LEXICAL TREE BASED PRUNING FOR SEGMENT-BASED SPEECH RECOGNITION [1]

Mike Phillips and David Goddeau

Spoken Language Systems Group
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139, U.S.A.

ABSTRACT

The goal of a fast match for continuous speech recognition is to efficiently reduce the set of possible words which may start at any point in time. In order to save computation, the fast match must use less computation than the complete search, and in order to not reduce accuracy, the fast match must not prune away words which would have been correctly recognized by the complete search.

The segmental approach employed in the MIT SUMMIT speech recognition system allows the use of a fairly simple and efficient fast match for large vocabulary continuous speech recognition. In the SUMMIT system, a segmental network is computed before the lexical search is performed. This network represents possible phonetic interpretations of the speech signal, and contains a vector of probabilities for all possible phones for each segment. The fact that this network is precomputed allows the use of an efficient lookahead to determine the possible words which may begin at any node in the network.

During training the phones are collapsed into a smaller set of phone classes and the pronunciations of the words in the lexicon are represented as a tree in terms of these phone classes. This tree is pruned to a depth of N for a lookahead of N phones. During recognition, to find the words which may begin at any node in the segment network, the lexical tree is matched against the phonetic network to find all nodes for which the phone class probabilities exceed some threshold. Since the phone class lexical tree has many less nodes than the total number of pronunciation arcs in the full lexicon, this search can proceed very quickly.

This paper will present the algorithm in more detail, discuss methods to select the phone classes, and explore the tradeoffs between length of the lookahead, phone class pruning threshold, computational savings, and accuracy.

INTRODUCTION

SUMMIT is a segment-based, speaker-independent, continuous speech recognition system[2]. In this segment-based approach the statistical models operate on hypothesized phonetic segments which allows the models to make explicit use of both the static and dynamic properties of segments and their surroundings. While it would be possible to consider all possible segmentations of the acoustic signal, for efficiency reasons, SUMMIT analyzes the acoustic signal to produce a segment network during the first stages of processing. The nodes of this network, specified as time points, represent putative phone boundaries. The arcs of this network are regions of the acoustic input representing putative phones. The segmentation process produces a network rather than a linear chain due to the uncertainties of segmenting based only on acoustic information.

The SUMMIT lexicon consists of a set of phonetic networks, each of which represents pronunciations of a word. These pronunciations are networks rather than phone strings to account for multiple pronunciations, phone deletions, etc.

While a number of different lexical search strategies have been employed in SUMMIT, all of these make use of an initial Viterbi search to find the best match between the acoustic network and a sequence of lexical networks. This phase of processing takes the majority of total computation time and is the part which scales with vocabulary size (the computation needed to produce the segment network for example, is independent of vocabulary size). This paper will therefore focus on the computation needed in the Viterbi search and on ways to reduce it.

RECOGNITION TASK AND SPEECH CORPUS

The experiments in the following sections were all performed using a version of SUMMIT with an approximately 4000 word recognition vocabulary, developed for the GALAXY system[3]. In order to run the experiments efficiently, the system used for these experiments used only context-independent models and a bigram language model.

The acoustic models used in the experiments were

[1] This research was supported by ARPA under Contract N00014-89-J-1332, monitored through the Office of Naval Research.

trained on xxx utterances from the ATIS[4] and VOYAGER corpora. The bigram was trained from a collection of sentences from the ATIS and VOYAGER corpora, plus a small set of sentences from other domains. The bigram and acoustic models are constant throughout the experiments. The testing set for all of the experiments was a set of 509 VOYAGER utterances from 12 speakers that were not included in the training of either the acoustic or language models.

VITERBI PRUNING

The computation needed by the full Viterbi search can be significantly reduced by pruning nodes which fall below some score threshold. There are various ways to perform this pruning. Typically, at every time point in the search (boundary in the segment network in the case of SUMMIT), a decision is made to retain only a subset of the nodes at this time point for future path extensions. This subset can be obtained by either retaining the M highest scoring nodes at this time point, or by retaining all nodes with scores within some threshold of the highest scoring node at this time point.

Figure 1 shows the result of using this score-threshold based pruning within SUMMIT's Viterbi search. The figure shows the word error rate of the system versus the overall percentage of nodes which remain active after the pruning. The points in the plot are obtained by varying the pruning threshold.

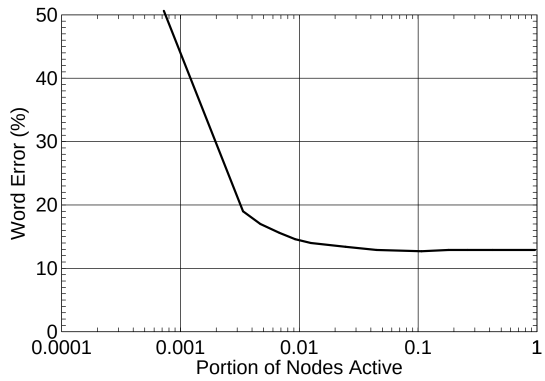

Figure 1: Word Error rate versus portion of nodes active.

While this figure shows that significant pruning can be achieved with only a moderate drop in accuracy (for example, the active nodes can be reduced by almost a factor of 20 with only a xx% increase in word error rate), closer inspection of the behavior of this pruning indicates possible improvement. Figure 2 plots the percentage of nodes active after pruning as a function of the distance from the beginning of the word to which they belong. This is plotted for a number of different score thresholds. It can be seen that pruning is more effective at nodes which occur eariler in the words, rather than later. This is to be expected since each surviving end-of-word node can result in the creation of many new begin-of-word node, and it may take a number of phones of acoustic evidence to prune away acoustically unlikely words.

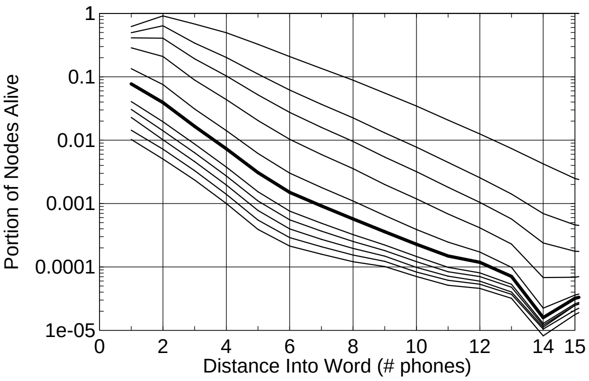

Figure 2: Portion of nodes active versus distance into word for a number of different pruning thresholds. The darker line corresponds to the pruning threshold that results in the first increase in word error rate (can be seen in Figure 1 at approximately 2.5% of nodes active).

This analysis points to a further refinement of the search. Since with the simple score-threshold based pruning, most of the computation is performed at the beginnings of words, we could benefit from a mechanism which is able to quickly reduce the set of words which may begin at any point in time.

PHONE-TREE BASED LOOKAHEAD

In order to reduce the set of words which may begin at any given point in time, we need some way to look forward in time to find the set of words which may match the future acoustics. However, since this must be performed at every time boundary (which may be 100-200 per second of speech), we need to be able to compute this lookahead very efficiently.

Fortunately, in SUMMIT, we have already precomputed a segment-network with scores for all phone possibilities computed for every acoustic segment. Using this along with the fact that many words share the same or similar initial phone sequences allows us to efficiently produce a subset of words which may begin at any point in time. This is accomplished by using a phone tree which captures the pronunciation prefixes of all words in the lexicon. By matching this tree against the acoustic network at each boundary, large sets of words can be simultaneously pruned.

Related Work

If any.....

Implementation

Two main implementation issues are presented by the phone-tree based pruning technique: the construction of the tree itself, and the efficient matching of the tree against the acoustic network.

Producing a phone-tree from a set of lexical networks is fairly straightforward. First, a tree containing all phone strings of a given length is constructed. Each node in this tree corresponds to a unique phone sequence. At each node, the list of words which can begin with that phone sequence is stored. With multiple pronunciations, a given word may appear in several lists. The resulting tree is then pruned, eliminating nodes which don't correspond to any words. The size of the resulting tree is a function of its depth and the size of the lexicon.

Matching the tree against the acoustic network is done at each phone boundary. In the normal Viterbi search, each word that ends at a given boundary propagates its score to the start nodes of every word which can follow (possibly adding a language model score as part of the propagation). The tree-match immediately precedes this computation and allows us to propagate scores only to words that are likely candidates.

To match the tree from a given boundary, the phone models corresponding to the arcs of the tree are either kept or pruned based on the score for that phone which has been precomputed for each acoustic segments and a list of active words is accumulated. Each path through the tree is scored against paths through the acoustic network. If a leaf node is reached with all scores for the sequence of phones leading to that node above a pruning threshold, all words associated with that node are added to the active word list. This computation takes time proportional to the number of nodes in the tree. At its completion, the score propagation is performed from words ending at this boundary to the start nodes of words on the active list.

Experimental Results

The goal of the phone-tree based pruning is to reduce the amount of computation required by the recognizer without significantly reducing recognition performance (measured here by word accuracy). The tradeoff between word accuracy and computation is determined by the properties of the lexicon and recognition system, the pruning threshold, and the depth of the phone tree.

Figure 3 shows word error rate versus portion of nodes alive for varying depths of the lookahead tree (number of phones in the lookahead).

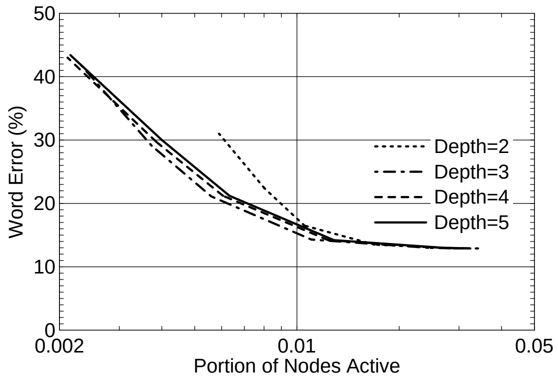

Figure 3: Word Error rates versus portion of nodes active for different depths of the lookahead tree (the points on each curve are obtained by varying the phone-tree pruning threshold)

PHONE-CLASS SEGMENT-TREE BASED LOOKAHEAD

It should be possible to reduce the size of the phone-prefix tree (and therefore the amount of computation needed to match the tree to the acoustic network) by collapsing the phones into classes. If similar phones tend to be pruned together (whenever phone x is pruned, phone y is usually pruned also), collapsing phones into classes for constructing and matching the tree will not significantly reduce the effectiveness of the pruning.

Use of Phone Classes

We can construct phone-class lexical trees simply by collapsing labels into the phone classes before constructing the tree. The leaf nodes in the tree then represent all words which begin with a given phone class sequence.

To match phone-class lexical trees to the acoustic network, we consider that a phone class should *not* be pruned if *any* of the phones in the class hae a score which exceeds the pruning threshold. This results in a list of unpruned phone classes for each segment in the segment network which can be used in the phone-tree matching procedure described previously.

Determining Phone Classes

We would like to choose the phone classes such that on average, the smallest number of phone classes are unpruned at any given point in time (for a given pruning threshold). Since the criterion we use in the search is that each phone class is considered to be unpruned if *any* of its members exceed the pruning threshold, we should select classes which minimize the number of instances where each phone has a score above the pruning threshold when other members of the class do *not* have scores which exceed the threshold over some training corpus.

We have selected classes using a bottom-up clustering procedure which makes use of this criterion. Starting with each phone it its own class, we merge the two classes where the combination minimizes the increase in the criterion described above. This step-wise optimization procedure results in a tree of phone classes. We can select any number of phone classes simply by stopping the merging at that number of classes.

Figure 4 shows the effects of phone class size on the size of the resulting lexical tree. It can be seen...

Figure 5 shows the effects of phone class size of performance of the lookahead. This figure shows word error rate versus portion of nodes alive for various class sizes.

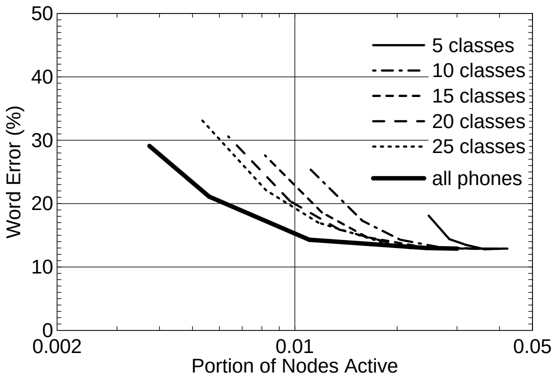

Figure 4: Word Error rates versus portion of of nodes active for different numbers of phone broad classes (the points on each curve are obtained by varying the phone-tree pruning threshold)

INTERACTION OF LOOKAHEAD AND VITERBI PRUNING

As the previous figures indicate, both the Viterbi pruning and lookahead can achieve computation reductions at some cost in accuracy. Since the two forms of pruning may be able prune different sets of nodes, it is likely that some combination of the two will result in the best trade-off between accuracy and computation. To investigate these trade-offs, we computed error rates versus portion of active nodes for a variety of combinations of Viterbi pruning and lookahead thresholds. Figure 6 show the results of these experiments using 15 phone classes and a lookahead of 3 phones.

COMPUTATION

So far, we have only shown computation needs in terms of portion of nodes alive rather than actual computation amounts. While this is a more clear measure since it is independent of implementation, in the end a tradeoff must be made between the actual amounts of computation needed to compute the lookahead and the computation saved by the use of this lookahead.

We have computed computation time needed...

DISCUSSION AND FUTURE WORK

This paper has described the application of phone-tree based pruning to improving the efficiency of Viterbi search in a segment-based recognition system. This pruning has resulted in a decrease in computation time in the search with little effect on error rate. We believe that a combination of phone-tree based pruning and Viterbi pruning provides a further decrease in computational effort.

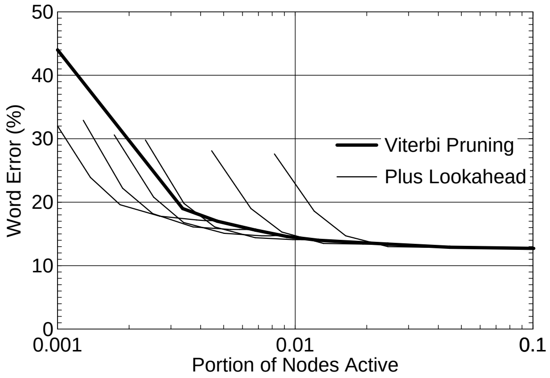

Figure 5: Word Error rates versus portion of nodes active for different combinations of Viterbi pruning and lookahead tree pruning thresholds. The darker line shows the results of Viterbi pruning alone. Each lighter line show the results of increasing amounts of lookahead pruning.

REFERENCES

[1] Zue, V., Glass, J., Goddeau, D., Goodine, D., Hirschman, L., Phillips, M., Polfroni, J., and Seneff, S. "The MIT ATIS System: February 1992 Progress Report," *Proc. DARPA Speech and NL Workshop*, 84–88, Harriman, NY, February 1992.

[2] Zue, et al., "The MIT SUMMIT Speech Recognition System: A Progress Report", *Proc. DARPA Speech and Natural Language Workshop*,179-189,Philadelphia, February 1989.

[3] Zue, et al., "GALAXY..., *these Proceedings*.

[4] MADCOW, "Multi-Site Data Collection for a Spoken Language Corpora", *Fifth DARPA Speech and Natural Language Workshop*,February,1992.

[5] Zue V., Daly, N., Glass, J., Goodine, D., Leung, H., Phillips, M., and Polifroni, J., Seneff, S., and Soclof, M., "The Collection and Preliminary Analysis of a Spontaneous Speech Database", *DARPA Speech and Natural Language Workshop*, 160-167,October,1989.