IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 22, 2019, accepted March 8, 2019, date of publication March 15, 2019, date of current version April 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2905252

Skip-Gram-KR: Korean Word Embedding for Semantic Clustering

SUN-YOUNG IHM, JI-HYE LEE, AND YOUNG-HO PARK
Department of IT Engineering, Sookmyung Women's University, Seoul 04310, South Korea
Corresponding author: Young-Ho Park (yhpark@sm.ac.kr)

This work was supported by the Institute for Information and Communications Technology Promotion (IITP) Grant funded by the Korean Government (MSIP) (SIAT CCTV Cloud Platform) under Grant 2016-0-00406.

ABSTRACT Deep learning algorithms are used in various applications for pattern recognition, natural language processing, speech recognition, and so on. Recently, neural network-based natural language processing techniques use fixed length word embedding. Word embedding is a method of digitizing a word at a specific position into a low-dimensional dense vector with fixed length while preserving the similarity of the distribution of its surrounding words. Currently, the word embedding methods for foreign language are used for Korean words; however, existing word embedding methods are developed for English originally, so they do not reflect the order and structure of the Korean words. In this paper, we propose a word embedding method for Korean, which is called Skip-gram-KR, and a Korean affix tokenizer. Skip-gram-KR creates similar word training data through backward mapping and the two-word skipping method. The experiment results show the proposed method achieved the most accurate performance.

INDEX TERMS Word embedding, natural language processing, Korean word embedding, text mining, deep learning, semantic clustering, machine learning.

I. INTRODUCTION

Recently, deep learning architectures and algorithms have reached a plateau of almost maximum performance in the fields of computer vision and pattern recognition, while research on using deep learning algorithms for natural language processing is expanding rapidly. The natural language processing using deep learning algorithms have been widely used in the field of text mining, user opinion mining and management information.

To use deep learning architectures or algorithms, text must be entered as vector values of a fixed length. To convert the text data to fixed-length vector values, word embedding methods are usually employed. Currently, there have been no word-embedding models developed independently in Korea. Instead, existing word embedding methods such as Skip-gram [7], [15], [16], [20], CBOW [15], and GloVe [18] are often used [3].

However, word embedding methods such as Skip-gram, CBOW, and GloVe are models which were developed to analyze English data. Korean and English are languages with different structures and grammars, so the same rules cannot be applied to both. There have been studies of languages that have different linguistic and grammatical structures with English [2], [8], [24]. This paper focuses on the problem of existing word embedding methods being unable to reflect the Korean language's word order and structure when such methods are applied to Korean.

There are also differences between the two languages when using word extraction methods for word embedding. Before discussing word extraction, some terms have been listed in Table 1 according to [9] and [27] so that the concepts used in this paper will be clear. In this paper, the embedding model's input values are all defined as words

Example 1 shows the difference when words are extracted from Korean and English sentences based on spacing.

Example 1: Comparison of extracting words from Korean and English sentences using spacing

·오늘 날씨가 춥다 → 오늘, 날씨가, 춥다
·Today weather is cold → Today, weather, is, cold

If most Korean text is separated by spaces as shown in Example 1, nouns and particles or verbs and endings are combined into the form of syntactic words. This is because

The associate editor coordinating the review of this manuscript and approving it for publication was Tao Zhou.

TABLE 1. Terminology definition.

Term	Definition
Syntactic Word	One unit of a sentence's structure, usually the same as a spaced unit
Root	The central part of a word which shows the actual meaning when the word is analyzed
Affix	A peripheral part of a word which is attached to a root when forming a word and which limits the root's meaning
Morpheme	The smallest morphological unit of language which has meaning
Syllable	The smallest phonetic unit which consists of a combination of vowels and consonants
Word	The input values of the word embedding model

English is an inflectional language which shows the grammar of a word through changes in the form of the word itself, while Korean is an agglutinative language which shows the grammar of a word by combining roots and affixes [27]. Because of this, additional preprocessing is performed for Korean word embedding to separate syntactic words into morphemes, which are the smallest units of meaning. Here, if the words are not properly extracted and the original meaning cannot be preserved, skewed information could be transferred to the deep learning model. Therefore, a word extraction method which preserves the original words is necessary.

This paper proposes two preprocessing methods for Korean word embedding. The first proposed method is Skip-gram-KR. This is a word embedding method which improves the Skip-gram model's training data creation method to reflect the word order and structure of the Korean language. The second proposed method is a Korean affix tokenizer (KR-affix tokenizer), which is a new word extraction method for Skip-gram-KR. The scope of this study is limited to word extraction and word embedding, which are steps performed before applying deep learning architectures and algorithms.

There have been previous studies on performing word embedding which reflects the characteristics of the Korean language [4], [17], but they are different from the proposed method in that they divided the words too minutely [4] or the experiments were performed by selecting only a single derivational affix [17]. This paper is also different from the previous two studies in that it has improved the structure of the word embedding model. This paper makes the following contributions.

A. PROPOSAL OF A WORD EMBEDDING METHOD FOR KOREAN

Skip-gram-KR is a word embedding method which improves on Skip-gram's training data creation method so that it can reflect the word order and structure of Korean. Skip-gram-KR considers the word order of Korean, in which important information appears at the end of sentences, and it creates similar word training data through backward mapping, which is introduced in Definition 1. It also reflects the structural characteristics of the Korean language in which roots and affixes are combined, and it separates roots from affixes and maps similar words through a two-word skipping method, which is introduced in Definition 2. This method removes the noise created by affixes that are combined with roots, and it reduces the error rate of the word embedding process.

B. PROPOSAL OF A KOREAN AFFIX TOKENIZER

The Korean affix tokenizer (KR-affix tokenizer) is a word extraction method which is proposed for use with Skip-gram-KR and separates syntactic words into roots and affixes. It uses published dictionary information to perform the initial filtering of root-affix candidates, and it only uses statistical information for the words which appear in all the texts when selecting roots-affixes. By separating affixes within syntactic words, it increases accuracy when extracting neologisms, unknown words, and compound nouns.

C. PROPOSAL OF EVALUATION INDEX FOR KOREAN SEMANTIC SIMILARITY

The Korean semantic similarity evaluation index (Korean semantic top-k score, KST) we developed is an evaluation index which quantifies the weight values of the cosine similarity ranking of word vectors. KST is used to quantify the extent to which each word embedding model clusters similar words in close positions.

D. VERIFYING PERFORMANCE OF PROPOSED MODEL THROUGH EXPERIMENTS

Experiments were performed to compare the tokenizing accuracy of the proposed Korean affix tokenizer with that of morpheme analyzers. In addition, the accuracy of a model which embeds words via the Korean affix tokenizer and Skip-gram-KR was compared to the accuracy of an existing word embedding model. We were able to derive meaningful results from these experiments.

II. RELATED WORK

This section introduces research related to the proposed word embedding methods for Korean and Korean affix tokenizers for word extraction. First, section A provides a detailed description of word embedding methods, in particular the Skip-gram method. Section B describes research on existing methods for finding word boundaries.

A. WORD EMBEDDING METHODS

This section introduces word embedding methods. The concept of word embedding is explained briefly, with the explanation focusing on the Skip-gram model. Word embedding is a method which quantifies high-dimension sparse vectors as low-dimension dense vectors while preserving the similarity of the distribution of a word at a particular location and its surrounding words. Early language processing used one-shot vectors in which the index of each word was assigned a '1' and the remaining indexes were filled with zeros for words in a document. However, this method was difficult to calculate

when the number of words became a very large number because the space occupied by the word vector expressions was large, but the information was sparse. Also, all word vectors were orthogonal, so the similarity of words could not be calculated. To overcome these inefficiencies, a neural network language model [1] was proposed, in which dense vector expressions are used in a low-dimension vector space. Nowadays, word2vec [15] is widely used as a word embedding model because it reduces the computational complexity of the neural network language model.

Normally, word2vec is a generic term for both the CBOW (Continuous Bag-of-Words) model, which looks at the words which appear before and after a word's position and predicts the word in the current position, and the Skip-gram model, which looks at the word in the current position and predicts the words appearing before and after it. This section provides an explanation focusing on the Skip-gram model, which is related to the method proposed in this paper.

In the Skip-gram model, the goal is to maximize the conditional probability of the appearance of word w_t at the current location t and a nearby word w_{t+j} as shown in (1) [26]. T in (1) is the position of each of the words expressed as $w_1, w_2, \ldots, w_t$ and c is the range of words which are expected to appear near the word at the current position w_t. As c becomes larger, more words are defined as words which are similar to the word at the current position. θ refers to all the variables optimized by (1). Typically, the vector representations of each word are adjusted to maximize the probability values in (1), and these vector expressions are denoted by θ.

$$J'(\theta) = \prod_{t=1}^{T} \prod_{-c \le j \le c, j \ne 0} p(w_{t+j}|w_t;\theta). \tag{1}$$

Equation (1) can be simplified into (2) by applying the negative log likelihood function [26]. If (1) is converted into (2), the objective of the equation changes to minimizing the log probability that word w_{t+j} appears near word w_t at the current position t.

$$J(\theta) = -\frac{1}{T} \prod_{t=1}^{T} \prod_{-c \le j \le c, j \ne 0} logp(w_{t+j}|w_t). \tag{2}$$

Here, $\frac{1}{T}$ is a normalization term based on the total number of word positions.

$p(w_{t+j}|w_t)$ is the probability of word w_t appearing at the current position t and w_{t+j} appearing nearby. It is expressed as a probability value between 0 and 1 using the softmax function as shown in (3) [16]. w_I in (3) is the unique word index of word w_t at the current position t, and it is an input value. w_o is the unique word index of the nearby word w_{t+j}, and it is an output value. W is the total number of words excluding duplications. v is a matrix of the vector expressions of the current position's word, and v' is a matrix of the vector expressions of the nearby words. The Skip-gram model adjusts the values of the two-word vector matrices v and v'. The values of the two matrices are different from each other, and a single word has two vector expressions. Here, probability is determined by the distance between the two word vectors of the different matrices v'_{w_O} and v_{w_I}.

$$p(w_O|w_I) = \frac{\exp(v'_{w_O} v_{w_I})}{\sum_{w=1}^{W} \exp(v'_w v_{w_I})}. \tag{3}$$

By substituting logarithms into (3), it can be written as (4) [16].

$$\log p(w_O|w_I) = \left(v'_{w_O} v_{w_I}\right) - \log\left(\sum\nolimits_{w=1}^{W} \exp\left(v'_w v_{w_I}\right)\right). \tag{4}$$

Skip-gram adjusts the vector so that the inner products of v_{w_I}, which is the center word's vector expression, and v'_{w_O}, which is the nearby word's vector expression, are the same and equal to 1. This problem can be solved as a binary regression problem as in (5) [16]. The term on the left side of (5) is the inner product of the current position's word and the nearby word. The term on the right side of (5) is the inner product of the current position's word and a randomly selected word that does not appear nearby, and these vectors are adjusted so that they are positioned far away in the vector space and their inner product is 0. As this process is repeated, clusters of similar words are formed in the vector space. In the existing softmax algorithm, the sum of the probabilities for all words is used as the denominator, but according to [19], a method which calculates all W number of words and normalizes $p(w_{t+j}|w_t)$ requires significant computation and is inefficient. Therefore, we used a negative sampling method which extracts k number of random words which do not appear near w^t to obtain a value similar to softmax. (5) uses the negative sampling.

$$\log \sigma\left(v'_{w_O} v_{w_I}\right) + \sum\nolimits_{i=1}^{k} E_{w_i \sim P_n(w)} \left[\log \sigma(-v'_{w_i} v_{w_I})\right]. \tag{5}$$

However, when the Skip-gram which we have described is used for Korean word embedding, problems occur because it is an algorithm made to analyze English. As shown in (1), the method of mapping the word at the current position and the words located before and after it as similar words does not properly reflect Korean word order. For example, in English the verb appears in any position in the sentence, so it is reasonable to view the words before and after it as similar words. However, in Korean, the core verb of a sentence is located at the end of the context. There is another difference in that modifiers of nouns are mainly located before the nouns in Korean, but can appear freely before or after the nouns in English.

GloVe [22] is a word embedding model which uses matrix decomposition to calculate the frequency of words. It addresses the fact that CBOW and Skip-gram require long learning times because they learn by repetitive analysis of all words based on position units and they cannot use statistical information from all of the data. In this method, a co-occurrence matrix for all the number of words is constructed, and the matrix is decomposed so that the similarity of two words becomes closer. However, this method also does not consider the word order of a language during the process of building the co-occurrence matrix of all words. The proposed

Skip-gram-KR method applies the word order and structure of the Korean language to the word embedding model.

B. METHODS FOR FINDING WORD BOUNDARIES

This section describes methods for finding the boundaries of Korean words. As an agglutinative language, Korean expresses grammar by combining roots and affixes [27]. Therefore, to perform word extraction, a process is performed which separates the combined language. The words used in word embedding during Korean natural language processing are mainly words that use morphemes, which are the smallest units among words that have meaning.

The problem when morphemes are used as words is that words' organic meanings can be lost in the process of separating them into morphemes. Also, this method is inconvenient because a dictionary must be constructed. After the Sejong corpus (a Korean corpus with the main parts of speech tagged, released by the National Institute of Korean Language as part of the 21st Century Sejong Plan) was published, methods [21] were proposed which found the minimum unit of meaning using the morpheme corpus' statistical information, but these methods still have difficulty managing neologisms, etc.

The proposed Korean affix tokenizer is a method which splits syntactic words into roots and affixes. It is a word extraction method for Skip-gram-KR, and it is also a method which preserves the sequential meaning of words. This method can accurately extract neologisms, unknown words, and compound nouns. It can also extract only necessary words when the language's morphological information is not required.

Currently, there has been almost no research on methods for splitting syntactic words into roots and affixes. This is because language data on extracting words which are split into roots and affixes has not been accumulated as much as language data on morphemes. Therefore, the proposed method does not use any particular language data. Instead, it uses word statistics from the dictionary and all texts to find word boundaries. By finding boundaries and separating the affixes of Korean syntactic words, it extracts semantic words and formal words.

Methods of using affixes to determine Korean word boundaries can be found in existing studies [12], [21]. The study in [21] is similar in that it proposes a method which attempts to exclude meaningless morphemes from the analysis results, but it does not use the number of types of affixes which can be combined with roots. In [12], removing endings or postpositional particles according to rules was only discussed in terms of the method's uncertainty. Our proposed method is different in that we use the number of types of affixes which can be combined with roots, and we remove values which are below a threshold from the classification target.

III. THE PROPOSED METHOD

This section introduces the proposed word embedding method for Korean and the Korean affix tokenizer for word extraction. First, section A discusses the Skip-gram-KR word embedding method which improves upon Skip-gram to reflect the structure and word order of Korean. Next, section B discusses the Korean affix tokenizer which splits syntactic words into roots and affixes to extract words from syntactic words. The Skip-gram-KR method in A uses the words extracted by the Korean affix tokenizer in B and performs word embedding. Therefore, the higher concept of the Skip-gram-KR is discussed first.

A. SKIP-GRAM-KR

This section proposes Skip-gram-KR, a word embedding method for Korean. Skip-gram-KR improves the training data creation method of Skip-gram so that it can express the linguistic characteristics of Korean. In this paper, the linguistic characteristics of Korean refer to the word order and structure of Korean sentences. Here, word order refers to the order in which words appear in Korean sentences where modifiers appear first and the important meaningful words appear afterwards. Structure refers to the grammatical structure of the syntactic words which are the elements that form the sentences. In English, words are brought together to form a sentence, but in Korean, syntactic words are brought together to create a sentence. Most Korean syntactic words are formed as a combination of roots and affixes. Skip-gram-KR is different from Skip-gram in that it reflects these characteristics of Korean in the word embedding process. Fig. 1 shows the structure of Skip-gram-KR.

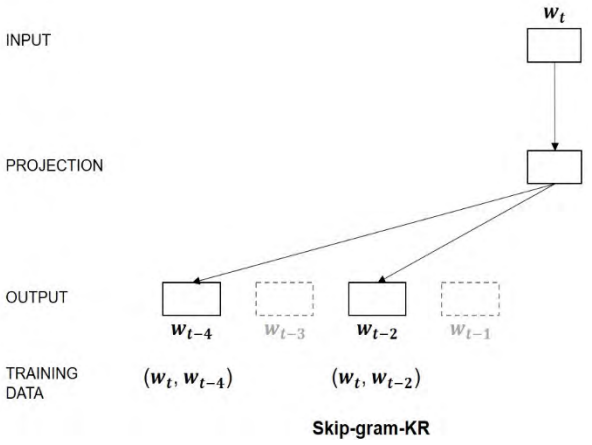

FIGURE 1. Skip-gram-KR structure.

In Fig. 1, the word w_t at the current position t is the input value, and the projection matrix expresses the dense vectors of input word w_t. The word w_{t-2} which is at the position $t-2$ before the current position t, and the word w_{t-4} which is at the position $t-4$, are the output values. Here, two words such as (w_t, w_{t-2}) and (w_t, w_{t-4}) which are mapped as input and output represent similar word training data which shares a similar context. In Skip-gram-KR, after creating the training data, if there is a word w_t at the current position t, the word's vector expressions are repeatedly adjusted to maximize the conditional probability of the appearance of a word which is

an even number of places before the current position. This is because Skip-gram-KR is a model which predicts words such as w_{t-2} and w_{t-4}, which are an even number of places before the current position when word w_t is at current position t.

The previously described method of taking only the words which are before the word w_t at current position t and mapping them to output values is called the backward mapping method, and it is explained in detail in subsection 1). The method of selecting as output values only the words which are an even number of places before the current position t rather than all words is called the two-word skipping method, and it is introduced in 2). Finally, 3) describes the process of using Skip-gram-KR to obtain dense vector expressions of Korean words.

1) BACKWARD MAPPING METHOD TO REFLECT KOREAN WORD ORDER

This section describes the backward mapping method which reflects Korean word order. For Skip-gram-KR's similar word training data, when there is a word w_t at current position t, words located before w_t are selected as similar words, and the premise behind this is explained below.

The word w_t at the current position t and the word w_{t-j} which appears before the current position t not only share a more similar context than the word w_{t+j} which appears after the current position, they also have a higher linguistic semantic correlation.

The existing Skip-gram defines words with a similar context as similar words. However, the proposed Skip-gram-KR considers both similar context and semantic correlation. This paper assumes that words which appear before the word at the current position have a higher semantic correlation than words which appear after the current position.

The premise of this paper confirms Korean word order by experience. For example, in Korean sentences, modifiers usually come before the nouns or verbs which have the main meaning, and the sentences' core verbs are located at the ends of sentences. This paper attempts to reflect this premise in the word embedding model. In this paper, the similar word training data mapping method which is based on this is defined as backward mapping, and it is described in Definition 1.

Definition 1: Backward mapping maps the word w_t at the current position t and words located before the current position t when creating Skip-gram-KR's similar word training data. As shown by $1 \leq j \leq c, j \neq 0$ in (6), a limited range of words is mapped to select the words before the current position t.

$$J'(\theta) = \prod_{t=1}^{T} \prod_{1 \leq j \leq c, j \neq 0} p(w_{t-2j} | w_t; \theta). \tag{6}$$

In (6), c is the number of similar words which Skip-gram-KR is learning, and as c becomes larger, more words can be learned as similar words.

In this section, we described the backward mapping method which reflects Korean word order. However, Skip-gram-KR uses the backward mapping method, but it does not map all words in order. The reason that words are mapped backwards by even numbers such as w_{t-2j} is explained in the next section.

2) TWO-WORD SKIPPING METHOD TO REFLECT KOREAN STURUCTURE

This section describes the two-word skipping method which reflects Korean structure. Here, Korean structure refers to the grammar of Korean syntactic words. Korean sentences are composed of syntactic words, and syntactic words are combinations of even smaller semantic words. The two-word skipping method is defined in Definition 2.

Definition 2: Two-word skipping is a method which selects similar word training data by selecting the word w_t in the current position t and the word w_{t-2j}, which are an even number of places before the current position.

The advantage of this method is that if there are two sequentially located words such as a root and an affix which share a similar context but do not have semantic correlation, they are prevented from being mapped to the similar word training data.

In order for two-word skipping to be performed accurately, we must perform word embedding which splits all syntactic words into roots which express meaning and affixes which perform grammatical functions. However, when words are extracted by conventional morpheme-based tokenizing, problems occur in cases such as when words that can be used independently are used as syntactic words, when roots appear sequentially, and when affixes appear sequentially. As a method to resolve these problems, Skip-gram-KR uses a Korean affix tokenizer which always splits syntactic words into roots and affixes. In the Korean affix tokenizer, if a syntactic word is a word being used independently, a padding code P is used at the affix position to maintain separation between the root and the affix (For details, see 3.B).

Fig. 2 and Fig. 3 compare Skip-gram and Skip-gram-KR's similar word training data creation. Fig. 2 is an example of conventional training data creation using the Skip-gram model, and Fig. 3 is an example of Skip-gram-KR's training data creation using backward mapping and two-word skipping. In Fig. 2, two words before and two words after ''technology'' are selected as similar words and mapped as (technology, natural language processing), (technology, P), (technology), (technology, based). Conversely, in Fig. 3, Skip-gram-KR selects the words which are an even number of places before ''technology'' as similar words and maps them as (technology, machine translation), (technology, natural language processing). Here, we can see that two-word skipping removed the noise of selecting an affix (e.g. based) as a similar word.

We could consider removing all affixes before word embedding as a means to remove the noise from affixes. However, in fields which must use rich and complex human-level language such as interactive platforms and text summarization, dense vector expressions of affixes are absolutely required. In these cases, it is necessary to have a

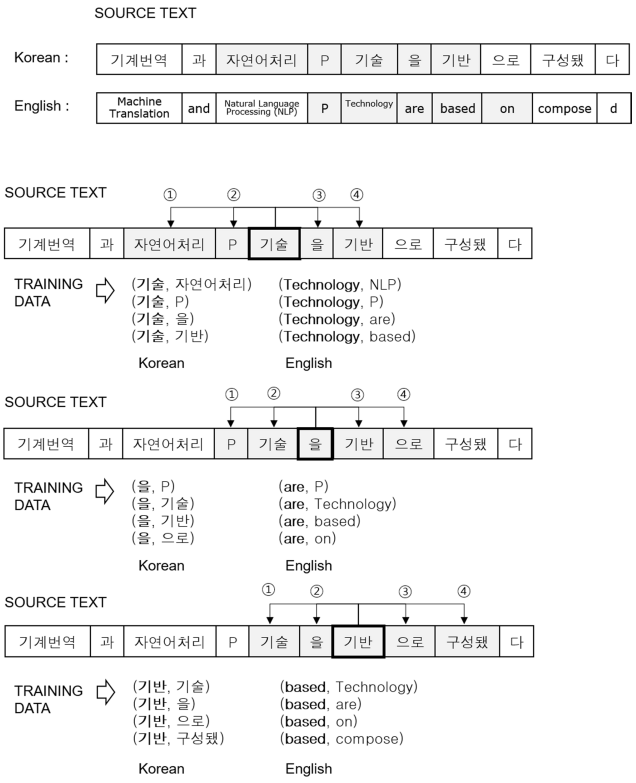

FIGURE 2. A training data creation of skip-gram.

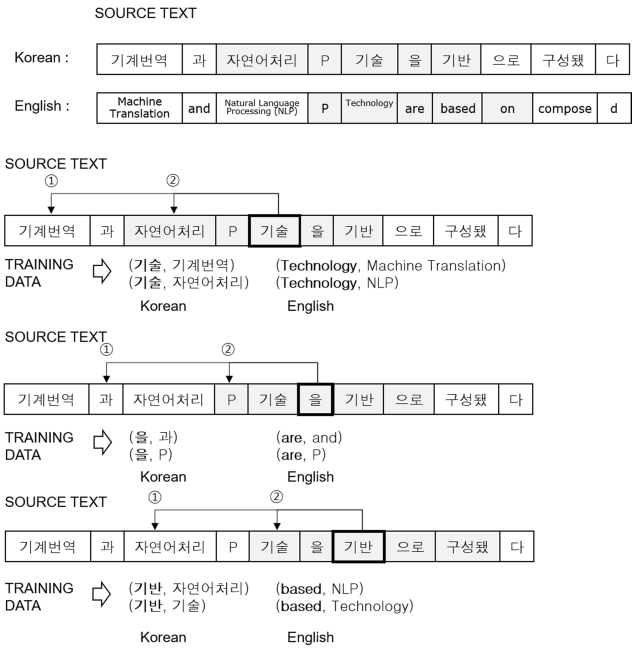

FIGURE 3. A training data creation of skip-gram-KR.

method which does not delete the affixes but does remove their noise.

This section describes the two-word skipping method which reflects Korean structure. Next, we describe the word embedding process which obtains Korean word vectors by using Skip-gram-KR and applying backward mapping and two-word skipping.

3) WORD EMBEDDING PROCESS USING SKIP-GRAM-KR

This section describes the process of using Skip-gram-KR to obtain dense vector expressions of Korean words. Equation (6) maximizes the conditional probability of the word w_{t-2j} appearing an even number of spaces before the current position t when word w_t appears at the current position t. The variable T in 6 is the position of each word expressed as $w_1, w_2, \ldots, w_T$. Here, w_t is the word located at the current position t, and w_{t-2} is the word located an even number of places before the current position t. When similar word training data is created, backward mapping and two-word skipping are used to map w_{t-2}, and this is the new method proposed by Skip-gram-KR. The term c, is the range of similar words for backward mapping, and is limited to $1 \leq j \leq c$. First, the product of the probabilities for c number of previous words is calculated, and the product of probabilities for each of T words is determined. θ represents all the variables which were optimized by (6), and the dense vector expressions of the words also correspond to θ. To make (6) easy to calculate, a negative log likelihood function is used to change it to (7), which minimizes the conditional log probability of the word w_{t-2j} appearing an even number of spaces before the current position t when word w_t appears at the current position t. The term $\frac{1}{T}$ in (7) is a normalization term based on the total number of word positions.

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{1\leq j\leq c, j\neq 0}\log p(w_{t-2j}|w_t). \quad (7)$$

The process described from (8) to (10) is the same as in Skip-gram. For the probability $p(w_{t-2j}|w_t)$, the softmax function is used as in (8) to obtain a probability value between 0 and 1.

$$p(w_O|w_I) = \frac{\exp(v'_{w_O} v_{w_I})}{\sum_{w=1}^{W}\exp(v'_w v_{w_I})}. \quad (8)$$

In (8), w_I is the unique word index of the word w_t at the current position t, and it is Skip-gram-KR's input value. w_O is the unique word index of w_{t-2}, which is the word an even number of places before the current position, and it is Skip-gram-KR's output value. v is a matrix of vector expressions of the word w_I at the current position, and v' is a matrix of vector expression of the word w_O an even number of places before the current position. W is the total number of words excluding duplications.

Skip-gram-KR adjusts the values of the two-word vector matrices v and v'. During this process, it separates roots and affixes and makes adjustments, thus differing from Skip-gram. This is because in the two-word skipping method, the vectors of words at an odd number of places previous and vectors of words at an even number of places previous to the current location do not affect each other.

In such a word embedding method, roots and affixes use separate vector spaces when the actual algorithm is performed. As a result, it can overcome errors in which words such as roots and affixes, which are not meaningfully similar but are used at the same time, are defined as similar words.

If the logarithm is substituted into (8), it can be expressed as (9). The $v'_{w_O}{}^{\top} v_{w_I}$ in (9) is the inner product of v_w, the vector of the word at the current location t, and v'_{w_O}, is the vector of the word at an even number of spaces previous. The two words are similar, so they are located close to each other in the vector space, and their positions are adjusted so that their inner product is 1.

$$\log p\left(w_O|w_I\right) = \left(v'_{w_O} v_{w_I}\right) - \log(\sum_{w=1}^{W} \exp(v'_w v_{w_I})) \quad (9)$$

Skip-gram-KR converts the problem to a binary regression problem as in (10) and uses negative sampling. The term on the left side is the inner product of the word at the current position and the previous word. Adjustments are made so that the two vectors are located close to each other in the vector space. On the other hand, the term on the right side is the inner product of the word at the current position and a randomly selected word which does not appear at an even number of spaces before the current position, so the two word vectors' positions are far apart. This process is performed repeatedly to obtain the Korean word vector expressions.

$$\log \sigma\left(v'_{w_O} v_{w_I}\right) + \sum_{i=1}^{k} E_{w_i \sim P_n(w)} \left[\log \sigma(-v'_{w_i} v_{w_I})\right]. \quad (10)$$

This section describes the process of using Skip-gram-KR to obtain Korean word vector expressions and compares it to Skip-gram. Next section describes the Korean affix tokenizer that aids in word extraction for Skip-gram-KR.

B. KOREAN AFFIX TOKENIZER

This section describes the Korean affix tokenizer which divides syntactic words into roots and affixes. The Korean affix tokenizer is a method which was designed for word embedding using Skip-gram-KR. In most Korean preprocessing procedures, morphemes are used as the word embedding units. However, the Skip-gram-KR method proposed in 3.A uses the two-word skipping method and has a structure which prevents roots and affixes from being mapped as similar words. For this reason, the Korean affix tokenizer proposed in this paper splits syntactic words into only two words, the root and the affix. Definition 3 defines the root-affix candidates which are used by this paper.

Definition 3: Root-affix candidates are all the pairs of roots and affixes which can be created as one syntactic word. The part of the word on the right side of the syntactic word's splitting position calculates the affix probability, so the search is performed in reverse. If a single syntactic word is composed of 7 syllables, $c_1, c_2, \ldots, c_T$, a total of 7 candidates are created. The root-affix candidates are created in the following order.

- Select the syntactic word's splitting position t in the reverse direction. The end of the right side is $t = 0$, and the last splitting position is $t = T - 1$.
- The part of the word on the right side of the splitting position is the affix candidate. If there is no affix candidate, the affix's place is filled with the padding token P. This is because Skip-gram-KR cannot be used if roots appear sequentially.
- The part of the word on the left side of the splitting position is the root candidate. There are cases where independent words are used as syntactic words, so the length of the root candidate is always 1.

Fig. 4 shows the tokenizing process for the example sentence "Suddenly, the payment with the shopping basket of the shopping mall has stopped." The example sentence is one of the frequently appearing sentence when user look at a shopping mall. First, a dictionary is used to filter the root-affix candidates. Next, root-affix scores are calculated based on statistical information from the words in all texts.

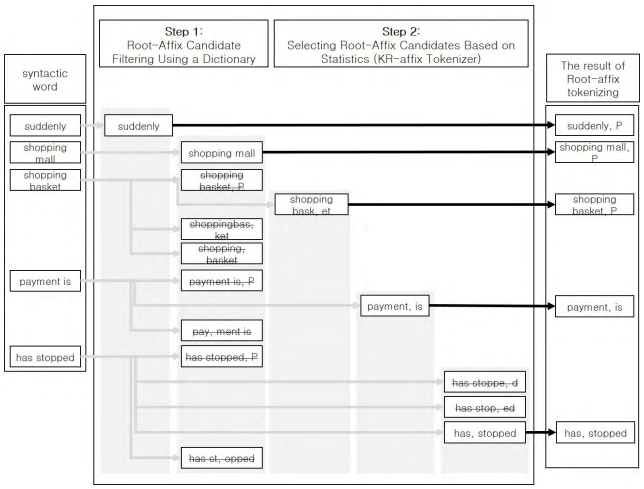

FIGURE 4. A root-affix tokenizing process.

Step 1 (Root-Affix Candidate Filtering Using a Dictionary): If the syntactic word is found in an adverb, determiner, or interjection dictionary, the syntactic word is not split, and it is output as a word. The word "suddenly" is an adverb, so it is output as (suddenly, P). Next, we remove the root-affix candidates in which the affix is not found in a postpositional particle or ending dictionary. For "shopping mall," there are no postpositional particles such as "mall" or "ping mall." Therefore, for this word, the entire syntactic word is seen as one word and is output as (shopping mall, P).

Step 2 (Selecting Root-Affix Candidates Based on Statistics): If the root-affix score is less than 0.15, the syntactic word is not split, and it is viewed as a word. Therefore, "shopping basket" must be split as (shoppingbas, ket) according to the statistics of the word, but it can be output as (shopping basket, P). If there is one candidate that has an affix score that exceeds 0.15, it is chosen to be a root and an affix. Fig. 4's (payment) is currently the only candidate. However, there are

cases where there are multiple candidates. In these cases, the word which has a larger number of affixes which can combine with the root is selected as the root. When selecting based on statistics, the frequency of ''has'' appearing is high, so (has, stopped) receives the highest score. Such statistical errors can be avoided if the word's boundary is determined according to the number of affixes which can be combined with the root. The Fig. 4 in Korean is in the Fig. 6 of Appendix.

First, in the dictionary-based root-affix candidate filtering stage, independently used syntactic words and candidates which do not include affixes are excluded from the probability calculations. To establish a standard for how the created candidates are judged for filtering, an independent word dictionary (which combines adverbs, determiners, and interjections) and a postposition particle dictionary (which combines detailed postposition particles and ending dictionaries) were created from among the electronic dictionaries provided by the National Korean Language Institute's Language Information Sharing Center. The independent word dictionary is a collection of words which are not used together with affixes. The words in the independent word dictionary can be differentiated based on spaces, and they can be used as independent words. However, if independent words include affixes, improper tokenizing can occur. For example, ''suddenly'' contains the ''ly'' affix, so there is a possibility it could be improperly split into ''sudden'' and ''ly.'' The independent word dictionary is used to resolve such errors. Words which are included in the independent word dictionary are deleted from the root-affix candidates.

The affix dictionary is also used to distinguish words which are used independently. Unlike roots, the category of affixes is small, and there are almost no changes in the language, so managing a dictionary is easy. For all root-affix candidates derived from syntactic words, if there is no word in the affix dictionary which matches the affix candidate, the candidate is viewed as a word that is not split. In this way, word filtering which uses the independent word dictionary and the affix dictionary can prevent errors caused by root-affixes which were improperly split. This also has the advantage of not performing the unnecessary process of calculating the root-affix score for words which must be used independently.

Next, we describe a method of selecting root-affix candidates via word frequency statistics for all text. This statistical information refers to the frequency at which words appear in the text. Word patterns which appear frequently in the entire analyzed text are considered information about words that can be trusted.

For a syntactic word made of T syllables $c_1, c_2, \ldots, c_T$, a total of T root-affix candidates are created. The Korean affix tokenizer determines a score for the word boundary of each candidate. Candidates with high scores are selected as roots and affixes, and the calculation method is called the score function. The score function is described in Definition 4.

Definition 4: The score function is a word boundary score for roots and affixes. A root-affix candidate with a high score is selected for splitting the syntactic word into a root and affix. The score is calculated by (11) and (12).

$$s = \frac{1}{2}\left(word\left(x_t\right) + affix\left(z_t\right)\right). \tag{11}$$

Above, s is the mean of the probability that the root candidate is a root and the probability that the affix candidate is an affix. t is the splitting position at which the word is divided into a root and affix i.e. $\{0, 1, 2, \ldots, T-1\}$. $word\left(x_t\right)$ is the probability that root candidate x_t (the word to the left of the splitting position t) is a root. $affix\left(z_t\right)$ is the probability that affix candidate z_t (the word to the right of the splitting position t) is an affix. s is used in the score function in (12). α is the threshold value, and 0.15 is used. n in (12) is the number of root-affix candidates. $affixVariety\left(x_t\right)$ is the number of all types of affixes which combine with root candidate $word\left(x_t\right)$.

$$score = \begin{cases} 0, & \text{if } s < \alpha \\ s, & \text{if } s \geq \alpha,\ n = 0 \\ \max\left(affixVariety\left(x_t\right)\right), & \text{if } s \geq \alpha,\ n > 1 \end{cases} \tag{12}$$

$word\left(x_t\right)$ and $affix\left(z_t\right)$ terms in Definition 4 are explained in detail via a cohesion probability [11] calculation method in Definition 5 and Definition 6, respectively. Cohesion probability [11] is a method which extracts words by numerically calculating the correlation of a syllable with the previous or next syllable in a word containing two or more syllables. In this study, rules for calculating a single syllable have been added to analyze roots with one syllable and affixes with one syllable.

Definition 5: word $\left(x_t\right)$ in Definition 4 is the conditional probability that x_t is a root. x_t is the word on the left side of the splitting position t when a syntactic word is split into a root-affix. t is the syntactic word's splitting position, and T is the number of syllables which make up the syntactic word. W is the number of all syntactic word types which appear in the text. k is the number of syllables which make up the partial word x_t which was split. When the splitting position t is $T-1$, this is a case of a single syllable root, so the probability that the root will appear as part of the entire word is calculated below.

$$word\left(x_t\right) = \begin{cases} \sqrt[2]{P(x_{T-1}|W)}, & \text{if } t = T-1 \\ \sqrt[k+1]{\prod_{T-1}^{1} P(x_{T-2}, \ldots, x_0 | x_{T-1}, \ldots, x_1)}, & \text{if } t < T-1 \end{cases} \tag{13}$$

Definition 6: affix $\left(z_t\right)$ in Definition 4 is the conditional probability that z_t is an affix. z_t is the word on the right side of the splitting position t when a syntactic word is split into a root-affix. t is the syntactic word's splitting position, and T is the number of syllables which make up the syntactic word. W is the number of all syntactic word types which appear in the text. k is the number of syllables which make up the partial word z_t which was split. When the splitting position t is 0,

this is a case where there is no affix, so the probability is not calculated. Otherwise, the probability is calculated as below.

$$affix\,(z_t) = \begin{cases} \sqrt[2]{P(z_1|W)}, & \text{if } t = 1 \\ \sqrt[k+1]{\prod_{t=1}^{T-1} P(z_2, \ldots, z_{T-1}|z_1, \ldots, z_{T-2})}, & \\ & \text{if } t > 1 \end{cases} \tag{14}$$

When the splitting position t is 1, there is a one-syllable affix, so the probability that it will appear in the entire word is calculated.

Definition 4's *affixVariety* (x_t) is a method of improving accessor variety [5] and is described in Definition 7. Accessor variety is a method which has been used to find word boundaries in Chinese text, which does not have spaces. Accessor variety uses the number of all syllables which can appear before or after a particular syllable. In this study, *affixVariety* is different in that it uses only the number of types of affixes which combine with a root. This approach allows us to more accurately select word boundaries. This is because when all syllables are used instead of affixes, improper splits are included in the number of cases, but when only the number of affix types is calculated, the improper splits can be removed.

Definition 7: Definition 4's *affixVariety* (x_t) is the number of types of affixes which combine with root candidate x_t when a syntactic word has been split into a root-affix at the splitting point t. The value is min-max scaled and normalized to a value between 0 and 1. Thus,

$$affixVariety\,(x_t) = \frac{affixVarietyNum\,(x_t) - affixVarietyMin}{affixVarietyMax - affixVarietyMin}. \tag{15}$$

affixVarietyNum (x_t) is the number of types of affixes which combine with root candidate x_t. *affixVarietyMin* is the minimum value of *affixVarietyNum* (x_t), and *ffixVarietyMax* is the maximum value of *affixVarietyNum* (x_t).

According to (12), when there are more than two root-affix candidates, *affixVariety* (x_t) is used to select the word boundary. Fig. 5 describes the selection of a word boundary using *affixVariety* (x_t).

As shown in Fig. 5, when the root and affix are split according to score, the (has, stopped) split occurs. This is because statistically "has" appears frequently. When only the probability values of the root and affix are used, the scores of words which statistically appear often will inevitably be high. *affixVariety* calculates the number of types of affixes which combine with the root to avoid the previously described error. The root of the syntactic word "has stopped" is "stopped," and it combines with several affixes other than "has." However, there is only one affix which can combine with the incorrectly split root "has stopped." The Fig. 5 in Korean is in the Fig. 7 of Appendix.

The Korean affix tokenizer proposed in this section is a tokenizing method which splits syntactic words into roots and affixes for word embedding using Skip-gram-KR. Only the affix grammatical forms are split from the roots, which has the advantage of preserving the words' sequential meaning.

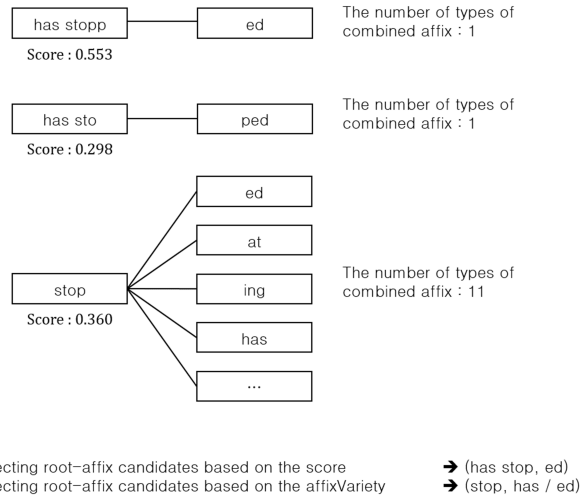

FIGURE 5. An example of selecting root-affix candidate using *affixVariety*.

IV. PERFORMANCE EVALUATION

This section compares the performance of the proposed method with that of an existing method. To do this, two types of experiments were performed. The first experiment compared the tokenizing accuracy of the Korean affix tokenizer with that of morpheme analyzers. Next, the second experiment compared the quality of the embedded words. The experiment results show that the Korean affix tokenizer's accuracy was the highest.

A. EXPERIMENT DATA

In these experiments, a web crawler was used to collect data from news articles about incidents of assault as well as data from Naver's Knowledge Encyclopedia and Dictionary. Table 2 shows a summary of the collected data. Here, the experiment data was limited to texts which followed the grammar and spacing of news and magazine articles.

TABLE 2. A summary of the collected data.

Corpus	Assault Incident News Articles	Knowledge Encyclopedia Dictionary
Number of Text	8,733	13,711
Time Period	Jan. 2010 ~ Sep. 2017	Aug. 2008 ~ Sep. 2017
Subject	Assaults	Architecture, Music, Art, Economy, Science, History, Literature, etc.
Size	7.3MB	139.7MB

B. EXPERIMENT RESULTS

This section presents the results of the Korean affix tokenizer accuracy experiments and the word embedding accuracy experiments.

1) EXPERIMENT 1: KOREAN AFFIX TOKENIZER ACCURACY

In this section, the word extraction accuracy of the Korean token analyzer which separates roots and affixes is compared to that of morpheme analyzers. The morpheme analyzers used in the experiment were the Hannanum morpheme analyzer, the Komoran morpheme analyzer, and a Twitter syntax analyzer. The experiment method was to randomly extract 100 sentences each from the knowledge encyclopedia and dictionary data and the assault incident news article data, and then compare the tokenizing accuracy of the extracted sentences.

When comparing tokenizing accuracy, the correct tokenizing data for the extracted sentences did not exist, so a technique was used for processing incorrect tokenizing as errors because if a single right answer is established when the evaluation is performed, it could favor or disfavor a certain tokenizer. For example, the proposed Korean affix tokenizer split ''controlled'' into (controll, ed), but the other analyzers split it into (control, l, ed). The reason this discrepancy occurred is because the proposed method does not change the forms of the words, but some of the other morpheme analyzers restore the words' original forms. Another example is that the Korean affix tokenizer extracted the word ''football referee'' as (football referee, P), but there were also morpheme analyzers which had a rule for classifying it as (football, referee). Therefore, to acknowledge such rule differences, only clear errors such as (horse racing, is) were processed as errors.

Table 3 presents the sentence tokenizing accuracy experiment results, and it shows the number of sentences in the two data sets which were tokenized without error. The Korean affix tokenizer had the highest accuracy with a mean value of 67 for the two data sets. For the assault incident news article data, the Komoran analyzer accurately tokenized the largest number of sentences at 76, and the proposed Korean affix tokenizer tokenized 74 sentences accurately, meaning it tokenized the sentences second most accurately to the Komoran analyzer. For all the knowledge encyclopedia data, the Korean affix tokenizer was able to tokenize a total of 60 sentences accurately, giving it the highest accuracy. The analyzer with the best performance for each data set is underlined.

In particular, the Korean affix tokenizer accurately extracted words from the names of people and places. The morpheme analyzers made the error of splitting words into syllable units when there was no information on a person or place's name, as can be seen in Table 4 and Table 5. Table 4 shows an example of a sentence with an accurately tokenized place name, and Table 5 shows an example for a person's name. The words containing errors are underlined. Also, the Table 4 and 5 in Korean are in the Table 6 and Table 7 of Appendix.

In Table 4, when the ''Aspen'' of ''Aspen Musical Festival'' was extracted as (A, spen) or (As, pen), it was processed as the wrong answer. The ''who, is'' in Table 4 could be seen as a wrong answer, but the ''인'' was reproduced as the affirmative copula ''is'' and the determiner type changing ending ''ㄴ'' so it was not the wrong answer. In Table 5, ''Hwang Wooman'' was extracted as (Hwangwoo, Man), and ''Hwang Woomin'' was extracted as (Hwangwoo, Min), which were seen as wrong answers. The Hannanum analyzer gave correct answers in both cases, but it made errors in extracting Table 4's ''composer'' (compos, er) and Table 5's ''weight lifter'' (weight lift, er).

TABLE 3. Tokenizing accuracy results.

Morpheme Analyzer	Assault Incident News Articles (7MB)	Entire Knowledge Encyclopedia (139.7MB)	Mean
Hannanum Morpheme Analyzer	46.0	43.0	44.5
Komoran Analyzer	**76.0**	56.0	66.0
Twitter Syntax Analyzer	42.0	42.0	42.0
Korean Affix Tokenizer	74.0	**60.0**	**67.0**

TABLE 4. Example of tokenizing a sentence that contains a place name.

Example	In 1960 in U.S. at the Aspen Music Festival and School again, Glass also studied under Darius Miyo, who is a composer and a member of the French group of Six	
	⇩	
Morpheme Analyzer	Sentence extracted by the analyzer	Wrong Answers
Hannanum Morpheme Analyzer	In 1960, in U.S., at, the Aspen, Summer, Music Festival, and, School again, Glass, studi, ed, under, Darius, Miyo, who, is, **a, composer,** and, a, member, of, the French, group, of, Six	Compos, er
Komoran Morpheme Analyzer	In, 1960, in U.S., at, **the, As pen**, Summer, Music Festival, and, School, again, Glass, studi, ed, under, Darius, Miyo, who, is, a, composer**,** and, a, member, of, the French group of Six	As, pen
Twitter Syntax Analyzer	In, 1960, in U.S., at, the, **As pen**, Summer, Music Festival, and School, again, Glass, studi, ed under Darius, Miyo, who, is, a composer and a member of the French, group of Six	As, pen
Korean Affix Tokenizer	In 1960, P, in U.S., P, at the, Aspen, P, Summer, Music Festival, P, and School, again, Glass, studi, ed under Darius, P, Miyo, who, is, a composer, P, and a member of the French, P, group of Six	-

2) EXPERIMENT 2: ACCURACY IN SEMANTIC CLUSTERING OF WORDS

In this section, 10 similar words are extracted for each of 10 embedded keywords, and the clustering accuracy of the similar words is compared. Here, the similar words are the words with the highest cosine similarity in the vector space.

TABLE 5. Example of tokenizing a sentence that contains a person's name.

Example	The weightlifter Hwang Wooman20's family argued that Sa Jaehyouk assaulted Hwang Woomin and dented his cheekbone at 11 p.m. on 31st December last year at a bar in Chuncheon, Gangwon-do, which would take 6 weeks to heal completely	
⇩		
Morpheme Analyzer	Sentence extracted by the analyzer	Wrong Answers
Hannanum Morpheme Analyzer	**The, weight, lifter**, Hwang Wooman 20's, family, argued, that, Jaehyouk Sa, assault, ed, Hwang Woomin, and, dented, his, cheekbone, at 11, p.m. on 31st, December, last year, at, a bar, in Chuncheon, Gangwon-do, which, would take, 6 weeks, to heal completely	The, weight, lifter
Komoran Morpheme Analyzer	The weightlifter, **Hwang, Wooman,**, 20's, family, argu, ed, that Sa Jaehyouk, assaulted, **Hwang, Woomin,** and, dented, his, cheekbone, at, 11, p.m., on 31st December, last year, at, a, bar, in Chuncheon, Gangwon-do, which, would, take, 6, weeks, to, heal, completely	Hwangwoo , Man Hwangwoo , Min
Twitter Syntax Analyzer	The weightlifter, **Hwang, Wooman,** 20's, family, argu, ed, that Sa Jaehyouk, assaulted, Hwang, Woomin, and, dented, his cheekbone, at, 11, p.m., on 31st, December, last year, at a, bar, in, Chuncheon, Gangwon-do, which would, take 6, weeks to, heal completely	Hwangwoo , Man
Korean Affix Tokenizer	The weightlifter, P, Hwang Wooman 20's, family argued, that Sa Jaehyouk, assaulted, Hwang Woomin, and dented his cheekbone at 11, P, p.m. on, P, 31st, P, December, last year, P, at a bar, in Chuncheon, P, Gangwon-do, which would, take, P, 6 weeks, to heal completely	-

The first step of the experimental method was to extract 10 words with a high degree of similarity for each of the keywords. Not all words could be compared, so the experiment was limited to the top 10 semantically similar words that were extracted. Next, a person confirmed whether or not the extracted words were semantically similar to the keywords by judging each true/false. Finally, the Korean semantic similarity evaluation index (Korean semantic top-k score, KST) was calculated according to the similarity rankings.

KST is an evaluation index first proposed in this study. By itself, calculating the number of top 10 similar words does not distinguish how close the word vectors are to the keyword. KST is a method which resolves the problem of ignoring the similarity rankings when simply calculating the number of similar words. It is described in Definition 8.

Definition 8: KST (Korean semantic top-k score) is defined in (16). It is a method which quantifies the ranking weights of words that are similar to keywords. K is the number of all extracted words, and k is the cosine similarity ranking.

$$KST = \begin{cases} \sum_{k=1}^{K} \dfrac{K+1-k}{K}, & if\ k = true \\ 0, & if\ k = false. \end{cases} \tag{16}$$

A score is given only if the correlation between the keyword and the similar word in the top-k is true. The KST score increases to the extent that words which are correlated with the keyword have high rankings.

TABLE 6. Comparison of semantic similarity of similar word and a certain keyword.

Model	Number of Similar Words (max of 10)	KST Score (max of 5.5)
Komoran + Skip-gram	5.7	3.58
Twitter + Skip-gram	5.7	3.45
Korean Affix Tokenizer + Skip-gram	6.2	3.97
Korean Affix Tokenizer + Skp-gram-KR	**6.7**	**4.05**

Table 6 shows the results of extracting and evaluating the semantic similarity of the top-10 words with the highest cosine similarity to 10 keywords which appear frequently in the assault incident news articles. The model with the highest number of similar words and the highest KST score is underlined. The embedding model which used the Korean affix tokenizer and Skip-gram-KR showed the best performance by recording a mean of 6.7 similar words out of a total of 10 and a KST score of 4.05 out of 5.5. In the case of Komoran + Skip-gram and Twitter + Skip-gram, the mean number of similar words was the same at 5.7, but Komoran had a higher KST score of 3.58.

Table 7 is an example of distinguishing the top-k similar words for "assault" and KST evaluations. The words which are similar to the "assault" keyword are underlined. Table 7 shows which words were clustered by the word embedding models for the "assault" keyword. In this case, the word embedding model which used the Korean affix tokenizer and Skip-gram-KR had the most accurate semantic word clustering with 9 similar words and a KST score of 5.1.

Table 8 shows the similar word evaluation results for "assault." This experiment was not included in Table 6's experiment. This is because the Komoran morpheme analyzer and the Twitter syntax analyzer did not extract a word in the form of "assault)." Table 8 shows the difference in word embedding between models which have the verb keyword "assault)," which is different from the noun keyword "assault." In Table 7, the proposed model extracted the concept words "violence," "beating," "assault incidence," and "revenge attack" as being similar words to the noun keyword "assault." However, in Table 8, action concepts such as "attack," "hit)," "beating," and "attack as a mob" were clustered for the "assault)" verb keyword.

The difference occurs due to the Korean affix tokenizer. That is, in cases where small differences in nouns and verbs

TABLE 7. Example of distinguishing top-10 similar words to noun "ASSAULT" and KST evaluations.

Keyword	Cosine Similarity Rank	Komoran + Skip-gram	Twitter + Skip-gram	Korean Affix Tokenizer + Skip-gram	Korean Affix Tokenizer + Skip-gram-KR
Assault	top1	**Beating**	**Beating**	**Beating**	**Was assaulted**
	top2	**Sexual harrassment**	**Abuse**	**Sexual abuse**	**Assaulted**
	top3	**Abuse**	**Sexual abuse**	**Mob violence**	**Beating**
	top4	**Attack**	Metaphor	**Violence**	**Verbal abuse**
	top5	**Rape**	**Attack**	**Insult**	**Violence**
	top6	**Sexual abuse**	**Fist blow**	**Harrassment**	**Assaulted**
	top7	Stab	**Hit**	Specified	Villager
	top8	**Surprise attack**	**Strike**	Baduki	**Assault**
	top9	**Strike**	Press	Ulilaterally	**Assault incidence**
	top10	Kidnap	**Sexual harrassment**	The other party	**Revenge attack**
	True KST	8 5.0	8 4.6	6 4.5	9 5.1

TABLE 8. Example of distinguishing top-10 similar words to verb "ASSAULT" and KST evaluations.

Keyword	Cosine Similarity Rank	Komoran + Skip-gram	Twitter + Skip-gram	Korean Affix Tokenizer + Skip-gram	Korean Affix Tokenizer + Skip-gram-KR
Assault	top1	-	-	**hit**	**assault**
	top2	-	-	**disturb**	**assault**
	top3	-	-	**harass**	**have assaulted**
	top4	-	-	**stab**	**hit**
	top5	-	-	**attack as mob**	**hit**
	top6	-	-	**push**	**beat**
	top7	-	-	**threaten**	**beating violently**
	top8	-	-	**threaten**	**attack as mob**
	top9	-	-	**beat**	**attack as mob**
	top10	-	-	**wield**	**attack as mob**
	True KST	- -	- -	10 5.5	10 5.5

must be distinguished, the Korean affix tokenizer can be used to increase the accuracy of word distinctions.

In this part, the words which are similar to the embedded words are extracted to check for semantic similarity. Extracting words with high correlation as similar words means the proposed method is clustering semantically similar words. Through experiments, it was possible to see that the Korean affix tokenizer and Skip-gram-KR placed contextually related words close together in the vector space. Also, the proposed KST evaluation index was used to express the similarity weighting numerically. Words embedded using the Korean affix tokenizer and Skip-gram-KR had a similarity score of 4.05 out of 5.5, which showed higher clustering accuracy than the other models.

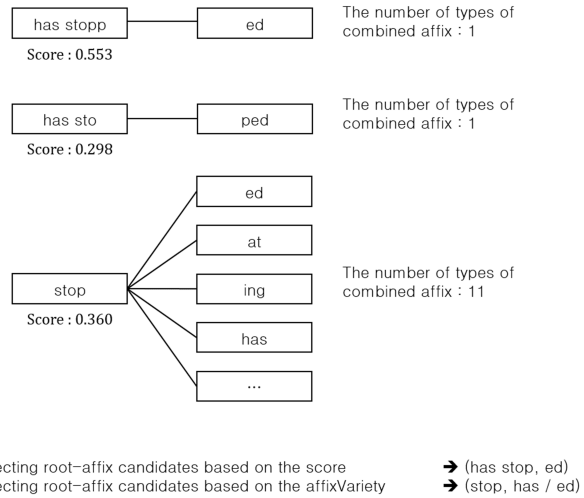

FIGURE 6. Fig. 4 in Korean.

In this part, the words which are similar to the embedded words are extracted to check for semantic similarity. Extracting words with high correlation as similar words means the proposed method is clustering semantically similar words. Through experiments, it was possible to see that the Korean affix tokenizer and Skip-gram-KR placed contextually related words close together in the vector space. Also, the proposed KST evaluation index was used to express the similarity weighting numerically. Words embedded using the Korean affix tokenizer and Skip-gram-KR had a similarity score of 4.05 out of 5.5, which showed higher clustering accuracy than the other models.

V. CONCLUSION

Word embedding is the first step toward using deep learning architectures in natural language processing. Today, word embedding models developed for understanding English are also being used in Korean word embedding processes owing to the absence of a word embedding model for Korean. However, given that English and Korean have differences in terms of linguistic structure and grammar, it is clear that existing word embedding models are limited in their ability to reflect the characteristics of the Korean language. To reflect these characteristics in the word embedding process, this paper has proposed Skip-gram-KR, which improves upon Skip-gram, and a Korean affix tokenizer, which is a method of extracting words for Skip-gram-KR.

Skip-gram-KR is a word embedding method which improves the similar word training data creation method by

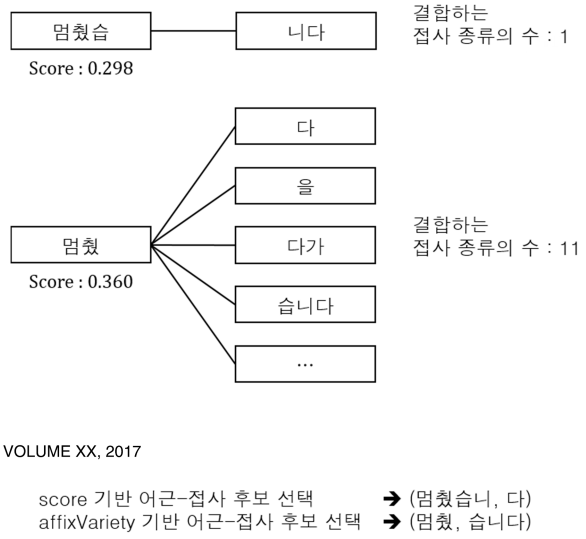

FIGURE 7. Fig.5 in Korean.

TABLE 9. Table 4 in Korean.

Example	1960 년 미국 아스펜 음악제의 여름 학교에서도 글래스는 프랑스 6 인조 가운데 하나인 작곡가 다리우스 미요를 사사했다

⇩

Morpheme Analyzer	Sentence extracted by the analyzer	Wrong Answers
Hannanum Morpheme Analyzer	1960 년, 미국, 아스펜, 음악제, 의, 여름, 학교, 에서도, 글래스, 는, 프랑스, 6 인조, 가운데, 하나, 이, ㄴ, **작곡, 가**, 다리우스, 미요, 를, 사사, 하, 었다	작곡, 가
Komoran Morpheme Analyzer	1960, 년, 미국, **아스, 펜**, 음악제, 의, 여름, 학교, 에서, 도, 글래스, 는, 프랑스 6 인조, 가운데, 하나, 이, ㄴ, 작곡가, 다리우스 미요, 를, 사사, 하, 았, 다	아스,펜
Twitter Syntax Analyzer	1960, 년, 미국, **아스, 펜**, 음악제, 의, 여름, 학교, 에서, 도, 글래스, 는, 프랑스, 6, 인조, 가운데, 하나, 인, 작곡가, 다리우스, 미요, 를, 사사, 했, 다	아스,펜
Korean Affix Tokenizer	1960 년, P, 미국, P, 아스펜, P, 음악제, 의, 여름, P, 학교, 에서도, 글래스, 는, 프랑스, P, 6 인조, P, 가운데, P, 하나, 인, 작곡가, P, 다리우스, P, 미요, 를, 사사했, 다	-

considering the word order and structure of Korean. Because Korean modifiers are mainly located before words with important meanings, a backward mapping method is used. This creates similar word training data pairs from only the words which appear before the word in the current position. Also, a two-word skipping method is used to overcome the problem of mapping words such as roots and affixes (which share context but do not have semantic correlation) as similar words.

TABLE 10. Table 5 in Korean.

Example	황우만 20 선수의 가족은 사재혁이 지난 12 월 31 일 오후 11 시께 강원 춘천시의 한 술집에서 황우민을 폭행해 얼굴 광대뼈가 함몰되는 등 전치 6 주의 중상을 입었다고 주장했다

⇩

Morpheme Analyzer	Sentence extracted by the analyzer	Wrong Answers
Hannanum Morpheme Analyzer	황우만 20, **서, ㄴ, 수**, 의, 가족, 은, 사재혁, 이, 지나, ㄴ, 12 월, 31 일, 오후, 11 시, 께, 강원, 춘천시, 의, 한, 술집, 에서, 황우민, 을, 폭행, 하, 어, 얼굴, 광대뼈, 가, 함몰, 되, 는, 등, 전치, 6 주, 의, 중상, 을, 입, 었다, 고, 주장, 하, 었다	서,ㄴ,수
Komoran Morpheme Analyzer	**황우, 만**, 20, 선수, 의, 가족, 은, 사재혁, 이, 지나, ㄴ, 12 월 31 일, 오후, 11, 시, 께, 강원, 춘천시, 의, 한, 술집, 에서, **황우, 민**, 을, 폭행, 하, 아, 얼굴, 광대뼈, 가, 함몰, 되, 는, 등, 전, 하, 지, 6, 주, 의, 중상, 을, 입, 었, 다고, 주장, 하, 았, 다	황우,만 황우,민
Twitter Syntax Analyzer	**황우, 만**, 20, 선수, 의, 가족, 은, 사재혁, 이, 지난, 12, 월, 31, 일, 오후, 11, 시, 께, 강원, 춘천시, 의, 한, 술집, 에서, 황우민, 을, 폭행, 해, 얼굴, 광대뼈, 가, 함몰, 되는, 등, 전치, 6, 주의, 중상, 을, 입었, 다고, 주장했, 다	황우,만
Korean Affix Tokenizer	황우만 20, P, 선수, 의, 가족, 은, 사재혁, 이, 지난, P, 12 월, P, 31 일, P, 오후, P, 11 시, 께, 강원, P, 춘천시, 의, 한, P, 술집, 에서, 황우민, 을, 폭행해, P, 얼굴, P, 광대뼈, 가, 함몰되, 는, 등, P, 전치, P, 6 주, 의, 중상, 을, 입었, 다고, 주장했, 다	-

The Korean affix tokenizer is a method of tokenizing Korean syntactic words as root-affix pairs. Unlike conventional word embedding which uses morphemes as words, the proposed method extracts roots and affixes as words. This method is a new word extraction method for Skip-gram-KR's two-word skipping structure. Because it separates only the affix from the syntactic word, it has the advantage of preserving the word's original meaning as much as possible. The method extracts neologisms and unknown words more accurately than conventional morpheme-based word extraction.

In experiments which tokenized sentences and compared word extraction accuracy, the proposed method achieved the most accurate performance. When random sentences were extracted and tokenized and the errors were analyzed, a mean of 67% sentences were tokenized without error. Also, to gauge its accuracy in semantic clustering of embedded words, 10 words with high cosine similarity to each keyword were extracted and their similarity was compared. In this experiment, KST was proposed for assigning weights to the cosine similarity rankings, and the results were quantified.

The proposed method showed the highest semantic clustering accuracy with a mean of 6.7 words out of 10 and a KST score of 4.05 out of 5.5.

Through experiments, it was shown that when the proposed method is used, the accuracy of dense vector expressions of Korean words is improved. If words which have been semantically clustered using this improved method are transferred to a deep learning architecture, it can be expected that this will improve the overall performance of the model.

APPENDIX

See Figures 6 and 7 and Tables 9 and 10.

REFERENCES

[1] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, pp. 1137–1155, Nov. 1985.
[2] G. Berardi, A. Esuli, and D. Marcheggiani, "Word embeddings go to Italy: A comparison of models and training datasets," in *Proc. Italian Inf. Retr. Workshop*, 2015, pp. 1–8.
[3] S. Choi, J. Seol, and S. G. Lee, "On word embedding models and parameters optimized for korean," in *Proc. 28th Conf. Korean Lang. Inf. Sci. Soc.*, 2016, pp. 252–256.
[4] C. Cinarel and B. T. Zhang, "Better word embedding for korean," in *Proc. Winter Conf. Korean Inst. Inf. Sci. Eng.*, 2016, pp. 627–629.
[5] H. Feng, K. Chen, X. Deng, and W. Zheng, "Accessor variety criteria for Chinese word extraction," *Comput. Linguistics*, vol. 30, pp. 75–93, Mar. 2004.
[6] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1243–1252.
[7] Y. Goldberg and O. Levy. (2014). "word2vec Explained: Deriving Mikolov *et al.*'s negative-sampling word-embedding method." [Online]. Available: https://arxiv.org/abs/1402.3722
[8] X. Jian, J. Liu, L. Zhang, Z. Li, and H. Chen, "Improve chinese word embeddings by exploiting internal structure," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2016, pp. 1041–1050.
[9] I. Koo, *Basic Korean Terms Dictionary for High School Students*. Seoul, South Korea: Shinwonbook, 2006.
[10] A. Kumar *et al.*, "Ask me anything: Dynamic memory networks for natural language processing," in *Proc. 33rd Int. Conf. Mach. Learn.*, New York, NY, USA, 2016, pp. 1378–1387.
[11] H. J. Kim, "Cleansing noisy text using corpus extraction and string match," M.S. thesis, Dept. College Eng., Seoul Nat. Univ., Seoul, South Korea, 2013.
[12] H. J. Kim, S. Cho, and P. Kang, "KR-WordRank: An unsupervised korean word extraction method based on WordRank," *J. Korean Inst. Ind. Eng.*, vol. 40, no. 1, pp. 18–33, 2014.
[13] G. Luo, X. Huang, C. Y. Lin, and Z. Nie, "Joint entity recognition and disambiguation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 879–888.
[14] L. Van Der Maaten, "Accelerating t-SNE using tree-based algorithms," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3221–3245, 2014.
[15] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. ICLR Workshop*, 2013, pp. 1–12.
[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. 26th Int. Conf. Neural Inf. Syst. (NIPS)*, 2013, pp. 3111–3119.
[17] S. Nam, Y. Hahm, and K. S. Choi, "Application of word vector with korean specific feature to Bi-LSTM model for named entity recognition," in *Proc. 29th Conf. Korean Language Inf. Sci. Soc.*, 2017, pp. 147–150.
[18] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Doha, Qatar, 2014, pp. 1532–1543.
[19] R. Socher *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proc. Conf. Empirical Methods Natural Language Processing (EMNLP)*, 2013, pp. 1631–1642.
[20] X. Rong. (2014). "word2vec parameter learning explained." [Online]. Available: https://arxiv.org/abs/1411.2738
[21] K. Shim, "Improving the performance of statistical korean morphological analyzer," in *Proc. Humanities Res. Inst. Sungshin Womans Univ.*, vol. 34, 2016, pp. 286–315. [Online]. Available: http://www.papersearch.net/thesis/article.asp?KEY=3419915
[22] A. Vaswani *et al.*, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NIPS)*, Long Beach, CA, USA, 2017, pp. 1–11.
[23] T. Young, D. Hazarika, S. Poria, and E. Cambria. (2017). "Recent trends in deep learning based natural language processing." [Online]. Available: https://arxiv.org/abs/1708.02709
[24] V. Yücesoy and A. Koç, "Co-occurrence weight selection in generation of word embeddings for low resource languages," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 18, no. 3, pp. 1–18, 2019.
[25] Y. Zhang, Z. Gan, and L. Carin, "Generating text via adversarial training," in *Proc. 30st Int. Conf. Neural Inf. Process. Syst. (NIPS)*, Barcelona, Spain, 2016, pp. 1–6.
[26] (2016). *CS224d: Deep Learning for Natural Language Processing*. [Online]. Available: http://cs224d.stanford.edu/
[27] *DooPedia*. Accessed: 2018. [Online]. Available: http://www.doopedia.co.kr

SUN-YOUNG IHM received the B.S. and M.S. degrees in multimedia science and the Ph.D. degree in IT engineering from Sookmyung Women's University, South Korea, in 2011, 2013, and 2017, respectively, where she is currently a Visiting Professor with the Department of IT Engineering, Engineering School. Her Ph.D. research includes efficient index building for multi-dimensional data.

She was with the Big Data Using Research Center, Sookmyung Women's University, as a Senior Researcher, from 2017 to 2018. Her research interests include machine learning, deep learning, data mining, text mining, natural language processing, top-k query, index building, and data analysis.

JI-HYE LEE received the B.A. degree in public relations and advertising and the M.S. degree in IT engineering from Sookmyung Women's University, in 2009 and 2018, respectively.

Her research interests include word embedding, text mining, and IT convergence with advertising and marketing.

YOUNG-HO PARK received the B.S. and M.S. degrees in computer engineering from Dongguk University, in 1990 and 1992, respectively, and the Ph.D. degree from the Department of Computer Science, Korea Advanced Institute of Science and Technology (KAIST), in 2005. His Ph.D. research includes efficient query processing in heterogeneous XML documents.

He was a Senior Research Staff at the ISDN Administration and Maintenance Division for TDX-10 ISDN, the Real-Time DBMS Division, and the Real-Time Operating System Division, Electronics and Telecommunication Research Institute (ETRI), from 1993 to 1999. He was also a Postdoctoral Researcher with the Advanced Information Technology Research Center (AITrc), KAIST, from 2005 to 2006, after receiving the Ph.D. degree. He is currently a Professor with the Department of IT Engineering, Engineering School, Sookmyung Women's University. Recently, his research interests include data science based on data analytics, data management systems (DBMS), information retrieval (IR), machine learning (ML), XML, and IT convergence with other fields, such as music, design, economy, business management, advertisement, and bio-informatics.