IMPROVED LEXICAL TREE SEARCH FOR LARGE VOCABULARY SPEECH RECOGNITION

Stefan Ortmanns, Andreas Eiden, Hermann Ney

Lehrstuhl für Informatik VI, RWTH Aachen – University of Technology
52056 Aachen, Germany
ortmanns@informatik.rwth-aachen.de

ABSTRACT

This paper describes some extensions to the language model (LM) look-ahead pruning approach which is integrated into the time-synchronous beam search algorithm. The search algorithm is based on a lexical prefix tree in combination with a word-conditioned dynamic search space organization for handling trigram language models in a one-pass strategy. In particular, we study several LM look-ahead pruning techniques. Further, we improve the efficiency of this look-ahead technique by exploiting subtree dominance. This method avoids the computation of redundant subtrees within the copies of the lexical prefix tree and thus reduces the memory requirements of the search algorithm. In addition, we present a pruning criterion depending on the state index. The experimental results on the 20 000-word NAB'94 task (ARPA North American Business Corpus) indicate that the computational effort can be reduced to 4 times real time on a ALPHA 5000 PC without a significant loss in the recognition accuracy.

1. INTRODUCTION

The paper presents an efficient one-pass time-synchronous beam search strategy for handling trigram language models, where the search algorithm uses word-conditioned copies of the tree-organized pronunciation lexicon (lexical prefix tree) [3, 4]. The novel contributions of this paper are: We describe the incorporation of an exact trigram (LM) look-ahead technique into the acoustic pruning process of the word conditioned one-pass search algorithm and compare this look-ahead technique with the use of two approximative LM look-ahead techniques, namely a bigram LM and a unigram LM look-ahead. The efficiency of this technique can be further refined by using the so-called 'subtree dominance' (SD) criterion [1]. The idea of this enhancement is to eliminate redundant calculations of subtrees which are due to the word-conditioned tree copy concept. To this purpose, we compute to the factored LM probabilities the so-called factored SD probabilities in order to compare subtrees of different tree copies. This enables us to prune worse scored subtrees without producing search errors. The factorization process of the LM and SD probabilities is only performed for active tree copies using a dynamic programming scheme described in [5]. In addition, we introduce a pruning criterion which depends on the state index. For each state of the lexical prefix tree, we only expand state hypotheses with a score relatively close to best state hypotheses corresponding to same state index.

The organization of this paper is as follows. In Section 2, we review the one-pass beam search using a lexical prefix tree lexicon in combination with a trigram language model. In Section 3, an improved LM look-ahead pruning technique will be presented. In particular, we describe the exact trigram LM look-ahead pruning and introduce the subtree dominance criterion. The state dependent pruning criterion will be described in Section 4. In Section 5, we give experimental results on the NAB'94 20 000-word task.

2. LEXICAL TREE SEARCH METHOD

In this section, we give a brief description of the word-conditioned tree search algorithm in the context of handling trigram language models. For a lexical prefix tree, we have to take into account that for a trigram the language model probabilities are conditioned on the immediate *two* predecessor words rather than on one predecessor word in the bigram case [3]. Therefore, the incorporation of a trigram language model into the word conditioned tree search method requires a separate copy of the lexical prefix tree for each two-word history (u, v). The formulation of the DP search algorithm is based on the two quantities $Q_{uv}(t, s)$ and $B_{uv}(t, s)$ depending on the two-word history (u, v) as introduced in [3, 4]:

$Q_{uv}(t, s) :=$ overall score of the best path up to time t that ends in state s of the lexical tree for the two-word history (u, v).

$B_{uv}(t, s) :=$ starting time of the best path up to time t that ends in state s of the lexical tree for the two-word history (u, v).

Within a tree the usual dynamic programming can be expressed as:

$$\begin{aligned} Q_{uv}(t, s) &= \max_{\sigma} \{ q(x_t, s|\sigma) \cdot Q_{uv}(t-1, \sigma) \} \\ B_{uv}(t, s) &= B_{uv}(t-1, \sigma^{max}_{uv}(t, s)) . \end{aligned}$$

$\sigma^{max}_{uv}(t, s)$ is the optimum predecessor state for the hypothesis (t, s) of the two-word history (u, v) and $q(x_t, s|\sigma)$ denotes the product of transition and emission probabilities. To perform the recombination across the word boundaries, we use the quantity:

$$H(v, w; t) \quad := \quad \max_{u} \{ p(w|u, v) \cdot Q_{uv}(t, S_w) \} ,$$

where $p(w|u, v)$ is the conditional trigram probability for the word triple (u, v, w) and where S_w is the terminal state of word w in the lexical prefix tree. To initialize the new tree start-up hypotheses, we have to pass on the score and the time index before processing the hypotheses for time frame t:

$$\begin{aligned} Q_{uv}(t-1, s=0) &= H(u, v; t-1) \\ B_{uv}(t-1, s=0) &= t-1 , \end{aligned}$$

where we have introduced the fictitious state $s = 0$ for initialization. The management of the dynamic search space is based on a list organization [2] refined by a hashing approach as described in [4]. The standard pruning approach of the lexical tree search method is performed every 10-ms time frame and consists of three pruning steps: acoustic pruning, language model pruning and histogram pruning. In addition, an approximative language model look-ahead, based on a unigram, is integrated into the standard pruning approach [7]. To reduce the computational cost of the log-likehood calculations, a fast log-likelihood calculation method is used [6].

3. IMPROVED LM LOOK-AHEAD

The aim of the LM look-ahead technique is the incorporation of the language model probabilities as early as possible into the search process so that tighter pruning thresholds in the acoustic pruning and a lower number of maximum state hypotheses per time frame in the histogram pruning can be used. In the following we present some extensions to the LM look-ahead pruning technique presented in [5]. The characteristic features of this technique can be summarized as follows: The calculation of the LM factored probabilities is based on a compressed lexical prefix tree (LM look-ahead tree) and on a dynamic programming procedure which allows us to compute the LM factored probabilities only for active trees *on demand*. The so-called LM look-ahead tree only represents phoneme arcs with more than one successor arc in the lexical prefix tree. For practical aspects, we limit the LM look-ahead tree to the first three arc generations of the lexical prefix tree. However, this LM look-ahead technique was only applied in the context of the word conditioned tree search method using a bigram language model [5] so that an extension of handling trigram language models is needed.

3.1. Trigram LM Look-Ahead

For a trigram language model, the probabilities are factored over the LM look-ahead tree in such a way that each state s of the tree corresponds to the maximum trigram probability over all words that are reachable via this specific state corresponding to the lexical tree copy with the two-word history (u, v). This can be expressed as:

$$\pi_{uv}(s) \;:=\; \max_{w \in \mathcal{W}(s)} p(w|u,v) \;,$$

where $\mathcal{W}(s)$ is the set of words that can be reached from tree state s. The term $p(w|u,v)$ denotes the conditional trigram probabilities. Strictly speaking, we should use the tree nodes (or arcs) rather than the states of the Hidden Markov models that are associated with each node. However, each initial state of a phoneme arc can be identified with its associated tree node. To incorporate the factored LM probabilities into the acoustic pruning process of the word conditioned tree search method, we have to modify the dynamic programming recursion across phoneme boundaries:

$$\begin{aligned} \tilde{Q}_{uv}(t,s) \quad &:= \quad \pi_{uv}(s) \cdot Q_{uv}(t,s) \\ \tilde{Q}_{AC}(t) \quad &:= \quad \max_{(s;uv)} \left\{ \tilde{Q}_{uv}(t,s) \right\} \;. \end{aligned}$$

A state hypothesis $(t, s; uv)$ will be pruned if

$$\tilde{Q}_{uv}(t,s) \quad < \quad f_{AC} \cdot \tilde{Q}_{AC}(t) \;,$$

where f_{AC} denotes the acoustic pruning factor. As mentioned before, instead of calculating the factored trigram probabilities for all possible tree copies beforehand, we calculate the factored probabilities on demand for each new tree hypotheses depending on the predecessor wordpair (u, v). These factored LM probabilities are then stored in a look-up table. The LM factorization can be efficiently achieved by a dynamic programming approach which is similar to the method introduced in [5]. In a first step, the leaves of the LM look-ahead tree depending on a tree copy (u, v) are associated with the conditional trigram probabilities $p(w|uv)$. Then, the LM factored probabilities are propagated backwards from the tree leaves to the tree root by using a dynamic programming recursion, which determines for each tree node the successor node with the maximum look-ahead probability. Table 1 summarizes the details of dynamic programming procedure of the on-demand LM factorization.

Table 1: Dynamic programming algorithm for the on-demand factorization of the trigram LM probabilities within a LM look-ahead tree.

for each new copy (u, v) of the lexical prefix tree
- create a separate copy of the LM look-ahead tree
- initialization: leaves S_w: $\pi_{uv}(S_w) := p(w|u,v)$ nodes s: $\pi_{uv}(s) := 0$
- propagate the LM factored probabilities from each node s to its parent node $\tilde{s}$: $\pi_{uv}(\tilde{s}) \;:=\; \max_{s} \{ \pi_{uv}(s) \}$
- store $\pi_{uv}(s)$ in the lookup table

However, the number of LM look-ahead trees increases with the number of different tree hypotheses during the search. Thus, it can be useful to approximate the trigram in the LM look-ahead by a bigram so that the LM look-ahead trees depend only on the immediate predecessor word v and not on the two predecessor words (u, v). Thus, we have to adapt the acoustic scores across phoneme boundaries as follows:

$$\tilde{Q}_{uv}(t,s) \quad := \quad \pi_v(s) \cdot Q_{uv}(t,s) \;,$$

where $\pi_v(s) = \max_{w \in \mathcal{W}(s)} p(w|v)$ denotes the factored bigram LM probabilities and $p(w|v)$ the conditional bigram probabilities.

3.2. Subtree Dominance

Nevertheless, when using such a LM look-ahead pruning technique additional computation effort has to be spend for the same part of the lexical prefix tree in different tree copies. To reduce the redundant calculations of these subtrees, we enlarge the LM look-ahead pruning concept by exploiting subtree dominance (SD) [1]. That means, an instance of a subtree is dominated by another subtree, if the overall score of this instance is lower than the score of another subtree considering the *worst* LM model probability in that subtree. Thus, the dominated subtree can be eliminated for further considerations. To incorporate the subtree dominance criterion into the LM look-ahead pruning, we associate each state s (strictly speaking arc or node) of the LM look-ahead tree with the worst LM probability (factored SD probability) of all reachable word ends from this state. In the case of a trigram, the factored SD probability $\overline{\pi}_{uv}(s)$ for state s depending of the two-word history (u, v) can be defined as:

$$\overline{\pi}_{uv}(s) \ := \min_{w \in \mathcal{W}(s)} p(w|uv) \ .$$

To apply the associated pruning criterion across phoneme boundaries, we compute for each state (t, s) the *best* factored SD probability $\overline{Q}_{SD}(t, s)$:

$$\begin{aligned} \overline{Q}_{uv}(t,s) &:= \overline{\pi}_{uv}(s) \cdot Q_{uv}(t,s) \\ \overline{Q}_{SD}(t,s) &:= \max_{(uv)} \left\{ \overline{Q}_{uv}(t,s) \right\} \ . \end{aligned}$$

Then, we prune a state hypothesis $(t, s; uv)$ if:

$$\tilde{Q}_{uv}(t,s) < \overline{Q}_{SD}(t,s) \ .$$

The computation of the factored SD probabilities $\overline{\pi}_{uv}(s)$ can be performed with the same dynamic programming procedure as shown in Table 1. For this, we propagate the worst factored LM probabilities $\overline{\pi}_{uv}(s)$ from the leaves to the tree root. This operation will be done in parallel to the computation of the usual factored LM probabilities $\pi_{uv}(s)$.

4. STATE DEPENDENT PRUNING

In the usual standard beam pruning or so-called acoustic pruning, state hypotheses with a score relatively close to the best of all active states are retained as active, the others are pruned. Thus, a state $(t, s; uv)$ is removed if:

$$Q_{uv}(t,s) < f_{AC} \cdot \max_{(s';uv)} \left\{ Q_{uv}(t,s') \right\} \ ,$$

where f_{AC} denotes the acoustic pruning factor. The acoustic pruning can be further refined by applying the concept of subtree dominance on state index level. In principle, we have to compare only hypotheses with the same state index (but with different word histories). This so-called state dependent pruning performs as follows: For each state s, we determine the best scoring hypothesis (t, s):

$$Q_D(t,s) = \max_{(uv)} \left\{ Q_{uv}(t,s) \right\} \ .$$

Denoting the so-called state dependent pruning factor with f_D, a state hypothesis $(t, s; uv)$ will be removed if

$$Q_{uv}(t,s) < f_D \cdot Q_D(t,s).$$

5. EXPERIMENTAL RESULTS

The experimental tests were carried out on the ARPA North American Business (NAB'94) H1 development corpus comprising 310 sentences with a total of 7 387 words. In all recognition experiments, we used a 22 411-word lexicon containing 2 434 pronunciation variants and a trigram language model with a perplexity of $PP_{tri} = 141.1$ [8]. 199 of the spoken words were out-of-vocabulary words. We used about 290 000 Laplacian mixture densities for each gender. All experiments were performed on an ALPHA 5000 PC (SpecInt'95: 15.4). The results of the recognition tests are depicted in Table 2. The Table shows the maximum number of LM look-ahead trees per time frame (LMLA-trees) required during the search process, the average search space per time frame in terms of state, arc and tree hypotheses, the recognition errors (deletions (DEL), insertion (INS), word error rate (WER)) and the real time factor (RTF) for different LM look-ahead techniques.

First, we investigated the effect of the LM look-ahead on the size of search space and the recognition accuracy. In an initial experiment, we performed two tests with a so-called unigram LM look-ahead [7]. The corresponding LM look-ahead tree consists of 68 587 phoneme arcs which conforms to the size of phoneme arcs of the lexical prefix tree. To achieve a word error rate of 14.3%, on average 21466 state hypotheses per time frame are needed resulting in a RTF of 9.1. Then, we tested the bigram and the trigram LM look-ahead for different pruning thresholds. The used LM look-ahead tree comprises 12 764 arcs. Considering the recognition results without exploiting subtree dominance, the bigram LM look-ahead reduces the size of the search space by a factor of 4 in comparison to the unigram LM look-ahead. The trigram LM look-ahead further reduces the size of search space by about 20%. However, the bigram LM look-ahead leads to a greater acceleration of the search since the LM look-ahead trees depends only on the immediate predecessor word. The maximum number of state hypotheses per time frame ($MaxHyp = 100\,000$) used in the histogram pruning was conservatively large adjusted beforehand so that a further reduction of the size of the search space can be expected by decreasing $MaxHyp$.

Then, we tested the subtree dominance (SD) pruning criterion. To show the efficiency of the SD pruning, we performed a first series of experiments with only one acoustic pruning parameter ($F_{LM} = \infty, MaxHyp = \infty$). Comparing these results to the results with a fixed LM threshold (F_{LM}) of 60, the search space is only increased by about $10 - 15\%$. Obviously, when exploiting subtree dominance less search errors are made and the memory cost of the LM look-ahead as well as the computational cost can be further decreased. In a further series of experiments, we added the state dependent pruning approach to the LM look-ahead using subtree dominance. As we can see from Table 2, a slight speed-up of the search can be achieved. Finally, we will consider the recognition cost for the search using LM look-ahead pruning combined with subtree dominance and state dependent pruning (Table 3). The computational time (real time factor, RTF) and the typical memory cost (mega bytes [MB]) are given for each of the main search operations: log-likelihood calculation, acoustic search, LM recombination, LM look-ahead and LM access measured on a subset of the test corpus (every first sentence of the 20 speakers). The results are shown in Table 3. Due to the high cost of the LM access during the trigram factorization, the bigram LM look-ahead performs slightly faster than the trigram LM look-ahead.

6. SUMMARY

In this paper, we described an improved lexical tree search method for handling trigram language models using LM look-ahead techniques. The efficiency of the LM look-ahead techniques can be substantially improved by exploiting subtree dominance with virtually no loss in the recognition accuracy. The state dependent pruning method leads to a further speed-up of the search. For the 20 000-word NAB'94 task, the search can be speeded up to 3.3 times real time without a significant effect on

Table 2: Effect of the several LM look-ahead pruning techniques on the search effort and recognition accuracy (NAB'94 H1 development set, trigram LM with $PP_{tri} = 141.1$; ALPHA 5000 PC (SpecInt'95: 15.4)).

Type of LM look-ahead		F_{AC}	F_{LM}	F_D	LMLA trees	search space states	arcs	trees	recognition errors [%] DEL / INS	WER	RTF
unigram LM look-ahead	without SD	120	60	–	1	21466	6403	64	1.7 / 2.7	14.3	9.1
$(PP_{uni} = 996.6)$		110	60	–	1	14414	4346	52	1.7 / 2.7	14.4	7.1
bigram LM look-ahead	without SD	100	60	–	195	5891	1818	35	1.8 / 2.7	14.2	5.8
$(PP_{bi} = 203.4)$		90		–	171	3238	1022	25	1.8 / 2.8	14.6	4.6
	with SD	100	∞	–	636	5285	2978	35	1.7 / 2.6	13.9	7.0
			60	–	192	5094	1532	26	1.8 / 2.6	14.0	5.4
				40	189	4340	1298	18	1.8 / 2.6	14.1	4.7
		90	∞	–	426	2879	889	21	1.8 / 2.7	14.3	4.8
			60	–	170	2858	882	18	1.8 / 2.7	14.4	3.5
				40	162	2595	801	13	1.8 / 2.7	14.4	3.3
trigram LM look-ahead	without SD	100	60	–	929	4914	1520	35	1.7 / 2.7	14.2	8.4
$(PP_{tri} = 141.1)$		90		–	785	2618	827	24	1.8 / 2.7	14.4	5.7
	with SD	100	∞	–	3221	4725	1706	35	1.7 / 2.6	13.9	10.7
			60	–	785	4200	1266	26	1.7 / 2.6	14.0	6.3
				40	638	3557	1067	19	1.7 / 2.7	14.1	6.0
		90	∞	–	1701	2580	789	21	1.8 / 2.7	14.3	7.9
		90	60	–	673	2322	718	18	1.7 / 2.7	14.3	4.8
				40	559	2098	648	14	1.7 / 2.7	14.4	4.3

Table 3: Recognition effort for a trigram LM look-ahead $(PP_{tri} = 141.1)$ and a bigram LM look-ahead $(PP_{bi} = 203.4)$ in combination with subtree dominance and state dependent pruning (Pruning thresholds: $F_{AC} = 100, F_{LM} = 60, F_D = 40$; Test conditions: subset of the NAB'94 H1 development set: 20 speakers = 20 sentences = 519 spoken words = 211.24 s; ALPHA 5000 PC (SpecInt'95: 15.4)).

LM look-ahead (LA)	bigram LA		trigram LA	
states/arcs/trees	4116 / 1244 / 17		3467 / 1054 / 25	
word error rate	72/519 = 13.8%		70/519 = 13.5%	
recognition effort	RTF	MB	RTF	MB
log-likelihood calc.	1.4	35	1.3	35
acoustic search	2.1	7	1.3	7
LM recombination	0.1	-	0.1	-
LM look-ahead	0.3	25	0.5	80
LM access	0.9	60	2.3	60
other operations	0.1	2	0.2	2
overall effort	4.9	129	5.7	184

the recognition accuracy.

Acknowledgement. This research was partly funded by grant 01 IV 701 T4 from the German Ministry of Science and Technology (BMBF) as a part of the VERBMOBIL project. The views and conclusions contained in this document are those of the authors.

7. REFERENCES

[1] F. Alleva, X. Huang, M.-Y Hwang: Improvements on the Pronunciation Prefix Tree Search Organization. Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing, Atlanta, GA, pp. 133 - 136, May 1996.

[2] H. Ney, R. Haeb-Umbach, B.-H. Tran, M. Oerder: Improvements in Beam Search for 10000-Word Continuous Speech Recognition. 1992 IEEE Int. Conf. on Acoustics, Speech and Signal Processing, San Francisco, CA, pp. 13-16, March 1992.

[3] H. Ney: Search Strategies for Large-Vocabulary Continuous-Speech Recognition. NATO Advanced Studies Institute, Bubion, Spain, June-July 1993, pp. 210–225, in A.J. Rubio Ayuso, J.M. Lopez Soler (eds.): 'Speech Recognition and Coding – New Advances and Trends', Springer, Berlin, 1995.

[4] S. Ortmanns, H. Ney, F. Seide, I. Lindam: A Comparison of Time Conditioned and Word Conditioned Search Techniques for Large Vocabulary Speech Recognition. Proc. Int. Conf. on Spoken Language Processing, Philadelphia, PA, pp. 2091-2094, October 1996.

[5] S. Ortmanns, A. Eiden, H. Ney, N. Coenen: Look-Ahead Techniques for Fast Beam Search. Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing, Munich, Germany, pp. 1783-1786, April 1997.

[6] S. Ortmanns, H. Ney, T. Firzlaff: Fast Likelihood Computation Methods for Continuous Mixture Densities in Large Vocabulary Speech Recognition. Fifth European Conference on Speech Communication and Technology, Rhodes, Greece, pp. 143-146, September 1997.

[7] V. Steinbiss, B.-H. Tran, H. Ney: Improvements in Beam Search. Proc. Int. Conf. on Spoken Language Processing, Yokohama, Japan, pp. 2143-2146, September 1994.

[8] F. Wessel, S. Ortmanns, H. Ney: Implementation of Word Based Statistical Language Models. Proc. Spoken Queries in European Languages Workshop on Multi-Lingual Information Retrieval Dialogs, Pilsen, Czech Republic, pp. 55-59, April 1997.