The N-Best Algorithm: An Efficient Procedure for Finding Top N Sentence Hypotheses

Yen-Lu Chow and Richard Schwartz

BBN Systems and Technologies Corporation
Cambridge, MA 02138

ABSTRACT

In this paper we introduce a new search algorithm that provides a simple, clean, and efficient interface between the speech and natural language components of a spoken language system. The N-Best algorithm is a time-synchronous Viterbi-style beam search algorithm that can be made to find the most likely N whole sentence alternatives that are within a given a "beam" of the most likely sentence. The algorithm can be shown to be *exact* under some reasonable constraints. That is, it guarantees that the answers it finds are, in fact, the *most likely* sentence hypotheses. The computation is linear with the length of the utterance, and faster than linear in N. When used together with a first-order statistical grammar, the correct sentence is usually within the first few sentence choices. The output of the algorithm, which is an ordered set of sentence hypotheses with acoustic and language model scores can easily be processed by natural language knowledge sources. Thus, this method of integrating speech recognition and natural language avoids the huge expansion of the search space that would be needed to include all possible knowledge sources in a top-down search. The algorithm has also been used to generate alternative sentence hypotheses for discriminative training. Finally, the alternative sentences generated are useful for testing overgeneration of syntax and semantics.

1 Introduction

In a spoken language system (SLS) we have a large search problem. We must find the most likely word sequence consistent with all knowledges sources (speech, statistical N-gram, natural language). The natural language (NL) knowledge sources are many and varied, and might include syntax, semantics, discourse, pragmatics, and prosodics. One way to use all of these constraints is to perform a top-down search that, at each point, uses all of the knowledge sources (KSs) to determine which words can come next, and with what probabilities. Assuming an exhaustive search in this space, we can find the most likely sentence. However, since many of these KSs contain "long-distance" effects (for example, agreement between words that are far apart in the input), the search space can be quite large, even when pruned using various beam search or best-first search techniques. Furthermore, a top-down search strategy requires that all of the KSs be formulated in a predictive, left-to-right manner. This may place an unnecessary restriction on the type of knowledge that can be used.

The general solution that we have adopted is to apply the KSs in the proper order to constrain the search progressively. Thus, we trade off the entropy reduction that a KS provides against the cost of applying that KS. Naturally, we can also use a pruning strategy to reduce the search space further. By ordering the various KSs, we attempt to minimize the computational costs and complexity for a given level of search error rate. To do this we apply the most powerful and cheapest KSs first to generate the top N hypotheses. Then, these hypotheses are evaluated using the remaining KSs. In the remainder of this paper we present the N-best search paradigm, followed by the N-best search algorithm. Finally, we present statistics of the rank of the correct sentence in a list of the top N sentences using acoustic-phonetic models and a statistical language model.

2 The N-best Paradigm

Figure 1 illustrates the general N-best search paradigm. We order the various KSs in terms of their relative power and cost. Those that provide more constraint, at a lesser cost, are used first in the N-best search. The output of this search is a list of the most likely whole sentence hypotheses, along with their scores. These hypotheses

are then rescored (or filtered) by the remaining KSs.

Depending on the amount of computation required, we might include more or less KSs in the first N-best search. For example, it is quite inexpensive to search using a first-order statistical language model, since we need only one instance of each word in our network. Frequently, a syntactic model of NL will be quite large, so it might be reserved until after the list generation. Given the list, each alternative can usually be considered in turn in a fraction of a second. If the syntax is small enough, it can be included in the initial N-best search, to further reduce the list that would be presented to the remainder of the KSs. Another example of this progressive filtering could be the use of high-order statistical language models. While the high-order model frequently provides added power (over a first-order model), the added power is usually not commensurate with the large amount of extra computation and storage needed for the search. In this case, a first-order language model can be used to reduce the choice to a small number of alternatives which can then be reordered using the higher-order model.

Besides the obvious computational advantages, there are several other practical advantages of this paradigm. Since the output of the first stage is a small amount of text, and there is no further processing required from the acoustic recognition component, the interface between the speech recognition and the other KSs is trivially simple, while still optimal. As such this paradigm provides a most convenient mechanism for integrating work among several research sites. In addition, the high degree of modularity means that the different component subsystems can be optimized and even implemented separately (both hardware and software). For example, the speech recognition might run on a special-purpose array processor-like machine, while the NL might run on a general purpose host.

3 The N-Best Algorithm

The optimal N-Best algorithm is quite similar to the time-synchronous Viterbi decoder that is used quite commonly, with a few small changes. It must compute probabilities of word-sequences rather than state-sequences, and it must find *all* such sequences within the specified beam.

At each state:

- Keep separate records for theories with different word sequence histories.
- Add probabilities for the same theory.
- Keep up to a specified maximum number N of theories whose probabilities are within a threshold of the probability of most likely word sequence at that state. Note that this state-dependent-threshold is distinct from the global beam search threshold.

This algorithm requires (at least) N times the memory for each state of the hidden Markov model. However, this memory is typically much smaller than the amount of memory needed to represent all the different acoustic models. We assume here, that the overall "beam" of the search is much larger than the "beam at each state" to avoid pruning errors. In fact, for the first-order grammar, it is even reasonable to have an infinite beam, since the number of states is determined only by the vocabulary size.

At first glance, one might expect that the cost of combining several sets of N theories into one set of N theories at a state might require computation on the order of N^2. However, we have devised a "grow and prune" strategy that avoids this problem. At each state, we simply gather all of the incoming theories. At any instant, we know the best scoring theory coming to this state at this time. From this, we compute a pruning threshold for the state. This is used to discard any theories that are below the threshold. At the end of the frame (or if the number of theories gets too large), we reduce the number of theories using a *prune and count* strategy that requires no sorting. Since this computation is small, we find, empirically that the overall computation increases with $\sqrt{N}$, or slower than linear. This makes it practical to use somewhat high values of N in the search.

4 Rank of the Correct Answer

Whether the N-best search is practical depends directly on whether we can assure that the correct answer is reliably within the list that is created by the first stage. (Actually, it is sufficient if there is any answer that will be accepted by all the NL KSs, since no amount of search would make the system choose the lower scoring correct answer in this case.) It is possible that when the correct answer is not the top choice, it might be quite far down the list, since there could be exponentially many other alternatives that score between the highest scoring answer and the correct answer. Whether this is true depends on the power of the acoustic-phonetic models and the statistical language model used in the N-best

search. Therefore we have accumulated statistics of the rank of the correct sentence in the list of N answers for two different language models: the statistical class grammar(perplexity 100), and no grammar(perplexity 1000).

Figure 2 plots the cumulative distribution of the rank for the two different language models. The distribution is plotted for lists up to 100 long. We have also marked the average rank on the distribution. As can be seen, for the case of no language model, the average rank is higher than that for the statistical grammar. In fact, about 20% of the time, the correct answer is not on the list at all. However, when we use the statistical class grammar, which is a fairly weak grammar for this domain, we find that the average rank is 1.8, since most of the time, the correct answer is within the first few choices. In fact, for this test of 215 sentences, 99 percent of the sentences were found within the 24 top choices. Furthermore, the acoustic model used in this experiment is an earlier version that results in twice the word error rate of the most recent models reported elsewhere in these proceedings. This means that when redone with better acoustic models, the rank will be considerably lower.

To illustrate the types of lists that we see we show below a sample N-best output. In this example, the correct answer is the fifth one on the list.

Example of N-best Output
Answer:

Set chart switch resolution to high.

Top N Choices:
Set charts which resolution to five.
Set charts which resolution to high.
Set charts which resolution to on.
Set chart switch resolution to five.
Set chart switch resolution to high. (***)
Set chart switch resolution to on.
Set charts which resolution to the high.
Set the charts which resolution to five.

5 Other Applications for N-Best Algorithm

We have, so far, found two additional application for the N-Best algorithm. The first is to generate alternative hypotheses for discriminative training algorithms. Typically, alternatives must be generated using a fast match procedure, or by using overall statistics of typical errors. Instead, we can generate all the actual alternatives that are appropriate to each particular sentence.

A second application for the N-best algorithm is to generate alternative sentences that can be used to test overgeneration in the design of natural language systems. Typically, if overgeneration is tested at all, it is by generating random sentences using the NL model, and seeing whether they make sense. One problem with this is that many of the word sequences generated this way would never, in fact, be presented to a NL system by any reasonable acoustic recognition component. Thus, much of the tuning is being done on unimportant problems. A second problem is that the work of examining the generated sentences is a very tedious manual process. If, instead, we generate N-best lists from a real acoustic recognition system, then we can ask the NL system to parse all the sentences that are known to be wrong. Hopefully the NL system will reject most of these, and we only need to look at those that were accepted, to see whether they should have been.

6 Conclusion

We have presented a new algorithm for computing the top N sentence hypotheses for a hidden Markov model. Unlike previous algorithms, this one is guaranteed to find the most likely scoring hypotheses with essentially constant computation time. This new algorithm makes possible a simple and efficient approach to integration of several knowledge sources, in particular the integration of arbitrary natural language knowledge sources in spoken language systems. In addition there are other useful applications of the algorithm.

Acknowledgement

This work was supported by the Defense Advanced Research Projects Agency and monitored by the Office of Naval Research under Contract No. N00014-85-C-0279.

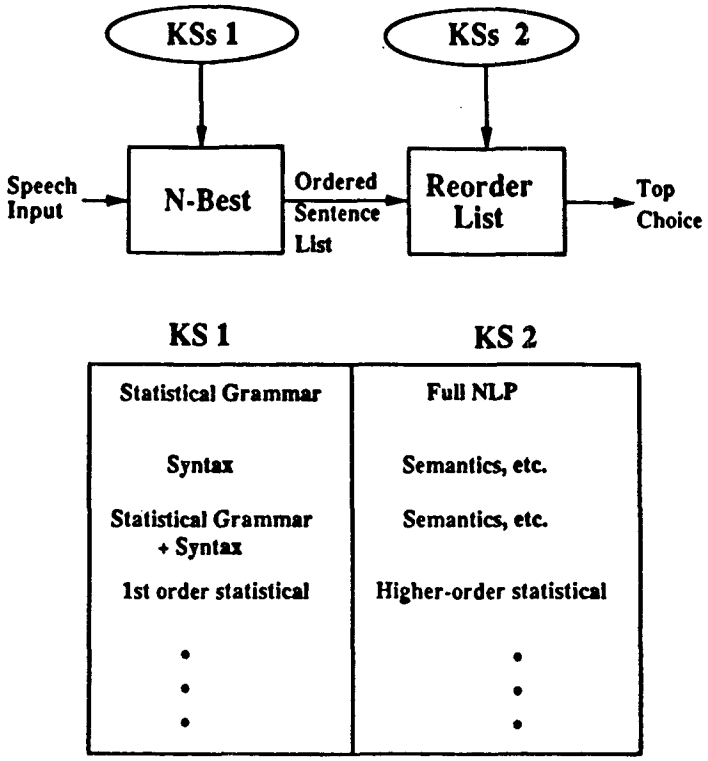

Figure 1: N-best Search Paradigm

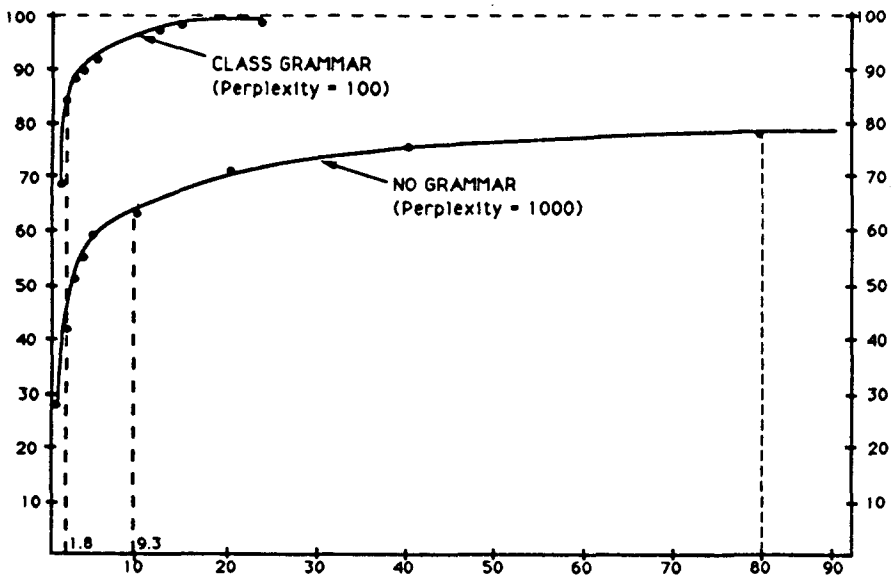

Figure 2: Cumulative Distribution of Rank of Correct Answer