Computer Speech & Language 65 (2021) 101134

Contents lists available at ScienceDirect

Computer Speech & Language

journal homepage: www.elsevier.com/locate/csl

A Korean named entity recognition method using Bi-LSTM-CRF and masked self-attention

Guozhe Jin[a,b], Zhezhou Yu[a,*]

[a] College of Computer Science and Technology, Jilin University, Qianjin Street: Jilin Province, 2699, China
[b] Department of Computer Science and Technology, Yanbian University, 977 Gongyuan Road, Yanji 133002, PR China

ARTICLE INFO

Article History:
Received 16 January 2020
Revised 17 May 2020
Accepted 8 July 2020
Available online 18 July 2020

Keywords:
Korean
named entity recognition
auxiliary tasks
Bi-LSTM-CRF

ABSTRACT

Named entity recognition (NER) is a fundamental task in natural language processing. The existing Korean NER methods use the Korean morpheme, syllable sequence, and part-of-speech as features, and use a sequence labeling model to tackle this problem. In Korean, on one hand, morpheme itself contains strong indicative information of named entity (especially for time and person). On the other hand, the context of the target morpheme plays an important role in recognizing the named entity(NE) tag of the target morpheme. To make full use of these two features, we propose two auxiliary tasks. One of them is the morpheme-level NE tagging task which will capture the NE feature of syllable sequence composing morpheme. The other one is the context-based NE tagging task which aims to capture the context feature of target morpheme through the masked self-attention network. These two tasks are jointly trained with Bi-LSTM-CRF NER Tagger. The experimental results on Klpexpo 2016 corpus and Naver NLP Challenge 2018 corpus show that our model outperforms the strong baseline systems and achieves the state of the art.

1. Introduction

Named entity recognition (NER) is a fundamental task in natural language processing. Generally, NER task aims to extract named entities from the sentence and classify these entities into predefined categories, such as person, location, organization, time, etc, while domain-specific NER tasks try to extract domain-specific entities such as drugs, weapons, and commodities. NER is the first step in some other NLP tasks such as information extraction, information retrieval, question answering, etc. With the development of machine learning, various statistics-based NER methods have been proposed. Such as Hidden markov model (Zhou and Su, 2002), SVM (Isozaki and Kazawa, 2002), CRF model (McCallum and Li, 2003), etc. These methods treat NER tasks as sequence labeling problem and use labeled NER corpora to train statistical models which take hand-crafted features as input. These models assign a NE tag to each token in a sentence. The performance of these models depends on the selected hand-crafted features, such as lexical feature, orthographic pattern, character level affixes of the target and surrounding words, part-of-speech, N-gram, etc.

Unlike the traditional machine learning approach, deep learning methods do not rely on feature engineering but learn these features automatically. These methods represent each word by a pre-trained word vector (Word2Vec Mikolov et al., 2013 or GloVe Pennington et al., 2014, etc.). These word vectors are commonly trained in large corpora through the Skip-gram or CBOW and contain rich semantic and syntactic information. These methods take word vectors as input, encode

*Corresponding author.
E-mail address: yuzz@jlu.edu.cn (Z. Yu).

https://doi.org/10.1016/j.csl.2020.101134

the entire sentence through LSTM, and use the output of each LSTM unit as the NER feature, and finally output the NE label of each word (token) (Lample et al., 2016; Chiu and Nichols, 2016; Dong et al., 2016; Ma and Hovy, 2016; Huang et al., 2015). In addition, to capture the dependencies between NE tags, these models add the CRF layer on top of the Bi-LSTM. There are a considerable number of deep learning-based NER methods that use Bi-LSTM-CRF as their backbone model to solve the sequence labeling problem since it delivers excellent performance.

Korean is agglutinative language and there is no clear word boundary in Korean, which is different from English. Each Korean sentence consists of multiple eojeols, and each eojeol consists of multiple morphemes. In Korean, the eojeol is a spacing unit, and the morpheme is the minimum semantic unit, and each character constituting the morpheme is called a syllable. In Korean, there are countless morphological combinations and an increasing number of foreign words (person names or and location). If we use eojeol as the basic processing unit, it will cause serious OOV(out of vocabulary) problem. Therefore, the syllable is used as the basic processing unit in deep learning-based Korean NER methods (Na et al., 2019; Kwon et al., 2019; 2017). These models encode the syllable sequence of each morpheme through CNN or LSTM and concatenate it with morpheme embedding, and then feed the result into the sentence-level Bi-LSTM-CRF. These methods solve the OOV problem well through the syllable-level LSTM and yield good performance. However, the existing Korean NER methods fail to fully consider the language-specific features. Many named entities(especially time and person name) in Korean have distinctive lexical features, for example, “1월7일” (January 7). In addition, even in the case of OOV morpheme, we can infer the NER tag of unknown morpheme from the surrounding morphemes. For example, in the Korean sentence “박지성은 자신감 넘치는 표정을 지어 보였다.”, The NE tag (which is PS in this case) of “박지성” can be inferred from the semantic information of the affix “은” and subsequent context morphemes. To enhance the lexical features of the morpheme, we add an independent NE tagger to the syllable-level LSTM, which predicts the NE label only through the syllable sequence constituting the morpheme. Besides, to enhance the morpheme context feature, we introduce a masked self-attention layer into our model to mask the target morpheme and predict the NER label only through the context information. We trained these two submodule and the sentence-level Bi-LSTM-CRF NE tagger jointly. The experiment results on Klpexpo2016 corpus and Naver NLP Challenge 2018 corpus show that our model surpasses the best baseline model.

2. Relative work

2.1. Machine learning-based NER methods

The performance of machine learning-based NER methods not only relies on the model itself to be used but also affected by the quality of feature engineering. Zhou and Su (2002) proposed the HMM-based NER method in which they integrate the internal and external features of the word into the model. Specifically, they use capitalization, digitalization, the semantic feature of important triggers, gazetteer in their model. Zhou and Su’s method achieved F-measures of 96.6% and 94.1% on MUC-6 and MUC-7 English NE tasks respectively. Isozaki and Kazawa (2002) used features such as POS tag, character type, target word, context words, etc., and trained an SVM classifier to tackle the NER problem. Their model obtained the best results on the IREX dataset. Another SVM-based NER method is proposed by Li et al. (2004), yielding 88.3% F-score on the CoNLL-2003 dataset. McCallum and Li (2003) introduced a CRF-based NER method. They got seeds for the lexicons from the annotated dataset, then uses the Web, HTML formatting regularities and a search engine service to significantly augment those lexicons, which improved NER performance.

2.2. Neural network-based NER methods

With the development of deep learning in recent years, the neural network-based NER methods have surpassed the traditional machine learning-based NER method. In the paper of Collobert et al. (2011), sentences are encoded using CNN, and NE tags are labeled by the CRF model. Huang et al. (2015) adopted a bidirectional LSTM network and added a CRF layer on top of it. Their model achieved 84.26% F1 score on English CoNLL 2003 dataset.

In addition to the improvement of neural network architecture, another research direction of NER is to enrich word representation, for example, character-level decomposing in English or radical-level decomposing in Chinese. These fine-grained features are good for solving the OOV problem which is known to be one of the major difficulties in the NER task since the words or phrases composing named entity are commonly low-frequency words. Lample et al. (2016) introduced character embeddings and encoded the word using character-level LSTM, which are used to capture orthographic sensitivity and alleviate the OOV problem. They combine the character feature with word embedding to enrich the representation of the word. In order to capture the dependencies between NER tags, a CRF layer is added on top of the LSTM in their model to jointly predict the NER tags. This model achieved state of the art in the NER tasks of the various languages and became the backbone model of many subsequent neural network-based NER methods. Santos and Guimaraes (2015) proposed to encode words using the CNN model. The word vector and character vector are trained simultaneously in an unsupervised manner, and these two vectors are used to perform the NER task.

2.3. Korean NER methods

Lee et al. (2006) used the CRF model to detect the boundaries of named entities and classified the detected named entities using the maximum entropy model. Their method achieved 78.6% F1 on a Korean NER dataset containing 147 fine-grained named entity labels. There are some other methods that are based on the neural network. These models extended the Bi-LSTM-CRF model, where various kinds of extra features are added. Na et al. (2019) proposed a Bi-LSTM-CRF based Korean NER model and incorporated the syllable sequence feature to solve the OOV problem in Korean NER. Lee et al. (2018) proposed a multi-task model to alleviate the error propagation problem which is caused by the Korean morphological analysis. The model trains the morphological analyzer and the NE tagger jointly and passes the results of the morphological analysis to the NE tagging as an additional feature. Kwon et al. (2019) used the syllable bigram as the basic processing unit and used the syllable level Bi-LSTM to encode the sequence of syllable bigram. Despite the success of methods mentioned above, we believe that some language features can be further extracted from the language nature of Korean to improve the performance of Korean NER. Our model is also based on Bi-LSTM-CRF which is known to be the best sequence labeling model. We extract the lexical information and contextual information from the Korean sentence by introducing a morphological-level NE tagger and masked self-attention mechanism.

3. Method

3.1. Problem definition

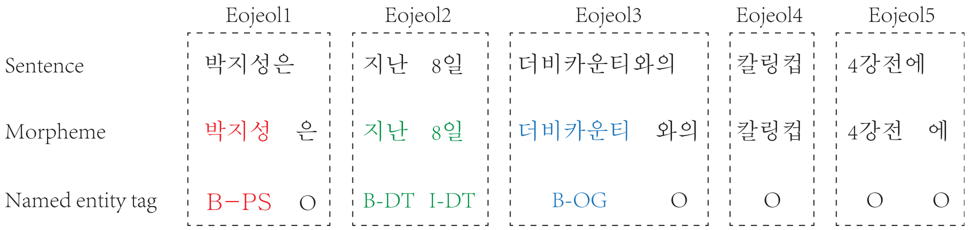
There is a clear boundary between the words in English, so NE tagging is conducted towards these words directly through an NER model. Compared with English, Korean is a morphologically rich language, and each eojeol may consist of multiple morphemes. The Korean NER is generally performed with morphemes as the basic processing unit. Therefore, we need to use some Korean morphological analysis tools to segment the original Korean sentence into morpheme sequence. For example, in Fig. 1, the first row is the original sentence, and the second row is the corresponding morpheme sequence.

The named entity may be composed of multiple morphemes, so we use the “BIO” scheme to represent the named entity boundary. For example, eojeol “지난 8일” representing time consists of two morphemes “지난” and “8일”. We use “B” to denote the first morpheme of a named entity and “I” to denote other morphemes and “O” to denote non-NE morpheme. In the “BIO” scheme, the boundary tag and NE tag will be combined, forming a unified label space. For example, “박지성” in Fig. 1 is represented by a boundary label “B” and a NE label “PS” (person name), and the two labels are concatenated with a symbol “-”.

Following the previous work, our model also uses Bi-LSTM-CRF as the backbone framework. We use a Bi-LSTM network to encode a syllable sequence and concatenate it with the NE Dictionary features and morpheme embedding. Then, these features are fed to a sentence-level Bi-LSTM to encode the whole sentence and the CRF layer is used to output the NE label for each morpheme. The proposed model is shown in Fig. 2. Besides the Bi-LSTM-CRF, we add a morpheme-level NE tagger and a masked self-attention module to fuse the lexical information and context information of morpheme to our model respectively. There are three objective functions in our model, which correspond to the above three sub-modules. During training, these three objective functions will be optimized jointly in a multi-task manner.

3.2. Morpheme-level NE tagging

Given the input sentence $x=\{w_1,\dots,w_n\}$, where w_i denotes the ith morpheme. Each morpheme is composed of multiple syllables, $w_i=\{s_1^i,\dots,s_{|w^i|}^i\}$, where s_j^i represents the jth syllable of morpheme w_i. Our model aims to predict NE tag sequence $\widehat{y}=\left\{\widehat{t}_1,\dots,\widehat{t}_n\right\}$, where $\widehat{t}_i$ means the NE tag of ith morpheme.



	Eojeol1	Eojeol2	Eojeol3	Eojeol4	Eojeol5
Sentence	박지성은	지난 8일	더비카운티와의	칼링컵	4강전에
Morpheme	박지성 은	지난 8일	더비카운티 와의	칼링컵	4강전 에
Named entity tag	B−PS O	B-DT I-DT	B-OG O	O	O O

Fig. 1. A dashed box in the figure represents an eojeol, and each eojeol is composed of multiple morphemes. The first row is the original Korean sentence, the second row is the morpheme sequence, and the third row is the named entity tag corresponding to each morpheme. The named entity may be composed of multiple morphemes, so we use the “BIO” scheme to represent the named entity boundary. For example, eojeol “지난 8일” representing time consists of two morphemes “지난” and “8일”. We use “B” to denote the first morpheme of a named entity, and “I” to denote other morphemes.

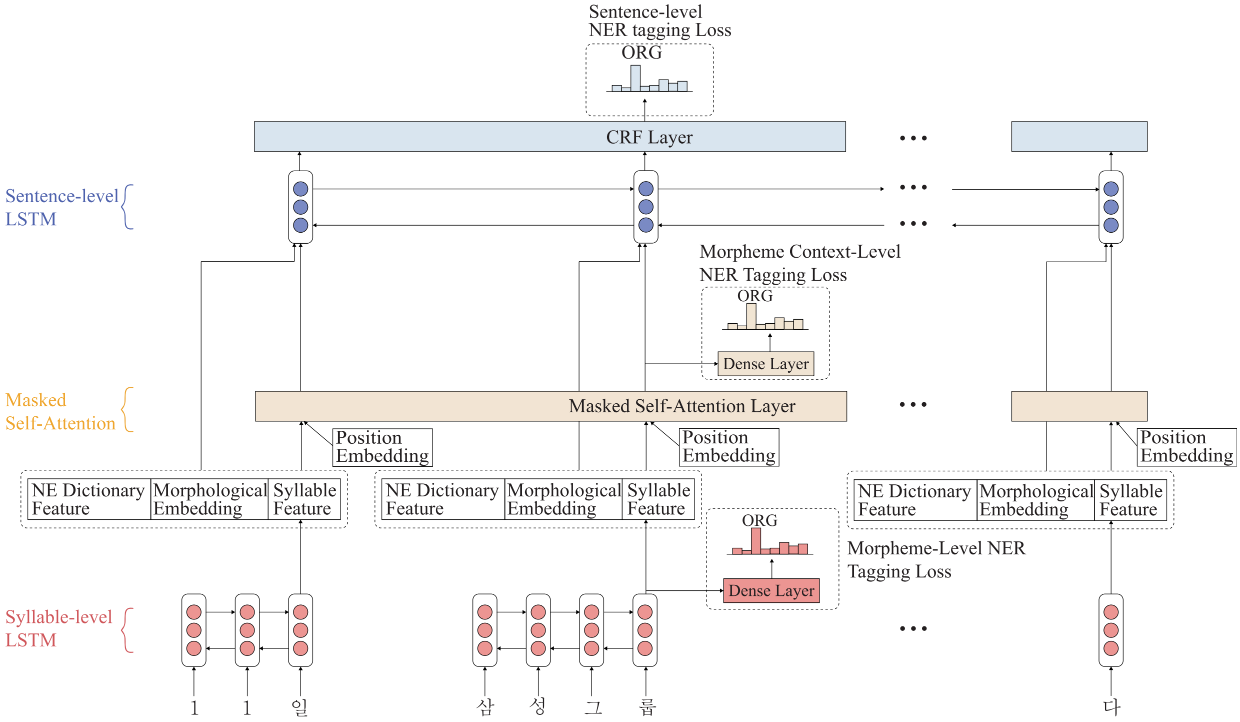


Fig. 2. The model consists of three parts: 1) syllable-level Bi-LSTM which is used to encode syllables composing morpheme, 2)the masked self-attention layer which is used to capture the context feature of morpheme without target morpheme, 3) the Bi-LSTM-CRF layer which is used to predict NE tags for each morpheme by combining all features.

Our model splits the input sentence x into n morphemes and convert each syllable s of morphemes into a low-dimensional real-valued vector which is called word vector generally. We use pre-trained Korean version of fastText (Grave et al., 2018) as sylabe embedding, which will be fine-tuned during training. We use notation $emb(s_j^i)$ to represent the jth syllable vector in the ith morpheme. To capture the internal lexical feature of morpheme, we adopt the Bi-LSTM model to encode the syllable sequence of each morpheme.

$$\begin{aligned} \overrightarrow{h_j} &= \overrightarrow{LSTM}_{eojeol}\left(emb(s_j^i), \overrightarrow{h_{j-1}}\right) \\ \overleftarrow{h_j} &= \overleftarrow{LSTM}_{eojeol}\left(emb(s_j^i), \overleftarrow{h_{j-1}}\right) \end{aligned} \tag{1}$$

In formula (1), we can get the last state of the forward LSTM $\overrightarrow{h}_{|w_i|}$ and last state of the backward LSTM $\overleftarrow{h}_1$. We concatenate these two vectors into one vector r^i which represents the ith morpheme.

$$r^i = \overrightarrow{h}_{|w_i|} \oplus \overleftarrow{h}_1 \tag{2}$$

As mentioned in Section 1, there are some patterns in named entities of Korean. For example, the pattern "number + 일" often appears in dates. Therefore, we added an independent NE label prediction module to the syllable-level Bi-LSTM, so that the model learns the internal NE features of Korean morphemes. We feed r^i to a fully-connected network and output the NE label of ith morpheme with a *softmax* function.

$$p^m(t) = softmax(W_1 r^i + b_1) \tag{3}$$

Where W_1 and b_1 are trainable parameters. $p^m(t)$ represents the probability distribution of NE labels generated by morpheme-level NE tagger. We add the following objective function to this module.

$$L^m = -\frac{1}{N}\sum_{j=1}^{N}\sum_{k=1}^{C} t_{jk} log(p_{jk}^m) \tag{4}$$

Where t represents the golden label, and t_{jk} means kth NE category of the jth morpheme. If the NE tag of the jth morpheme belongs to the kth NE category, then t_{jk} is 1, otherwise 0. p_{jk}^m is the probability that the jth morpheme belongs to the kth NE category, which is generated by the model. C is the number of NE categories, and N is the number of morphemes in the corpus.

3.3. Masked self-attention layer

As mentioned before, the morpheme context information plays an important role in Korean NER. Even in the case where an unknown morpheme is encountered, the NE tag of the target morpheme can be inferred from the semantic information of the surrounding morpheme. We can infer the NE label (organization) of the morpheme “더비카운티” (the third eojeol in Fig. 1) by its suffix morpheme “와의” and the semantic information of other surrounding eojeol. Based on this observation, we introduce a masked self-attention network to our model to enhance the context feature of the morpheme. In Transformer model (Vaswani et al., 2017), the mask mechanism is proposed to block future information in the process of decoding. In BERT (Devlin et al., 2019), some words are randomly masked by the special token to train the masked language model. In our model, the mask mechanism is involved in the attention calculating process, which is similar to that in the paper Vaswani et al. (2017). The difference is that our model hides all the current morphemes through a special mask matrix, which forces the model to learn the NE feature of the current morpheme from the surrounding morphemes. These features will provide complementary information to the upper LSTM layer.

The input of the masked self-attention layer consists of four parts: The first one is r^i obtained via Eq. (2), which represents the feature of syllable sequence.

The second one is morphological embedding, which is denoted as $emb^m(w_i)$ and will be initiated with the fastText word vectors which have dimension 300.

The third one is NE dictionary feature, which is obtained through the lookup table of the NE label. We built the Korean NE dictionary using the infobox on the Korean Wikipedia page.[1] We extract pages with infobox from the downloaded Korean Wikipedia dump file. The fields in infobox vary across the Wikipedia pages, even for the same type of entities. Therefore, we use common fields for each type of entity. For example, we employ birth date and name for Person entity, coordinates, population, area for Location entity and CEO, Industry Profit, Established for Organization entity. In the end, we built a Korean NE dictionary with 58,440 Person entity, 37,963 Location entity and 26,198 Organization entity. Each entry of the Korean NE dictionary consists of two parts, the first one is the entity, and the second one is the NE label corresponding to the entity. After that, we build NE label embedding via the Korean NE dictionary. Specifically, each morpheme is compared with the entities in the Korean NE dictionary. If there exists a matching entry in the NE dictionary, then the corresponding entity label will be taken out and transformed into a low-dimensional real vector, which will form the NE dictionary feature. If more than one entity in the Korean NE dictionary matches the current morpheme, the longest matching entity will be chosen.

The last one is position embedding. Since context information is encoded only by the attention network, the position information of the morpheme will be lost. Thus, we add the position embedding to our model, which is denoted as $emb^p(w_i)$. We adopt simple absolute position embedding. Specifically, we input the position index of each morpheme into the position lookup table and obtain the corresponding position embedding by querying the lookup table.

These four vectors are concatenated and fed to the masked self-attention network.

$$input_i^a = r^i \oplus emb^m(w_i) \oplus emb^d(w_i) \oplus emb^p(w_i) \tag{5}$$

We use $I = [input_1^a; \cdots; \mathrm{input}_n^a]$ to represent an input sentence. Where I is a matrix which is composed of n input vectors. Note that the superscript a of *input* is used to distinguish from the input of Bi-LSTM-CRF. We convert I into Key, Value, Query matrices through three different fully connected layers.

$$\begin{aligned} K &= W^K \cdot I \\ Q &= W^Q \cdot I \\ V &= W^V \cdot I \end{aligned} \tag{6}$$

Then we use the following formula to calculate the attention score S.

$$S = Q^T \cdot K \tag{7}$$

Where $S \in \mathbb{R}^{n \times n}$, n is the sentence length. In order to mask out the current target morpheme, we expect that the diagonal value of the attention matrix will be 0. Thus, before calculating *softmax*, we introduced a mask matrix $M \in \mathbb{R}^{n \times n}$ into the formula.

$$M = \begin{bmatrix} -\infty & 0 & \cdots & 0 \\ 0 & -\infty & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -\infty \end{bmatrix} \tag{8}$$

Next, the probability distribution of attention is calculated by the following equation.

$$A = softmax(S + M) \tag{9}$$

Next, the matrix V and the probability distribution of attention are multiplied to obtain the output vector H.

$$H = V \cdot A \tag{10}$$

[1] https://ko.wikipedia.org/

Each column in the matrix A represents the attention probability distribution over the whole sentence for one morpheme. Note that the diagonal elements in the attention matrix A are all 0. Therefore, the feature of the ith morpheme is not included in h_i^a which is column vector of $H = [h_1^a; \cdots; \mathrm{h}_\mathrm{n}^\mathrm{a}]$. Finally, the model output the probability distribution of NE labels for each morpheme through another fully connected layer and the *softmax* function.

$$p^a(t) = softmax(W_2 h_i^a + b_2) \tag{11}$$

We use a cross-entropy objective function in this layer to supervise the model to learn the context features around morphemes.

$$L^a = -\frac{1}{N}\sum_{j=1}^{N}\sum_{k=1}^{C} t_{jk} log(p_{jk}^a) \tag{12}$$

Note that the definition of t_{jk}, N and C is the same as in Eq. (4).

3.4. Bi-LSTM-CRF layer

Finally, the aggregated features are fed to a sentence-level Bi-LSTM-CRF network. Specifically, the input of sentence-level Bi-LSTM consists of four parts. The first three parts are the r^i, $emb^m(w_i)$, and $emb^d(w_i)$ which are also used in the masked self-attention layer. The fourth feature is the context feature h_i^a obtained from the masked self-attention layer. These four vectors are concatenated and fed to the Bi-LSTM-CRF layer.

$$input_i^s = r^i \oplus emb^m(w_i) \oplus emb^d(w_i) \oplus h_i^a \tag{13}$$

Where $input_i^s$ is the input vector of the sentence-level Bi-LSTM.

$$\begin{aligned} \overrightarrow{h}_i &= \overrightarrow{LSTM}_{sentence}\left(input_i^s, \overrightarrow{h}_{i-1}\right) \\ \overleftarrow{h}_i &= \overleftarrow{LSTM}_{sentence}\left(input_i^s, \overleftarrow{h}_{i-1}\right) \end{aligned} \tag{14}$$

Where $\overrightarrow{LSTM}_{sentence}$ denotes sentence-level forward LSTM, and $\overleftarrow{LSTM}_{sentence}$ denotes sentence-level backward LSTM. At each step of the LSTM, we will get a forward output $\overrightarrow{h}_i$ and backward output $\overleftarrow{h}_i$. We concatenate these two vectors into one.

$$h_i^s = \overrightarrow{h}_i \oplus \overleftarrow{h}_i \tag{15}$$

We use a fully connected layer to transform the morpheme feature h_i^s into the unnormalized NE probability p_i^s.

$$p_i^s = tanh(W_3 h_i^s + b_3) \tag{16}$$

Considering the dependencies between NE tags, we added a CRF layer on top of the sentence-level Bi-LSTM. We use a transition matrix U to model the dependencies between NE labels, and $U_{i,j}$ is the transition probability from NE label i to NE label j. The score of the NE path $\widehat{y}$ corresponding to x is calculated by Eq. (17).

$$s(x, \widehat{y}) = \sum_{i=0}^{n} U_{t_i, t_{i+1}} + \sum_{i=1}^{n} p_i^s \tag{17}$$

The Bi-LSTM-CRF is optimized by the following objective function.

$$L^s = -s(x, y) + log(\sum_{\tilde{y} \in Y_x} e^{s(x, \tilde{y})}) \tag{18}$$

Y_x represents all possible NE tag paths corresponding to the morpheme sequence x. During decoding, the Viterbi algorithm can be used to obtain the optimal path. We optimize the objective function L, which is a weighted sum of the three objective functions mentioned in this section.

$$L = \lambda_1 L^m + \lambda_2 L^a + \lambda_3 L^s \tag{19}$$

Where, λ_1, λ_2, λ_2 are hyper-parameters, which are set to 1 in our experiment.

4. Experiments

4.1. Experimental settings

To evaluate the proposed model, we used two different datasets. The first one is Klpexpo 2016 NER corpus[2]. It was released for the Korean language information processing system contest 2016. It contains 3,555 training sentences, 500 development sentences, and 1,000 test sentences. Klpexpo 2016 NER corpus contains a total of 12,372 manually labeled NE tags of 5 categories. The

[2] Available at http://ithub.korean.go.kr/user/contest/contestIntroLastView.do

Table 1
Parameter setting on two dataset.

Parameter	Klpexpo2016	Naver NLP challenge 2018
batch size	16	32
morpheme embedding size	300	300
syllable embedding size	300	300
LSTM hidden size	128	128
optimizer	Adam	Adam
learning rate	0.0001	0.001
epoch	50	20

second one is the Naver NLP challenge 2018 dataset[3] which contains 90,000 sentences. Since the test dataset is not publicly available, we randomly shuffled these 90,000 sentences and split them manually. Specifically, We use 80,000 sentences for training and 1,000 for development and the remaining 9,000 for testing.

We use the F1-score as evaluation matrics in our experiments, where F1-score is the geometric mean of precision and recall. Since a named entity may be composed of multiple morphemes, the prediction is considered to be correct only if both the NE labels and the boundary labels of a named entity are correct. F1-score is calculated as follows.

$$\begin{aligned} F_1 &= \frac{2\times Precision\times Recall}{Precision+Recall}\\ Precision &= \frac{of\ correctly\ predicted\ NEs}{of\ predicted\ NEs\ by\ model}\\ Recall &= \frac{of\ correctly\ predicted\ NEs}{of\ true\ NEs} \end{aligned} \tag{20}$$

We trained our model in a single Nvidia 1070 GPU. Due to different data sizes, different parameter settings are used in two datasets. These parameters are selected by fine-tuning on the target dataset. We list the best setting of parameters in Table 1.

4.2. Experimental results

Our model is compared with the 5 baseline methods listed below.

- Choi et al. (2016) uses the DBSCAN algorithm to perform clustering based on the part-of-speech information. This feature and word2vec are fed to a CRF model to solve the Korean NER problem.
- Nam et al. (2017) uses 11-dimensional vector as a extra feature, which contains morphological features and part-of-speech features.
- Yu and Ko (2017) adds pre-trained Korean word embedding, part-of-speech embedding, syllable embedding, and named entity dictionary features to the LSTM-CRF model. This model can be regarded as a simplified version of our model without the morpheme NE tagger and masked self-attention layer.
- Kwon et al. (2019) uses the syllable bi-gram vector representation and uses the syllable-level Bi-LSTM to encode each syllable bi-gram. They use the encoded syllable bigrams as an extra feature and feed them to the high-level Bi-LSTM-CRF model to perform NER labeling.
- Na et al. (2019) uses a CNN and a Bi-LSTM to encode a syllable sequence respectively. These two features combining with morpheme embedding are fed to sentence-level Bi-LSTM-CRF.

As can be seen from Table 2, the performances of the Bi-LSTM-CRF based method (the latter 5 methods) are generally better than the CRF-based method (the first method). On one hand, although word embedding is used as a feature in the model of Choi (2016), the parameters of word embedding are not updated during training, and it is not an end-to-end neural model essentially. On the other hand, it also indicates that Bi-LSTM is better than the simple CRF model in modeling the entire sentence. Although the CRF layer is also used in the model of Nam (2017), the CRF layer is mainly used to predict the NE boundary label in a pipeline manner. The NE label is predicted at each time step by Bi-LSTM, which means the model is unable to consider the dependencies between successive NE labels. Therefore, the performance is worse than the end-to-end Bi-LSTM-CRF model and its variants (the last 4 models in Tables 2 and 3). The model of Yu and Ko (2017) and Kwon et al. (2019)yield similar performance since the model architecture and the features used by these two methods are similar. Comparing with other Bi-LSTM-CRF series models, Na(2019)'s model uses CNN and Bi-LSTM simultaneously, which enrich the syllable feature. Besides, Na(2019)'s model uses more features than other models. For example, the POS tag feature, word spacing info, lexical feature, and other additional features. These features will also help to improve performance. We believe that it is the reason why the performance of Na(2019)'s model is better than the model of Yu and Ko (2017) and Kwon et al. (2019).

[3] Available at http://air.changwon.ac.kr

Table 2
Performances on Klpexpo2016 dataset.

Name	Model	Features used in model	F1-score
Choi et al. (2016)	CRF	word embedding, POS feature	82.29
Nam et al. (2017)	Bi-LSTM-CRF	word embedding, POS feature, morpheme feature	84.73
Yu and Ko (2017)	Bi-LSTM-CRF	word embedding, syllable embedding, NE Dictionary feature	85.49
Kwon et al. (2019)	Bi-LSTM-CRF	word embedding, syllable feature(LSTM)	85.53
Na et al. (2019)	Bi-LSTM-CRF	word embedding, syllable feature(LSTM+CNN), POS tag feature, word spacing info, lexical feature	85.71
ours	Bi-LSTM-CRF + masked self attention	word embedding, syllable feature(LSTM), NE Dictionary feature, Position embedding	86.27

Table 3
Performances on Naver NLP Challenge 2018 dataset. The columns of Model and Feature are omitted for simplification.

Name	F1-score
Choi et al. (2016)	85.72
Nam et al. (2017)	86.28
Yu and Ko (2017)	88.13
Kwon et al. (2019)	88.24
Na et al. (2019)	90.15
ours	91.07

Our model achieved the best performance on both Klpexpo 2016 and Naver NLP Challenge 2018 datasets. Comparing with the Na(2019)'s model which is the previous best model, our model obtains absolute improvement of 0.56 and 1.56 on the datasets respectively with fewer additional features. On one hand, our model enhances the feature of the syllable sequence through the independent NE tagger. On the other hand, we capture the features of the morpheme context through the masked self-attention layer, which further improves the performance. We will further discuss these two modules in the ablation study. As can be seen from Tables 2 and 3), our model obtains more improvement on the Naver NLP Challenge corpus than on the Klpexpo2016 corpus. The reason is that our model can automatically learn the NE feature of syllable sequence and morpheme context feature through the neural network. Thus, our model performs better on the larger dataset.

4.3. Variants of our model

To further explore the effect of each module on the performance, we conducted some comparative experiments on variants of our model. First of all, we compared LSTM and CNN models in syllable encoder. We refer to the model using CNN based syllable encoder as our model+CNN syllable encoder. Secondly, in terms of model architecture, to measure the impact of the position of the masked self-attention layer, we put the masked self-attention layer on top of the sentence-level BiLSTM layer, which forms another variant of our model, we refer to it as our model+self-attention over sentence-level BiLSTM. Finally, to verify the effectiveness of the masked self-attention layer in the BERT-based model, we compare the following two models. The first one is the pre-trained multilingual version of BERT. We add the CRF layer on top of BERT to predict NE tags. We refer to this model as BERT +fine-tuning. The other one is BERT with masked self-attention layer. In this model, BERT is regarded as the morphological feature extractor. The morphological feature generated by BERT is concatenated with the feature generated by the masked self-attention, which is fed to the CRF layer to predict the NE tag. We refer to this model as BERT+masked self-attention. The experimental results are shown in Table 4.

Table 4
Performances of variant models.

Model	Klpexpo 2016	Naver NLP challenge 2018
our model	86.27	91.07
our model+CNN syllable encoder	86.34	91.03
our model+self-attention over sentence-level BiLSTM	86.02	90.69
BERT+fine-tuning	86.78	91.80
BERT+masked self-attention	86.91	91.96

As can be seen from table 4, CNN based syllable encoder yields similar performance compared with RNN based syllable encoder. The possible reason is that Korean morpheme length is generally short, hence, it can capture the local context information of syllable sequence well in both CNN and BiLSTM. From the comparison between our model and our model+self-attention over sentence-level BiLSTM, we can see that if we add the self-attention layer on top of the sentence-level BiLSTM layer, it will lead to slight performance degradation. The reason is that BiLSTM already modeled the context information of each morpheme, which results in the masked self-attention of the upper layer can obtain the information of the current morpheme from the surrounding morphemes even though it masks out the feature of the current morpheme. This hinders the ability of the masked self-attention to infer the current NE label from surrounding morphemes. From the comparison between the last two models and the first three models, we can see that the BERT based models perform better than CNN and LSTM based model. In addition, from the comparison of the last two models, we can see that adding masked self-attention layer on BERT can still improve performance slightly, which shows the effectiveness of our method. However, in BERT based model, the masked self-attention layer is not as effective as in LSTM based model. The reason is that BERT itself is a multi-layer self-attention architecture, in which the morpheme context has been well modeled.

4.4. Ablation study

To verify the effectiveness of the morpheme-level NE tagger and the masked self-attention layer adopted by our model, we conduct some ablation experiments. First, to verify the effectiveness of the morpheme-level NE tagger, we removed the morpheme-level NE tagger and the corresponding objective function. We call this model “ours w/o morpheme-level NE tagger”. Second, we remove the masked self-attention layer from our model, which we call “ours w/o masked self-attention layer”. The last model is the full model, which we call “ours”. The comparison results of these models are shown in Table 5.

As we can see from Table 5, the masked self-attention layer is the most important factor for our model. If the masked self-attention layer is removed from the model, the F1-score drop by 0.63 and 1.97 on Klpexpo 2016 dataset and Naver NLP Challenge 2018 dataset, respectively. It indicates that the contextual information around morphemes plays an important role in the Korean NER task. In Korean, the morphemes have different semantic meanings according to their location. Therefore, context information is useful information for named entity recognition, and it is beneficial to solve OOV problems. The morpheme-level NE tagger also improves performance, but it is not as obvious as the masked self-attention layer. The main reason is that the syllable level LSTM network can obtain sufficient supervision signal from the upper sentence-level LSTM network, so the supervision of the morpheme-level NE tagger added to the syllable level Bi-LSTM network is weakened.

To further verify the effectiveness of the masked self-attention layer, we analyzed a self-attention matrix(A in Eq. (9)) generated by our model. We chose a Korean sentence “국립원예특작과학원 최영훈 박사 입니다.” as an example. It contains two NEs, which are “국립원예특작과학원(B-OG)” and “최영훈(B-PS)”. The self-attention matrix is shown in Fig. 3.

As shown in the second column of Fig. 3, when analyzing the person name “최영훈”, the subsequent morpheme “박사” (Ph.D.) plays a key role. In Korean, a title is most likely preceded by a person’s name. Therefore, the model pays more attention to “박사” (title), which may provide useful information for predicting the NE tag of the previous morpheme as a PS(person name). The model assigns more attention weight to “최영훈” and “박사”, which may help us to predict the NE tag of morpheme “국립원예특작과학원”. But in this case, our model relies more on the morpheme-level NE tagger. By analyzing the morpheme-level NE tagger, we find that it assigns the correct NE label “OG” to the morpheme. It means that the morpheme-level NE tagger successfully captures the NE feature of the syllable sequence and transfers it to the sentence-level Bi-LSTM-CRF. In addition, we find that the model pays more attention to the morphemes which are close to the target morpheme when predicting non-NE morphemes(the last four morphemes in Fig. 3).

5. Conclusions

In this paper, we propose a novel Korean named entity recognition approach which involves two additional sub-tasks. To enhance the syllable features of the morpheme, we introduce an independent morpheme-level NE tagger to our model, which predicts the NE tag according to the syllable sequence composing the morpheme. Besides, to capture the context feature of morpheme, we propose a masked self-attention network to mask out the target morpheme and predict the NE label of target morpheme only using surrounding morphemes. These two tasks are jointly trained with Bi-LSTM-CRF NER Tagger. The experimental results on Klpexpo2016 corpus and Naver NLP Challenge 2018 corpus show that our model outperforms the strong baseline systems and achieves the state of the art.

Table 5
F1-score of our model without some part of the module.

Model	Klpexpo2016	Naver NLP Challenge 2018
ours w/o morpheme-level NE tagger	85.92	90.28
ours w/o masked self-attention layer	85.64	89.10
ours	86.27	91.07

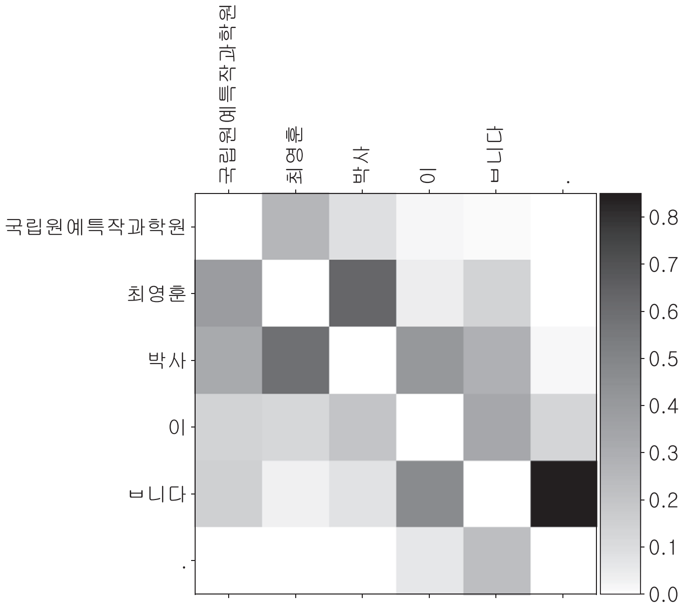


Fig. 3. Since we mask target morphemes, the diagonal elements of the self-attention matrix are all 0. Each column in the figure represents the attention distribution of the corresponding morpheme in the horizontal axis. In other words, the sum of each column in the figure is 1. The darker color means higher attention weight.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Chiu, J.P., Nichols, E., 2016. Named entity recognition with bidirectional LSTM-CNNs. Trans. Assoc. Comput.Ling. 4, 357–370.

Choi, H., Kwon, S., Seo, J., 2016. Korean named entity recognition using clustered according to part of speech. In: Proceedings of HCI KOREA 2017, pp. 397–400.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P., 2011. Natural language processing (almost) from scratch. J. Mach. Learn. Res. 12 (Aug), 2493–2537.

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., 2019. 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, Volume 1 (Long and Short Papers). Association for Computational Linguistics, Minneapolis, Minnesota, pp. 4171–4186. https://doi.org/10.18653/v1/N19-1423.URL https://www.aclweb.org/anthology/N19-1423

Dong, C., Zhang, J., Zong, C., Hattori, M., Di, H., 2016. Character-based LSTM-CRF with radical-level features for chinese named entity recognition. Natural Language Understanding and Intelligent Applications. Springer, pp. 239–250.

Grave, E., Bojanowski, P., Gupta, P., Joulin, A., Mikolov, T., 2018. Learning word vectors for 157 languages. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018).

Huang, Z., Xu, W., Yu, K., 2015. Bidirectional LSTM-CRF models for sequence tagging. arXiv:1508.01991.

Isozaki, H., Kazawa, H., 2002. Efficient support vector classifiers for named entity recognition. COLING 2002: The 19th International Conference on Computational Linguistics.URL https://www.aclweb.org/anthology/C02-1054

Kwon, S., Ko, Y., Seo, J., 2019. Effective vector representation for the korean named-entity recognition. Pattern Recognit. Lett. 117, 52–57.

Kwon, S., Ko, Y., Seo, J., 2017. A robust named-entity recognition system using syllable bigram embedding with eojeol prefix information. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. ACM, pp. 2139–2142.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C., 2016. Neural architectures for named entity recognition. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, San Diego, California, pp. 260–270. https://doi.org/10.18653/v1/N16-1030.URL https://www.aclweb.org/anthology/N16-1030

Lee, C., Hwang, Y.-G., Oh, H.-J., Lim, S., Heo, J., Lee, C.-H., Kim, H.-J., Wang, J.-H., Jang, M.-G., 2006. Fine-grained named entity recognition using conditional random fields for question answering. Asia Information Retrieval Symposium. Springer, pp. 581–587.

Lee, H.-g., Park, G., Kim, H., 2018. Effective integration of morphological analysis and named entity recognition based on a recurrent neural network. Pattern Recogni. Lett. 112, 361–365.

Li, Y., Bontcheva, K., Cunningham, H., 2004. SVM based learning system for information extraction. International Workshop on Deterministic and Statistical Methods in Machine Learning. Springer, pp. 319–339.

Ma, X., Hovy, E., 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. arXiv:1603.01354.

McCallum, A., Li, W., 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, pp. 188–191.URL https://www.aclweb.org/anthology/W03-0430

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J., 2013. Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems, pp. 3111–3119.

Na, S.-H., Kim, H., Min, J., Kim, K., 2019. Improving LSTM CRFs using character-based compositions for korean named entity recognition. Comput. Speech Lang. 54, 106–121.

Nam, S., Hahm, Y., Choi, K.-S., 2017. Application of word vector with korean specific feature to Bi-LSTM model for named entity recognition. Annual Conference on Human and Language Technology. Human and Language Technology, pp. 147–150.

Pennington, J., Socher, R., Manning, C., 2014. Glove: global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532–1543.

Santos, C. N. d., Guimaraes, V., 2015. Boosting named entity recognition with neural character embeddings. arXiv:1505.05008.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I., 2017. Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 30. Curran Associates, Inc., pp. 5998–6008.URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

Yu, H., Ko, Y., 2017. Expansion of word representation for named entity recognition based on bidirectional LSTM CRFs. J. KIISE 44 (3), 306–313.

Zhou, G., Su, J., 2002. Named entity recognition using an HMM-based chunk tagger. proceedings of the 40th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, pp. 473–480.