Session 13: CSR Search

Richard Schwartz

BBN Systems and Technologies
Cambridge, MA 02138

ABSTRACT

This session had five papers related to different topics in CSR Search. The topics ranged from integration of many knowledge sources within a practical system, to different search algorithms for real-time large vocabulary speech recognition.

1. Papers

This section contains one or two paragraphs about each of the papers presented. More details can be found in the papers that follow.

1. Sadaoki Furui from NTT presented a paper describing the integration of several knowledge sources for a system that performed very large vocabulary recognition of names and address for directory assistance. Some experiments were performed with directory listings for 70,000 customers. Knowledge sources at the phonological, lexical, and grammatical level were used to make the search feasible. A smaller scale system with 2,300 subscribers was used in trials in order to reduce computation and error rate.

2. Hy Murveit from SRI presented some new algorithms for recognition of continuous speech using continuous densities. The first algorithm used a tree structure for the unigram back-off part of the language model (which usually accounts for most of the computation), and the usual bigram structure for those few bigrams observed in the training. The second class of algorithms covered techniques for reducing the observation computation for continuous densities. Then, he presented a wide range of extensive experiments trading off different approaches for reducing computation and size. The result was that the recognition could run in about three times real time with accuracy only a little bit worse than that for the best research conditions. In addition, it could run in real time with about three times the error rate.

3. Doug Paul from MIT Lincoln Laboratory presented some improvements in the Stack Decoder search. The improvements were made in the fast match algorithms and in the implementation of the search components. For example, caching algorithms to reduce look-up costs, and quantization algorithms were employed to reduce size requirements. In addition, techniques for tree-clustering of different allophones of a phoneme, and techniques for incremental speaker adaptation were presented.

4. Julian Odell from Cambridge University presented a search algorithm in which all of the constraints were used in a single time-synchronous pass over the data. Thus, all knowledge sources, including trigram language models and between-word coarticulation models were compiled dynamically into a tree. While the search was somewhat expensive, it was quite interesting that it was possible at all, since the size of the search space would be tremendous if fully expanded. This work showed that using the available knowledge as early as possible greatly reduces the computation.

5. Long Nguyen from BBN described experiments aimed at reducing the perceived search errors that might result from using the N-best Search strategy. The search algorithm, which was related to Progressive Search technique proposed by Murveit, built a lattice to cover a wide range of choices. Then, this lattice of alternatives was decoded again using trigram and between-word triphone models. The result, however, showed that there were very few search errors caused by the original N-best algorithm. However, the new lattice search algorithm was faster than rescoring the n-best alternatives. Finally, the other essential uses for the n-best paradigm were reviewed.

2. Conclusions

Some general conclusions can be made from the various attempts at improving the efficiency and accuracy of the search algorithms.

First, while there are various tradeoffs that can be made related to pruning back the number of active hypotheses, or the size of the language model, etc, in general, these compromises quickly become damaging, in that they also increase the word error rate.

The more effective approaches make use of two general techniques: shared computation, and multiple-pass strategies.

2.1. Shared Computation

Two effective ways to share computation are to use tree structures, and to perform bottom-up processing.

Tree structures, both at the phonetic and language modeling

level reduce computation by a large factor since the computation for the initial portions of similar words can be shared. By the time the computation gets to the ends of the words, most of the words have been eliminated.

Bottom-up processing means that a system examines the input without regard to the surrounding context, and uses these scores in various combinations depending on the global context. Thus, the repeated scoring of the same acoustic events in different language model contexts is avoided.

2.2. Multiple-Pass Strategies

There are several multi-pass search strategies that have found beneficial use when real-time is desired. The problem is that, even though it would be nice to use all of the knowledge sources at once to obtain their full integration, this is just too expensive for the size of problems we are trying to handle, and the currently available hardware. The single-pass search employed by Cambridge University certainly showed that there is much to be gained from efficient sharing, primarily through the use of dynamically compiled tree structures. However, at the current time, it seems unlikely that this approach could be pushed all the way to real-time processing.

The multiple-pass strategies discussed here include fast match algorithms, using vector quantization as an approximation to eliminate most of the computation for Gaussians, use of N-best searches with reduced models followed by rescoring with more detailed models, and use of lattices in much the same way. In addition, the use of the forward-backward search technique allows the later passes to make more effective use of the pruning information derived from earlier passes.

The multiple-pass search strategies often can save several orders of magnitude in search computation, thus making real-time conceivable.