6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

LARGE VOCABULARY KOREAN CONTINUOUS SPEECH RECOGNITION USING A ONE-PASS ALGORITHM

Ha-Jin Yu, Hoon Kim, Joon-Mo Hong, Min-Seong Kim, and Jong-Seok Lee

Information Technology Lab. MI group
LG Electronics Institute of Technology
hjyu@bulsai.kaist.ac.kr

ABSTRACT

In this paper, we describe problems in recognizing large-vocabulary Korean continuous speech, and proposed solutions to them. Korean sentences consist of *eojeol*s, which are separated by spaces in text and consist of morphemes. When we use morpheme units, there are many word insertion and deletion errors because morpheme units are too short. We introduce a between-word phone variation lexicon that can represent many alternatives of phones of words in one structure. The decoding algorithm is composed of one pass, which is a modification of token-passing algorithm. In this algorithm, we allowed multiple tokens in a state at a time to get global best path without expanding the states when we use trigram language models. We confirmed thatbetween-word phone variation lexicon is useful for morpheme-based recognition by observing that the improvement is higher for morpheme units than for *eojeol* units. Allowing multiple tokens at a state also improved the performance.

1. INTRODUCTION

In this paper, we described a large vocabulary Korean continuous speech recognizer by using a one-pass algorithm. The main difference between Korean and English is that the concepts of words are not the same. In Korean language, a sentence consists of *eojeol*s, which are separated by spaces in text and are agglomerates of morphemes[1]. We use morpheme as the word unit, because *eojeol* is a long unit similar to a phrase in English. The problem is that a morpheme is a small unit, which includes only few phones in many cases. The short unit means more words in a sentence, so there are heavy influences of between-word interferences. To solve this problem, we proposed between-word phone variation lexicon, which describes the phone variations caused by neighboring words.

The second problem caused by the short words is that the short words tend to be missed or inserted in the first pass and the errors can hardly be recovered in later passes. We can mitigate the errors by using trigrams in the first pass. However, it takes a large memory space or the result may not be a optimal solution. In this research, we solved the problem by allowing multiple tokens at a state, and delaying the decision of paths to an appropriate point.

The following section described the approaches in detail, and the experimental results are presented in section 3.

2. THE APPROACHES

2.1. Between-word phone variation lexicon

Applying cross-word modeling to Korean language is not as simple as in the case of English, because the phones at the boundaries of the morphemes are subject to change in various contexts. That is, merely replacing monophones at the boundaries of the words to triphones may not give large improvement. Figure 1 shows an example. The phone ‘k’ at the end of a Korean word ‘ /hanguk/(Korea)’ changes to ‘ng’ or ‘g’ according to the following words. The first phone ‘h’ also changes to four other phones according the preceding words. To reflect these variations, we can regard the variations as separate words [2], or insert phones optionally [5]. In this research, we represent a word by a graph structure. Then we convert the monophones in the graph to triphones as shown in Figure 2. The nodes at each end of the word are connected to the nodes at the beginnings of other words according to the word-pair frequencies and the phone context. As the result, a network of phones is constructed. As we traverse the network during the search, we can search through the phone sequences of the original *eojeol*s, as if the lexicon is constructed based on the *eojeol*s.

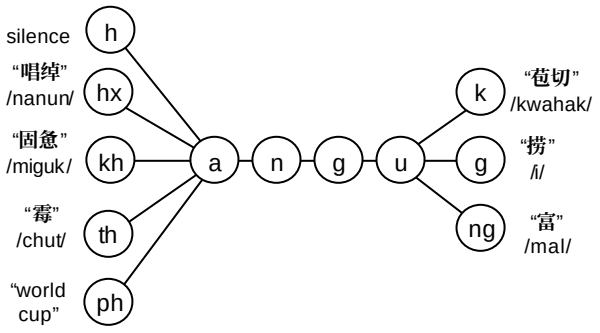


Figure 1: An example of the between-word phone variation lexicon

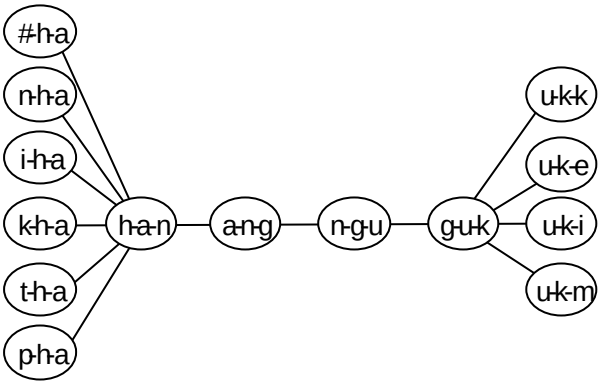


Figure 2: Trigram representation of Figure 1.

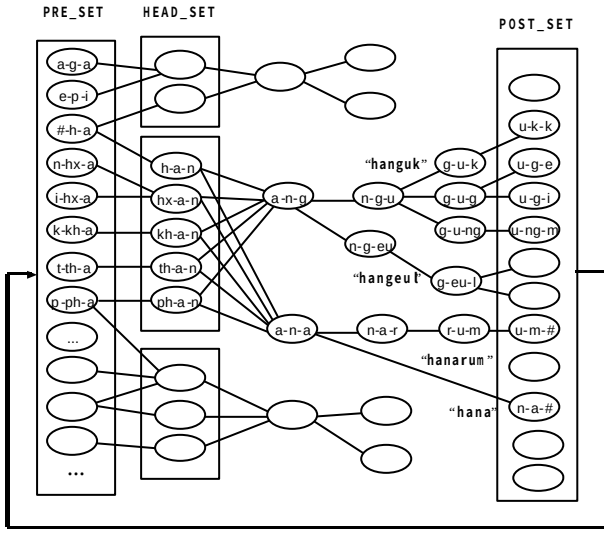


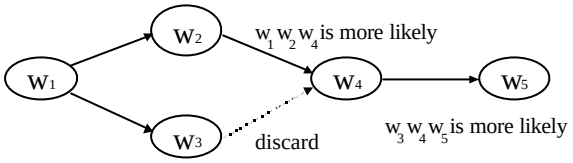
Figure 3: The tree structured recognition network. The PLUs in the PRE_SET, HEAD_SET and the POST_SET are shared by many words.

When we build the recognition network, it is necessary that making the lexicon into tree structure to speed up the search. However it is not easy to build a tree-structured lexicon using the word structure shown in Figure 2, because there should be many roots for a word. Hence, we make a generalized root by grouping PLUs which are changed from one phone, and the roots are shared by a group of words as shown in Figure 3. In this network, the words that have the same first three phones share a set of nodes, and the third phones of all words become the roots of the trees. In the figure, PRE_SET and POST_SET are sets of all PLUs which can be at the beginning and end of words respectively, and are shared by all words. HEAD_SET is a set of group of PLUs shared by the words whose second phones are the center phone of the PLU, first phone the left context, and third the right context. Let the first three phones of a word are *p, q, r,* and the first phone *p* can be changed to phones in the set P = {p_1, p_2, ..} according the context. Then the word shares the group of PLUs whose center phones are *q*, and the left and right contexts are the phones in the set P and r, respectively.

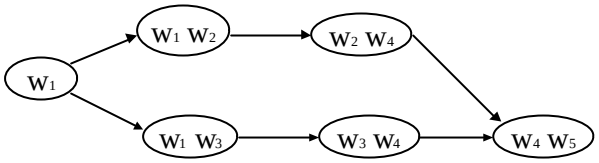
2.2. Finding the global best path

Many continuous speech recognizers employ multi pass strategy to reduce recognition time[3]. In multi pass algorithm, the errors produced in the first pass can hardly be recovered in later passes. If we use detailed information such as trigram in the first pass, the errors will be reduced, but it requires huge memory space or the result may not be global best path. Consider a typical example of the suboptimal usage of trigrams shown in Figure 4(a)[6]. If there are more than one transition to a node, than the Viterbi algorithm selects only one transition and discards the rest. This is based on the dynamic- programming principle: The best way through a particular, intermediate place is the best way to it from the starting place, followed by the best way from it to the goal. However, if we use a trigram grammar, the principle may not be valid. In the example, let us suppose that $w_1w_3w_4w_5$ is the best path, but we may choose the transition into the root node for word w_4 from the final states of w_2, and discarded transition w_3w_4, because transition $w_1w_2w_4$ is more likely trigram than $w_1w_3w_4$. The discarded path is not considered further, so we cannot get the best path. One of the conventional solutions to this problem is to expand the nodes according to the previous nodes as shown in Figure (b), but it requires a large memory space. Hence, the node expansion is usually used after the first pass when the number of candidate nodes is reduced to a reasonable size.

In this research, we solved this problem without node expansion by adapting token-passing algorithm [4]. In the token-passing algorithm, the state transition is modeled by moving tokens along the nodes, and the paths are kept in linked list. In our algorithm, we allowed multiple tokens at a state, so that multiple hypotheses can exist at one time. The algorithm does not select a token with highest score when two tokens meet at a state, but it delays the decision for one more word transition.



(a) Suboptimal usage of trigrams



(b) A conventional solution

Figure 4: The local optimal problem and a conventional solution

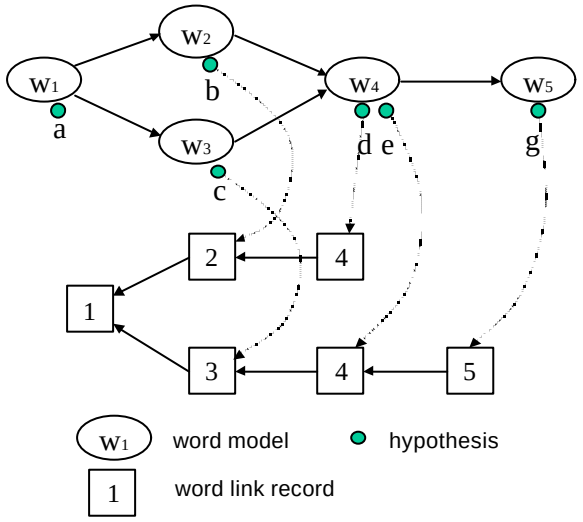


Figure 5: A search of global optimal path

In Figure 5, the best path up to the beginning node of word w_4 cannot be determined at the beginning of the word, but at the beginning of word w_5 if we use trigram. Hence, the tokens *d* and *e* are not discarded until they reach word w_5. We can choose a path at the beginning of word w_5 because w_2 and w_3 have no effect on deciding the best path after word w_5. The token *d* is discarded at the beginning of w_5 and the token *e* moves to w_5. By using this algorithm, we can acquire the global best path. We delayed the decision only one word when we use trigram, but we can get the best path by delaying n-2 words when we use n-gram in general, because a word can have effect only on the decision of up to after n-1 words.

3. EXPERIMENTS

3.1. The Conditions of Experiments

We use 80 hours of male and female speech data for training the acoustic models. The test utterances are read by seven speakers who are not included in the training. The script of the test data consists of 400 sentences selected from broadcast news script. We used 3.5 billion *eojeol*s of text data for language model training. The text includes newspaper text, broadcast news scripts, novels, and essays. The *eojeol*s in the text are divided to seven billion morphemes.

The speech data are sampled at 16 kHz 16 bits, a Hamming window with a width of 20 ms is applied at every 10 ms. Then a set of 12 LPC-derived melcepstral coefficients and their derivatives are computed. We use four feature streams - 12 cepstra, 12 delta cepstra, 12 delta delta cepstra, and power including its first and second derivatives. We use shared-state context dependent phone CDHMMs. Each left-to-right HMM model has three states. The number of base phones is 52, and the number of context-dependent phones is 6,000.

3.2. Comparing Morphemes and *Eojeol*s

We considered two vocabulary sets as shown in Table 1. The first is based on 20k morphemes; the second consists of 20k *eojeol*s. We used tagger to build a morpheme-based lexicon from the training text. There are 140k of different morphemes in the training text, which include spelling, spacing, and tagging errors. We choose the most frequent 20k morphemes in the set. As the table shows, out-of-vocabulary rate for the training text is 1.36%, so we can expect that the vocabulary set can cover most of general text. The OOV rate when using the *eojeol*s units is more than twice of that when using morpheme units. We can also get higher recognition rate by using morpheme units. For all of the experiments through out this paper, the parameters are set so that the recognition time is five times real time on Pentium III 500 MHz.

Table 1: Comparing Morphemes and *Eojeol*s.

Unit		**Morpheme**	***Eojeol***
OOV Rate	Training set	1.36 %	3.35 %
	Test set	1.01 %	2.08 %
Word error rate	Counted in morphemes	15.4%	18.1 %
	Counted in syllables	9.2%	9.7 %

3.3. The Effect of Between Word Modeling

We showed the effect of the proposed between-word phone variation lexicon by comparing the performance with and without the phone variation. In experiment I in table 2, no cross-word modeling is considered. In experiment II, cross-word modeling is considered, but the base phone of the beginning and ending of the words are not changed. In experiment III, the base phones of the beginning and ending phones of the words are changed according to the neighboring words. As the result shows, the relative error rate is decreased by 20.1% when we use the between-word phone variation lexicon. Figure 6 shows that the effect is higher when we use morpheme units than when we use *eojeol* units, because the length of the morphemes are shorter than *eojeol*s, and the shorter units are more likely to affected by contexts.

Table 2: The effect of between-word modeling of morpheme unit. Error rates are counted in morphemes.

	Cross-word modeling	Error (%)	Relative error reduction (%)	Correct (%)
I	No cross-word modeling	32.9 %	-	69.4 %
II	Cross-word modeling	19.4 %	20.8 %	82.8 %
III	Phone variation + Cross-word modeling	15.4 %	20.1 %	87.5 %

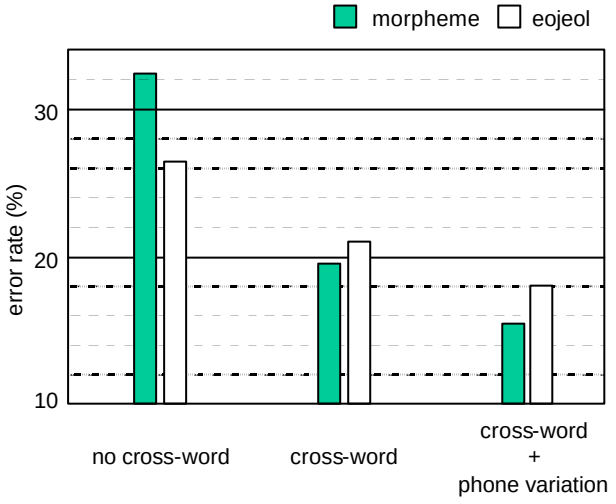


Figure 6: The effect of between-word modeling

3.4. The Effect of the Multiple-Token Algorithm

As explained in Section 2.2, we can get global best path when we use trigram in the first pass by allowing multiple tokens at a state and delaying the decision. As shown in Table 3, the relative error rate decrease is 52% when we allow 100 tokens at a state. The number of active tokens increases drastically in beam search, but we can limit the number of active tokens by using histogram pruning.

Table 3: The effect of the number of tokens in a state

Number of tokens in a state	Error(%)	Correct(%)
Limited to 1	32.3	70.9
Limited to 100	15.4	87.5

4. CONCLUSIONS

In this paper, we described some problems in recognizing Korean continuous speech and proposed some solutions. We use morphemes for the recognition word units, which are very short, so the influence of the between-word interference is a serious problem. We proposed between-word phone variation lexicon to solve the problem. By using the lexicon, we can describe phone variations at each end of words in graphical form, and we can build a recognition network by connecting them. The tree structured recognition network can be build by sharing first two phones of the words. We also proposed a one-pass algorithm which can get global best path using trigrams.

As the result, we can get higher recognition rate when we use morphemes as the recognition unit instead of *eojeol*s. The proposed between-word phone variation lexicon is found to be effective especially when we use morpheme units. We also observe that allowing multiple tokens at a state and delaying the decision can improve the performance of the recognizer.

5. REFERENCES

1. Ha-Jin Yu, Hoon Kim, Jae-Seung Choi, Joon-Mo Hong, Kew-Suh Park, Jong-Seok Lee, and Hee-Youn Lee, "Automatic Recognition of Korean Broadcast News Speech," The Fifth International Conference on Spoken Language Processing, ICSLP98, December 1998
2. Oh-Wook Kwon, Kyuwoong Hwang, and Jun Park, "Korean large vocabulary continuous speech recognition using pseudomorpheme units," Eurospeech99, pp. 483-486, September 1999.
3. R. Schwartz, Y.L. Chow, "The N-Best Algorithm: An efficient and exact procedure for finding the N most likely sentence hypotheses," Proc. of ICASSP, pp.81-84, 1990
4. S. J. Young, N.H. Russell, J.H.S Thornton, "Token passing: A simple conceptual model for connected speech recognition systems," Technical Report CUED/F-INFENG/TR38, Cambridge University Engineering Dept, 1989
5. J. L. Gauvain, L. F. Lamel, G. Adda, M. Adda-Decker, “Speaker-independent continous speech dictation,” *Speech Communication*, November 1994
6. M. K. Ravishankar, “Efficient Algorithms for Speech Recognition,” Carnegie Mellon University, Ph.D. thesis, 1996