ONE-PASS LVCSR ALGORITHM USING LINEAR LEXICON SEARCH AND 1-BEST APPROXIMATION TREE-STRUCTURED LEXICON SEARCH

Norihide Kitaoka†‡, Ying Liang†, Nobutoshi Takahashi‡, Seiichi Nakagawa‡

†Naogya University, Nagoya, 464−8603, Japan
‡Toyohashi University of Technology, Toyohashi, Aichi, 441−8580 Japan
{kitaoka, liang338, nobutosi, nakagawa}@slp.ics.tut.ac.jp

ABSTRACT

We propose a combinational use of linear lexicon and tree-structured lexicon in a 1-best approximation search algorithm for large vocabulary continuous speech recognition. The algorithm based on 1-best approximation with a tree-structured lexicon is efficient but frequently loses the optimal sentence hypothesis. The linear lexicon search can find the optimal hypothesis but needs much computational cost. Thus, we propose a search method using these two search algorithms in parallel to achieve efficient and accurate decoding. We also adopt trigram and 4-gram language models to this algorithm to achieve 1-pass tri-gram and 4-gram decoding. These combination loses optimal hypotheses, but the usage of the linear lexicon reduced such errors. We introduce ‘likelihood difference index’ to realize accurate inter-word context-dependent modeling. We evaluated this new search algorithm and obtained significant improvement of recognition performance without severe increase of computational cost.

1. INTRODUCTION

Large vocabulary continuous speech recognition (LVCSR) based on hidden Markov model (HMM), acoustic modeling and N-gram language modeling, has been developed to a practical level and applied to practical systems such as dictations and broadcast news captioning, and further improvement of the technology will make it used more widely. HMM-based LVCSR generally has to take computational cost proportional to its vocabulary size if all the HMM states of the words in the vocabulary are expanded on the memory. A tree-structured lexicon in which prefixes expressed in sub-word units such as phones or syllables are shared among words is often used to reduce the HMM states [1].

The LVCSR system expands partial sentence hypotheses by concatenating possible succeeding words, but this expansion makes the number of hypotheses growing exponentially. To avoid this, some approximations on word histories are often used: we adopted the 1-best approximation [2] because of the smaller computational cost, but this method may lose an optimal hypothesis when integrated with a bigram language model and a tree-structured lexicon because of its inaccurate language look-ahead (that is, linguistic likelihood estimation based on an uncertain word history).

In this paper, we propose a new efficient and accurate search algorithm using the 1-best approximation tree-structured lexicon search and linear lexicon search. The linear lexicon search can apply language look-ahead exactly to each hypothesis, thus covers the drawbacks of the combination of the 1-best approximation and tree-structured lexicon search. We also adopt trigram and 4-gram language models to this algorithm. This combination may lose an optimal hypothesis, but the usage of the linear lexicon reduced such errors. Then, we show that this algorithm can model inter-word context-dependency with small modification with ‘likelihood difference index.’

2. COMBINATIONAL USE OF DYNAMICALLY EXPANDED LINEAR LEXICON AND STATIC TREE-STRUCTURED LEXICON

2.1. Conventional methods

2.1.1. Linear lexicon and tree-structured lexicon

HMM-based LVCSR takes more computational cost when the vocabulary size gets larger. When using a linear lexicon illustrated in Figure 1 (a), the number of expanded HMM states gets large proportionally to the vocabulary size and then the computational costs also proportional to the vocabulary size. The tree-structured lexicon illustrated in Figure 1 (b) is conventionally used to reduce the number of expanded HMM states.

When using N-gram language models, the likelihood of a hypothesis at time frame i which is generated by connecting a word w to the word sequence W is:

\[ P_{Ww}(i) = \max_j \{ P_W(j - 1) + Q_w(j, i) + \text{lang}(w|W) \}, \]

where j satisfies \( j < i \), \( P_W(j - 1) \) is the sum of the likelihoods of the hypothesis W at time j, \( Q_w(j, i) \) is the acoustic likelihood of the word w for period \( [j, i] \) and \( \text{lang}(w|W) \) is the language probability of word w given the word history W. If we use a linear lexicon, \( \text{lang}(w|W) \) can be considered at the time of connection of w and W, but if the lexicon is structured as a tree, the root node and many other nodes corresponding to word prefixes are shared among words and thus \( \text{lang}(w|W) \) cannot be added to \( P_{Ww}(i) \) at the time point on the fly.

2.1.2. Approximations of word history dependency for tree-structured lexicon search

The likelihood and boundary of a word depend on the word history. So we have to execute the forward decoding procedure independently for all new hypotheses generated by concatenating a word to all the possible hypotheses. Since many hypotheses...
are generated in this process, we have to make an assumption on the word history dependency and bundle the hypotheses using an approximation based on the assumption.

The word-pair approximation [3] assumes that the word boundaries depend on just one previous words. Under this assumption, Viterbi forwarding is executed on previous word-dependent HMM networks. This means that HMM networks have to be copied according to the word histories and the Viterbi search has to be executed on the networks in parallel.

We consider the combination of bigram language models and HMM acoustic models. Using tree-structured HMM networks, the language probability \( P(w_k|w_{k-1}) \) is multiplied to the acoustic score with an appropriate weight at the leaf node of the network because the word for the node is uniquely identified only at the leaf. Each network has a unique word history under the word-pair approximation, so \( P(w_k|w_{k-1}) \) can be exactly applied.

On the other hand, the 1-best approximation [2] does not assume any word-history dependencies. This approximation reduces the computational cost very much because system does not need copies of the HMM network and uses only one reenterant network. We can determine the preceding word (that is, the word history) by back-tracing from the leaf of the network and thus we can apply the bigram probability \( P(w_k|w_{k-1}) \). Even if only one hypothesis is expanded to new hypotheses on the HMM network at every time frame, we can obtain approximately other hypotheses keeping other histories \( w'_{k-1}s \) at each frame and compensating the scores by two ways: (1) the differences of history scores between the score of the history ended with \( w_{k-1} \) and that ended with \( w'_{k-1} \), (2) the difference of language scores between \( P(w_k|w_{k-1}) \) and \( P(w_k|w'_{k-1}) \). But this approximation is not correct. The HMM network is shared by the hypotheses expanded at all the time frame and thus many paths are overridden and wiped off by the other path. So the path which is optimal when considering the language score which can be applied at the leaf node may be rejected by this mechanism. In this way, the optimal word sequence is not guaranteed to be obtained by tree-structured lexicon search under the 1-best approximation.

Ogata et al. [5] proposed a method in which trees were dynamically expanded only for the words which had bigram probabilities. Static trees were also used for unigram back-off. This method guarantees the optimal search for the best hypothesis, but consumes many memories. Hori et al. [6] assumed that the word boundary depended on only the preceding short phoneme sequence instead of the preceding word. This method needs smaller computational cost than word-pair approximation and also alleviates the word boundary approximation. But both methods still need much more computational costs than the 1-best approximation.

### 2.1.3. Recognition using linear lexicon

We compare the linear lexicon search and the tree-structured lexicon search. Linear lexicons for vocabulary words are connected each other from the tails to the heads in an HMM network. Using this network, any sub-word HMMs are not shared among words unlike the tree lexicon and determine the word associated to each node. This enables that the system applies the bigram language scores at the first node of each word whereas the tree-structured lexicon does not. This means that the system can use language look-ahead, resulting in efficient constraints of the search space.

Figure 2 (a) shows an example of the new word attachment to hypotheses using a linear lexicon. The word “a-sa-ga-o” is attached to the hypothesis D which maximizes the sum of the total score of hypothesis D and the language score \( P(\text{a-sa-ga-o}|D) \). On the contrary, the HMM network is attached to the hypothesis A in the case of a tree-structured lexicon as illustrated in Figure 2 (b) because of its highest likelihood among the hypotheses not considering the word probabilities when given the histories.

The HMM trellis is shared among words and histories under the 1-best approximation method and thus conflicts often occur. The HMM nodes in linear lexicons are also shared among histories and thus the optimal N-best hypotheses are not guaranteed to be obtained, but there are no conflicts among words and the first best hypothesis is guaranteed to be obtained.

But a linear lexicon does not allow node sharing among words and thus the computational cost needed is much higher than that for a tree-structured lexicon.

### 2.2. How to combine linear lexicon and static tree-structured lexicon

When using bigram language models, the probabilities of word pairs which are not or rarely found in training data are smoothed by unigram back-off. The combination of the bigram and the 1-best approximation search on a tree-structured lexicon looses the optimal search path, but unigram does not. The method proposed in [5] uses these characteristics well, but this method cannot use language look-ahead yet.

So we propose a new search method using search on dynamically expanded linear lexicon along with the 1-best approximation search on a tree-structured lexicon. That is, we basically use the 1-best approximation tree-structured lexicon search and also use the linear lexicon search in parallel. Only the small number of words \( N_{\text{lin}} \), which are dynamically selected, are expanded in a linear lexicon network.

In our dynamic selection of the words to be expanded in a linear network, we evaluate all the vocabulary words at every time frame using the following equations:

\[
Q_{\text{tree}}(w, t) = \max_v \{ P_{\text{hyp}}(v, t) + \text{lang}(w|v) \},
\]

\[
Q_{\text{lin}}(w, t) = \max_{s_w} \{ P_{\text{state}}(s_w, t) \},
\]

where \( P_{\text{hyp}}(v, t) \) is the likelihood of the final HMM state of the hypothesis which ends with the word \( v \), \( P_{\text{state}}(s_w, t) \) is the likelihood of the HMM state \( s_w \) in word \( w \) which is expanded on a linear HMM network at the time frame \( t \) and \( \text{lang}(w|v) \) is a look-ahead probability which is the appearance probability of word \( w \) given a word history \( v \). \( Q_{\text{tree}}(w, t) \) in Equation (2) means the maximum value among the sums of likelihood of preceding word sequence ended by word \( v \) in a tree-structured network and the look-ahead probability. \( Q_{\text{lin}}(w, t) \) in Equation (3), which means the maximum value among the HMM states of the word \( w \), is calculated for the word \( w \) which is already expanded in the linear HMM network. If a word \( w \) corresponds to both \( Q_{\text{tree}}(w, t) \) and \( Q_{\text{lin}}(w, t) \), then the maximum value of \( Q_{\text{tree}}(w, t) \) and \( Q_{\text{lin}}(w, t) \) is selected as the evaluation value for the word.

---

**Fig. 2** Word attachment with partial sentence hypothesis (assuming \( P(\text{A}) > P(\text{B}) \), \( P(\text{C}), P(\text{D}) \) and \( P(\text{a-sa-ga-o}|\text{D}) \geq P(\text{a-sa-ga-o}|\text{A}) \))
3. EXPAND BIGRAM LANGUAGE MODELS TO TRIGRAM AND 4-GRAM

Similar with using bigram language models, we expand bigram language models to trigram language models and 4-gram language models. If we adopt trigram language models and 4-gram language models, Equation (2) would turn into Equations (4) and (5), respectively:

\[ Q_{\text{tree}}(w, t) = \max_v (P_{\text{hyp}}(v, t) + \text{lang}(w|u, v)), \quad (4) \]
\[ Q_{\text{tree}}(w, t) = \max_v (P_{\text{hyp}}(v, t) + \text{lang}(w|s, u, v)), \quad (5) \]

where \( \text{lang}(w|u, v) \) and \( \text{lang}(w|s, u, v) \) are look-ahead probabilities which are the appearance probabilities of word \( w \) given word history \( u, v \) and \( u, s, u, v \), respectively.

When adopting bigram language models, the optimal hypothesis is guaranteed to be obtained using a linear lexicon. But adopting trigram and 4-gram language models, optimal hypotheses may be lost, because optimal language score for the current word according to two or three word history cannot obtain even if we use a linear lexicon. The two or three word history is traced by backtracking \( P_{uv} \) and \( P_{sv} \), as shown in Figure 3.

4. INTER-WORD CONTEXT DEPENDENT MODELING

Because Japanese syllables consist of a consonant and vowel pair, there are relatively small number (approx. 120) of syllables and we only need to consider small variations (approx. 7 or 8 including vowels and pause) of left context. So we can use full syllable modeling considering C-V context dependency and also easily extend the CI-HMMs to full left CD-HMMs to consider V-C context dependency.

As well known, intra-word context dependent modeling can be realized by describing the CD syllables in the dictionary. But inter-word modeling should be realized by recognition process.

As described in Figure 4 (a), we only need to make branches for the head syllable according to the contexts and the paths are merged at the second syllable for the linear lexicon. For the tree-structured lexicon, branches are made in a similar way. At the end node of a word the language scores have to be compensated considering the inter-word context, but the scores of contexts other than that of the best history have lost because of the merge at the second syllable. To solve this problem, we introduce the

\[ \text{'likelihood difference index'} \] (Figure 4(b)). The likelihood differences between phonetic contexts are calculated at the merging points and the difference index is inherited to the succeeding nodes. Using this index, language scores are accurately compensated considering inter-word phonetic contexts at the end node of the words.

5. EXPERIMENTS

We evaluated the effectiveness of the combination of the 1-best approximation tree-structured lexicon search and the accurate linear lexicon search.

5.1. Experimental conditions

To evaluate the performance, we used two kinds of data: set A: 100 sentences from Japanese newspaper read speech in JNAS corpus, and set B: 175 sentences from Japanese NHK broadcast news. All of speech data were sampled with a sampling frequency of 16 kHz, and the signal was pre-emphasized by a factor of 0.98. A Hamming window of 25 ms length was applied and shifted with the step of 10 ms. 38 dimensional feature vectors were used including 12 dimensional MFCCs, their first and second deviation coefficients and the first and second deviations of log power.

116 Japanese context-independent syllable HMMs (strictly speaking, mora HMMs [8]) including short pause and silence were trained using 27992 utterances read by 175 male speakers (INAS corpus). Using these CI-HMMs as base models, we also trained 928 context-dependent HMMs (CD-HMMs) with 8 contexts (5 vowels, silence, /N/, and short pause including /q/). Each continuous density HMM had 5 states, and 4 of them had pdfs of output probability. Each pdf consisted of four Gaussians with full-covariance matrices.

Bigram, trigram and 4-gram language models with a 20000 word vocabulary size were trained from the text of 45 months Mainichi Japanese newspaper with 90 million words.

We used a 2-pass decoder as the baseline. In the first pass, the CI-HMMs and bigram language models were used. In the second pass, rescoring using trigram (and CD-HMMs in the case of context dependent modeling) was applied on 200-best hypotheses generated by the first pass. Our proposed method was
implemented in the first pass to make it a one-pass trigram/4-gram decoder. Beam width at the word end was set to 30 according to preliminary experiments.

5.2. Linear lexicon vs. tree-structured lexicon

We compared the linear lexicon search and the 1-best approximation tree-structured lexicon search. Results are shown in Table 1. ‘Optimal rescore’ means the case that the best hypothesis is selected from each 200-best list by the second pass, that is, “oracle”. Comparing the results of the first pass, the 1-best approximation tree-structured lexicon search was inferior to the linear lexicon search. We can find the same relation in the results of optimal rescore and thus the final results had also the performance difference between these two methods. On the other hand, the search on the linear lexicon, which had approximately 300k HMM nodes, was less efficient than that on the tree-structured lexicon with approximately 100k HMM nodes. The realtime factors of the tree-structured lexicon search and linear lexicon search were 7.2 and 161.4, respectively.

5.3. Combination of linear lexicon and tree-structured lexicon

We evaluated the combination of dynamically expanded linear lexicon and static tree-structured lexicon described in Section 2.2. Table 2 (a) shows the results for set A. We set the size of linear lexicon \( N_{lin} = 250 \) and the performance with small \( N_{lin} \) achieved almost the same performance as using a linear lexicon only. Table 2 (b) shows that our proposed method worked well for more spontaneous speech. Computational cost of this combinational lexicon search was only 1.08 times as much as that of tree-structured lexicon search.

\(^{2}\)This value is much larger than theoretically predicted value (approx. 60). This is because we used slow implementation of sorting algorithm \( O(\sum_{i=1}^{N}N_{lin}) \) for beam search, which would not be dominant if we had used an alternative fast algorithm.

\(^{4}\)This is because we used slow implementation of sorting algorithm \( O(\sum_{i=1}^{N}N_{lin}) \) for beam search, which would not be dominant if we had used an alternative fast algorithm.

| Table 1. Comparison of linear lexicon search and tree lexicon 1-best approximation search for set A [%] |
|---------------------------------|---------------|-----------------|-----------------|
|                                | 1-pass (bigram) | optimal rescore | 2-pass (trigram) |
|                                | from 200 best   | from 200 best   | from 200 best   |
| (a) Set A (read speech) |     |     |     |     |     |     |
| tree | 82.5 | 87.8 | 91.4 | 94.0 | 85.7 | 90.2 |
| linear | 85.4 | 89.7 | 93.9 | 95.7 | 89.0 | 91.5 |
| (b) Set B (broadcast news) |     |     |     |     |     |     |
| tree | 60.9 | 69.4 | 68.1 | 73.6 | 63.5 | 70.6 |
| linear | 61.7 | 70.5 | 69.7 | 75.1 | 64.9 | 72.1 |

| Table 2. Comparison of combinational lexicon and tree-structured lexicon for set A (read speech) in one-pass decoding \( (N_{lin} = 250 \) combination method) [%] |
|---------------------------------|---------------|---------------|---------------|
|                                | bigram | trigram | 4-gram |
|                                | Acc. | Cor. | Acc. | Cor. | Acc. | Cor. |
| (a) Set A (read speech) |     |     |     |     |     |     |
| tree | 82.3 | 87.8 | 88.8 | 90.6 | 86.3 | 90.8 |
| combination | 85.4 | 89.7 | 89.8 | 91.7 | 91.1 | 92.0 |
| (b) Set B (broadcast news) |     |     |     |     |     |     |
| tree | 60.9 | 69.4 | 63.9 | 71.2 | 64.3 | 71.2 |
| combination | 61.7 | 70.5 | 65.7 | 72.5 | 66.1 | 72.6 |

5.4. Inter-word context dependent modeling

We evaluated the inter-word context dependent modeling with \( N_{lin} = 250 \). Results are shown in Table 3. We compared our proposed method to a method with acoustic and linguistic rescoring. The CD modeling in the first pass achieved significant improvement of recognition performance.

6. CONCLUSION

In this paper, we proposed a combination of linear lexicon and tree-structured lexicon for efficient and accurate LVCSR. We evaluated the method and proved that the small number of words dynamically expanded in the linear lexicon made the recognition performance improved to almost the same as the upper bound.

7. REFERENCES