JOINT MCE ESTIMATION OF VQ AND HMM PARAMETERS FOR GAUSSIAN MIXTURE SELECTION

Shawn M. Herman and Rafid A. Sukkar

Lucent Technologies Bell Laboratories
2000 N. Naperville Rd., Naperville, IL 60566, USA
{smherman,sukkar}@lucent.com

ABSTRACT

Vector Quantization (VQ) has been explored in the past as a means of reducing likelihood computation in speech recognizers which use hidden Markov models (HMMs) containing Gaussian output densities. Although this approach has proved successful, there is an extent beyond which further reduction in likelihood computation substantially degrades recognition accuracy. Since the components of the VQ frontend are typically designed after model training is complete, this degradation can be attributed to the fact that VQ and HMM parameters are not *jointly* estimated. In order to restore the accuracy of a recognizer using VQ to aggressively reduce computation, joint estimation is necessary. In this paper, we propose a technique which couples VQ frontend design with Minimum Classification Error training. We demonstrate on a large vocabulary subword task that in certain cases, our joint training algorithm can reduce the string error rate by 79% compared to that of VQ mixture selection alone.

1. INTRODUCTION

Current state of the art speech recognizers use continuous density hidden Markov models (HMMs) to represent acoustic events. Unfortunately, the intensive nature of the state likelihood calculations required by such models can limit their use in real-time high channel density applications. In [1–3], Vector Quantization (VQ) was explored as a means of reducing the computational cost of using continuous state output distributions which were represented by Gaussian mixture densities. The general idea behind using VQ to reduce computation is the following: If an observation vector can be assigned to a certain region of the acoustic space via VQ, those mixtures which are "far away" from this region can be neglected since they will make negligible contributions to subsequent likelihoods. In speech recognition, VQ is typically implemented as a frontend consisting of a codebook for quantizing observation vectors and a neighbor table for identifying which mixtures are "near" each VQ codevector. Typically, the VQ codebook and neighbor table are designed offline after the HMM training process is complete.

In [2], it was found that vector quantization could reduce the average number of mixture likelihood calculations per frame by over 70% on a large vocabulary subword task without a significant loss of accuracy. This result is understandable if we assume that the mixtures within each HMM state can be clustered into "substates" which represent mutually exclusive acoustic events. Thus, as long as the VQ frontend permits the mixture likelihood calculations corresponding to the correct substate for the current pair of HMM state and VQ index, recognition accuracy is not degraded. However, if the desired reduction in likelihood computation requires mixtures *within* the selected substate to be excluded, recognition accuracy will decline. In this sense, some amount of mixture reduction is typically achievable with minimal performance degradation, and the vector quantizer design can be conducted apart from the HMM training. However, as the extent of reduction becomes aggressive enough to impact this substate level, recognition accuracy degrades significantly. To restore recognition accuracy, joint estimation of VQ and HMM parameters is necessary .

In this paper, we introduce an algorithm which couples VQ frontend design with Minimum Classification Error (MCE) training. The joint estimation permits the model parameters to be adjusted to maximize discrimination between competing HMMs for a recognizer which is using VQ based mixture selection to reduce computation. This has the effect of enabling aggressive mixture selection without significantly reducing accuracy.

The outline of the rest of the paper is as follows. In Section 2, we review some of the fundamentals of vector quantization and how they apply to Gaussian mixture selection. In Section 3, we describe how our algorithm couples VQ mixture selection with MCE training, while in Section 4, we discuss how the MCE algorithm is modified to enable joint estimation. In Section 5, we present experimental evidence on a large vocabulary subword task demonstrating that the string accuracy of a recognizer using VQ based Gaussian mixture selection can be improved significantly through joint VQ/MCE training. In Section 6, we conclude.

2. VQ BASED GAUSSIAN MIXTURE SELECTION

Consider a vector of HMM parameters, $\mathbf{\Lambda}$, corresponding to a model set containing a total of S unique states and M unique mixture densities. Let M_s denote the set of indices for the mixture components within state s and $\mathcal{N}(\cdot\,;\,\mu_m, \sigma^2_m)$ represent a Gaussian density with mean μ_m and variance σ^2_m (we assume diagonal covariance matrices throughout). The likelihood of observing vector x given state s can then be expressed as

$$l_s(x) = \sum_{m \in M_s} c_m \mathcal{N}(x\,;\,\mu_m, \sigma^2_m) \tag{1}$$

where $\sum_{m \in M_s} c_m = 1$.

We wish to design a VQ frontend consisting of a size-N codebook, C_N, and a $N \times M$ neighbor table, $T_{N,M}$. During recognition, C_N is used to provide a quantization index, i, for each observation vector, x, while $T_{N,M}$ maps i to a set of non-negligible mixtures which are needed for state likelihoods involving x. To

design C_N, we adopt the approach introduced in [1]. Define as distance measure

$$\delta(x, y) = \frac{1}{d} \sum_{j=1}^{d} \frac{(x(j) - y(j))^2}{w(j)} \quad (2)$$

where x and y are d-dimensional vectors and $w(j)$ is the average of the j^{th} variance component over all σ_m^2. With this notation,

$$C_N = \{y_i \ : \ i = 1, \ldots, N\} \quad (3)$$

is designed by clustering the vectors $\mu_m \in \mathbf{\Lambda}$ in an unsupervised manner with δ as distance measure.

For the design of $T_{N,M}$, we use the variable threshold technique introduced in [2]. Recall that we do not wish to calculate mixture likelihoods that will have negligible effects on their respective state likelihoods. Thus, we define a maximum distance, Θ_i, that a mixture mean may lie from a VQ codeword before being considered negligible in likelihood calculations involving observation vectors quantized into cell i. More specifically, if Q denotes the quantization function and $Q(x) = i, \mathcal{N}(\cdot \ ; \ \mu_m, \sigma_m^2)$ will be considered negligible in terms of $l_s(x)$ if

$$\delta(y_i, \mu_m) \ > \ \Theta_i \quad \text{where } m \in M_s. \quad (4)$$

With this notation, an entry in the neighbor table is defined as

$$T_{N,M}(i, m) = \begin{cases} 0 & \text{if } \delta(y_i, \mu_m) \ > \ \Theta_i, \\ 1 & \text{if } \delta(y_i, \mu_m) \ \leq \ \Theta_i, \end{cases} \quad (5)$$

where $i = 1, 2, \ldots, N$ and $m = 1, 2, \ldots, M$. Once $T_{N,M}$ has been populated, the reduced mixture approximation to $l_s(x)$ can be formulated as

$$\hat{l}_s(x) = \sum_{m \in M_s} T_{N,M}(i, m) \ c_m \ \mathcal{N}(x \ ; \ \mu_m, \sigma_m^2) \quad (6)$$

where $Q(x) = i$. Assuming N is large enough and C_N is well designed, careful choice of each Θ_i should ensure $\hat{l}_s(x) \approx l_s(x)$ for all x.

3. JOINT ESTIMATION OF VQ AND HMM PARAMETERS USING MCE TRAINING

Our algorithm proceeds in an iterative fashion by alternately designing $\{C_N, T_{N,M}\}$ for a given $\mathbf{\Lambda}$, and then optimizing $\mathbf{\Lambda}$ for that $\{C_N, T_{N,M}\}$. More specifically, we initialize the process with a boot model, $\mathbf{\Lambda}^{(0)}$, that has been discriminatively trained with *full* likelihood computation permitted. After using $\mathbf{\Lambda}^{(0)}$ to design $\{C_N^{(0)}, T_{N,M}^{(0)}\}$ as described in Section 2, $\mathbf{\Lambda}^{(0)}$ is then adjusted using the modified MCE algorithm described in the next section in order to reduce classification error of the recognizer *when using VQ based Gaussian mixture selection.* We denote this improved model set as $\mathbf{\Lambda}^{(1)}$. Since the VQ codebook is created from the HMM parameters, a new (more optimal) VQ codebook and neighbor table, $\{C_N^{(1)}, T_{N,M}^{(1)}\}$, is then designed corresponding to $\mathbf{\Lambda}^{(1)}$. This entire procedure is repeated until the performance degradation introduced by VQ based Gaussian mixture selection is minimized. Figure 1 depicts this relationship. Thus, our iterative algorithm is composed of the following two steps: (1) design $\{C_N, T_{N,M}\}$ given $\mathbf{\Lambda}$ and (2) improve the reduced mixture performance of $\mathbf{\Lambda}$ given $\{C_N, T_{N,M}\}$.

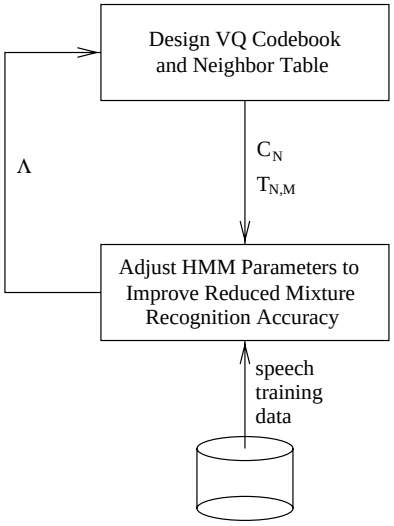

Figure 1: Illustration of the joint VQ/MCE training algorithm.

4. MODIFIED MCE TRAINING FOR IMPROVING $\mathbf{\Lambda}$ GIVEN $\{C_N, T_{N,M}\}$

Having already detailed how to design $\{C_N, T_{N,M}\}$ given $\mathbf{\Lambda}$ in Section 2, we now proceed to discuss our modification of the MCE training algorithm. Given a vector of HMM parameters, $\mathbf{\Lambda}$, and its corresponding VQ frontend, $\{C_N, T_{N,M}\}$, we wish to increase the discrimination between competing models for a recognizer which is using VQ based Gaussian mixture selection. In [4], the MCE training algorithm was introduced. The algorithm uses a smoothed approximation to the classification error count which enables gradient descent methods to operate. We need to modify its implementation slightly so that the algorithm can operate upon our reduced mixture recognizer.

To summarize the development in [4], let W be an arbitrary word string composed by concatenating models from $\mathbf{\Lambda}$. If X is a finite sequence of observation vectors, we denote the log-likelihood score of X along its optimal path through the models composing W as $g(X, W, \mathbf{\Lambda})$. The top N string hypotheses can be defined inductively as

$$W_1 = \arg\max_{W} \ g(X, W, \mathbf{\Lambda}) \ , \quad (7)$$

$$W_k = \arg\max_{W \neq W_1, \ldots, W_{k-1}} \ g(X, W, \mathbf{\Lambda}) \ . \quad (8)$$

If N_{best} is the number of N-best string hypotheses provided by the recognizer, the misclassification measure proposed in [4] can then be defined as

$$d(X, \mathbf{\Lambda}) = -g(X, W_{lex}, \mathbf{\Lambda}) + \log \left[\frac{1}{N_{best} - 1} \sum_{W_k \neq W_{lex}} e^{g(X, W_k, \mathbf{\Lambda})\eta} \right]^{\frac{1}{\eta}} \quad (9)$$

where η is a positive number, and W_{lex} is the correct string. A misclassification error is indicated by $d(X, \mathbf{\Lambda}) \gg 0$. This misclassification measure is then embedded in a smoothed loss function

defined as

$$l(X,\mathbf{\Lambda}) = \frac{1}{1+e^{-\gamma d(X,\mathbf{\Lambda})}} \tag{10}$$

where γ is positive. The MCE algorithm proceeds by adjusting $\mathbf{\Lambda}$ according to the gradient of the loss function, $\nabla l(X,\mathbf{\Lambda})$ [4].

In order for the MCE algorithm to reduce classification error for a recognizer using Gaussian mixture selection, $g(X,W_{lex},\mathbf{\Lambda})$ and $g(X,W_k,\mathbf{\Lambda})$ in (9) need to be replaced by the optimal log-likelihood scores of X along paths within the reduced mixture representations of W_{lex} and W_k. This implies that $\{C_N, T_{N,M}\}$ must be used when scoring X for each W. We denote these approximations to the actual likelihoods as $\hat{g}(X,W_{lex},\mathbf{\Lambda})$ and $\hat{g}(X,W_k,\mathbf{\Lambda})$, respectively. However, use of $\hat{g}(X,W_{lex},\mathbf{\Lambda})$ and $\hat{g}(X,W_k,\mathbf{\Lambda})$ in (9) alters the update formulas of the various model parameters through $\nabla l(X,\mathbf{\Lambda})$. Our new update formulas are related to the established ones through the partial derivatives

$$\frac{\partial\,\hat{l}_s(x)}{\partial\mu_m} = T_{N,M}(i,m)\frac{\partial\,l_s(x)}{\partial\mu_m}\,, \tag{11}$$

$$\frac{\partial\,\hat{l}_s(x)}{\partial\bar{\sigma}^2_m} = T_{N,M}(i,m)\frac{\partial\,l_s(x)}{\partial\bar{\sigma}^2_m}\,, \tag{12}$$

$$\frac{\partial\,\hat{l}_s(x)}{\partial\bar{c}_m} = c_m\{\,T_{N,M}(i,m)\,\mathcal{N}(x\,;\,\mu_m,\sigma^2_m) - l_s(x)\,\} \tag{13}$$

where $Q(x)=i$ and $\bar{\sigma}^2_m$ and $\bar{c}_m$ are related to the original HMM parameters through the following equations:

$$\sigma^2_m = e^{\bar{\sigma}^2_m}\,, \tag{14}$$

$$c_m = \frac{e^{\bar{c}_m}}{\sum_{m\in M_s} e^{\bar{c}_m}}\,. \tag{15}$$

The update formula for transition probabilities is not affected by VQ mixture selection.

5. EXPERIMENTAL RESULTS

In this section we will demonstrate how our joint VQ/MCE training algorithm can restore much of the recognition performance lost through aggressive VQ based Gaussian mixture selection. We also investigate the empirical rate of convergence of our training technique. We consider the task of speaker independent subword based large vocabulary recognition. The recognizer lexicon consisted of 6963 company names whose lexical representations were obtained from either dictionary lookup or a text-to-speech frontend. The average phoneme length for a company name was 18.9, and the average word length was 3.7. The testing database consisted of 3913 utterances from 843 speakers collected over the U.S. telephone network. The model set for this task consisted of 41 context independent subword HMMs, one of which represented silence. All non-silence HMMs were 3 state with 16 Gaussian mixtures while the silence HMM was 1 state with 32 Gaussian mixtures. The initial model set, $\mathbf{\Lambda}^{(0)}$, was built using standard MCE training (i.e., with full likelihood computation) on a database consisting of 9865 phonetically balanced sentences and phrases. The feature vector for the task consisted of the following 39 parameters: 12 LPC derived cepstral coefficients, 12 delta cepstral coefficients, 12 delta-delta cepstral coefficients, normalized log energy, delta log energy, and delta-delta log energy.

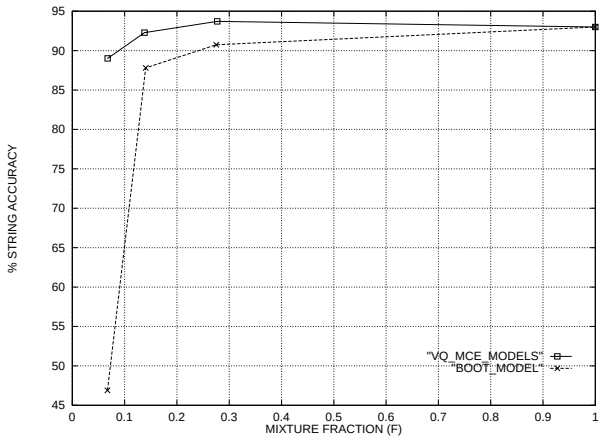

Figure 2: Percent string accuracy versus mixture fraction for a recognizer using $\mathbf{\Lambda}^{(0)}$ and $\mathbf{\Lambda}^{(5)}_F$.

In our experiments, we considered 3 different levels of mixture reduction in order to vary the extent of degradation caused by VQ. The level of mixture reduction is indicated by the mixture fraction which we define as

$$F = \frac{\text{total \# mixture likelihoods computed with VQ}}{\text{total \# mixture likelihoods computed without VQ}} \tag{16}$$

where the totals in (16) are computed over all strings in the test set. At each value of F, we performed 5 iterations of joint VQ/MCE training using the same phonetically balanced training database described earlier. This provided us with 15 new model sets, $\mathbf{\Lambda}^{(j)}_F$, where $j=1,2,\ldots,5$ refers to the training iteration and the subscript F identifies the targeted mixture fraction. During our modified MCE adjustment of $\mathbf{\Lambda}^{(j)}_F$, $\{C^{(j)}_N, T^{(j)}_{N,M}\}$ was updated every 20 strings to assure convergence. Therefore, we now take iteration to mean a complete pass through all 9865 training strings, not a single cycle through Figure 1 (i.e., a single iteration now represents 493 cylces through Figure 1). In all cases, a size-128 VQ codebook was used. In our results, we will compare string accuracy after each VQ/MCE iteration to that achieved by $\mathbf{\Lambda}^{(0)}$ with VQ based Gaussian mixture selection. (Note that $\mathbf{\Lambda}^{(0)}$ does not have a subscript F since the boot model does not vary with mixture fraction.)

5.1. Improvement in String Accuracy

In this section, we demonstrate how joint VQ/MCE training can significantly improve the accuracy of a recognizer which is using VQ based Gaussian mixture selection. The 3 mixture fractions we considered correspond to selecting approximately 1, 2, and 4 non-silence mixtures per state. Figure 2 presents the string accuracies at each of these mixture fractions for a recognizer using $\mathbf{\Lambda}^{(0)}$ and $\mathbf{\Lambda}^{(5)}_F$. As expected, the string accuracy attainable by $\mathbf{\Lambda}^{(0)}$ decreases as likelihood computation is reduced, but beneath a mixture fraction of approximately 0.1, accuracy falls significantly to 46.9%. Clearly, at $F=0.07$, the extent of mixture reduction is so aggressive that non-negligible mixtures are being discarded. At this lowest mixture fraction, we find that our joint VQ/MCE training is able to reduce the string error rate by 79% and increase

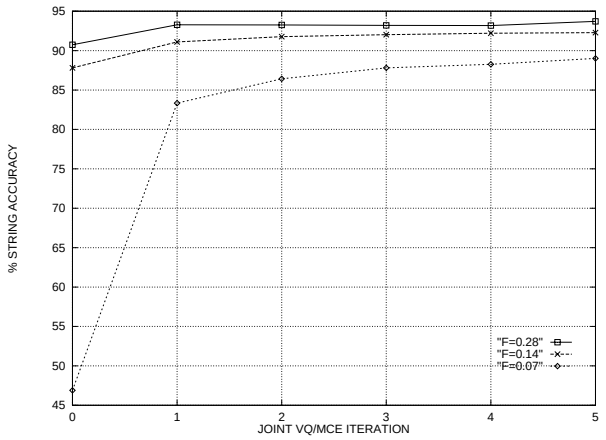

Figure 3: Percent string accuracy versus joint VQ/MCE iteration for various mixture fractions.

string accuracy from 46.9% to 89.0%. At the other 2 mixture fractions, our technique increases accuracy from 87.8% to 92.3% and from 90.7% to 93.7%. Since $\mathbf{\Lambda}^{(0)}$ with full likelihood computation attains an accuracy of 93.0%, we find that our technique is indeed able to remove most of the degradation introduced by VQ based Gaussian mixture selection. (The data point at $F = 1.0$ in Figure 2 represents the accuracy attainable using $\mathbf{\Lambda}^{(0)}$ with *full* likelihood computation.)

5.2. Empirical Rate of Convergence

Since our training algorithm is iterative in nature, it is useful to investigate its empirical rate of convergence. To accomplish this, we plot the string accuracy versus VQ/MCE iteration at the 3 mixture fractions previously considered. The curves in Figure 3 are labeled according to their targeted value of mixture fraction. We see that in all 3 cases, the majority of the degradation introduced by VQ based Gaussian mixture selection is removed with just 3 iterations of the proposed VQ/MCE training technique. Thus, if the training database for the selected task is extensive, our algorithm will impose very little overhead in design time.

6. CONCLUSION

Although vector quantization offers a valuable means of reducing likelihood computation, we showed that aggressive VQ based Gaussian mixture selection can degrade recognition accuracy significantly. This loss of accuracy can be attributed to the fact that the VQ frontend is designed after the HMM training is complete. In this paper, we have proposed a training technique which *jointly* estimates the VQ and HMM parameters using MCE training. We provided experimental evidence on a large vocabulary subword task that demonstrated that our technique was able to remove most of the degradation introduced by VQ mixture reduction at the aggressive mixture fractions we considered. Furthermore, we demonstrated that our joint VQ/MCE training had a rapid empirical rate of convergence so that only a few iterations were needed to achieve substantial improvements in accuracy.

Acknowledgements

The authors would like to thank Carl Mitchell for his valuable software support.

7. REFERENCES

[1] E. Bocchieri, "Vector quantization for the efficient computation of continuous density likelihoods," in *Proc. ICASSP*, vol. 2, pp. 692–695, Apr. 1993.

[2] S. M. Herman and R. A. Sukkar, "Variable threshold vector quantization for reduced continuous density likelihood computation in speech recognition." To appear in *Proc. Automatic Speech Recognition and Understanding Workshop '97*.

[3] K. M. Knill, M. J. F. Gales, and S. J. Young, "Use of gaussian selection in large vocabulary continuous speech recognition using hmms," in *Proc. ICSLP*, vol. 1, pp. 470–473, Oct. 1996.

[4] W. Chou, C.-H. Lee, and B.-H. Juang, "Minimum error rate training based on n-best string models," in *Proc. ICASSP*, vol. 2, pp. 652–655, Apr. 1993.