ABSTRACT

Phonetic decision-tree based acoustic modeling has been widely used in speech recognition systems. However, the assumption that all states clustered in the same leaf node share both their Gaussians and mixture weights restricts the improvement of the acoustic models. In this paper, we propose a new structure called a two-level decision-tree. With this structure we can make better use of training data and improve the model accuracy and robustness. Two-level decision trees provide more flexibility to control the number of parameters. By tuning the balance of the first and second level tree nodes, we can get better performance with even fewer parameters than the traditional decision-tree based approach. Experiments on the Wall Street Journal tasks show that our approach can achieve about a 10% word error rate reduction over the conventional approach.

1. INTRODUCTION

In large vocabulary continuous speech recognition systems, context-dependent phones, typically triphones, and continuous density HMM models are often used to get high accuracy acoustic models. The huge number of triphones and multivariate Gaussian mixture distributions results in too many parameters in a system. It is a key problem to maintain a good balance between the model complexity and the number of parameters that can be robustly estimated from the limited training data. The use of phonetic decision trees provides a good solution to this problem. It has two advantages over the bottom-up based approaches. First, by incorporating the phonetic knowledge of the target language into the tree, it can synthesize unseen models or contexts, which don't appear in the training data but occur during recognition. Second, the splitting procedure of decision trees provides a way of maintaining the model complexity and the number of parameters to be robustly estimated.

A phonetic decision tree is a type of classification and regression tree (CART). The book by Beriman et al. [1] provides the theoretical framework for developing decision trees. In decision-tree based acoustic modeling, phonetic decision trees are constructed either for each phone model or for each HMM state of each phone. Since the state-based approach provides a more detailed level of sharing and outperforms the model-based approach [2], the state-based approach is widely used. The phonetic decision tree is a binary tree in which a yes-no question about the phonetic context is attached to each node. An example question is "Is the phone on the right of the current phone a vowel?" A set of states can be recursively partitioned into subsets according to the answers to the questions at each node when traversing the tree from the root node to its leaf nodes. All states that reach the same leaf nodes are considered similar and are clustered together. The question set can be either manually pre-defined using linguistic and phonetic knowledge of the language, or automatically generated [3].

The tree construction is a top-down data driven process based on a one-step greedy tree growing algorithm [2]. The goodness-of-split criterion is based on maximum likelihood (ML) of the training data. Initially all corresponding HMM states of all triphones that share the same basic phone are pooled in the root node and the log-likelihood of the training data is calculated based on the assumption that all the states in the node are tied. This node is then split into two by the question that gives the maximum increase in log-likelihood of the training data when partitioning the states in the node. This process is repeated until the increase falls below a threshold. To ensure that each leaf node has sufficient training data to robustly estimate the state, a minimum data count for the leaf node is also applied.

Although the traditional method provides an effective and efficient way to build a decision tree for continuous density HMM models based on the maximum likelihood criterion, it has several problems. One is due to the assumption that the parametric form of the initial unclustered states should be based on only single mixture Gaussian distributions. After the tree is built, the clustered states have more training data and the number of Gaussian components in each state is increased by a mixture-splitting procedure until the performance of the model set peaks on a development set. The use of single Gaussian distributions during tree contraction is due to
the fact that the multiple mixture Gaussian distribution for a tree node needs to be re-estimated from the training data, whereas the parameters of the single mixture Gaussian distribution can be calculated efficiently from the cluster members without re-accessing the original training data. However, the single Gaussian distribution is a very crude representation of the acoustic space of each state and decision trees based on such initial models may not give good clustering of states. There are many efforts to address this problem. Chou et al. [4] incorporate a so-called m-level optimal subtree into the traditional tree construction to get a multiple mixture Gaussian distribution parameterization of each node although each member state still has only single Gaussian distribution as in the traditional approach. Kim et al. [5] directly estimate, by making some assumptions, the multiple mixture Gaussian distribution for a tree node from the statistics of the member states which also have multiple mixture Gaussian distributions. Both of their approaches achieve some improvement. Nock et al. [6] estimate the multiple mixture Gaussian distributions of the un-clustered states by using the fixed state alignment provided by a previously trained and accurate model set. However, this approach hasn’t been shown to give any improvement in terms of performance. Another problem with the standard tree-building process is due to the fact that construction of an optimal tree is an NP-hard problem. Instead, a sub-optimal one-step greedy algorithm is utilized. To make better decisions at each node split, look-ahead search is studied in [7]. However, no improvement is obtained. Many efforts address other aspects of the traditional decision-tree based state-clustering approach, such as applying other goodness-of-split criteria [6] [9], using cross-validation to automatically determine the size of the trees by pruning back instead of using thresholds which have to be determined by many experiments [6], and expanding the question set to incorporate more knowledge of the language [7] [10].

In this paper, we propose a new structure called a two-level decision tree to improve the decision-tree based acoustic modeling. It is based on the traditional decision-tree approach and is built with the same goodness-of-split criterion. By decoupling the Gaussian pool and the corresponding mixture weights using the two-level decision tree structure, we can make better use of training data and thereby improve the accuracy and robustness of the clustered acoustic models. Experimental results on large vocabulary continuous speech recognition tasks show the effectiveness of this new approach.

The structure of this paper is as follows. The next section explains a problem in the traditional decision-tree based acoustic modeling and the idea of two-level decision trees is proposed to address the problem. Section 3 gives the experimental results followed by a brief conclusion in Section 4.

2. TWO-LEVEL DECISION TREES

During the decision tree construction, if all the data associated with all the states in a node is less than a threshold, the node is no longer split and becomes a leaf node. This is to ensure that all the states in the node can be robustly estimated, since too little data won't be statistically representative enough and models estimated on it will over-fit the data and generalize poorly, causing performance loss. In practice, the threshold is determined by experiments. However, this data-count thresholding will cause problems. Some states, particularly those states of rarely-seen triphones which have very few training data, are clustered together not because they are acoustically very similar, but because they don't have enough training data. This can be a cause of performance degradation. In the traditional decision-tree based approach, all the Gaussian components and the corresponding weights are shared across all the member states in the same leaf node, and no distinctions are provided among those states clustered together. This assumption restricts the improvement of the model accuracy.

Based on the traditional decision tree, we propose a new structure called a two-level decision tree to address this problem. Figure 1 illustrates the two-level decision tree structure. Essentially, it is still a regular phonetic decision tree. The difference is that it has two levels (here level means the type of leaf nodes). In the first-level (called Gaussian-sharing level) tree, all the states in the same leaf node share the same Gaussian pool. The nodes of the second-level (called weight-sharing level) tree, are obtained by further splitting the first-level leaf nodes using the same method. All the second-level leaf nodes that come from the same first-level leaf node will share the same Gaussian pool, but they have their own set of weights for the Gaussians. All the states in the same second-level leaf node will share their mixture weights. If we don't expand the first-level leaf nodes, the tree is the same as a traditional decision tree (for contrast, we refer to it as a one-level tree), in which all states in the same leaf node share not only Gaussians but also corresponding mixture weights. By using the second-level nodes, better resolution can be achieved among those states in the same first-level leaf node by distinguishing their mixture weights, especially for those rarely-seen triphone states that have to be clustered together due to the minimum count thresholding. By controlling the number of the second-level nodes, the increased number of mixture
weights adds only a very small percentage of parameters. For example, if each first-level leaf node is split into only two second-level nodes, the two-level tree only adds about 1/79 of the number of parameters (assuming the speech feature vector has 39 elements and the covariance matrix of each Gaussian is diagonal). The increased number of mixture weights still can be robustly estimated since estimating mixture weights requires very few training data given a robustly estimated Gaussian pool in the first-level leaf node. (Actually they are estimated jointly to get optimal values.) By decoupling Gaussians and corresponding weights in the traditional decision tree and creating a two-level tree structure, we can make better use of training data, especially for those rarely seen triphones. From this point of view, the proposed structure is similar to the tied-mixture or semi-continuous HMM and thus has the similar advantage over the one-level decision-tree based acoustic modeling. But tied-mixture HMMs only have one global set of Gaussians that are shared by all the states in all models. Here two-level decision-tree structure gives a much finer level of sharing. So far, there has been an implicit assumption that two-level trees are built by expanding the optimal one-level trees and thereby the two-level tree based acoustic models have more parameters than the one-level tree based models. However, this is not necessary. In practice, we found that by tuning the balance of the first-level and second-level leaf nodes, we can get better performance with even fewer parameters than an optimal one-level tree system.

Normally a global threshold of data count is used for all the trees and the value is tuned until the final state-clustered model set based on the decision tree achieves best performance on a development set. Even with this global optimal value, it is still possible that some leaf nodes are under-trained and should be further split to get higher resolution and some leaf nodes are over-trained and should be pruned back (although this problem is lessened by allowing a variable number of Gaussian components in the final model set). This problem can be lessened in the two-level decision tree structure since it provides more flexibility to control the total number of parameters. By using relatively conservative value of data count threshold in the first-level (Gaussian-sharing level) tree, we can ensure that the Gaussian pools in all of the leaf nodes can be robustly estimated. And by using the second-level (weight-sharing level) nodes, we can get high resolutions. Although a global data count threshold (smaller than the one used in the first-level tree) is still used, the impact of this problem is smaller since it only affects the mixture weights.

3. EXPERIMENTAL RESULTS

The experiments were based on the 1998 OGI-FONIX Broadcast News Transcription system [8]. But we chose the ARPA Wall Street Journal (WSJ) 5K and 20K tasks as the test environments because they are difficult enough to represent some applications, and have an acceptable computational cost to speed up the experiment turn-around time. The system is a speaker-independent, decision-tree based state-tying system using continuous HMMs. The acoustic feature vector has 39 elements, consisting of 12 mel-frequency cepstral coefficients (MFCC) and normalized energy plus their first and second order time derivatives. The cepstral mean for each sentence was calculated and removed. All phone models have three emitting states and left-to-right topology.

For the WSJ 5K task, the acoustic training data is the standard set S184, consisting of 7193 sentences from 84 different speakers (42 male and 42 females). Two test sets are used. The si_dt_05 development set consists of 442 sentences from 10 different speakers and the 5K ARPA evaluation set in 1992, referred to as NoV92, consists of 330 sentences from 8 speakers. For the WSJ 20K task, the training data is the standard set S1284, consisting of about 36,000 sentences from 284 different speakers (142 male and 142 female). The testing set is the 20K ARPA evaluation set in November 1992. It consists of 332 sentences from 8 different speakers.

For the WSJ 5K task, a one-pass decoder was used to perform a beam search through a tree-structured network. For the WSJ 20K task, to speed up the experiments, a two-pass decoder was used. First pass decoding used within-word triphone models together with a bigram language model, and generated lattices. Second pass decoding was performed on the lattice using cross-word triphones and a trigram language model. Since the search error in the first pass is irrecoverable in the second pass, a very wide search beam was used to ensure that the correct sentence would be in the lattices. The comparative experiments were conducted in the second pass. No adaptation was applied.

Since the performance of the acoustic models based on both the one-level and two-level decision trees were sensitive to the particular combination of thresholds used in the tree construction and training procedure, it caused some difficulty in evaluating the new approach. Thus,
although both systems were optimized over a limited range of thresholds, the figures might still be susceptible to some degree of variation. Both the minimum data count and the threshold of log-likelihood increase for the second level are smaller than those used in the first level in order to grow the second-level nodes.

<table>
<thead>
<tr>
<th>Test set</th>
<th>One-level</th>
<th>Two-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nov92</td>
<td>9.90%</td>
<td>8.80%</td>
</tr>
<tr>
<td>si_dt_05</td>
<td>11.20%</td>
<td>10.30%</td>
</tr>
</tbody>
</table>

Table 1. Word error rate on the WSJ 5K task

Table 1 gives the best performances for both the standard one-level tree system and the two-level tree system on two WSJ 5K test sets. The one-level tree system has 3871 states (or leaf nodes) and 8 Gaussian components per state. The minimum data count in tree construction is 80. The two-level tree system has a total of 3295 first-level leaf nodes and 12729 second-level leaf nodes. Each first-level node has 8 Gaussian components. The minimum data counts used in the construction of the first level and second level are 100 and 40, respectively. There are a total of 2.5 million parameters in the one-level tree system and 2.2 million parameters in the two-level tree system. With a 12% reduction in the number of parameters, 11% and 8% word error rate reductions are obtained for the two test sets.

<table>
<thead>
<tr>
<th>Test set</th>
<th>One-level</th>
<th>Two-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nov92</td>
<td>12.20%</td>
<td>10.80%</td>
</tr>
</tbody>
</table>

Table 2. Word Error Rate on the WSJ 20K task

Table 2 gives the results for the WSJ 20K task. The one-level tree system has 7671 states and each state has from 4 to 12 Gaussian components according to available training data. The minimum data count is 80. The two-level tree system has a total of 6957 first-level leaf nodes and 20187 second-level leaf nodes. Each first-level leaf node has up to 12 Gaussian components. The minimum data counts used in the construction of the first level and second level are 100 and 40, respectively. There are a total of 7.3 and 6.8 million parameters in the one-level tree system and two-level tree system, respectively. With a 7% reduction in the number of parameters, we obtained 11% word error rate reduction.

The experimental results confirmed that using the proposed two-level decision tree is advantageous over the traditional decision-tree based acoustic modeling.

### 4. CONCLUSIONS

In this paper, a two-level decision-tree based approach is proposed to achieve more accurate and robust acoustic models than the traditional decision-tree based approach. By decoupling the Gaussian components and their corresponding weights and using the two-level decision tree structure, the limited training data can be more effectively used. Experimental results on the Wall Street Journal tasks show that the two-level decision-tree based acoustic modeling is superior to the traditional decision-tree based acoustic modeling.

### ACKNOWLEDGEMENTS

This work was started under the financial support of Fonix Corporation. The first two authors are currently funded by Intel Corporation.

### REFERENCES