DISCRETE MMI PROBABILITY MODELS FOR HMM SPEECH RECOGNITION

J. T. Foote
Cambridge University Engineering Department, Cambridge CB2 1PZ, UK

ABSTRACT

This paper presents a method of non-parametrically modeling HMM output probabilities. Discrete output probabilities are estimated from a tree-based MMI partition of the feature space, rather than the usual vector quantization. One advantage of a decision-tree method is that very high-dimensional spaces can be partitioned. Time variation can then be explicitly modeled by concatenating time-adjacent vectors, which is shown to improve recognition performance. Though the model is discrete, it provides recognition performance better than 1-component Gaussian mixture HMMs on the ARPA Resource Management (RM) task. This method is not without drawbacks: because of its non-parametric nature, a large number of parameters are needed for a good model and the available RM training data is probably not sufficient. Besides the computational advantages of a discrete model, this method has promising applications in talker identification, adaptation, and clustering.

1. INTRODUCTION

Most recent speech recognition work has centered on continuous-density hidden Markov models, which have shown a clear performance advantage over discrete-output HMMs. This paper presents an alternative: a discrete output probability model that is comparable with (but does not exceed) the performance of a standard Gaussian mixture HMM recognition system. Unlike K-means vector quantization (VQ), the tree-based quantization is supervised, which means the feature space may be profitably discretized into many more regions than the conventional minimum-distortion vector quantizers. In addition, the tree-based method is arguably more robust in high-dimensional feature space. This is exploited by concatenating adjacent feature vectors, resulting in a high-dimensional space which contains information about the time dependence of the features.

1.1. Tree Output Probability Models for Speech

The feature space is partitioned into a number of discrete regions (analogous to the Voronoi polygons surrounding VQ reference vectors) by a decision tree. Unlike K-means reference vector estimation, the tree is grown in a supervised fashion. Each decision in the tree involves comparing one element of the vector with a fixed threshold, and going to the left or right child depending on the outcome. Each threshold is chosen to maximize the mutual information $I(X;C)$ between the data X and the associated class labels C (obtained from Viterbi alignment) that indicate the acoustic class of each datum.

1.2. Tree Construction

Because the construction of optimal decision trees is NP-hard, they are typically grown using a greedy strategy [1]. The first step of the greedy algorithm is to find the decision hyperplane that maximizes the mutual information metric. While other researchers have searched for the best general hyperplane using a gradient-ascent search [2], the approach taken here is to consider only hyperplanes normal to the feature axes, and to find the maximum mutual information (MMI) hyperplane from the optimal one-dimensional split. This is computationally reasonable, easily optimized, and has the advantage that the search cost increases only linearly with dimension.

To build a tree, the best MMI split for all the training data is found by considering all possible thresholds in all possible dimensions. The MMI split threshold is a hyperplane parallel to all feature axes except dimension d, which it intercepts at value t. This hyperplane divides the set of N training vectors X into two sets $X = \{Xa, Xb\}$, such that

$$Xa: \quad x_d \geq t_d \tag{1}$$

$$Xb: \quad x_d < t_d \tag{2}$$

This first split corresponds to the root node in the classification tree. The left child then inherits Xb, the set of training samples less than the threshold, while the right child inherits the complement, Xa. The splitting process is repeated recursively on each child, which results in further thresholds and nodes in the tree. Each node in the tree corresponds to a hyper-rectangular region or "cell" in the feature space, which is in turn subdivided by descendants. Cells corresponding to the leaves of the tree completely partition the feature space into non-overlapping regions, as shown in Figure 1.1..

To calculate the mutual information $I(X;C)$ of a split, consider a threshold t in dimension d. The mutual information from the split is easily estimated from the training data in the following manner. Over the volume of the current cell, count the relative frequencies:

$$\begin{aligned} N_{ij} &= \text{Number of data points in cell } j \text{ from class } i \\ N_j &= \text{Total number of data points in cell } j \\ &= \sum_i N_{ij} \\ A_i &= \text{Number of data points from class } i : x_d \geq t_d \end{aligned}$$

In the region of cell j, define $Pr(c_i)$ to be the probability of class i and $Pr(a_i)$ as the probability that a member of

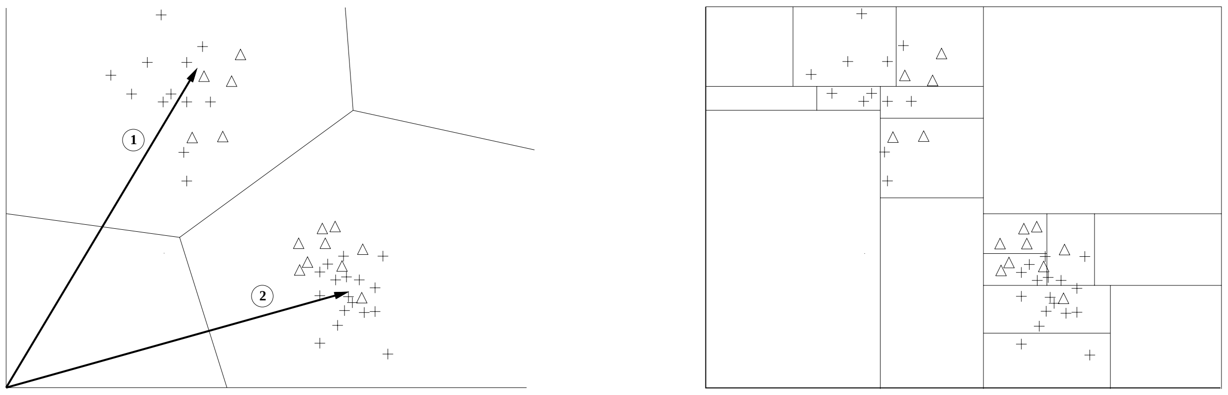

Figure 1. VQ (left) and MMI tree (right) feature space partitions.

class i is above the given threshold. These probabilities are easily estimated as follows:

$$\Pr(c_i) \approx \frac{N_{ij}}{N_j} \tag{3}$$

$$\Pr(a_i) \approx \frac{A_i}{N_{ij}} \tag{4}$$

With these probabilities, the mutual information given the threshold may be estimated in the following manner (for clarity of notation, conditioning on the threshold is not indicated):

$$I(X;C) = H(C) - H(C|X) \tag{5}$$

$$= -\sum_i \Pr(c_i)\log_2 \Pr(c_i) + \sum_i \Pr(c_i) H_2\left(\Pr(a_i)\right) \tag{6}$$

$$\approx -\sum_i \frac{N_{ij}}{N_j}\log_2 \frac{N_{ij}}{N_j} + \sum_i \frac{N_{ij}}{N_j} H_2\left(\frac{A_i}{N_{ij}}\right), \tag{7}$$

where H_2 is the binary entropy function

$$H_2(x) = -x\log_2(x) - (1-x)\log_2(1-x). \tag{8}$$

Equation 7 is a function of the (scalar) threshold t, and may be quickly optimized by a region-contraction search.

This splitting process is repeated recursively on each child, which results in further thresholds and nodes in the tree. At some point, a stopping rule decides that further splits are not worthwhile and the splitting process is stopped. The MMI criterion works well for finding good splits, but is a poor stopping condition because it is generally non-decreasing. (Imagine a tiny cell containing only two data points from different classes: any hyperplane between the points will yield an entire bit of mutual information. Bigger cells with overlapping distributions generally have less mutual information.) Also, if the number of training points in a cell is small, the probability estimates for that cell may be unreliable. This motivates a stopping metric where the best-split mutual information is weighted by the probability mass inside the cell l_j to be split:

$$\text{stop}(l_j) = \left(\frac{N_j}{N}\right) I_j(X;C) \tag{9}$$

where N is the total number of available training points. Further splits are not considered when this metric falls below some threshold. This mass-weighted MMI criterion thus insures that splitting is not continued if either the split criterion is small, or there is insufficient probability mass in the cell to reliably estimate the split threshold.

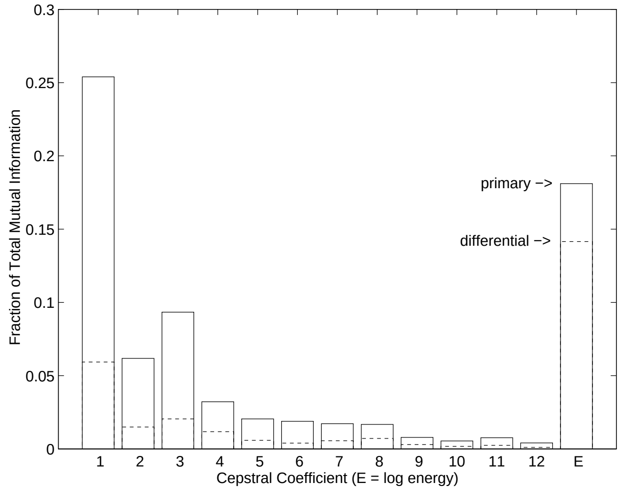

Figure 2. Fraction of mutual information by feature.

1.3. Dimensional Importance

An interesting side-effect of tree construction is that the relative importance of feature-space dimensions can be estimated. The maximum mutual information given by a split in a particular dimension will depend on how well the feature values are correlated with the class labels. The relative "importance" of each feature may then be judged by looking at its contribution to the total mutual information. Figure 2 shows the fraction of mass-weighted mutual information given by each dimension for a tree grown on the RM data of Section 3. The relative importance of 12 mel-cepstral coefficients, log energy, and differences thereof are plotted, with the differential values indicated by a dashed horizontal line. Figure 2 shows clearly that the energy, delta energy, and low-order cepstra are the most important features. These results are in good agreement with those of Bocchieri and Wilpon [3].

2. TREE-BASED QUANTIZATION

The tree partitions the feature space into L non-overlapping regions or "cells," each of which corresponds to a leaf of the tree. For speech recognition, the tree may be used as a vector quantizer front-end for a discrete HMM system. The HMM output probability model therefore consists of discrete probabilities $p_j(i)$, the probability that the output

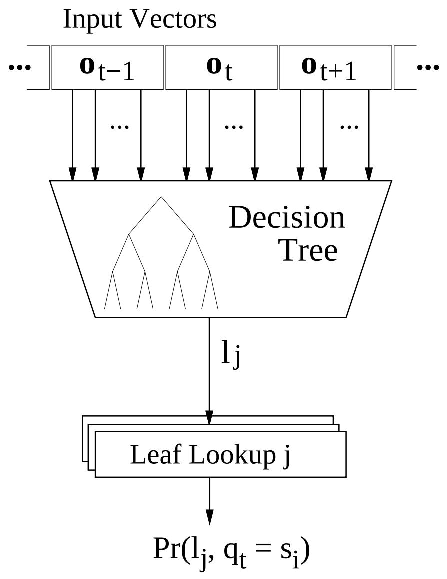

Figure 3. Contextual input to decision tree.

of HMM state s_i is a vector falling in leaf l_j.

2.1. Estimating Tree Probabilities

HMM output probabilities may be estimated from the Baum-Welch algorithm as well as from Viterbi training (used in the experiments presented later). Given Viterbi-aligned data and a decision tree, it is straightforward to count N_{ij}, the number of data points from state s_i (i.e. the number of observation vectors aligned with state s_i) that wind up in leaf l_j. From this, the conditional probability of an output falling in leaf node l_j given the state s_i may be estimated as

$$p_j(i) \approx \frac{N_{ij}}{\sum_j N_{ij}}. \tag{10}$$

Thus the tree output probabilities can be quickly estimated by Viterbi-aligning novel data and counting the relative frequencies at each leaf. In practice, limited training data leads to zero N_{ij} counts for many i and j. In this case, $p_j(i)$ values of zero are set to some small floor value and the ensemble is renormalized.

2.2. Tree Models for Speech

Tree-based vector quantizers have some interesting advantages over conventional vector quantizers. Perhaps most importantly, MMI-constructed trees can arguably handle the "curse of dimensionality" better than a minimum-distortion VQ, in part because only one dimension is considered at each split. Dimensions that do not help class discrimination are ignored, in contrast to a distortion metric which is always computed across all dimensions.

An immediate application is that inter-frame time variation can be explicitly modeled by concatenating several adjacent observation vectors, as shown schematically in Figure 3. Though this drastically increases the feature space dimensionality, it is found to significantly improve the performance of a tree-based HMM recognition system.

Another advantage is that trees are easily pruned to the desired size, allowing the number of free parameters (proportional to the number of leaves) to be tailored to the problem. Where data is sparse (as in talker ID), smaller trees are more robust to undertraining, though the resulting model is coarser. Conversely, where data is plentiful (as in the recent Wall Street Journal corpora), the tree model may be made extremely detailed, as the computational cost increases only logarithmically with the number of leaves.

Another way to simplify high-dimensional problems is to use overlapping trees. In a manner similar to using multiple "codebooks," multiple trees may be grown independently for lower-dimensional subsets of the feature space. Output probabilities are then computed as the product of the tree probabilities. These will be underestimated if the subspaces are not truly independent, but this is usually outweighed by the higher spatial resolution obtainable (it increases exponentially with the number of overlapping trees).

2.3. Model Parameter Requirements

A drawback of non-parametric models is that they typically require more training data and parameter storage than comparable parametric models. For comparison, let S be the number of model states, M be the number of mixture components in a Gaussian-mixture model, D be the dimensionality of the feature space, and L the number of leaves, and T the number of overlapping trees (equivalent to the number of streams or codebooks.)

Using the values from the HTK models of section 3., D is 39 and M varies from 1 to 15. Assuming diagonal covariance matrices, a continuous density HMM output probability model requires roughly $2MDS$ parameters. Ignoring the MS mixture weights, this is from 78 to 1170 parameters per state. Comparatively, tree models need LTS parameters (ignoring the size of the trees themselves which is $\mathcal{O}(L)$). For the trees used here, L ranges from 1024 to 4096 and T is 2, so each state requires from 2048 to 8192 parameters. Note that this is *independent* of D, the feature space dimension. Thus trees require roughly an order of magnitude more parameters than comparable Gaussian models.

3. RECOGNITION EXPERIMENTS

The HTK recognition system was used as a baseline for the tree experiments presented here. All models were trained with the RM SI-109 training set (3990 sentences) and evaluated on the speaker independent October '89 test set, using the the standard RM word-pair grammar. For all experiments, 48 3-state monophone models were used, augmented with an optional 1-state inter-word silence model and function-word specific phone models for the 32 most common words in the RM training data, as in [4].

Tree models were "bootstrapped" by growing and training them on data aligned with the 10-mixture monophone and function word models. Tree models were grown on a subset of the SI-109 training set consisting of two sentences from each speaker. Once grown, the output probability models were trained on the entire SI-109 set. This is essentially one iteration of Viterbi training. Trees were constructed using context windows of 1, 3, 5, and 7 adjacent vectors. The feature space was 12 mel-frequency cepstral coefficients and log energy, and the first differential of each, for a 26-dimensional feature space. This was (somewhat arbitrarily) divided into two 13-dimensional subspaces containing the primary and differential parameters respectively; two separate trees were grown for each subspace.

4. RESULTS AND ANALYSIS

Figure 4 shows the effect of adding vectors to the context window, for a tree size of 1024 leaves. It is clear that the additional information provided by the context helps recogni-

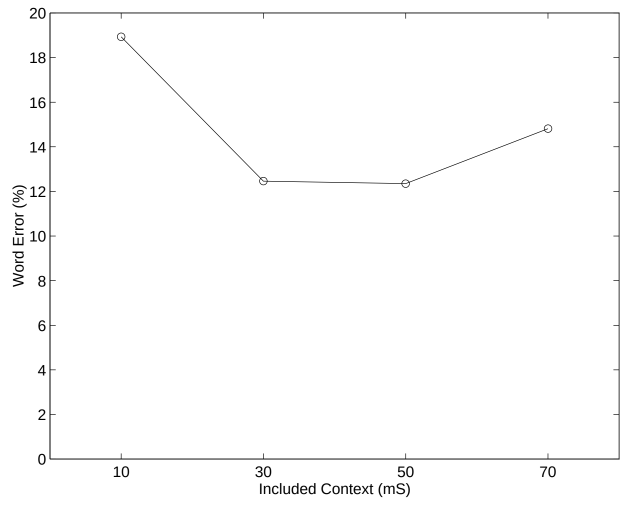

Figure 4. Word error vs. context (1024-leaf tree).

tion performance, even while using the first-differential parameters. Because of the many different contexts that are tied in a monophone system, the explicit context modeling is not as effective as it could be in system using context-dependent triphone or word-level models. Experiments on a small-vocabulary word-model system [5] show optimal performance with longer context windows of 70 to 90 milliseconds.

Figure 5 shows the word error rate versus tree size for a context window of 50 ms (5 adjacent vectors). It is clear that the larger tree models are substantially undertrained, because the vast majority of the N_{ij} counts are zero. More speculatively, the fact that larger tree sizes (more detailed models) do not result in better performance could be due to lack of sufficient training data. The need for large amounts of training data is a serious drawback of non-parametric models in general and the tree models in particular. The problem is not insurmountable; several methods are being investigated to extract better model estimates from the training data:

- Only one iteration of Viterbi training was used for the tree models. More iterations would improve the overall model likelihood,
- Viterbi training was used primarily for simplicity. Baum-Welch training might make better use of the limited training data, because it results in probabilistic rather than the zero-one state assignments from the Viterbi training.
- No states were tied. The divergence (relative entropy) between the discrete HMM output probabilities can be computed and used as a distance measure; similar states can then be tied, reducing the number of parameters to be estimated.

5. CONCLUSIONS

The smallest word-error rate of 12.4% was achieved with 1024-leaf trees using 30 ms context windows. For comparison, the HTK CD monophone system (using identical model topology and data) yielded a 17.3% error rate for 1-mixture monophone models and 11.3% obtained with 2-mixture monophone models. Obviously, this tree-based system is not competitive with the ultimate 15-mixture monophone models, which resulted in a 5.7% error rate. This is almost certainly due to the lack of sufficient training data.

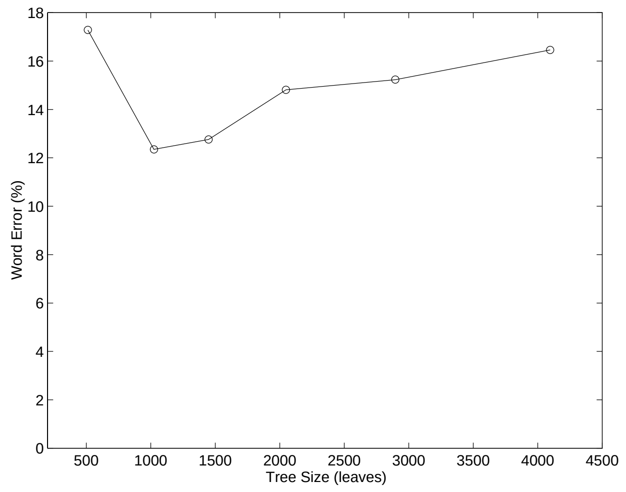

Figure 5. Word error vs. tree size (50-ms context).

The tree models presented here are not yet competitive with the best CD models, yet they have advantages that may outweigh this. Tree models may be trained quite rapidly on novel data by Viterbi-alignment and relative frequency estimation. This has been shown to be useful for talker adaptation and identification [7]. Also, because the tree models may be made extremely detailed at little additional cost, they may find applications in large-corpus speech recognition experiments such as Wall Street Journal, where there is sufficient data to train them properly.

6. ACKNOWLEDGEMENT

The author thanks Phil Woodland and Steve Young for the continuous-density RM models and use of the HTK recognition software.

REFERENCES

[1] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wadsworth International Group, Belmont, Calif., 1984.

[2] M. Anikst et al. The SSI large-vocabulary speaker-independent continuous-speech recognition system. In *Proc. 1991 ICASSP*, pages 337–340, 1991.

[3] E. L. Bocchieri and J. G. Wilpon. Discriminative analysis for feature reduction in automatic speech recognition. In *Proc. 1992 ICASSP*, volume I, 1992.

[4] P. C Woodland and S. J Young. The HTK tied-state continuous speech recogniser. In *Proc. Eurospeech*, volume 2, pages 2207–2210, 1993.

[5] Jonathan T. Foote. *Decision-Tree Probability Modeling for HMM Speech Recognition.* Ph.D. thesis, Brown University, Providence, RI, 1993.

[6] M. Anikst, W. Meisel, M. Soares, and K. Lee. Experiments with tree-structured MMI encoders on the RM task. In *Proc. Third DARPA Speech and NL Workshop*, June 1990.

[7] J. T. Foote and H. F. Silverman. A model distance measure for talker clustering and identification. In *Proc. 1994 ICASSP*, volume S1, pages 317–320, April 1994.