PREDICTING UNSEEN TRIPHONES WITH SENONES

Mei-Yuh Hwang Xuedong Huang Fileno Alleva

April 5, 1993
CMU-CS-93-139

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

In large-vocabulary speech recognition, the decoder often encounters triphones that are not covered in the training data. These unseen triphones are usually represented by corresponding diphones or context independent monophones. We propose to use decision-tree based senones to generate needed senonic baseforms for unseen triphones. A decision tree is built for each individual Markov state of each phone, and the leaves of the trees constitute the senone codebook. To find the senone a Markov state of any triphone is associated with, we traverse the corresponding tree until we reach a leaf node, where a senone is represented. We used the DARPA 5,000-word speaker-independent Wall Street Journal dictation task to evaluate the proposed method. The word error rate was reduced by 11% when unseen triphones were modeled by the decision-tree based senones. When there were at least 5 unseen triphones in each test utterance, the error rate could be reduced by more than 20%.

This research was sponsored by the Defense Advanced Research Projects Agency, DOD and monitored by the Space and Naval Warfare System Command under contract N00039-91-C-0158, ARPA Order 7239.

Keywords: Shared-distribution models, senones, generalized triphones, decision trees, entropy, cross entropy

Contents

1. INTRODUCTION

The shared-distribution model (SDM) [9] in the SPHINX-II system [7] has been shown to provide superior speech recognition accuracy over the generalized triphone approach [19]. It clusters Markov states instead of entire phonetic hidden Markov models (HMMs), leading to clustered output distributions called senones [10]. The senone provides not only an improved recognition accuracy but also a pronunciation-optimization capability.

Because it is a bottom-up clustering approach, the SDM is unable to handle unseen triphones. In large-vocabulary speech recognition, new triphones are often encountered during testing due to the limited amount of training data. Diphone or context-independent monophone models are usually adopted to represent these unseen triphones in real applications. Both context-independent and diphone models are not as detailed as triphones because they lose information at the left and/or right phoneme context. For unseen triphone modeling, decision trees have been used to model allophones as a top-down generalization approach [3, 2, 5]. This technique made it possible to replace an unseen triphone with an existing leaf model. To incorporate this prediction capability and maintain the senone performance, we propose using a senonic decision-tree for triphone modeling, where Markov output distributions, not entire HMMs, are classified by the decision tree with binary questions.

In this paper, the SDM is first improved to be applied to very large vocabulary tasks. Phone dependency, which prohibits clustering of output distributions of triphone HMMs that represent different phonemes, is still followed. Meanwhile, state dependency is introduced to speed up the clustering procedure, where output distributions are allowed to be merged only if they are associated with the same k-th Markov state in the model topology. Finally, to more reliably estimate the similarity of output distributions, global information about the entire triphone HMM is incorporated while clustering each individual Markov state.

This distribution sharing principle is then extended to the top-down decision tree in order to model unseen triphones by senones, for unseen triphones occur often in very large vocabulary tasks. A decision tree is built for each individual Markov state of each phoneme, as constrained by both phone dependency and state dependency. Leaves of the decision trees constitute the senone codebook for the recognition system. To determine which senone it is to be associated with, a Markov state of any triphone model, either seen or unseen in the training data, traverses the corresponding decision tree until a leaf node is reached. In the same manner as the SDM improves the generalized triphone by clustering at a finer level, the senonic decision-tree inherits the merits of the SDM; but more importantly, it also has the ability to model unseen triphones like the allophonic model [5].

In the DARPA 5,000-word speaker-independent Wall Street Journal (WSJ) dictation experiments, we found that when the unseen triphones were not modeled, the top-down approach performed, as expected, slightly worse than the SDM since the SDM has much more freedom in optimizing the clustering of seen triphones. However, when the unseen triphones were predicted by the senonic decision-tree, the top-down approach was able to outperform the SDM which had no elegant method of modeling unseen triphones. We noted that an 11% error rate reduction was achieved when the senonic decision-tree predicted unseen triphones, which underlines the importance of accurately predicting unseen triphones. When there were at least 5 unseen triphones for each utterance in the test set, the error rate was reduced by more than 20%. Even when the test set contained few or no unseen triphones, modeling unseen triphones still helped the decoder to prune the wrong paths in which unseen triphones existed.

The paper is organized as follows. Section 2 introduces the improved SDM. Section 3 extends the shared-distribution principle to the senonic decision-tree. Finally in Section 4 we present performance results for the SDM and the senonic decision-tree.

2. THE IMPROVED SHARED-DISTRIBUTION MODEL

2.1. Phone Dependency and State Dependency

The distribution clustering procedure for the SDM disallows clustering of Markov states across different phonemes. For example, the output distributions of any $/ae/$-triphone HMM are never clustered with those of any $/ts/$-triphone. We call this constraint *phone dependency*. Within the same phoneme, we virtually permit all possible configurations by allowing elements to move from one cluster to another. Consequently, the computational complexity is an exponential function of the number of objects being clustered, which is the number of triphones times the number of output distributions per triphone model. When applying the SDM to a large vocabulary task like WSJ, the cost of the clustering procedure becomes intolerable. To reduce complexity and incorporate state-location information, *state dependency* is enforced, where two output distributions are allowed to be merged only if they come from the same k-th Markov state in the model topology. This decision was informed by the SDM studies on the DARPA Resource Management task [9], where we found output distributions which were in the same cluster were mostly from the same Markov state.

The bottom-up clustering for the SDM is summarized in Figure 1. More details can be found in [9].

For each phoneme p,

- Estimate all p-triphone HMMs.
- For each Markov state k in the model topology, cluster all the k-th output distributions in all p-triphones.
  1. Create a singleton cluster for the k-th output distribution of every p-triphone.
  2. Merge the most similar pair of clusters.
  3. Move one element from one cluster to another if it results in an improvement.
  4. Go to step 2 unless some convergence criterion is met.

Figure 1: The algorithm for output-distribution clustering, with phone-dependency and state-dependency.

For discrete output probabilities, the similarity measure between two clusters at step 2 of Figure 1 is explained in detail in the following section. A slight modification to replace summation with integration has to be made when applying it to continuous densities [13, 16].

2.2. Usage of Global Information

In addition to state dependency, global information about the entire Markov model is incorporated to measure the distortion when two output distributions are merged. To elaborate, let's start with the definition of entropy and cross entropy.

2.2.1. Entropy and Cross Entropy

Given a discrete probability distribution (pd) $\mathcal{P}$ with L entries, the entropy is defined as [1]:

$$H(\mathcal{P}) \triangleq -\sum_{i=1}^{L} p_i \log_2 p_i$$

where p_i is an individual probability entry in pd $\mathcal{P}$. Entropy measures the expected uncertainty of the source that generates the possible events. The more uncertain, the bigger the entropy. When $\mathcal{P}$ is uniform, $H(\mathcal{P})$ reaches its maximum, $\log_2 L$.

Cross entropy is known as a good measure in determining the distance between two pds [11, 17, 15, 14]. The cross entropy (directed divergence) [17] between two pds, $\mathcal{P}$ and $\mathcal{Q}$, is defined as

$$\begin{aligned} \chi(\mathcal{P}, \mathcal{Q}) &\triangleq \sum_i p_i \log \frac{p_i}{q_i} \\ &= \sum_i p_i \log p_i - \sum_i p_i \log q_i \end{aligned}$$

which is not symmetric (and thus *directed*). To understand the meaning of cross entropy, consider the probability of generating event i for p_i times, given pd $\mathcal{Q}$:

$$Pr(\text{data } p_i \mid \text{pd } \mathcal{Q}) = {q_i}^{p_i}$$

assuming each occurrence is independent. In other words,

$$\begin{aligned} Pr(\text{data } \mathcal{P} \mid \text{pd } \mathcal{Q}) &= \prod_i {q_i}^{p_i} \\ \log Pr(\text{data } \mathcal{P} \mid \text{pd } \mathcal{Q}) &= \sum_i p_i \log q_i \leq \sum_i p_i \log p_i \end{aligned}$$

The equality holds when pd $\mathcal{Q}$ mimics data $\mathcal{P}$'s distribution, i.e. when $\mathcal{P} = \mathcal{Q}$ [1]. The cross entropy with respect to data $\mathcal{P}$ thus measures how well pd $\mathcal{Q}$ resembles pd $\mathcal{P}$.

2.2.2. Distortion Measure at a Fixed Markov State

When two pds, $\mathcal{P}$ and $\mathcal{Q}$, are merged, the corresponding count entries are summed to get the occurrence counts for the resulting pd. The distortion incurred from merging two pds is defined as the difference in entropy, weighted by the occurrence counts:

$$
\begin{aligned}
\triangle\tilde{H} &\stackrel{\triangle}{=} (P+Q)H(\mathcal{P}+\mathcal{Q}) - P\;H(\mathcal{P}) - QH(\mathcal{Q}) \\
&= P\sum_i p_i \log \frac{p_i}{(p_i P + q_i Q)/(P+Q)} \;+ Q\sum_i q_i \log \frac{q_i}{(p_i P + q_i Q)/(P+Q)} \\
&= P\,\chi(\mathcal{P},\mathcal{P}+\mathcal{Q}) \;+\; Q\,\chi(\mathcal{Q},\mathcal{P}+\mathcal{Q})
\end{aligned} \tag{1}
$$

where P is the summation of the $\mathcal{P}$ entries *before* normalization, that is, the summation of the occurrence counts. Similarly, Q is the summation of the occurrence counts in $\mathcal{Q}$. Thus the distortion is derived to be the cross entropies between the original pds and the resulting one, weighted by the original occurrence counts. The smaller the distortion is, the more similar the two pds are.

Weighting cross entropy by the occurrence count takes into account how well each pd is trained. Pds that appear infrequently (small counts) in the training data result in small distortion and thus will be merged first in comparison with the pds which occur frequently in the training corpus. This makes the infrequent pds become more trainable after clustering. Another advantage of weighting cross entropy by the occurrence count is its ability to prevent a big cluster getting blindly bigger. To see this, suppose $\mathcal{P}$ is a big cluster with many elements, compared with $\mathcal{Q}$, as shown in Figure 2. Because of the significant gap between the occurrence counts of $\mathcal{P}$ and $\mathcal{Q}$, the count entries in $\mathcal{P}+\mathcal{Q}$ are almost the same as those in $\mathcal{P}$, resulting in $\chi(\mathcal{P},\mathcal{P}+\mathcal{Q}) \approx 0$. Had the cross entropy *not* been weighted by the occurrence count, $\mathcal{P}$ and $\mathcal{Q}$ could have been merged before anther small cluster $\mathcal{R}$ got a chance to be collapsed with $\mathcal{Q}$, since $\chi(\mathcal{R},\mathcal{R}+\mathcal{Q})+\chi(\mathcal{Q},\mathcal{R}+\mathcal{Q})$ might be greater than $\chi(\mathcal{Q},\mathcal{P}+\mathcal{Q})$, even when $\mathcal{R}$ and $\mathcal{Q}$ are quite similar.

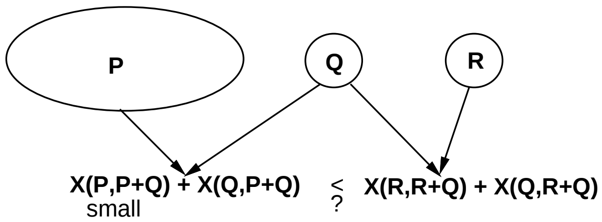

Figure 2: Hypothesizing the similarity measure without weighting cross entropy by the occurrent count. Both $\mathcal{Q}$ and $\mathcal{R}$ are small clusters, while $\mathcal{P}$ is a big one.

2.2.3. Global Information across Different Markov States

Formula (1) defines the distance between two pds which are from a particular Markov state k of any two triphones that represent the same phoneme. To more reliably estimate the distortion, we attach to each cluster not only the k-th output distribution, but also those output distributions associated with the rest Markov states. When two clusters are merged, all pairs of pd counts are added to compute the resulting cluster. The net distortion which arises from merging the k-th pds of two clusters is defined to be a weighted summation of the individual distortions:

$$D(k) = \sum_i w_k(i)\triangle\tilde{H}(i) \tag{2}$$

where $\triangle\tilde{H}(i)$ is defined in (1) with the i-th pd considered, and $\sum_i w_k(i) = 1.0$ in order to normalize the net distortions across different Markov states. $w_k(i)$ is inversely proportional to the topological distance between state k and state i. This emphasizes the information near the interesting spot k. In other words, the form of the clustering for each Markov state is exactly the same in the sense that all the pds for the entire phonetic HMM have to be computed. The only difference is the combining weights in (2). When state 1 is considered, $\triangle\tilde{H}(1)$ gets the highest weight; when state 2 is considered, $\triangle\tilde{H}(2)$ has the maximal weight. The generalized triphone is thus the special case of the SDM in which uniform weights are adopted for all Markov states.

Formula (2) was the distortion measure for all the experiments conducted in this paper.

3. THE SENONIC DECISION-TREE

To predict unseen triphones with senones, we extend the principle of distribution sharing to decision trees [5]. In particular, we build a decision tree for each Markov state of each phoneme, as constrained by both phone dependency and state dependency. Our system has 50 phonemes for English. When there are 5 states for each phonetic HMM, that means 250 trees in total. A decision tree is a binary tree, with a (possibly composite) linguistic question associated with each node. Figure 3 shows an example tree for the first state of $/k/$-triphones.

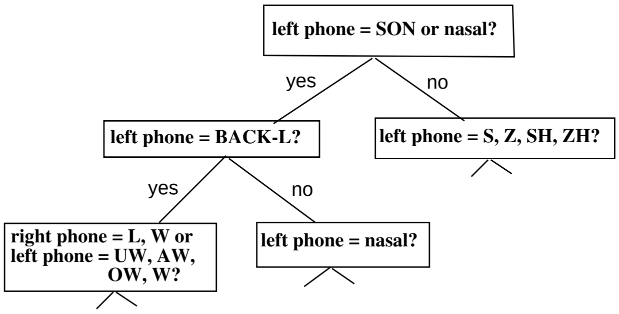

Figure 3: The partial decision tree for Markov state 1 of $/k/$-triphones.

Like the SDM, each node (cluster) is represented by a pd whose count entries are the summations of the count entries of all the component output distributions. To bear the global information, pds for the states other than the one being classified are also computed for evaluating the net distortion in (2). The leaves of these decision trees are the senones.

To find the senone associated with any Markov state of any triphone, we traverse the corresponding tree by answering the questions until a leaf is reached, where a senone is represented. Unseen triphones, which are triphones needed in the decoder but which never occur in the training data, traverse the tree exactly the same way as seen triphones. Thus we are able to elegantly model unseen triphones by senones.

3.1. Construction of the Decision Tree

Figure 4 summarizes the construction of the decision tree. The linguistic questions we located were simple questions like "is the right context of the triphone a back vowel?" or "is the triphone the first phone of a word?".

Interested readers can find more details in [4]. However, two basic differences should be noted. First of all, the classification is on subphonetic units (i.e., the output distributions) rather than on entire HMMs. Secondly, the entropy reduction is now defined by (2), with $\mathcal{P} + \mathcal{Q}$ representing the parent node and $\mathcal{P}$ and $\mathcal{Q}$ representing the children.

- Locate a set of appropriate linguistic questions manually.
- For each phoneme p,
  - Estimate all p-triphone HMMs.
  - For each Markov state k in the model topology, classify all the k-th output distributions in all p-triphones.
    1. Add all the k-th output distributions of all p-triphones into the root, with the summed pds for the other states stored aside.
    2. Find the best composite question for each non-expanded node.
    3. Split the best node, in terms of maximum entropy reduction, according to the best composite question associated with it.
    4. Go to step 2 unless some convergence criterion is met.

Figure 4: The algorithm for output-distribution classification, with phone-dependency, state-dependency, and global information.

3.1.1. Composite Questions

At step 2 of Figure 4, composite questions, formed by conjunctions, disjunctions and/or negations of the simple linguistic questions, are built to alleviate the data fragmentation problem, in which members of a cluster are unable to move around between siblings once a decision is made, resulting in similar elements falling into different leaves. However, even with composite questions, the decision-tree clustering usually still has less freedom in reconfiguration than the SDM clustering does. This will be evidenced in the experiments in Section 4.

To form a composite question for a node, we grow a tree with simple questions under the node, using the same algorithm but with simple questions only. Next, we combine the leaves into two clusters, based on minimizing the total entropy of the two clusters. The composite question is then formed by disjoining all the paths to one of the two clusters. Each path is formed by conjoining all the questions along the path. For example, in Figure 5, the composite question at the root for the black cluster is

$$q_1 q_2 + q_1 \bar{q}_2 q_4 + \bar{q}_1 q_3$$

where the plus sign stands for disjunction, concatenation for conjunction, and the bar sign for negation.

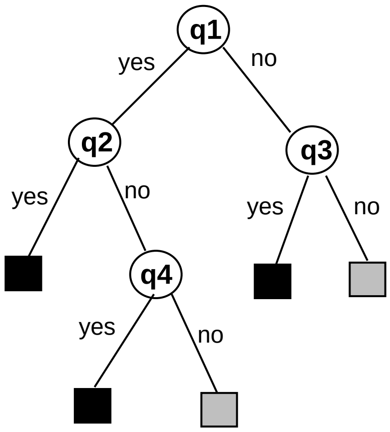

Figure 5: A composite question, $q_1q_2 + q_1\bar{q}_2q_4 + \bar{q}_1q_3$, for the root. q_i is the best simple question at each node. Black leaves satisfy the composite question. Shaded leaves do not.

The computation for grouping leaves into two clusters is an exponential function of the number of leaves under the subtree in question, which is at most the total number of elements beneath it (when all leaves are singleton clusters). This number is a factor less than the entire number of elements at the composite root, which the SDM distribution clustering always considers at step 3 of Figure 1. Our empirical results reflect this predicted cost savings, as the decision-tree clustering runs much faster than the bottom-up agglomerative clustering.

Given a configuration created by the bottom-up agglomerative clustering, it is possible to form an equivalent tree with each leaf corresponding to each cluster, as long as we allow the complexity of the composite questions to be unlimited and there are a sufficient number of detailed simple questions. At that extreme, the decision-tree clustering converges to the bottom-up agglomerative clustering.

3.1.2. Cross Validation

Cross validation, like deleted interpolation [12], is a technique to make the established system (e.g., the decision trees) more robust against new testing conditions (e.g., a new task). The basic idea is to divide the entire available data into two sets. For each data set, we train the set of triphone HMMs represented in the data. We then use one set of trained HMMs to build the decision trees, and the other to validate the trees. This validation keeps the tree structure fixed and prunes, from the leaves to the root, the nodes which have insignificant entropy reduction computed by the second set of triphone HMMs. That is, the second set of HMMs is used to validate the entropy reduction computed by the first set. When a vocabulary-independent system [4] is not pursued, cross validation becomes less important, especially when pruning is not serious. Particularly, in our experiments on the Wall Street Journal task, cross validation offered little advantage.

3.2. Tree Examples

Figures 3 and 6 show the partial trees for Markov state 1 and 5 of $/k/$-triphones when the 5-state Bakis topology is used [9]. It is clear that although the same global information is used for both trees, the entropy combining weights are able to select appropriate questions for different state locations.

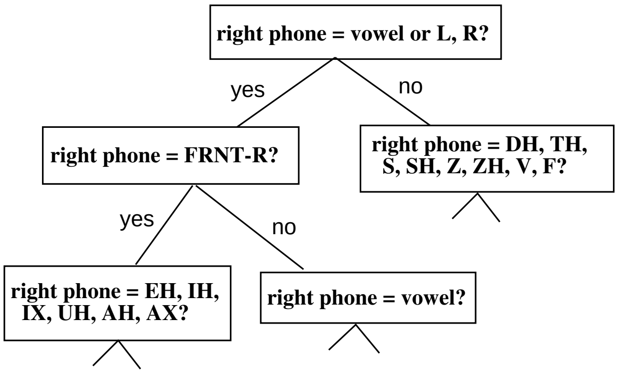

Figure 6: The partial decision tree for Markov state 5 of $/k/$-triphones.

4. PERFORMANCE AND DISCUSSION

4.1. The Database and SPHINX-II System

To evaluate the senonic decision tree, the SPHINX-II system for the DARPA 5,000-word WSJ continuous speech recognition task (non-verbalized punctuation) was chosen as our experimental testbed. The standard bigram language model, which has a test-set perplexity of 118, was used in all the experiments conducted here. There are 7200 utterances from 84 speakers in the official training set, resulting in approximately 23,000 triphones when position-dependent between-word and within-word triphones are considered [8, 18]. Triphones that occur only once or twice (about 6000 in total) were discarded to avoid noisy decision during clustering.

The development set consists of 367 utterances from 9 new speakers and has 2.2 unseen triphones per utterance on average[1]. The test set, evaluated in November 1992, consists of 330 utterances from 8 new speakers and has 2.6 unseen triphones per utterance. During recognition, about 48,000 triphones, including those crossing all word pairs due to the backoff language model, were pre-constructed. This number showed that a large portion of the triphones needed for decoding did not occur in the training data.

The 5-state Bakis topology was employed for all triphone HMMs. For the sake of simplicity, one-codebook discrete HMMs were used during clustering. To build the decision trees, we defined 43 linguistic questions manually, plus 3 questions about the word position of a triphone [4].

[1]It originally consists of 410 utterances from 10 speakers. The utterances of the outlier speaker 422, who has a fast speaking rate and a high-pitched voice, and the mis-transcribed utterance 053c100n are discarded.

The weights in (2) before normalization were:

$\lvert k-i \rvert$	0	1	2	3	4
$w_k(i)$	1.00	0.30	0.15	0.10	0.01

For instance, for the second Markov state, the net distortion measure was

$$\frac{0.3}{1.85} \triangle\tilde{H}(1) + \frac{1.0}{1.85} \triangle\tilde{H}(2) + \frac{0.3}{1.85} \triangle\tilde{H}(3) + \frac{0.15}{1.85} \triangle\tilde{H}(4) + \frac{0.10}{1.85} \triangle\tilde{H}(5)$$

where $1.85 = 0.3 + 1.0 + 0.3 + 0.15 + 0.1$.

After the cluster mapping table was generated, 4-codebook sex-dependent senonic semi-continuous HMMs (SCHMMs) [6] were trained by the forward-backward algorithm, during which the triphone output distributions were shared according to the mapping table. The four-codebook features included mel-frequency cepstral coefficients, 1st-order and 2nd-order differences, and power and its differences [7]. Cepstral vectors were normalized with the utterance-based cepstral mean. 7,000 senones were determined after tuning on the development set. Transition probabilities were context-independent. The Viterbi beam search [21, 20] was run in parallel on both male and female models; the word sequence with the higher score was output as the recognized sentence.

4.2. Performance and Discussion

4.2.1. Bottom-up versus Top-down Clustering

Table 1 shows the word error rates, including insertions, deletions, and substitutions, of several systems we constructed. The first two rows used context-independent phones to represent unseen triphones. The decision-tree based senone system performed slightly worse than the SDM (8.2% v.s. 7.8% word error rate), as expected, since the SDM was more flexible in considering all possible configurations for seen triphones.

The last column of Table 1 lists the number of unique senone sequences for all the triphones used. This is essentially the true number of different triphone models used in the system. For seen triphones, the number of unique senone sequences in the SDM system was more than 1.5 times of that of the decision-tree based senones. This demonstrated the SDM's freedom to reconfigure clusters, versus the senonic decision-tree's limited grouping under subtrees. Despite the limited reconfiguration, the senonic decision tree was able to achieve a minimal performance degradation through the help of linguistic information.

4.2.2. Effects of Modeling Unseen Triphones

The strength of decision trees resides in its ability to model unseen triphones using detailed parameters. When unseen triphones were added to the senonic decision-tree, the error rate was reduced to 7.3%, as the last row of Table 1 shows. As indicated in the table, the number of unique senone sequences climbed to a comparable level as that of the SDM system. It is obvious that with the same number of parameters, senonic systems (both the SDM and the senonic trees) were able to form many more triphone models (and thus more detailed models)

system	Dev set	Nov92 set	overall error rate	# unique senone sequ.
SDM	7.5%	8.2%	7.8%	14872
tree senones	8.0%	8.4%	8.2%	8921
+ unseen triphones	7.5%	7.1%	7.3%	13270

Table 1: Word error rates on the WSJ task.

than the generalized allophones. For example, had 7000/5 decision-tree based allophones been used, the system would always have had 7000/5 allophonic models even when unseen triphones were added. On the other hand, the senonic systems require more computation during decoding because the true number of triphone models is much larger. However, since the computation for semi-continuous output probabilities is only a small part of the computation in decoding and the search beam actually becomes smaller due to the more accurate models provided by senones, the total computation is empirically only doubled.

To further understand the contribution of unseen triphone modeling, we analyzed the results from the senonic decision-tree with and without unseen triphones. Table 2 illustrates the word error rates, in terms of the number of unseen triphones contained in each utterance. It also lists the number of qualified utterances among these 697 utterances. For the overall set, the error rate reduction by modeling unseen triphones was 11%. When the number of unseen triphones was at least 5, we were able to reduce the error by more than 20% [2]. Even for utterances that contained no unseen triphones, we believe the improved modeling for unseen triphones helped the decoder prune the wrong paths in which unseen triphones were needed.

# unseen triphones	# utter-ances	without unseens	with unseens	error rate reduction
≥ 0	697	8.2%	7.3%	11%
≥ 1	555	8.5%	7.5%	12%
≥ 3	276	9.9%	8.5%	14%
≥ 5	129	9.7%	7.6%	22%

Table 2: Word error rates of the senonic decision-tree with and without unseen triphones modeled by senones.

[2] Actually the Viterbi decoder we implemented here pruned at the first phoneme of a word, rather than at the second phoneme. Since most unseen triphones were between-word triphones, some additional error reduction made by modeling unseen triphones might be blocked.

4.2.3. Four-Codebook SCHMMs for Clustering

As described in Section 4.1, one-codebook discrete HMMs were used for clustering in the experiments conducted above. To get the best performance, we also trained a set of 4-codebook SCHMMs to generate the mapping table. To make feasible the training of 17,000 4-codebook triphone SCHMMs, we segmented the training data into phonemes by an existing set of HMMs, using the Viterbi alignment algorithm. Then phoneme-based training was run instead of utterance-based training.

Unfortunately, this 4-codebook mapping table did not yield a satisfactory improvement, as shown in Table 3, where the the word error rates using the 1-codebook mapping table were also included for a clear comparison. Perhaps the one-codebook discrete HMM had already offered enough information in terms of distortion measure, even though it was not so detailed as the 4-codebook SCHMMs.

system / test set	Dev		Nov92	
mapping table	1-codebook	4-codebook	1-codebook	4-codebook
SDM	7.5%	7.0%	8.2%	8.0%
tree senones + unseens	7.5%	7.1%	7.1%	7.0%

Table 3: Word error rates on the WSJ task, using respectively 1-codebook discrete HMMs and 4-codebook SCHMMs for the clustering procedure.

Interestingly, we found from Table 1 and Table 3 that the November evaluation set favored the unseen-triphone modeling. We believe it was mostly likely because there were more unseen triphones in certain utterances in the November set.

5. CONCLUSION

Accurate modeling of unseen triphones is important for large-vocabulary speech recognition with a limited amount of training data. This paper presents the senonic decision tree that not only inherits the merits of distribution sharing but also offers unseen triphone handling. A senonic decision tree is constructed for each individual context-independent phonetic Markov state, with the assistance of a global information about the entire HMM. As expected, the decision tree based senones perform slightly worse than the SDM when both approaches replace unseen triphones with context-independent phones. The strength of the senonic decision tree lies in its unseen triphone modeling, as evidenced by the substantial error rate reduction (11% – 22%).

Acknowledgements

This research was a large extension to Hon and Lee's work [5]. The authors would like to express their gratitude to Professor Raj Reddy's encouragement and support, and other members of the CMU speech group for their help.

References

[1] Abramson, N. **Information Theory and Coding**. McGraw-Hill, New York, 1963, 11-16 pp.

[2] Bahl, L., de Souza, P., Gopalakrishnan, P., Nahamoo, D., and Picheny, M. *Decision Trees for Phonological Rules in Continuous Speech*. in: **IEEE International Conference on Acoustics, Speech, and Signal Processing**. 1991, pp. 185–188.

[3] Chen, F. and Shrager, J. *Automatic Discovery of Contextual Factors Describing Phonological variation*. in: **DARPA Speech and Language Workshop**. 1989.

[4] Hon, H. *Vocabulary-Independent Speech Recognition: : The VOCIND System*. School of Computer Science, Carnegie Mellon University, February 1992.

[5] Hon, H. and Lee, K. *CMU Robust Vocabulary-Independent Speech Recognition System*. in: **IEEE International Conference on Acoustics, Speech, and Signal Processing**. Toronto, Ontario, CANADA, 1991, pp. 889–892.

[6] Huang, X. *Phoneme Classification Using Semicontinuous Hidden Markov Models*. **IEEE Transactions on Signal Processing**, vol. 40 (1992), pp. 1062–1067.

[7] Huang, X., Alleva, F., Hwang, M., and Rosenfeld, R. *An Overview of the SPHINX-II Speech Recognition System*. in: **ARPA Human Language Technology Workshop**. 1993.

[8] Hwang, M., Hon, H., and Lee, K. *Modeling Between-Word Coarticulation in Continuous Speech Recognition*. in: **Proceedings of Eurospeech**. Paris, FRANCE, 1989, pp. 5–8.

[9] Hwang, M. and Huang, X. *Shared-Distribution Hidden Markov Models for Speech Recognition*. **IEEE Transactions on Speech and Audio Processing**, vol. 1 (1993).

[10] Hwang, M. and Huang, X. *Subphonetic Modeling with Markov States — Senone*. in: **IEEE International Conference on Acoustics, Speech, and Signal Processing**. 1992.

[11] Jeffreys, H. **Theory of Probability**. Oxford University Press, 1948.

[12] Jelinek, F. and Mercer, R. *Interpolated Estimation of Markov Source Parameters from Sparse Data*. in: **Pattern Recognition in Practice**, edited by E. Gelsema and L. Kanal. North-Holland Publishing Company, Amsterdam, the Netherlands, 1980, pp. 381–397.

[13] Johnson, R. *Axiomatic Characterization of the Directed Divergences and their Linear Combinations*. **IEEE Transactions on Information Theory**, vol. IT-25 (1979).

[14] Juang, B. and Rabiner, L. *A Probabilistic Distance Measure for Hidden Markov Models*. **The Bell System Technical Journal**, vol. 64 (1985), pp. 391–408.

[15] Kailath, T. *The Divergence and Bhattacharyya Distance Measures in Signal Selection*. **IEEE Transactions on Communication Technology**, vol. COM-15 (1967), pp. 52–60.

[16] Kazakos, D. and Papantoni-Kazakos, P. *Spectral Distance Measures Between Gaussian Processes*. **IEEE Transactions on Automatic Control**, vol. AC-25 (1980), pp. 950–959.

[17] Kullback, S. **Information Theory and Statistics**. Dover, New York, 1959.

[18] Lee, C., Giachin, E., Rabiner, R., L. P., and Rosenberg, A. *Improved Acoustic Modeling for Continuous Speech Recognition*. in: **DARPA Speech and Language Workshop**. Morgan Kaufmann Publishers, San Mateo, CA, 1990.

[19] Lee, K. *Context-Dependent Phonetic Hidden Markov Models for Continuous Speech Recognition*. **IEEE Transactions on Acoustics, Speech, and Signal Processing**, April 1990, pp. 599–609.

[20] Schwartz, R., Chow, Y., Kimball, O., Roucos, S., Krasner, M., and Makhoul, J. *Context-Dependent Modeling for Acoustic-Phonetic Recognition of Continuous Speech*. in: **IEEE International Conference on Acoustics, Speech, and Signal Processing**. 1985, pp. 1205–1208.

[21] Viterbi, A. J. *Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm*. **IEEE Transactions on Information Theory**, vol. IT-13 (1967), pp. 260–269.