In smart homes, information appliances interact with residents via social network services. To capture residents' intentions, the information appliances should analyze short text messages entered typically through small mobile devices. However, most information appliances have hardware constraints such as small memory, limited battery capacity, and restricted processing power. Therefore, it is not easy to embed intelligent applications based on natural language processing (NLP) techniques, which traditionally require large memory and high-end processing power, into information appliances. To overcome this problem, lightweight NLP modules should be implemented. We propose an automatic word spacing system, the first step module of NLP for many languages with their own word spacing rules, which is designed for information appliances with limited hardware resources. The proposed system consists of a word spacing dictionary and a pattern-matching module. When a sentence is entered, the pattern-matching module inserts spaces by simply looking up the word spacing dictionary in a back-off manner. In comparative experiments with previous models, the proposed method showed low memory usage (0.79 MB) and high character-unit accuracy (0.9460) without requiring complex arithmetical computations. On the basis of these experiments, we conclude that the proposed system is suitable for information appliances with many hardware limitations.

1. Introduction

Household appliances are being rapidly evolved into information appliances (e.g., smartphones and personal digital assistants (PDAs)), which are usable for the purposes of computing, telecommunicating, reproducing, and presenting encoded information in myriad forms and applications. With the appearance of social networking services such as Twitter (http://www.twitter.com) and Facebook (http://www.facebook.com), the information appliances have begun to provide social networking functionality [1]. In the near future, smart homes will offer social networking to their residents or their appliances. Some information appliances will be registered as users’ friends. Then, the registered information appliances will execute various commands that are received from users via social networking services. To realize such smart homes, information appliances should understand users’ natural language commands which are in the form of short text messages (e.g., tweets and recognized speech inputs) [2]. Natural language processing (NLP) techniques can be used to convert statements of a natural language into the statements of a highly reduced topic-oriented language, a formal language that information appliances can understand [3]. Unfortunately, hardware limitations of information appliances must be considered for using NLP techniques in smart home environments. Many information appliances have limited input and output capabilities, limited bandwidth, limited memory, limited battery capacity, and restricted processing power. In spite of these hardware limitations, lightweight NLP applications (e.g., a tweet clustering system [4], an appointment extraction system [5], a news summarization system (http://summly.com), and a translation assistance system [6]) and NLP-based smart home applications (e.g., a smart home security system [2] and a smart home social networking system [1]) have been successfully developed. To implement these NLP-based applications, words in a sentence should first be tokenized. Tokenization is not difficult in English with strict rules of word spacing. However, some agglutinative languages such as Japanese and Turkish do not have any word spacing rules,
whereas some other agglutinative languages like Korean have very complex word spacing rules. In addition to these complex word spacing rules, Koreans often ignore word spaces, especially in information appliances such as smartphones and PDAs, in order to avoid the inconvenience of pressing special buttons. In the case of both languages without word spacing rules and languages with complex word spacing rules, automatic word spacing may be the first step toward the implementation of user interface systems based on NLP techniques. Unfortunately, the hardware limitations of information appliances make it difficult to implement statistical word spacing models that necessitate complex computations. Although many high-performance information appliances are being developed at present, lightweight word spacing models are still needed to efficiently realize NLP applications because high-level linguistic models (e.g., morphological analysis, named entity recognition, syntactic parsing, and so on) require large memory and high-performance processor. To overcome these constraints, we propose a reliable method to perform automatic word spacing by simply looking up a word spacing dictionary containing mixed character n-grams. The current version of the proposed method operates in Korean, but switching to a different language will not be difficult because the method uses character n-grams that can be easily extracted from a raw corpus. However, if the target language is significantly different from Korean, the size of the context (i.e., \( n \) in character n-grams) can be adjusted to the characteristics of the target language.

This paper is organized as follows. In Section 2, we review previous works on automatic word spacing systems. In Section 3, we present a pattern-matching model for automatic word spacing. In Section 4, we analyze the results of our experiments. Finally, we draw conclusions in Section 5.

2. Previous Works

The previous approaches on automatic word spacing can be classified into two groups: rule-based models [7, 8] and statistical models [9–14]. Rule-based models recognize words in a sentence from left to right or from right to left by using morphological information, such as functional word lists and language-dependent word patterns. These models then insert spaces between words using heuristic rules, such as longest-match-preference rules, shortest-match-preference rules, and error patterns. Although rule-based models are simple and clear, they have some drawbacks. First, they need handcrafted linguistic knowledge, whose construction and maintenance are considerably expensive. Second, they cannot effectively handle unknown word patterns because they use lexical levels of predefined patterns. For example, a user can command an information appliance to check whether his/her father is in a room by using the sentence without word spaces, “appa ga-bang-e-it-eo?” (Is the father in a room?) via social networking. To aid the readability of examples in this paper, we use Romanized Korean characters called Hangeul and insert hyphen symbols between Korean characters called eumjeol’s. In this example, “ga-bang (bag)” can be considered as a word by longest-match-preference rules because it is longer than “bang (room).” As a result, the sentence may be understood as “appa ga-bang-e-it-eo?” (Is the father in a bag?) Statistical models extract the space statistics for adjacent characters from a large raw corpus. They then decide whether to place a space between two adjacent characters by using these space statistics. Kang [10], Lee and Kim [12], Lee et al. [13], and Lee et al. [14] proposed state-of-the-art statistical models on Korean word spacing. Kang [10] proposed a traditional n-gram model that uses the statistics of character bigrams as context information. Reference [13] is a first-order hidden Markov model (HMM) that uses character trigrams as observation information. Reference [14] is a second-order HMM that observes two earlier labels and characters each. Furthermore, [12] is a model using structural support vector machines (SVMs) to relax the independence assumptions required by HMMs. Cohen et al. [9] proposed a word boundary detection model based on a change of entropy values. Since statistical models can automatically obtain the necessary information, they do not require the construction and maintenance of linguistic knowledge. In addition, they are generally more robust against unknown word patterns than rule-based models. In the example above, “appa ga bang (father room)” more frequently occurs in a Korean corpus than “appa ga-bang (father bag).” Therefore, most statistical models will correctly insert a space between “ga” and “bang” and the sentence will be understood as “appa ga-bang-e-it-eo? (Is the father in a room?)” However, in information appliances with a small main memory, it is impractical to use earlier statistical models based on character bigrams or longer combinations because they require a large amount of memory for loading the statistical information [15–17]. The number of characters that are often used in modern Korean is about \( 10^4 \), which implies that the number of character bigrams reaches \( 10^8 \). Assuming that the frequency of each character bigram is represented by two bytes, information appliances should equip about 200 MB (megabytes) of memory. To overcome this problem, Park et al. [18] proposed a combined model of rule-based and memory-based learning. Song and Kim [15] combined character unigram statistics and error-correction rules of character bigrams or longer combinations. Kim et al. [16, 17] proposed a modified HMM that uses probabilities of word spacing patterns as observation probabilities. These models showed high accuracy despite low memory consumption. Park et al. [19] proposed a self-organizing HMM that automatically adapts the context size from a character unigram to a character bigram. However, they demand complex arithmetical computations to find optimal paths because they are based on statistical sequence tagging methods. Although these computations are not burdensome for desktop computers, they can be a critical drawback in the case of information appliances with low computational power.

3. Back-Off Pattern-Matching Model for Automatic WordSpacing

The proposed model consists of two submodels: one is a model to construct a word spacing dictionary that contains
lexical clues useful for automatic word spacing and the other is a model to perform automatic word spacing by matching input sentences against entities in the word spacing dictionary.

Figure 1 shows the overall architecture of the proposed model.

3.1. Pattern-Matching Based on a Back-Off Language Model.

To insert spaces between words, we propose a compact pattern-matching method based on a back-off language model without the requirement of any arithmetical computations except for dictionary retrieval. Let $S_n$ denote an input sentence that consists of a sequence of $n$ characters $w_1, w_2, \ldots, w_n$, and let $w_0$ denote a symbol that represents the beginning of a sentence. By moving character-by-character from the beginning to the end of $S_n$, the proposed method extracts character trigrams. If the character trigram $w_{i-1}w_iw_{i+1}$ exists in a word spacing dictionary, the proposed method inserts a space between the bigram $w_{i-1}w_i$ and $w_iw_{i+1}$. If not, the proposed method checks whether the character bigram $w_iw_{i+1}$ exists in the word spacing dictionary. Then, it inserts a space between $w_i$ and $w_{i+1}$. If the character bigram does not exist in the word spacing dictionary, the proposed method checks whether both the character unigrams $w_iS$ and $Sw_{i+1}$ exist in the word spacing dictionary. Then, it inserts a space next to $w_i$. Algorithm 1 shows the algorithm of the proposed pattern-matching method executed in C language.

3.2. Construction of a Word Spacing Dictionary Based on Character $n$-Grams.

Table 1 shows the number of unique $n$-grams extracted from the Korean corpus of the 21st Century Sejong Project (337,201 sentences; http://www.sejong.or.kr/ eindex.php).

In Table 1, the parenthesized numbers are the numbers of unique word $n$-grams extracted from the same corpus. As shown in Table 1, if each Korean $n$-gram set is loaded into main memory, approximately 9.18 MB (1,529,260 × 6 B), 1.06 MB (264,592 × 4 B), and 0.012 MB are required for character trigram, bigram, and unigram sets, respectively, because a Korean character is represented by a two-byte code. If we assume that the upper limit of memory usage for automatic word spacing in information appliances is approximately 1 MB, a language model based on character trigrams will be too large to be used in information appliances. In information theory [20], the information quantity of the pattern $x$ is calculated as $-\log_2 p(x)$. The information quantity means that there is an inverse relationship between the frequency of a pattern in the training data and its information, with more frequent patterns carrying less information. Therefore, this is problematic because trigrams contain considerably more linguistic information than either bigrams or unigrams. To resolve this problem, we construct a word spacing dictionary that consists of character trigrams or shorter combinations. First, we extract all character trigrams from large corpora. Second, if character bigrams that comprise a part of character trigrams have sufficient quantity of information to resolve word spacing problems, we replace the character trigrams with the character bigrams. Third, if pairs of unigrams in character bigrams have adequate quantity of information to resolve word spacing problems, we replace the character bigrams with the character unigrams. Lastly, if there are character unigrams with high probabilities for word spacing, we add the character unigrams to the word spacing dictionary.

Figure 2 shows the construction process of the word spacing dictionary.

In Figure 2, $w_i$ represents the $i$th character in a sentence and $S$ represents a space. Then, $w_iS$, $w_iSw_{i+1}$ and $w_{i-1}w_iSw_{i+1}$ represent the case that there is a space after $w_i$, a space between $w_i$ and $w_{i+1}$, and a space between the bigram $w_{i-1}w_i$ and $w_{i+1}$, respectively. Further, $F(X)$ and $P(X)$ represent frequency and probability, respectively, of the $n$-gram $X$. To compute $F(X)$, we count the number of $X$’s occurring in the Korean corpus. To compute $P(X)$, we divide $F(X)$ by the number of all $n$-grams (e.g., all bigrams if $X$ is a bigram) in the Korean corpus. The minimum frequency $\alpha$ and the minimum probability $\beta$ were set on an experimental basis. As shown in Figure 2, when the character bigram $w_{i-1}w_i$ (i.e., the last two characters of character trigram $w_{i-1}w_iw_{i+1}$) does not satisfy the preconditions $F(w_iw_{i+1}) \geq \alpha$ and $P(w_iSw_{i+1}) \geq \beta$, the character trigram $w_{i-1}w_iw_{i+1}$ is added into the word spacing dictionary for inserting a space between $w_{i-1}w_i$ and $w_{i+1}$. Similarly, when a pair of character unigrams, $w_i$ and $w_{i+1}$, does not satisfy the preconditions $|F(w_i) + F(w_{i+1})| / 2 \geq \alpha$ (i.e., when the average frequency is greater than or equal to $\alpha$) and $P(w_iS) \geq \beta$ (i.e., when the product of independent probabilities is greater than or equal to $\beta$), the character bigram $w_iw_{i+1}$ is added into the word spacing dictionary for inserting a space between $w_i$ and $w_{i+1}$. Finally, the character unigram $w_i$ is added into the word spacing dictionary if it satisfies the relatively difficult preconditions $F(w_i) \geq \alpha$ and $P(w_iS)^2 \geq \beta$. Figure 3 shows the construction process of the word spacing dictionary for a given example.

Let $N$ denote the number of word spaces in the Korean corpus. In the scan procedure for extracting character $n$-grams, the proposed method has the time complexity $3 \times O(N)$ (i.e., the sum of $O(N)$ in scanning character trigrams, $O(N)$ in scanning character bigrams, and $O(N)$ in scanning character unigrams). In the search and update procedure for calculating frequencies and probabilities of character $n$-grams, the proposed method has the time complexity $O(N \log(N))$ (i.e., the sum of $O(N \log(N))$ in binary search of character trigrams, $O(N \log(N))$ in binary search of character bigrams, and $O(N \log(N))$ in binary search of character unigrams). Therefore, the construction process of the word spacing dictionary has the time complexity of $O(N \log(N))$ (i.e., $3 \times O(N) + 3 \times O(N \log(N)) = O(N \log(N))$).

4. Evaluation

4.1. Datasets and Experimental Settings.

To experiment with the proposed method in terms of various criteria, we used

| Table 1: Statistic of character $n$-grams. |
|------------------------------------------|
| $n$-grams | Trigram | Bigram | Unigram |
|-----------|---------|--------|---------|
| Number of unique | 1,529,260 | 264,592 | 6,049 |
| Character 1 | (6,733,216) | (2,311,928) | (196,782) |
Algorithm 1: Pattern-matching algorithm based on a back-off method executed in C language.
Figure 2: Construction process of the word spacing dictionary.

five kinds of evaluation data: SEJONG, ETRI-POS, ETRI-PARSE, HANTEC-HKIB94, and CLMET. In order to independently evaluate various performance criteria (e.g., memory usage, processing time, average accuracy, and accuracy variation), we used SEJONG as training data for constructing a word space dictionary, ETRI-PARSE as held-out data for parameter tuning, and ETRI-POS as test data. To compare the accuracy of the proposed method with those of the previous models, we used their evaluation datasets, ETRI-POS, and HANTEC-HKIB94. SEJONG is the colloquial corpus of the 21st Century Sejong Project (6669614 words; http://www.sejong.or.kr/eindex.php). ETRI-POS and ETRI-PARSE are the parts-of-speech (POS) tagged corpus (288269 words) and syntactic parsing corpus (2200925 words; very different from ETRI-POS) distributed by the Electronics and Telecommunications Research Institute (ETRI; http://www.etri.re.kr/eng), respectively. HANTEC-HKIB94 is a set of news articles (3224198 words) distributed by the Korea Institute of Science and Technology Information (KISTI; http://www.kisti.re.kr/english). CLMET is a corpus of Late Modern British English (9818326 words; https://perswww.kuleuven.be/~u0044428/). We use CLMET in order to evaluate the performance variations in languages with quite different structures.

To experiment with the proposed method, we removed any word boundary information in the test data. Then, we evaluated the proposed method by the character-unit accuracy, as shown in (1).

Consider

\[ A_{\text{char}} = \frac{\text{the number of correctly spaced characters}}{\text{the number of characters}} \]  

(1)

4.2. Experimental Result. The first experiment was conducted to evaluate the performance variations according to the amount of main memory used. We computed the performances of the proposed method at various threshold values (i.e., minimum frequencies and minimum probabilities), as listed in Figure 4.

In Figure 4, the proposed method used SEJONG for constructing the word spacing dictionary and ETRI-PARSE for testing performance and parameter tuning. As shown in Figure 4, as the minimum frequencies were increased, both memory usage and character-unit accuracy generally
Input sentence: *jip-e ga* (go home)

- **Check**
  - $F(jip-e-ga) \geq \alpha$
  - $P(jip-e\ S\ ga) \geq \beta$
  - **Yes**

**There is no bigram to replace the trigram.**
Therefore, **insert the trigram "jip-e-ga"**

- **Check**
  - $F(e-ga) \geq \alpha$
  - $P(e\ S\ ga) \geq \beta$
  - **Yes**

**There is no unigram to replace the bigram.**
(= The trigram can be replaced by the bigram)
Therefore, **insert the bigram "e-ga"**

- **Check**
  - $(F(e) + F(ga))/2 \geq \alpha$
  - $(P(e \ S) + (S \ ga))/2 \geq \beta$
  - **Yes**

The trigram can be replaced by the unigrams.
Therefore, insert the unigrams "eS" and "S ga"

**Figure 3:** An example of the word spacing dictionary construction.

decreased. When the minimum frequencies were less than or equal to three, the memory usage increased because some character bigrams with low frequencies were replaced by character unigrams with high probabilities. For example, spaces probably follow some case markers, such as "eun," "neun," "i," and "ga," which consist of one character because Korean is a language with well-developed morphemic markers. Based on this experimental result, we conclude that the proposed model with parameters set at $\alpha = 4$ and $\beta = 0.65$ is most suitable for information appliances with memory capacities less than 1 MB.

In the second experiment, we compared the accuracy of the proposed model with those of previous models using the same training and test data, because it was not easy to implement all of the previous models, as summarized in Figure 5.

In Figure 5, the solid bars represent the accuracies at which models are trained using SEJONG and are tested using ETRI-POS. The oblique-lined bars represent the accuracies at which models are trained using 90% of HANTEC-HKIB94 and are tested using 10% of HANTEC-HKIB94. The 95% confidence intervals of the proposed model ($\alpha = 4$ and $\beta = 0.65$) in ETRI-POS and HANTEC-HKIB94 were [0.9458, 0.9462] and [0.9429, 0.9433], respectively. The 95% confidence interval of the proposed model ($\alpha = 6$ and $\beta = 0.65$) in ETRI-POS was [0.9429, 0.9433], and that of the proposed model ($\alpha = 1$ and $\beta = 0.65$) in HANTEC-HKIB94 was [0.9486, 0.9490]. As shown in Figure 5, the proposed model ($\alpha = 4$ and $\beta = 0.65$) showed a slightly better than or similar performance to the previous models for information appliances even though it does not use any arithmetical computations except for dictionary retrieval.

Table 2: Performance comparison with state-of-the-art machine learning models.

| Model                     | $A_{char}$ | Response time (sec/MB) | Memory usage (MB) |
|---------------------------|------------|------------------------|-------------------|
| Proposed method ($\alpha = 4$, $\beta = 0.65$) | 0.9460     | 1.395                  | 0.79              |
| Lee et al. [14]           | 0.9748     | 5.174                  | 67.30             |
| Lee and Kim [12]          | 0.9837     | 20.48                  | 303.00            |

It showed a higher accuracy of 0.22% than in Kim et al. [17] although it used only approximately 0.28 MB more memory. When we set $\alpha$ and $\beta$ to 6 and 0.65, respectively, our model showed similar accuracy and memory usage to Kim et al. [17]. In addition, it had a higher accuracy than all the bigram-based models (i.e., Kang and Woo [11] and Lee et al. [13]). However, the proposed model showed slightly lower accuracy than in Park et al. [18]. However, Park et al.’s model [18] requires considerably more processing time because it is based on an HMM. When we set $\alpha$ and $\beta$ to 1 and 0.65, respectively, the proposed model showed better performance (accuracy: 0.9488, memory usage: 1.28 MB) than in Park et al. [18]. Note that the amount of memory consumed by Park et al. [18] and Lee et al.’s models [13] was unknown because the respective authors did not reveal memory usages in their corresponding papers.

In the third experiment, we compared the performance of the proposed model with that of state-of-the-art machine learning models, as shown in Table 2.

The models in Table 2 were evaluated on a PC with an Intel Core i5 CPU, 8 GB memory, and Windows 7 operating
system, using the C++ programming language. As shown in Table 2, each of the state-of-the-art models demonstrated a higher accuracy than the proposed model. However, both state-of-the-art models needed a comparatively large amount of main memory and both additionally required longer processing times. Although the computing power of information appliances is rapidly increasing, the memory usage and the processing times of the state-of-the-art models are still limiting factors for information appliances. In particular, in mobile devices with restricted battery capacity, long processing times lead to rapid battery consumption.

In the fourth experiment, we compared the performance of the proposed model in languages with quite different structures.

English versions of the proposed model in Table 3 use two times longer n-grams (i.e., character bigrams, character 4-grams, and 6-grams) than Korean versions because a Korean character is represented by two bytes which are equal to the

**Table 3:** Performance variations in Korean and English ($\alpha = 4, \beta = 0.65$).

| Training data | Test data | $A_{\text{char}}$ |
|---------------|-----------|------------------|
| SEJONG        | ETRO-POS  | 0.9460           |
| CLMET (90%)   | CLMET (10%) | 0.9696          |
size of memory necessary for saving two English characters. As shown in Table 3, the performances of the proposed model were not seriously affected by language changes.

In the last experiment, we analyzed the component ratios of character $n$-grams in the word spacing dictionary, as well as performance variations, when each character $n$-gram class is not used, as shown in Table 4.

In Table 4, "All-gram" represents the proposed method when all character $n$-grams in the word spacing dictionary are used. Further, "-Unigram," "-Bigram," and "-Trigram" denote the proposed methods when unigrams, bigrams, and trigrams, respectively, are excluded from the word spacing dictionary. As shown in Table 4, the component ratios and the performance variations showed similar tendencies in two kinds of training data. This reveals that the size of the word spacing dictionary will not be dramatically changed according to training data. In addition, the performance variations show that the importance of character $n$-grams in the proposed model is as follows: bigram > trigram > unigram.

The main reasons why the proposed method inserted word spaces incorrectly were similar to the reasons why the previous models behaved in the same way. First, there were many unknown words in the test data, and low-frequency characters raised many errors. These sparse-data problems are very critical because new words (including intentionally misspelled words) are continuously being coined, especially in mobile environments, because mobile device users often input abbreviated or idiomatic words that are intentionally generated to reduce the length of sentences or to express familiarity between friends [21]. To alleviate these sparse-data problems at a basic level, we should increase the amount of training data. Second, there were many cases in which even by manually observing character five-grams as contextual information, it could not be determined whether a space needed to be inserted. To resolve this problem, we should study methods to select and use long-distance characters (i.e., characters that are over five characters away from a target position) as contextual information.

5. Conclusion

We proposed an automatic word spacing method based on pattern matching for information appliances with limited computational power. The proposed method consists of a word spacing dictionary and a pattern-matching module. The word spacing dictionary contains a set of character $n$-grams in which trigrams are complementary mixed with unigrams and bigrams based on frequency and word spacing probability. When a sentence is input, the pattern-matching module inserts spaces by simply looking up the word spacing dictionary in a back-off manner. In comparative experiments with previous probabilistic models, the proposed method showed considerably good performance (memory use: 0.79 MB, character-unit accuracy: 0.9460) even though it does not require complex probabilistic computations. On the basis of these experiments, we conclude that the proposed method is suitable for information appliances with many hardware limitations because it requires less memory and consumes less battery power.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2013R1A1A4A01005074). This research was also supported by the IT R&D program of MOTIE/MSIP/KEIT. [10041678, The Original Technology Development of Interactive Intelligent Personal Assistant Software for the Information Service on Multiple Domains]

References

[1] A. Kamilaris and A. Pitsillides, “Social networking of the smart home,” in Proceedings of the IEEE 21st International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC ’10), pp. 2632–2637, September 2010.
[2] M. B. Chandak and R. Dharaskar, “Natural language processing based context sensitive, content specific architecture & its speech based implementation for smart home applications,” International Journal of Smart Home, vol. 4, no. 2, pp. 1–10, 2010.
[3] A. Hurson, Connected Computing Environment, Academic Press, New York, NY, USA, 2012.
[4] O. Tsur, A. Littman, and A. Rappoport, “Efficient clustering of short messages into general domain,” in Proceedings of the International AAAI Conference on Weblogs and Social Media, 2013.
[5] C.-N. Seon, H. Kim, and J. Seo, “Efficient appointment information extraction from short messages in mobile devices with limited hardware resources,” Pattern Recognition Letters, vol. 32, no. 2, pp. 127–133, 2011.

[6] C. N. Seon, H. Kim, and J. Seo, “Translation assistance system based on selective weighting and cluster-based searching methods,” International Journal on Artificial Intelligence Tools, vol. 21, no. 6, Article ID 12500281, 13 pages, 2012.

[7] J. Choi, “Automatic Korean spacing words correction system with bidirectional longest match strategy,” in Proceedings of the 9th Conference on Hangul and Korean Information Processing, pp. 145–151, 1997 (Korean).

[8] S. Kang, “Eojeol-block bidirectional algorithm for automatic word spacing of Hangul sentences,” Journal of Korea Information Science Society, vol. 27, no. 4, pp. 441–447, 2000 (Korean).

[9] P. Cohen, N. Adams, and B. Heeringa, “Voting experts: an unsupervised algorithm for segmenting sequences,” Intelligent Data Analysis, vol. 11, no. 6, pp. 607–625, 2007.

[10] S. Kang, “Automatic correction of word-spacing errors using by syllable bigram,” Journal of Korean Society of Speech Sciences, vol. 8, no. 2, pp. 83–90, 2001 (Korean).

[11] S. Kang and C. Woo, “Automatic segmentation of words using syllable bigram statistics,” in Proceedings of the 6th Natural Language Processing-Pacific Rim Symposium, pp. 729–732, 2001.

[12] C. Lee and H. Kim, “Automatic Korean word spacing using Pegasos algorithm,” Information Processing and Management, vol. 49, no. 1, pp. 370–379, 2013.

[13] D. Lee, S. Lee, H. Rim, and H. Lim, “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, 2002.

[14] D.-G. Lee, H.-C. Rim, and D. Yook, “Automatic word spacing using probabilistic models based on character n-grams,” IEEE Intelligent Systems, vol. 22, no. 1, pp. 28–35, 2007.

[15] Y. Song and H. Kim, “Automatic word spacing of erroneous sentences in mobile devices with limited hardware resources,” Information Processing and Management, vol. 47, no. 4, pp. 518–527, 2011.

[16] S. Kim, S. Yang, and Y. Ko, “A word spacing system based on syllable patterns for memory-constrained devices,” Journal of KIIS, vol. 37, no. 8, pp. 653–658, 2010 (Korean).

[17] S. Kim, S. Yang, and Y. Ko, “A novel discriminative probabilistic model for automatic word spacing on mobile devices,” International Journal of Innovative Computing, Information and Control, vol. 8, no. 7, pp. 1–11, 2012.

[18] S.-B. Park, E.-K. Lee, and Y.-S. Tae, “Automatic word spacing in korean for small memory devices,” in Innovations in Applied Artificial Intelligence, vol. 3533 of Lecture Notes in Computer Science, pp. 249–258, 2005.

[19] S.-B. Park, Y.-S. Tae, and S.-Y. Park, “Self-organizing n-gram model for automatic word spacing,” in Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL ’06), pp. 633–640, July 2006.

[20] R. Ash, Information Theory, Wiley-Interscience, New York, NY, USA, 1965.

[21] H. Noh, J. Cha J, and G. G. Lee, “A joint statistical model for simultaneous word spacing and spelling error correction,” Journal of KIIS, vol. 34, no. 2, pp. 131–139, 2007 (Korean).