electronics

MDPI

Article

An Empirical Study of Korean Sentence Representation with Various Tokenizations

Danbi Cho, Hyunyoung Lee and Seungshik Kang *

Department of Computer Science, Kookmin University, 77 Jeongneung-ro, Seongbuk-gu, Seoul 02707, Korea; daanv319@kookmin.ac.kr (D.C.); hyunyoung2@kookmin.ac.kr (H.L.)
* Correspondence: sskang@kookmin.ac.kr

Abstract: It is important how the token unit is defined in a sentence in natural language process tasks, such as text classification, machine translation, and generation. Many studies recently utilized the subword tokenization in language models such as BERT, KoBERT, and ALBERT. Although these language models achieved state-of-the-art results in various NLP tasks, it is not clear whether the subword tokenization is the best token unit for Korean sentence embedding. Thus, we carried out sentence embedding based on word, morpheme, subword, and submorpheme, respectively, on Korean sentiment analysis. We explored the two-sentence representation methods for sentence embedding: considering the order of tokens in a sentence and not considering the order. While inputting a sentence, which is decomposed by token unit, to the two-sentence representation methods, we construct the sentence embedding with various tokenizations to find the most effective token unit for Korean sentence embedding. In our work, we confirmed: the robustness of the subword unit for out-of-vocabulary (OOV) problems compared to other token units, the disadvantage of replacing whitespace with a particular symbol in the sentiment analysis task, and that the optimal vocabulary size is 16K in subword and submorpheme tokenization. We empirically noticed that the subword, which was tokenized by a vocabulary size of 16K without replacement of whitespace, was the most effective for sentence embedding on the Korean sentiment analysis task.

Keywords: Korean sentence embedding; subword; tokenization; sentiment analysis

Citation: Cho, D.; Lee, H.; Kang, S. An Empirical Study of Korean Sentence Representation with Various Tokenizations. *Electronics* **2021**, *10*, 845. https://doi.org/10.3390/electronics10070845

Academic Editors: Chang Wook Ahn and Pankoo Kim

Received: 27 February 2021
Accepted: 29 March 2021
Published: 1 April 2021

1. Introduction

Embedding is a fundamental step when representing text data in vector space for natural language process tasks, namely, text classification, machine translation, and generation. To represent discrete data such as words and sentences in vector space, much research has been conducted on various embedding methods based on words, morphemes, and subwords. Most embedding research has focused on word embedding, which learns a vector from the word token unit [1–4]. However, word embedding is problematic due to unknown tokens raising the out-of-vocabulary (OOV) problem. To alleviate the OOV problem, morpheme embedding was introduced [5,6]. Since a word is decomposed into a sequence of morpheme tokens based on morphological meanings, morpheme embedding has fewer unknown tokens and is more robust to the OOV problem than word embedding [5,7]. Subword tokenization is an attractive approach in sentence embedding researches [8,9] and it acquired a good performance on the political-bias classification task [10]. The subword token is decomposed by data-driven statistical algorithms, for example, byte pair encoding (BPE) and SentencePiece (https://github.com/google/sentencepiece) (accessed on 9 December 2020) [11,12]. It was proved that subword embedding is effective for machine translation tasks [8,9], and the language models based on the transformer, namely, BERT and KoBERT, achieved state-of-the-art results on sentiment analysis by conducting subword tokenization [13,14]. However, it is not clear whether subword tokenization is an effective method for Korean sentence embedding. We raise the question: "What is the optimal token unit for Korean sentence embedding?" To find the answer, we explored sentiment analysis

task with sentence representation methods, which is known as sentence embedding, based on the various token units (word, morpheme, subword, and submorpheme).

The language models based on the transformer (e.g., BERT, KoBERT, and ALBERT) learn the pretraining tasks with the large corpora and subword tokenization for improving performance [13–15]. However, because we focused on determining whether subword tokenization is an important factor for improving performance, we controlled other factors for improving performance, excluding tokenization for sentence embedding. To control factors other than embedding methods, we used simple classifiers, such as a support vector machine (SVM) (https://www.cs.cornell.edu/people/tj/svm_light/) (accessed on 9 December 2020), multi-layer perceptron (MLP), and long short-term memory (LSTM). That is, we constructed the sentence embedding by applying the token sequence based on the various token units to the sentence representation methods and evaluated the performance of sentence embedding using the simple classifiers on a Korean sentiment analysis task. Our research is structured as follows. In Section 2, we introduce the related works of the sentence representation methods based on tokenization. In Section 3, we describe the properties of token units in the Korean language and how to obtain the sentence vector from the sequence of token units in a sentence. In Section 4, we evaluate the performance of the sentence embedding according to the token unit defining the sentence in the Korean sentiment analysis task. In Section 5, we analyze the experimental results, considering the properties of token units. Lastly, Section 6 summarizes our task and presents the expected effect.

2. Related Work

Most previous studies inputted the word token unit in the embedding models, namely, Word2vec and GloVe [1–3,16–18], to represent the word as a continuous vector. Tang (2014) and Severyn (2015) conducted the word embedding by inputting word tokens in skip-gram for the sentence classification task in English [1,2]. Zhao (2018) also explored the word embedding in English utilizing GloVe to eliminate the gender stereotype caused by the biased data [4]. Lee (2019) carried out CBOW for word embedding on the Korean spam message filtering task [3]. The morpheme has the advantage of expressing an internal meaning of a word because a word is decomposed into a sequence of morpheme tokens by morphological meaning [5–7]. In English, Botha (2014) verified that morpheme embedding is effective to represent the meaning of tokens than the word embedding, and Tang (2020) introduced the tokenization strategies for resolving the OOV problem based on morpheme embedding [5,7]. Lee (2018) combined morphological features in a word to demonstrate the effect of morpheme embedding in the Korean language [6]. Subword tokenization was recently introduced with BPE and SentencePiece algorithms [8–10]. Banerjee (2018) and Wang (2020) tackled the OOV problem by utilizing the BPE algorithm on the machine translation task using multiple languages [8,9]. Cho (2020a) explored subword embedding with SentencePiece for the bias classification task using the Korean news article dataset [10]. In addition, BERT was introduced by Devlin (2019) as the pretraining language model based on a transformer, and it was expanded to KoBERT and ALBERT [13–15]. These utilize the subword embedding to pretrain the large corpora, and these achieved the state-of-the-art results on various NLP tasks.

3. Sentence Representation with Tokenization

3.1. Tokenization

Tokenization lists a sentence as a token sequence. As shown in Table 1, tokenization results differ depending on the token unit. Thus, we describe the tokenization results with properties of the Korean language in the order of word, morpheme, subword, and submorpheme. In all tokenization, we did not deal with punctuation marks.

Table 1. Example of tokenization results according to token units for "너무나 감동적인 영화" (very touching movie); We marked the token, which is we can not represent in English, as "-" (hyphen).

Token Unit	Tokenization Result
Raw Text	너무나 감동적인 영화 (very touching movie)
Word	너무나/감동적인/영화 (very/touching/movie)
Morpheme	너무나/감동/적/인/영화 (very/touch/-/ing/movie)
Subword	_너무/나/_감동/적인/_영화 (_very/-/_touch/-ing/_movie)
Submorpheme	_너무/나/_감동/_적/_인/_영화 (_very/-/_touch/_-/_ing/_movie)

Word tokenization is the simplest method of decomposing a sentence by whitespace. For word tokenization, we did not handle whitespace as a particular symbol. As shown in Table 1, word tokenization represents "너무나 감동적인 영화" (very touching movie) as "너무나" (very), "감동적인" (touching), and "영화" (movie). As a word is a synthesis of morphemes in the Korean language, a word appears differently depending on which morphemes are combined. For example, a noun "영화" (movie) is represented as "영화가" or "영화는" when used with "가" or "는" as a morpheme representing the subject, and then "영화를" when used with "를" as a morpheme representing the object. ("가," "는," and "를" are Korean postpositions to indicate the role of a noun in a sentence). These properties of word tokenization have the disadvantages of high complexity of morphological meaning and many unknown tokens, causing the OOV problem. To improve the disadvantages of word tokenization, much research decomposes a word into a morpheme sequence.

The morpheme is the smallest token unit semantically and has the advantages of alleviating the complexity of morphological meaning and decreasing the number of unknown tokens. Morpheme tokenization represents the sentence of Table 1 as "너무나" (very), "감동" (touch), "적" (-), "인" (ing), and "영화" (movie). As shown in the result, we found that morpheme tokenization decomposes the sentence more finely than word tokenization. For example, "감동적인" (touching) in word tokenization is decomposed to "감동" (touch), "적" (-), and "인" (ing) in morpheme tokenization. In other words, because morpheme tokenization alleviates the complexity of morphological meaning in a word and decreases the number of unknown tokens, it is effective at resolving the OOV problem than the word.

A subword is decomposed by SentencePiece in our work. SentencePiece is a data-driven statistical algorithm based on a language model [11]. SentencePiece replaces the whitespace in a sentence with a particular symbol, "_" (underbar), to restore the sentence for the machine translation task [19], and decomposes a sentence into a subword sequence. For example, "너무나 감동적인 영화" is decomposed to "_너무" (_very), "나" (-), "_감동" (_touch), "적인" (-ing), and "_영화" (_movie) in the tokenization result of the subwords of Table 1. However, we found that "_영화" (_movie) and "영화" (movie) have different vectors, even though two tokens have the same meaning as "movie". We hypothesize that it is unnecessary to replace the whitespace in the sentiment analysis task because replacement is for restoring sentences in the machine translation task. To verify our hypothesis, we carried out the subword task in two cases: SWU (SubWord with Underbar) and SWT (SubWord Token without underbar). SWU is the original result of SentencePiece, so the tokenization results of SWU include a particular symbol for replacing the whitespace "_" (underbar). In contrast, SWT removes the symbol from SWU. It means that the SWT removes the symbol after applying a sentence to SentencePiece. For example, "_영화" (_movie) in SWU is transformed to "영화" (movie) in SWT.

A submorpheme is created by applying the morpheme sequence to the SentencePiece after tokenizing a sentence by the morphological analyzer. We expected that if we applied a morpheme sequence to SentencePiece, the performance of sentence embedding with subword tokenization would improve because it has been proven that morpheme-based sentence embedding outperforms word-based sentence embedding. Thus, we thought of a way to apply a morpheme sequence to SentencePiece instead of a word sequence, called the a submorpheme. For example, the tokenization results of submorpheme are represented as "_너무" (_very), "나" (-), "_감동" (_touch), "_적" (_-), "_인" (_ing), and "_영화" (_movie)

based on morpheme tokenization; "너무나" (very), "감동" (touch), "적" (-), "인" (ing), and "영화" (movie). As the subword tokenization, we explore submorpheme tokenization in two cases: SMU (SubMorpheme with Underbar) and SMT (SubMorpheme Token without underbar) in the same way as subword. That is, SMU is the result of applying a morpheme sequence to SentencePiece, whereas SMT removes the symbol from SMU.

3.2. Sentence Representation Methods

There are two methods with which to represent a sentence vector. One is averaging sequential token vectors of a sentence. This method makes the sentence vector by averaging the sequential token vectors without considering the order of tokens in the sentence. When a sentence consists of a token sequence $S = (t_1, t_2, ..., t_n)$, where t_i is the i-th token in the sentence, we compose the sentence vector V_S as $average(v_{t_1}, v_{t_2}, ..., v_{t_n})$, where v_{t_i} is the vector of i-th token t_i. The other considers the order of tokens in a sentence by sequentially inputting the token vector of sentence in LSTM. We carried out the two methods to compose a sentence vector according to whether the order of tokens in a sentence is considered. Before representing the sentence vector, we pretrained the downstream task dataset as a continuous vector by using skip-gram (https://code.google.com/archive/p/word2vec/) (accessed on 9 December 2020) [16,17]. We set up the hyperparameters: iteration 300, min-count 1, window size 5; and then vector sizes 200, 250, and 300, respectively. When pretraining the downstream task dataset, we used the only trainset.

In the sentence representation method ignoring the order of tokens in a sentence, we averaged the token vectors, which mapped the tokens of the input sentence to the pretrained vector, to compose a sentence vector. The sentence vector averaging the sequential token vectors was input into two classifiers; SVM and MLP. To verify the performance of the classifiers, we divided the trainset in a ratio of 8:2 in all experiments. SVM aims to maximize the distance between the decision boundary and support vector [20]. In the experiment of SVM, we used hinge loss from Equation (1) by setting Δ to 1 when minimizing the loss between actual score y and predicted score $\hat{y}$.

$$hinge\ loss = max\{0, \Delta - (\hat{y} \times y)\} \tag{1}$$

We implemented the MLP simplifying network with hidden units of half of the vector size and set up the hyperparameters as follows: *ReLU* as an activation function in the hidden layer; stochastic gradient descent (SGD) as an optimizer with a learning rate of 0.001, 32 batch size, and 100 epochs. We minimized the loss using the cross-entropy function through Equation (2). In the output layer, we used the *softmax* function to output the predicted probability for the input vector.

$$Cross\ Entropy\ Loss = -\sum_{i=1}^{n} y_i log(\hat{y}_i) \tag{2}$$

LSTM is effective to process sequential data, for example, the sensor data for detecting abnormal kick patterns on Taekwondo matches [21]. It is because the weight of LSTM in time t is trained by including the histories of previous time $\{(t-1), (t-2), ...\}$ for sequential data [22,23]. To consider the order of tokens in a sentence, which is likewise sequential data, we construct the sentence embedding by sequentially training the histories of previous times in LSTM. We input sequentially the token vector, which maps the token of sentence to pretrained vector, to LSTM. LSTM is constructed by hidden units of 128 and using *tanh* as the activation function. We output the sequential vectors of hidden units at every time t and concatenated them. The loss of LSTM also is minimized by the cross-entropy function from Equation (2), as for MLP. We divided the trainset into a ratio of 8:2 to validate the performance of the model and set up the hyperparameters of LSTM as SGD with a learning rate of 0.001, 32 batch size, and 100 epochs. LSTM outputs the probability for input sentence by a *softmax* function in the output layer.

4. Experiments

We empirically investigate the effective token unit for the Korean sentence embedding here. In detail, we focus on the following three research questions:

1. Which tokenization is more robust to the OOV problem?
2. Is the symbol replacing whitespaces meaningful in the Korean sentiment analysis?
3. What is the optimal vocabulary size for sentence embedding?

4.1. Dataset and Token Analysis

We were inspired by Cho (2020a), which is carried out the political-bias classification with the news article dataset, to find optimal tokenization for Korean sentences [10]. In Korean, the informal text includes many variations of token unlike the formal text such as the news article dataset. For example, "강력한 추천" (strong recommendation) is abbreviated as "강추". Additionally, "대박" (wow) is a coined word in Korean and "멋있다" (nice) makes a typo as "머싰다". Because these properties of informal text are heavily influenced by the token unit that makes up a sentence, we used the informal text, naver sentiment movie corpus (NSMC) (https://github.com/e9t/nsmc) (accessed on 2 December 2020), for the sentiment analysis task in Korean. The NSMC dataset has a trainset (150,000 reviews) and a testset (50,000 reviews), and it has binary sentiments of positive and negative [24,25]. A review of NSMC dataset consists of no more than 140 characters, so we regarded a review as a sentence in our work. In other words, we carried out the sentiment analysis using the NSMC dataset to evaluate the sentence embedding based on various token units (word, morpheme, subword, and submorpheme).

For the morpheme tokenization, we chose high-speed morphological analyzers, namely, Mecab, Okt (https://konlpy-ko.readthedocs.io/ko/v0.4.3/) (accessed on 2 December 2020), and KLT2000 that is run by index2018.exe (https://cafe.naver.com/nlpkang/3) (accessed on 2 December 2020) [26]. The tokenization results differ depending on which morphological analyzer is used. For example of Table 1, Mecab and Okt is decomposed into "너무나" (very), "감동" (touch), "적" (-), "인" (ing), and "영화" (movie), but KLT2000 is decomposed into "너무나" (very), "감동적" (touch-), and "영화" (movie). Thus, we compared the performances of sentiment analysis with sentence embedding utilizing three morphological analyzers.

We used the SentencePiece for subword tokenization and hypothesized that it is unnecessary to replace the whitespace of SentencePiece on the Korean sentiment analysis task. For subword tokenization, we carried out the SWU and SWT methods depending on whether the whitespace is replaced with a particular symbol, "_" (underbar). The SentencePiece algorithm for subword tokenization creates the vocabulary according to what vocabulary size is set. Cho (2020c) tested the SentencePiece with vocabulary sizes 50K, 75K, 100K, and 125K, and then the results showed a performance improvement when vocabulary size was small—50K [27]. To prove that the smaller the vocabulary size, the better the performance for vocabulary sizes smaller than 50K, we explored the vocabulary sizes 2K, 4K, 8K, 16K, and 32K. For the submorpheme tokenization, we utilized the same morphological analyzers with morpheme tokenization (Mecab, Okt, and KLT2000) and SentencePiece. As we expected that submorpheme tokenization has the same trend of subword tokenization, we explored the sentence embedding with submorpheme tokenization in two cases, namely, SMU and SMT. As for subword, we explored the vocabulary sizes 2K, 4K, 8K, 16K, and 32K in the experiments of submorpheme tokenization with Okt and KLT2000. In submorpheme tokenization with Mecab, we tested the vocabulary sizes 2K, 4K, 8K, and 16K because vocabulary size 32K does not work when a morpheme sequence using Mecab is applied to SentenePiece.

We confirmed the data distribution of the NSMC dataset using each token unit (e.g., the number of tokens, OOV rate, and average token length). In Table 2, $N(token)$ refers to the number of tokens in the trainset or testset of the NSMC datatset. OOV rate is the ratio of the number of unknown tokens to the number of testset tokens, as in Equation (3). Avg_length is the average length of tokens in the trainset or testset of the NSMC dataset.

The average length of tokens is composed by the sum of length of all token over the number of total tokens in the dataset as Equation (4), where n is the number of total tokens in the dataset and t_i is the i-th token in the total tokens. For example, "너무나/감동적인/영화" by word tokenization has an average token length of $\frac{3+4+2}{3} = 3$ because this example has token lengths of 3(너무나), 4(감동적인), and 2(영화), and then n is 3.

$$\text{OOV rate} = \frac{N(\text{unknown token})}{N(\text{testset})} \tag{3}$$

$$\text{Avg\_length} = \frac{\sum_{i=1}^{n} length(t_i)}{n} \tag{4}$$

Table 2. Data distribution for tokenization.

Tokenization			N(token)		OOV Rate	Avg_Length	
			Trainset	Testset		Trainset	Testset
Word			1,137,736	380,472	26.48	3.778	3.778
Morpheme	Mecab		2,750,801	921,106	1.032	1.562	1.560
	Okt		2,306,158	771,970	2.56	1.993	1.991
	KLT2000		1,399,134	468,458	6.53	2.337	2.333
Subword	SWU	2K	3,986,624	1,332,618	0.028	1.363	1.364
		4K	2,951,483	986,317	0.047	1.842	1.843
		8K	2,536,203	849,876	0.077	2.143	2.139
		16K	2,262,086	762,284	0.121	2.403	2.385
		32K	2,062,594	701,982	0.202	2.635	2.590
	SWT	2K	3,930,835	1,314,239	0.028	1.093	1.094
		4K	2,934,992	980,809	0.044	1.464	1.466
		8K	2,528,770	847,393	0.068	1.700	1.696
		16K	2,257,431	760,706	0.105	1.904	1.890
		32K	2,058,975	700,666	0.173	2.087	2.051
Submorpheme	Mecab_SMU	2K	4,143,841	1,385,508	0.029	1.701	1.702
		4K	3,475,644	1,163,011	0.045	2.028	2.028
		8K	3,338,314	1,118,941	0.057	2.111	2.108
		16K	3,291,225	1,104,407	0.084	2.142	2.136
	Mecab_SMT	2K	4,101,705	1,371,610	0.029	1.048	1.048
		4K	3,450,677	1,154,684	0.042	1.246	1.245
		8K	3,315,186	1,111,229	0.053	1.296	1.294
		16K	3,268,987	1,097,026	0.071	1.315	1.310

Table 2 shows that the larger the number of tokens in trainset and testset, the lower the OOV rate. Besides, in subword and submorpheme tokenizations, we found that minor differences between the data distributions of SWU and SWT (or SMU and SMT), even between the vocabulary sizes. In Table 2, the word tokenization has the smallest number of tokens and a higher OOV rate of 26.48% compared to other token units. This means the word tokenization is not robust to the OOV problem caused by unknown tokens. Among morpheme tokenizations, the KLT2000 showed the smallest number of tokens and the highest OOV rate of 6.53%, whereas the Mecab and Okt had the larger numbers of tokens and smaller OOV rates of 1.032% and 2.56%, respectively, compared to KLT2000. In subword tokenizations, SWU and SWT show similar data distributions, and then SWT shows a slightly lower OOV rate than SWU. We expect that SWT eliminates some noise by duplicated token due to the symbols such as "_영화" (_movie) and "영화" (movie). We also found that both SWU and SWT in subword tokenization have large numbers of tokens and lower OOV rates with small vocabularies, but their difference is trivial. The submorpheme tokenization is similar to the trend of subword tokenization. Table 2 shows a representative data distribution of submorpheme with Mecab. The specific data distributions of the submorpheme unit, including Okt and KLT2000, are presented in Table A1.

4.2. Experiments and Results

To evaluate the sentence embedding based on various token units, we carried out the Korean sentiment analysis task using SVM, MLP, and LSTM as classifiers. Table 3 shows the sentiment analysis accuracy of sentence embedding based on word and morpheme units. Tables 4 and 5 show the sentiment analysis accuracy of sentence embedding based on subword and submorpheme units, respectively. In Table 3–5, these indicate the performance according to the vector sizes 200, 250, and 300. Overall, the accuracy of LSTM was better than those of SVM and MLP. This means that the sentence representation method considering the order of tokens in the sentence is more effective than not considering the order. Thus, based on the performances of LSTM, we analyzed the experimental results of sentence embedding.

As shown in Table 3, morpheme-based sentence embedding outperforms word-based sentence embedding. Word-based sentence embedding showed an accuracy of 81.27%, but it was significantly less accurate than morpheme-based sentence embedding. Among the morphological analyzers for morpheme-based sentence embedding, when we utilized Mecab, the morpheme-based sentence embedding achieved the best accuracy at 85.39%.

Table 3. Sentiment analysis accuracy with the sentence representation methods based on word and morpheme tokenization.

Tokenization		SVM			MLP			LSTM		
		200	250	300	200	250	300	200	250	300
Word		71.84	77.79	66.25	76.31	77.53	68.76	***81.27***	80.8	80.81
Morpheme	Mecab	81.08	82.06	82.07	81.06	82.08	81.95	85.32	***85.39***	84.96
	Okt	81.86	83.13	82.82	82.83	83.16	82.44	***85.17***	82.94	85.11
	KLT2000	82.07	81.05	82.36	81.81	82.4	82.24	83.86	***84.07***	83.91

Table 4 shows the performances of sentiment analysis using the subword-based sentence embedding according to vocabulary sizes comparing the SWU and SWT. As shown in Table 4, we found the two key points. First, SWT outperformed SWU among the subword-sentence embedding methods. Second, the performance improved when the vocabulary size was large. The sentence embedding based on SWT achieved 85.67% accuracy, whereas the sentence embedding based on SWU achieved 85.42% accuracy. The difference between performances of sentence embedding utilizing SWU and SWT was trivial, but the sentence embedding based on SWT indicates higher accuracy than the sentence embedding based on SWU. In the comparison of vocabulary sizes, the sentence embedding based on SWU and SWT achieved the accuracies of 81.57% and 81.59% for vocabulary size 2K, respectively, whereas for vocabulary size 32K, the sentence embedding based on SWU and SWT achieved the accuracies of 85.34% and 85.52%. Although sentence embedding with SWU and SWT achieved the best accuracies of 85.42% and 85.67% for vocabulary size 16K, respectively, we found a tendency that the larger the vocabulary size improves performance in Table 4.

Table 4. Sentiment analysis accuracy with the sentence representation methods based on subword tokenization.

Tokenization		SVM			MLP			LSTM		
		200	250	300	200	250	300	200	250	300
SWU	2K	78.77	79.02	79.26	78.91	79.04	79.32	81.57	81.56	81.44
	4K	81.52	81.6	81.78	81.89	81.93	82.17	84.49	84.13	83.98
	8K	82.87	82.88	82.96	83.06	83.16	83.32	85.23	84.99	84.98
	16K	83.09	83.11	83.23	83.38	83.49	83.57	***85.42***	85.33	85.35
	32K	83.19	83.22	83.33	83.48	83.6	83.69	85.34	84.96	84.69
SWT	2K	63.67	74.75	76.09	65.67	78	76.33	81.59	81.57	81.58
	4K	78.99	78.72	78.77	79.5	79.21	78.84	84.12	83.8	83.26
	8K	82.77	82.75	83.03	83.13	82.96	83.12	85.05	85.18	84.81
	16K	83.13	83.06	83.33	83.43	83.33	83.68	***85.67***	84.93	85.27
	32K	83.27	83.43	83.56	83.58	83.57	83.94	85.52	85.24	84.71

We confirmed the performance of sentence embedding based on submorpheme tokenization in Table 5. First of all, unlike our expectation that submorpheme-based sentence embedding outperforms subword-based sentence embedding, submorpheme-based sentence embedding had lower performance than subword-based sentence embedding. The submorpheme-based sentence embedding with Okt_SMU achieved 85.32% accuracy as its best performance, whereas the subword-based sentence embedding achieved the better performance of 85.67% in SWT. We further found three tendencies. First, among the morphological analyzers utilized in submorpheme tokenizations, Okt showed better performance than the other morphological analyzers. Second, SMU outperformed SMT among the submorpheme-based sentence embedding methods, unlike the experimental results of subword-based sentence embedding. Lastly, the performance improved when the vocabulary size was large, similarly to the subword-based sentence embedding. In Table 5, the sentence embedding based on SMU indicated the performances of 84.83%, 85.32%, and 84.82% in Mecab, Okt, and KLT2000, respectively, whereas the sentence embedding based on SMT indicated the performances of 84.74%, 85.16%, and 84.83%. The performance of submorpheme-based sentence embedding was improved when the vocabulary size was 32K compared to 2K, just like subword-based sentence embedding.

Table 5. Sentiment analysis accuracy with the sentence representation methods based on submorpheme tokenization.

Tokenization		SVM			MLP			LSTM		
		200	250	300	200	250	300	200	250	300
Mecab_SMU	2K	79.11	79.4	79.57	79.25	79.4	79.67	82.74	82.87	82.44
	4K	81.02	81.3	81.47	81.38	81.55	81.72	84.55	84.72	84.76
	8K	81.05	81.27	81.34	81.41	81.51	81.6	84.56	84.79	84.75
	16K	80.71	81.23	81.48	81.15	81.45	81.74	84.76	***84.83***	84.43
Mecab_SMT	2K	74.3	76.68	78.19	74.97	76.85	79.27	82.12	82.05	81.29
	4K	79.65	79.65	79.47	80.84	79.03	80.9	84.1	84.17	84.24
	8K	81.03	79.32	77.92	80.87	78.16	79.15	84.35	84.34	84.39
	16K	79.87	80.88	80.4	80.15	81.5	79.58	84. 48	84.60	***84.74***
Okt_SMU	2K	78.91	79.13	79.45	79.1	79.38	79.43	82.37	82.23	82.32
	4K	81.44	81.59	81.71	81.83	82	81.98	84.34	84.52	83.7
	8K	82.11	82.32	82.52	82.53	82.72	82.94	***85.32***	84.94	85.03
	16K	82.49	82.56	82.83	82.76	83.19	83.18	85.0	85.25	84.76
	32K	82.72	82.74	82.86	83.04	83.11	83.37	85.01	84.5	83.96
Okt_SMT	2K	76.36	77.39	68.06	78.75	78.9	71.39	82.01	82.03	81.33
	4K	80.62	81.23	81.55	81.36	81.66	81.47	83.69	84.03	84.05
	8K	82.05	82.57	82.48	82.62	83.06	82.68	84.73	85.10	84.81
	16K	82.38	82.66	82.83	82.87	83.12	83.23	85.06	84.77	***85.16***
	32K	82.49	82.59	82.66	82.9	83.12	83.1	84.93	84.61	84.77
KLT2000_SMU	2K	78.35	78.44	78.68	78.52	78.71	78.92	81.62	81.44	81.3
	4K	80.84	80.91	81.34	81.43	81.32	81.64	83.84	83.8	83.3
	8K	82.07	82.02	82.03	82.6	82.28	82.46	84.69	84.22	84.5
	16K	82.14	82.08	82.37	82.45	82.49	82.81	***84.82***	84.27	84.24
	32K	82.21	82.23	82.36	82.58	82.76	82.96	84.55	84.47	84.48
KLT2000_SMT	2K	77.9	76.13	76.26	77.84	77.05	76.96	81.04	81.3	80.76
	4K	80.75	81.23	80.98	81.42	81.22	81.4	83.57	83.49	83.05
	8K	81.93	81.93	82.41	82.31	82.54	82.95	84.49	84.5	84.46
	16K	82.21	82.38	82.43	82.58	82.88	82.85	84.66	***84.83***	84.47
	32K	82.21	82.24	82.35	82.42	82.51	82.86	84.35	84.31	84.26

As shown in Tables 3–5, the best tokenization method for the Korean sentence embedding is SWT of subword tokenization with vocabulary size 16K—85.67% accuracy. Multi-lingual BERT and KoBERT achieved 87.5% and 90.1% accuracy, respectively, on the sentiment analysis using the NSMC dataset. However, our method is competitive with multi-lingual BERT and KoBERT because the capacities of multi-lingual BERT and KoBERT have a lot of computation by conducting the pretraining tasks with large corpora, whereas

our method has a small amount of computation with simple classifiers. Our research further is valuable in suggesting the most efficient token units for sentence embedding.

5. Analysis and Discussion

We can now answer the three questions on the data distribution and performance of sentence embedding on the Korean sentiment analysis task.

1. Which tokenization is more robust to the OOV problem?
2. Is the symbol replacing whitespaces meaningful in the Korean sentiment analysis?
3. What is the optimal vocabulary size for sentence embedding?

It is known that word embedding causes the OOV problem and that morpheme embedding is effective at alleviating the OOV problem. Considering those facts, we compared the performances of sentence embedding based on word, morpheme, subword, and submorpheme tokenization by correlation with OOV rate. As shown in Table 2, the OOV rate was the lowest, from 0.028% to 0.202%, in subword tokenization, followed by the submorpheme tokenization, from 0.029% to 0.084%, the morpheme tokenization, from 1.032% to 6.53%, and then the word tokenization (the highest) at 26.48%. The performances of subword-based sentence embedding outperformed those of the sentence embedding based on other tokenizations as shown in Tables 3 and 4. Although the submorpheme tokenization had a lower OOV rate like the subword tokenization, the performance of submorpheme-based sentence embedding (at 85.32% accuracy) was lower than that of subword-based sentence embedding (at 85.67% accuracy). We expect that submorpheme tokenization loses the syntactic and semantic meanings of a token because it is too finely decomposed by the combination of data-driven algorithm and morphological analyzer. Through the result that performance is generally improved when the OOV rate is low, we got the solution to: "Which tokenization is more robust to the OOV problem?" Subword tokenization is more robust to the OOV problem than other tokenizations.

In subword and submorpheme tokenizations, we confirmed the ratio of duplicated tokens due to the symbol "_ (underbar)" in SWU to analyze the effect of replacing the whitespace with a particular symbol. The duplicated token rate was calculated from Equation (5)—the difference in the number of tokens in SWU and SWT over the number of tokens in SWU.

$$\text{Duplicated token rate}(\%) = \frac{N(\text{SWU}) - N(\text{SWT})}{N(\text{SWU})} \times 100 \quad (5)$$

We obtained the duplicated token rates in the trainset of each vocabulary size. The results show 12% ($\pm$4.755), which is the mean $\pm$ standard deviation for the duplicated token rate of each vocabulary size. It means that SWU learns noise of 12% ($\pm$4.755) in the trainset and it supports the result that SWT outperforms SWU among the subword-based sentence embedding methods. Thus, we conclude that the replacement with symbol is not effective in the Korean sentence embedding. In the submorpheme tokenizations, however, the results are different from the subword tokenization. SMU had better performance than SMT among the submorpheme-based sentence embedding methods. We expect that was because the submorpheme tokenization proceeds with the additional tokenization by SentencePiece from the morpheme tokens that have already been decomposed by morphological analyzer; the symbol of submorpheme tokenization does not have a role for whitespace.

Additionally, while expanding the experiment of Cho (2020c), we obtained the solution to: "What is the optimal vocabulary size for Korean sentence embedding?" Cho (2020c) empirically concluded that the performance improves as the vocabulary size lowers, while experimenting with vocabulary sizes 50K, 75K, 100K, and 125K [27]. To verify that a vocabulary size smaller than 50K has better performance, we tested the subword and submorpheme tokenization with vocabulary sizes 2K, 4K, 8K, 16K, and 32K. The experimental results showed a tendency to improve performance when the vocabulary size was

large in Tables 4 and 5, contrary to the conclusion of Cho (2020c). To analyze these results, we confirmed the average token lengths of our experiments (vocabulary sizes 2K, 4K, 8K, 16K, and 32K) and those of Cho (2020c) (vocabulary sizes 50K, 75K, 100K, and 125K). As shown in Table 2, the average token length increases when vocabulary size increases in both subword and submorpheme tokenizations. Vocabulary sizes 50K, 75K, 100K, and 125K, which are tested in Cho (2020c), got the average token lengths of 2.769, 2.889, 2.962, and 3.016 for the trainset of SWU, respectively. With the trainset of SWT, the vocabulary sizes got the average token lengths of 2.193, 2.288, 2.346, and 2.389, respectively. In our work, the subword-based sentence embedding in both SWT and SWU had the best accuracy with vocabulary size 16K, and the average token length closed to about 2 or 2.5, even with submorpheme-based sentence embedding. When analyzing the results with our average token length, we came to a new conclusion that the optimal vocabulary size is 16K because the average token length closes to about 2 or 2.5.

6. Conclusions

To find the optimal token unit for constructing the Korean sentence vector, we carried out the Korean sentiment analysis task according to the sentence embedding with various token units: word, morpheme, subword, and submorpheme. When representing the sentence vector from the token unit sequence, we carried out the two sentence representation methods: considering the order of tokens in the sentence or not. We empirically answered that when the SWT of subword tokenization is utilized in the sentence representation method considering the order of tokens in a sentence, the performance is best at 85.67% accuracy. We investigated the properties of token units to analyze our experimental results and found some key points. (1) Subword tokenization is more robust to the OOV problem than morpheme tokenization because of the lower OOV rate. (2) Replacing the whitespace with a particular symbol is not effective for subword and submorpheme tokenization for the Korean sentiment analysis task. (3) In the tokenizations utilizing the SentencePiece algorithmm such as subword and submorpheme tokenization, the vocabulary size 16K achieved the best performance because the average token length closed to about 2 or 2.5. We expect that our research will present the foundations for research on effective sentence embedding with simple computations.

Author Contributions: Conceptualization, H.L.; methodology, D.C. and H.L.; software, D.C.; validation, D.C. and H.L.; formal analysis, D.C.; investigation, D.C. and H.L.; data curation, D.C. and H.L.; writing—original draft preparation, D.C.; writing—review and editing, D.C., H.L., and S.K.; visualization, D.C.; supervision, H.L. and S.K.; project administration, S.K.; funding acquisition, S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Ministry of the Republic of Korea and the National Research Foundation of Korea (NRF-2019S1A5A2A03046571), and this research was supported by The sports promoting fund of the Korean Sports Promotion Foundation (KSPO) from the Ministry of Culture, Sports and Tourism.

Data Availability Statement: Data available in a publicly accessible repository that does not issue DOIs at https://github.com/e9t/nsmc (accessed on 2 December 2020).

Acknowledgments: This paper is an extension of the paper presented at the SMA2020 conference under the title of "Subword-based Sentence Representation Model for Sentiment Classification".

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

We present the data distribution of each tokenization, namely, word, morpheme, subword, and submorpheme as shown in Table 2. Table 2 shows only data distribution of submorpheme with Mecab, excluding Okt and KLT2000. Thus, we present the data distribution of submorpheme including the Mecab, Okt, and KLT2000 in Table A1.

Table A1. Data Distribution for Submorpheme Tokenization in detail.

Tokenization			N(Token)		OOV Rate	Avg_Length	
			Trainset	Testset		Trainset	Testset
Submorpheme	Mecab_SMU	2K	4,143,841	1,385,508	0.029	1.701	1.702
		4K	3,475,644	1,163,011	0.045	2.028	2.028
		8K	3,338,314	1,118,941	0.057	2.111	2.108
		16K	3,291,225	1,104,407	0.084	2.142	1.136
	Mecab_SMT	2K	4,101,705	1,371,610	0.029	1.048	1.048
		4K	3,450,677	1,154,684	0.042	1.246	1.245
		8K	3,315,186	1,111,229	0.053	1.296	1.294
		16K	3,268,987	1,097,026	0.071	1.315	1.310
	Okt_SMU	2K	3,976,273	1,330,247	0.028	1.623	1.623
		4K	2,992,410	1,001,011	0.048	2.157	2.157
		8K	2,683,202	899,937	0.071	2.405	2.399
		16K	2,531,714	851,292	0.105	2.549	2.537
		32K	2,460,044	829,834	0.17	2.624	2.602
	Okt_SMT	2K	3,940,007	1,318,202	0.028	1.091	1.090
		4K	2,985,670	998,713	0.044	1.440	1.439
		8K	2,680,690	899,083	0.062	1.603	1.599
		16K	2,530,037	850,718	0.089	1.699	1.690
		32K	2,458,502	829,318	0.13	1.748	1.733
	KLT2000_SMU	2K	3,051,372	1,019,474	0.034	1.530	1.531
		4K	2,188,325	731,568	0.057	2.133	2.134
		8K	1,906,344	639,554	0.095	2.449	2.441
		16K	1,748,764	589,289	0.162	2.669	2.649
		32K	1,659,831	563,303	0.27	2.812	2.772
	KLT2000_SMT	2K	3,013,758	1,007,099	0.034	1.085	1.085
		4K	2,182,286	729,598	0.055	1.498	1.498
		8K	1,904,128	638,851	0.087	1.717	1.711
		16K	1,747,658	588,951	0.145	1.871	1.856
		32K	1,658,897	562,998	0.23	1.971	1.941

References

1. Tang, D.; Wei, F.; Yang, N.; Zhou, M.; Liu, T.; Qin, B. Learning Sentiment-specific Word Embedding for Twitter Sentiment Classification. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MA, USA, 22 June 2014; Volume 1, pp. 1555–1565.
2. Severyn, A.; Moschitti, A. Twitter Sentiment Analysis with Deep Convolutional Neural Networks. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Sandiago, Chille, 9–13 August 2015; pp. 959–962.
3. Lee, H.; Kang, S. Word Embedding Method of SMS Message for Spam Message Filtering. In Proceedings of the 2019 IEEE International Conference on Big Data and Smart Computing, Kyoto, Japan, 27 February–2 March 2019; pp. 1–4.
4. Zhao, J.; Zhou, Y.; Li, Z.; Wang, W.; Chang, K. Learning Gender-neutral Word Embeddings. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 4847–4853.
5. Tang, Y.; Tang, C.; Zhu, C. Resolve Out of Vocabulary with Long Short-Term Memory Networks for Morphology. In Proceedings of the 2020 IEEE International Conference on Artificial Intelligence and Computer Applications, Dalian, China, 27–29 June 2020; pp. 115–119.
6. Lee, D.; Lim, Y.; Ted Kwon, T. Morpheme-based efficient Korean word embedding. *J. Korea Inf. Sci. Soc.* **2018**, *45*, 444–450.
7. Botha, J.; Blunsom, P. Compositional Morphology for Word Representations and Language Modelling. In Proceedings of the International Conference on Machine Learning, Beijing, China, 21–26 June 2014; Volume 32, pp. 1899–1907.
8. Banerjee, T.; Bhattacharrya, P. Meaningless yet Meaningful: Morphology Grounded Subword-level NMT. In Proceedings of the Second Workshop on Subword/Character level Models, New Orleans, LA, USA, 6 June 2018; pp. 55–60.
9. Wang, C.; Cho, K.; Gu, J. Neural Machine Translation with Byte-level Subwords. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 9154–9160.
10. Cho, D.; Lee, H.; Jung, W.; Kang, S. Automatic classification and vocabulary analysis of political bias in news articles by using subword tokenization. *J. Kips Trans. Softw. Data Eng.* **2020**, *10*, 1–8.
11. Kudo, T. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; pp. 66–75.
12. Sennrich, R.; Haddow, B.; Birch, A. Neural Machine Translation of Rare Words with Subword Units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 1715–1725.

13. Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
14. Korean BERT (KoBERT). Available online: https://github.com/SKTBrain/KoBERT (accessed on 25 March 2021).
15. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *arXiv* **2019**, arXiv:1909.11942.
16. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, US, 5–8 December 2013; pp. 3111–3119.
17. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. In Proceedings of the 1st International Conference on Learning Representations, Scottsdale, AZ, USA, 2–4 May 2013.
18. Pennington, J.; Socher, R.; Manning, C. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
19. Domingo, M.; Garcia-Marinez, M.; Helle, A.; Casacuberta, F.; Herranz, M. How much does Tokenization Affect Neural Machine Translation? *arXiv* **2018**, arXiv:1812.08621.
20. Pang, B.; Lee, L.; Vaithyanathan, S. Thumbs up? Sentiment Classification using Machine Learning Techniques. In Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing, Philadelphia, PA, USA, 6–7 July 2002; pp. 79–86.
21. Cho, D.; Lee, H.; Kang, S. A Study on the Detection of Anomalous Kicks in Taekwondo games by using LSTM. In Proceedings of the Korea Information Processing Society 2020, Online, 6–7 November 2020b; pp. 1025–1027.
22. Karim, F.; Majumdar, S.; Darabi, H.; Chen, S. LSTM fully convolutional networks for time series classification. *IEEE Access* **2017**, *6*, 1662–1669. [CrossRef]
23. Rao, G.; Huang, W.; Feng, Z.; Cong, Q. LSTM with sentence representations for document-level sentiment classification. *Neurocomputing* **2018**, *308*, 49–57. [CrossRef]
24. Kim, G.; Lee, C. Korean Text Generation and Sentiment Analysis Using Model Combined VAE and CNN. In Proceedings of the Annual Conference on Human and Language Technology, Seoul, Korea, 12–13 October 2018; pp. 75–78.
25. Lee, S.; Jang, H.; Baik, Y.; Park, S.; Shin, H. Kr-bert: A Small-scale Korean-Specific Language Model. *arXiv* **2020**, arXiv:2008.03979.
26. Won, H.; Lee, H.; Kang, S. A Performance Comparison of Korean Morphological Analyzers for Large-scale Text Analysis. In Proceedings of the Korean Computer Congress 2020, Online, 2–4 July 2020; pp. 401–403.
27. Cho, D.; Lee, H.; Kang, S. Subword-based Sentence Representation Model for Sentiment Classification. In Proceedings of the 9th International Conference on Smart Media and Applications, Jeju, Korea, 17–19 September 2020.