 1 1 1 1
five	5 2 1 1 1 1 1 1 1 2 1 1 4 1 1 1 1 1 1 2 4
six	2 2 2 3 4 4 4 4 4 6 4 4 4 5 4 3 4 4 3 3 3 1 2 3
seven	1 2 1 3 3 3 3 3 3 3 3 2 3 3 3 2 2 1 2 3 3 4 3 3 4 3 1
eight	2 2 4 1 2 1 1 3 4 4 3 4 1 2 2
nine	1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 3
oh	2 7 7 7 6 4 5 5 7 5 8 7 7 8 5
zero	4 5 8 3 3 3 3 4 4 3 4 4 3 4 2 2 2 3 3 5 3 3 3 2 2 2 1
sil	3 1 1 1 3
pau	2

Table 2: Number of mixture components for each HMM state after 10 iterations of MMIE based splitting. For each model, the number of mixture components per state is given for each state of the model, listed from left to right. As can be seen, the number of states per model is variable.

5. CONCLUSION

The proposed technique was shown to make very efficient use of a small number of Gaussian mixture components, obtaining substantially better performance than MLE-trained models with several times more mixture components. The results obtained were very good, considering the fact that none of the techniques that gave us our best results on this task (gender-specific models, cross-word context-dependent models, parameter weighting) were used in this study.

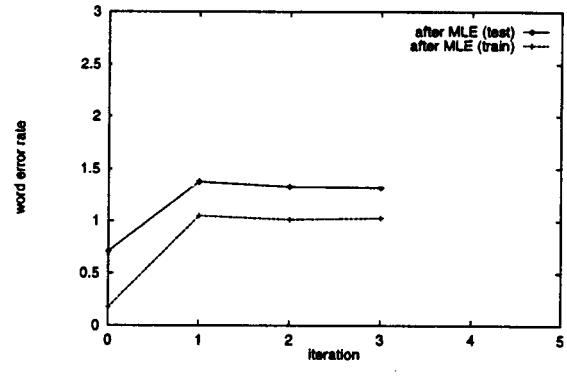

Figure 3: Error rate as a function of the MLE training iteration when starting from the 10-th iteration MMIE models (iteration 0, in the figure).

6. REFERENCES

[1] L.R. Bahl, P.F. Brown, P.V. de Souza and R.L. Mercer, "Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition," *Proc. ICASSP-86*, pp. 49-52, Tokyo, 1986

[2] J.R. Bellegarda and D. Nahamoo, "Tied Mixtures Continuous Parameter Modeling for Speech Recognition," *Proc. ICASSP-89*, pp. 13-16, Glasgow, 1989

[3] V. Digalakis and H. Murveit, "High-Accuracy Large-Vocabulary Speech Recognition Using Mixture Tying and Consistency Modeling," *Proceedings of the ARPA Human Language Technology Workshop*, March 1994

[4] C.H. Lee, L.R. Rabiner, R. Pieraccini, and J.G. Wilpon, "Acoustic Modeling for Large Vocabulary Speech Recognition," *Computer Speech and Language*, vol. 4, no. 2, April 1990

[5] Y. Normandin, "Hidden Markov Models, Maximum Mutual Information Estimation, and the Speech Recognition Problem," *Ph.D. Thesis*, McGill University, Montreal, June 1991

[6] S. Young, J. Odell, and P. Woodland, "Tree-Based State Tying for High Accuracy Acoustic Modelling," *Proceedings of the ARPA Human Language Technology Workshop*, March 1994