A Grapheme-level Approach for Constructing a Korean Morphological Analyzer without Linguistic Knowledge

Jihun Choi, Jonghem Youn and Sang-goo Lee
Department of Computer Science and Engineering
Seoul National University
Seoul, Republic of Korea, 08826
Email: {jhchoi, jonghm, sglee}@europa.snu.ac.kr

Abstract—**Morphological analysis is an essential step for processing the Korean language, due to highly agglutinative properties of the language. In this paper, we propose a novel approach for constructing a Korean morphological analyzer that can capture linguistic properties using graphemes as basic processing units. Since our model does not utilize prior linguistic knowledge, the model can be applied to other training corpora with ease. Our model performs morphological analysis through two consecutive sequence labeling tasks: *lexical form recovery* and *part-of-speech tagging*. In the lexical form recovery step, morphological changes of an input sentence are restored to the original form. Then in the part-of-speech step, corresponding part-of-speech tags are attached to the recovered form. Experimental results show that our model outperforms previous models which are constructed without prior knowledge.**

1. Introduction

Morphological analysis of the Korean language is a complex problem, due to the agglutinative properties of the language. In inflectional languages such as English, part-of-speech tagging is performed by attaching the appropriate tag to each word which is separated by spaces or punctuation marks. However in Korean, the chunk of text separated by spaces (called 'eojeol') is not a single word but a combination of morphemes; substantive with postposition or predication with ending. Hence, an eojeol can contain multiple semantic and syntactic information and, consequently, the number of possible eojeols is combinatorially large. For instance, an eojeol '*gassda*' (갔다; which means "went") is composed of three morphemes: a verb '*ga*' (가), a prefinal ending '*ϵass*' (았), and an ending '*da*' (다). Note that a change in form occurs when combining '*ga*' (가) and '*ϵass*' (았) into '*gass*' (갔). Furthermore, the verb '*ga*' (가) can be combined with other morphemes to represent eojeols with different semantic and syntactic information such as '*gago*' (가고; which means "go and") or '*gal*' (갈; which means "will go").

Therefore, morphological analysis is an essential step for processing the Korean language, where each eojeol is divided into semantically and syntactically distinct units (morphemes), and the size of the vocabulary set can be reduced as a result. The typical process of Korean morphological analysis is composed of two tasks: 1) separating each eojeol of a sentence into morphemes, and 2) attaching the appropriate part-of-speech tag to each morpheme. Since an eojeol can have multiple candidate morpheme sequences, selecting the proper morpheme sequence by considering context is important. For example, the eojeol '*neon*' (넌) can be analyzed into either '*neo*/NP + *neun*/JKS' (너/NP + 는/JKS; which means "you are") or '*neol*/VV + *n*/EC' (널/VV + ㄴ/EC; which means "hang"), based on the context around the eojeol.

In previous works, morphological analysis is performed based on rules [1], [2], or statistical machine learning methods such as hidden Markov models (HMMs) and conditional random fields (CRFs) [3], [4]. In rule-based approaches, rules are handcrafted based on sophisticated expert knowledge to separate eojeols into morphemes or to attach part-of-speech labels to morphemes [5], [6]. In machine learning-based approaches, large corpora annotated with morphological analysis results are required for training models. Recent machine learning-based works adopt syllable-level models that use syllables as processing units [7], [8], [9], which are differ from previous works which adopted morphemes as processing units.

Even in machine learning-based approaches, many of them depend on heavy linguistic knowledge in constructing training data, extracting features, or separating eojeols into morphemes [7], [8], [9]. Examples of linguistic knowledge include irregular conjugation, vowel coalescence, morpheme dictionary, pre-analyzed-and-annotated dictionary, phoneme constraint rules, joint constraint rules for part-of-speech tags, etc. Due to the extensivity and complexity of linguistic knowledge considered, rule construction is a highly time-consuming and tedious task. Furthermore, since the annotation guideline of a certain corpus may not be compatible with that of another corpus, the rules must be adjusted, if not rebuilt, for use with other corpora.

In this paper, we propose a novel grapheme-based approach to perform morphological analysis without utilizing handcrafted rules. In our model, morphological analysis is performed through two sequence labeling tasks: *lexical form recovery* and *part-of-speech tagging*, both of which use grapheme as a processing unit. In the lexical form recovery step, morphological changes in an input sentence are restored to the original form (which is a concatenated string of morphemes) using bidirectional long short-term memory (BLSTM) [10]. Then in the part-of-speech tagging step, the part-of-speech tag is attached to each grapheme of the recovered lexical form using BLSTM with the additional conditional random field layer (BLSTM-CRF) [11], [12]. We explain our model in detail in Section 3.

The motivation behind adopting a grapheme-level approach starts with the fact that a grapheme is the minimal unit of the Korean writing system. In other words, in the Korean writing system, character is not an indivisible unit. A Korean character is composed of three types of graphemes: initial, medial, and optional final. For example, the character '*neun*' (는) is composed of initial '*n*' (ㄴ), medial '*eu*' (ㅡ), and final '*n*' (ㄴ). Similarly, the character 'ϵo' (오) is decomposed into initial ϵ (ㅇ) and medial '*o*' (ㅗ)[1]. We observe that, when morphemes are combined into an eojeol, most of the form changes occur in the grapheme-level. Thus, using character as a processing unit is not sufficient for modeling the complex morphological properties of the Korean language. Our grapheme-level approach processes the language in a microscopic level, and thus, able to capture and generalize the complex properties of the language in a more direct way.

Our contributions can be summarized as follows. First, to our knowledge, the model proposed in this paper is the first work that introduces the grapheme as a processing unit for the machine learning-based morphological analysis of the Korean language. Second, our model can learn fine-grained linguistic features because grapheme is the lowest information unit in the language. Third, we achieve significant improvements in accuracy compared to previous works. Lastly, our model can be applied to other corpora conveniently since it does not need any prior (handcrafted) linguistic knowledge.

2. Related Work

There exist some previous works which aim to build a morphological analyzer without exploiting external knowledge. In the works of [13] and [14], a morphological analysis is modeled using three types of conditional distributions (eojeol-, morpheme-, and syllable-unit). The distributions are obtained solely from the corpus, thus the models can be applied to other corpora without modification. In line with their works, [15] proposed a similar model that solves the problem through three simple subtasks: morpheme recovery, segmentation, and tagging, to reduce complexity of analysis. Our model differs from their approaches in some aspects. First, we adopt grapheme as a unit, which makes our model be able to capture linguistic properties appearing in a microscopic level. In addition, since we use BLSTM, which is a recurrent neural network-based model, longer dependencies than using n-grams can be considered.

The works of [16] and [17] suggest models for morphological analysis using a statistical method. The work of [16] achieves a good accuracy, but they use a treebank corpus to build the model. Although a treebank corpus contains more structured information, the average size of available treebank corpora is not as large as that of corpora annotated with morphological analysis results. The model proposed in [17] is able to be built without linguistic knowledge, but the model performs accurately only when combined with the handcrafted rules.

To avoid the feature extraction process which is needed in statistical machine learning methods like hidden Markov models (HMMs) or conditional random fields (CRFs), [18] proposed a model that uses a feed-forward neural network for part-of-speech tagging. However, the model achieves good accuracy only when used jointly with CRFs, and the model still needs morpheme segmentation module which uses handcrafted features.

3. Proposed Model

Our model consists of two consecutive steps: *lexical form recovery*[2] and *part-of-speech tagging*. In both steps, graphemes are used as processing units. In the lexical form recovery step, the model recovers the lexical form of a input graphemes. And in the part-of-speech tagging step, the model attaches the corresponding part-of-speech tag to each grapheme of the recovered lexical form.

For example, the sentence '*jib*ϵ*eulo* ϵ*wassda*' (집으로 왔다; which means "came home") has to be analyzed to '*jib*/NNG + ϵ*eulo*/JKB, ϵ*o*/VV + ϵ*ass*/EP + *da*/EF' (집/NNG + 으로/JKB, 오/VV + 았/EP + 다/EF). Input graphemes and the lexical form of the example sentence are defined as follows.

- Input graphemes: *j* + *i* + *b* + ϵ + *eu* + *l* + *o* + *[sp]*[3] + ϵ + *wa* + *ss* + *d* + *a*
(ㅈ + ㅣ + ㅂ + ㅇ + ㅡ + ㄹ + ㅗ + *[sp]* + ㅇ + ㅘ + ㅆ + ㄷ + ㅏ)
- Lexical form: *j* + *i* + *b* + ϵ + *eu* + *l* + *o* + *[sp]* + ϵ + *o* + ϵ + *a* + *ss* + *d* + *a*
(ㅈ + ㅣ + ㅂ + ㅇ + ㅡ + ㄹ + ㅗ + *[sp]* + ㅇ + ㅗ + ㅇ + ㅏ + ㅆ + ㄷ + ㅏ)

Fig. 1 describes how the model performs for the above example. The left part of the figure describes the process of the lexical form recovery step, and the right part describes the process of the part-of-speech tagging step. Note that the number starts from 9 to show how the model works when there exists a difference between input graphemes and the lexical form. (See $\mathbf{h}_{10}$, $\mathbf{g}_{10}$, $\mathbf{g}_{11}$, and $\mathbf{g}_{12}$ in the figure.)

1. Note that in Korean, the initial *i-eung* (ㅇ) is soundless and omitted in Romanization, but we denote it explicitly as ϵ for explanation throughout the paper. And for the sake of brevity in explanation, we also call every non-Korean character including symbols and the whitespace character as graphemes.

2. Throughout this paper, the term 'lexical form' means the concatenation of graphemes composing the original morpheme sequence.

3. *[sp]* denotes the whitespace character.

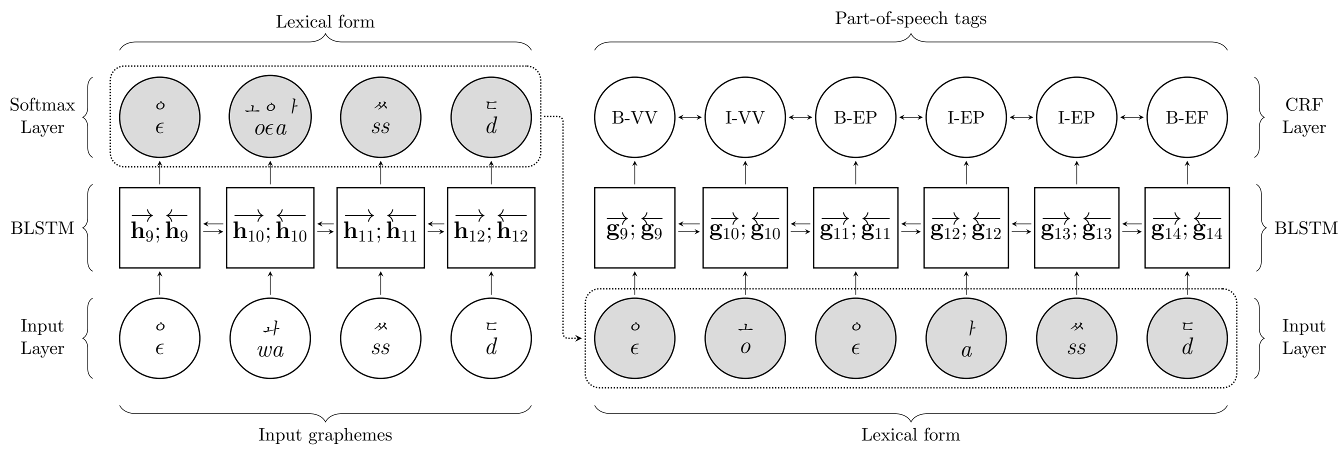

Figure 1. Overview of the model

3.1. Lexical Form Recovery

Our model treats lexical form recovery as a sequence labeling problem, where for each input grapheme the model predicts corresponding graphemes of the lexical form. Grapheme is a minimal unit of the Korean writing system, thus by using grapheme as a unit the model can learn the linguistic knowledge needed for recovering the lexical form efficiently.

In order to apply a sequence labeling algorithm, we associate each grapheme of the input with grapheme(s) of the lexical form. In most cases, an input grapheme is related to one grapheme of the lexical form, but it is possible for an input grapheme to be related to no grapheme or multiple graphemes of the lexical form, if the surface form is changed while morphemes are combined into an eojeol.

For instance, in the left part of Fig. 1, the grapheme '*ss*' (ㅆ) in the input is associated with '*ss*' (ㅆ) in the output. The grapheme '*wa*' (ㅘ) is associated with '*oϵa*' (ㅗ ㅇ ㅏ), which is a concatenation of three graphemes: '*o*' (ㅗ), '*ϵ*' (ㅇ), and '*a*' (ㅏ). In the lexical form recovery step, we treat concatenated graphemes related to a input grapheme as a single term.

As a sequence labeling algorithm, we use a bidirectional long short-term memory (BLSTM) [10] network with peephole connections [19]. A long short-term memory (LSTM) network is known to be able to capture long-term context, and by adding bidirectional connections, the model uses the context from both directions for prediction. Since an LSTM network is a neural network-based model, extracting features using extensive linguistic knowledge is not needed, compared to models using hidden Markov models (HMMs) and conditional random fields (CRFs). Formally, $\mathbf{h}_t$, the hidden state of an LSTM network at time t, is calculated as follows.

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i)\\
\mathbf{f}_t &= \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{W}_{cf}\mathbf{c}_{t-1} + \mathbf{b}_f)\\
\mathbf{c}_t &= \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c)\\
\mathbf{o}_t &= \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{co}\mathbf{c}_t + \mathbf{b}_o)\\
\mathbf{h}_t &= \mathbf{o}_t \circ \tanh(\mathbf{c}_t),
\end{aligned}$$

where σ is the element-wise sigmoid function and $\circ$ is an element-wise product operator. $\mathbf{x}_t$ is the vector representation of an input grapheme at time t, which is obtained by lookup of the embedding matrix. $\mathbf{W}$s and $\mathbf{b}$s are parameters of an LSTM network, where $\mathbf{W}$s are weight matrices and $\mathbf{b}$s are bias vectors.

To build a bidirectional network, two LSTM networks which share the input are used. At time t, one LSTM network processes the input sequence left to right to obtain the forward hidden state $\overrightarrow{\mathbf{h}}_t$, and another LSTM network does it in the opposite direction to obtain the backward hidden state $\overleftarrow{\mathbf{h}}_t$. Then they are concatenated into the vector $\mathbf{h}_t = \left[\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t\right]$. $\mathbf{h}_t$ is multiplied by the weight matrix to calculate unnormalized probability of each output at time t, and the softmax function is applied to calculate the normalized probabilities.

The network is trained using the dataset which is generated by the method which will be described in Subsection 3.4.

3.2. Part-of-speech Tagging

In the part-of-speech tagging step, the proposed model attaches the appropriate part-of-speech tag to each grapheme of the lexical form. To indicate the boundaries between morphemes, we prepend B (beginning) and I (inside) information to each part-of-speech tag. One example that addresses the need for B and I information is compound noun; without B and I information, it lacks of information to separate the compound noun into multiple nouns.

The right part of Fig. 1 describes how the task is done using the same example as in the lexical form recovery step. Recall that in the lexical form recovery step, graphemes associated with an input grapheme is treated as a single term. In this step, they are separated into individual graphemes, as shown in the figure that the output *oϵa* (ㅗ ㅇ ㅏ) of the lexical form recovery step is broken into three graphemes *o* (ㅗ), ϵ (ㅇ), and *a* (ㅏ) in the input of the part-of-speech tagging step.

Similar to the previous step, a BLSTM network is used for sequence labeling, but a linear-chain CRF layer is added to the output layer of a BLSTM network. We will call this model BLSTM-CRF. The idea of adding CRF layer to LSTM is proposed in [11], [12].

Unlike a BLSTM network which calculates the probability of a label relying only on the current hidden state, BLSTM-CRF considers the probability of an entire label sequence (a part-of-speech tag sequence in our case) given an input sequence. For an input sequence $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ and a predicted label sequence $\mathbf{y} = (y_1, y_2, \ldots, y_n)$, the score is defined as

$$s(\mathbf{X}, \mathbf{y}) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} P_{i, y_i},$$

where A_{y_i, y_j} is the score of a transition from label y_i to y_j. P_{i, y_i} is the unnormalized probability for time i and for label y_i which is obtained from a BLSTM unit. y_0 and y_{n+1} denote special labels, *start* and *end*, which mean the start state and the end state of a label sequence, respectively.

Using the score, the probability of $\mathbf{y}$ given $\mathbf{X}$ is calculated as

$$p(\mathbf{y}|\mathbf{X}) = \frac{e^{s(\mathbf{X}, \mathbf{y})}}{\sum_{\mathbf{y}' \in \mathbf{Y_X}} e^{s(\mathbf{X}, \mathbf{y}')}},$$

where $\mathbf{Y_X}$ is a set of all possible label sequences for an input $\mathbf{X}$. We train the model to maximize the log probability

$$\log\left(p(\mathbf{y}|\mathbf{X})\right) = s(\mathbf{X}, \mathbf{y}) - \log\left(\sum_{\mathbf{y}' \in \mathbf{Y_X}} s(\mathbf{X}, \mathbf{y}')\right).$$

The latter term (which is a logarithm of the normalizing factor) can be efficiently calculated using dynamic programming (by the forward algorithm). For inference, finding $\mathbf{y}^*$ that maximizes the log probability can be found by the Viterbi algorithm.

Motivation of adding a CRF layer is that a plain BLSTM network does not have dependencies between predicted outputs. Though the dependencies are indirectly considered through recurrent connections between LSTM units, these indirect dependencies cannot correct the probability of a prediction based on prediction results of its neighbors. In our model, the direct dependencies between prediction results are especially important, because 1) graphemes composing the same morpheme must have the same part-of-speech tag and 2) neighboring B and I information should affect prediction of part-of-speech labels. For instance, predicting I-NNP should be favored than I-NNG, when the neighboring part-of-speech tags are B-NNP and I-NNP. We will demonstrate that adding a CRF layer makes substantial improvement in Subsection 4.3.

3.3. Generation of Morphological Analysis Result

After we recover the lexical form from the input (Subsection 3.1) and attach the corresponding part-of-speech tag to each grapheme of the lexical form (Subsection 3.2), we can generate the final morphological analysis result in a straightforward way. First, using B and I information, we can group the lexical form into several groups. Then graphemes of each group are composed into Korean syllables similar to the way the Korean typewriting system works.

We use the same example as previous subsections to describe this process. For the input (1), we decompose it into graphemes (2). Then we predict the lexical form (3) using the network proposed in Subsection 3.1. For the lexical form (3), we apply part-of-speech tagging to predict the grapheme-level result (4) according to Subsection 3.2. Using B and I information of part-of-speech tags, we can group (4) into (5). Finally graphemes of each group is composed into morphemes (6), which is the final result of the morphological analysis. (The tag 'SPACE' indicates the special part-of-speech tag which is attached to the whitespace character.)

(1) *jibϵeulo ϵwassda* (came home)
(집으로 왔다)

(2) *j* + *i* + *b* + *ϵ* + *eu* + *l* + *o* + *[sp]* + *ϵ* + *wa* + *ss* + *d* + *a*
(ㅈ + ㅣ + ㅂ + ㅇ + ㅡ + ㄹ + ㅗ + *[sp]* + ㅇ + ㅘ + ㅆ + ㄷ + ㅏ)

(3) *j* + *i* + *b* + *ϵ* + *eu* + *l* + *o* + *[sp]* + *ϵ* + *o* + *ϵ* + *a* + *ss* + *d* + *a*
(ㅈ + ㅣ + ㅂ + ㅇ + ㅡ + ㄹ + ㅗ + *[sp]* + ㅇ + ㅗ + ㅇ + ㅏ + ㅆ + ㄷ + ㅏ)

(4) *j*/B-NNG + *i*/I-NNG + *b*/I-NNG + *ϵ*/B-JKB + *eu*/I-JKB + *l*/I-JKB + *o*/I-JKB + *[sp]*/SPACE + *ϵ*/B-VV + *o*/I-VV + *ϵ*/B-EP + *a*/I-EP + *ss*/I-EP + *d*/B-EF + *a*/I-EF
(ㅈ/B-NNG + ㅣ/I-NNG + ㅂ/I-NNG + ㅇ/B-JKB + ㅡ/I-JKB + ㄹ/I-JKB + ㅗ/I-JKB + *[sp]*/SPACE + ㅇ/B-VV + ㅗ/I-VV + ㅇ/B-EP + ㅏ/I-EP + ㅆ/I-EP + ㄷ/B-EF + ㅏ/I-EF)

(5) (*j*/B-NNG + *i*/I-NNG + *b*/I-NNG), (*ϵ*/B-JKB + *eu*/I-JKB + *l*/I-JKB + *o*/I-JKB), *[sp]*/SPACE, (*ϵ*/B-VV + *o*/I-VV), (*ϵ*/B-EP + *a*/I-EP + *ss*/I-EP), (*d*/B-EF + *a*/I-EF)
((ㅈ/B-NNG + ㅣ/I-NNG + ㅂ/I-NNG), (ㅇ/B-JKB + ㅡ/I-JKB + ㄹ/I-JKB + ㅗ/I-JKB), *[sp]*/SPACE, (ㅇ/B-VV + ㅗ/I-VV), (ㅇ/B-EP + ㅏ/I-EP + ㅆ/I-EP), (ㄷ/B-EF + ㅏ/I-EF))

(6) *jib*/NNG + *ϵeulo*/JKB, *ϵo*/VV + *ϵass*/EP + *da*/EF
(집/NNG + 으로/JKB, 오/VV + 았/EP + 다/EF)

3.4. Dataset Generation

We use the Sejong corpus[4] for the dataset. The Sejong corpus is one of the largest Korean corpora which contains

4. https://ithub.korean.go.kr/user/main.do

morphological analysis results aligned in eojeol-level. The corpus composed of 10,092,499 eojeols in 871,889 sentences.

To train the lexical form recovery step, we need training data aligned in grapheme-level. To generate the training data, we perform a simple preprocessing using a sequence alignment algorithm, based on the intuition that most of graphemes in eojeol remain unchanged in their lexical form.

We use the Needleman-Wunsch algorithm [20] to align the corpus in grapheme-level. Despite the simplicity of the algorithm, we observed that it can find appropriate relationship between graphemes in the eojeol and those in the lexical form.

We describe how this preprocessing is conducted using an example eojeol ϵ*wassda* (왔다), whose grapheme representation is (a) ϵ + *wa* + *ss* + *d* + *a* (ㅇ + ㅘ + ㅆ + ㄷ + ㅏ). Since the corpus is aligned in an eojeol-level, the morphological analysis result of the eojeol is given in the corpus: ϵ*o*/VV + ϵ*ass*/EP + *da*/EF (오/VV + 았/EP + 다/EF). From that result, we can obtain the lexical form for the eojeol: (b) ϵ + *o* + ϵ + *a* + *ss* + *d* + *a* (ㅇ + ㅗ + ㅇ + ㅏ + ㅆ + ㄷ + ㅏ). Aligning (a) and (b) using the sequence alignment algorithm, we get

(1) ϵ + $\varnothing$ + $\varnothing$ + *wa* + *ss* + *d* + *a*
(ㅇ + $\varnothing$ + $\varnothing$ + ㅘ + ㅆ + ㄷ + ㅏ)

(2) ϵ + *o* + ϵ + *a* + *ss* + *d* + *a*
(ㅇ + ㅗ + ㅇ + ㅏ + ㅆ + ㄷ + ㅏ),

where $\varnothing$ denotes a gap. Then, each grapheme of the eojeol is associated to graphemes of the lexical form according to Algorithm 1. For the parameters of the algorithm, we set a match score to 4, a mismatch score to -1, a gap opening penalty to -2, and a gap extension penalty to -1.

Algorithm 1: Associating each grapheme of an eojeol with those of a lexical form

```
$x_1, x_2, \ldots, x_n \leftarrow$ aligned graphemes of an eojeol;
$y_1, y_2, \ldots, y_n \leftarrow$ aligned graphemes of a lexical
  form;
$s \leftarrow []$;
for $i = 1$ to $n$ do
    if $y_i \neq \varnothing$ then
        $s \leftarrow s + [y_i]$;
    end
    if $x_i \neq \varnothing$ then
        associate $x_i$ with $s$;
        $s \leftarrow []$;
        $k \leftarrow i$;
    end
end
if $|s| > 0$ then
    add $s$ to the association of $x_k$;
end
```

Since the association is done in a consistent manner, we can achieve a certain level of robustness even for the case that the sequence alignment algorithm does not align sequences as intended.

TABLE 1. Lexical form recovery accuracy

Method	Grapheme	Eojeol	Sentence	Changed
Naive	98.77	91.02	39.01	0.00
LSTM	99.52	96.20	66.46	70.29
BLSTM	**99.90**	**99.24**	**92.03**	**94.98**

Aligning an eojeol to the analyzed results is unnecessary for training the part-of-speech tagging task, because only the morphologically analyzed results of the corpus are used for training. The part-of-speech tag of each morpheme is directly mapped to graphemes constructing the morpheme, and B or I information is prepended to the part-of-speech tag according to the position of a grapheme in its corresponding morpheme.

4. Experiment

4.1. Experimental Setup

For both networks, we set the sizes of hidden layer to 200. From preliminary experiments, we found that the corpus is large enough for the model to generalize, thus we do not use regularization method. The size of mini-batch is set to 128 and we use truncated backpropagation through time (BPTT) [21] by setting the maximum backpropagation length to 64. As an optimization algorithm, Adam optimizer [22] is used. All experiments were conducted on the machine with GTX 960 GPU. The network for lexical form recovery was trained for about 12 hours, and the network for part-of-speech tagging was trained for about a day. For implementation, we used Theano [23] library.

The dataset is generated from the Sejong corpus using the method stated in Subsection 3.4. We randomly use 85% of the dataset as training data, 5% as validation data, and 10% as test data.

4.2. Lexical Form Recovery

Table 1 shows the accuracy of the lexical form recovery task. Each column presents accuracy scores in different granularities. In the column named Grapheme, accuracy is measured in grapheme-level. In Eojeol column and Sentence column, accuracy is measured in eojeol- and sentence-level respectively, which means that a prediction result is considered to be correct when all graphemes in an eojeol or a sentence are correct. The column named Changed presents grapheme-level accuracy among graphemes whose forms are changed in the lexical form. Accuracy scores in the Changed column indicate the capabilities of different methods to capture linguistic changes.

The Naive method simply predicts the lexical form to be identical to the input. Since it simply copies the input, it cannot recover graphemes whose forms are changed. The reason we conducted experiments using this method is to validate the importance of predicting graphemes whose forms are changed. The scores of the Naive method tell

TABLE 2. PART-OF-SPEECH TAGGING ACCURACY

Method	Grapheme	Eojeol	Sentence
LSTM	81.23	33.83	2.45
LSTM-CRF	96.81	92.60	50.39
BLSTM	97.12	89.95	40.92
BLSTM-CRF	**97.88**	**95.50**	**64.62**

us that merely observing the grapheme-level accuracy can be misleading. Since there exist much more graphemes which do not experience form changes in the lexical form, predicting the lexical form as the input itself achieves a good accuracy in grapheme-level. However, the scores degrade vastly in eojeol- and sentence-level.

These results show that graphemes whose forms change in the lexical form play a key role. Though they are not frequent in grapheme-level, the results indicate that they are evenly distributed so that they belong to many eojeols and sentences, which makes it important to predict graphemes with form changes. We see that our proposed model recovers the lexical form accurately, especially for graphemes whose forms are changed in the lexical form.

Comparing results of LSTM and BLSTM, we validate that the network with bidirectional connections predicts lexical forms much more accurately, including graphemes experienced linguistic changes. From this observation, we can say that it is important for lexical form recovery to consider the left and the right context both. This is also compatible to previous works where the context from both sides is used for crafting rules or extracting features.

4.3. Part-of-speech Tagging

In this subsection, we verify the effect of bidirectional connections in the LSTM network and the additional CRF layer. Table 2 describes accuracy scores for LSTM, LSTM-CRF, BLSTM, and BLSTM-CRF. For the experiments, we assume that the lexical form recovery is done without error.

In all granularities of accuracy, BLSTM substantially works better than LSTM, especially in eojeol- and sentence-level. This large improvement on accuracy tells us of the importance of considering the context from both sides in predicting part-of-speech tags. Bidirectional connections also improve the accuracy of LSTM-CRF, but it shows smaller improvement than that without CRF layers. This can be explained by the fact that the neighboring context is also considered by the CRF layer, because the model with the CRF layer considers the probability of an entire part-of-speech sequence. We observe that BLSTM-CRF performs best among four methods, since it considers the adjacent context using both bidirectional connections and the CRF layer.

Interestingly, despite small difference in grapheme-level accuracy scores between BLSTM and BLSTM-CRF, the differences get larger in eojeol- and sentence-level accuracy scores. Similarly, although BLSTM predicts more accurately than LSTM-CRF in grapheme-level, it works worse than LSTM-CRF in eojeol- and sentence-level. Some eojeols selected from the results of BLSTM and BLSTM-CRF can account for this phenomenon.

Table 3 lists the selected samples that show the influence of the CRF layer. Without the CRF layer, some morphemes are broken into multiple segments and wrong results are produced. These are frequently found in nouns, since similar part-of-speech tags such as NNP (proper noun) and NNG (common noun) are attached mixedly into graphemes of the same morpheme. With the CRF layer, the model does not suffer from this problem because the label transition matrix corrects the probability based on its neighboring predictions.

4.4. End-to-end Evaluation

In this subsection, we conduct experiments to measure end-to-end performance, which can be used to compare our model with previous works which used the same corpus (the Sejong corpus). In experiments of this subsection, we apply lexical form recovery for an input sentence and apply part-of-speech tagging to the recovered lexical form.

In Table 4, we can see that our model outperforms previous works which have the same goal as ours. In [13], more than one probabilistic models are proposed. The accuracy score presented in the table is the best score among multiple models. It is same for the score of [14]. Also note that their models can fail to produce results on certain inputs, compared to our model that does not fail. The work of [17] uses both handcrafted rules and statistical information. The score on the table is a score of the model that uses only statistical information.

The model proposed in [15] also shares the same objective, but it is not possible to compare the performance since they only present answer inclusion rate, which is defined to be the ratio that an answer eojeol is contained in K candidates. In the work of [15], it is stated that the answer inclusion rate is 95.9% when $K = 5$ and 97.2% when $K = 10$. In [13] and [14], the answer inclusion rate is 99.38% when $K = 6.60$ and 99.38% when $K = 5.92$, respectively. Through those results, we can indirectly compare the model with ours, and validate that our model outperforms it.

In addition, our model also performs better than previous methods in predicting unknown words[5]. The model of [14] analyzes unknown words with the accuracy of 65.81% with failure rate of 0.74%, which is compared to our model which analyzes unknown words correctly with 67.25% accuracy with no failure. We suppose that the improvement on analyzing unknown words is due to the ability of our model to capture a microscopic modification and to learn general morphological aspects by using grapheme as a unit.

5. Conclusion

In this paper, we presented a novel methodology for Korean morphological analysis by introducing grapheme as

5. The term 'unknown word' indicates the eojeol containing a morpheme which is not appeared in the training data.

TABLE 3. SELECTED SAMPLES SHOWING THE INFLUENCE OF THE CRF LAYER

Eojeol	**Without CRF**	**With CRF**	**Answer**
ϵyeoneu (여느; ordinary)	ϵ/MM + *yeon*/NNG + *eu*/MM (ㅇ/MM + ㅕㄴ/NNG + ㅡ/MM)	ϵyeoneu/MM (여느/MM)	ϵyeoneu/MM (여느/MM)
mose (모세; Moses)	*m*/NNP + *o*/NNG + *s*/MAG + *e*/NNP (ㅁ/NNP + ㅗ/NNG + ㅅ/MAG + ㅔ/NNP)	*mose*/NNP (모세/NNP)	*mose*/NNP (모세/NNP)
chaϵuseseuku (차우세스쿠; Ceausescu)	*chang*/NNG + *useseuk*/NNP + *u*/NNG (창/NNG + ㅜ세슥/NNP + ㅜ/NNG)	*chaϵuseseuku*/NNP (차우세스쿠/NNP)	*chaϵuseseuku*/NNP (차우세스쿠/NNP)

TABLE 4. END-TO-END ACCURACY

Method	**Eojeol**	**Sentence**
[13]	92.96	-
[17]	93.12	-
[14]	92.95	-
Proposed method	**94.89**	61.00

a processing unit. The key aspect of our model is that it does not rely on external knowledge. All components of the model are learned solely from the corpus, so the model is compact and easy to adopt another corpus with heterogeneous annotation standards.

From the experimental results, we showed that our model outperforms previous morphological analysis models. In both tasks, we found that algorithms considering the bidirectional dependencies improves accuracy greatly in the agglutinative language. We also observed that introducing a conditional random field (CRF) layer to the network used in part-of-speech tagging algorithm improves the performance substantially.

For future work, we suggest to design a method by which it does not need to preprocess the corpus to prepare training data for lexical form recovery. There exist models which are able to predict sequences with unaligned dataset, such as connectionist temporal classifier (CTC) [24] and RNN transducer [25] built for speech recognition, or encoder-decoder RNNs [26], [27] built for machine translation. Although using CTC or RNN transducer is problematic in our case where a target sequence is longer than an input sequence and encoder-decoder RNNs were found to be inefficient in our preliminary experiments, improvements on these architectures would make the model more robust and accurate.

Acknowledgment

This work was supported by BK21 Plus for Pioneers in Innovative Computing (Dept. of Computer Science and Engineering, SNU) funded by National Research Foundation of Korea (NRF) (21A20151113068).

References

[1] S.-S. Kang and Y. T. Kim, "A computational analysis model of irregular verbs in Korean morphological analyzer," *Journal of the Korea Information Science Society*, vol. 19, no. 2, pp. 151–164, 1992.

[2] D.-B. Kim, S.-J. Lee, K.-S. Choi, and G.-C. Kim, "A two-level morphological analysis of Korean," in *Proceedings of the 15th International Conference on Computational Linguistics*. Association for Computational Linguistics, 1994, pp. 535–539.

[3] S.-z. Lee, J.-i. Tsujii, and H.-C. Rim, "Hidden markov model-based Korean part-of-speech tagging considering high agglutinativity, word-spacing, and lexical correlativity," in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2000, pp. 384–391.

[4] S.-H. Na, S.-I. Yang, C.-H. Kim, O.-W. Kwon, and Y.-K. Kim, "CRFs for Korean morpheme segmentation and POS tagging," in *Proceedings of the 24th Annual Conference on Human and Cognitive Language Technology*, 2012.

[5] S.-H. Yang and Y.-S. Kim, "A high-speed Korean morphological analysis method based on pre-analyzed partial words," *Journal of KIISE: Software and Applications*, vol. 27, no. 3, pp. 290–301, 2000.

[6] K. Shim and J. Yang, "High speed Korean morphological analysis based on adjacency condition check," *Journal of KIISE: Software and Applications*, vol. 31, no. 1, pp. 89–99, 2004.

[7] K. Shim, "Morpheme restoration for syllable-based Korean POS tagging," *Journal of KIISE: Software and Applications*, vol. 40, no. 3, pp. 182–189, 2013.

[8] C. Lee, "Joint models for Korean word spacing and POS tagging using structural SVM," *Journal of KIISE: Software and Applications*, vol. 40, no. 12, pp. 826–832, 2013.

[9] C.-H. Lee, J.-H. Lim, S. Lim, and H.-K. Kim, "Syllable-based korean POS tagging based on combining a pre-analyzed dictionary with machine learning," *Journal of KIISE: Software and Applications*, vol. 43, no. 3, pp. 362–369, 2016.

[10] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[11] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[12] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *arXiv preprint arXiv:1603.01360*, 2016.

[13] D.-G. Lee and H.-C. Rim, "Probabilistic models for Korean morphological analysis," in *Companion to the Proceedings of the International Joint Conference on Natural Language Processing*, 2005, pp. 197–202.

[14] ——, "Probabilistic modeling of Korean morphology," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 5, pp. 945–955, 2009.

[15] J.-S. Lee, "Three-step probabilistic model for Korean morphological analysis," *Journal of KIISE: Software and Applications*, vol. 38, no. 5, pp. 257–268, 2011.

[16] C.-H. Han and M. Palmer, "A morphological tagger for Korean: Statistical tagging combined with corpus-based morphological rule application," *Machine Translation*, vol. 18, no. 4, pp. 275–297, 2004.

[17] Y.-M. Ahn and Y.-H. Seo, "Korean part-of-speech tagging using disambiguation rules for ambiguous word and statistical information," in *Convergence Information Technology, 2007. International Conference on*. IEEE, 2007, pp. 1598–1601.

[18] S.-H. Na and S. Jung, “Deep learning for Korean POS tagging,” in *Proceedings of the Korean Information Science Society 2014 Winter Conference*, 2014, pp. 426–428.

[19] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, “Learning precise timing with LSTM recurrent networks,” *Journal of Machine Learning Research*, vol. 3, no. Aug, pp. 115–143, 2002.

[20] S. B. Needleman and C. D. Wunsch, “A general method applicable to the search for similarities in the amino acid sequence of two proteins,” *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, 1970.

[21] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Cognitive Modeling*, vol. 5, no. 3, p. 1, 1988.

[22] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[23] The Theano Development Team, “Theano: A Python framework for fast computation of mathematical expressions,” *arXiv preprint arXiv:1605.02688*, 2016.

[24] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in *Proceedings of the 23rd International Conference on Machine learning*, 2006, pp. 369–376.

[25] A. Graves, “Sequence transduction with recurrent neural networks,” *arXiv preprint arXiv:1211.3711*, 2012.

[26] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

[27] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Advances in neural information processing systems*, 2014, pp. 3104–3112.