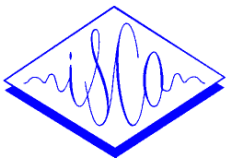

3rd European Conference on
Speech Communication and Technology
EUROSPEECH'93
Berlin, Germany, September 19-23, 1993

A* WORD NETWORK SEARCH FOR CONTINUOUS SPEECH RECOGNITION[1]

I. Lee Hetherington, Michael S. Phillips, James R. Glass, and Victor W. Zue

Spoken Language Systems Group
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139 USA

ABSTRACT

In this paper we describe an algorithm for generating word networks in a continuous speech recognition system. Recently, N-best search strategies have gained considerable popularity and have been used for multi-stage searches including interfacing speech recognition and natural language systems as well as applying more computationally expensive constraints in later stages. However, examination of N-best lists reveals significant overlap between different hypotheses, with differences typically localized to regions where the acoustic signal is not robust. In order to improve both computational and representational efficiencies, we have developed a word network search. This search is very similar to our A* N-best search, but contains an additional path-merging step. The resulting word networks contain the same N complete hypotheses that are within a specified score threshold of the best complete score, but in a much smaller form that is faster to compute. These word networks can then be used as search spaces by subsequent search stages.

INTRODUCTION

Until recently, most speech recognition systems have only been faced with the task of producing the single best word string for a given input utterance. Because of this, the search strategies used (e.g., the Viterbi search [1]) were typically able to efficiently find the top scoring word string, but were not able to provide any other sentence hypotheses. Furthermore, typically these search strategies were only been able to make use of local grammatical constraints. However, with recent research effort in developing speech understanding systems [2] (i.e., systems that combine speech recognition with language understanding to perform interactive problem solving) it has become desirable to either integrate more complex language models into the search, or to have the speech recognition component provide multiple sentence hypotheses, which can then be filtered by the natural language component.

Initial work in combining speech recognition and natural language technology used a modification of the Viterbi search to provide the N-best sentence hypotheses [3]. This work showed that at least for some tasks, the correct answer was very often in the N-best sentence list for fairly small N, and therefore an N-best list would provide a useful interface between a speech recognition system and a natural language parser. Based on this success, other more efficient N-best search strategies were developed, including other modifications of the Viterbi search [4, 5] as well as algorithms based on the A* search [4, 6, 7].

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

N-best algorithms have found widespread use in systems that combine speech recognition and natural language understanding [4, 5]. Although work has been done on integrating the natural language constraints into the search itself [8, 9], N-best strategies have remained popular not only because of their ease of implementation, but also because they greatly improve the efficiency of the development effort since one can precompute the N-best lists for a large corpus to use as input for natural language experiments.

In addition, an unanticipated but important application of N-best searches has been in speeding up the development of improved recognition algorithms. One can use N-best resorting experiments as a mechanism for improving recognition systems. For example one can test a new acoustic model by using it to resort N-best lists rather than integrating this new model into the search directly [10, 11, 12]. Resorting N-best lists can require many orders of magnitude less computation than performing the complete search, and may even allow the use of constraints that would not be possible in the complete search (e.g., acoustic models that depend on long-distance contextual factors).

While N-best search strategies have been very useful, they are beginning to encounter problems as we move towards more difficult speech understanding tasks. As both the utterance length and vocabulary size grow, increasingly larger lists of sentence hypotheses are required to capture the necessary amount of ambiguity. This is because sentence hypotheses on the N-best list often differ minimally in highly localized regions where the acoustic signal is not very robust. However, a network representation can capture the same information in a much more compact form. Recently, there has been interest in computing such networks [13, 14].

In this paper, we will describe modifications to our A* N-best search algorithm to allow it to directly compute word networks. Not only does this algorithm produce output in a more compact form, but it is also able to compute a network which is guaranteed to contain the N-best sentence hypotheses using much less computation than the corresponding N-best search.

A* WORD NETWORK SEARCH

A* Search

An A* search is a best-first search with a particular evaluation function defined as:

$$f^*(p) = g(p) + h^*(p),$$

where $f^*(p)$ is the estimated score of the best path containing partial path p, $g(p)$ is the score for the match from the beginning of the utterance to the end of the partial path p, and $h^*(p)$ is an estimate of the best-scoring extension of the partial path p to the end of the utterance [15]. The search proceeds by keeping a stack of partial paths, extending the best-scoring partial path by all possible next words and putting the resulting paths back on the stack. The search terminates when the top-scoring path covers the entire utterance. This search is admissible if $h^*(p)$ is an upper bound on the actual best-scoring extension of partial path p to the end. This can be seen by observing that when a path covers the entire utterance, $h^*(p) = 0$, therefore $f^*(p)$ is no longer an estimate but the true score of the path p. Since all other paths on the stack have upper bounds on their scores which are less than this completed score, the completed path must be the highest-scoring path.

To efficiently apply the A* search to spoken language systems, it is important to have as tight a bound as possible for $h^*(p)$, since the number of path extensions needed to find the best-scoring path decreases as this estimate approaches the actual score for the completion of the partial path. We can use a Viterbi search to compute this upper bound by searching back from the end of the utterance to find the best score to the end for each lexical node at each point in time. If the constraints we use in the Viterbi search to compute the best score to the end are the same as the constraints used in the forward direction of the A* search, then this estimate of the best-scoring completion of the path is exact and the search proceeds very quickly. In this case the search only expands partial paths that are part of the best path (or paths that happen to have exactly the same score as the best path). Another consequence of using a Viterbi search as a first pass is that it computes the score of the best complete path. We can use this score to limit partial path extension to those extensions whose scores fall within a specified threshold of the best score.

A* N-Best Search

We can turn A* into an N-best search simply by modifying the stopping criterion to wait for N completed paths to reach the top of the stack. By the same reasoning that showed the admissibility for A*, we can guarantee that the A* N-best search results in the N top scoring sentence hypotheses. However in this case, the Viterbi estimate is no longer an exact estimate of the future of the partial path, but becomes a looser upper bound as N increases.

A* Word Network Search

If the A* algorithm utilizes local constraints such as a word bigram, then all partial paths that end at a particular time and lexical node will have the same set of extensions. If we keep track of the time/word endpoints of partial paths, then we can merge paths that share common endpoints. This merging creates a *network* instead of a set of word *strings* as the N-best search does.

Our A* word network search consists of the basic A* algorithm described above with the addition a path-merging step. Before we push a partial path onto the stack, we check to see if any previously encountered paths ended at the same time/word endpoint. If there are any, we compare their scores to the current path's score. If the current score is the best so far, we push the current path onto the stack and remove the previous best from the stack, otherwise we do not alter the stack. Either way, for every partial path extension, we add an arc to the growing word network. The basic idea is to have at most one partial path, the best one, in the stack for each time/word endpoint. The search continues until the stack is empty or some arbitrary computation limit (e.g., number of path extensions) is reached. When the search successfully completes with an empty stack, *the word network contains all word arcs that are part of complete paths whose scores are within a specified threshold of the score of the best-scoring complete path.* Note that the pruning that goes into building the network is based on global (complete sentence) path scores. This network is not as compact is it might be because it can contain complete paths with identical word strings differing only in time alignment. If this is undesirable, it should be possible to prune arcs that contribute only to different alignments.

One of the more subtle issues of this word network search involves merging paths and the order of path extension in the A* search. It would be undesirable if a partial path was extended in the search, only to merge with another partial path that has a better score. According to our algorithm, all extensions of the first (lower-scoring) partial path would have to be removed from the stack, wasting the search effort that went into generating them. However, by the same logic that shows admissibility of the A* search, a partial path to a given word/time point cannot both be selected for extension and subsequently be beaten out by another partial path to the same word/time point. Therefore, the order of evaluation in the A* word network search guarantees that no path extensions will be duplicated. Eliminating all duplicate extensions is where the word network search gains its computational efficiency.

EXPERIMENTS

The potential advantages of the word network search include both smaller representation and better computational efficiency. We have explored both of these within the context of recognition of spontaneously produced continuous speech in the ATIS domain [16].

We have compared various measures of computation requirements and representation size as a function of search depth and utterance length. We define the search depth to be the amount of search needed to guarantee that the output contains all sentence hypotheses within some specified score threshold relative to the best-scoring hypothesis. Both the N-best and word network searches can produce all word strings within a specified score threshold from the best score by simply running the search until the stack becomes empty. Note that our word networks contain all time-aligned word strings, even those that differ only in alignment. However, our N-best search only outputs distinct word strings irrespective of alignment differences. Even so, we have found significant efficiency improvements with the word network search.

Data and Recognition System

The corpus used for this evaluation was a subset of DARPA November 1992 ATIS evaluation [17]. To reduce the amount of computation needed, only the utterances from the first session for each speaker were used. We also threw away a few of the longest utterances, because we were not able to compute the N-best to the search depth used in the experiments. This left us with 196 utterances from 29 speakers.

The recognition system used was the SUMMIT system as described in [18]. For these experiments, we used context-independent acoustic models and a bigram language model. (In the SUMMIT system, context-dependent models and higher-order language models are applied after the initial N-best or word network search.) This stripped-down version of the system had a first choice word accuracy of 76.4% on these 196 utterances.

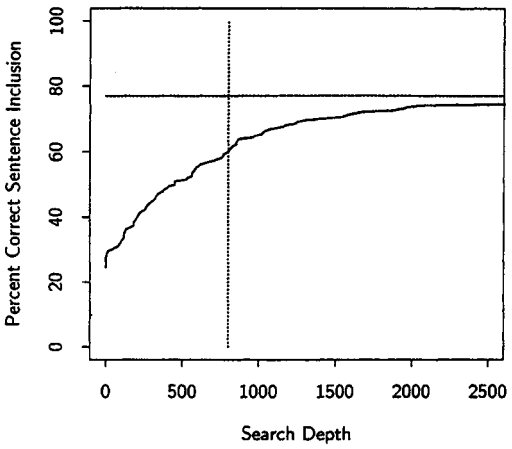

Figure 1: Sentence Accuracy vs. Score Threshold

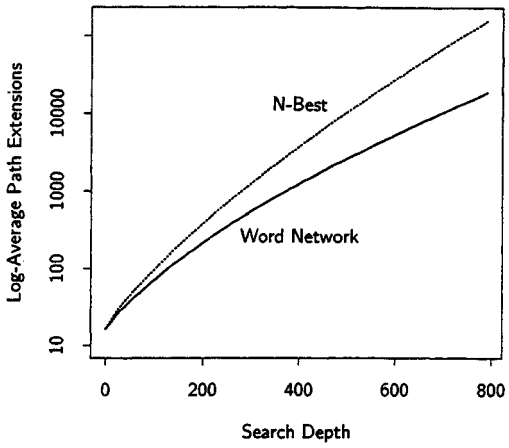

Figure 2: Partial Path Extensions vs. Search Depth

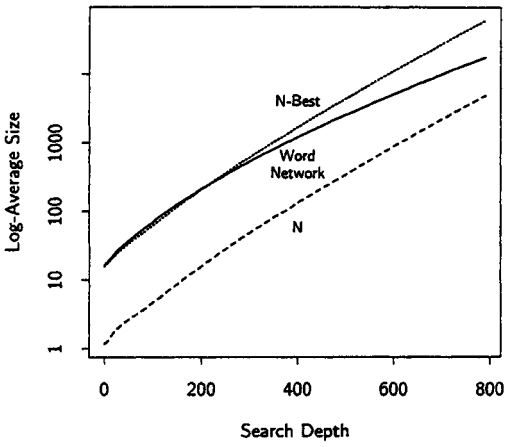

Figure 3: Representation Size vs. Search Depth

For these experiments we have compared various measures of efficiency versus search depth, but the necessary range of search depths depends on the requirements of subsequent processing stages. If the word networks are being used as an initial search for a multi-stage search, then the depth needed in the first stage depends on the relative strengths of the constraints used in the first and later stages of the search. If later stages of the search perform large scoring changes, we need a relatively loose threshold in the initial search.

To get some idea of the range of search depths of interest (at least for this recognition task and this system), Figure 1 shows the percentage of correct sentences contained within a given search depth. Note that this is a spontaneous speech task and contains out-of-vocabulary words. Therefore, the sentence accuracy will not go to 100% no matter how deep we search since we make no attempt to address out-of-vocabulary words. For this experiment, 76.6% of the sentences did not contain out-of-vocabulary words. This ceiling is displayed in the figure as a horizontal line. The vertical line shows the maximum search depth used in the following experiments.

Computational Efficiency

It is difficult to compare in complete detail the computational needs of these two algorithms, since the overall computational efficiency depends on the details of the various parts of the computation (e.g., the implementation of the stack mechanism). Instead, we have focused our attention on the portion of the computation that the two algorithms have in common: extending partial paths. We used the number of partial path extensions as our measure of computation. This is a reasonable measure since this is where both algorithms spend the majority of time.

Figure 2 displays the geometric mean across utterances of the number of path extensions needed to search to a given search depth. We chose the geometric mean since the arithmetic mean would be dominated by the worst-case utterances and there is a very large variation between different utterances. This figure shows that for a given search depth, the word network search requires fewer partial path extensions, and, thus runs faster. Further, as the search depth increases, the difference between the A* search and the word network search increases. This difference is due to the significant amount of path merging in the word network. Note that our N-best search also performs some merging (where partial paths ending at the same point in time have identical word sequences but differ in time alignment).

Efficiency of Representation

We have examined the efficiencies of the N-best and word network representations versus search depth. For N-best, a reasonable measure for complexity of representation is the number of pops from the stack since every partial path popped from the stack corresponds to a word in the N-best list (subject to word-string pruning). For word networks, the number of arcs in the network is the number of partial path extensions. Figure 3 displays the relative sizes of the two representations as the search depth increases. We've also plotted N, the number of unique sentence hypotheses. This comparison is somewhat unfair because the word networks are the raw networks produced by the search and contain *all* alignments, whereas the N-best do not. Depending on the intended use of the word networks, it might be desirable to apply network reduction algorithms to prune them. This would be especially useful if we were concerned only with word strings (as contained in the N-best output) rather than all possible alignments of these strings. Even without such pruning, the word networks have smaller representations.

Dependence on Utterance length

We have noticed a very large utterance-to-utterance variation in the size of the search for a given search depth (e.g., the number of path extensions ranges from 562 to well over 8,000,000 for the maximum depth of 800 in the plots). While

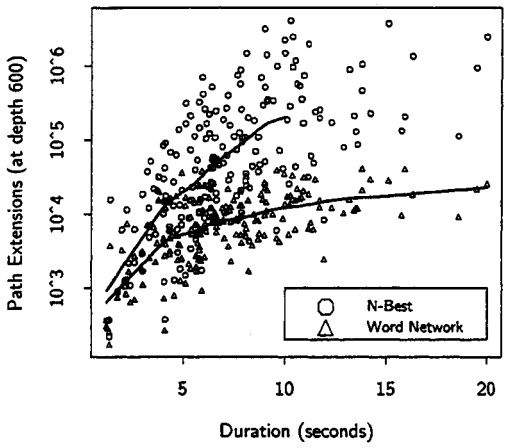

Figure 4: Partial Path Extensions vs. Utterance Duration

some of this variation is certainly due to the strength of the acoustic evidence, there is also a strong dependence on utterance length. Figure 4 shows the scatter plot of the number of partial path extensions versus utterance duration with the search depth fixed at 600. For both searches, we've overlayed lines produced by a scatter plot smoother (*lowess* in S). (The line for N-best stops at 10 seconds because some longer sentences required too much computation to reach the search depth and aren't included in the plot. However, we had no difficulty computing the corresponding word networks.) As duration increases, the required search effort increases for both algorithms, but the increase is much more substantial for the N-best search. The word network search is better behaved, requiring about two orders of magnitude fewer expansions in the worst cases.

DISCUSSION

As we have shown, a straightforward change to the A* N-best search that we had used previously results in an A* word network search that has significant computational and space advantages. On average, we found a factor of 9 reduction in the number of path extensions needed for a search depth of 800. The savings for deeper searches are even greater.

We have found the word networks to be a convenient intermediate representation for later stages of processing. We can efficiently apply additional constraints in a second stage search by using another A* search. In this case, the word network constrains the search by providing a list of next-word extensions for any partial path. By saving the acoustic and language model scores, as well as $g(p)$ and $h^*(p)$, we can avoid the recomputation of these scores during the second stage of the search. Since this second stage A* search is very similar to the original search, it can also produce single hypotheses, N-best lists, or even new word networks to be used by yet another stage of processing.

Due to the computational advantages of the word-network over the N-best representation we have incorporated the networks into our SUMMIT speech recognition system. In our experiments on the ATIS domain, we have searched word networks using a class 4-gram grammar to reduce the word error on the February 1992 test set by 17% [18]. In the future we plan to incorporate more of our expensive knowledge sources into word network searches.

REFERENCES

[1] A. Viterbi, "Error bounds for convolutional codes and an asymptotic optimal decoding algorithm," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 260–269, Apr. 1967.

[2] V. W. Zue, "Development of spoken language systems." To be published in *IEEE Expert*, 1993.

[3] Y. Chow and R. Schwartz, "The N-best algorithm," in *Proc. DARPA Speech and Nat. Lang. Workshop* (Harwichport), pp. 199–202, Oct. 1989.

[4] V. Zue, J. Glass, D. Goodine, H. Leung, M. Phillips, J. Polifroni, and S. Seneff, "Integration of speech recognition and natural language processing in the MIT VOYAGER system," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Toronto), pp. 713–716, May 1991.

[5] R. Schwartz and S. Austin, "A comparison of several approximate algorithms for finding multiple (N-best) sentence hypotheses," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Toronto), pp. 701–704, May 1991.

[6] F. K. Soong and E.-F. Huang, "A tree-trellis based fast search for finding the N best sentence hypotheses in continuous speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Toronto), pp. 705–708, May 1991.

[7] P. Kenny, R. Hollan, V. Gupta, M. Lennig, P. Mermelstein, and D. O'Shaughnessy, "A*-admissible heuristics for rapid lexical access," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Toronto), pp. 689–692, May 1991.

[8] D. Goodine, S. Seneff, L. Hirschman, and M. Phillips, "Full integration of speech and language understanding in the MIT spoken language system," in *Proc. European Conf. Speech Comm. and Tech.* (Genoa), pp. 845–848, Sept. 1991.

[9] H. Murveit and R. Moore, "Integrating natural language constraints into HMM-based speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Albuquerque), vol. 1, pp. 573–576, Apr. 1990.

[10] M. Phillips, J. Glass, and V. Zue, "Modelling context dependency in acoustic-phonetic and lexical representations," in *Proc. DARPA Speech and Nat. Lang. Workshop* (Pacific Grove), pp. 71–76, Feb. 1991.

[11] M. Ostendorf, A. Kannan, S. Austin, O. Kimball, R. Schwartz, and J. R. Rohlicek, "Integration of diverse recognition methodologies through reevaluation of N-best sentence hypotheses," in *Proc. DARPA Speech and Nat. Lang. Workshop* (Pacific Grove), pp. 83–87, Feb. 1991.

[12] R. Schwartz, S. Austin, F. Kubala, J. Makhoul, L. Nguyen, P. Placeway, and G. Zavaliagkos, "New uses for the N-best sentence hypotheses within the BYBLOS speech recognition system," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (San Francisco), vol. 1, pp. 1–4, Mar. 1992.

[13] H. Murveit, J. Butzberger, V. Digalakis, and M. Weintraub, "Large-vocabulary dictation using SRI's DECIPHER speech recognition system: Progressive search techniques," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Minneapolis), vol. 2, pp. 319–322, Apr. 1993.

[14] M. Oerder and H. Ney, "Word graphs: An efficient interface between continuous-speech recognition and language understanding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Minneapolis), vol. 2, pp. 119–122, Apr. 1993.

[15] A. Barr, E. Feigenbaum, and P. Cohen, *The Handbook of Artificial Intelligence*. Los Altos, CA: William Kaufman, 1981.

[16] L. Hirschman, M. Bates, D. Dahl, W. Fisher, J. Garofolo, K. Hunicke-Smith, D. Pallett, C. Pao, P. Price, and A. Rudnicky, "Multi-site data collection for a spoken language corpus," in *Proc. Int. Conf. Spoken Language Processing* (Banff), pp. 903–906, Oct. 1992.

[17] D. S. Pallett, J. G. Fiscus, W. M. Fisher, and J. S. Garofolo, "Benchmark tests for the DARPA spoken language program," in *Proc. ARPA Human Lang. Tech. Workshop* (Plainsboro), Mar. 1993.

[18] J. Glass, D. Goddeau, D. Goodine, L. Hetherington, L. Hirschman, M. Phillips, J. Polifroni, C. Pao, S. Seneff, and V. Zue, "The MIT ATIS system: January 1993 progress report," in *Proc. DARPA Spoken Lang. Systems Tech. Workshop* (Cambridge), Jan. 1993.