한국어 말덩이 정의와 구묶음: 한국어 말덩이 부착 말뭉치와 Bi-LSTM/CRFs 모델을 활용하여

Defining Chunks and Chunking using Its Corpus and Bi-LSTM/CRFs in Korean

저자 (Authors)
남궁영, 김창현, 천민아, 박호민, 윤호, 최민석, 김재균, 김재훈
Young Namgoong, Chang-Hyun Kim, Min-ah Cheon, Ho-min Park, Ho Yoon, Min-seok Choi, Jae-kyun Kim, Jae-Hoon Kim

출처 (Source)
정보과학회논문지 47(6), 2020.6, 587-595(9 pages)
Journal of KIISE 47(6), 2020.6, 587-595(9 pages)

발행처 (Publisher)
한국정보과학회
The Korean Institute of Information Scientists and Engineers

URL
http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE09353167

APA Style
남궁영, 김창현, 천민아, 박호민, 윤호, 최민석, 김재균, 김재훈 (2020). 한국어 말덩이 정의와 구묶음: 한국어 말덩이 부착 말뭉치와 Bi-LSTM/CRFs 모델을 활용하여. 정보과학회논문지, 47(6), 587-595

ISSN 2383-630X(Print) / ISSN 2383-6296(Online)
Journal of KIISE, Vol. 47, No. 6, pp. 587-595, 2020. 6
https://doi.org/10.5626/JOK.2020.47.6.587

한국어 말덩이 정의와 구묶음: 한국어 말덩이 부착 말뭉치와 Bi-LSTM/CRFs 모델을 활용하여

(Defining Chunks and Chunking using Its Corpus and Bi-LSTM/CRFs in Korean)

남 궁 영 [†] (Young Namgoong) 김 창 현 [††] (Chang-Hyun Kim) 천 민 아 [†] (Min-ah Cheon) 박 호 민 [†] (Ho-min Park)
윤 호 [†††] (Ho Yoon) 최 민 석 [†††] (Min-seok Choi) 김 재 균 [†††] (Jae-kyun Kim) 김 재 훈 [††††] (Jae-Hoon Kim)

요 약 한국어 의존구조를 분석하는 데에는 몇 가지 고질적인 문제가 있다. 그 중 하나는 중심어 위치 문제이고 다른 하나는 구성성분의 단위 문제이다. 이와 같은 문제는 구묶음을 수행함으로써 어느 정도는 해결된다. 구묶음은 형태소 분석과 구문분석의 중간 단계에 위치하면서 말덩이라 하는 구성성분을 찾는 과정이다. 본 논문에서는 한국어 말덩이의 정의와 의의를 살펴보고 한국어 말덩이 부착 말뭉치를 구축한다. 또한 본 논문에서는 구축된 말뭉치와 Bi-LSTM/CRFs를 이용한 한국어 구묶음을 제안한다. 실험을 통해서 제안된 구묶음 모델은 98.54%의 F1점수를 보여 실용적으로 사용할 수 있을 것으로 판단된다. 또한 다양한 입력 표상에 따른 성능을 분석하여 fastText가 가장 좋은 성능을 보였다. 또한 오류 분석을 통해 제안된 시스템의 문제를 분석하여 향후 시스템 개선에 적극 활용할 계획이다.

키워드: 구묶음, 입력표상, 심층학습, 구문분석

Abstract There are several notorious problems in Korean dependency parsing: the head position problem and the constituent unit problem. Such problems can be somewhat resolved by chunking. Chunking seeks to locate and classify constituents referred to as chunks into predefined categories. So far, several studies in Korean have been conducted without a clear definition of chunks partially. Thus, we define chunks in Korean thoroughly and build a chunk-tagged corpus based on the definition as well as propose a Bi-LSTM/CRF chunking model using the corpus. Through experiments, we have

· 이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(R7119-16-1001, 지식증강형 실시간 동시통역 원천기술 개발)과 2017년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2017M3C4A7068187, 한국어 정보처리 원천 기술 연구 개발)

† 학생회원 : 한국해양대학교 컴퓨터공학과 학생
young_ng@kmou.ac.kr
minah2018@kmou.ac.kr
homin2006@hanmail.net
†† 정 회 원 : 한국전자통신연구원
chkim@etri.re.kr
††† 비 회 원 : 한국해양대학교 컴퓨터공학과 학생
4168615@naver.com
ehdgus5136@naver.com
jgk20000@naver.com
†††† 종신회원 : 한국해양대학교 컴퓨터공학과 교수(KMOU)
jhoon@kmou.ac.kr
(Corresponding author임)

논문접수 : 2020년 3월 4일
(Received 4 March 2020)
논문수정 : 2020년 3월 25일
(Revised 25 March 2020)
심사완료 : 2020년 3월 29일
(Accepted 29 March 2020)

정보과학회논문지 제47권 제6호(2020. 6)

shown that the proposed model achieved a F1-score of 98.54% and can be used for practical applications. We analyzed performance variations according to word embedding and so fastText showed the best performance. Error analysis was performed so that it could be used to improve the proposed model in the near future.

Keywords: chunking, word embedding, deep learning, parsing

1. 서 론

한국어는 교착어이다. 따라서 영어와 달리 한 어절 내에도 의미적으로 중요한 부분과 그렇지 않은 부분이 있다. 이를 고려하지 않고 형태소들의 나열로만 펼쳐 놓은 뒤 구문분석 또는 의미분석을 하는 것은 계산적인 낭비일 뿐만 아니라 한국어의 언어적 특징과도 거리가 있다. 예를 들어, '할 수 있게 된다'라는 구절을 보면 '하', 'ㄹ', '수', '있', '게', '되', 'ㄴ다'와 같이 총 일곱 개의 형태소로 이루어져 있다. 이에 대해 구문을 분석하게 되면 그림 1[1]처럼 트리 형식으로 나타낼 수 있다.

또한 같은 구절에 대해 어절 단위로 의존구조 분석을 하면 그림 2[2]와 같다. 그림 2에서 각 노드는 어절을 나타내고, 화살표 꼬리는 지배소를 가리키며, 머리는 의존소를 가리킨다.

그림 1과 그림 2 모두 중심어 후위 원칙에 의거하여 '된다'를 중심어(head)로 분석하고 있다. 하지만 실제 이 예시에서 의미적 중심어는 '하다(하)'이다. 또한 의존명사인 '수'를 형용사 '있다(있게)'의 주어로 분석하게 되는데, 실질적 의미가 결여되어 있는 의존명사 '수'가 주어가 되는 문제가 발생한다.

이 같은 문제를 해결하기 위해 한국어에도 구묶음(chunking)이라는 개념을 적용할 수 있다[2-5]. 구묶음은 문장 내에서 말덩이(chunk)라고 하는 구성성분을 찾는 과정이다[2]. 말덩이는 문장 내에서 문법적으로나 의미적으로 같은 역할을 하는 일련의 형태소들을 말한다.

```
(VP (S (NP_SBJ (VP_MOD 하/VV + ㄹ/ETM)
(NP_SBJ 수/NNB))
(VP 있/VV + 게/EC))
(VP_MOD 되/VV + ㄴ다/EF))
```

그림 1 한국어 구문분석 예시

Fig. 1 An example of parsing in Korean

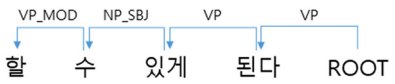

그림 2 의존구조 분석 결과 예시

Fig. 2 An example of dependency parsing

구묶음을 통해 구문분석의 복잡도와 정확도 면에서 효율성을 제고할 수 있으며 한국어의 특성을 살려 구문적 중심어와 의미적 중심어를 일치하도록 구문분석을 할 수 있다[6].

그동안 한국어처리에서는 구묶음의 대상이 되는 말덩이의 명확한 정의 없이 부분적으로 연구가 진행되어 왔다[3-5]. 본 논문에서는 한국어 말덩이를 구체화하여 그 정의 및 의의를 살펴 보고 한국어 말덩이 부착 말뭉치를 구축한다. 또한 본 논문에서는 Bi-LSTM/CRFs를 이용한 한국어 구묶음을 제안한다. 실험을 통해서 제안된 구묶음 시스템은 98.54%의 F1점수를 보였으며 실용적으로 사용할 수 있을 것으로 생각된다. 또한 다양한 입력 표상에 따른 성능을 분석하여 fastText가 가장 좋은 성능을 보였다.

논문의 구성은 다음과 같다. 2장에서는 한국어 구묶음의 정의 및 필요성에 대해 기술하고, 3장에서는 구묶음 과정을 자세히 기술한다. 4장에서는 실험을 통해서 구묶음기의 성능과 그 오류를 분석하고, 5장에서 결론과 향후 연구에 대해 논한다.

2. 한국어 구묶음의 정의 및 의의

구묶음은 문장 내에서 말덩이라고 하는 구성성분을 찾는 과정이다[2]. 말덩이는 문장 내에서 문법적으로나 의미적으로 같은 역할을 하는 일련의 형태소들을 말한다. 한국어 구묶음은 [7]에 의거해 수행할 수 있으며, 구묶음 수행결과로 나오는 말덩이는 표 1과 같이 크게 내용어 말덩이(content chunk)와 기능어 말덩이(function chunk)가 있다[8]. 또한, 한국어에서 말덩이 개념을 이용하면 입력 문장을 문장성분 단위로 나타낼 수 있게 된다.

예를 들어, 서론에서 들었던 '할 수 있게 된다'라는 예문에서 서로 같은 기능을 하고 있는 형태소들을 모아보면 실제적인 의미를 담당하고 있는 '하'와 이에 문법적 기능, 즉 보조용언 역할을 하는 'ㄹ 수 있게 되'와 종결어미를 담당하는 'ㄴ다'로 구묶음을 할 수 있다. 이때 의미를 담당하는 '하'는 내용어 말덩이, 후자의 두 경우는 기능어 말덩이에 속하게 된다. 그리고 결국 '할 수 있게 된다'는 구절은 그 전체가 어떤 문장에서 서술어 역할을 한다는 것을 알 수 있다. 다시 말해, 구묶음을 했을 때 내용어 말덩이 하나와 하나 이상의 기능어 말덩이가 하나의 문장성분을 이루게 된다. 이를 정규식으

1) 그림 1에서 사용된 표지들은 세종 형태 표지 및 구문분석 표지[1]이다.
2) 그림 2에서 사용된 표지는 그림 1과 마찬가지로 세종 구문분석 표지이며, '구문표지(_기능표지)'와 같이 구성되어 있다.

표 1 한국어 말덩이의 종류, 표지 및 예시
Table 1 Types, labels, and examples of Korean chunks

Chunk labels	Examples
Content chunks	
체언구(NX)	[헌/MM 책/NNG+들/XSN]을 읽었다.
본용언구(PX)	철저히 [시행/XR+되/XSV]고 있다.
지정사구(CX)	지켜야할 [도리/NNG+이/VCP]다.
부사구(AX)	[너무/MAG 너무/MAG] 좋았다.
관형사구(MX)	[저/MM] 예쁜 꽃
독립어구(IX)	[아/IC], 벌써 가을이 왔나 보다.
Function chunks	
격조사구(JKX)	친구[에/JKB 대하/VV+어/EC]
관형격조사구(JMX)	사람[을/JKO 위하/VV+느/ETM] 배려
보조사구(JUX)	이것 [만/JX+도/JX] 못하다.
접속조사구(JCX)	영희[뿐/JX+만/JX 아니/VA+라/EC]
호격조사구(JVX)	그대[여/JKV], 아무 걱정하지 말아요.
보조용언구(PUX)	배우[ㄹ/ETM 수/NNB 있/VV]다.
선어말어미구(EPX)	진지 잡수[시/EP+었/EP]어요.
연결어미구(ECX)	크[ㄹ/ETM 뿐/NNB 아니/VA+라/EC]
전성어미구(ETX)	먹고살[기/ETN+에/JKB] 충분하다.
종결어미구(EFX)	진행하고 있[습니다/EF].
문장부호구(SYX)	벌써[.../SE+.../SE] 끝나다니[!/SF]

로 표현하면 아래와 같다.

문장성분 = [내용어말덩이][기능어말덩이]*

문장 단위의 예시를 들면 다음과 같다. '아기도 수영을 할 수 있게 된다'라는 문장에 구묶음을 적용하면 표 2와 같은 결과를 얻을 수 있다.

같은 문장에 대해 구묶음을 적용하여 구문을 분석하게 되면 그림 3과 같이 의미를 훼손하지 않으면서도 한국어의 특성에 맞게 문장성분 단위로 분석할 수 있게 된다.

표 2 구묶음을 적용한 예시
Table 2 An example of Korean chunking.

Content chunk	Function chunk
(**NX** 아기)	(**JUX** 도)
(**NX** 수영)	(**JKX** 을)
(**PX** 하)	(**PUX** ㄹ 수 있게 되) (**EFX** ㄴ다)

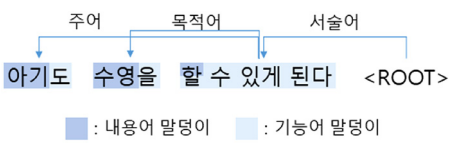

그림 3 구묶음을 적용한 의존구조 분석 예시
Fig. 3 An example of Korean dependency parsing using chunking

3. 한국어 구묶음

본 장에서는 먼저 한국어 말뭉치 구축에 대해서 간단히 기술하고 순차 표지 부착(sequence labeling)을 활용한 한국어 구묶음 모델을 제안한다.

3.1 한국어 말덩이 부착 말뭉치

한국어 말덩이 부착 말뭉치[3]는 [12]에서와 마찬가지로 세종 형태 분석 말뭉치를 이용하여 [7]에서 정의한 기준을 토대로 구묶음을 수행한 말뭉치이다. 말뭉치를 구축할 때 [13]의 언어 정보 부착 시스템을 활용하여 반자동 형식으로 수행한다. 구축한 말뭉치는 CoNLL 형식으로 정리하며 그림 4와 같다.

그림 4에서 'sent_id'는 문장의 순번을 나태내고 'text'는 원문을 나타낸다. 각 열은 좌측부터 차례로 순번, 형태소, 품사, 띄어쓰기 여부, 말덩이 표지를 나타낸다.

구축한 말뭉치의 표지별 개수는 표 3과 같으며 정량적인 정보는 표 4와 같다.

```
# sent_id = sjdc-0544_6-18
# text = 꽃순이의 말이었다.
1	꽃순이	NNP	0	B-NX
2	의	JKG	1	B-JMX
3	말	NNG	0	B-CX
4	이	VCP	0	I-CX
5	었	EP	0	B-EPX
6	다	EF	0	B-EFX
7	.	SF	0	B-SYX
```

그림 4 한국어 구묶음 말뭉치 예시
Fig. 4 An example of the Korean chunking corpus

표 3 말덩이 표지별 개수
Table 3 The number of chunks

Label	#	Label	#	Label	#
NX	37,708	JKX	22,656	PUX	6,114
PX	26,987	JMX	3,826	EPX	9,717
CX	3,057	JUX	7,935	ECX	9,733
AX	9,128	JCX	714	ETX	7,220
MX	2,065	JVX	11	EFX	13,145
IX	247	NAX	7	SYX	15,245

3.2 Bi-LSTM/CRFs를 이용한 순차 표지 부착

영어권에서는 구묶음 문제에 변형기반학습, 기억기반, SVM 등의 방법이 이용되어 왔다[14,15]. 본 논문에서는 심층학습 모델을 이용하여 순차 표지 부착을 통해 한국어 구묶음을 수행하며, 제안된 모델은 [12]에서 사용한 Bi-LSTM/CRFs 모델의 전체적인 구조는 그대로 사용하지만 입력 표상 및 최적화기, 활성함수 등을 부분적으로 개선한 모델이다. 이는 기존의 순환 신경망에서 장기 의존성을 보완하기 위해 고안된 장단기 기억(LSTM,

3) https://github.com/kmounlp/Chunking

표 4 한국어 말덩이 부착 말뭉치 통계
Table 4 The statistics of the Korean chunk corpus

# of senteces	13,113
# of sentence constituents	79,192
# of chunks	content: 79,192 funcion: 96,323 total: 175,515
# of morphemes	204,322
Avg. # of sentence constituent per sentence	6.04
Avg. # of chunks per sentence	13.38
Avg. # of morphemes per sentence	15.58
Avg. # of chunks per sentence constituent	2.22
Avg. # of morphemes per sentence constituent	2.58

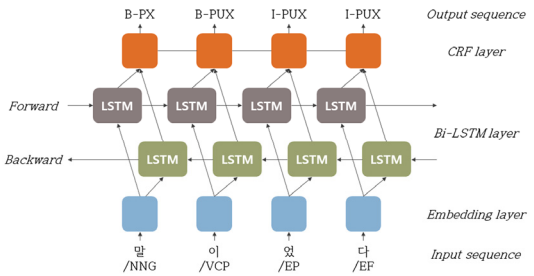

그림 5 한국어 구묶음 모델의 구조
Fig. 5 The structure of the proposed chunking model

Long-Short Term Memory) 계층을 양방향(bidirectional)으로 이용하고, 출력 계층에서 CRFs(Conditional Random Fields)를 이용하여 가장 적합한 말덩이 표지를 출력하는 방식이다. 본 논문에서 이용한 구묶음 모델의 구조는 그림 5와 같다.

먼저 입력열(Input sequence)이 들어오면 단어 표상층(Embedding layer)에서 각 입력 단위들을 벡터로 표현한다. 이후 양방향 LSTM 층(Bi-LSTM layer)을 통해 각 벡터들 간의 문맥을 반영하여 입력 단위에 대한 표지를 예측한다. CRF 층(CRF layer)에서는 예측한 표지 사이의 의존성을 고려하여 여러 개의 예측 표지열 후보 중 가장 높은 점수를 갖는 구묶음 표지열(Output sequence)을 출력하게 된다.

그림 5에서 구묶음의 결과는 IOB 표기법[15]을 이용하되 모든 형태소들은 하나의 말덩이에 속하게 되므로 O 표지는 사용하지 않는다. 말덩이의 종류는 내용어, 기능어 말덩이를 포함하여 총 17개이고[7], 각 말덩이에 속해있는 형태소들에 B, I를 포함한 구묶음 표지를 부여하여 구묶음을 수행하였다. 이때, 세종 말뭉치[1]에 분석 불능범주를 예외처리하기 위해 B-NAX태그를 추가로

표 5 한국어 구묶음의 출력 예시
Table 5 An example of outputs of Korean chunking

Input	'꽃순이/NNP', '의/JKG', '말/NNG', '이/VCP', '었/EP', '다/EF', './SF'
Pred.	'B-NX', 'B-JMX', 'B-CX', 'I-CX', 'B-EPX', 'B-EFX', 'B-SYX'

사용했다. 구묶음 모델의 결과는 표 5와 같이 얻을 수 있다. 표 5에서 품사와 말덩이의 표지는 각각 [1]과 [7]을 따른다. 표 5에서 첫째 줄(Input)은 구묶음의 입력으로 형태소와 품사의 열로 구성되며, 둘째 줄(Pred., prediction)은 각 입력에 대응하는 말덩이 표지이다.

4. 성능 평가 및 오류 분석

실험은 Bi-LSTM/CRFs를 이용한 구묶음 모델의 입력 표상을 달리하여 성능을 측정하고 그에 따른 오류를 분석하는 방식으로 진행했다.

4.1 실험 환경

4.1.1 실험 말뭉치

실험에 사용한 말뭉치는 3.2절의 말덩이 부착 말뭉치이며 문장 및 형태소의 개수는 표 6과 같다. 표 6에서 보는 바와 같이 전체 말뭉치는 약 1.3만 여 문장이며, 일반적인 학습 말뭉치와 같이 세 부분(학습(Train), 개발(Dev), 실험(Test))으로 나누어서 실험에 사용하였다.

표 6 실험에 사용한 문장 및 형태소 개수(단위: 개)
Table 6 The statistics of the Korean chunking corpus

	# of sentences	# of morphemes[4)]
Train	10,490	163,660
Dev	1,312	20,444
Test	1,311	20,218
Total	13,113	204,322

4.1.2 입력표상

[12]의 모델에서는 빈도수에 따라 정수로 표현된 단어들을 케라스(Keras) 표상층을 이용하여 단어 표상을 생성하였다. 본 논문에서는 추가적으로 Word2Vec[16], GloVe[17], fastText[18]를 입력표상으로 이용하였다. 각각의 입력표상은 세종 말뭉치를 이용해 학습했으며, 각 모델은 기본 매개변수들을 이용하나, 비교를 위해 입력 표상의 차원은 300차원으로 같게 하였다.

4.1.3 모델의 매개변수

실험의 편의를 위해서 Bi-LSTM/CRFs 모델의 기본적인 매개변수를 표 7과 같이 설정한다.

4) [12]에서와 달리 문장 길이에 제한을 두지 않고 가장 긴 문장의 길이를 기준으로 실험에 사용했다.

표 7 구묶음 모델에 사용된 매개변수
Table 7 Hyper-parameters of the chunking model

Hyperparameters	Value
batch size	32
embedding dimension	300
LSTM cell units	100
recurrent dropout	0.1
activation	ELU
optimizer	Adam

4.2 성능 평가

모델의 입력은 형태소(morpheme), 형태소/품사(morpheme/pos)의 두 가지 형태를 가지고 실험했다. 모델의 성능은 CoNLL-2000 shared task에서와 마찬가지로 F1-점수를 이용하여 성능을 측정했으며, MUC[19]와 SemEval[20]에서 구묶음과 유사한 과제인 개체명 인식 시스템을 평가할 때 사용한 방법을 이용하였다. 본 논문에서는 이러한 평가 방법 중 구묶음 경계와 표지를 모두 제대로 예측한 것을 기준으로 하였다.

4.2.1 입력 자질 및 표상에 따른 성능 평가

입력 자질 및 표상에 따른 구묶음 모델의 성능은 표 8과 같다.

표 8에서 볼 수 있듯이 형태소(morpheme)와 품사(POS)를 모두 입력하고 입력 표상으로는 fastText를 이용하였을 때 가장 성능이 높게 측정되었다.

표 8 입력 자질 및 표상에 따른 모델의 성능(%)
Table 8 The performance of chunking model according to input features and representations (%)

Word Embedding	Morpheme	Morpheme/POS
Basic[12]	96.20	97.02
Word2Vec	95.66	96.93
GloVe	96.15	97.36
fastText	95.71	**98.34**

4.2.2 한국어 구묶음의 성능 평가

가장 높은 성능을 보인 입력 표상 방법인 fastText를 이용하여 형태소와 품사 표지를 모두 입력으로 사용한 경우를 기준으로 하여 매개변수에 따른 모델의 성능 변화를 살펴보면 표 9와 같다. 조절한 매개변수는 LSTM 셀의 개수(LSTM units), 활성함수(activation), 최적화기(optimizer)이다. 각 성능은 3번씩 측정하여 오차 범위를 나타내었다.

LSTM 셀의 개수를 150개로 하고 활성함수로 ReLU, 최적화기로 Nadam을 사용하였을 때 평균적으로 성능이 가장 높게 측정되었다. 이 중 가장 높은 성능을 낸 모델을 기준으로 앞서 언급한 MUC 및 SemEval에서 사용한 평가 방법을 통해 경계와 표지(strict)를 모두 맞춘 경우 뿐만 아니라 경계만 맞춘 경우(exact), 경계가 겹치는 경우(partial), 표지가 일치하는 경우(type) 등에 대해서 정밀도(precision)와 재현율(recall), F1-점수(F1-score)[26]를 측정한 결과는 표 10과 같다.

표 9 사용자 매개변수에 따른 모델의 성능 변화(%)
Table 9 The model performances according to hyper-parameters (%)

LSTM units	100		150	
activation / optimizer	ReLU[21]	ELU[22]	ReLU	ELU
RMSprop[23]	98.47 ±0.05	98.28 ±0.08	98.34 ±0.05	98.25 ±0.13
Adam[24]	98.15 ±0.11	98.25 ±0.09	98.12 ±0.13	98.23 ±0.04
Nadam[25]	98.32 ±0.06	98.39 ±0.06	**98.49 ±0.04**	98.44 ±0.07

표 10 평가 방식에 따른 모델의 성능(%)
Table 10 The performance of the model according to the evaluation methods (%)

	Strict	Exact	Partial	Type
Precision	98.56	98.68	98.88	98.77
Recall	98.54	98.66	98.85	98.75
F1-score	**98.55**	98.67	98.87	98.76

실험 결과, 경계와 표지를 모두 맞춘 경우의 F1-점수를 기준으로 했을 때 [12]의 모델은 97.02%, 본 논문에서의 모델은 98.54%로 성능이 1.52%p 상승하였다. 이는 부분 단어 정보를 반영할 수 있는 단어 표상인 FastText를 활용해 형태소 단위에서 미등록어 문제가 완화되어 나타난 결과로 보인다. 또한 심층학습의 특성상 매개변수 조절로 인해 모델의 최적화가 이루어져 더욱 구묶음 표지를 잘 예측한 것으로 보인다.

4.3 오류 분석

표 11은 [12]에서 실험한 기본적인 모델의 구묶음 결과에 대한 혼동행렬(confusion matrix)을 간략히 나타낸 것이다. 표 12는 실험에서 가장 높은 성능을 낸 모델의 혼동행렬을 나타낸 것이다. 각 표의 첫 번째 열(true)은 정답 표지를 나타내며, 첫 번째 행(pred)은 모델이 예측한 표지를 나타낸다. 대각선 행렬은 실제 정답과 모델이 예측한 바가 일치하는 경우를 뜻한다.

단어 표상을 바꾼 것만으로 표 11에서 보이던 본용언구 말덩이(PX)를 부사구 말덩이(AX)나 보조용언구 말덩이(PUX)로 잘못 예측한 경우가 표 12에서는 대폭 줄어든 것을 볼 수 있다. 표 13은 [12]에서 본용언구 말덩이를 부사구 말덩이로 잘못 예측하던 것을 본 논문에서 실험한 모델에서는 제대로 예측한 경우를 보인 것이다.

표 11 [12]의 구묶음 모델에 대한 혼동행렬

Table 11 The confusion matrix for the basic chunking model [12]

true \ pred	B-AX	I-AX	B-CX	I-CX	B-ECX	I-ECX	B-EFX	B-EPX	I-EPX	B-ETX	I-ETX	⋯	B-NX	I-NX	B-PUX	I-PUX	B-PX	I-PX	B-SYX	I-SYX
B-AX	872	0	1	0	0	0	0	0	0	0	0		17	1	0	2	0	0	0	0
I-AX	0	63	0	0	2	0	0	0	0	0	0		0	0	1	1	1	0	0	0
B-CX	0	0	268	7	0	0	0	0	0	0	0		2	0	0	7	0	0	0	0
I-CX	0	0	2	318	0	0	0	0	0	0	0		0	1	0	8	0	0	0	0
B-ECX	5	22	0	0	879	0	0	0	0	13	1		1	2	39	2	0	0	0	0
I-ECX	1	3	0	0	1	140	0	0	0	0	13		6	0	0	14	1	0	0	0
B-EFX	0	0	0	0	0	0	1312	0	0	0	0		0	0	0	0	0	0	0	0
B-EPX	0	0	0	0	0	0	0	956	0	0	0		0	0	0	0	0	0	0	0
I-EPX	0	0	0	0	0	0	0	0	1	0	0		0	0	0	0	0	0	0	0
B-ETX	2	0	0	0	4	0	0	0	0	716	0		0	0	13	5	0	0	0	0
I-ETX	0	0	0	0	0	1	0	0	0	1	111		3	1	0	19	0	0	0	0
⋯																				
B-NX	58	1	4	1	0	5	0	0	0	0	1		3663	48	0	0	1	0	0	0
I-NX	1	14	0	0	0	0	0	0	0	0	0		37	760	0	0	1	1	0	0
B-PUX	0	0	3	0	6	0	0	0	0	8	0		0	0	548	8	1	0	0	0
I-PUX	1	0	6	9	0	1	0	0	0	0	4		1	0	2	907	8	1	0	0
B-PX	26	4	0	1	0	0	0	0	0	0	1		2	0	0	43	2644	0	1	0
I-PX	1	2	0	0	0	0	0	0	0	0	1		0	0	0	4	3	197	0	0
B-SYX	0	1	1	0	0	1	0	1	0	0	0		7	2	0	0	0	0	1528	0
I-SYX	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	8

표 12 가장 높은 성능을 낸 구묶음 모델의 혼동행렬

Table 12 The confusion matrix for the chunking model with the highest F1-score

true \ pred	B-AX	I-AX	B-CX	I-CX	B-ECX	I-ECX	B-EFX	B-EPX	I-EPX	B-ETX	I-ETX	⋯	B-NX	I-NX	B-PUX	I-PUX	B-PX	I-PX	B-SYX	I-SYX
B-AX	873	0	0	0	0	0	0	0	0	0	0		13	1	0	0	8	0	0	0
I-AX	0	56	0	0	7	2	0	0	0	0	0		0	0	1	1	0	0	0	0
B-CX	0	0	274	3	0	0	0	0	0	0	0		1	0	1	5	0	0	0	0
I-CX	0	0	1	320	0	0	0	0	0	0	0		0	0	0	8	0	0	0	0
B-ECX	0	0	0	0	949	1	0	0	0	4	0		0	0	10	1	0	0	0	0
I-ECX	0	0	0	0	1	170	0	0	0	0	8		0	0	0	4	1	0	0	0
B-EFX	0	0	0	0	0	0	1312	0	0	0	0		0	0	0	0	0	0	0	0
B-EPX	0	0	0	0	0	0	0	957	0	0	0		0	0	0	0	0	0	0	0
I-EPX	0	0	0	0	0	0	0	0	1	0	0		0	0	0	0	0	0	0	0
B-ETX	0	0	0	0	6	0	0	0	0	723	0		0	0	7	5	0	0	0	0
I-ETX	0	0	0	0	0	7	0	0	0	1	114		3	1	0	9	1	1	0	0
⋯																				
B-NX	5	0	2	1	0	2	0	0	0	0	1		3751	27	0	0	2	0	0	0
I-NX	0	1	0	0	0	0	0	0	0	0	0		18	795	0	0	1	1	1	0
B-PUX	0	0	2	0	10	0	0	0	0	4	0		0	0	553	5	0	0	0	0
I-PUX	0	0	4	6	1	5	0	0	0	2	5		1	0	2	901	12	1	0	0
B-PX	0	0	0	1	0	1	0	0	0	0	0		0	0	0	15	2708	1	0	0
I-PX	0	0	0	0	0	1	0	0	0	0	0		0	0	0	2	1	207	0	0
B-SYX	0	0	1	0	0	0	0	0	0	0	0		3	2	0	0	1	0	1535	0
I-SYX	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	8

표 13 본용언구 말덩이를 제대로 예측한 경우

Table 13 An example of a proper prediction of a predicate phrase chunk (PX)

Input	'혼란/NNG', '을/JKO', **'틈타/VV', '아/EC'**, '그/NP', '가/JKS'
Basic [12]	'B-NX', 'B-JKX', **'B-AX', 'I-AX'**, 'B-NX', 'B-JKX'
Ours	'B-NX', 'B-JKX', **'B-PX', 'B-ECX'**, 'B-NX', 'B-JKX'

표 14 체언구 말덩이의 경계 여부를 잘 예측한 경우

Table 14 An example of a proper prediction of the boundary of noun phrase chunks (NX)

Input	**'내일/NNG', '책방/NNG'**, '에/JKB', '난로/NNG', '를/JKO', '피우/VV', '자/EF', './SF'
Basic [12]	**'B-NX', 'I-NX'**, 'B-JKX', 'B-NX', 'B-JKX', 'B-PX', 'B-EFX', 'B-SYX'
Ours	**'B-NX', 'B-NX'**, 'B-JKX', 'B-NX', 'B-JKX', 'B-PX', 'B-EFX', 'B-SYX'

또한 체언구 말덩이(NX)를 부사구 말덩이(AX)로 잘못 예측하는 오류도 개선 되었으며, 체언이 연속될 때 말덩이의 경계 여부(B-NX, I-NX)를 잘 예측하게 된 것도 볼 수 있었다. 표 14의 예문을 보면, '내일 책방'은 의미적으로 하나의 대상을 지칭하는 것이 아니므로 한 말덩이로 묶이지 않아야 하며, 이러한 부분이 이번 실험에서 개선되었다.

또한 연결어미구 말덩이(ECX)를 전성어미구(ETX) 또는 보조용언구 말덩이(PUX)로 잘못 예측하거나, 전성어미구 말덩이(ETX)를 보조용언구 말덩이(PUX)로 잘못 예측하는 경우도 상당수 완화되었다.

하지만 전성어미구 말덩이(ETX)를 연결어미구 말덩이(ECX)로 잘못 예측하는 경우는 여전히 문제가 되었다. 이 중 표 15의 예문처럼 '데, 만큼, 등'과 같은 의존명사를 포함하여 오류를 범하는 경우가 있었다.

반면, 같은 유형의 오류라도 표 16의 예문과 같이 본 논문에서 제안한 모델이 조금 더 말덩이의 의도에 맞게 연결어미구 말덩이(ECX) 표지로 예측한 경우도 있었다.

또한, 보조용언구 말덩이(PUX)를 연결어미구 말덩이(ECX)나 본용언구 말덩이(PX)로 오인하는 경우도 문제가 되었다. 전자의 경우 말덩이의 정의에 의해 보조용언으로 분류한 용언을 모델이 예측할 때 입력 품사의 표지

표 15 전성어미구 말덩이를 연결어미구 말덩이로 잘못 예측한 경우

Table 15 An example of a wrong prediction of a transformative ending chunk (ETX)

Input	'좋/VA', **'은/ETM', '데/NNB', '로/JKB'**, '시집/NNG', '만/JX', '가/VV', '면은/EC', '야/JX'
True	'B-PX', **'B-ETX', 'B-NX', 'B-JKX'**, 'B-NX', 'B-JUX', 'B-PX', 'B-ECX', 'I-ECX'
Ours	'B-PX', **'B-ECX', 'I-ECX', 'I-ECX'**, 'B-NX', 'B-JUX', 'B-PX', 'B-ECX', 'I-ECX'

표 16 연결어미구 말덩이로 제대로 예측한 경우

Table 16 An example of a proper prediction of a connective ending chunk (ECX)

Input	'아내/NNG', '가/JKS', **'되/VV', 'ㄴ/ETM', '이후/NNG'**, '벌써/MAG', '해/NNG', '를/JKO'
True	'B-NX', 'B-JKX', **'B-PX', 'B-ETX', 'B-NX'**, 'B-AX', 'B-NX', 'B-JKX'
Ours	'B-NX', 'B-JKX', **'B-PX', 'B-ECX', 'I-ECX'**, 'B-AX', 'B-NX', 'B-JKX'

표 17 보조용언구 말덩이를 연결어미구 말덩이로 잘못 예측한 경우

Table 17 An examle of mis-estimating an auxiliary predicate chunk (PUX) as a connective ending chunk (ECX)

Input	'기다리/VV', '고/EC', '있/VX', '었/EP', **'는지/EC', '도/JX', '모르/VV'**, 'ㄴ다/EF'
True	'B-PX', 'B-PUX', 'I-PUX', 'B-EPX', **'B-PUX', 'I-PUX', 'I-PUX'**, 'B-EFX'
Ours	'B-PX', 'B-PUX', 'I-PUX', 'B-EPX', **'B-ECX', 'I-ECX', 'B-PX'**, 'B-EFX'

표 18 보조용언구 말덩이를 본용언구 말덩이로 잘못 예측한 경우

Table 18 An examle of mis-estimating an auxiliary predicate chunk (PUX) as a predicate chunk (PX)

Input	'알/VV', '고/EC', '싶/VX', **'어/EC', '하/VV'**, 'ㄹ/ETM', '것/NNB', '이/JKS'
True	'B-PX', 'B-PUX', 'I-PUX', **'I-PUX', 'I-PUX'**, 'B-ETX', 'I-ETX', 'I-ETX'
Ours	'B-PX', 'B-PUX', 'I-PUX', **'B-ECX', 'B-PX'**, 'B-ETX', 'I-ETX', 'I-ETX

지에 과도하게 편향하여 예측하는 경우로 보였다. 후자의 경우 형태소 분석 단계에서 보조용언을 본용언으로 잘못 분석한 오류가 전파된 것으로 분석된다. 표 17은 보조용언구 말덩이를 연결어미구 말덩이로 예측한 경우이며, 표 18은 본용언구 말덩이로 예측한 경우이다.

표 11과 표 12를 비교해 보면 기존의 모델에서 문제가 되었던 점이 대부분 개선되었다. 하지만 본 논문에서 실험한 모델에서는 기존의 모델에서 어느 정도 잘 예측했던 전성어미구와 보조용언구 말덩이를 오히려 연결어미구 말덩이로 잘못 예측하는 등의 오류도 생긴 것을 볼 수 있다. 즉, 반드시 높은 F1-점수를 내는 모델이 모든 면에서 항상 더 나은 결과를 보장한다고 볼 수 없다. 따라서 무조건적으로 성능이 높은 모델을 시스템에 적용하는 것보다 오류 분석을 통해 모델의 특성을 고려하여 이용하는 것이 바람직할 것이다. 또한 성능이 높은 모델에서 추가적인 자질을 학습하는 등 오류를 개선하는 것이 필요하다.

5. 결 론

본 논문에서는 한국어 구묶음의 정의와 의의를 기술하고 Bi-LSTM/CRFs를 이용한 순차 표지 부착 모델을 이용하여 구묶음을 수행하였다. 또한, 구묶음 모델의 입력 표상을 달리하여 성능을 측정하고 결과에 대한 오류를 분석하였다. 실험 결과 입력 표상으로 fastText를 사용하고 활성함수와 최적화기로 ReLU와 Nadam을 사용했을 때 F1-점수가 98.54%로 가장 높은 성능을 보였다. 하지만 실제 구묶음 결과를 살펴보면 성능이 가장 높다고 하여 모든 문제가 전반적으로 개선되는 것이 아니라 특정 표지의 경우 오히려 예측을 제대로 하지 못하는 경우도 있었다. 따라서 오류 분석을 통해 모델의 특징을 고려하여 시스템에 적용해야 할 것이다.

References

[1] CORPUS, Sejong, 21st Century Sejong Project, The National Institute of the Korean Language, 2010. (in Korean)

[2] S. Abney, "Parsing by chunks," *Principle-based parsing*, eds. Berwick, R. Abney, S. and Tenny, C., Kluwer Academic Publishers, 1991.

[3] J. Kim, "A survey on partial parsing methods," *Korea Information Processing Society Review*, Vol. 7, No. 6, pp. 83-96, 2000. (in Korean)

[4] E. Park and D. Ra, "Processing dependent nouns based on chunking for Korean syntactic analysis," *Korean Journal of Cognitive Science*, Vol. 17, No. 2, pp. 119-138, Jun. 2006. (in Korean)

[5] K. Lee and J. Kim, "Implementing Korean partial parser based on rules," *Trans. of the Korean Information Processing Society*, Vol. 10-B, No. 4, pp. 389-396, Feb. 2003. (in Korean)

[6] Y. Namgoong, C. Kim, M. Cheon, H. Park, H. Yoon, M. Choi, J. Kim, and J. Kim, "Building Korean dependency treebanks reflected chunking," *Proc. of the 31th Annual Conference on Human and Cognitive Language Technology*, pp. 133-138, 2019.

(in Korean)
[7] Y. Namgoong and J. Kim, "Definition of Korean chunk tags," KMOU-NLP-2018-002, 2018. (in Korean) https://github.com/kmounlp/Chunking/blob/master/Chunking%20manual_v.3.2.3.pdf
[8] Y. Namgoong, C. Kim, M. Cheon, H. Park, H. Yoon, M. Choi, and J. Kim, "Defining chunks for parsing in Korean," *Proc. of the 30th Annual Conference on Human and Cognitive Language Technology*, pp. 409-412, Oct. 2018. (in Korean)
[9] Y. Namgoong, C. Kim, M. Cheon, H. Park, H. Yoon, M. Choi, J. Kim, and J. Kim, "Korean chunking using Bi-LSTM/CRF," *Proc. of the KIISE Korea Computer Congress*, pp. 631-633, 2019. (in Korean)
[10] K. Noh, C. Kim, M. Choi, H. Yoon, and J. Kim, "LiAS: A linguistic information annotation system for linear structures of language based on incremental expansion of dictionary and machine learning," *Journal of the Korean Society of Marine Engineering*, Vol. 21, No. 7, pp. 580-586, Sep. 2018. (in Korean)
[11] T. Kudoh and Y. Matsumoto, "Chunking with support vector machines," *Proc. of the 2nd meeting of the North American Chapter of Association for Computational Linguistics*, pp. 1-8, 2001.
[12] S. Buchholz, J. Veenstra, and W. Daelemans, "Cascaded grammatical relation assignment," *Proc. of EMNLP/VLC-99'*, pp. 239-246, 1999.
[13] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Proc. of Advances in Neural Information Processing Systems*, Vol. 26, pp. 3111-3119, 2013.
[14] P. Jeffrey, R. Socher, and C. D. Manning, "GloVe: Global Vectors for word representation," *Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532-1543, 2014.
[15] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. of the Association for Computational Linguistics*, Vol. 5, pp. 135-146, 2017.
[16] N. Chinchor and P. Robinson, "Appendix E: MUC-7 Named Entity Task Definition (version 3.5)," *Proc. of the 7th Message Understanding Conference*, https://www.aclweb.org/anthology/M98-1028 (accessed 2019.05.03.) 1998.
[17] I. Segura-Bedmar, P. Martínez, and M. H. Zazo, "SemEval-2013 Task 9 : Extraction of Drug-Drug Interactions from Biomedical Texts (DDIExtraction 2013)," *Proc. of the 2nd Joint Conference on Lexical and Computational Semantics and the 7th International Workshop on Semantic Evaluation*, pp. 341-350, 2013.
[18] V. Nair and G. Hinton, "Rectified linear units improve restricted boltzmann machines, *Proc. of the 27th International Conference on Machine Learning*, pp. 807-814, 2010.
[19] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by Exponential Linear Units (ELUs)," *International Conference on Learning Representations*, 2016.
[20] G. Hinton, N. Srivastava, and K. Swersky, Slides of Neural Networks for Machine Learning - Lecutre 6e. https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf, 2010.
[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Proc. of the 3rd International Conference on Learning Representations*, 2015.
[22] T. Dozat, "Incorporating nesterov momentum into Adam," *Proc. of the 4th International Conference on Learning Representations*, Workshop Track, 2016.
[23] C. J. van Rijsbergen, *Information Retrieval*. Butter-sworth, 1979.

남 궁 영
2015년 고려대학교 컴퓨터정보학과(학사). 2020년 한국해양대학교 컴퓨터공학과(석사). 2020년~현재 한국해양대학교 컴퓨터공학과 박사과정. 관심분야는 자연언어처리, 구묶음, 의존구조 분석

김 창 현
1991년 홍익대학교 전기계산학과(학사) 1993년 한국과학기술원 전산학과(석사) 2001년 한국과학기술원 전산학과(박사수료). 2001년~현재 한국전자통신연구원(책임연구원). 관심분야는 자연언어처리, 기계번역, 대화처리

천 민 아
2014년 한국해양대학교 컴퓨터정보공학과(학사). 2016년 한국해양대학교 컴퓨터공학과(석사). 2016년~현재 한국해양대학교 컴퓨터공학과 박사과정. 관심분야는 자연언어처리, 개체명인식, 문장생성

박 호 민
2017년 한국해양대학교 컴퓨터정보공학과(학사). 2019년 한국해양대학교 컴퓨터공학과(석사). 2019년~현재 한국해양대학교 컴퓨터공학과 박사과정. 관심분야는 자연언어처리, 정보검색, 감정분석

윤 호

2018년 한국해양대학교 컴퓨터정보공학과(학사). 2020년 한국해양대학교 컴퓨터공학과(석사). 2020년~현재 한국해양대학교 컴퓨터공학과 박사과정. 관심분야는 자연언어처리, 형태소분석, 개체명인식, 단어표상

최 민 석

2018년 한국해양대학교 컴퓨터정보공학과(학사). 2020년 한국해양대학교 컴퓨터공학과(석사). 2020년~현재 한국해양대학교 컴퓨터공학과 박사과정. 관심분야는 자연언어처리

김 재 균

2018년 한국해양대학교 컴퓨터정보공학과(학사). 2019년~현재 한국해양대학교 컴퓨터공학과(석사). 관심분야는 자연언어처리, 기계학습, 문장생성

김 재 훈

1986년 계명대학교 전기계산학과(학사) 1988년 한국과학기술원 전산학과(석사) 1988년 한국과학기술원 전산학과(박사) 1988년~1997년 한국전자통신연구원(선임) 2001년~2002년 Information Sciences Institute USC(방문연구원). 2007년~2008년 Beckman Institute UIUC(방문연구원). 1997년~현재 한국해양대학교 컴퓨터공학과(교수). 관심분야는 자연언어처리, 정보검색, 코퍼스언어학, 감정분석