EFFICIENT 2-PASS N-BEST DECODER

Long Nguyen, Richard Schwartz

BBN Systems & Technologies
70 Fawcett Street
Cambridge, MA. 02138, USA.
ln@bbn.com

ABSTRACT

In this paper, we describe the new BBN BYBLOS efficient 2-Pass N-Best decoder used for the 1996 Hub-4 Benchmark Tests. The decoder uses a quick fastmatch to determine the likely word endings. Then in the second pass, it performs a time-synchronous beam search using a detailed continuous-density HMM and a trigram language model to decide the word starting positions. From these word starts, the decoder, without looking at the input speech, constructs a trigram word lattice, and generates the top N likely hypotheses. This new 2-pass N-Best decoder maintains comparable recognition performance as the old 4-pass N-Best decoder, while its search strategy is simpler and much more efficient.

1. INTRODUCTION

As previously described in [2], the old BBN BYBLOS decoder used a multi-pass search strategy consisting of 4 passes to generate the top N most likely hypotheses, which were then rescored using more detailed, but expensive knowledge sources. These N best hypotheses were then reordered and the top-1 hypothesis constituted the recognition result. For large vocabulary, the decoder was actually run in 2 separate modular programs with the first program doing the first 3 passes and saving the forward-backward information onto disk space. The second program, based on the forward-backward information, constructed a trigram cross-word lattice and then carried out another beam search on this lattice to determine the N most likely hypotheses.

For recognition tasks with very-large vocabulary and/or with higher error rates, the forward-backward information needed to be saved is substantially large. This used up a lot of disk space and created heavy network IO traffic. Furthermore, the process of constructing the full cross-word triphones lattice with copies to accomodate a trigram language model is rather complex and requires substantially large memory. These two issues of huge intermediate disk storage and a complicated algorithm to construct the lattice prompted us to look for an efficient simplified search strategy that could preserve the recognition performance.

The new 2-pass decoder, implemented in just a single program consists of the usual fastmatch forward pass followed by a time-synchronous beam search backwards using a trigram language model and a new N-Best algorithm as a combination of both the Traceback-Based and the Word Lattice N-Best algorithms as described in [4].

In this paper, first we will describe an efficient way to use the trigram language model during the time-synchronous beam search. Then we will explain the new N-Best algorithm. And finally, we will present the recognition performance of the new 2-pass decoder in contrast to that of the old 4-pass decoder.

2. ALGORITHMS

The new 2-pass decoder still uses the same Forward-Backward Search algorithm [1] within the Multiple-Pass Search strategy [4] where the first forward pass is just a *fastmatch* as in the old BYBLOS 4-pass N-Best decoder [3] [2]. The sole purpose of the fastmatch is to record the word ending times and scores to guide the next pass. At each time frame, we record the score of the final state of each word ending. We will denote the set of words whose final states are active at time t of the utterance as Ω^t and the scores of the final states of each word w in Ω^t as $\alpha(w,t)$. Each $\alpha(w,t)$ represents the probability of the speech from the beginning of the utterance up to time t given the most likely word sequence ending with word w times the probability of the language model for that word sequence.

The search algorithm of the second pass is essentially the time-synchronous beam search (with one copy of each of the few active words) constrained within the reduced search space produced by the first pass, working backward from the end to the beginning of the utterance, with some augmentation. The goal of the second pass is to record the word beginning times, scores, and their path history for N-Best generation later. At each frame, t, of the utterance, the exit score, $\beta(w_2,t)$, of each active word, w_2 (which really corresponds to the initial HMM state of the word, because the HMMs are operating in reverse), is propagated backwards through the grammar to give an input score for each next word, w_3. Each $\beta(w_2,t)$

represents the probability of the speech from the end of the utterance back to time t given the most likely word sequence ending with word w_2 times the probability of the language model for that word sequence. Each w_3 of the set Ω^{t-1} will be activated only if the product

$$\alpha(w_3, t-1)\beta(w_2, t)Pr(w_3|w_1, w_2)$$

is greater than some forward-backward pruning threshold, where w_1 is the best preceding word of w_2. Before moving on to the next frame of speech, the decoder saves enough information about those $w_2's$ and their path history for N-Best generation later.

When the decoder finishes matching back to the beginning of the utterance, from the saved word starts and their path history, it constructs a trigram word lattice and, without looking at the input speech again, generates the N most likely hypotheses (or just the top-1 hypothesis) as the recognition result.

2.1. Using Trigrams During the Beam Search

It is extremely expensive to exhaustively use a trigram language model during the time-synchronous beam search. In order to apply the trigram language model score, that is, to evaluate the $\Pr(w_3|w_1, w_2)$ for the will-be-active w_3, it's necessary to decode with separate copies of w_2 depending on each active w_1. A slightly suboptimal algorithm would be to look back at the history for all possible pairs (w_1, w_2). This requires intensive computation and storage. Furthermore, in a typical beam search, w_3 might be activated over a long span of time, where the trigram language model score needs to be calculated at each frame. The straight-forward algorithm could just calculate these scores on the fly which is very costly. However, any attempt to cache these computations requires a lot of storage.

Nevertheless, it's possible to approximate these trigram calculations without incurring much computation compared to the search that uses a bigram language model, while maintaining the power of the detailed trigram language model. In order to activate w_3 given the current w_2, it's sufficient to use only the best w_1 history that w_2 used. Looking up the trigram probability requires only slightly more computation than a bigram language model, i.e. calculating $\Pr(w_3|w_2)$, since w_2 maintains the best w_1 as its history. One can argue that this sub-optimal usage of the trigram language model would hurt the performance of the beam search in comparison to the exhaustive usage of the trigram language model. However, when used with the new N-Best algorithm described in the next subsection, most of the suboptimality is repaired and the decoder can maintain the recognition performance as it would with an exhaustive trigram usage.

2.2. New N-Best Algorithm

The second pass, using the sub-optimal trigram language model, works backward in time, that is, it starts by taking the final frame of the utterance and works its way back, matching frames earlier in the utterance until it reaches the start of the utterance. At each frame, t, of the utterance, for each ending word w_2, we record the word, its partial score $\beta(w_2, t)$, its starting time, and its best history w_1 for the traceback-based word lattice construction later. We denote the set of words ending at time t as Ψ^t.

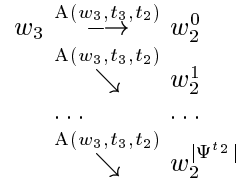
Traceback-Based Word Lattice Construction. First we want to construct a time-dependent left-to-right word lattice where each node represents a word and the arc between two nodes represents the segmental acoustic score of the word on the left node.

Given the set of tuples $(w_3, \beta(w_3, t_3), w_2, \beta(w_2, t_2))$ associated with all paths reaching the start of the utterance, we can construct recursively from the beginning of the utterance a word lattice of the most likely word sequences that matched the speech.

At each time frame, for a starting word w_3 and its associated tuple $(w_3, \beta(w_3, t_3), w_2, \beta(w_2, t_2))$, we can create a subgraph connecting w_3 to *all* w_2^i of the set Ψ^{t_2}. First we derive the acoustic score for w_3 within the time interval $[t_3, t_2)$ as

$$\mathrm{A}(w_3, t_3, t_2) = \frac{\beta(w_3, t_3)}{\beta(w_2, t_2)Pr(w_3|w_1, w_2)}$$

where w_1 is the best history of w_2. This $w_3's$ acoustic score $\mathrm{A}(w_3, t_3, t_2)$ will serve as the transition cost going from w_3 to each of the w_2^i ending at time t_2. For illustration purposes, the subgraph we want to create would look like the following:



For each word w_2^i of the set $\Psi^{t_2}, 0 \leq i < |\Psi^{t_2}|$, (that is, those words ended at time t_2), we create a lattice node representing that word and create an arc connecting w_3 to it having the segment acoustic score $\mathrm{A}(w_3, t_3, t_2)$ as the initial transition cost. If this is the first time the node w_2^i is created, we repeat the process with w_2^i now acting as w_3, then remember this w_2^i for later use. Otherwise, that is, if this w_2^i has been processed before, we clone the out arcs of the remembered w_2^i to make the out arcs for the current w_2^i. It can be shown that the resulting lattice

consists of transitions strictly from left to right in terms of time. Also, if either the start or the end time or both of a word are different, the word will have different nodes.

Then we need to apply the trigram language model scores on the arcs before doing the N-Best traceback. This can be done easily by a recursion starting from the furthest right node of the lattice, i.e. the node representing the end of the utterance, working backward. At each node, say w_1, all different coming arcs into w_1 actually originating from the same word w_2 but at different nodes due to the different word w_3 to accomodate the different trigram contexts. On these incoming arcs, we multiply the initial transition costs $\mathrm{A}(w_2, t_2, t_1)$, by the trigram language model score $Pr(w_3|w_1, w_2)$ to complete the transition costs:

$$w_3 \stackrel{\cdots}{\longrightarrow} w_2 \stackrel{\mathrm{A}(w_2,t_2,t_1)Pr(w_3|w_1,w_2)}{\longrightarrow} w_1$$

By this construction algorithm, all the trigram language model scores used sub-optimally before are now replaced with the optimal scores after the word boundaries have been determined. At first glance, this replacement doesn't seem to completely remove the sub-optimality of the trigram language model scores. However, experiments have shown that when used with fairly-detailed acoustic models earlier on, there's not much error in determining the word boundaries.

Finally, the resulting trigram word lattice needs to be sorted to facilitate the N-Best traceback later. This is also done recursively starting from the furthest left node of the lattice, working forward. At the node w_3, for each leaving out arcs to the nodes w_2^i of the set $\Psi^{t_2}, 0 \leq i < |\Psi^{t_2}|$, the arc cost is now updated with the best partial score $\beta'(w_2^i, t_2)$ for the word sequence from the end of the utterance back to this w_2^i. That is,

$$w_3 \stackrel{\mathrm{A}(w_3,t_3,t_2)Pr(w_3|w_1,w_2^i)\beta'(w_2^i,t_2)}{\longrightarrow} w_2^i \stackrel{\cdots}{\longrightarrow} w_1$$

Note that the $\beta'(w_2^i, t_2)$ is possibly different than the original $\beta(w_2^i, t_2)$ since we have just updated it with the 'exact' trigram score. These leaving arcs out of w_3 to the w_2^i are then sorted in descending order such that the first arc out of w_3 has the highest score.

N-Best Traceback. We start from the left-most node representing the beginning of the utterance, recursively going depth-first while maintaining a single global array to accumulate the words of the hypothesis from left to right. At each node of the lattice, we accumulate the word of that node into the global array and its delta score (compared to the best score at that time). When we reach the end of the utterance, a hypothesis is complete at the global array with some delta score. This complete hypothesis is then copied into the N-Best storage. The recursion then back-tracks one level up and goes depth-first again.

Since we need only the top N best hypotheses, we can speed up the traceback immensely by using pruning, aborting the recursion on some path part-way if its accumulated delta score is below some threshold.

3. EXPERIMENTAL RESULTS

We performed an experiment to compare the recognition accuracy and the resource utilization of this new 2-pass decoder with the old 4-pass decoder.

3.1. Acoustic Models

At BBN, we have different acoustic models with different levels of detail to be used at different stages of the multiple-pass search strategy. The broadest model is the Phonetically-Tied-Mixture (PTM), consisting of a set of 46 mixture densities modeling the 46 different phonemes, which is fairly cheap in terms of computation. Each mixture density consists of 256 diagonal Gaussians shared by all triphones having the same phoneme.

A more detailed model is the within-word triphone State-Clustered Tied-Mixture (SCTM-n). In this model, the different states of a triphone HMM have different sets of Gaussian mixtures. However, several corresponding states of several different triphones having the same phoneme (we call them a state cluster) may share a set of Gaussian mixtures.

The most detailed model is the cross-word triphone SCTM (SCTM-x) which takes into account the coarticulation effect across word boundaries.

3.2. Configuration

In the configurations of both the old and new decoders, the initial fastmatch used only a bigram language model and a PTM acoustic model. An identical trigram language model was used both in the second pass of the new decoder and the second and third passes of the old decoder. For acoustic models, the old 4-pass decoder, in the second and third passes used the same PTM model whereas the new decoder used the SCTM-n model in its second pass. In the fourth pass, the old decoder used a trigram language model and the SCTM-x model.

3.3. Results

Both decoders generated N-Best hypotheses which were then rescored using the same cross-word SCTM acoustic model and a trigram language model. We then compared the word error rates of the top-1 of

these two sets of optimized N-Best hypotheses. On the development testset h1d94 of the WSJ corpus, using the standard 20K open language model, the results of the new decoder was insignificantly different from the old decoder as shown in the table below.

Testset	*Old Decoder*	*New Decoder*
	WER(%)	*WER(%)*
h1d94	11.17	11.34

Table 1: *Comparison of Word Error Rate (WER)*

On another development testset of the new broadcast news corpus, using an in-house 20K closed language model, we also observed the same insignificant degradation as that of the WSJ testset.

3.4. Running Time

In the second comparison experiment on the development testset of the Broadcast News corpus running on the (rather obsolete) Silicon Graphic Indy R4400 machines, the new 2-pass decoder was almost three times faster than the old 4-pass decoder . The timing is broken down in the table below.

Pass	*Old Decoder*	*New Decoder*
	X Realtime	*X Realtime*
1	45.42	46.05
2	22.60	34.46
3	26.38	
4	121.73	
Total	216.13	80.51

Table 2: *Comparison of Running Times*

4. DISCUSSION

In general, the new 2-pass decoder is superior than the old 4-pass decoder. It still utilizes the Forward-Backward Search algorithm within the Multiple-Pass Search Paradigm suitable for very-large vocabulary recognition tasks. The new decoder can maintain a comparable recognition accuracy as the old decoder while its running time is almost three times faster. Furthermore, the new decoder doesn't require intermediate disk storage as compared to the old 4-pass decoder.

As in the past, we still consider the N-Best paradigm as a simple but very handy tool for research. Therefore, a decoder with the ability to generate N-Best is a requirement in our system development. At the minimal usefulness, the N-Best hypotheses are very good for optimization of system parameters. They can also be used for a quick probe of a new acoustic or language model through rescoring.

We would like to emphasize here that this new N-Best algorithm looks simple and seems weak (in contrast to the Word-Dependent N-Best algorithm). However, when used with fairly detailed acoustic and language models (such as SCTM and trigram), it performs almost as well as other algorithms. The interesting aspect of this algorithm is that it allows us to use the trigram language model approximately during the beam search without incurring much more computation in comparison to using a bigram language model and then to fix up the approximation later in an economical way. In one experiment to measure this effect, we observed that after fixing up the trigram approximation, the accuracy of the top-1 hypothesis is relatively 5% better than the result directly from the beam search.

5. CONCLUSION

We developed a new efficient 2-pass search strategy that allowed us to incorporate a trigram language model and a fairly detailed acoustic model early. We also developed a new simple N-Best algorithm that performed as well as other algorithms. This new 2-pass decoder achieved almost identical recognition performance as the old 4-pass decoder while running almost 3 times faster than the old decoder. Furthermore, the new decoder didn't require any intermediate disk storage.

Acknowledgements

This work was supported by the Advanced Research Projects Agency and monitored by Ft. Huachuca under contract No. DABT63-94-C-0063. The views and findings contained in this material are those of the authors and do not necessarily reflect the position or policy of the Government and no official endorsement should be inferred.

References

1. Austin, S., Schwartz, R., Placeway, P., "The Forward-Backward Search Algorithm" Proc. of IEEE ICASSP-91, Toronto, Canada, May 1991, pp. 697-700.
2. Nguyen, L., Schwartz, R., et al., "Is N-Best Dead", *Proc. of ARPA Human Language Technology Workshop*, Princeton, NJ, Mar. 1994, pp. 411-414.
3. Nguyen, L., Anastasakos, T., Kubala, F., LaPre, C., Makhoul, J., Schwartz, R., Yuan, N., Zavaliagkos, G., Zhao, Y., "The 1994 BBN/BYBLOS Speech Recognition System", *Proc. of ARPA Spoken Language Systems Technology Workshop*, Austin, TX, Jan. 1995, pp. 77-81.
4. Schwartz, R., Nguyen, L., Makhoul, J., "Multiple-Pass Search Strategy", *Automatic Speech and Speaker Recognition: Advanced Topics*, Kluwer Academic Publishers, Boston, 1996, pp.429-456.