A Word Graph Based N-Best Search in Continuous Speech Recognition

Bach-Hiep Tran, Frank Seide, Volker Steinbiss

Philips GmbH Forschungslaboratorien Aachen
Weißhausstraße 2, D-52066 Aachen, Germany
E-mail: tran@pfa.research.philips.com

ABSTRACT

In this paper, we introduce an efficient algorithm for the exhaustive search of N best sentence hypotheses in a word graph. The search procedure is based on a two-pass algorithm. In the first pass, a word graph is constructed with standard time-synchronous beam search. The actual extraction of N best word sequences from the word graph takes place during the second pass.
With our implementation of a tree-organized N-Best list, the search is performed directly on the resulting word graph. Therefore, the parallel bookkeeping of N hypotheses at each processing step during the search is not necessary. It is important to point out that the proposed N-Best search algorithm produces an exact N-Best list as defined by the word graph structure. Possible errors can only result from pruning during the construction of the word graph.
In a postprocessing step, the N candidates can be rescored with a more complex language model with highly reduced computational cost. This algorithm is also applied in speech understanding to select the most likely sentence hypothesis that satisfies some additional constraints.

1. Introduction

Several previous algorithms on search strategy for finding N best sentence hypotheses can be found in [2], [10], [9], [8], [3].
Recently, the two-pass word graph algorithm [7], [6] has been sucessfully applied in continuous speech recognition for large vocabulary. By decoupling the acoustic search from the application of language knowledge sources, it is possible to subsequently use the constructed word graph at phrase level with high efficiency.
The purpose of this paper is to propose an algorithm, which is based on the resulting word graph, for finding top N sentence candidates with a simple language model. Subsequently the top N word sequences obtained can be rescored using a more complex language model.
Another application of this algorithm is found in speech understanding. In dialogue systems, N is not determined in advance. The alternate sentence hypotheses are generated automatically and are checked one after the other whether they satisfy certain consistency constraints according to data base entries. The first hypothesis satisfying the constraints is chosen.

2. Basic Principle

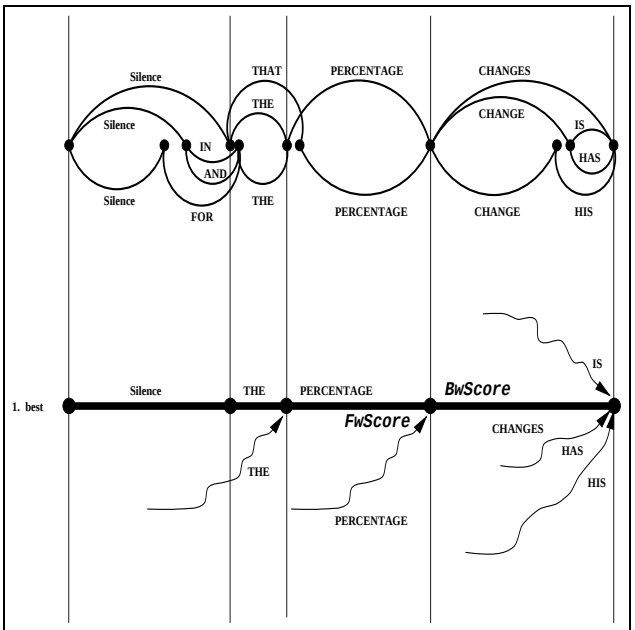

Figure 1: Possible candidates for the second best path.

The principle of the approach presented is based on the following consideration:
When several paths lead to the same node in the word graph, according to the Viterbi criterion, only the best scored path is expanded. The remaining paths, by reason of this recombination, are not considered any further.
Assuming that the first best sentence hypothesis was found by Viterbi decoding through a given word graph, the second best path is the path which competed with the best one but was recombined at some node of the best path. Thus, in order to find the second best sentence hypothesis, we have to consider all possible partial paths in the word graph which

reach some node (including the terminal node) of the best path and might share the remaining section with the best path.
By applying this procedure repeatedly, N best sentence hypotheses can be successively extracted from the given word graph. Figure 1 illustrates this principle.

3. N-Best Search Algorithm

The search procedure consists of two consecutive levels. First, a word graph is generated with standard time-synchronous beam search ([4], [5], [11]).
Second, the N-Best algorithm is applied at the phrase level. In the first pass, we calculate (see [6], [7]), for each node in the word graph, the best way of arriving at that node. This yields cumulative scores[1] and backpointers for each word hypothesis which are stored for later processing. The second pass uses the output of this first step to find the N best sentence hypotheses sequentially.
The best path can be determined simply by comparing the cumulative scores of all possible paths leading to the terminal node of the word graph.
In order to ensure that this best word sequence is not taken into account while searching for the second best path, the complete best path is copied into a so-called N-Best tree. Using this structure, a backward cumulative score for each word copy is computed and stored at the corresponding tree node. This allows for fast and efficient computation of the complete path scores required to determine the next best sentence hypotheses.
The second best sentence hypothesis can be found by taking the path with the best score among the candidate paths which might share a remaining section of the best path. The partial path of this sentence hypothesis is then copied into the N-Best tree.
Based on these two separate structures, it is guaranteed that no sentence hypothesis is considered twice, and complete path scores are simply computed by combining the cumulative forward scores in the word graph with the corresponding cumulative backward scores in the N-Best tree.
Assuming that N best paths have been found, the (N+1)-th best path can be determined by examining all existing nodes in the N-Best tree, because it can share the last part of some path among the top N paths. Thus, this algorithm performs a full search within the word graph and delivers exact results as defined by the word graph structure.
Figure 2 shows a word graph and the expanded N-Best tree for four best sentence hypotheses.
The complete algorithm of the N-Best search procedure with bigram is shown in Table 1. This algorithm can be extended to M-gram with $M \geq 3$.

The variables used in Table 1 have the following meaning:

Arc^i: outgoing arc at node i in the N-Best tree.
$Arc_{i,k}$: incoming arcs at node i in the N-Best tree.
$Edge_{j,l}$: incoming edges at node j in the word graph.
$FwSco(Edge_{j,l})$: cumulative forward score of the best partial path leading to the Edge $Edge_{j,l}$ in the word graph.
$BwSco(Node_i)$: cumulative backward score from sentence ending to node i in the N-Best tree.
$LMFactor$: scaling factor for bigram score.

PREPROCESSING STEP:

- rescore the word graph with bigram in forward direction using the Viterbi algorithm at phrase level
- store cumulative path score $FwSco$ and backpointer for each edge

INITIALIZE:

- create root for N-Best tree
- assign root to the terminal node in the word graph

For n=1 to N do

For each node i of N-Best tree do

Consider the corresponding node j in the word graph

For each edge $Edge_{j,l}$ at node j in the word graph do

IF $Edge_{j,l} \neq Arc_{i,k}$ for all k THEN

- compute complete path score
$$Score_{j,l} = FwSco(Edge_{j,l}) + BwSco(Node_i) - LMFactor * \ln(P(Arc^i|Edge_{j,l}))$$
- remember the edge with the best score
$$j_{Opt}, l_{Opt} = \underset{j,l}{\operatorname{argmin}}[Score_{j,l}]$$

END IF

EXPAND N-BEST TREE:

- trace back from $Edge_{j_{Opt},l_{Opt}}$ to sentence start in the word graph
- copy this partial path to the N-Best tree ($Arc_{i,k}$) to get complete path
- compute backward cumulative score $BwSco(Node_i)$ for each newly created node
- output word sequence

Table 1: N-Best Search Algorithm.

[1] scores: negative log probabilities

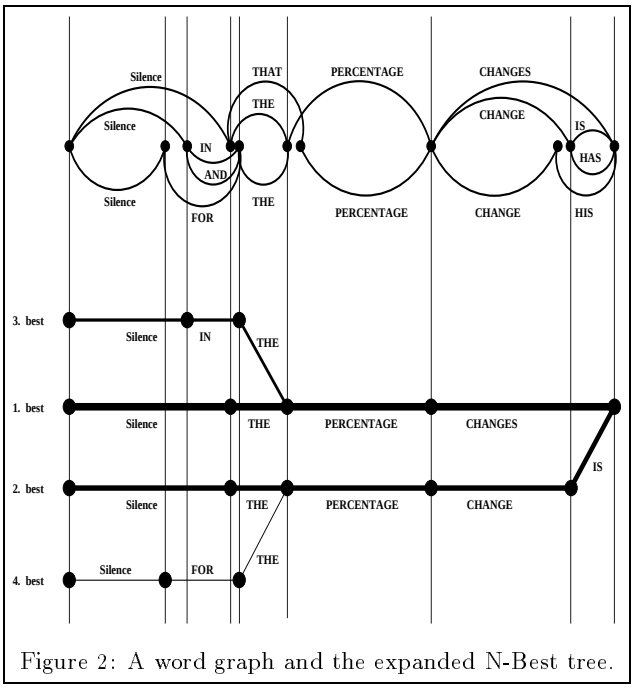

Figure 2: A word graph and the expanded N-Best tree.

4. Complexity

As explained in the preceding section, the tree-organized N-Best list (N-Best tree) must be updated, after the N-th best path was found to avoid extracting the same sentence hypothesis twice. Thus, the search effort depends on the current size of the N-Best list. Assuming a sentence with M words and a very large N, the expected cost of computation is:

$$\sum_{n=1}^{N} \frac{M}{2}(n+1) = \frac{M}{4}N(N+3) \approx O(N^2)$$

To give an example, we need approximately 0.3 seconds to find the first 10 alternative word sequences for a spoken sentence with 23 words (DEC station 5000-240/30 MIPS).

5. Experimental Results

The algorithm has been evaluated on the WSJ test set of Nov. 92 (5k closed vocabulary) for four male speakers using bigram and trigram language models with perplexities of 111 and 57 respectively. The N-Best error rate is calculated by choosing the sentence with minimal number of errors among the top N sentence hypotheses. The error rate for N=∞ is interpreted as the lower limit (i.e. the minimum word error rate achievable when exploiting the word graph) and is calculated by searching the sentence hypothesis in the word graph which best matches the spoken sentence. In order to avoid search errors, the word graphs have been constructed with a relatively high graph-density, namely 600 (the graph-density is defined as average number of word hypotheses per spoken word).

It is important to realize that in general the N-Best error rate depends on the sentence length and is the optimistic estimation for potential improvement. In our test set the average sentence length amounts to 16. The results given in Tables 2 and 3 show that with the top 10 sentence candidates, the error rate is decreased by approximately 50%. By considering the top 50 choices, the error rate is reduced to one third of the first best word error rate.

N	Spk. 1	Spk. 2	Spk. 3	Spk. 4	Aver.
1	7.07%	4.71%	5.53%	7.76%	6.27%
10	4.15%	2.74%	2.47%	3.04%	3.10%
20	3.69%	1.98%	1.46%	2.44%	2.40%
50	2.92%	1.82%	0.87%	1.83%	1.86%
100	2.15%	1.82%	0.87%	1.37%	1.55%
200	1.84%	1.52%	0.87%	1.22%	1.36%
∞	0.46%	0.61%	0.44%	0.15%	0.42%

Table 2: N-Best error rate using bigram (perplexity=111).

N	Spk. 1	Spk. 2	Spk. 3	Spk. 4	Aver.
1	5.68%	3.04%	3.06%	5.48%	4.32%
10	3.84%	1.52%	1.16%	2.13%	2.07%
20	3.07%	1.37%	1.16%	1.98%	1.90%
50	2.30%	1.22%	0.87%	1.52%	1.48%
100	2.15%	1.22%	0.87%	1.37%	1.40%
200	1.84%	1.22%	0.73%	0.91%	1.18%
∞	0.46%	0.61%	0.44%	0.15%	0.42%

Table 3: N-Best error rate using trigram (perplexity=57).

6. Application of N-Best

We present below results of applying the N-Best search algorithm to improve speech understanding and to evaluate subsequently long-span language models with reduced computational cost.

6.1. Speech Understanding

This approach has been successfully applied in our automatic directory assistance demonstrator (for detailed description see [1]). In the course of a dialogue, the alternate sentence hypotheses for a spoken sentence are generated one after the other. By incorporating data base constraints, these hypotheses can be checked one after the other whether they satisfy certain consistency constraints with regard to this data base. The first sentence hypothesis satisfying the constraints is chosen. In comparison with first best, this technique reduced the word error rate from 28.9% to 24.4%.

6.2. Subsequent Employment of More Complex Language Models

Using the N best sentence hypotheses generated with bigram and trigram, we can evaluate a long-span language model, e. g. a quadrogram, by subsequently rescoring alternate hypotheses with this language model. The sentence hypothesis with the best score (acoustic score and new language model score) is chosen. Tables 4 and 5 present the results from rescoring with a quadrogram (perplexity=38).

N	Spk. 1	Spk. 2	Spk. 3	Spk. 4	Aver.
1	7.07%	4.71%	5.53%	7.76%	6.27%
10	5.83%	3.34%	3.20%	4.57%	4.24%
20	5.99%	3.19%	3.06%	4.41%	4.16%
50	5.53%	3.19%	2.77%	4.41%	3.98%
100	4.76%	3.19%	2.77%	4.57%	3.82%
200	4.61%	3.19%	2.77%	4.57%	3.79%

Table 4: Error rate of sentences with best scores obtained by rescoring N-Best hypotheses with quadrogram using N-Best list generated with bigram.

N	Spk. 1	Spk. 2	Spk. 3	Spk. 4	Aver.
1	5.68%	3.04%	3.06%	5.48%	4.32%
10	5.99%	2.89%	2.77%	4.26%	3.98%
20	5.53%	2.58%	2.77%	4.41%	3.82%
50	5.38%	2.58%	2.77%	4.57%	3.82%
100	5.83%	2.58%	2.77%	4.57%	3.94%
200	5.38%	2.58%	2.77%	4.57%	3.82%

Table 5: Error rate of sentences with best scores obtained by rescoring N-Best hypotheses with quadrogram using N-Best list generated with trigram.

As shown in Tables 4 and 5, the quadrogram language model applied is able to compensate some uncertainty introduced by parameter estimation in the acoustic models. A one pass search using a quadrogram language model is estimated to reach about the same recognition performance (3.8%). However, in comparison with the error rate given in Tables 2 and 3, a significant potential improvement for even more complex language models is still likely. Please also note that rescoring sentence hypotheses generated by bigram using a quadrogram yields about the same results as rescoring sentence hypotheses generated by trigram. In the latter case saturation is observed at lower values of N. In either case, most of the recognition errors which can be removed using the quadrogram language model are among the top 10 best sentence hypotheses.

7. Conclusion

In this paper we have presented a straight-forward algorithm to successively extract N best sentence hypotheses from a given word graph. By splitting the acoustic search into two passes, it is inexpensive to create the top N sentence hypotheses even for medium-sized language models, e.g trigram because the search is performed at the phrase level. Furthermore, the proposed algorithm delivers exact results as defined by the underlying word graph. This algorithm is successfully applied in evaluation of WSJ test set, as well as in our telephone inquiry system.

8. REFERENCES

1. Kellner, A., Rueber, B., and Seide, F. "Improving Speech Understanding by Incorporating Database Constraints and Dialogue History". *Elsewhere in these Proc. ICSLP*, 1996.
2. Lee, C.-H. and Rabiner, L. R. "A Network-Based Frame-Synchronous Level Building Algorithm For Connected Word Recognition". In *Proc. ICASSP-88*, pages 410–413, New York, 1988.
3. Soong, F. and Huang, E.-F. "A Tree-Trellis Based Fast Search for Finding the N-Best Sentence Hypotheses in Continuous Speech Recognition". In *Proc. ICASSP-91*, pages 705–708, Toronto, Canada, 1991.
4. Ney, H., Mergel, D., Noll, A., and Paeseler, A. "Data-driven search organization for continuous speech recognition". *IEEE Transactions on Signal Processing*, 40(2):272–281, 1992.
5. Ney, H., Haeb-Umbach, R., Tran, B.-H., and Oerder, M. "Improvements in beam search for 10000-word continuous speech recognition". In *Proc. ICASSP-92*, pages 9–12, San Francisco, 1992.
6. Ney, H. and Aubert, X. "A Word Graph Algorithm for Large Vocabulary Continuous Speech Recognition". In *Proc. ICSLP*, pages 1355–1358, Yokohama, Japan, 1994.
7. Oerder, M. and Ney, H. "Word graphs: An efficient interface between continuous-speech recognition and language understanding". In *Proc. ICASSP-93*, pages 119–122, Minneapolis, 1993.
8. Schwartz, R. and Austin, S. "A Comparison of Several Approximate Algorithms For Finding Multiple (NBEST) Sentence Hypotheses". In *Proc. ICASSP-91*, pages 701–704, Toronto, Canada, 1991.
9. Schwartz, R. and Chow, Y.-L. "The N-Best Algorithm: An Efficient and Exact Procedure for Finding the Most Likely Sentence Hypotheses". In *Proc. ICASSP-90*, pages 81–84, Albuquerque, NM, 1990.
10. Steinbiss, V. "Sentence-Hypotheses Generation in a Continuous-Speech Recognition System". In *Proc. Eurospeech-89*, pages 51–54, Paris, 1989.
11. Steinbiss, V. and Tran, B.-H. "Improvements in Beam Search". In *Proc. ICSLP*, pages 2143–2146, Yokohama, Japan, 1994.