DECISION-TREE BASED QUANTIZATION OF THE FEATURE SPACE OF A SPEECH RECOGNIZER

M. Padmanabhan, L. R. Bahl, D. Nahamoo, P. de Souza
IBM T. J. Watson Research Center
P. O. Box 218, Yorktown Heights, NY 10598

1 ABSTRACT

We present a decision-tree based procedure to quantize the feature-space of a speech recognizer, with the motivation of reducing the computation time required for evaluating gaussians in a speech recognition system. The entire feature space is quantized into non overlapping regions where each region is bounded by a number of hyperplanes. Further, each region is characterized by the occurence of only a small number of the total alphabet of allophones (sub-phonetic speech units); by identifying the region in which a test feature vector lies, only the gaussians that model the density of allophones that exist in that region need be evaluated.

The quantization of the feature space is done in a heirarchical manner using a binary decision tree. Each node of the decision tree represents a region of the feature space, and is further characterized by a hyperplane (a vector $\underline{v}_n$ and a scalar threshold value h_n), that subdivides the region corresponding to the current node into two non-overlapping regions corresponding to the two children of the current node. Given a test feature vector, the process of finding the region that it lies in involves traversing this binary decision tree, which is computationally inexpensive.

We present results of experiments that show that the gaussian computation time can be reduced by as much as a factor of 20 with negligible degradation in accuracy.

2 INTRODUCTION

We present a decision-tree based procedure to quantize the feature-space of a speech recognizer. Typically, in speech recogition systems, given a feature vector the conditional probability of the feature vector has to be obtained for several phonetic classes. This is often done by modelling the density of each class as a mixture of gaussians, and evaluating the probability of the feature vector for each of these gaussians [1]. It is not unusual to use tens or even hundreds of thousands of different gaussians in such models. This represents a very large computational burden, consequently, methods to reduce this computation are an important focus of current research. Prior techniques to reduce this computation involved vector quantization of the gaussians, or gaussian selection where the full set of gaussians were organized into groups via some heirarchical scheme; and for a test feature vector, only the gaussians belonging to one of the groups were evaluated [2, 3, 4]. We present an alternative technique here, that is shown to reduce the gaussian computation by a factor of 20 with very little degradation in word error rate.

The entire feature space is quantized into non overlapping regions where each region is the intersection of a number of hyperplanes. Further, each region is characterized by the occurence of only a small number of the total alphabet of allophones (sub-phonetic speech units); consequently, if it can be determined that a given test feature vector lies in a particular region, then only the gaussians that model the density of allophones that exist in that region need be evaluated.

The quantization of the feature space is done in a heirarchical manner using a binary decision tree. Each node of the decision tree represents a region of the feature space, and is further characterized by a hyperplane that divides the region corresponding to the current node into two non-overlapping regions corresponding to the two children of the current node [1]. The hyperplane associated with the node is defined by a vector and a scalar threshold value, hence if the

[1] A similar technique was reported in [5] for the application of lexical access, however the decision tree in [5] was constructed using phonetic-context questions to partition the training data, and subsequently designing a hyperplane to best partition the acoustic data corresponding to the classes, whereas in this paper we present a more direct and completely different criterion for constructing the tree.

inner product of a feature vector with the hyperplane vector is less than the threshold value, the feature vector is hypothesized to lie on one side of the hyperplane, and if the inner product is larger than the threshold, the feature vector is hypothesized to lie on the other side of the hyperplane.

Given a test feature vector, the process of finding the region that it corresponds to involves traversing this binary decision tree, and is computationally inexpensive.

3 CONSTRUCTION OF THE DECISION TREE

We start the process with a large amount of labelled training data, i.e., a sequence of feature vectors and the allophones that each of them are assigned to (obtained from a viterbi alignment). It is possible to compute the empirical distribution of allophone classes in the training data, and the objective in the construction of the decision tree is to partition the feature space using linear hyperplanes, such that the average of the entropy of the allophone classes in each partition is minimized.

Before going into the details of the procedure, we will first define the notation that will be used: underlined variables will be used to denote vectors, and double-underlined variables will be used to denote matrices. Let $\underline{x}_t^n$ denote the t^{th} feature vector at node n of the decision tree, and l_t^n) denote the corresponding allophone that the feature vector has been aligned to.

The procedure for growing the tree starts with all the aligned training data at the root node of the tree ($n = 1$).

(i) From the training data at node n, compute the empirical distribution of the allophones at the current node, and compute the entropy, H_n, of this distribution.

The objective is to repartition the training data at the current node, n, into two child nodes, n_1 and n_2, such that the average entropy of the allophone distribution at the child nodes is minimized.

(ii) Next, compute a linear discriminant, $\underline{v}_n$, based on the training data at node n. From [6], this is simply the leading eigenvector of the matrix $\underline{\underline{W}}_n^{-1}\underline{\underline{T}}_n$, where $\underline{\underline{W}}_n$ and $\underline{\underline{T}}_n$ are, respectively, the average between-covariance and the total-covariance matrices of the training data at node n.

The design of the linear discriminant may also be interpreted as follows : let us consider the inner product of the discriminant vector $\underline{v}_n$ and a feature vector, $\underline{x}_t^n$, $s_t^n = \underline{v}_n^T \underline{x}_t^n$. Further, assume that the value of the scalar s_t^n lies in the range (l_c, u_c) for the vectors that correspond to allophone class c, i.e., for which $l_t^n = c$. The design of $\underline{v}_n$ tries to make these range of values maximally non-overlapping for different values of c; in other words, the scalar s_t^n maximally separates the allophones at that node.

(iii) Next, pick a threshold, h, that can be compared to s_t^n such that all feature vectors that have s_t^n less than the threshold are assigned to the left child node, and the feature vectors that have s_t^n larger than or equal to the threshold are assigned to the right child node. Hence, the training data at the current node is re-partitioned into two sets corresponding to the left child node and the right child node, with the average entropy of the child nodes being smaller than the entropy of the current node. The reduction in entropy is then computed for the assumed value of h, and the value of h that gives the maximum reduction in entropy is then chosen to partition the data into the two child nodes.

In our implementation, we evaluated the reduction in entropy for about 100 values of h uniformly spaced between the minimum and maximum values of s_t^n.

(iv) Store the vector $\underline{v}_n$, and the optimal value of h, h_n, in association with the node n of the decision tree.

(v) the above four steps are then repeated for each child node until the data at a node falls below a specified threshold.

3.1 Characterization of terminal nodes

Once the tree has been fully grown, the next step is to determine how to use the tree. It is clear that the terminal nodes of the tree represent non-overlapping regions of the feature space that are each characterized by the fact that only a few of the allophones lie in each region. Hence, we could maintain a histogram of the allophones that occur in each region, and during decoding, evaluate only the gaussians modelling these allophones. This scheme will be referred to by the term 'allophone-lists' in the experiments to be described later.

The computation can be further reduced by keeping a list of individual gaussians that occur in a region, rather than a list of allophones. These lists are obtained by finding, for each $\underline{x}_t^n$, the most likely gaussian in the mixture model for l_t^n and including it in the list for the terminal node. The benefit in doing this clearly is that fewer gaussians need be evaluated per frame during decoding, and the cost may be an increase in error rate. This scheme will be referred to by the term 'gaussian-lists' in the following experiments.

4 USE OF THE DECISION TREE DURING DECODING

In the previous section, we described the motivation and process of constructing a decision tree to partition the feature space, and also discussed the characterization of the terminal nodes of the tree. We will next describe how this information may be used in the decoding process.

For every feature vector, $\underline{y}_t$, in the test set, the first step is to traverse the decision tree and find the terminal node, that the feature vector belongs to. The process of traversing the decision tree is as follows : starting from the root node of the tree, the projection of the feature vector on the hyperplane, $\underline{v}_n^T \underline{y}_t$ is computed; the projection is then compared to the threshold, h_n, and depending on whether the value is smaller or greater than the threshold, the left or right child node of the current node is selected, and the process repeated for the selected child node. The process terminates when the child node is a terminal node of the decision tree.

The terminal node of the decision tree is now characterized either by a list of the allophone classes, or the gaussians that occur at that terminal node, and when computing the allophone likelihoods, we evaluate only this subset of gaussians. The allophones that don't occur in the list are given a default low probablity. It is possible that for a test feature vector, the correct allophone may not be in the list of the terminal node corresponding to the feature vector. This would lead to the recognizer making a few additional errors, however, experimental results have shown that the relative degradation in accuracy is negligible.

5 EXPERIMENTAL RESULTS

In this section we describe the diagnostics associated with the process of construction of the tree, and also the results of decoding experiments where the decision tree was used to decrease the computation time.

5.1 Tree growing process

As mentioned earlier, the objective in growing the tree is to reduce the entropy of the allophone class distribution in the training data. The average entropy of the class distribution as a function of the depth of the tree is shown in Fig. 1 [2] (the total number of allophone classes was approximately 6000). It can be seen that in the initial stages of the tree growing process, the linear hyperplane splits provide close to the maximum possible gain of 1 bit, but this diminishes as the tree is grown deeper.

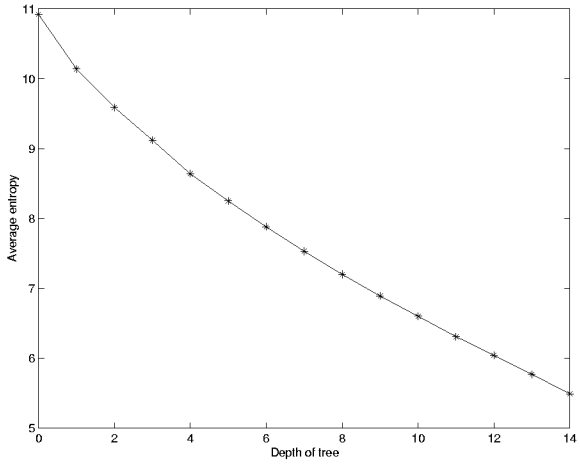

Figure 1: Reduction in entropy

5.2 Decoding results

We carried out a number of experiments on a large-vocabulary task (20k-vocab, the North American Business News (NAB) 1994 eval data). The baseline was a rank-based system (for details of rank-based systems, see [1]), that used $\approx$ 6000 allophones. We experimented with three different systems where each allophone was modelled with a maximum of one, six, and sixty gaussians; these systems respectively had approximately 6000, 34000, and 168000 gaussians in total. The same decision tree, including the allophone lists at the terminal nodes of the tree, was used in all the experiments; however, the gaussian lists characterizing the terminal nodes of the tree were computed separately for the three systems. We experimented with various sizes of the decision tree, in conjunction with each of the three systems described above, and the results of the experiments are summarized in Tables I-III. In the tables, '# terminal nodes' refers to the number of terminal nodes in the decision tree, '# gaussians evaluated' refers to the number of gaussians evaluated per frame, 'error rate' refers to the word error rate, 'allophone-list' implies that the terminal nodes of the decision tree are characterized by lists of allophones, 'gaussian-list' implies that the terminal nodes of the decision tree are characterized by lists of gaussians.

From the above tables, we may draw the following

[2] Note that the maximum reduction in entropy for a binary split is 1 bit.

Table I (One gaussian per leaf)			
# terminal nodes		2048	8192
# gaussians evaluated	5741	553	283
error rate	18.9	18.5	18.5

Table II (Six gaussians per leaf)			
allophone-list			
# terminal nodes		4096	16384
# gaussians evaluated	33844	3037	1576
error rate	14.1	14.4	14.5
gaussian-list			
# gaussians evaluated	33844	1588	680
error rate	14.1	14.0	15.6

conclusions :
(i) a reduction in computational complexity by a factor of 20 may be achieved with negligible degradation in accuracy
(ii) in general, for the same computation complexity, better performance is obtained with gaussian-list than with allophone-list
(iii) computational complexity can be traded off against error rate by changing the size of the tree.

6 DISCUSSION

We described a way to quantize the feature space of a speech recognizer using a binary decision tree that uses linear hyperplanes to divide the feature space into disjoint regions, with each region containing only a small number of the total alphabet of allophones or individual gaussians.
• Experimental results showed that computational complexity could be reduced by a factor of 20 with negligible degradation in accuracy.

• The construction of the decision tree is only dependent on the feature space and not on the actual gaussian model used in the speech recognizer. Only the gaussian lists at the terminal nodes of the decision tree are dependent on the actual gaussian model that is used, and these may be computed relatively inexpensively.

Table III (Sixty gaussians per leaf)			
allophone-list			
# terminal nodes		4096	16384
# gaussians evaluated	168195	17500	9500
error rate	12.1	12.5	13.2
gaussian-list			
# gaussians evaluated	168195	2803	1126
error rate	12.1	13.3	15.5

• When speaker adaptation [7] is used to modify the gaussian parameters, preliminary experiments (not reported here) indicate that the speaker-independent tree and gaussian lists can be used with no loss in accuracy, even though the gaussian parameters have now been tuned to the particular speaker.
• The tree is dependent on the feature-space, consequently, if the feature space were to change drastically (due to additive noise, alternate microphone, etc.), then the tree needs to be adapted to the new feature space. We are currently investigating techniques to adapt the tree, as well as alternatives to linear hyperplanes that are more robust to changes in the feature space.

REFERENCES

[1] L. R. Bahl et al., "Performance of the IBM large vocabulary continuous speech recognizer on the ARPA Wall Street Journal task", Proceedings of the ICASSP, pp 41-44, 1995.

[2] K. M. Knill, M. J. F. Gales, S. J. Young, "Use of gaussian selection in large vocabulary continuous speech recognition using HMMs", Proceedings ICSLP, pp 470-473, 1996.

[3] E. Bocchieri, "Vector quantization for efficient computation of continuous density likelihoods", Proceedings ICASSP, vol II, pp 692-695, 1993.

[4] H. Murveit et al., "Techniques to achieve an accurate real-time large vocabulary speech recognition system", Proceedings ARPA Workshop on Human Language Technology, pp 368-373, 1994.

[5] C. Waast, L. Bahl and M. El-Beze, "Fast-Match based on decision tree", Proceedings of EUROSPEECH, 1995.

[6] P. O. Duda and P. E Hart, "Pattern classification and scene analysis", Wiley, New York, 1973.

[7] M. Padmanabhan, L. R. Bahl, D. Nahamoo and M. A. Picheny, "Speaker Clustering and Transformation for Speaker Adaptation", Proceedings of the ICASSP, Atlanta, vol II, pp 701-704, May 1996, also to appear in IEEE Trans. Speech and Audio Processing.