논문 2022-17-25

딥러닝을 이용한 한국어 Head-Tail 토큰화 기법과 품사 태깅

(Korean Head-Tail Tokenization and Part-of-Speech Tagging by using Deep Learning)

김 정 민, 강 승 식, 김 혁 만*
(Jungmin Kim, Seungshik Kang, Hyeokman Kim)

Abstract : Korean is an agglutinative language, and one or more morphemes are combined to form a single word. Part-of-speech tagging method separates each morpheme from a word and attaches a part-of-speech tag. In this study, we propose a new Korean part-of-speech tagging method based on the Head-Tail tokenization technique that divides a word into a lexical morpheme part and a grammatical morpheme part without decomposing compound words. In this method, the Head-Tail is divided by the syllable boundary without restoring irregular deformation or abbreviated syllables. Korean part-of-speech tagger was implemented using the Head-Tail tokenization and deep learning technique. In order to solve the problem that a large number of complex tags are generated due to the segmented tags and the tagging accuracy is low, we reduced the number of tags to a complex tag composed of large classification tags, and as a result, we improved the tagging accuracy. The performance of the Head-Tail part-of-speech tagger was experimented by using BERT, syllable bigram, and subword bigram embedding, and both syllable bigram and subword bigram embedding showed improvement in performance compared to general BERT. Part-of-speech tagging was performed by integrating the Head-Tail tokenization model and the simplified part-of-speech tagging model, achieving 98.99% word unit accuracy and 99.08% token unit accuracy. As a result of the experiment, it was found that the performance of part-of-speech tagging improved when the maximum token length was limited to twice the number of words.

Keywords : Morphological Analysis, Tokenizer, POS Tagging, Head-Tail, Deep Learning, BERT

Ⅰ. 서 론

기계이해 등 언어처리 시스템에서는 입력 자질로 형태소 단위 또는 부분단어 (subword) 토큰화 기법을 사용한다. 한국어는 교착어로써 어휘형태소와 문법형태소가 하나의 어절을 구성하여 문장내 관계나 문법적인 역할을 한다. 형태소 단위 토큰화 기법에서는 불규칙 용언이나 탈락 현상에 의해 원형 변형이 일어난 어절을 형태소 분석을 통해 어휘형태소와 문법형태소의 원형을 복원하고, 원형이 복원된 형태소가 토큰이 된다. 예를 들어, “니트는 당초 수능 영어를 2015년부터 대체할 계획이었지만 2019년까지 미뤄지는 등 난항을 겪어왔다.”에서 “미뤄지는”이라는 어절은 “미루다”라는 어휘형태소와 “어+지+는”이라는 문법형태소가 결합하여 축약 현상이 발생하였고, 형태소 분석기는 각 형태소에 대한 원형 복원을 하게 된다. 품사 태깅 (part-of-speech tagging)은 어절을 구성하는 각 형태소에 대해 품사를 결정하는 작업이다. 한국어의 품사 태깅은 어절을 구성하는 각 형태소를 분리하는 과정에서 원형을 복원한 후에 품사를 부착하는 방식을 취하였다. 그런데 형태소 단위의 품사 태깅은 복합명사를 단위명사로 분리하기도 하고, “어지는”과 같이 중요도가 낮은 복합형태소를 “어+지+는”이라는 단위형태소들로 분해함으로써 문장내에서 문법적인 관계를 파악할 때 2개 이상의 토큰으로 처리하기 때문에 문제를 더 복잡하게 만들기도 한다. 딥러닝 언어처리에서는 “미뤄지는”이 “미루_어_지_는”으로 토큰화되어 불필요하게 많은 토큰으로 분할하여 입력 토큰의 길이가 길어진다.

본 연구에서는 어절의 토크나이저 문제를 ‘어휘형태소 부분’을 Head로 정의하고, 나머지 ‘문법형태소 부분’을 Tail로 정의하여 음절단위로 Head와 Tail을 분리하여 한 어절이 두 개의 토큰만을 가지는 Head-Tail 토큰화 기법과 Head와 Tail 기반의 품사 태깅 기법을 제안한다. 모든 어절의 자질 추출 문제를 Head-Tail 토큰 두 가지로만 정의하였기 때문에 하나의 문장에서 발생되는 토큰의 최대 길이는 어절 개수의 두배 이하로 생성된다. 따라서 기존의 형태소 단위의 토큰화 기법에 비해 각 문장들의 최대 토큰 길이가 줄어드

*Corresponding Author (hmkim@kookmin.ac.kr)
Received: Jul. 7, 2022, Revised: Aug. 2, 2022, Accepted: Aug. 8, 2022.
J.M. Kim: Kookmin University (MS, Student)
S.S. Kang, H.M. Kim: Kookmin University (Prof.)
※ 본 논문은 정부 (과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2021R1F1A1061433).

ⓒ IEMEK J. Embed. Sys. Appl. 2022 Aug. 17(4) 199-208
ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2022.17.4.199

는 장점이 있다. 기존의 딥러닝을 이용한 형태소 분석기에서 원형복원 문제를 sequence-to-sequence 방식으로 해결할 때 미등록어 분석시에 원문의 변형이 일어날 수 있는데 비해, 제안하는 모델은 원형복원을 하지 않고 음절 단위로 토큰화하고 품사 태깅을 수행하기 때문에 원문의 변형이 일어나지 않는 장점이 있다.

Ⅱ. 관련 연구

딥러닝 이전 시대에서부터 문서분류, 개체명 인식기, 댓글 필터링, 감성 분류, 긍정 부정 분류, 문서 클러스터링, 기계번역, 검색 엔진 등 NLP 태스크를 다양한 모델에 적용 시켜왔다 [1-3]. 자연어처리는 다른 분야에 비해 모델 학습에 필요한 자질을 언어의 중의성문제와 모호성 문제 때문에 추출하기가 까다롭다. 한국어는 어휘형태소에 다양한 문법형태소가 붙어 관계를 부여하거나 문법적 기능을 하면서 형태소의 원형의 변형이 일어난다. 그렇기 때문에 하나의 어절에서 두개 이상의 형태소가 결합되어 있는 형태를 취한다. 이러한 결합된 형태소를 분리하기 위한 것이 형태소 분석기이다 [4-6].

BERT는 다음 단어를 예측하는데 현재 단어에서 관련 있는 정보에만 집중하는 attention [7] 기법 중 self-attention 기법을 이용한 sequence-to-sequence 모델인 트랜스포머(transformer) [8]에서 문장의 토큰과 문맥 간의 연관성을 분석하는 인코더 (encoder)와 인코더의 출력을 입력으로 하여 문장을 생성하는 디코더 (decoder) 중 인코더를 채택하여 사전 학습하였다. 인코더는 입력 임베딩이 문장 안의 i 시점의 토큰이 다른 토큰들에 얼마나 영향을 주는지 score를 책정하기 위해 입력 문장 자신에 대해 attention을 수행하여 attention score를 계산한다. 이를 self-attention이라고 한다. BERT는 multi-head self-attention 기법으로 양방향 문맥을 학습한 후, 필요한 태스크에 적합하게 추가학습(fine-tuning)하여 사용한다 [9].

Ⅲ. Head-Tail 말뭉치

Head-Tail 토크나이저는 원형이 변형된 어절에서 의미를 가지는 어휘형태소 음절 경계 부분을 Head로 정의하며, 나머지 부분은 어휘 형태소로 정의하여, 하나의 어절에서 발생할 수 있는 토큰의 개수는 1~2개 이다. 문법적 의미를 가지는 Tail의 토큰 경우는 하나의 문법형태소가 여러 개의 문법형태소로 이루어져 있어. Tail의 형태소 태그는 Tail을 구성하는 모든 어휘형태소의 태그를 '_'로 구분하여 복합 태그로 사용한다.

Head-Tail 토크나이저와 품사 태거를 위한 데이터셋으로 품사 태깅 말뭉치를 Head-Tail 형식으로 변환하여 사용하였다. 이 말뭉치는 국민대에서 구축한 KCC150원시 말뭉치[1]를

표 1. KCC150 형태 분석 말뭉치 예제
Table 1. A Sample of KCC150 Morphological Analysis Corpus

1 니트는 니트+는 NNG+JX
2 당초 당초NNG
3 수능 수능 NNG
4 영어를 영어+를 NNP+JKO_
5 2015년부터 2015+년+부터 SN+NNB+JX
6 대체할 대체하+ㄹ VV+ETM
7 계획이었지만 계획+이+었+지만 NNG+VCP+EP+EC
8 2019년까지 2019+년+까지 SN+NNB+JX
9 미뤄지는 미루+어+지+는 VV+EC+VX+ETM
10 등 등 NNB
11 난항을 난항+을 NNG+JKO
12 겪어왔다 겪+어+오+았+다 VV+EC+VX+EP+E
13 . . SF

표 2. Head-Tail 말뭉치 예제
Table 2. A Sample of Head-Tail Corpus

Head-Tail Token:
니트+는 당초 수능 영어+를 2015년+부터 대체할
계획+이었지만 2019년+까지 미뤄+지는 등 난항+을
겪+어왔다.
Head-Tail POS-TAG:
NNG+JX NNG NNG NNP+JKO SN_NNB+JX VV_ETM
NNG+VCP_EP_EC SN_NNB+JX VV+EC_VX_ETM NNB
NNG+JKO VV+EC_VX_EP_EF SF

가공하여 부산외대에서 구축한 품사 태깅 말뭉치[2]이다. 표 1은 KCC150 품사 태깅 말뭉치의 예이며, 표 1에서 3번째 열의 각 어절에 대한 형태소 분석 결과에서 첫번째 어휘 즉 어휘형태소 부분의 길이만큼 어절의 앞 경계부분을 어휘형태소 부분을 Head로 정의하고 나머지 부분을 문법형태소 Tail로 정의한다.

표 2는 표 1의 말뭉치 데이터에 대한 Head-Tail 말뭉치 구축의 결과이다. 7번째 어절 '계획이었지만'을 보면 '계획+이+었+지만'으로 토큰화된 것을 보여준다. 토큰화 결과의 첫번째 토큰 '계획'의 길이인 2음절만큼 어절을 분할하여 Head는 '계획', Tail은 '이었지만'이 된다. 해당 어절은 'NNG+VCP+EP+EC' 태그로 구성되었으므로, '계획'의 태그는 'NNG', '이었지만'의 태그는 'VCP_EP_EC'이 된다.

Ⅳ. Head-Tail 토큰나이저와 품사 태깅 모델

본 연구에서 제안하는 모델의 Head-Tail 형태소 분석 과정을 도식화한 구성도는 그림 1과 같다. 1차적으로, 입력문장은 특수 문자를 제거하는 전처리 과정을 거쳐 음절 임베딩 기법에 의해 임베딩 벡터로 구성된다. 토크나이저의 입력으로 입력 문장의 음절 임베딩이 입력되며 토크나이저는

1) http://nlp.kookmin.ac.kr/kcc/

2) https://github.com/bufsnlp2030/BUFS-JBNUCorpus2020

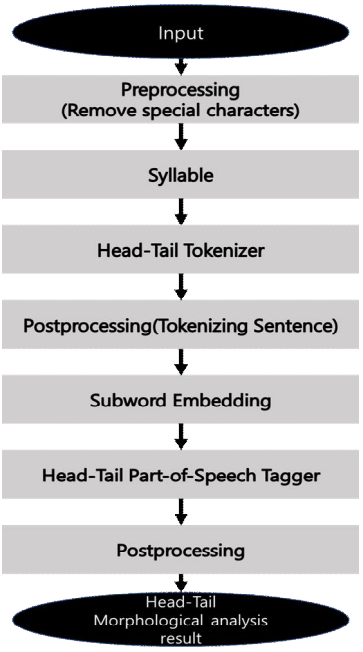

그림 1. Head-Tail 형태소 분석 과정
Fig. 1. Head-Tail Morphological Analysis Process

그림 2. Head-Tail 토크나이저의 입출력
Fig. 2. Head-Tail Tokenizer Input & Output

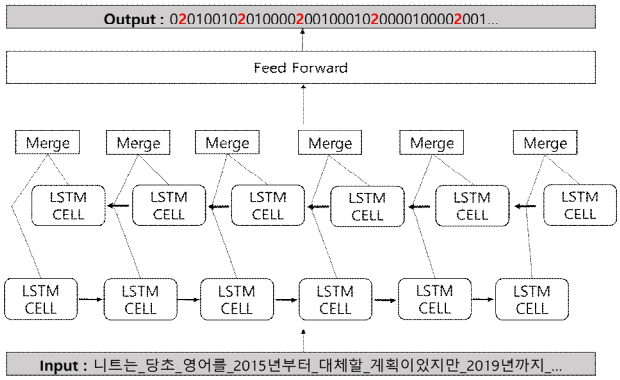

그림 3. BiLSTM을 이용한 Head-Tail 토크나이저 모델
Fig. 3. Head-Tail Tokenizer Model by using BiLSTM

Head-Tail 토큰화 결과를 출력한다. 토크나이저의 토큰화 결과를 Head-Tail 토큰 형식의 부분단어 임베딩으로 변환하여 품사 태거의 입력으로 전달하고, 품사 태거는 Head와 Tail에 대한 품사 태깅 결과를 출력한다. 이 단계가 품사 태깅 과정이며, 품사 태깅 결과를 후처리하여 Head-Tail 형태소 결과를 얻게 된다. 이처럼 토크나이저와 품사 태깅을 동시에 수행하는 방식으로 Head-Tail 단위의 형태소 분석을 수행한다.

1. 토크나이저 모델

Head-Tail 토큰화 기법에서는 Head와 Tail을 음절을 분해하지 않고 음절 경계로 어휘형태소와 문법형태소를 분리하기 때문에 Head와 Tail 부분으로 분리되는 음절 경계를 찾는 과정으로 정의된다. 이러한 음절 단위의 임베딩 기법은 복합명사나 자동 띄어쓰기에서 명사의 경계나 띄어쓰기가 일어나는 부분을 찾기 위해 사용되기도 한다 [10-12]. 불규칙 변형 또는 축약 어절의 경우에 원형을 복원하지 않고 음절 경계로 Head 부분과 Tail 부분을 분리하기 때문에 동일한 어휘형태소로 이루어졌더라도 변형이 일어난 Head와 Tail을 음절 단위로 토큰화하고, 복합형태소를 단위 형태소로 분해하지 않는다. 따라서 기존의 sequence-to-sequence 모델을 사용한 모델에서 복합명사 분해 오류와 미등록어의 형태소 분석 오류에 의해 원문이 훼손되는 문제를 완화할 수 있는 장점이 있다. 그림 2는 Head-Tail 토크나이저의 입력 출력 그리고 토큰화 결과를 보여주고 있다. '2'로 태깅된 태그와 결과값에서 '+' 좌측에 표시된 음절은 Head 경계의 마지막 음절을 의미한다. 그림 3은 Head-Tail 토크나이저의 모델 구성도이다. BiLSTM을 이용하여 모델을 구성하고 음절단위 토큰을 입력으로 어절의 Head-Tail 경계를 학습하였다. 전방향 LSTM와 역방향 LSTM의 출력값을 결합하기 위해 두 출력 값을 concat 하였다.

2. 품사 태깅 모델

2.1. 부분단어 토큰 단위의 bigram 임베딩

딥러닝의 이용한 NLP처리에서 자질로서 Ngram 토큰 임베딩 기법이 많이 사용되고 있으며 그 성능 또한 입증 되어 왔다 [13-15]. 기존의 bigram 자질 추출은 음절 단위 혹은 어절 단위로 이루어져 왔다. 음절단위 처리의 경우 희소한 bigram 토큰의 출현이 낮아지는 장점이 있지만 음절단위의 처리로 인해 그 의미가 소실될 우려가 있다. 어절단위 bigram 처리의 경우 의미를 가지게 되지만 한국어는 형태소의 변형 현상이 일어나게 되어 희소 토큰 (sparse token)의 출현이 많아져서 OOV 문제가 많은 빈도로 발생한다.

부분단어 단위 토큰의 경우 말뭉치에서 자주 출현한 음절 쌍들을 하나로 묶어 토큰화하기 때문에 음절단위보다는 의미를 가지는 토큰의 출현할 가능성이 높고 어절단위 처리보다는 희소 토큰의 출현이 줄어든다 [16, 17]. 본 연구에서는 토큰화된 문장에서 음절 단위와 어절단위의 장점을 일부 내포 하고 있는 부분단어 단위 토큰으로 bigram을 구성할 경우 음절 단위와 어절단위 bigram보다 성능이 높아질 것으로 예상하고, BERT 품사태거의 추가 자질로 사용하여 품사 태깅 실험을 하였다.

2.2 모델 구성

BERT에 자질을 추가하기 위해서 부분단어 단위로 bigram을 분석한다. 분석된 부분단어 bigram은 LSTM 네트워크 층에 전달되어 부분단어 bigram 임베딩을 얻게 되며,

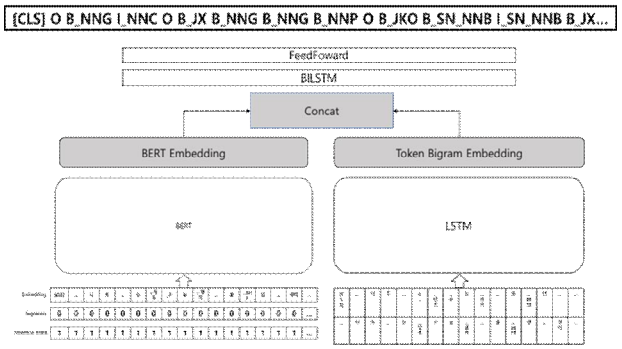

그림 4. Head-Tail 품사 태깅 모델
Fig. 4. Head-Tail Part-of-Speech Tagging Model

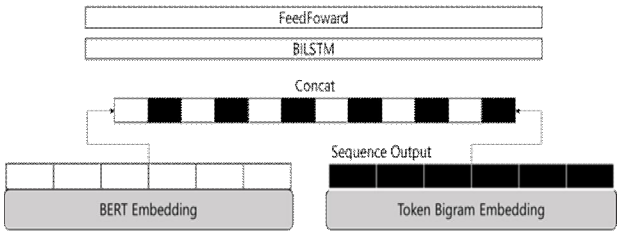

그림 5. Sequence Output Concat
Fig. 5. Sequence Output Concat

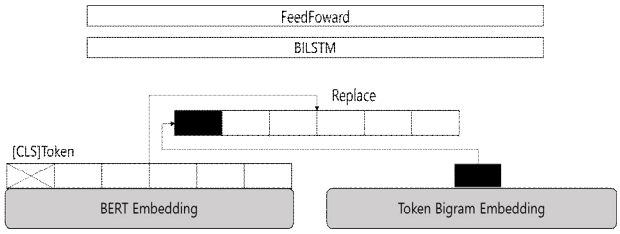

그림 6. One Output Embedding Concat (Replace)
Fig. 6. One Output Embedding Concat (Replace)

부분단어 bigram 자질이 가지는 의미 벡터와 BERT 벡터가 가지는 의미 임베딩을 하나로 묶기 위해 두개의 자질들을 concat 하여 벡터를 얻어 품사 태그를 학습하였다. 그림 4는 품사 태깅 모델의 구성도이다.

2.3. 임베딩 기법

본 연구에서는 BERT 임베딩과 bigram 임베딩의 의미를 잘 표현하는 벡터를 찾기 위해 bigram 임베딩과, BERT 임베딩을 concat 하는 2가지 방법을 사용하여 실험하였다.

1) Sequence output을 이용한 concat

부분단어 bigram의 LSTM 임베딩을 Sequence 형식으로 출력하여 그림 5와 같이 BERT 임베딩과 부분단어 토큰과 bigram 자질 임베딩 토큰을 1대1로 concat하여 bigram과 부분단어 토큰의 관계를 학습하도록 학습을 진행하였다.

2) CLS 토큰을 제거한 bigram 임베딩 concat

BERT의 첫번째 임베딩은 '[CLS]' 토큰에 대한 임베딩이다. '[CLS]' 토큰은 NSP 예측시 사용되는 NSP에 대한 라벨 임베딩이므로, '[CLS]' 토큰은 품사 태그 부착 대상이 아니다. '[CLS]' 토큰이 추가학습시에 관련성이 있는지 확인하기 위해, BERT 학습시에 그림 6과 같이 '[CLS]' 토큰 위치의 임베딩을 하나의 임베딩에 모든 bigram 정보를 기억하고 있는 LSTM output 임베딩으로 대치시켜 학습을 진행하였다.

V. 실험 및 성능평가

1. Head-Tail 품사 태깅 데이터셋

Head-Tail 품사 태깅 데이터셋은 Head-Tail 토큰과 Tag가 1:1로 매칭된다. BERT는 입력 토큰으로 Head-Tail 토큰 단위가 아닌 부분단어 토큰을 사용하므로 개체명 인식에서 주로 사용하는 BIO 태그 방식으로 태깅 데이터셋을 변형하였다. 'B'로 시작하는 태그는 품사 태그의 시작을, 'I'는 B태그와 이어지는 하나의 태그임을 의미하며, 'O'는 의미가 없는 태그임을 명시한다. 표 3은 BIO 태그 데이터셋 예이다.

1.1. 토크나이저 정확도

50,000 라인 (663,984 어절) 규모의 테스트 데이터에 대해 실험을 진행하였으며, 실험 결과는 표 4와 같다. Head-Tail 토크나이저를 BERT, BILSTM와 CRF [18], 통계적 기법인 TnT 태거 [19]에 대해서 테스트하였고, 정확도는 전체 어절에서 Head-Tail 토큰이 모두 일치한 어절에 대한 정확도이다. 실험 (1), (2)는 음절단위만을 입력으로 하였으며, 실험 (3)은 음절 BiLSTM 임베딩과 음절 bigram BiLSTM 임베딩을 concat 하였다, 실험 (4)는 sklearn-pycrfsuite를 사용하였고, i시점에서의 음절, i시점에서의 음절 trigram, i+1,i+2,i-1,i-2번째의 음절을 자질로 사용하였다, 실험 (5)는 NLTK (natural language tool kit)에서 제공하는 TnT 태거를 사용하였으며, 자질은 음절단위가 아닌 음절 trigram을 사용하였다. 문장의 끝 두 음절은 trigram을 이루지 못하여 해당 시점의 unigram 자질을 사용하였다.

표 3. Subword 토큰 단위의 BIO 데이터셋 예
Table 3. BIO Dataset Example by Subword Tokens

Train Corpus	Tag	Subword	BIO Tag
미뤄	VV	_미뤄	B_VV
지는	EC_VX_ETM	_지, 는	B_EC_VX_ETM, I_EC_VX_ETM

표 4. 토크나이저 모델별 정확도
Table 4. Accuracy of Tokenizer Models

Model	Correct Eojeal / Total 663,984 Eojeal	Accuracy (%)	Time
(1) BERT	659,670	99.40	486.49
(2) BILSTM	658,504	99.11	90.48
(3) BILSTM BIGRAM	660,691	99.47	141.49
(4) CRF [18]	654,031	98.42	70.65
(5) TnT [19]	654,784	97.09	137.80

실험 결과로 가장 높은 정확도를 보인 것은 실험 (3)의 음절 단위의 bigram 임베딩을 사용한 토크나이저 99.47%이다. 이 모델은 실행 시간이 3번째로 빠르고, 분석 시간도 BERT보다 빠른 성능을 보였다.

1.2. 품사 태깅 정확도

토큰화된 테스트 데이터셋의 품사 태깅 결과에 대한 모델별 정확도이다. 전체 50,000라인의 테스트 데이터에 대한 결과이며, 어절 개수는 663,984어절 Head-Tail의 토큰 개수는 1,030,058개이다. 표 5는 품사 태거 실험의 결과이다.

실험 (1)~(3)은 BERT를 이용하여 학습을 진행하였고, 실험 (1) 추가자질없이 BERT만을 사용하였고, 실험 (2)는 추가자질로 음절 bigram을 사용하여, [CLS] 토큰과 대치시켰다. 음절 bigram은 부분단어 토큰과 시퀀스의 길이가 다르고 위치 별 대응되는 토큰이 연관성이 없는 문제가 있어 부분단어 토큰과 1대1로 대응하여 합하여 자질을 주는 실험을 진행하지 않았다. 실험 (3)은 추가 자질로 부분단어 bigram을 사용하였으며, (3)-(a)는 부분단어 토큰과 부분단어 bigram을 1대1로 대응시키는 실험을 진행하였다. (3)-(b)는 부분단어 토큰 bigram을 하나의 임베딩으로 표현하여 BERT의 첫번째 토큰인 '[CLS]' 토큰과 대치시켰다. 아래부터는 해당 방법을 각각 sequence concat replace라고 칭한다. 실험 (4)는 부분단어 단위 태깅을 BiLSTM을 사용하여 수행하였고, (4)-(a)는 부분단어 토큰을 BiLSTM으로 출력한 값과 부분단어 토큰 bigram 값을 LSTM으로 출력한 값을 sequence concat하여 태그를 학습하여 태깅 정확도를 평가하였으며, (4)-(b)는 부분단어 토큰을 BiLSTM으로 출력한 값과 부분단어 bigram을 LSTM으로 출력한 값을 replace모델을 사용하여 태그를 학습하였다. 실험 (5)는 어절 단위로 통계적 기법에 기반한 TnT태거로 태그를 학습시키고 평가하였다.

표 5. 품사 태깅 모델별 정확도
Table 5. Accuracy of Part-of-Speech Tagging Model

Model	BERT			
Feature	(1) X (Only Bert)	(2) Syllable Bigram	(3) Subword Bigram	
Concat Mode	(a)	(a) Replace	(a) Sequence Concat	(b) Replace
Correct Word	651,566	653,244	653,017	**653,420**
Accuracy of Word	98.11	98.37	98.32	**98.38**
Correct Token	1,016,413	1,018,432	1,018,231	**1,018,665**
Accuracy of Token	98.67	98.87	98.84	**98.88**
Analysis Time (sec)	635.48	667.47	773.64	763.72

Model	BiLSTM		TnT
Feature	(4) Subword bigram		(5) Syllable unit
Concat Mode	(a) Sequence Concat	(b) Replace	X
Correct Word	645,760	640,917	661446
Accuracy of Word	97.28	96.45	92.63
Correct Token	1,010,222	1,005,861	1, 025,132
Accuracy of Token	98.10	97.60	94.86
Analysis Time (sec)	413.37	412.11	191.60

1.3. 토크나이저 + 품사 태거 정확도

형태소 분석의 과정은 토크나이저와 품사정보를 동시에 사용 가능한 경우도 있으며, 또한 필요 없는 토큰들을 제거하기 위해 품사정보를 이용해서 토큰을 제거하기 때문에 토크나이저와 태깅은 동시에 일어나는 것이 일반적이다. 그러므로 토크나이저와 태깅을 하나의 태스크로 간주하여 정확도와 실행 시간을 측정하였다. 또한 BiLSTM을 이용한 토크나이저로 품사 태깅을 하였을 때 BiLSTM 토크나이저와 정확도를 비교하기 위해 BERT 토크나이저에 대해서도 실험하였다. 품사 태거는 표 12의 (3)-(b)모델을 사용하였다.

표 6의 실험 (1)은 BiLSTM 토크나이저를 음절과 bigram 임베딩 자질을 사용하였을 때이며, BERT토크나이저 대비 음절과 bigram 임베딩을 사용한 BiLSTM 토크나이저의 성능상승률은 어절 정확도가 0.03%, 토큰 정확도가 0.03% 상승한 것을 볼 수 있다. 이와 비교하여 분석 시간과 정확도를 고려하였을 때 BiLSTM+bigram 토크나이저를 사용하는 것이 더 효율적임을 알 수 있다.

1.4. 태깅 오류 분석

표 7은 에러가 발생한 단일 복합 태그의 분포이며, 표 8은 오분류 태그에 대한 대분류 태그의 분포이다. 에러가 발생한 태그에 분포를 조사한 결과, 대분류와 순서를 모두 맞

표 6. 토크나이저 태거 통합 정확도
Table 6. Merge Accuracy of Tokenizer & Tagger

Evaluation (Tokenizer /POS Tagger)	Accuracy of Token & Tag after Tokenizing(%) / Accuracy of Token(%) / Accuracy of Tagging after Tokenizing(%)	
	(1)BiLSTM(Bigram)/BERT (CLS Token Replace)	(2)BERT/BERT (CLS Token Replace)
Word	**97.94 / 99.49 / 98.46**	97.91 / 99.42 / 97.93
Token	**98.36 / 99.44 / 98.46**	98.33 / 99.38 / 98.44
Total Analysis Time(sec) (Tokenizer /POS Tagger)	**157.56 / 795.27 Total 952.83**	480.72 / 794.98 Total 1275.7

표 7. 에러가 발생한 단일/복합 태그의 분포
Table 7. Distribution of Single/Complex Tags with Errors

Total Tag Count:1,267개	
Number of complex tags with errors	544
Number of simple tags with errors	37

표 8. 오분류 태그의 대분류 태그 분포
Table 8. Large Category Tag Distribution of Misclassified Tags

Number of incorrect tagging cases: 1,689	
Large category is matched	5,613
First category is matched	7,502
First to Second category is matched	1,172
Last category is matched	7,391
From Last First to Second category is matched	1,226

춘 태그의 경우가 5,613개, 첫번째 토큰의 첫번째 대분류 태그를 하나라도 맞춘 경우는 7,502건, 대분류를 앞에서 두개의 태그라도 맞춘 경우는 1,172건이며, 모두 틀린 경우는 3,891건이며, 마지막 태그 하나를 맞춘 경우 7502건, 마지막 태그 2개를 맞춘 경우 1,226건이며, 맞춘 경우의 수는 각 경우의 수에 중복 허용되었다. 발생한 에러에서 잘못 예측한 태그에 대한 경우의 수는 1,689개로 그 중에서 전체 정답 태그의 수는 581개 그중 정답 복합태그의 수는 544개로 거의 발생한 에러는 대부분 복합태그에서 일어나고 있다. 이러한 분포로 보아 복합태그로 인한 태그 개수는 1,267개로써 분류하여야 할 태그가 많이 존재함으로써 토큰의 복합태그를 예측하는데 방해가 되고 있음을 관찰하였다.

태그가 많음은 의미를 파악할 때에도 복잡해지는 문제가 있다. 그러므로 복합태그의 구성을 세부태그로 구성을 하는 것이 아닌 복합태그의 대분류로만 태그를 축소하여 실험을 평가하였다. 예를 들어, EC_VX_ETM 같은 태그의 경우 연결어미, 보조용언, 관형형 어미로 이루어져 있는 경우에 이를 어미, 용언, 어미의 구성인 E_V_E로 축소하여 성능을 평가하였다.

2. 태그 간소화 후 태깅 정확도

표 9는 태그 간소화 후 토큰화된 학습 데이터에 대해 BERT 모델에 대한 태깅 정확도이다. (1)-(a)는 BERT만을 사용하였을 때, (2)-(a)는 음절 bigram replace, (3)-(a)는 subword bigram sequence concat, (3)-(b)는 subword bigram replace 모델을 사용하였다. 태그를 간소화함으로써 발생한 태그 수는 304개로 간소화전인 1267개 보다 963개의 태그가 감소하였다. 태그를 간소화함으로써 전체적으로 추론시간이 줄어든 것을 알 수가 있다. 그리고 간소화 전 태그 정확도가 98% 대에 머물던 것에 비해서 간소화 후 태그의 복잡성이 줄어 99%대로 향상된 것을 알 수가 있다.

하지만 추가 자질을 사용하였을 때 간소화전 가장 높은 성능을 보여주었던 subword bigram replace 모델에 대해서는 일반 BERT 모델에 비해 성능이 정확도가 3번째로 높아 음절 bigram 자질 replace 모델에 비해 성능 향상이 없었다. 태그 간소화 전과는 반대로 음절 bigram 자질 replace 모델이 99.45% 어절 정확도 99.60% 토큰 정확도로 일반 간소화 태그 BERT 모델 대비 어절 정확도가 0.05%, 토큰 정확도가 0.03% 향상되었다. 이는 학습 시점에 따라 충분히 변화할 수 있는 수치이므로 자질로 인한 성능 향상이라고 단정하기 어렵다.

표 10은 토큰화 후 간소화된 태그의 통합 토크나이저 태깅 정확도이며, 품사 태거는 간소화 전과 같은 subword bigram replace 모델 (표 18 (3)-(b) 모델)을 사용하였다. 음절 bigram BiLSTM 토크나이저를 사용한 경우 간소화 전 어절 단위의 토큰화-태깅 정확도가 97.94%에서 98.99%로 토큰단위 토큰화-태깅 정확도는 98.36%에서 99.08%로 어절 정확도는 1.05% 토큰 정확도는 0.72% 향상되었으며, 분석 시간 또한 952.83초에서 614.4초로 338.43초 빨라졌다.

표 9. 태그 간소화후 품사태거 모델별 정확도
Table 9. Accuracy by Model after Tag Simplification

			Correct Word		Accuracy of Word		Correct Token		Accuracy of Token		Analysis Time (sec)	
Model	Feature	Concat Mode	asis	tobe	asis	tobe	asis	tobe	asis	tobe	asis	tobe
BERT	(1)	(a)	651,566	660,041	98.11	99.40	1,016,413	1,025,548	98.67	99.57	635.48	530.26
	(2)	(a)	653,244	660,403	98.37	99.45	1,018,432	1,025,968	98.87	99.60	667.47	558.31
	(3)	(a)	653,017	660,281	98.32	99.44	1,018,231	1,025,817	98.84	99.59	773.64	549.66
		(b)	653,420	660,172	98.38	99.43	1,018,665	1,025,714	98.88	99.58	763.72	549.62

표 10. 태그 간소화 후 토크나이저 태거 통합 정확도
Table 10. Accuracy of Tokenizer and Tagger after Tag Simplification

Evaluation (Tokenizer /POS Tagger)		Accuracy of Token & Tag after Tokenizing(%) / Accuracy of Token(%) / Accuracy of Tagging after Tokenizing(%)	
		(1)BiLSTM(Bigram)/BERT (CLS Token Replace)	(2)BERT/BERT (CLS Token Replace)
Word	asis	97.94/99.49/97.96	97.91/99.42/97.93
	tobe	**98.99/99.49/99.01**	98.94/99.43/98.96
Token	asis	98.36/99.44/98.46	98.33/99.38/98.44
	tobe	**99.08/99.44/99.18**	99.03/99.39/99.15
Total Analysis Time(sec) (Tokenizer /POS Tagger)	asis	952.83	1275.7
	tobe	614.4 (159.06/455.34)	946.63 (491.02 / 455.61)

표 11. 품사 태깅 모델의 성능 평가
Table 11. Performance Evaluation of Part-of-Speech Tagging Model

		Tokenizing	POS Tagging		Tokenizing+Tag	
		Accuracy of Word	Accuracy of Word	Accuracy of Token	Accuracy of Word	Morph (Token) F1-score
Subword Bigram Replace	asis	**99.49**	**97.96**	**98.46**	**97.94**	**98.41**
	tobe	**99.49**	**99.01**	**99.18**	**98.99**	**99.10**
(1) 윤준영 [20]		99.99	97.36	-	95.99	97.94
(2) 이건일 [21] (형태소+ 태그)		-	-	-	95.40	96.91
(3) 최용석 [22] (형태소+ 태그)		-	-	-	-	98.27
(4) 김정민 Head-Tail [23]		-	-	99.39	-	-
(5) 김동명 [24]		-	97.78	-	-	-
(6) 김재훈 [25]		-	95.85	97.68	-	-
(7) 심광섭 [26]		-	96.31	-	-	-

3. 기존 모델과 비교

표 11은 Head-Tail과 다른 형태소 분석기 혹은 품사 태거에 대한 실험 결과이다. 태그 간소화 전과 비교하였을 때 가장 높은 정확도를 보인 것은 김정민 (4)의 Head-Tail 품사 태거이다. 학습데이터는 대용량 말뭉치 7,882,000 라인을 사용하였다. 이는 본 연구에서 사용하는 200,000 라인과는 많은 차이로 인한 것으로 고려된다.

(1) 의 윤준영은 품사 어절 정확도 97.36, 형태소 품사 태그 동시 수행 어절정확도 95.99, F1-score 97.94로 본 연구의 모델과 0.6의 품사 어절 정확도, 1.95의 형태소+품사 어절 정확도, 0.47의 형태소+품사 F1-score의 차이를 보인다. 토큰화 어절 정확도는 99.99로 Head-Tail 토크나이저보다 0.5높다.

(2), (3)의 이건일, 최용석의 경우 토큰화 및 태깅을 동시에 수행하여 토큰+태그 정확도만 표기하고 있다. 본 연구의 모델과 (2)는 2.54의 어절 정확도, 1.5의 F1-score 차이를 보인다. (3)의 경우 0.14의 F1-score 차이를 보인다.

(5)의 김동명은 HMM 모델을 이용하여 일반 어휘와 동형이의어를 동시에 품사 태깅하여 품사 어절 정확도 97.78로 제안한 모델의 97.96과 비교해 0.18 낮은 정확도를 보이는 것을 알 수 있다.

(6)의 김재훈은 품사 어절 정확도 95.85로 본 연구에 비해 2.11 낮은 정확도를 보이고 있으며, 토큰 (형태소) 단위 정확도는 97.68로 본 연구의 98.46보다 0.78 낮은 정확도를 보이고 있다.

(7)의 심광섭은 형태소 분석기를 사용하지 않고 음절 단위에 대해 품사 태깅을 하였다. 그 결과 품사 어절 정확도 96.31을 달성하였고 제안한 방법이 1.65 높은 성능임을 알 수 있다.

태그 간소화 후는 윤준영 (1)은 1.65의 품사 어절 정확도, 3.0 형태소+품사 어절 정확도, 0.47의 형태소+품사 F1-score 차이를 보였으며, 이건일 (2)의 경우 3.59 어절 정확도, 2.19 F1-score차이, 최용석 (3)의 경우 0.83의 F1-score 차이이다. 김정민 (4)의 경우는 0.21의 품사 토큰 정확도 차이로 결과 (4)가 더 높은 것을 보인다. 김동명 (5)의 경우 1.23 품사 어절 정확도 차이를 김재훈 (6)의 경우 3.16의 품사 어절 정확도, 1.5의 토큰단위 정확도 차이를 가지는 것을 볼 수 있으며, 심광섭 (7)의 경우 2.7의 품사 어절 정확도 차이를 보이는 것을 알 수 있다. 서현재 (2022)는 BiLSTM 기법을 이용한 Head-Tail 품사 태깅 기법을 제안하였는데, 본 연구에서는 BERT 임베딩 기법을 이용한 점에서 차이가 있다 [27].

4. Head-Tail 실용성 평가

Head-Tail 토크나이저 & 품사 태거의 형태소로써 자질의 가치가 있는지 평가하기 위해 KoNLPy 라이브러리 의 형태소 분석기, KLT2000 형태소 분석기, Head-Tail 분석기로 토큰화하고 품사 태그를 부착하여 데이터를 생성하여 딥러닝 모델의 자질로 사용하였다. 데이터셋은 조단비 (2021)의 github 공개된 일베 데이터의 혐오성 분류 연구에서 라벨링된 데이터를 이용하여 실험을 진행하였다 [28].

표 12. 형태소 분석기별 분류 실험 결과
Table 12. Classification Results by Morphological Analyzer

		Okt (1)	Kkma (2)	Komoran (3)	KLT2000 (4)
Top Token Length		159	262	285	**161**
Avg Token Length		9.99	13.53	15.37	**16.50**
Token & Pos	Accuracy	81.58	80.62	80.73	**81.80**
	F1-score	83.98	83.15	83.27	**84.15**
Only Token	Accuracy	81.02	80.61	80.49	**81.35**
	F1-score	83.72	83.39	83.28	**83.88**

		(5)	(6)	(7)	(8)	(9)
Top Token Length		138		126		100
Avg Token Length		10.09		8.33		8.35
Token & Pos	Accuracy	80.82	80.93	80.92	80.94	81.73
	F1-score	83.41	83.52	83.65	83.56	84.27
Token Only	Accuracy	79.36		80.31		81.10
	F1-score	82.37		83.12		83.87

* (5)HT(간소화) (6)HT(세부태그) (7)Head(간소화) (8)Head(세부태그) (9)KLT2000 (9)(First morph)

Head-Tail 품사정보는 간소화되지 않은 태그와 간소화된 태그를 사용하였으며, Head-Tail 토크나이저는 BiLSTM bigram 토크나이저를 사용하였다. 간소화되지 않은 태그의 품사 태거와 간소화된 태그의 품사 태거로는 BERT-Subword-Bigram- Replace 모델을 사용하여 실험하였다. 표 12는 형태소 분석기의 분류 실험 결과이다. 비교 대상 모델은 한국어 형태소 분석기로 (1) Okt, (2) Kkma, (3) Komoran, (4) KLT2000을 사용하여 실험한 후 Head-Tail (형태소 분석) 실험 (5)~(8)과 비교하였다.

토크나이저와 품사 태깅의 성능을 비교하는 모델은 분류 모델에 대체적으로 많이 사용되고 있는 BiLSTM 분류기를 사용하였으며, 자질은 토큰과 품사 태그를 사용하였다. 형태소 분석기 중 가장 적은 토큰의 길이를 사용하는 것은 최대 길이 138의 Head-Tail 토크나이저이며, 토큰 길이를 가장 많이 할당하는 것은 최대 길이 285의 실험 (3)의 Komoran 형태소 분석기이다. 형태소 분석기의 가장 높은 성능을 보인 형태소 분석기는 정확도 81.80 (토큰 태그) 81.35 (토큰), F1-score 84.15 (토큰+태그) 83.88 (토큰)을 기록한 실험 (4)의 KLT2000 형태소 분석기이다.

Head-Tail 형태소를 사용한 실험 (5)~(6)은 정확도 80.82%/80.93% (토큰 태그, 태그 간소화 / 세부 태그), 79.36 (토큰만 사용) F1-score 83.41/83.52 (토큰 태그, 태그간소화/세부태그) 82.37 (토큰만 사용)를 기록하였다. 또한, 실험 (7), (8)은 Head와 Tail 토큰 중에서 Tail 토큰은 문법형태소로 어휘형태소인 Head 토큰보다는 상대적으로 덜 중요한 정보이다. Tail 토큰의 중요도를 알아보기 위해 Head 토큰에 대해서만 분류 실험을 실시하였다. KoNLPy의 형태소 분

석기들은 분석 결과에서 어절경계 정보가 없어 어휘형태소만 추출이 어려워서 Head-Tail과 KLT2000에 대해서만 실험을 하였으며, 실험 (9)의 경우 KLT2000 어절 형태소 분석 결과의 첫번째 형태소만을 사용하였다.

Head 토큰과 간소화된 품사 태그를 사용한 실험 (7)의 경우 80.92의 정확도와 83.65의 F1-score를 보여주었으며, Head 토큰과 세부 태그를 함께 사용한 실험 (8)의 경우는 80.94의 정확도와 83.56의 F1-score를 보여주었다. 이는 Head-Tail 토큰을 모두 사용하였을 경우와 Head 토큰을 사용한 경우 Tail 토큰이 정확도에 미치는 영향은 간소화 태그에서 성능 향상이 있었으며, 세부태그를 사용하였을 경우는 성능 향상이 없었다. 품사 정보 없이 Head 토큰만을 사용하여 분류를 실험한 결과는 실험 (5), (6)의 Head-Tail 토큰을 모두 사용한 경우보다 Head만 사용한 경우에 정확도가 79.36%에서 80.31%로, F1-score는 82.37에서 83.12로 성능이 향상되었다. 이는 Head-Tail 토큰화에서 Tail 토큰은 상대적으로 중요하지 않은 문법형태소 정보이며, 문법형태소가 어휘형태소에 비해 상대적으로 중요하지 않은 정보임을 알 수 있었다. 문법형태소를 하나로 통합함으로 인해 토큰 개수가 많아지는 문제 때문으로 분석된다.

실험 (9)는 첫번째 형태소의 토큰과 품사정보를 사용하였을 경우 81.73%의 정확도와 84.27의 F1-score를 보여주었고, 토큰 정보만을 사용하였을 경우 81.10%정확도, 83.87의 F1-score를 보여주었으며 문법형태소를 포함하여 사용하였을 때인 실험 (4)와 비교하여 성능 향상은 없었다. Okt와 KLT2000의 경우를 제외한 나머지 두 개의 형태소 분석기 Kkma, Komoran의 경우에 토큰과 품사 태그를 모두 사용하여 실험을 진행한 경우는 Head-Tail 형태소 분석에 비해 낮거나 비슷한 성능을 보여주고 있다. 하지만 품사 정보없이 Head-Tail 토큰만을 사용하였을 경우에는 어절의 음절단위 처리로 인한 토큰 개수가 다른 형태소 분석 기법에 비해 많아져서 가장 낮은 성능을 보여주고 있다. 이는 KLT2000 대비 0.74/0.63(토큰+태그 자질 사용: 태그 간소화/세부 태그) 1.51 (토큰만 사용) F1-scroe 차이를 보이고, 또한 실험 (2), (3)의 형태소 분석기 대비 비슷하거나 더 높은 성능을 보이고 있다. KLT2000 토큰 길이와 (문장의 어절 100개로 제한) Head-Tail 토큰 길이 감소한 것에 비해 낮은 점수 하락을 보이는 것을 알 수 있으며, 이는 Head-Tail 기법이 딥러닝 모델에서 활용 가치가 있음을 알 수 있다.

Ⅵ. 결 론

형태소 분석 문제를 어휘형태소와 문법형태소 부분으로 구분하는 Head-Tail 토큰화 기반의 품사 태깅 기법을 제안하였다. 이 기법은 복합어를 단위 형태소들로 분해하지 않으며, 용언의 경우에 불규칙 원형 복원이나 축약된 음절을 분해하지 않기 때문에 원형 어간과 변형된 어간이 구분되는 특징이 있다. Head-Tail 토큰화 기법과 딥러닝 태깅 모델을 이용한 품사 태깅의 성능을 비교하였으며, 최대 토큰 길이를 어절 개수의 2배로 제한했을 때 성능이 향상됨을 확인하였다. Head-Tail 품사 태그에 대해 BERT와 음절 bigram, subword bigram 정보를 이용하여 품사 태깅 실험을 수행하였으며, 그 결과 음절 bigram과 subword bigram 모두 일반 BERT 대비 성능 향상 효과가 있음을 확인하였다. 세분화된 태그로 인해 방대한 복합 태그가 생성되고 태깅 정확도가 낮아지는 문제를 해결하기 위하여 대분류 태그로 구성된 복합태그로 태그수를 줄임으로써 태그 예측 정확도를 향상시켰다. Head-Tail 토큰화 모델과 간소화된 품사 태깅 모델을 통합하여 토크나이저가 토큰화한 문장에 대한 품사 태깅을 수행 결과로 어절단위 정확도 98.99%, 토큰단위 정확도 99.08%를 달성하였다. Head-Tail 형태소 분석과 기존의 형태소 분석기를 통한 비교 실험을 통해 BiLSTM 분류기에 대해서 실험을 실시하였으며, 기존의 형태소 분석기 대비 가장 적은 토큰 길이로 자질로서의 활용 가치가 충분함을 확인하였다. 추가로, Head-Tail 토큰 중에서 Tail 토큰을 학습에 사용하지 않는 실험에서는 Head 토큰만 사용하더라도 우수한 성능을 보일 수 있음을 확인하였다.

References

[1] D. H. Park, W. S. Choi, H. J. Kim, S. L. Lee, “Web Document Classification Based on Hangeul Morpheme and Keyword Analyses”, Journal of Korean Information Processing Society D, Vol. 19, No. 4, pp. 263-270, 2012 (in Korean).

[2] W. Cho, G. Shin, W. Lee, S. Son, H. Song, J. Lee, H. Lee, S. Jo, “KoELECTRA-Based Named Entity Recognition Using Korean Morphological Analyzers”, Proceedings of 2021 Korea Computer Congress, pp. 1897-1899, 2021 (in Korean).

[3] Y. Ha, J. Cheon, I. Wang, M. Park, G. Woo, “A Filtering Method of Malicious Comments Through Morpheme Analysis”, Journal of Korea Contents Association, Vol. 21, No. 9, pp. 750-761, 2021 (in Korean).

[4] S. S. Kang, “Analysis of Korean Irregular Verbs by using Syllable Characteristics”, Proceedings of 5th Hangul and Korean Information Processing, pp. 385-394, 1993 (in Korean).

[5] S. S. Kang, “Multi-level Morphology and Morphological Analysis Model for Korean”, Proceedings of 6th Hangul and Korean Information Processing, pp. 140-145, 1994 (in Korean).

[6] J. S. Lee, “Three-Step Probabilistic Model for Korean Morphological Analysis”, Journal of KIISE, Vol. 38, No. 5, pp. 257-268, 2011 (in Korean).

[7] D. Bahdanau, K. Cho, Y. Bengio, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015, arXiv:1409.0473, 2015.

[8] A. Vaswan, N. Shazeer., N. Parmar., J. Uszkoreit., L.

Jones.,N. Aidan. Gomez, L. Kaiser., L. Polosukhin., “Attention Is All You Need”, Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS), arXiv:1706.03762v5, 2017.

[9] J. Devlin., M. W. Chang., K. Lee., K. Toutanova., “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, preprint arXiv:1810.04805, 2018.

[10] H. Y. Lee, S. S. Kang, “Bi-LSTM-CRF and Syllable Embedding for Automatic Spacing of Korean Sentences”, Proceedings of 30th Hangul and Korean Information Processing, pp. 605-607, 2018 (in Korean).

[11] K. S. Shim, “Automatic Word Spacing based on Conditional Random Fields”, Korean Journal of Cognitive Science, Vol. 22, No. 2, pp. 217-233, 2011 (in Korean).

[12] H. Y. Lee, S. S. Kang, “Compound Noun Decomposition by using Syllable-based Embedding and Deep Learning”, Journal of Smart Media, Vol. 8, No. 2, pp. 74-79, 2019 (in Korean).

[13] J. H. Shin, H. R. Park, “A Statistical Model for Korean Text Segmentation by using Syllable-level Bigrams”, Proceedings of 9th Hangul and Korean Information Processing, pp. 255-260, 1997 (in Korean).

[14] S. S. Kang, “Automatic Correction of Word-spacing Errors by using Syllable Bigram”, Journal of Speech Science, Vol. 8, No. 2, pp. 83-90, 2001 (in Korean).

[15] D. H. Lim, Y. J. Chun, H. J. Kim, S. S. Kang, “Word Segmentation by using Extended Syllable Bigram”, Proceedings of 17th Hangul and Korean Information Processing, pp. 189-193, 2005 (in Korean).

[16] T. Kudo.., “Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates”, Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, pp. 66-75, 2018.

[17] T. Kudo, J. Richardson., “SentencePiece: A Simple and Language Independent Subword Tokenizer nd Detokenizer for Neural Text Processing”, Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System, pp. 66-71. 2018.

[18] J. Lafferty, A. McCallum, F. C. N. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data”, Proceedings of the 18th International Conference on Machine Learning 2001, pp. 282-289, 2001.

[19] T. Brants., “TnT - A Statistical Part-of-Speech Tagger”, Proceedings of the 6th Applied Natural Language Processing Conference, pp. 224-231, 2000.

[20] J. Youn, J. Lee, “A Deep Learning-based Two-Steps Pipeline Model for Korean Morphological Analysis and Part-of-Speech Tagging”, Journal of KIISE, Vol. 48, No. 4, pp. 444-452, 2021.

[21] J. Li, E. H. Lee, J. H. Lee, “Sequence-to-sequence based Morphological Analysis and Part-Of-Speech Tagging for Korean Language with Convolutional Features”, Journal of KIISE, Vol. 44, No. 1, pp. 57-62, 2017 (in Korean).

[22] Y. Choi, K. Lee, “Performance Analysis of Korean Morphological Analyzer based on Transformer and BERT”, Journal of KIISE, Vol. 47, No. 8, pp. 730-741, 2020 (in Korean).

[23] J. M. Kim, H. J. Seo, S. S. Kang, “Korean Head-Tail POS-Tagger by using Transformer”, Proceedings of 33th Hangul and Korean Information Processing, pp. 544-547, 2021 (in Korean).

[24] D. Kim, Y. Bae, C. Ock, H. Choi, C. Kim, “Korean POS and Homonym Tagging System using HMM”, Proceedings of 20th Hangul and Korean Information Processing, pp. 12-16, 2008 (in Korean).

[25] J. H. Kim, “Korean Part-of-Speech Tagging using a Weighted Network”, Journal of KIISE, Vol. 25, No. 6, pp. 951-959, 1998 (in Korean).

[26] K. S. Shim, “Syllable-based POS Tagging without Korean Morphological Analysis”, Korean Journal of Cognitive Science, Vol. 22, No. 3, pp. 327-345, 2011 (in Korean).

[27] J. M. Kim, H. J. Seo, S. S. Kang, “Korean Head-Tail POS-Tagger by using Transformer”, Journal of Smart Media, Vol. 11, No. 5, pp. 17-25, 2022 (in Korean).

[28] D. Cho, “Hate Speech Detection by using Word Embedding and Deep Learning Model”, M.S. Thesis, Kookmin University, 2021 (in Korean).

Jungmin Kim (김 정 민)

2017 Computer Science from Dongseo University (B.S.)
2022 Computer Science from Kookmin University (M.S.)
2022~AIRI Inc. (Researcher)

Field of Interests: Natural Language Processing, Machine Learning, Bigdata Analysis
Email: kimjm@kookmin.ac.kr

Seungshik Kang (강 승 식)

1986 Computer Engineering from Seoul National University (B.S.)
1988 Computer Engineering from Seoul National University (M.S.)
1993 Computer Engineering from Seoul National University (Ph.D.)
2001~Department of Artificial Intelligence at Kookmin University (Prof.)
Career:
1993~2001 Hansung University (Assoc. Prof.)
Field of Interests: Natural Language Processing, Machine Learning, Bigdata Analysis
Email: sskang@kookmin.ac.kr

Hyeokman Kim (김 혁 만)

1985 Computer Engineering from Seoul National University (B.S.)
1987 Computer Engineering from Seoul National University (M.S.)
1996 Computer Engineering from Seoul National University (Ph.D.)
1999~Department of Computer Science at Kookmin University (Prof.)
Career:
1996~1999 Korea Telecom
Field of Interests: XML Schema, Multimedia Database
Email: hmkim@kookmin.ac.kr