Self-Organizing n-gram Model for Automatic Word Spacing

Seong-Bae Park **Yoon-Shik Tae** **Se-Young Park**
Department of Computer Engineering
Kyungpook National University
Daegu 702-701, Korea
{sbpark,ystae,sypark}@sejong.knu.ac.kr

Abstract

An automatic word spacing is one of the important tasks in Korean language processing and information retrieval. Since there are a number of confusing cases in word spacing of Korean, there are some mistakes in many texts including news articles. This paper presents a high-accurate method for automatic word spacing based on self-organizing n-gram model. This method is basically a variant of n-gram model, but achieves high accuracy by automatically adapting context size.

In order to find the optimal context size, the proposed method automatically increases the context size when the contextual distribution after increasing it dose not agree with that of the current context. It also decreases the context size when the distribution of reduced context is similar to that of the current context. This approach achieves high accuracy by considering higher dimensional data in case of necessity, and the increased computational cost are compensated by the reduced context size. The experimental results show that the self-organizing structure of n-gram model enhances the basic model.

1 Introduction

Even though Korean widely uses Chinese characters, the ideograms, it has a word spacing model unlike Chinese and Japanese. The word spacing of Korean, however, is not a simple task, though the basic rule for it is simple. The basic rule asserts that all content words should be spaced. However, there are a number of exceptions due to various *postpositions* and *endings*. For instance, it is difficult to distinguish some postpositions from incomplete nouns. Such exceptions induce many mistakes of word spacing even in news articles.

The problem of the inaccurate word spacing is that they are fatal in language processing and information retrieval. The incorrect word spacing would result in the incorrect morphological analysis. For instance, let us consider a famous Korean sentence: "아버지가방에들어가셨다." The true word spacing for this sentence is "아버지가#방에#들어가셨다." whose meaning is that my father entered the room. If the sentence is written as "어버지#가방에#들어가셨다.", it means that my father entered the bag, which is totally different from the original meaning. That is, since the morphological analysis is the first-step in most NLP applications, the sentences with incorrect word spacing must be corrected for their further processing. In addition, the wrong word spacing would result in the incorrect index for terms in information retrieval. Thus, correcting the sentences with incorrect word spacing is a critical task in Korean information processing.

One of the most simple and strong models for automatic word spacing is n-gram model. In spite of the advantages of the n-gram model, its problem should be also considered for achieving high performance. The main problem of the model is that it is usually modeled with fixed window size, n. The small value for n represents the narrow context in modeling, which results in poor performance in general. However, it is also difficult to increase n for better performance due to data sparseness. Since the corpus size is physically limited, it is highly possible that many n-grams which do not appear in the corpus exist in the real world.

Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, pages 633–640,
Sydney, July 2006.

The goal of this paper is to provide a new method for processing automatic word spacing with an n-gram model. The proposed method automatically adapts the window size n. That is, this method begins with a bigram model, and it shrinks to an unigram model when data sparseness occurs. It also grows up to a trigram, fourgram, and so on when it requires more specific information in determining word spacing. In a word, the proposed model organizes the windows size n online, and achieves high accuracy by removing both data sparseness and information lack.

The rest of the paper is organized as follows. Section 2 surveys the previous work on automatic word spacing and the smoothing methods for n-gram models. Section 3 describes the general way to automatic word spacing by an n-gram model, and Section 4 proposes a self-organizing n-gram model to overcome some drawbacks of n-gram models. Section 5 presents the experimental results. Finally, Section 6 draws conclusions.

2 Previous Work

Many previous work has explored the possibility of automatic word spacing. While most of them reported high accuracy, they can be categorized into two parts in methodology: *analytic approach* and *statistical approach*. The analytic approach is based on the results of morphological analysis. Kang used the fundamental morphological analysis techniques (Kang, 2000), and Kim et al. distinguished each word by the morphemic information of postpositions and endings (Kim et al., 1998). The main drawbacks of this approach are that (i) the analytic step is very complex, and (ii) it is expensive to construct and maintain the analytic knowledge.

In the other hand, the statistical approach extracts from corpora the probability that a space is put between two syllables. Since this approach can obtain the necessary information automatically, it does require neither the linguistic knowledge on syllable composition nor the costs for knowledge construction and maintenance. In addition, the fact that it does not use a morphological analyzer produces solid results even for unknown words. Many previous studies using corpora are based on *bigram* information. According to (Kang, 2004), the number of syllables used in modern Korean is about 10^4, which implies that the number of bigrams reaches 10^8. In order to obtain stable statistics for all bigrams, a great large volume of corpora will be required. If higher order n-gram is adopted for better accuracy, the volume of corpora required will be increased exponentially.

The main drawback of n-gram model is that it suffers from data sparseness however large the corpus is. That is, there are many n-grams of which frequency is zero. To avoid this problem, many smoothing techniques have been proposed for construction of n-gram models (Chen and Goodman, 1996). Most of them belongs to one of two categories. One is to pretend each n-gram occurs once more than it actually did (Mitchell, 1996). The other is to interpolate n-grams with lower dimensional data (Jelinek and Mercer, 1980; Katz, 1987). However, these methods artificially modify the original distribution of corpus. Thus, the final probabilities used in learning with n-grams are the ones distorted by a smoothing technique.

A maximum entropy model can be considered as another way to avoid zero probability in n-gram models (Rosenfeld, 1996). Instead of constructing separate models and then interpolate them, it builds a single, combined model to capture all the information provided by various knowledge sources. Even though a maximum entropy approach is simple, general, and strong, it is computationally very expensive. In addition, its performance is mainly dependent on the relevance of knowledge sources, since the prior knowledge on the target problem is very important (Park and Zhang, 2002). Thus, when prior knowledge is not clear and computational cost is an important factor, n-gram models are more suitable than a maximum entropy model.

Adapting features or contexts has been an important issue in language modeling (Siu and Ostendorf, 2000). In order to incorporate long-distance features into a language model, (Rosenfeld, 1996) adopted triggers, and (Mochihashi and Mastumoto, 2006) used a particle filter. However, these methods are restricted to a specific language model. Instead of long-distance features, some other researchers tried local context extension. For this purpose, (Schütze and Singer, 1994) adopted a variable memory Markov model proposed by (Ron et al., 1996), (Kim et al., 2003) applied selective extension of features to POS tagging, and (Dickinson and Meurers, 2005) expanded context of n-gram models to find errors in syntactic anno-

tation. In these methods, only neighbor words or features of the target n-grams became candidates to be added into the context. Since they required more information for better performance or detecting errors, only the context extension was considered.

3 Automatic Word Spacing by n-gram Model

The problem of automatic word spacing can be regarded as a binary classification task. Let a sentence be given as $S = w_1 w_2 \dots w_N$. If i.i.d. sampling is assumed, the data from this sentence are given as $D =< (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N) >$ where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{true, false\}$. In this representation, $\mathbf{x}_i$ is a contextual representation of a syllable w_i. If a space should be put after w_i, then y_i, the class of $\mathbf{x}_i$, is *true*. It is *false* otherwise. Therefore, the automatic word spacing is to estimate a function $f : \mathbb{R}^n \rightarrow \{true, false\}$. That is, our task is to determine whether a space should be put after a syllable w_i expressed as $\mathbf{x}_i$ with its context.

The probabilistic method is one of the strong and most widely used methods for estimating f. That is, for each w_i,

$$f(\mathbf{x}_i) = \underset{y_i \in \{true, false\}}{\arg\max} P(y_i|\mathbf{x}_i),$$

where $P(y_i|\mathbf{x}_i)$ is rewritten as

$$P(y_i|\mathbf{x}_i) = \frac{P(\mathbf{x}_i|y_i)P(y_i)}{P(\mathbf{x}_i)}.$$

Since $P(\mathbf{x}_i)$ is independent of finding the class of $\mathbf{x}_i$, $f(\mathbf{x}_i)$ is determined by multiplying $P(\mathbf{x}_i|y_i)$ and $P(y_i)$. That is,

$$f(\mathbf{x}_i) = \underset{y_i \in \{true, false\}}{\arg\max} P(\mathbf{x}_i|y_i)P(y_i).$$

In n-gram model, $\mathbf{x}_i$ is expressed with n neighbor syllables around w_i. Typically, n is taken to be two or three, corresponding to a bigram or trigram respectively. $\mathbf{x}_i$ corresponds to $w_{i-1}w_i$ when $n = 2$. In the same way, it is $w_{i-2}w_{i-1}w_i$ when $n = 3$. The simple and easy way to estimate $P(\mathbf{x}_i|y_i)$ is to use maximum likelihood estimate with a large corpus. For instance, consider the case $n = 2$. Then, the probability $P(\mathbf{x}_i|y_i)$ is represented as $P(w_{i-1}w_i|y_i)$, and is computed by

$$\begin{aligned} P(w_{i-1}w_i|y_i) &= \frac{P(w_{i-1}w_i \& y_i)}{P(y_i)} \\ &= \frac{C(w_{i-1}w_i \& y_i)}{C(y_i)}, \end{aligned} \quad (1)$$

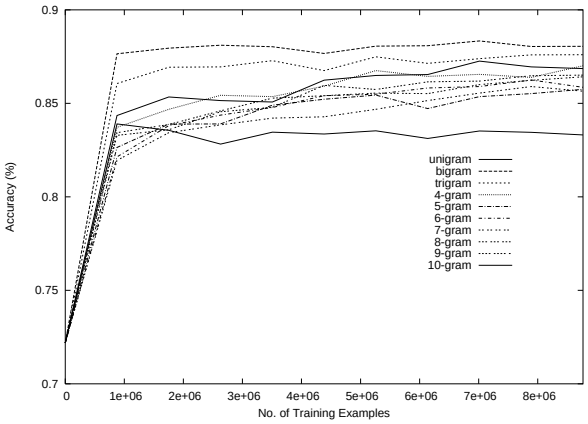

Figure 1: The performance of n-gram models according to the values of n in automatic word spacing.

where C is a counting function.

Determining the context size, the value of n, in n-gram models is closely related with the corpus size. The larger is n, the larger corpus is required to avoid data sparseness. In contrast, though low-order n-grams do not suffer from data sparseness severely, they do not reflect the language characteristics well, either. Typically researchers have used $n = 2$ or $n = 3$, and achieved high performance in many tasks (Bengio et al., 2003). Figure 1 supports that bigram and trigram outperform low-order ($n = 1$) and high-order ($n \geq 4$) n-grams in automatic word spacing. All the experimental settings for this figure follows those in Section 5. In this figure, bigram model shows the best accuracy and trigram achieves the second best, whereas unigram model results in the worst accuracy. Since the bigram model is best, a self-organizing n-gram model explained below starts from bigram.

4 Self-Organizing n-gram Model

To tackle the problem of fixed window size in n-gram models, we propose a self-organizing structure for them.

4.1 Expanding n-grams

When n-grams are compared with $(n+1)$-grams, their performance in many tasks is lower than that of $(n+1)$-grams (Charniak, 1993). Simultaneously the computational cost for $(n+1)$-grams is far higher than that for n-grams. Thus, it can be justified to use $(n+1)$-grams instead of n-

Function *HowLargeExpand*($\mathbf{x}_n$)
Input: $\mathbf{x}_n$: n-grams
Output: an integer for expanding size

1. Retrieve $(n+1)$-grams $\mathbf{x}_{n+1}$ for $\mathbf{x}_n$.
2. Compute
$$D = KL(P(y|\mathbf{x}_n)||P(y|\mathbf{x}_{n+1})).$$
3. **If** $D < \theta_{\text{EXP}}$ **Then return** 0.
4. **return** *HowLargeExpand*($\mathbf{x}_{n+1}$) + 1.

Figure 2: A function that determines how large a window size should be.

grams only when higher performance is expected. In other words, $(n+1)$-grams should be different from n-grams. Otherwise, the performance would not be different. Since our task is attempted with a probabilistic method, the difference can be measured with conditional distributions. If the conditional distributions of n-grams and $(n+1)$-grams are similar each other, there is no reason to adopt $(n+1)$-grams.

Let $P(y_i|\mathbf{x}_{ni})$ be a class-conditional probability by n-grams and $P(y_i|\mathbf{x}_{(n+1)i})$ that by $(n+1)$-grams. Then, the difference $D(n, n+1)$ between them is measured by Kullback-Leibler divergence. That is,

$$D(n, n+1) \quad = \quad KL(P(y_i|\mathbf{x}_{ni})||P(y_i|\mathbf{x}_{(n+1)i}),$$

which is computed by

$$\sum_{z \in \{true, false\}} p(z|\mathbf{x}_{ni}) \log \frac{p(z|\mathbf{x}_{ni})}{p(z|\mathbf{x}_{(n+1)i})}. \qquad (2)$$

$D(n, n+1)$ that is larger than a predefined threshold θ_{EXP} implies that $P(y_i|\mathbf{x}_{(n+1)i})$ is different from $P(y_i|\mathbf{x}_{ni})$. In this case, $(n+1)$-grams is used instead of n-grams.

Figure 2 depicts an algorithm that determines how large n-grams should be used. It recursively finds the optimal expanding window size. For instance, let bigrams ($n = 2$) be used at first. When the difference between bigrams and trigrams ($n = 3$) is larger than θ_{EXP}, that between trigrams and fourgrams ($n = 4$) is checked again. If it is less than θ_{EXP}, then this function returns 1 and trigrams are used instead of bigrams. Otherwise, it considers higher n-grams again.

Function *HowSmallShrink*($\mathbf{x}_n$)
Input: $\mathbf{x}_n$: n-grams
Output: an integer for shrinking size

1. **If** $n < 1$ **Then return** 0.
2. Retrieve $(n-1)$-grams $\mathbf{x}_{n-1}$ for $\mathbf{x}_n$.
3. Compute
$$D = KL(P(y|\mathbf{x}_n)||P(y|\mathbf{x}_{n-1})).$$
4. **If** $D > \theta_{\text{SHR}}$ **Then return** 0.
5. **return** *HowSmallShrink*($\mathbf{x}_{n-1}$) - 1.

Figure 3: A function that determines how small a window size should be used.

4.2 Shrinking n-grams

Shrinking n-grams is accomplished in the direction opposite to expanding n-grams. After comparing n-grams with $(n-1)$-grams, $(n-1)$-grams are used instead of n-grams only when they are similar enough. The difference $D(n, n-1)$ between n-grams and $(n-1)$-grams is, once again, measured by Kullback-Leibler divergence. That is,

$$D(n, n-1) \quad = \quad KL(P(y_i|\mathbf{x}_{ni})||P(y_i|\mathbf{x}_{(n-1)i}).$$

If $D(n, n-1)$ is smaller than another predefined threshold θ_{SHR}, then $(n-1)$-grams are used instead of n-grams.

Figure 3 shows an algorithm which determines how deeply the shrinking is occurred. The main stream of this algorithm is equivalent to that in Figure 2. It also recursively finds the optimal shrinking window size, but can not be further reduced when the current model is an unigram.

The merit of shrinking n-grams is that it can construct a model with a lower dimensionality. Since the maximum likelihood estimate is used in calculating probabilities, this helps obtaining stable probabilities. According to the well-known *curse of dimensionality*, the data density required is reduced exponentially by reducing dimensions. Thus, if the lower dimensional model is not different so much from the higher dimensional one, it is highly possible that the probabilities from lower dimensional space are more stable than those from higher dimensional space.

Function *ChangingWindowSize*($\mathbf{x}_n$)
Input: $\mathbf{x}_n$: n-grams
Output: an integer for changing window size

1. **Set** `exp` := *HowLargeExpand*($\mathbf{x}_n$).
2. **If** `exp` > 0 **Then return** `exp`.
3. **Set** `shr` := *HowSmallShrink*($\mathbf{x}_n$).
4. **If** `shr` < 0 **Then return** `shr`.
5. **return** 0.

Figure 4: A function that determines the changing window size of n-grams.

4.3 Overall Self-Organizing Structure

For a given i.i.d. sample $\mathbf{x}_i$, there are three possibilities on changing n-grams. First one is not to change n-grams. It is obvious when n-grams are not changed. This occurs when both $D(n, n+1) \geq \theta_{\text{EXP}}$ and $D(n, n-1) \leq \theta_{\text{SHR}}$ are met. This is when the expanding results in too similar distribution to that of the current n-grams and the distribution after shrinking is too different from that of the current n-grams.

The remaining possibilities are then *expanding* and *shrinking*. The application order between them can affect the performance of the proposed method. In this paper, an expanding is checked prior to a shrinking as shown in Figure 4. The function *ChangingWindowSize* first calls *HowLargeExpand*. The non-zero return value of *HowLargeExpand* implies that the window size of the current n-grams should be enlarged. Otherwise, *ChangingWindowSize* checks if the window size should be shrinked by calling *HowSmallShrink*. If *HowSmallShrink* returns a negative integer, the window size should be shrinked to (n + `shr`). If both functions return zero, the window size should not be changed.

The reason why *HowLargeExpand* is called prior to *HowSmallShrink* is that the expanded n-grams handle more specific data. $(n+1)$-grams, in general, help obtaining higher accuracy than n-grams, since $(n+1)$-gram data are more specific than n-gram ones. However, it is time-consuming to consider higher-order data, since the number of kinds of data increases. The time increased due to expanding is compensated by shrinking. After shrinking, only lower-oder data are considered, and then processing time for them decreases.

4.4 Sequence Tagging

Since natural language sentences are sequential as their nature, the word spacing can be considered as a special POS tagging task (Lee et al., 2002) for which a hidden Markov model is usually adopted. The best sequence of word spacing for the sentence is defined as

$$\begin{aligned} y^*_{1,N} &= \arg\max_{y_{1,N}} P(y_{1,N}|\mathbf{x}_{1,N}) \\ &= \arg\max_{y_{1,N}} \frac{P(\mathbf{x}_{1,N}|y_{1,N})P(y_{1,N})}{P(\mathbf{x}_{1,N})} \\ &= \arg\max_{y_{1,N}} P(\mathbf{x}_{1,N}|y_{1,N})P(y_{1,N}) \end{aligned}$$

by where N is a sentence length.

If we assume that the syllables are independent of each other, $P(\mathbf{x}_{1,N}|y_{1,N})$ is given by

$$P(\mathbf{x}_{1,N}|y_{1,N}) = \prod_{i=1}^{N} P(\mathbf{x}_i|y_i),$$

which can be computed using Equation (1). In addition, by Markov assumption, the probability of a current tag y_i conditionally depends on only the previous k tags. That is,

$$P(y_{1,N}) = \prod_{i=1}^{N} P(y_i|y_{i-k,i-1}).$$

Thus, the best sequence is determined by

$$y^*_{1,N} = \arg\max_{y_{1,N}} \prod_{i=1}^{N} P(\mathbf{x}_i|y_i) \cdot P(y_i|y_{i-k,i-1}). \quad (3)$$

Since this equation follows Markov assumption, the best sequence is found by applying the Viterbi algorithm.

5 Experiments

5.1 Data Set

The data set used in this paper is the `HANTEC` corpora version 2.0 distributed by KISTI[1]. From this corpus, we extracted only the HKIB94 part which consists of 22,000 news articles in 1994 from Hankook Ilbo. The reason why HKIB94 is chosen is that the word spacing of news articles is relatively more accurate than other texts. Even though this data set is composed of totally 12,523,688 Korean syllables, the number of unique syllables is just

[1] http://www.kisti.re.kr

Methods	Accuracy (%)
baseline	72.19
bigram	88.34
trigram	87.59
self-organizing bigram	**91.31**
decision tree	88.68
support vector machine	89.10

Table 1: The experimental results of various methods for automatic word spacing.

2,037 after removing all special symbols, digits, and English alphabets.

The data set is divided into three parts: *training* (70%), *held-out* (20%), and *test* (10%). The held-out set is used only to estimate θ_{EXP} and θ_{SHR}. The number of instances in the training set is 8,766,578, that in the held-out set is 2,504,739, and that in test set is 1,252,371. Among the 1,252,371 test cases, the number of positive instances is 348,278, and that of negative instances is 904,093. Since about 72% of test cases are negative, this is the baseline of the automatic word spacing.

5.2 Experimental Results

To evaluate the performance of the proposed method, two well-known machine learning algorithms are compared together. The tested machine learning algorithms are (i) decision tree and (ii) support vector machines. We use C4.5 release 8 (Quinlan, 1993) for decision tree induction and SVM^{light} (Joachims, 1998) for support vector machines. For all experiments with decision trees and support vector machines, the context size is set to two since the bigram shows the best performance in Figure 1.

Table 1 gives the experimental results of various methods including machine learning algorithms and self-organizing n-gram model. The 'self-organizing bigram' in this table is the one proposed in this paper. The normal n-grams achieve an accuracy of around 88%, while decision tree and support vector machine produce that of around 89%. The self-organizing n-gram model achieves 91.31%. The accuracy improvement by the self-organizing n-gram model is about 19% over the baseline, about 3% over the normal n-gram model, and 2% over decision trees and support vector machines.

In order to organize the context size for n-grams online, two operations of *expanding* and *shrinking* were proposed. Table 2 shows how much the number of errors is affected by their application order. The number of errors made by expanding first is 108,831 while that by shrinking first is 114,343. That is, if shrinking is applied ahead of expanding, 5,512 additional errors are made. Thus, it is clear that expanding should be considered first.

Order	No. of Errors
Expanding then Shrinking	108,831
Shrinking then Expanding	114,343

Table 2: The number of errors caused by the application order of context expanding and shrinking.

The errors by expanding can be explained with two reasons: (i) the expression power of the model and (ii) data sparseness. Since Korean is a partially-free word order language and the omission of words are very frequent, n-gram model that captures local information could not express the target task sufficiently. In addition, the class-conditional distribution after expanding could be very different from that before expanding due to data sparseness. In such cases, the expanding should not be applied since the distribution after expanding is not trustworthy. However, only the difference between two distributions is considered in the proposed method, and the errors could be made by data sparseness.

Figure 5 shows that the number of training instances does not matter in computing probabilities of n-grams. Even though the accuracy increases slightly, the accuracy difference after 900,000 instances is not significant. It implies that the errors made by the proposed method is not from the lack of training instance but from the lack of its expression power for the target task. This result also complies with Figure 1.

5.3 Effect of Right Context

All the experiments above considered left context only. However, Kang reported that the probabilistic model using both left and right context outperforms the one that uses left context only (Kang, 2004). In his work, the word spacing probability $P(w_i, w_{i+1})$ between two adjacent syllables w_i and w_{i+1} is given as

$$\begin{aligned} P(w_i, w_{i+1}) &= 0.25 \times P_L(w_{i-1}, w_i) \\ &\quad 0.5 \times P_M(w_i, w_{i+1}) \\ &\quad 0.25 \times P_R(w_{i+1}, w_{i+2}), \end{aligned} \tag{4}$$

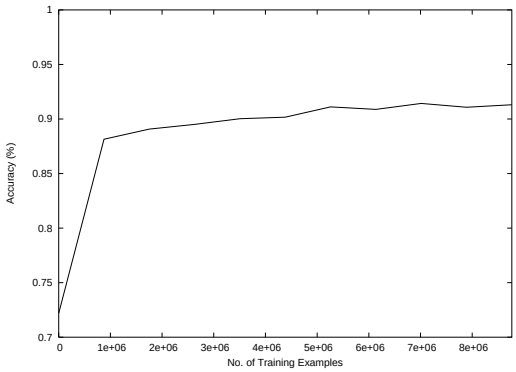

Figure 5: The effect of the number of training examples in the self-organizing n-gram model.

Context	Accuracy (%)
Left Context Only	91.31
Right Context Only	88.26
Both Contexts	**92.54**

Table 3: The effect of using both left and right context.

where P_R, P_M, P_L are computed respectively based on the syllable frequency.

In order to reflect the idea of bidirectional context in the proposed model, the model is enhanced by modifying $P(w_{i-1}w_i|y_i)$ in Equation (1). That is, the likelihood of $P(w_{i-1}w_i|y_i)$ is expanded to be

$$\begin{aligned}\tilde{P}(w_{i-1}w_i|y_i) &= \alpha_1 \times P(w_{i-2}w_{i-1}|y_i) \\ &\quad \alpha_2 \times P(w_{i-1}w_i|y_i) \\ &\quad \alpha_3 \times P(w_i, w_{i+1}|y_i).\end{aligned}$$

Since the coefficients of Equation (4) were determined arbitrarily (Kang, 2004), they are replaced with parameters α_i of which values are determined using a held-out data.

The change of accuracy by the context is shown in Table 3. When only the right context is used, the accuracy gets 88.26% which is worse than the left context only. That is, the original n-gram is a relatively good model. However, when both left and right context are used, the accuracy becomes 92.54%. The accuracy improvement by using additional right context is 1.23%. This results coincide with the previous report (Lee et al., 2002). The α_i's to achieve this accuracy are $\alpha_1 = 0.5, \alpha_2 = 0.3$, and $\alpha_3 = 0.2$.

Method	Accuracy(%)
Normal HMM	92.37
Self-Organizing HMM	**94.71**

Table 4: The effect of considering a tag sequence.

5.4 Effect of Considering Tag Sequence

The state-of-the-art performance on Korean word spacing is to use the hidden Markov model. According to the previous work (Lee et al., 2002), the hidden Markov model shows the best performance when it sees two previous tags and two previous syllables.

For the simplicity in the experiments, the value for k in Equation (3) is set to be one. The performance comparison between normal HMM and the proposed method is given in Table 4. The proposed method considers the various number of previous syllables, whereas the normal HMM has the fixed context. Thus, the proposed method in Table 4 is specified as 'self-organizing HMM.' The accuracy of the self-organizing HMM is 94.71%, while that of the normal HMM is just 92.37%. Even though the normal HMM considers more previous tags ($k = 2$), the accuracy of the self-organizing model is 2.34% higher than that of the normal HMM. Therefore, the proposed method that considers the sequence of word spacing tags achieves higher accuracy than any other methods reported ever.

6 Conclusions

In this paper we have proposed a new method to learn word spacing in Korean by adaptively organizing context size. Our method is based on the simple n-gram model, but the context size n is changed as needed. When the increased context is much different from the current one, the context size is increased. In the same way, the context is decreased, if the decreased context is not so much different from the current one. The benefits of this method are that it can consider wider context by increasing context size as required, and save the computational cost due to the reduced context.

The experiments on HANTEC corpora showed that the proposed method improves the accuracy of the trigram model by 3.72%. Even compared with some well-known machine learning algorithms, it achieved the improvement of 2.63% over decision trees and 2.21% over support vector machines. In addition, we showed two ways for improving the

proposed method: considering *right context* and *word spacing sequence*. By considering left and right context at the same time, the accuracy is improved by 1.23%, and the consideration of word spacing sequence gives the accuracy improvement of 2.34%.

The n-gram model is one of the most widely used methods in natural language processing and information retrieval. Especially, it is one of the successful language models, which is a key technique in language and speech processing. Therefore, the proposed method can be applied to not only word spacing but also many other tasks. Even though word spacing is one of the important tasks in Korean information processing, it is just a simple task in many other languages such as English, German, and French. However, due to its generality, the importance of the proposed method yet does hold in such languages.

Acknowledgements

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (KRF-2005-202-D00465).

References

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. 2003. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, Vol. 3, pp. 1137–1155.

E. Charniak. 1993. *Statistical Language Learning*. MIT Press.

S. Chen and J. Goodman. 1996. An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 310–318.

M. Dickinson and W. Meurers. 2005. Detecting Errors in Discontinuous Structural Annotation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pp. 322–329.

F. Jelinek and R. Mercer. 1980. Interpolated Estimation of Markov Source Parameters from Sparse Data. In *Proceedings of the Workshop on Pattern Recognition in Practice*.

T. Joachims. 1998. *Making Large-Scale SVM Learning Practical*. LS8, Universität Dortmund.

S.-S. Kang, 2000. Eojeol-Block Bidirectional Algorithm for Automatic Word Spacing of Hangul Sentences. *Journal of KISS*, Vol. 27, No. 4, pp. 441–447. (*in Korean*)

S.-S. Kang. 2004. Improvement of Automatic Word Segmentation of Korean by Simplifying Syllable Bigram. In *Proceedings of the 15th Conference on Korean Language and Information Processing*, pp. 227–231. (*in Korean*)

S. Katz. 1987. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*. Vol. 35, No. 3, pp. 400–401.

K.-S. Kim, H.-J. Lee, and S.-J. Lee. 1998. Three-Stage Spacing System for Korean in Sentence with No Word Boundaries. *Journal of KISS*, Vol. 25, No. 12, pp. 1838–1844. (*in Korean*)

J.-D. Kim, H.-C. Rim, and J. Tsujii. 2003. Self-Organizing Markov Models and Their Application to Part-of-Speech Tagging. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pp. 296–302.

D.-G. Lee, S.-Z. Lee, H.-C. Rim, and H.-S. Lim, 2002. Automatic Word Spacing Using Hidden Markov Model for Refining Korean Text Corpora. In *Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization*, pp. 51–57.

T. Mitchell. 1997. *Machine Learning*. McGraw Hill.

D. Mochihashi and Y. Matsumoto. 2006. Context as Filtering. *Advances in Neural Information Processing Systems 18*, pp. 907–914.

S.-B. Park and B.-T. Zhang. 2002. A Boosted Maximum Entropy Model for Learning Text Chunking. In *Proceedings of the 19th International Conference on Machine Learning*, pp. 482–489.

R. Quinlan. 1993. *C4.5: Program for Machine Learning*. Morgan Kaufmann Publishers.

D. Ron, Y. Singer, and N. Tishby. 1996. The Power of Amnesia: Learning Probabilistic Automata with Variable Memory Length. *Machine Learning*, Vol. 25, No. 2, pp. 117–149.

R. Rosenfeld. 1996. A Maximum Entropy Approach to Adaptive Statistical Language Modeling. *Computer, Speech and Language*, Vol. 10, pp. 187– 228.

H. Schütze and Y. Singer. 1994. Part-of-Speech Tagging Using a Variable Memory Markov Model. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pp. 181–187.

M. Siu and M. Ostendorf. 2000. Variable N-Grams and Extensions for Conversational Speech Language Modeling. *IEEE Transactions on Speech and Audio Processing*, Vol. 8, No. 1, pp. 63–75.