Predicting Unseen Triphones with Senones

Mei-Yuh Hwang, *Member, IEEE*, Xuedong Huang, *Member, IEEE*, and Fileno A. Alleva, *Associate Member, IEEE*

***Abstract*— In large-vocabulary speech recognition, we often encounter triphones that are not covered in the training data. These unseen triphones are usually backed off to their corresponding diphones or context-independent phones, which contain less context yet have plenty of training examples. In this paper, we propose to use decision-tree-based senones to generate needed senonic baseforms for these unseen triphones. A decision tree is built for each Markov state of each base phone; the leaves of the trees constitute the senone pool. To find the senone associated with a Markov state of any triphone, the corresponding tree is traversed until a leaf node is reached. The effectiveness of the proposed approach was demonstrated in the ARPA 5000-word speaker-independent Wall Street Journal dictation task. The word error rate was reduced by 11% when unseen triphones were modeled by the decision-tree-based senones instead of context-independent phones. When there were more than five unseen triphones in each test utterance, the error rate reduction was more than 20%.**

I. Introduction

THE shared-distribution model (SDM) [9] in the SPHINX-II system [8] is an effective method to make full use of limited amount of training data. It clusters Markov states instead of entire phonetic hidden Markov models (HMM's), leading to a set of clustered output distributions called senones [11]. Senones significantly improve recognition accuracy and provide a pronunciation-optimization capability.

The agglomerative SDM approach in [9] has complete freedom to form a shared configuration across different Markov states based on the training data. Because it is purely data driven, it is difficult to model a triphone that never occurs in the training data. These unseen triphones are usually backed off to the corresponding diphones or context-independent phones, which contain less context yet have plenty of training examples. In dictating large vocabulary tasks or switching to new tasks, we often run into new triphones. If we simply back off to context-independent phones or diphones, the quality of these less detailed models is often not good enough, leading to increased search time and search errors. Decision trees have been used to model allophones as a top-down generalization approach [3], [2], [6] that can be used to model unseen triphones. It replaces an unseen triphone with an existing triphone that has sufficient context similarity. To incorporate this prediction capability, we propose to use decision trees for Markov state modeling. The individual output distributions, not the entire phonetic HMM's, are classified by decision trees with linguistic binary questions on the tree nodes.

We build one decision tree for each Markov state of each base phone. In other words, we impose two constraints, for the sake of simplicity, while constructing our decision trees: *phone dependency*, which prohibits HMM output distributions of different phones from being clustered, and *state dependency*, which allows HMM output distributions to be merged only if they are associated with the same kth Markov state in the model topology. However, to relate the trees for different Markov states of the same base phone, global information about the entire phonetic model is utilized, which will be elaborated in Section II-B. To determine the associated senone for a given triphone state, the corresponding tree is traversed until a leaf is reached. The traversal is guided by answering the linguistic question associated with each nonleaf node, which has a yes-child and a no-child corresponding to the question. The senonic decision tree inherits the merits of the agglomerative SDM. More importantly, it also provides the ability to model unseen triphones.

In the ARPA 5000-word speaker-independent Wall Street Journal (WSJ) dictation experiments, we found that if unseen triphones were always represented by context-independent phone models, the decision-tree-based senone performed, as expected, slightly worse than the agglomerative SDM since the latter had much more freedom in optimizing the clustering of seen triphones. However, when the unseen triphones were modeled by the senonic decision tree, the tree-based approach was able to outperform the agglomerative SDM, which had no elegant method of modeling unseen triphones. We observed an 11% error rate reduction when the senonic decision tree predicted unseen triphones, which underlines the importance of accurately modeling unseen triphones. When there were at least five 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 accurately could help the decoder prune those wrong paths containing unseen triphones. The proposed method [10]–[12] was later used/modified in many systems with similar improvements [19], [4], [25].

The paper is organized as follows. Section II introduces an improved version of the bottom-up agglomerative SDM. Section III extends the shared-distribution principle to the senonic decision tree. Finally, in Section IV, we present performance results for the improved agglomerative SDM and the senonic decision tree, followed by a conclusion.

Manuscript received May 9, 1993; revised April 14, 1996. This work was supported 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. The associate editor coordinating the review of this paper and approving it for publication was Dr. Jean-Claude Junqua.

The authors were with the School of Computer Science, Carnegie-Mellon University, Pittsburgh, PA 15213 USA. They are now with Microsoft Corporation, Redmond, WA 98052 USA (e-mail: mhwang@microsoft.com).

Publisher Item Identifier S 1063-6676(96)08327-7.

Authorized licensed use limited to: MICROSOFT. Downloaded on January 7, 2009 at 21:44 from IEEE Xplore. Restrictions apply.

II. The Improved Agglomerative Shared-Distribution Model

A. Phone Dependency and State Dependency

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

The agglomerative clustering for the SDM is summarized in Fig. 1. It starts from single-element clusters with each HMM output distribution as the only element and merges consecutively the most similar pair of clusters. Each merge step deletes the two merged clusters and creates a new cluster that is represented by the average output distribution of the component elements. Moreover, to escape from the local optimal configuration, an element reshuffling step is performed after the merge step because otherwise, the algorithm would become a greedy algorithm, where an earlier decision could not be undone.

The distance measure between two clusters at step 2 of Fig. 1 is explained in detail in the following section for discrete output probabilities. A slight modification to replace summation with integration has to be made when applying it to continuous densities [15], [18]. In fact, it is easy to prove that this distance function also achieves the goal of minimizing the decrease in the likelihood of generating the training data as the one used in [25] for HMM's with continuous-density output functions. The convergence criterion at step 4 will be explained in Section III-A together with Fig. 3. More details can be found in [9]. The difference between the improved SDM and the one in [9] is underlined in the figure.

B. Use of Global Information

In addition to state dependency, global information about the entire phonetic model is also incorporated to measure the distance of two output distributions so that there is still some relationship between the clusterings of different state locations of the same base phone. To elaborate, let's start with the definition of entropy and cross entropy.

For each base phone p (there are about 50 basic phones in American English),

- Estimate all p-triphone HMMs (typically using Baum-Welch algorithm).
- For each Markov state k in the model topology (such as the 5-state Bakis topology), cluster all the k-th output distributions of all p-triphones:
  1. Create a singleton cluster for the k-th output distribution of each p-triphone.
  2. Merge the most similar pair of clusters. The dissimilarity measure will be explained in Section II-B and Formula (2).
  3. Re-shuffle: Move one element from one cluster to another if it results in an improvement. Here an improvement means a decrease in the total entropy of all clusters.
  4. Go to step 2 unless some convergence criterion is met (see more explanation in Section III-A).

Fig. 1. Bottom-up agglomerative algorithm for output-distribution clustering with phone dependency and state dependency.

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 i. 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 pd's [13], [20], [17], [16]. The cross entropy (directed divergence) [20] between two pd's $\mathcal{P}$ and $\mathcal{Q}$ is defined as

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

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 independently, given pd $\mathcal{Q}$

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

In other words,

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

$$\log \Pr(\text{data } \mathcal{P} \mid \text{pd } \mathcal{Q}) = \sum_i p_i \log q_i \le \sum_i p_i \log p_i$$

with equality being true 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 $\mathcal{P}$.

Distortion Measure at a Fixed Markov State: When two HMM output distributions $\mathcal{P}$ and $\mathcal{Q}$ are merged, the corresponding count entries are summed to get the occurrence counts for the resulting pd. We define the distortion incurred from merging two pd's (or the dissimilarity between two pd's)

Authorized licensed use limited to: MICROSOFT. Downloaded on January 7, 2009 at 21:44 from IEEE Xplore. Restrictions apply.

as the difference in entropy, weighted by the occurrence counts

$$\begin{aligned}\Delta\tilde{H} &\triangleq (P+Q)H(\mathcal{P}+\mathcal{Q}) - PH(\mathcal{P}) - QH(\mathcal{Q})\\ &= P\sum_i p_i \log\frac{p_i}{(p_iP+q_iQ)/(P+Q)}\\ &\quad + Q\sum_i q_i \log\frac{q_i}{(p_iP+q_iQ)/(P+Q)}\\ &= P\chi(\mathcal{P},\mathcal{P}+\mathcal{Q}) + Q\chi(\mathcal{Q},\mathcal{P}+\mathcal{Q}) \qquad (1)\end{aligned}$$

where P is the summation of the count entries in $\mathcal{P}$; 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. The notation H is used to remind us that the distortion measure is entropy related. The tilde sign above it means it is modified entropy (modified by counts).

Weighting cross entropy by the occurrence count takes into account how well each pd is trained. Pd's that appear infrequently (small counts) in the training data result in small distortion and thus will be merged first in comparison with those pd's that occur frequently in the training corpus. This makes the infrequent pd's 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 very big cluster, compared with $\mathcal{Q}$. 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 $\chi(\mathcal{Q},\mathcal{P}+\mathcal{Q})$. Formula (1) has been used by earlier systems such as [21].

Global Information Across Different Markov States: Formula (1) defines the distance between two pd's that are associated with a particular Markov state k of two triphones that represent the same base phone. To incorporate more information for estimating the distance between two Markov states and to ensure some relationship among the clusterings of different states of the same base phone, we propose to attach to each cluster not only the kth pd but also the pd's associated with the remaining Markov states. When two clusters are merged, all pairwise pd's are added to compute the resulting cluster. The net distortion, which arises from merging the kth pd's of two clusters, is then defined to be a weighted summation of the pairwise distortions

$$D(\mathbf{k}) \triangleq \sum_i w_{\mathbf{k}}(i)\Delta\tilde{H}(i) \qquad (2)$$

where $\Delta\tilde{H}(i)$ is defined in (1) with the ith pair of pd's considered, and $\sum_i w_{\mathbf{k}}(i) = 1.0$ in order to normalize the net distortions across different Markov states. $w_{\mathbf{k}}(i)$ is inversely proportional to the topological distance between state k, which is the state in question, and state i, which is one of the remaining states. This emphasizes the information close to the interesting spot k. In other words, the form of the clustering for each Markov state of the same base phone is exactly the same in the sense that all the pds for the entire phonetic HMM have to be computed. The difference is the combining weights in (2). When state 1 is considered, $\Delta\tilde{H}(1)$ gets the highest weight; when state 2 is considered, $\Delta\tilde{H}(2)$ has the maximal weight. When uniform weights are adopted, the clustering becomes exactly the same for different state locations of the same phone. That is, when the first states of two triphone models are in the same cluster, so will the other pairwise states. In other words, it degenerates to the generalized triphone. When $w_{\mathbf{k}}(i) = \delta_{ki}$, where $\delta_{ki} = 1$ if $k = i$, otherwise 0, information from neighboring states is not used. In [9], it was shown that the SDM with $w_{\mathbf{k}}(i) = \delta_{ki}$ was superior than the generalized triphone model due to its ability to allow partial triphone parameter sharing.

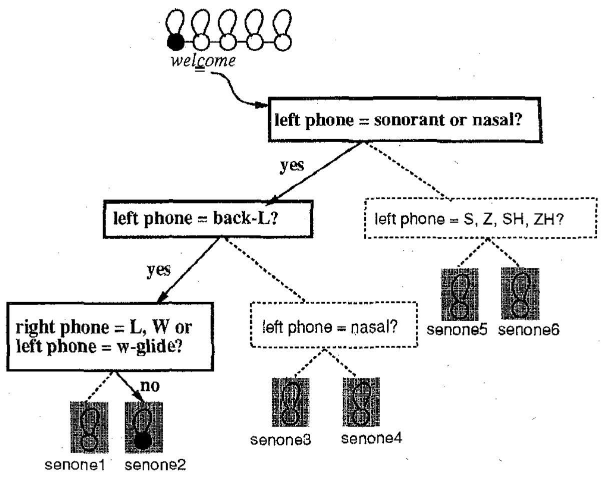

welcome
left phone = sonorant or nasal?
yes
left phone = back-L?
left phone = S, Z, SH, ZH?
yes
right phone = L, W or left phone = w-glide?
no
left phone = nasal?
senone1
senone2
senone3
senone4
senone5
senone6

Fig. 2. Decision tree for the first Markov state of /k/ triphones.

Formula (2) was the distortion measure for all the experiments conducted in this paper. We sometimes refer it as *entropy increase* after merging $\mathcal{P}$ and $\mathcal{Q}$, *entropy decrease* after splitting $\mathcal{P}+\mathcal{Q}$, or simply *delta entropy*.

III. The Senonic Decision Tree

To predict unseen triphones with senones, we extend the principle of distribution sharing to decision trees [6]. In particular, we build a decision tree for each Markov state of each base phone, as constrained by both phone dependency and state dependency. Our system has 50 phones for English. When there are five states for each phonetic HMM, that makes 250 trees in total. A decision tree is a binary tree, with a linguistic question associated with each nonleaf node. Fig. 2 shows an example tree for the first state of /k/ triphones.

To find the senone associated with any Markov state of any triphone, we traverse from the root of 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 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 instead of always using either diphones or monophones.

Authorized licensed use limited to: MICROSOFT. Downloaded on January 7, 2009 at 21:44 from IEEE Xplore. Restrictions apply.

1. Locate a (small) set of appropriate linguistic questions manually.
2. For each base phone p,
   - Estimate all p-triphone HMMs (typically using Baum-Welch algorithm).
   - For each Markov state k in the model topology, classify all the k-th output distributions in all p-triphones, using a binary tree:
     (a) Put all the k-th output distributions of all p-triphones into the root node.
     (b) Find the best *composite question* (see explanation in Section III-A) for the root and compute the delta entropy the question renders according to Formula (2). Insert the root into a heap of non-expanded nodes.
     (c) Remove from the heap the node with maximal delta entropy and split the node. By splitting, all the pds that come from the triphones which answer yes to the composite question go to the *yes*-child node; those which answer no to the *no*-child.
     (d) Find the best composite question and compute the delta entropy for each of the newly created children. Insert the children to the heap of non-expanded nodes.
     (e) Go to step (c) unless some stop-growing criterion is met.

Fig. 3. Algorithm for output-distribution top-down classification using binary trees.

A. Construction of the Decision Tree

Fig. 3 summarizes the construction of the decision tree. The linguistic questions selected at step 1 are simple categorical questions, querying about the left or right context of a triphone, such as "is the left/right context of the triphone a nasal?" or "is the triphone the first phone of a word?" Some of the useful categories that most English linguists agree are fricative, liquid, glides, round-vocalic, labial, alveolar, stops, high and low vowels, tense and lax vowels, and so on. Interested readers can find more details in [5].

Like the agglomerative SDM, each tree node is represented by a pd whose count entries are the summations of the count entries of all the component pd's. In addition, to bear the global information, pd's for the remaining 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.

Two fundamental differences between the work here and the work in [5] should be noted. First of all, the classification is on subphonetic units (i.e., the output distributions) rather than on entire phonetic HMM's. Second, the entropy reduction is now defined as (2) shows, with $\mathcal{P}+\mathcal{Q}$ representing the parent node and $\mathcal{P}$ and $\mathcal{Q}$ representing the children.

The stop criterion of Fig. 3 step 2(e) and that of Fig. 1 step 4 are similar. Useful criteria are such as minimal delta entropy and minimal occurrence counts of each node (cluster). Particularly, in our experiments, we solved the problem in two phases. In the first phase of the bottom-up agglomerative clustering, we did not stop merging until there was only one cluster left for each Markov state of each base phone. Similarly, for the top-down classification tree, we grew each tree to the fullest until a minimal entropy reduction failed.

In the second phase, we pruned the merges that had the most delta entropy or pruned the tree splits that had the least delta entropy until we expected the number of clusters (or leaves) left could be well trained by the given training corpus. This optimal number of parameters for a given training corpus often needs to be tuned by running recognition on some independent development data. This is true for almost all the speech recognition systems. Based on our experiences on the Resource Management task [22], we quickly tuned the number of senones to be around 7000 for the WSJ phase 0 training corpus, which will be elaborated in Section IV.

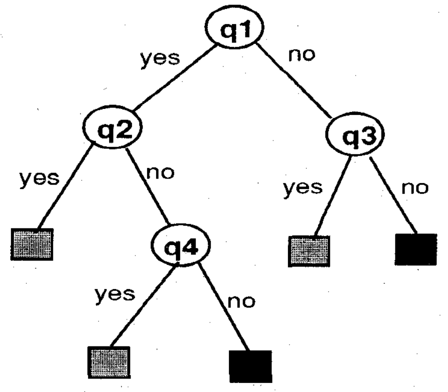


Fig. 4. Composite question $q_1\bar{q}_2\bar{q}_4+\bar{q}_1\bar{q}_3$ for the root. q_i is the best simple question at each node. Black leaves satisfy the composite question. Grey leaves do not.

Dividing the clustering task into the above two phases makes it unnecessary to do the pd merging or tree expanding over and over again for tuning the parameter size. The pruning phase takes little effort once the pd merging or tree growing is finished.

Composite Questions: At steps 2(b) and 2(d) of Fig. 3, composite questions, which are formed by conjunctions, disjunctions, and/or negations of the simple linguistic questions, are built to alleviate the data fragmentation problem, which results from the fact that once a split is made, members of a node are unable to move around between siblings, resulting in similar elements falling into different leaves.

To form a composite question for a node x, we first grow a tree of exclusively simple questions under x, using a similar algorithm with x as the root. We usually grow the *simple* tree by splitting only four to eight nodes in order to limit the number of leaves (note n splits result in $n+1$ leaves). Next, we try all combinations of distributing the leaves into two groups and select the one that has the minimal total entropy. Randomly mark one of the groups the black group, and mark the other one grey. The composite question is then formed by disjoining all the paths to the black group. Each path is formed by conjoining all the questions along the path. For example, in Fig. 4, the composite question for the root is

$$q_1\bar{q}_2\bar{q}_4+\bar{q}_1\bar{q}_3$$

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

Thus, with disjunction operations, we can pull similar leaves together to alleviate the data fragmentation problem. However, keep in mind that even with composite questions, the decision tree usually still suffers less freedom in reconfiguration compared with the reshuffling step of the bottom-up clustering in Fig. 1. This will be evidenced in the experiments in Section IV.

The computation for distributing the leaves of the simple tree rooted at node x into two groups is an exponential function

Authorized licensed use limited to: MICROSOFT. Downloaded on January 7, 2009 at 21:44 from IEEE Xplore. Restrictions apply.

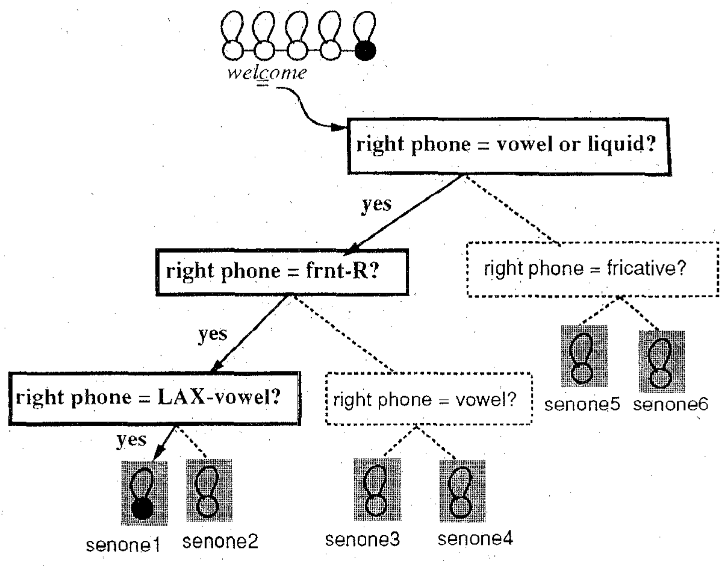


Fig. 5. Decision tree for the last Markov state of $/k/$ triphones.

of the number of leaves of the *simple* tree, which is, at most, the total number of elements in x, when each of the leaves contains exactly one element. This number is a factor less than the entire number of elements in the *composite* root, which the bottom-up distribution clustering always considers at the reshuffling of Fig. 1. Our empirical results reflected this predicted cost savings, as the decision-tree classification ran much faster than the bottom-up agglomerative clustering.

Given a configuration created by the bottom-up agglomerative clustering, it is, in theory, possible to form an equivalent tree with each leaf corresponding to exactly one of the final clusters if we allow the complexity of the composite question to be unlimited, and there are a sufficient number of detailed simple questions. At that extreme, the decision-tree classification converges to the bottom-up agglomerative clustering in terms of the reshuffling capability.

Cross Validation: Cross validation, like deleted interpolation [14], 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 a set of triphone HMM's represented in the data. We then use one set of trained HMM's to build the decision trees (phase one described in Section III-A), and the other to validate the trees (phase two). This validation is a process that prunes, from the tree built by the first set of triphone HMM's, those nodes with small entropy reduction calculated by the second set of triphone HMM's. When a vocabulary-independent system [5] is not pursued, cross validation becomes less important, especially when there is not much pruning. Particularly, in our experiments on the WSJ task, cross validation offered little advantage.

B. Tree Examples

Figs. 2 and 5 show our experimental partial trees for Markov state 1 and 5 of $/k/$ triphones when the five-state Bakis topology is used[1] [9]. It is clear that although the same form of global information is used for both trees, the entropy combining weights are able to select appropriate questions for different state locations.

IV. Performance and Discussion

A. Speech Corpus and SPHINX-II

To evaluate the senonic decision tree, the SPHINX-II system for the ARPA 5000-word WSJ speaker-independent continuous speech recognition task (nonverbalized 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 corpus, resulting in approximately 23 000 triphones when position-dependent between-word and within-word triphones are considered. Triphones that occur only once or twice (about 6000 in total) were discarded to avoid noisy decision during clustering.

The development set (which is denoted as si_dev5b) consists of 367 utterances from nine new speakers and has 2.2 unseen triphones per utterance on average.[2] The test set, which was evaluated in November 1992, consists of 330 utterances from eight new speakers and has 2.6 unseen triphones per utterance. During recognition, about 48 000 triphones were preconstructed, including those crossing all possible word-pairs due to the backoff language model. This number showed that a large portion of the triphones needed for decoding did not occur in the training data.

The five-state Bakis topology with skipping arcs was employed. For simplicity, one-codebook discrete HMM's were used for the agglomerative clustering and tree construction. For

[1] For simplicity, the graph does not show the arcs that skip states.

[2] 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 mistranscribed utterance 053c100n are discarded.

Authorized licensed use limited to: MICROSOFT. Downloaded on January 7, 2009 at 21:44 from IEEE Xplore. Restrictions apply.

building the decision trees, we defined 43 linguistic questions manually plus three questions about the word position of a triphone, for the first step of Fig. 3.

To decide the weights in (2), we put more weights on closer states since neighboring speech has more influence than far-away speech. According to our laboratory experiments, putting small weights on immediate neighboring states gave slightly better accuracy than no weights at all for neighbors. Therefore, we empirically decided the following weights for the experiments reported in this paper:

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

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

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

where $1.85 = 0.3 + 1.0 + 0.3 + 0.15 + 0.1$ as $\sum w_k(i) = 1$.

At the end of the agglomerative clustering or the decision-tree classification, we obtain a *senone mapping table* that maps each state of each triphone to a senone label. After the senone mapping table was generated, 4-codebook sex-dependent semi-continuous HMM's (SCHMM's) [7] were trained by the forward-backward algorithm, during which the triphone output distributions were tied according to the senone mapping table. The 4-codebook features included mel-frequency cepstral coefficients, first- and second-order differences, and power and its differences [8]. Cepstral vectors were normalized with the utterance-based cepstral mean. Both the discrete and semicontinuous HMM's had a VQ size of 256. Seven thousand senones were determined after tuning on the development set. Transition probabilities were context independent. The Viterbi beam search [24], [23] was run on male and female models separately for each testing utterance; the word sequence with the higher score was output as the recognized sentence.

B. Performance and Discussion

Agglomerative SDM's versus Decision Tree-Based Senones: Table I 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 senonic system performed slightly worse than the agglomerative SDM (8.2% versus 7.8% word error rate), as expected, since the agglomerative approach was more flexible in considering all possible configurations for seen triphones.

The last column of Table I lists the number of unique senone sequences for all the triphones used. This is essentially the true number of unique triphone models. For seen triphones, the number of unique senone sequences in the agglomerative SDM system was more than 1.5 times that of the decision-tree-based senones. This demonstrated the agglomerative approach'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.

TABLE I
WORD ERROR RATES ON THE WSJ TASK. *unseen* STANDS FOR UNSEEN TRIPHONES. *ci* STANDS FOR CONTEXT-INDEPENDENT PHONES

system	si_dev5b	nov92	overall	# unique senone seq.
agglomerative SDM (unseen = ci)	7.5%	8.2%	7.8%	14872
tree senones (unseen = ci)	8.0%	8.4%	8.2%	8921
tree senones (unseen = senones)	7.5%	7.1%	7.3%	13270

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 represented by the decision tree-based senones instead of by context-independent phones, the error rate was reduced to 7.3%, as the last row of Table I shows.[3] As indicated in the table, the number of unique senone sequences climbed to a comparable level as that of the agglomerative system. It is obvious that with the same number of parameters, senonic systems (both the agglomerative and the tree systems) were able to form many more triphone models (and thus more detailed models) 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.

To further understand the contribution of unseen triphone modeling, we analyzed the results from the senonic decision tree with and without senonic unseen triphones. Table II 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 in each utterance, we were able to reduce the error by more than 20%.[4] Even for utterances that contained no unseen triphones, we believe the improved modeling for unseen triphones helped the decoder prune those wrong paths in which unseen triphones were needed.

Four-Codebook SCHMM's for Clustering: As described in Section IV-A, one-codebook discrete HMM's were used for clustering in the experiments conducted above. To get the best performance, we also trained a set of 4-codebook SCHMM's

[3] Our official result on the November 1992 set was 6.9%, which used a close variant of this system and scored the best in the November 1992 evaluation.

[4] Actually, the Viterbi decoder we implemented here pruned at the beginning (rather than at the end) of the first phone of a word. Since most unseen triphones were between-word triphones, some additional error reduction made by modeling unseen triphones might have been blocked.

Authorized licensed use limited to: MICROSOFT. Downloaded on January 7, 2009 at 21:44 from IEEE Xplore. Restrictions apply.

TABLE II
WORD ERROR RATES OF THE DECISION-TREE-BASED SENONES ON SI_DEV5B + NOV92, WITH AND WITHOUT MODELING UNSEEN TRIPHONES BY SENONES

# unseen triphones	# utterances	unseen=ci	unseen=senones	improvement
≥ 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 III
WORD ERROR RATES ON THE WSJ TASK, USING 4-CODEBOOK SCHMM'S VERSUS 1-CODEBOOK DHMM'S FOR THE CLUSTERING PROCEDURE TO GENERATE THE SENONE MAPPING TABLE. NOTE THE FINAL MODELS FOR RECOGNITION WERE ALWAYS 4-CODEBOOK SCHMM'S

system	si_dev5b		nov92	
	1cdbk	4cdbk	1cdbk	4cdbk
agglomerative SDM (unseen = ci)	7.5%	7.0%	8.2%	8.0%
tree senones (unseen = senones)	7.5%	7.1%	7.1%	7.0%

for the purpose of generating the senone mapping table. To make feasible the training of 17 000 4-codebook triphone SCHMM's, we segmented the training data into phones by an existing set of HMM's, using the Viterbi alignment algorithm. Then, phone-based training was run instead of sentence-based training.

Unfortunately, this 4-codebook mapping table did not yield a satisfactory improvement as shown in Table III. The results of using 1-codebook mapping table were copied from Table I. This could be explained by the fact that the 1-codebook feature actually included cepstrum, power, and their first-order differences in one big vector. What the 4-codebook features offered additionally was only the second-order differences and the assumption of four independent feature streams. While SCHMM's offer better smoothing through multiple VQ codewords for every frame of speech, smoothing is not as important in measuring the similarity between Markov states as it is in recognizing new data. Therefore, as far as the distortion measure for constructing the senone mapping table was concerned, the 4-codebook SCHMM was only marginally better than the 1-codebook DHMM.

Interestingly, we found from Tables I and III that the November evaluation set favored the unseen-triphone modeling. We believe it was mainly because there were more unseen triphones in certain utterances in the November set.

V. 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 Markov state of each base phone, with the assistance of a global information about the entire phonetic HMM. The strength of the senonic decision tree lies in its detailed unseen triphone modeling, as evidenced by the substantial error rate reduction (11–22%).

ACKNOWLEDGMENT

The authors would like to express their gratitude to R. Reddy's encouragement and support, H. W. Hon and K. F. Lee's earlier work, and other members of the CMU speech group for their help.

REFERENCES

[1] N. Abramson, *Information Theory and Coding*. New York: McGraw-Hill, 1963, pp. 11–16.
[2] L. R. Bahl, P. V. de Souza, P. S. Gopalakrishnan, D. Nahamoo, and M. A. Picheny, "Decision trees for phonological rules in continuous speech," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, May 1991, pp. 185–188.
[3] F. Chen and J. Shrager, "Automatic discovery of contextual factors describing phonological variation," in *Proc. DARPA Speech Natural Language Workshop*, Feb. 1989, pp. 284–289.
[4] V. Digalakis and H. Murveit, "Genones: Optimizing the degree of mixture-tying in a large vocabulary hidden Markov model based speech recognizer," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 1, 1994, pp. 537–540.
[5] H. W. Hon, "Vocabulary-independent speech recognition: The VOCIND system," Thesis, School of Comput. Sci., Carnegie Mellon Univ., Feb. 1992.
[6] H. W. Hon and K. F. Lee, "CMU robust vocabulary-independent speech recognition system," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Toronto, Ontario, Canada, May 1991, pp. 889–892.
[7] X. D. Huang, "Phoneme classification using semicontinuous hidden Markov models," *IEEE Trans. Signal Processing*, vol. 40, pp. 1062–1067, 1992.
[8] X. D. Huang *et al.*, "The SPHINX-II speech recognition system: An overview," *Comput. Speech Language*, vol. 2, pp. 137–148, 1993.
[9] M. Y. Hwang and X. D. Huang, "Shared-distribution hidden Markov models for speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 1, no. 4, pp. 414–420, Oct. 1993.
[10] ——, "Acoustic classification of phonetic hidden Markov models," in *Proc. Eurospeech*, Sept. 1991.
[11] ——, "Subphonetic modeling with Markov states—Senone," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Processing*, vol. I, Mar. 1992, pp. 33–36.
[12] M. Y. Hwang, X. D. Huang, and F. Alleva, "Predicting unseen triphones with senones," in *IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. II, Apr. 1993, pp. 311–314.
[13] H. Jeffreys, *Theory of Probability*. Oxford, UK: Oxford University Press, 1948.
[14] F. Jelinek and R. L. Mercer, "Interpolated estimation of Markov source parameters from sparse data," in *Pattern Recognition in Practice*, E. S. Gelsema and L. N. Kanal, Eds. Amsterdam, The Netherlands: North-Holland, 1980, pp. 381–397.
[15] R. Johnson, "Axiomatic characterization of the directed divergences and their linear combinations," *IEEE Trans. Inform. Theory*, vol. IT-25, no. 6, 1979.
[16] B. H. Juang and L. R. Rabiner, "A probabilistic distance measure for hidden Markov models," *Bell Syst. Tech. J.*, vol. 64, no. 2, pp. 391–408, Feb. 1985.
[17] T. Kailath, "The divergence and Bhattacharyya distance measures in signal selection," *IEEE Trans. Commun. Technol.*, vol. COM-15, no. 1, pp. 52–60, 1967.
[18] D. Kazakos and P. Papantoni-Kazakos, "Spectral distance measures between Gaussian processes," *IEEE Trans. Automat. Contr.*, vol. AC-25, no. 5, pp. 950–959, Oct. 1980.
[19] F. Kubala *et al.*, "Comparative experiments on large vocabulary speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. 1, Apr. 1994, pp. 561–564.

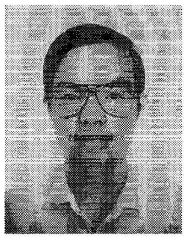
Authorized licensed use limited to: MICROSOFT. Downloaded on January 7, 2009 at 21:44 from IEEE Xplore. Restrictions apply.

[20] S. Kullback, *Information Theory and Statistics*. New York: Dover, 1959.
[21] K. F. Lee, "Context-dependent phonetic hidden Markov models for continuous speech recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, Apr. 1990, pp. 599–609.
[22] P. J. Price, W. Fisher, J. Bernstein, and D. Pallett, "A database for continuous speech recognition in a 1000-word domain," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Apr. 1988, pp. 651–654.
[23] R. Schwartz *et al.*, "Context-dependent modeling for acoustic-phonetic recognition of continuous speech," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Apr. 1985, pp. 1205–1208.
[24] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. IT-13, no. 2, pp. 260–269, Apr. 1967.
[25] S. J. Young, J. Odell, and P. Woodl, "Tree-based state tying for high accuracy acoustic modeling," in *Proc. ARPA Human Language Technol. Workshop*, Mar. 1994, pp. 307–312.

Mei-Yuh Hwang (S'91–M'96) was born in Taiwan in 1964. She received the B.Sc. degree in computer science from National Taiwan University, Taipei, ROC, in 1986, the M.Sc. degree in computer science from Carnegie Mellon University (CMU), Pittsburgh, PA, in 1989, and the Ph.D. degree in computer science from CMU in 1993.

From 1987 to 1993, she was a Ph.D. student at CMU, where she worked on between-word triphone and subphonetic acoustic modeling in the spoken language understanding group and published over 15 conference papers. Since graduation, she has been working at Microsoft Corporation, Redmond, WA, as a researcher in speech recognition. Her current interests lie in speech recognition, speech synthesis, stochastic language modeling, neural networks. and natural language understanding.

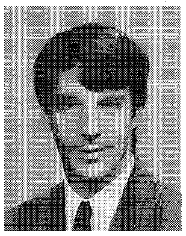
Dr. Hwang received the 1992 Allen Newell Research Excellence Medal from CMU and is a member of Phi Tau Phi.



Xuedong Huang (M'89) was born in China on October 20, 1962. He received the B.Sc. degree in computer science from Hunan University, Changsha, China, in 1982, the M.Sc. degree in computer science from Tsinghua University, Beijing, China, in 1984, and the Ph.D. degree in electrical engineering from the University of Edinburgh, Edinburgh, UK, in 1989.

He joined Carnegie Mellon University (CMU), Pittsburgh, PA, in 1989 and directed CMU's Sphinx-II speech recognition project from 1991 to 1993. He currently holds an adjunct Senior Research Computer Scientist position at CMU. In 1993, he joined Microsoft Research, Redmond, WA, as a Senior Researcher/Research Manager, where he has been managing Microsoft's speech recognition and speech synthesis efforts. His current interests lie in speech recognition, text to speech, human factors, spoken language systems, neural networks, and digital signal processing.

Dr. Huang served as an associate editor for IEEE TRANSACTIONS ON SPEECH AUDIO PROCESSING from 1992 to 1995. He has authored one book (*Hidden Markov Models for Speech Recognition* (Edinburgh, UK: Edinburgh University Press, 1990)) and more than 50 papers on spoken language technology. His awards include the National Education Commission of China's 1987 Science and Technology Progress Award, the IEEE 1993 Paper Award in the speech processing area, and the 1992 Allen Newell Research Excellence Medal.



Fileno A. Alleva (A'88) received the B.S degree in mathematics from Carnegie Mellon University (CMU), Pittsburgh, PA, in 1980.

He joined Microsoft Research, redmond, WA, in 1993 as a researcher in the Speech Technology Group. Before joining Microsoft, he contributed to the Sphinx I and Sphinx II systems as a Project Scientist at CMU. His current professional interests are in all areas of automatic speech recognition, particularly heuristic search and language modeling.

Authorized licensed use limited to: MICROSOFT. Downloaded on January 7, 2009 at 21:44 from IEEE Xplore. Restrictions apply.