Subword-level Word Vector Representations for Korean

Sungjoon Park [1], **Jeongmin Byun** [1], **Sion Baek** [2], **Yongseok Cho** [3], **Alice Oh** [1]
[1] Department of Computing, KAIST, Republic of Korea
[2] Program in Cognitive Science, Seoul National University, Republic of Korea
[3] Natural Language Processing team, Adecco, Republic of Korea
{sungjoon.park, jmbyun}@kaist.ac.kr, sioning1122@snu.ac.kr
yongseok.cho84@gmail.com, alice.oh@kaist.edu

Abstract

Research on distributed word representations is focused on widely-used languages such as English. Although the same methods can be used for other languages, language-specific knowledge can enhance the accuracy and richness of word vector representations. In this paper, we look at improving distributed word representations for Korean using knowledge about the unique linguistic structure of Korean. Specifically, we decompose Korean words into the *jamo* level, beyond the character-level, allowing a systematic use of subword information. To evaluate the vectors, we develop Korean test sets for word similarity and analogy and make them publicly available. The results show that our simple method outperforms word2vec and character-level Skip-Grams on semantic and syntactic similarity and analogy tasks and contributes positively toward downstream NLP tasks such as sentiment analysis.

1 Introduction

Word vector representations built from a large corpus embed useful semantic and syntactic knowledge. They can be used to measure the similarity between words and can be applied to various downstream tasks such as document classification (Yang et al., 2016), conversation modeling (Serban et al., 2016), and machine translation (Neishi et al., 2017). Most previous research for learning the vectors focuses on English (Collobert and Weston, 2008; Mikolov et al., 2013b,a; Pennington et al., 2014; Liu et al., 2015; Cao and Lu, 2017) and thus leads to difficulties and limitations in directly applying those techniques to a language with a different internal structure from that of English.

The mismatch is especially significant for morphologically rich languages such as Korean where the morphological richness could be captured by subword level embedding such as character embedding. It has been already shown that decomposing a word into subword units and using them as inputs improves performance for downstream NLP such as text classification (Zhang et al., 2015), language modeling (Kim et al., 2016), and machine translation (Ling et al., 2015; Lee et al., 2017). Despite their effectiveness in capturing syntactic features of diverse languages, decomposing a word into a set of n-grams and learning n-gram vectors does not consider the unique linguistic structures of various languages. Thus, researchers have integrated language-specific structures to learn word vectors, for example subcharacter components of Chinese characters (Yu et al., 2017) and syntactic information (such as prefixes or post-fixes) derived from external sources for English (Cao and Lu, 2017).

For Korean, integrating Korean linguistic structure at the level of *jamo*, the consonants and vowels that are much more rigidly defined than English, is shown to be effective for sentence parsing (Stratos, 2017). Previous work has looked at improving the vector representations of Korean using the character-level decomposition (Choi et al., 2017), but there is room for further investigation because Korean characters can be decomposed to *jamos* which are smaller units than the characters.

In this paper, we propose a method to integrate Korean-specific subword information to learn Korean word vectors and show improvements over previous baselines methods for word similarity, analogy, and sentiment analysis. Our first contri-

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 2429–2438
Melbourne, Australia, July 15 - 20, 2018.

bution is the method to decompose the words into both character-level units and jamo-level units and train the subword vectors through the Skip-Gram model. Our second major contribution is the Korean evaluation datasets for word similarity and analogy tasks, a translation of the WS-353 with annotations by 14 Korean native speakers, and 10,000 items for semantic and syntactic analogies, developed with Korean linguistic expertise. Using those datasets, we show that our model improves performance over other baseline methods without relying on external resources for word decomposition.

2 Related Work

2.1 Language-specific features for NLP

Recent studies in NLP field flourish with development of various word vector models. Although such studies aim for universal usage, distinct characteristics of individual languages still remain as a barrier for a unified model. The aforementioned issue is even more prominent when it comes to languages that have rich morphology but lack resources for research (Berardi et al., 2015). Accordingly, various studies dealing with language specific NLP technique proposed considering linguistics traits in models.

A large portion of these papers was dedicated to Chinese. Since Chinese is a logosyllabic language, (Yu et al., 2017) relevant studies focused on incorporation of different subword level features on word embedding, such as word internal structure (Wang et al., 2017), subcharacter component,(Yu et al., 2017), syllable (Assylbekov et al., 2017), radicals (Yin et al., 2016), and sememe (Niu et al., 2017).

The Korean language is a member of the agglutinative languages (Song, 2006), so previous studies have tried fusing the complex internal structure into the model. For example, a grammatical composition called 'Josa' in combination with word embedding is utilized in semantic role labeling (Nam and Kim, 2016) and exploiting *jamo* to handle morphological variation (Stratos, 2017). Also considered in prior work to obtain the word vector presentations for Korean is the syllable (Choi et al., 2017).

2.2 Subword features for NLP

Applying subword features to various NLP tasks has become popular in the NLP field. Typically, character-level information is useful when combined with the neural network based models. (Vania and Lopez, 2017; Assylbekov et al., 2017; Cao and Lu, 2017) Previous papers showed performance enhancement in various tasks including language modeling (Bojanowski et al., 2017, 2015), machine translation (Ling et al., 2015), text classification (Zhang et al., 2015; Ling et al., 2015) and parsing (Yu and Vu, 2017). In addition, the character n-gram fused model was suggested as a solution for a small dataset due to its robustness against data sparsity (Cao and Lu, 2017).

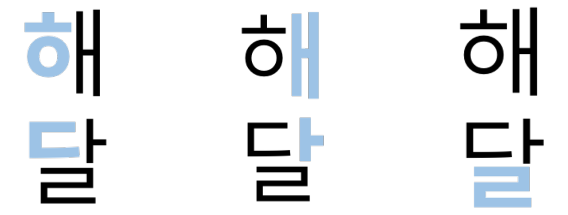
3 Model

We introduce our model training Korean word vector representations based on a subword-level information Skip-Gram. First, we briefly explain the hierarchical composition structure of Korean words to show how we decompose a Korean word into a sequence of subword components (*jamo*). Then, we extract character and *jamo* n-grams from the decomposed sequence to compute word vectors as a mean of the extracted n-grams. We train the vectors by widely-used Skip-Gram model.

3.1 Decomposition of Korean Words

Korean words are formed by an explicit hierarchical structure which can be exploited for better modeling. Every word can be decomposed into a sequence of characters, which in turn can be decomposed into *jamo*s, the smallest lexicographic units representing the consonants and vowels of the language. Unlike English which has a more flexible sequences of consonants and vowels making up syllables (e.g., "straight"), a Korean "character" which is similar to a syllable in English has a rigid structure of three *jamo*s. They have names that reflect the position in a character: 1) chosung (syllable onset), 2) joongsung (syllable nucleus), and 3) jongsung (syllable coda). The prefix *cho* in *chosung* means "first", *joong* in *joongsung* means "middle", and *jong* in *jongsung* means "end" of a character. Each component indicates how the character should be pronounced. With the exception of empty consonants, chosung and jongsung are consonants while joongsung are vowels. The *jamo*s are written with the chosung on top, with joongsung on the right of or below chosung, and jongsung on the bottom (see Fig. 1).

As shown in the top of Fig. 1, some characters such as '해Sun' lack jongsung. In this case, we add



(a) chosung (b) joongsung (c) jongsung

Figure 1: Example of the composition of a Korean character. Each character is comprised of 3 parts as shown in example of '달Moon'. On the other hand, as in the top case '해Sun', some characters lack the last component, 'jongsung'.

an empty jongsung symbol e such that a character always has three (*jamo*s). Thus, the character '달Moon' is decomposed into {ㄷ, ㅏ, ㄹ}, and '해Sun' into {ㅎ, ㅐ, e}.

When decomposing a word, we keep the order of the characters and the order of *jamo*s (chosung, joongsung, and jongsung) within the character. By following this rule, we ensure that a Korean word with N characters will have $3N$ *jamo*s in order. Lastly, the symbols for start of a word < and end of a word > are added to the sequence. For example, the word '강아지puppy' will be decomposed to a sequence of *jamo*s: {<, ㄱ, ㅏ, ㅇ, ㅇ, ㅏ, e, ㅈ, ㅣ, e, >}.

3.2 Extracting N-grams from *jamo* Sequence

We extract the following *jamo*-level and character-level n-grams from the decomposed Korean words: 1) character-level n-grams, and 2) inter-character *jamo*-level n-grams. These two levels of subword features can be successfully integrated into *jamo*-level n-grams by ensuring a character has three *jamo*s, adding empty jongsung symbol to the sequence. For better understanding, we start with the word '먹었다ate'.

Character-level n-grams. Since we add the empty jongsung symbol e when decomposing characters, we can find *jamo*-level trigrams representing a single character in the decomposed *jamo* sequence of a word. For example, there are three character-level unigrams in the word '먹었다ate':

{ㅁ, ㅓ, ㄱ}, {ㅇ, ㅓ, ㅆ}, {ㄷ, ㅏ, e}

Next, we find character-level n-grams by using the extracted unigrams. Adjacent unigrams are attached to construct n-grams. There are two character-level bigrams, and one trigram in the example:

{ㅁ, ㅓ, ㄱ, ㅇ, ㅓ, ㅆ}, {ㅇ, ㅓ, ㅆ, ㄷ, ㅏ, e}
{ㅁ, ㅓ, ㄱ, ㅇ, ㅓ, ㅆ, ㅇ, ㅓ, ㅆ, ㄷ, ㅏ, e}

Lastly, we add the total *jamo* sequence of a word including < and > to the set of extracted character-level n-grams.

Inter-character *jamo*-level n-grams. Since Korean is a member of the agglutinative language, a syntactic character is attached to the semantic part in the word, and this generates many variations. These variations are often determined by *jamo*-level information. For example, usage of the subjective case '이' or '가' is determined by the existence of jongsung in the previous character. In order to learn these regularities, we consider *jamo*-level n-grams across adjacent characters as well. For instance, there are 6 inter-character *jamo*-level trigrams in the example:

{<, ㅁ, ㅓ}, {ㅓ, ㄱ, ㅇ}, {ㄱ, ㅇ, ㅓ},
{ㅆ, ㄷ, ㅏ}, {ㅓ, ㅆ, ㄷ}, {ㅏ, e, >}

3.3 Subword Information Skip-Gram

Suppose the training corpus contains a sequence of words $\{..., w_{t-2}, w_{t-1}, w_t, w_{t+1}, w_{t+2}, ...\}$, the Skip-Gram model maximizes the log probability of context word w_{t+j} under a target word w_t:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}^{2c}\log p(w_{t+j}|w_t) \tag{1}$$

where c is the size of context window, t is total number of words in the corpus. The original Skip-Gram model use softmax function outputs for $\log p(w_{t+j}|w_t)$ in Eq. 1, however, it requires large computational cost. To avoid computing softmax precisely, we approximately maximize the log probability by Noise Contrastive Estimation, and it can be simplified to the negative sampling using the binary logistic loss:

$$\log(1+e^{-s(w_{t+j},w_t)})+\sum_{n=1}^{n_c}\log(1+e^{s(w_{t+j},w_n)}) \tag{2}$$

where n_c is the number of negative samples, and $s(w_{t+j}, w_t)$ is a scoring function. The function computes the dot product between the input of the target word vector w_t and the output of the context word vector w_{t+j}. In Skip-Gram (Mikolov et al., 2013a), an input of a word w_t is uniquely assigned over the training corpus; however, the vector in the Subword Information Skip-Gram model (Bojanowski et al., 2017) is the mean vector of the set of n-grams extracted from the word. Formally, the scoring function $s(w_t, w_{t+j})$ is:

$$\frac{1}{|G_t|}\sum_{g_t \in G_t}^{|G_t|} \mathbf{z}_{g_t}^{\mathrm{T}} \mathbf{v}_{t+j} \tag{3}$$

where the decomposed set of n-grams of w_t is G_t and its elements are g_t, $|G_t|$ is total number of elements of G_t. In general, the n-grams for $3 \leq n \leq 6$ is extracted from a word, regardless of the subword-level or compositionality of a word.

Similarly, we construct a vector representation of a Korean word by using the extracted two types of n-grams. We compute the sum of *jamo*-level n-grams, sum of character-level n-grams, and compute mean of the vectors. Let us denote character-level n-grams of w_t to G_{ct}, and inter-character *jamo*-level n-grams G_{jt}, then we obtain the scoring function $s(w_t, w_{t+j})$ as follows:

$$\frac{1}{N}\Big(\sum_{g_{ct} \in G_{ct}}^{|G_{ct}|} \mathbf{z}_{g_{ct}}^{\mathrm{T}} \mathbf{v}_{t+j} + \sum_{g_{jt} \in G_{jt}}^{|G_{jt}|} \mathbf{z}_{g_{jt}}^{\mathrm{T}} \mathbf{v}_{t+j}\Big) \tag{4}$$

where $\mathbf{z}_{g_{jt}}$ is the vector representation of the *jamo*-level n-gram g_{jt}, and $\mathbf{z}_{g_{ct}}$ is that of the character-level n-gram g_{ct}. N is sum of the number of character-level n-grams and the number of inter-character *jamo*-level n-grams $|G_{ct}| + |G_{jt}|$.

4 Experiments

4.1 Corpus

We collect a corpus of Korean documents from various sources to cover a wide context of word usages. The corpus used to train the models include: 1) Korean Wikipedia, 2) online news articles, and 3) Sejong Corpus. The corpus contains 0.12 billion tokens with 638,708 unique words. We discard words that occur fewer than ten times in the entire corpus. Details of the corpus are shown in Table 1.

Korean Wikipedia. First, we choose Korean Wikipedia articles[1] for training word vector representations. The corpus contains 0.4M articles, 3.3M sentences and 43.4M words.

Online News Articles. We collect online news articles of 5 major press from following sections: 1) society, 2) politics, 3) economics, 4) foreign, 5) culture, 6) digital. The articles were published from September to November, 2017. The corpus contains 3.2M sentences and 47.1M words.

	# of words	# of sentences	# of unique words
Wikipedia	43.4M	3.3M	299,528
Online News	47.1M	3.2M	282,955
Sejong Corpus	31.4M	2.2M	231,332
Total	121.9M	8.8M	638,708

Table 1: Number of tokens, sentences and unique words of corpus used to train the word vector representations. We aggregate three sources to make the corpus containing 0.12 billions word tokens with 0.6M unique words.

Sejong Corpus. This data is a publicly available corpus[2] which is collected under a national research project named the "21st century Sejong Project". The corpus was developed from 1998 to 2007, and contains formal text (newpapers, dictionaries, novels, etc) and informal text (transcriptions of TV shows and radio programs, etc). Thus, the corpus covers topics and context of language usage which could not be dealt with Wikipedia or news articles. We exclude some documents containing unnatural sentences such as POS-tagged sentences.

4.2 Evaluation Tasks and Datasets

We evaluate the performance of word vectors through word similarity task and word analogy task. However, to best of our knowledge, there is no Korean evaluation dataset for either task. Thus we first develop the evaluation datasets. We also test the word vectors for sentiment analysis.

4.2.1 Word Similarity Evaluation Dataset

Translating the test set. We develop a Korean version of the word similarity evaluation set. Two graduate students who speak Korean as native language translated the English word pairs in WS-353 (Finkelstein et al., 2001). Then, 14 Korean native speakers annotated the similarity between pairs by giving scores from 0 to 10 for the translated pairs, following written instructions. The original English instructions were translated into Korean as well. Among the 14 scores for each pair, we exclude the minimum and maximum scores and compute the mean of the rest of the scores. The correlation between the original scores and the annotated scores of the translated pairs is .82, which

[1] https://dumps.wikimedia.org/kowiki/20171103/

[2] https://ithub.korean.go.kr/user/main.do

indicates that the translations are sufficiently reliable. We attribute the difference to the linguistic and cultural differences. We make the Korean version of WS-353 publicly available.[3]

4.2.2 Word Analogy Evaluation Dataset

We develop the word analogy test items to evaluate the performance of word vectors. The evaluation dataset consists of 10,000 items and includes 5,000 items for evaluating the semantic features and 5,000 for the syntactic features. We also release our word analogy evaluation dataset for future research.

Semantic Feature Evaluation To evaluate the semantic features of word vectors, we refer to the English version of the word analogy test sets. (Mikolov et al., 2013a; Gladkova et al., 2016). We cover the features in both sets and translated items into Korean. The items are clustered to five categories including miscellaneous items. Each category consists of 1,000 items.

- *Capital-Country (Capt.)* includes two word pairs representing the relation between the country name and its capital:

  아테네Athens : 그리스Greece = 바그다드Baghdad : 이라크Iraq
- *Male-Female (Gend.)* evaluates the relation between male and female:

  왕자prince:공주princess = 신사gentlemen:숙녀ladies
- *Name-Nationality (Name)* evaluates the relation between the name of celebrities or stars and their nationality:

  간디Gandhi : 인도India = 링컨Lincoln : 미국USA
- *Country-Language (Lang.)* evaluates the relation between the country name and its official language:

  아르헨티나Argentina : 스페인어Spanish = 미국USA : 영어English
- *Miscellaneous (Mics.)* includes various semantic features, such as pairs of a young animals, sound of animals, and Korean-specific color-words or regions, etc..

  개구리Frog : 올챙이tadpole = 말horse : 망아지pony

  닭chicken:꼬꼬댁cackling=호랑이tiger:으르렁growl

  파란blue:새파란bluish=노란yellow:샛노란yellowish

  부산Busan : 경상남도South Gyeongsang Province
  = 대구Daegu : 경상북도North Gyeongsang Province

Syntactic Feature Evaluation We define five representative syntactic categories and develop Korean-specific test items, rather than trying to cover the existing categories in the original sets (Mikolov et al., 2013a; Gladkova et al., 2016). This is because most of the syntactic features in these sets are not available in Korean.

We develop the test set with linguistic expert knowledge of Korean. The following case is a good example. In Korean, the subject marker is attached to the back of a word, and other case markers are also explicit at the word level. Here, word level refers to 'a phrase delimited by two whitespaces around it'. Unlike Korean, in English, subjects are determined by the position in a sentence (i.e., subject comes before the verb), so the case is not explicitly marked in the word. Similarly, there are other important and unique syntactic features of the Korean language, of which we choose the following five categories to evaluate the word vectors:

- *Case* contains various case markers attached to common nouns. This evaluates a case in Korean which is represented within a word-level:

  교수Professor : 교수가Professor+case가
  = 축구soccer : 축구가soccer+case가
- *Tense* includes a verb variation of two tenses, one of which is a present tense and a past tense for the other:

  싸우다fight : 싸웠다fought = 오다come : 왔다came
- *Voice* has a pair of verb voice, one for an active voice and a passive voice for the other. It evaluates the voice which is represented by a verbal suffix:

  팔았다sold : 팔렸다be sold
  = 평가했다evaluated : 평가됐다was evaluated
- *Verb ending form* includes various verb ending forms. The various forms are part of verbal inflection in Korean:

  가다go : 가고go+form고
  = 쓰다write : 쓰고write+form고
- *Honorific (Honr.)* evaluates a morphological variation for verbs in Korean. An honorific expression is one of the most distinctive feature in Korean compared to other languages. This test set introduces the honorific morpheme '-시-' which is used in verbs:

  도왔다helped : 도우셨다helped+honorific시
  = 됐다done : 되셨다done+honorific시

4.2.3 Sentiment Analysis

We perform a binary sentiment classification task for evaluation of word vectors. Given a sequence

[3] https://github.com/SungjoonPark/KoreanWordVectors

of words, the trained classifier should predict the sentiment from the inputs while maintaining the input word vectors fixed.

Dataset We choose Naver Sentiment Movie Corpus[4]. Scraped from Korean portal site Naver, the dataset contains 200K movie reviews. Each review is no longer than 140 characters and contain binary label according to its sentiment (1 for positive and 0 for negative). The number of samples in both sentiments is equal with 100K of positives and 100K of negatives in sum. We sample from the dataset for training (100K), validation (25K), and test set (25K). Again, each set's ratio of sentiment class is balanced. Although we apply simple preprocessing of stripping out punctuation and emoticon, the dataset is still noisy with typos, segmentation errors and abnormal word usage since its original source is raw comments from portal site.

Classifier In order to build sentiment classifier, we adopt single layer LSTM with 300 hidden units and 0.5 dropout rates. Given the final state of LSTM unit, sigmoid activation function is applied for output prediction. We use cross-entropy loss and optimize parameters through Adam optimizer (Kingma and Ba, 2014) with learning rate of 0.001.

4.3 Comparison Models

We compare performance of our model to comparison models including word-level, character-level, and *jamo*-level Skip-Gram models trained by negative sampling. Hyperparameters of each models are tuned over word similarity task. We fix the number of training epochs 5.

Skip-Gram (SG) We first compare the performance with word-level Skip-Gram model (Mikolov et al., 2013a) where a unique vector is assigned for every unique words in the corpus. We set the number of dimensions as 300, number of negative samples to 5, and window size to 5.

Character-level Skip-Gram (SISG(ch)) splits words to character-level n-grams based on subword information skip-gram. (Bojanowski et al., 2017). We set the number of dimensions as 300, number of negative samples to 5, and window size to 5. The n was set to 2-4.

***Jamo*-level Skip-Gram with Empty Jongsung Symbol (SISG(jm))** splits words to *jamo*-level n-grams based on subword information skip-gram. (Bojanowski et al., 2017). In addition, if a character lacks jongsung, the symbol $\in$ is added. We set the number of dimensions as 300, number of negative samples to 5, and window size to 5. The n was set to 3-6. Note that setting n=3-6 and adding the jongsung symbol makes this model as a specific case of our model, containing *jamo*-level n-grams (n=3-6) and character-level n-grams (n=1-2) as well.

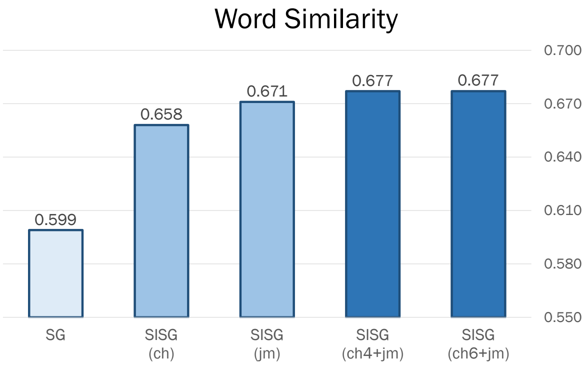


Figure 2: Spearman's correlation coefficient of word similarity task for each models. The results show higher consistency to human word similarity judgment on our method.

4.4 Optimization

In order to train our model, we apply stochastic gradient descent with linearly scheduled learning rate decay. Initial learning rate is set to .025. To speed up the training, we train the vectors in parallel with shared parameters, and they are updated asynchronously.

For our model, we set n of character n-grams to 1-4 or 1-6, and n of inter-character *jamo*-level n-grams to 3-5. We name both model as SISG(ch4+jm) and SISG(ch6+jm), respectively. The number of dimension is set to 300, window size to 5, and negative samples to 5. We train our model 5 epochs over training corpus.

5 Results

Word Similarity. We report Spearman's correlation coefficient between the human judgment and model's cosine similarity for the similarity of word pairs. Fig. 2 presents the results. For word-level skip-gram, Spearman's correlation is .599. If we decompose words into characters n-grams in order to construct word vectors (SISG(Ch)), performance is highly improved to .658. It indicates that decomposing words itself is helpful to learn good

[4] https://github.com/e9t/nsmc

Model	Analogy									
	Semantic					Syntactic				
	Capt	Gend	Name	Lang	Misc	Case	Tense	Voice	Form	Honr
SG	0.460	0.551	**0.537**	0.435	0.574	0.521	0.597	0.594	0.685	0.634
SISG(ch)	0.469	0.584	0.608	0.439	0.614	0.422	0.559	0.550	0.656	0.489
SISG(jm)	0.442	0.515	0.574	0.362	0.565	0.228	0.421	0.434	0.537	0.367
SISG(ch4+jm)	0.431	0.504	0.570	0.361	0.556	0.212	0.415	0.434	**0.501**	**0.364**
SISG(ch6+jm)	**0.425**	**0.498**	0.561	**0.354**	**0.554**	**0.210**	**0.414**	**0.426**	0.507	0.367

Table 2: Performance of our method and comparison models. Average cosine distance for each category in word analogy task are reported. Overall, our model outperforms comparison models, showing close distance between predicted vector $a + b - c$ and the target vector d (a:b=c:d). Specifically, performance is improved more in syntactic analogies.

Korean word vectors, which is morphologically rich language. Moreover, if the words are decomposed to deeper level (SISG(jm)), performance is further improved to .671.

Next, addition of an empty jongsung symbol e to *jamo* sequence, which reflects Korean-specific linguistic regularities, improves the quality of word vectors. SISG(jm), specific case of our model, shows higher correlation coefficient than the other baselines. Lastly, when we extend number of characters to learn in a word to 4 or 6, our models outperform others.

Word Analogy. In general, given an item a:b=c:d and corresponding word vectors u_a, u_b, u_c, u_d, the vector $u_a + u_b - u_c$ is used to compute cosine distances between the vector and the others. Then the vectors are ranked in terms of the distance by ascending order and if the vector u_d is found at the top, the item is counted as correct. Top 1 accuracy or error rate for each category is frequently used metric for this task, however, in this case these rank-based measures may not be an appropriate measure since the total number of unique n-grams (e.g., SISG) or unique words (e.g., SG) over the same corpus largely differ from each other. For fair comparison, we directly report cosine distances between the vector $u_a + u_b - u_c$ and u_d of each category, rather than evaluating ranks of the vectors. Formally, given an item a:b=c:d, we compute 3COSADD based metric:

$$1 - \cos(\mathbf{u}_a + \mathbf{u}_b - \mathbf{u}_c, \mathbf{u}_d) \tag{5}$$

We report the average cosine distance between predicted vector $u_a + u_b - u_c$ and target vector u_d of each category.

In semantic analogies, decomposing word into character helps little for learning semantic features. However, *jamo*-level n-grams help representing overall semantic features and our model show higher performance compared to baseline models. One exception is Name-Nationality category since it mainly consists of items including proper nouns, and decomposing these nouns does not help learning the semantic feature of the word. For example, it is obvious that the semantic features of both words '간디Ghandi' and '인도India' could not be derived from that of characters or *jamo* n-grams comprising those words.

On the other hand, decomposing words does help to learn syntactic features for all categories, and decomposing a word to even deeper levels makes learning those features more effectively. Our model outperforms all other baselines, and the amount of decreased cosine distances compared to that of word-level Skip-Gram is larger than semantic categories. Korean language is agglutinative language that character-level syntactic affixes are attached to the root of the word, and the combination of them determines final form the word. Also, the form can be reduced with *jamo*-level transformation. This is the main reason that we can learn syntactic feature of Korean words if we decompose a word into character-level and *jamo*-level simultanously. We observe similar tendency when using 3COSMUL distance metric. (Levy and Goldberg, 2014)

Sentiment Analysis. We report accuracy, loss, precision, recall and f1 score for binary sentiment classification task over test set. Although overall performance is homogeneous, our method which decompose a word to 1-6 character n-grams and 3-5 *jamo* n-grams show slightly higher performance over comparison models. In addition, our approach show better results compared to character-

Model	Acc. (%)	Prc.	Rec.	F1
SG	76.15	.746	.792	.768
SISG(ch)	76.26	.774	.741	.757
SISG(jm)	76.53	**.790**	.722	.754
SISG(ch4+jm)	76.28	.755	.776	.765
SISG(ch6+jm)	**76.54**	.750	**.795**	**.772**

Table 3: Performance of sentiment classification task. 3-5 *jamo* n-grams and 1-6 chracter n-grams show slightly higher performance in terms of accuracy and f1-score over comparison models.

Word Sim.		# of chars			
		4	5	6	all
# of *jamo*s	2-4	0.660	0.655	0.659	0.651
	3-4	0.660	0.650	0.652	0.660
	3-5	**0.677**	0.672	**0.677**	0.675
	3-6	0.665	0.663	0.664	0.669

Table 4: Spearman's correlation coefficient of Word similarity task by n-gram of *jamo*s and characters. Performance are improved when the 3-5 gram of *jamo*s and 1-4 or 1-6 gram of characters.

level SISG or *jamo*-level SISG. On the other hand, word-level Skip-Gram show comparable F1-score to our model, and is even higher than other comparison models. This is because the dataset contains significant amount of proper nouns, such as movie or actor names, and these word's semantic representations are captured better by word-level representations, as shown in word analogy task.

Effect of Size n in both n-grams. Table. 4 shows performance of word similarity task for each number of inter-character *jamo*-level n-grams and character-level n-grams. For the n of *jamo*-level n-grams, including n=5,6 of n-grams and excluding bigrams show higher performance. Meanwhile, n of character-level n-grams, including all of the character n-grams while decomposing a word does not guarantee performance improvement. Since most of the Korean word consists of no more than 6 characters (97.2% of total corpus), it seems maximum number of n=6 in character n-gram is large enough to learn word vectors. In addition, words with no more than 4 characters takes 82.6% of total corpus, so that n=4 sufficient to learn character n-grams as well.

6 Conclusion and Discussions

In this paper, we present how to decompose a Korean character into a sequence of *jamo*s with empty jongsung symbols, then extract character-level n-grams and intercharacter *jamo*-level n-grams from that sequence. Both n-grams construct a word vector representation by computing the average of n-grams, and these vectors are trained by subword-level information Skip-Gram. Prior to evaluating the performance of the vectors, we developed test set for word similarity and word analogy tasks for Korean.

We demonstrated the effectiveness of the learned word vectors in capturing the semantic and syntactic information by evaluating these vectors with word similarity and word analogy tasks. Specifically, the vectors using both *jamo* and character-level information can represent syntactic features more precisely even in an agglutinative language. Furthermore, sentiment classification results of our work indicate that the representative power of the vectors positively contributes to downstream NLP task.

Decomposing Korean word into *jamo*-level or character unigram helps capturing syntactic information. For example, Korean words add a character to the root of the word (e.g., '-은' subjective case, '-었' for past tense '-시-' for honorific, '-히-' for voice, and '-고-' for verb ending form.) Then composed word can be reduced to have fewer characters by transforming *jamo*s, such as '되었다' to '됐다'. Hence, the inter-character *jamo*-level n-grams also help capture these features. On the other hand, larger n-grams such as character-level trigram will learn unique meaning of that word since those larger component of the word will mostly occur with that word. By leveraging both features, our method produces word vectors reflecting linguistic features effectively, and thus, outperforms previous word-level approaches.

Since Korean words are divisible once more into grapheme level, resulting in longer sequence of *jamo*s for a given word, we plan to explore potential applicability of deeper level of subword information in Korean. Meanwhile, we will further train our model over noisy data and investigate how it is dealing with noisy words. Generally, informal Korean text contains intentional typos ('맛잇다'delicious' with typo'), stand-alone *jamo* as a character, ('ㅋㅋlol') and segmentation errors. ('같이가다'go together' without space'). Since these errors occur frequently, it is important to apply the vectors in training NLP models over real-word data. We plan to apply these vectors for various neural network based NLP models, such as conversation modeling. Lastly, since our method can capture Korean syntactic features through *jamo* and character n-grams, we can apply the same idea to other tasks such as POS tagging and parsing.

Acknowledgments

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2017-0-01778, Development of Explainable Human-level Deep Machine Learning Inference Framework).

References

Zhenisbek Assylbekov, Rustem Takhanov, Bagdat Myrzakhmetov, and Jonathan N Washington. 2017. Syllable-aware neural language models: A failure to beat character-aware ones. In *Proc. of EMNLP*.

Giacomo Berardi, Andrea Esuli, and Diego Marcheggiani. 2015. Word embeddings go to italy: A comparison of models and training datasets. In *IIR*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the ACL* .

Piotr Bojanowski, Armand Joulin, and Tomas Mikolov. 2015. Alternative structures for character-level rnns. *arXiv preprint arXiv:1511.06303* .

Shaosheng Cao and Wei Lu. 2017. Improving word embeddings with convolutional feature learning and subword information. In *Proc. of AAAI*.

Sanghyuk Choi, Taeuk Kim, Jinseok Seol, and Sanggoo Lee. 2017. A syllable-based technique for word embeddings of korean words. In *Proc. of the First Workshop on Subword and Character Level Models in NLP*. pages 36–40.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proc. of ICML*.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proc. of WWW*.

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. 2016. Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't. In *Proc. of the NAACL Student Research Workshop*. pages 8–15.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Proc. of AAAI*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the ACL* .

Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the Eighteenth CoNLL*.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. 2015. Character-based neural machine translation. In *Proc. of ACL*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2015. Learning context-sensitive word embeddings with neural tensor skip-gram model. In *Proc. of IJCAI*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS*.

Kyeong-Min Nam and Yu-Seop Kim. 2016. A word embedding and a josa vector for korean unsupervised semantic role induction. In *AAAI*. pages 4240–4241.

Masato Neishi, Jin Sakuma, Satoshi Tohda, Shonosuke Ishiwatari, Naoki Yoshinaga, and Masashi Toyoda. 2017. A bag of useful tricks for practical neural machine translation: Embedding layer initialization and large batch size. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017)*.

Yilin Niu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Improved word representation learning with sememes. In *Proc. of ACL*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proc. of EMNLP*.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proc. of AAAI*.

Jae Jung Song. 2006. *The Korean language: Structure, use and context*. Routledge.

Karl Stratos. 2017. A sub-character architecture for korean language processing. In *Proc. of EMNLP*. pages 721–726.

Clara Vania and Adam Lopez. 2017. From characters to words to in between: Do we capture morphology? In *Proc. of ACL*.

Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2017. Exploiting word internal structures for generic chinese sentence representation. In *Proc. of EMNLP*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proc. of NAACL*.

Rongchao Yin, Quan Wang, Peng Li, Rui Li, and Bin Wang. 2016. Multi-granularity chinese word embedding. In *Proc. of EMNLP*. pages 981–986.

Jinxing Yu, Xun Jian, Hao Xin, and Yangqiu Song. 2017. Joint embeddings of chinese words, characters, and fine-grained subcharacter components. In *Proc. of EMNLP*. pages 286–291.

Xiang Yu and Ngoc Thang Vu. 2017. Character composition model with convolutional neural networks for dependency parsing on morphologically rich languages. In *Proc. of ACL*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proc. of NIPS*.