KIPS Trans. Softw. and Data Eng.
Vol.11, No.7 pp.291~298
ISSN: 2287-5905 (Print), ISSN: 2734-0503 (Online)

https://doi.org/10.3745/KTSDE.2022.11.7.291

Exploiting Chunking for Dependency Parsing in Korean

Young Namgoong† · Jae-Hoon Kim††

ABSTRACT

In this paper, we present a method for dependency parsing with chunking in Korean. Dependency parsing is a task of determining a governor of every word in a sentence. In general, we used to determine the syntactic governor in Korean and should transform the syntactic structure into semantic structure for further processing like semantic analysis in natural language processing. There is a notorious problem to determine whether syntactic or semantic governor. For example, the syntactic governor of the word "먹고 (eat)" in the sentence "밥을 먹고 싶다 (would like to eat)" is "싶다 (would like to)", which is an auxiliary verb and therefore can not be a semantic governor. In order to mitigate this somewhat, we propose a Korean dependency parsing after chunking, which is a process of segmenting a sentence into constituents. A constituent is a word or a group of words that function as a single unit within a dependency structure and is called a chunk in this paper. Compared to traditional dependency parsing, there are some advantage of the proposed method: (1) The number of input units in parsing can be reduced and then the parsing speed could be faster. (2) The effectiveness of parsing can be improved by considering the relation between two head words in chunks. Through experiments for Sejong dependency corpus, we have shown that the USA and LAS of the proposed method are 86.48% and 84.56%, respectively and the number of input units is reduced by about 22%p.

Keywords : Dependency Structure, Constituent, Chunking, Parsing

한국어에서 의존 구문분석을 위한 구묶음의 활용

남궁 영† · 김 재 훈††

요 약

본 논문은 한국어에 대해서 구묶음을 수행한 후에 의존구조를 분석하는 방법을 제안한다. 의존구조 분석은 단어의 지배어를 결정하는 과정이다. 지배어를 정할 때, 문법적인 지배어를 정할 것인지 의미적인 지배어를 정할 것인지가 고질적인 문제이다. 일반적으로는 문법적인 지배어를 정하고 있다. 예를 들면 문장 "밥을 먹고 싶다"에서 어절 "먹고"의 지배어로 "싶다"를 정한다. 그러나 "싶다"는 보조용언으로 의미적으로 지배어가 될 수 없다. 이와 같은 방법으로 구문을 분석하면 의미분석을 위해서 또 다른 변환이 있어야 한다. 본 논문에서는 이런 문제를 다소 완화하기 위해서 구묶음을 수행한 후에 구문을 분석하는 방법을 제안한다. 구묶음은 문장을 구성성분 단위로 분할하는 과정이며 구성성분은 내용어 말덩이와 기능어 말덩이로 구성된다. 구묶음을 수행하면 구문 분석의 입력이 되는 문장 성분의 수가 줄어들므로 구문 분석 속도가 개선될 수 있으며, 문장에서 중심어를 중심으로 하나의 말덩이로 묶이므로 말덩이에 대해서만 그 의존 관계를 파악할 수 있어 구문 분석의 효율성을 높일 수 있다. 본 논문은 세종의존말뭉치를 사용해서 성능을 분석했으며 UAS와 LAS가 각각 86.48%와 84.56%였으며 입력의 노드 수도 약 22% 정도 줄일 수 있었다.

키워드 : 의존구조, 구성성분, 구묶음, 구문분석

1. 서 론

구문 분석은 각 단어 간의 통사적인 구조를 분석하여 문장의 구조를 결정하고 이를 통해 문장의 의미적 중의성을 해소하는 것이다. 구문 분석은 문장을 바라보는 관점에 따라 크게 구 구조 분석과 의존구조 분석으로 나뉜다. 구구조 분석은 구 구조 문법으로 통사 구조를 표현하며, 문장의 구성성분을 파악하여 문장의 구조를 분석하는 방법이다. 반면 의존구조 분석은 의존 문법으로 통사 구조를 표현하며, 어절(단어) 간의 의존 관계(지배소-의존소 관계)를 파악함으로써 문장의 구조를 분석하는 방식이다. 최근에는 구구조 분석보다는 의존구조 분석에 대한 연구가 매우 활발히 진행되고 있다[1-3].

한편 한국어는 부분자유어순을 가지고 있다. 즉 구성성분들 사이의 어순은 비교적 자유로우나 구성성분 내에서 단어(형태소)들 사이의 어순은 엄격히 제한된다. 예를 들면 "사과 한 개를 먹을 수 있다"라는 문장에서 문장의 구성성분은 "사과 한 개를"과 "먹을 수 있다"이며 첫 번째 구성성분인 "사과 한 개를"

※ 이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원(NRF-2017M3C4A7068187, 한국어 정보처리 원천 기술 연구 개발)을 받아 수행된 연구임.
※ 본 논문은 2020년 남궁영의 한국해양대학교 컴퓨터공학과 석사논문인 "구묶음을 반영한 한국어 의존구조 분석"을 확장한 것입니다.
† 비 회 원 : 한국해양대학교 컴퓨터공학과 박사과정
†† 종신회원 : 한국해양대학교 컴퓨터공학과 및 해양인공지능융합전공 교수
Manuscript Received : August 4, 2021
First Revision : September 28, 2021
Second Revision : November 19, 2021
Accepted : December 31, 2021
* Corresponding Author : Jae-Hoon Kim(jhoon@kmou.ac.kr)

에서 어순을 변경하여 “개를 한 사과”라고 하면 비문이 된다. 또한 두 번째 구성성분인 “먹을 수 있다”에서도 “수 먹을 있다”로 표현되면 비문이 된다. 즉 구성성분을 찾을 때, 어떤 경우에는 엄격한 어순을 고려해야 한다. 본 논문에서는 한국어의 이와 같은 특성을 고려하여 문장이 주어지면 먼저 구성성분을 분리하고 구성성분을 토대로 구문을 분석하는 방법을 고려한다.

구성성분을 찾는 과정을 구묶음(chunking)이라고 한다. 다시 말해서 구묶음은 문장을 구성성분 단위로 분리하는 과정이다[4-6]. 한국어는 교착어로서 하나의 구성성분이 내용어와 기능어로 구성되며, 본 논문에서는 구성성분을 내용어와 기능어로 분리하며 이들을 말덩이(chunk)라고 한다. 따라서 엄밀히 말하면 구묶음은 말덩이를 분리하는 과정이다. 말덩이란 인간이 한번에 받아들이는 언어의 단위로, 문법적·의미적으로 하나의 기능을 수행하며, 연속성, 비중첩성, 비재귀성이라는 특징을 가진다[5,6].

구묶음을 수행한 후에 구문분석이 수행되면 아래와 같은 몇 가지 이점이 있을 수 있다. 첫째, 구분분석의 입력 단위의 수가 줄어들므로 구문 분석기의 성능이 개선될 수 있다[7]. 둘째, 구성성분은 구문적 중심어와 의미적 중심어가 일치하므로 구분분석의 결과인 구문분석 트리를 의미분석에서 그대로 사용할 수 있다[8]. 본 논문에서는 구묶음을 수행한 후, 의존구조 분석을 제안하고, 이를 기존의 구문분석과 비교하며 이를 통해서 구묶음을 고려한 의존구조 분석의 유용성을 살펴보고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 구묶음과 구문분석에 대한 기존 연구들을 살펴본다. 3장에서는 구묶음을 반영한 의존구조 분석을 기술하고 4장에서 성능을 평가한다. 5장에서는 결론 및 향후 연구에 대해 논한다.

2. 관련 연구

본 장에서는 구묶음과 한국어 말덩이 부착 말뭉치에 대해서 기술하고 심층학습 기반 한국어 의존구조 분석에 대해 간략히 기술할 것이다.

2.1 구묶음

구묶음은 문장을 구성성분으로 분리하는 작업이다[4-6]. 구묶음은 구문 분석의 전처리 단계로 간주하여 부분 구문 분석(partial parsing)이라고도 한다. 이는 문장에서 구문적으로 같은 기능을 수행하는 형태소들을 하나의 말덩이로 묶어 구문 분석의 입력 단위를 줄일 뿐 아니라 분석 결과의 모호성을 조기에 해소하는 등 구문 분석의 문제들을 미리 줄일 수 있다[5,6]. 구묶음은 규칙기반, 통계기반, 기계학습 기반, 심층학습 기반으로 진행되었다. 규칙기반 방법으로는 주로 유한 유한상태 오토마타(finite-state automata)나 특별한 문법으로 말덩이의 패턴을 정의하여 말덩이를 인식한다[5, 9,10]. 이 방법은 1990년대에 주로 이루어졌다. 통계기반 방법은 품사 부착 모델과 같은 방법으로 HMM(Hidden Markov Model) 기반으로 말덩이를 인식한다[11]. 기계학습 기반 방법은 변형 기반 학습(transformation-based learning)[12], 메모리 기반의 학습(memory-based learning)[13-15] 등 다양한 모델을 이용하여 순서적 표지 부착(sequence labeling) 문제로 모형화하였다. 심층학습 기반 방법은 Bi-LSTM/CRF[16]와 같은 다양한 모델을 이용해서 기계학습과 같이 순서적 표지 부착 문제로 모형화하였다.

한국어 구묶음은 영어와 같은 언어와 크게 다르지 않다. 그러나 앞에서 언급했듯이 한국어는 교착어이므로 구성성분이 내용어와 기능어로 구성되어 있다는 점에서 구묶음을 어떻게 정의하느냐는[6] 많은 차이가 있을 수 있다. 한국어의 경우의 기존 연구는 대부분 규칙 기반으로 진행되었다. 최근에 한국어 구묶음 말뭉치가 공개되면서[1] 이 분야에도 활발한 연구가 진행될 것으로 기대된다. 최근에는 구문 분석을 위한 전처리 단계로서 한국어 문장의 모든 구성 성분에 대해 구묶음을 수행하기 위한 말덩이의 기준 및 그 표지를 제시하였다[16,17].

2.2 한국어 말덩이 부착 구축

말덩이란 동시에 받아들이는 언어의 기본 단위로, 문법적으로나 의미적으로 동일한 역할을 수행하는 구를 말한다[5]. 한국어에서 말덩이는 내용어 말덩이와 기능어 말덩이로 나누어진다. 내용어 말덩이는 의미적 중심어이고, 기능어 말덩이는 내용어 말덩이를 문법적으로 보조하는 말덩이이다. 본 논문에서는 세종 말뭉치[18]의 품사표지를 중심으로 총 16개의 말덩이[2]로 정의하였다[16,17]. 이를 토대로 말덩이 부착 말뭉치를 구축하였으며, GitHub를 통해서 공개되었다.

2.3 심층학습 기반 한국어 의존구조 분석

의존구조 분석은 문장에서 지배소와 의존소 간의 의존 관계를 파악하는 과정이다. 한국어 의존구조 분석은 일반적으로 지배소 후위 원칙(head final constraint), 지배소 유일의 원칙(single head constraint), 투영성의 원칙(projective constraint)이 적용된다[19,20]. 의존구조 분석은 크게 전이 기반 분석과 그래프 기반 분석이 있으며, 최근 이들을 심층학습에 적용하여 성능을 개선하였다[20,21]. 심층 전이 기반 방식은 단어 표상(word embedding)과 원거리 의존 관계를 효율적으로 해결하려고 노력하였으며[22-24], 심층 그래프 기반 방식은 주로 biaffine attention 모델을 이용하고 단어 표상층은 ELMo나 BERT, XLNet과 같은 문맥 단어 표상을 이용하였다[25-26]. 또한 포인터 네트워크를 이용한 연구도 활발히 진행되었다[20,27].

1) https://github.com/kmounlp/Chunking/tree/master/Korean_chunking_corpus
2) 내용어 말덩이는 6개의 종류(체언구(NX), 본용언구(PX), 지정사구(CX), 부사구(AX), 관형사구(MX), 독립어구(IX))로 정의하였고, 기능어 말덩이는 11개의 종류(격조사구(JKX), 관형격조사구(JMX), 보조사구(JUX), 접속조사구(JCK), 호격조사구(JVX), 보조용언구(PUX), 선어말어미구(EPX), 연결어미구(ECX), 전성어미구(ETX), 종결어미구(EFX), 문장부호구(SYX), 분석불능구(NAX))로 정의하였다.

3. 구묶음을 반영한 의존구조 분석

본 논문은 구묶음을 수행한 후에 의존구조를 분석하는 방법을 제안한다. 본 장에서는 말덩이 기반 의존구조에 대해서 살펴보고 이를 기반으로 한국어 말덩이 부착 말뭉치 구축 방법에 대해서 살펴본다. 그리고 나서 구축된 말뭉치를 이용한 한국어 의존 구조 분석을 기술한다.

3.1 말덩이 기반 한국어 의존구조

Fig. 1은 “사과 한 개를 먹을 수 있다”라는 문장의 통사적인 의존구조(a)과 말덩이 기반 의존구조(b)와 의미구조(c)를 보인다. 통사적 의존구조는 어절을 기반으로 통사적인 지배어를 결정한다. 이 경우, 문장의 중심어는 “먹다”가 아니라 “있다”가 된다. 사실 “있다”는 어떤 의미를 전달하는 것이 아니라 “먹다”의 의미를 보충한다. 또한 “먹다”의 목적어가 “사과”가 아니라 “개를”이 된다. 이와 같은 구조로 의미분석을 진행할 경우, 별도의 트리 변환 과정이 필요하다[8]. 트리 구조를 변환하는 과정은 복잡한 고차원의 변환 과정이나 구묶음은 선형 구조를 변환하는 문제로서 순서적 표지 부착 문제로 쉽게 해결할 수 있다. 또한 구문을 분석하거나[20], 의미역을 결정할[28] 경우에도 고차원의 자질이 필요하다. 그러나 구묶음을 수행하면 Fig. 1 (b)와 같은 구조가 되므로 “먹다”의 목적어인 “사과”를 자식 노드에서 쉽게 찾을 수 있을 것이다. 더구나 Fig. 1 (c)와 같은 의미구조를 분석할 때도 트리 구조를 변환하지 않고 그대로 수행할 수 있을 것이다. 이와 같은 장점을 고려하여 본 논문에서는 말덩이 기반 의존구조를 분석하는 방법을 제안한다.

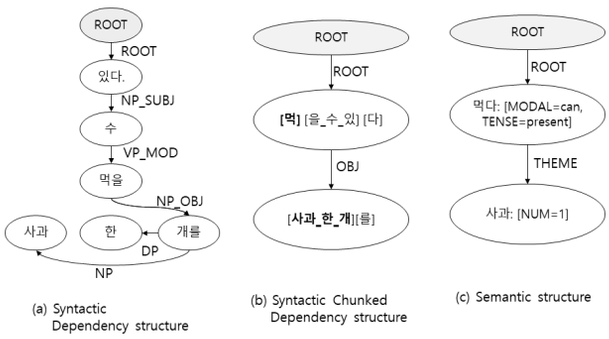

Fig. 1. Syntactic Dependency, Syntactic Chunked Dependency, and Semantic Structure

```
#ORGSENT:프랑스의 세계적인 의상 디자이너 엠마누엘 웅가로가 실내 장식용 직물 디자이너로 나섰다.
ID  FORM        LEMMA       UPOSTAG XPOSTAG             FEATS  HEAD  DEPREL  DEPS  MISC
1   프랑스의    프랑스 의   PROPN   NNP+JKG             _      4     nmod    _     _
2   세계적인    세계 적 이 ㄴ ADJ     NNG+XSN+VCP+ETM     _      4     acl     _     _
3   의상        의상        NOUN    NNG                 _      4     nmod    _     _
4   디자이너    디자이너    NOUN    NNG                 _      6     nmod    _     _
5   엠마누엘    엠마누엘    PROPN   NNP                 _      6     nmod    _     _
6   웅가로가    웅가로 가   PROPN   NNP+JKS             _      11    nsubj   _     _
7   실내        실내        NOUN    NNG                 _      8     nmod    _     _
8   장식용      장식 용     NOUN    NNG+XSN             _      9     nmod    _     _
9   직물        직물        NOUN    NNG                 _      10    nmod    _     _
10  디자이너로  디자이너 로 NOUN    NNG+JKB             _      11    obl     _     _
11  나섰다.     나서 었 다 . VERB    VV+EP+EF+SF         _      0     root    _     _
```

Fig. 2. Korean Dependency Corpus According to CoNLL-U Format

본 논문에서는 Fig. 2와 같은 구조의 말덩이 기반 의존구조 말뭉치를 구축한다. Fig. 3은 말덩이 기반 의존구조 말뭉치의 일부를 보인다. 구성성분은 내용어 말덩이(FORM(conts))와 기능어 말덩이(FORM(func))로 표현되고, 말덩이 표지는 Fig. 3의 일곱 번째 열(CHUNKTAG)에 추가되었으며 나머지는 기존의 말뭉치 Fig. 2와 동일하다.

1) 말뭉치 변환

말뭉치 변환 과정은 4단계(구묶음, 구성성분(FORM) 결정, 지배소(HEAD) 결정, 관계명(DEPREL) 결정)으로 구성된다[32]. 구묶음 단계는 순서적 표지 부착 모델을 이용하여 구묶음을 수행한다[16]. 구성성분 결정 단계에서는 아래와 같은 정규표현에 따라서 구성성분을 결정한다.

구성성분 = [내용어 말덩이][기능어 말덩이]*

즉 하나의 내용어를 중심으로 0개 이상의 기능어 말덩이로 구성된다.

지배소 결정 단계에서는 Table 1과 같은 두 말뭉치의 어절 대응 관계를 만들고 이를 이용해서 변환된 말뭉치의 지배어(HEAD)를 결정한다.

예를 들어, Fig. 2의 1번 어절(‘프랑스의’)의 지배어는 4번 어절(‘디지이너’)이며, Table 1에 의하며 변환된 말뭉치에서는 3번 구성성분에 해당하므로 Fig. 3의 지배어(HEAD) 열의 값이 3이 된다. 이와 같은 방법으로 변환된 말뭉치의 지배어를 결정할 수 있다. Fig. 2의 1번 어절과 같이 변환된 말뭉치에 그대로 하나의 구성성분이 되는 경우는 위의 예와 같이 비교적 단순한 방법으로 결정할 수 있으나 변환된 말뭉치에서 여러 개의 어절이 하나의 구성성분이 될 경우는 좀 복잡하다(Fig. 4 참조).

```
# text = 프랑스의 세계적인 의상 디자이너 엠마누엘 웅가로가 실내 장식용 직물 디자이너로 나섰다.
ID  FORM(conts)                FORM(func)  LEMMA                          UPOSTAG  XPOSTAG                  CHUNKTAG          HEADS  DEPREL
1   프랑스                     의          프랑스 의                      PROPN    NNP+JKG                  NX+JMX            3      nmod
2   세계_적_이                 ㄴ          세계 적 이 ㄴ                  ADJ      NNG+XSN+VCP+ETM          CX+ETX            3      acl
3   의상_디자이너_엠마누엘_웅가로  가      의상 디자이너 엠마누엘 웅가로 가  PROPN    NNG+NNG+NNP+NNP+JKS      NX+JKX            5      nsubj
4   실내_장식_용_직물_디자이너  로          실내 장식 용 직물 디자이너 로  NOUN     NNG+NNG+XSN+NNG+NNG+JKB  NX+JKX            5      obl
5   나서                       었_다_.     나서 었 다 .                   VERB     VV+EP+EF+SF              PX+EPX+EFX+SYX    0      root
```

Fig. 3. An Example of Chunked Dependency Corpus

3) https://universaldependencies.org/format.html

Table 1. Inverse ID Mapping between Original and Transformed Corpus

Original ID	Transformed ID	Original ID	Transformed ID
1	1	7	4
2	2	8	4
3	3	9	4
4	3	10	4
5	3	11	5
6	3		

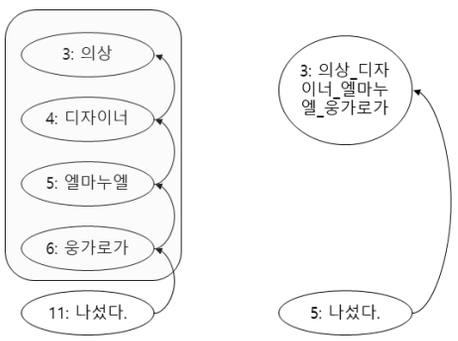

Fig. 4. Determining the Transformed Governor (HEAD)

기존 말뭉치에서 여러 개의 어절이 변환된 말뭉치에서 하나의 구성성분으로 구묶음이 이루어지면 구성성분의 외부에 있는 지배어는 반드시 유일하다. 예를 들면 Fig. 4 (a)에서 보는 바와 같이 “의상”과 “디자이너”와 “엘마누엘”의 지배어는 구성성분 내에 있으나 “웅가로가”의 지배어는 구성성분 밖에 있는 “나섰다”가 지배어가 된다.

관계명 결정 단계는 지배어 결정 단계에서 결정된 의존관계를 그대로 결정된다. 예를 들면 변환된 말뭉치에서 “의상_디자이너_엘마누엘_웅가로가”와 “나섰다”의 의존관계는 기존 말뭉치에서 “웅가로가”와 “나섰다”의 의존관계인 nsubj가 된다.

2) 변환된 의존구조 말뭉치의 특성

본 논문에서 변환된 의존구조 말뭉치의 특성은 Table 2와 같다. 기존의 말뭉치(original dependency corpus)[29]은 총 62,345문장으로 이루어져 있으나 변환된 말뭉치는 총 49,292문장으로 구성되었다. 약 13,000문장은 각종 오류(구묶음 오류, 변환 오류, 형태소 오류 등)가 포함되어 수동으로 제외한다. 정제된 말뭉치(purified dependency corpus)는 변환된 말뭉치에 포함된 문장만을 추출한 기존 말뭉치며 4장에서 실험을 위해서 구축한다. 구문분석의 입력 단위는 기존의 말뭉치의 경우에는 어절 단위이지만 변환된 말뭉치에는 구성성분 단위이다. 기존의 말뭉치와 비교할 때, 변환된 말뭉치의 입력 단위의 수가 현저히 줄어드는 것을 확인할 수 있다.

3.3 구묶음을 반영한 한국어 구문 분석

본 논문에서 구문 분석 모델은 스택-포인터 네트워크

Table 2. The Characteristics of Korean Chunked Dependency Corpus

	Dependency corpus		
	Original	Purified	Chunked
No. of sentences	62,345	49,292	49,292
No. of tags	45	45	18
No. of morphemes	1,566,560	1,054,859	1,054,859
No. of chunks	NA	NA	804,854
Input unit	word	word	constituent
No. of input units	713,238	526,378	377,301
No. of dependency relations	50	50	50

(stack-pointer network)[33]을 이용한다. Fig. 5는 스택-포인터 네트워크 모델의 구조이며, 포인터 네트워크(pointer network)와 내부 스택으로 구성된 인코더(encoder)와 디코더(decoder)로 구성되어 있다. 인코더는 문장의 의미를 하나의 벡터로 표현하고, 디코더는 깊이 우선 방식으로 문장의 지배소($)에서부터 의존구조 트리를 구성하는 하향식 모델이다. 하나의 중심어는 여러 의존소를 가질 수 있으므로 내부 스택에 의존소를 저장하여 하나의 지배소가 여러 번 의존소를 찾을 수 있도록 한다. 포인터 네트워크는 각 단계에서 내부 스택의 최상위 노드의 의존소가 될 자식 노드를 선택한다. 이와 같은 방법을 반복적으로 수행해서 문장을 분석한다.

학습할 때 인코더에서 순환 신경망을 통해 입력 단어에 대한 은닉 표상(s_i)을 생성하고, 내부 스택은 root로 초기화된다. 디코더는 매 단계(t)마다 내부 스택의 최상위 노드의 입력 표상을 받아 디코더의 은닉 표상(h_t)을 생성한다. 주의 집중은 식 (1)과 같이 계산된다[33].

$$e^t = (e_1^t, ..., e_n^t),\ e_i^t = score(h_t, s_i)$$
$$a^t = softmax(e^t) \quad (1)$$

여기서, a^t는 주의 집중 벡터이고, e^t는 주의 집중 점수이다. $score(\cdot)$는 h_t와 s_i간의 주의 집중 정도를 구하는 함수이다. 주의 집중 점수를 통해 의존소가 될 단어의 위치 c를 결정하고, c에 있는 단어 w_c를 지배소 w_h의 의존소가 되고, w_c를 스택에 삽입한다. 만일 $c=h$이면 w_h의 모든 의존소를 찾았음을 의미하고 스택에서 w_h를 제거한다. 이와 같은 과정을 반복해서 내부 스택에 문장의 지배소만 남을 때, 구문분석이 종료된다. 구체적으로 Fig. 5를 통해서 간략히 살펴보면 스택에 문장의 지배소를 시작해서 h_1을 구하고 h_1을 대상으로 $(s_{1,...,}s_4)$과 a^1을 구해서 의존소 위치 c는 4가 되므로 ‘배신하지_않는다’가 스택에 들어가고 다음 단계$(t=2)$의 입력은 $(\varnothing, s_4, s_1)$[4]가 되고 h_2를 구해서 같은 과정을 반복하여 c=3을

4) 스택-포인터 네트워크의 입력은 3개의 원소로 구성되는데 첫 번째 원소는 의조 구조에서 형제 노드(ϕ; 없음)이고 두 번째 원소는 가장 최근에 의존소가 된 노드(s_4)이고 세 번째 원소는 두 번째 원소의 지배소(s_1)가 된다.

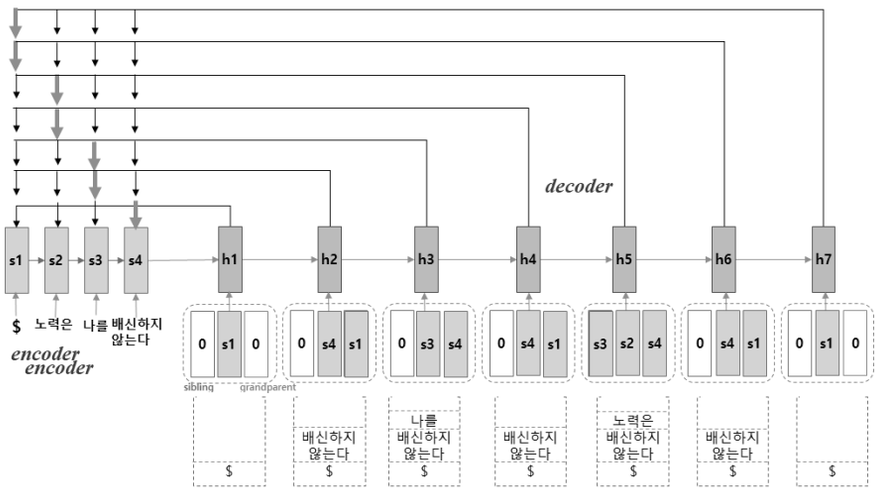

Fig. 5. The Structure of Stack-pointer Network for Korean Dependency Parsing with Chunking

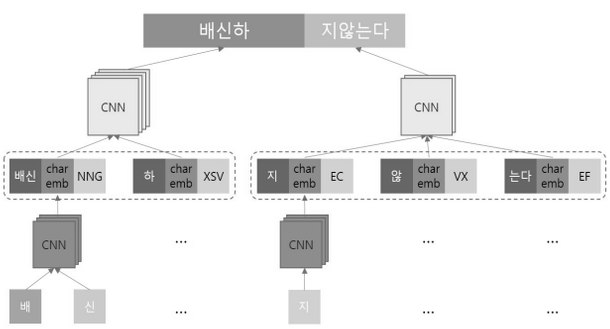

Fig. 6. Constituent Representation of the Constituent "[배신하] [지_않_는다]"

구한다. 다음 단계($t=3$)의 입력은 $(\varnothing, s_3, s_4)$이며, c=3이므로 스택에서 원소를 제거한다(pop). 따라서 다음 단계($t=4$)의 입력은 $(t=2)$와 같이 $(\varnothing, s_4, s_1)$이 되며 c는 1을 구한다. 이와 같은 과정을 반복해서 $t=7$까지 반복해서 구문분석을 종료한다.

앞에서 언급한 바와 같이 구성성분은 내용어 말덩이와 기능어 말덩이로 구성된다. 내용어 말덩이는 의미적 중심어이며, 기능어 말덩이는 내용어 말덩이의 문법적인 역할을 담당한다. 이러한 특징을 살려 구성성분의 표상은 내용어 말덩이와 기능어 말덩이의 표상을 결합하여 사용한다(Fig. 6). 각각의 말덩이는 여러 형태소가 결합되므로 각 형태소의 표상을 합성곱 신경망(Convolutional Neural Network; CNN)에 적용하여 하나의 말덩이의 표상을 구한다. 형태소 표상은 형태소 자체의 표상과 문자 표상과 품사 표상을 결합한다. 문자 표상은 미등록어에도 잘 적응할 수 있도록 각 문자 표상을 CNN에 적용하여 구한다.

4. 실험 및 평가

본 장에서는 입력 단위가 구성성분(구묶음을 적용한 경우)인 경우와 어절(기존의 구문 분석)인 경우의 의존구조 분석을 비교하고 토의한다. 가능하면 객관적인 비교를 위해서 3.3절에서 언급한 같은 신경망 구조인 스택-포인터 네트워크를 사용하고 입력 표상 외에는 같은 구조를 가진다.

Table 3. The Statistics of Chunked Dependency Corpus

Corpus	No. of Sentences	No. of input units
Training	33,313	269,793
Validation	4,165	47,641
Test	4,165	42,073
Total	41,643	359,507

4.1 실험 환경

실험을 위한 말뭉치는 3.2절에서 기술한 정제된 말뭉치(어절 기반 의존구조 말뭉치)와 변환된 말뭉치(말덩이 기반 의존구조 말뭉치)이며, 학습을 위해서 세 개의 말뭉치(학습, 검증, 평가)로 나누어 사용한다(Table 3). Table 3에서 보는 바와 같이 전체 말뭉치의 1/10을 각각 평가말뭉치와 검증말뭉치로 사용하였다.

Table 4는 본 논문에서 사용한 스택-포인터 네트워크의 매개변수이다. 형태소 표상은 세종 말뭉치를 이용하여 GloVe [34]를 이용하였고, 음절과 품사 표상은 정수 표현을 이용해서 매개변수를 초기화하였다. 구성성분의 표상의 크기는 내용어 표상과 기능어 표상을 2:1이 구성되도록 합성곱 신경망의 필터 수를 각각 기존의 2/3, 1/3으로 조절하였다.

4.2 성능 평가

평가 척도는 구문분석에는 널리 사용되는 UAS(Unlabeled Attachment Score)와 LAS(Labeled Attachment Score)를 이용한다[1]. Table 5는 구묶음을 반영한 경우(입력 단위: 구성성분, chunk)와 그렇지 않은 경우(입력 단위: 어절, Word)의 성능을 보이고 있다.

구문분석의 결과(predicted)를 단순히 비교하면 구성성분

Table 4. Hyper-parameters of Stack-pointer Networks

Layer	Hyper-parameter	Value
Embedding	morphemes dimension	300
	characters dimension	50
	part-of-speeches dimension	50
CNN	# of character filters	128
	character windows size	3
	# of eojul filters	300
	eojul wiondows size	3
RNN	RNN Mode	LSTM
	encoder layers	3
	encoder size	512
	decoder layers	2
	decoder size	256
	arc space	512
	type space	128
Dropout	dropout	0.2
Learning	optimizer	Adam
	learning rate	1e-3
	weight decay	1e-5
	gradient clipping	5.0
Dependency	prior order	inside-out

Table 5. The Evaluation Results

Metric	Input unit		
	Chunk	Word	
	Predicted	Estimated from chunk	Predicted
UAS	82.59%	86.48	82.98%
LAS	80.13%	84.56	80.45%

의 경우가 다소 떨어진다. 즉 UAS와 LAS가 각각 0.39%p와 0.32%p가 떨어진다. 이 경우는 입력 단위가 서로 달라서 직접적으로 비교될 수 없다. 본 논문에서 객관적인 비교를 위해서 Fig. 4 (a)에서 보는 바와 같이 구성성분 내에서 어절의 지배소가 있는 경우는 구성성분의 패턴에 따라 어절의 지배소를 명확하게 정할 수 있다. 따라서 본 논문에서는 각 구성성분 내의 어절의 지배소를 결정한 후에 어절 단위로 새로이 평가하면 (estimated) 구성성분을 입력 단위로 했을 경우가 UAS와 LAS에 대해 각각 3.5%p와 4.11%p가 증가됨을 알 수 있다.

구묶음을 반영한 구문 분석의 경우 정확도나 효율성 면에서 기존의 방법보다 나은 결과를 보였으며, 구문적인 관점뿐만 아니라 의미적인 요소도 쉽게 분석할 수 있다. 따라서 한국어 처리에서도 지속적으로 구묶음을 반영한 구문 분석에 대한 연구가 이루어져야 할 것이다. 이를 위해 구묶음 자체에 대한 오류 분석과 구묶음을 반영한 말뭉치의 효용성에 관한 연구도 다각도에서 검증되어야 할 것이다. 또한, 내용어와 기능어의 비중이 구문 분석에 미치는 영향에 관한 연구도 흥미 있는 주제로 남아있다.

4.3 비교 분석

Table 6은 기존의 구묶음에 대한 연구를 비교하고 분석한 표이며 대부분의 연구들은 규칙 기반(rule-based)으로 진행되었다. 그 이유는 구묶음 말뭉치가 공개되지 않았으며 구묶음에 대한 명확한 정의가 부족했던 것으로 판단된다. 최근 기계학습 및 심층학습에 대한 연구가 활발이 진행되면서 구묶음에 대한 필요성이 대두되었고 본 연구에 의해서 공개되어 이 분야에 대한 연구도 활발해 질 것으로 기대된다.

Table 6. Comparison with Existing Dependency Parsers

Ref.	LAS	UAS	Corpus (No. Sent.)		Model
			Training	Test	
This	84.56	86.48	33,313	4,165	Pointer Networks
[24]	88.56	90.69	-	-	Transition-Based
[27]	89.48	91.79	53,842	5,817	Pointer Networks
[21]	90.42	93.38	60,751	7,308	Self-Attn
[23]	90.75	-	57,265	5,726	Transition-Based, CV
[20]	91.35	92.63	55,686	1,000	Pointer Networks
[25]	91.00	93.12	-	-	BERT+Biaffine Attn.
[26]	92.00	94.06	53,842	5,817	BERT+Biaffine Attn.
[39]	94.17	96.31	-	-	KorBERT(l)+Transformer

※ Ref.: Reference; Attn : Attention;
No. Sent.: The number of Sentences;
CV: Cross-Validataion Evaluation,

Table 7. Comparison with Existing Korean Chunkers

Ref.	Methods	Scope	Remarks
This	Deep Learning(Bi-LSTM-CRF)	All	
[35]	Machine Learning(CRF)	All	
[9]	Rule-based	All	141 rules
[36]	Rule-based(Heuristics)	Noun	
[37]	Rule-based(Dictionary+Heuristics)	Dep. noun	
[38]	Rule-based	Noun, Verb	

※ Ref.: Reference; Bi-LSTM: Bidirectional LSTM;
CRF: Conditional Random Fields; Dep. noun: Depent noun;
All: All phrases

Table 7은 세종 말뭉치를 사용하는 기존의 의존 구조 분석기들을 비교하고 분석한 표이다. 참고로 모두 세종 말뭉치를 사용하더라도 [26]과 [27]을 제외하고는 모두 서로 다른 학습 말뭉치를 사용하므로 객관적인 성능을 비교하는 것은 좀 무리가 있다. 대부분의 논문들이 본 논문에서 제안한 의존 분석기보다는 좋은 성능을 보이고 있다. 본 논문의 목적은 성능을 개선하는 것이 아니라 구묶음을 반영할 의존 구조 분석의 유용성을 살피는 것이며 서론과 4.2절(성능면)에서 보았듯이 그 유용성을 확인할 수 있었다. 의존 구조 분석에 대한 전체적인 경향은 다른 자연어 처리 분야와 마찬가지로 BERT와 같은 사전 학습 언어 모델을 사용하는 경우가 좋은 성능을 보이고 있다. 또한 [23]과 같이 어휘망을 이용해서 기존에 구축된 언어지식을 신경망과 결합하여 성능을 개선할 수 있다.

5. 결 론

본 논문에서는 구묶음을 고려한 한국어 의존구조 분석을 제안하였다. 구묶음은 문장을 구성성분으로 분리하고 과정이며, 하나의 구성성분은 내용어 말덩이와 기능어 말덩이로 구성된다. 일반적으로 한국어 구문분석의 입력 단위는 어절인 반면에 본 논문에서의 입력 단위는 구성성분이 된다. 이와 같이 구성성분을 입력 단위로 하면 입력 단위의 수가 줄어들 뿐 아니라 의미분석을 보다 용이하게 할 수 있을 것이다. 실험을 통해서 UAS와 LAS는 각각 86.48%와 84.56%를 보였다. 이 결과는 어절을 입력할 경우에 비해 UAS와LAS가 각각 약 3.5%p와 4.1%p의 성능 향상을 보였다. 결과적으로 구묶음이 의존구조 분석에 어느 정도 도움이 될 수 있음을 보였다.

향후에 좀 더 다양한 종류의 분석기를 통하여 유용성과 효용성을 보일 필요가 있으며, 한국어 구문 분석에 있어 내용어와 기능어의 표상 비중을 조절하여 성능 변화를 분석할 필요가 있을 것이다. 그리고 구묶음을 고려한 구문 분석의 원활한 수행을 위하여 기반 기술인 부분 구문 분석기의 오류 분석 및 성능 향상이 지속해서 이루어져야 할 것이다. 또한 구묶음을 반영한 말뭉치의 효용성에 관한 연구도 다각도에서 검증되면 한국어 구문 분석에 더욱 이바지하는 바가 클 것으로 생각한다.

References

[1] S. Kübler, R. McDonald, and J. Nivre, "Dependency Parsing," Morgan and Claypool Publishers, 2009.

[2] N. Jian, "A review of graph-based dependency parsing," *Proceedings of the 5th International Conference on Computer*, Automation and Power Electronics, pp.25-30, 2017.

[3] G. Bouma, D. Seddah, and D. Zeman, "From raw text to enhanced universal dependencies: The parsing shared task at IWPT 2021," *Proceedings of the 17th International Conference on Parsing Technologies*, pp.146-157, 2021.

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

[5] S. P. Abney, "Part-of-speech and partial parsing," Corpus-Based Methods in Language and Speech Processing, eds. Young, S and Bloothooft, G., Kluwer Academic Publishers, pp.118-173, 1996.

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

[7] C. Kim, C. Jung, Y. Kim, and Y. Seo, "An efficient Korean syntactic analyzer using partial combination of words," *Proceedings of the 27th KISS Fall Conference*, pp.597-600, 1995 (in Korean).

[8] I. Mel'čuk and J. Milićević, "An Advanced Introduction to Semantics: A Meaning-Text Approach," Cambridge: Cambridge University Press, 2020.

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

[10] G. Grefenstette, "Light parsing as Finite State Filtering," In *Proceedings of the Workshop on Extended Finite State Models of Language*, pp.20-25, 1996.

[11] A. Molina and F. Pla, "Shallow parsing using specialized HMMs," *Journal of Machine Learning Research*, Vol.2, pp.595-613, 2002.

[12] L. Ramshaw and M. Marcus, "Text chunking using transformation-based learning," *arXiv:cmp-lg/9505040*, 1995.

[13] W. Daelemans, S. Buchholz, and J. Veenstra, J. "Memory-based shallow parsing," *Proceedings of the Conference on Computational Natural Language Learning*, pp.53-60, 1999.

[14] Y. Hwang, H. Chung, S. Park, Y. Kwak, and H. Rim, "Improving the performance of Korean text chunking by machine learning approaches based on feature set selection," *Journal of KISS: Software and Applications*, Vol.29, No.9/10, pp.654-668, 2002 (in Korean).

[15] S. Park and B. Zhang, "A hybrid of rule based method and memory based learning for Korean text chunking," *Journal of KISS: Software and Applications*, Vol.31, No.3, pp.369-378, 2004 (in Korean).

[16] Y. Namgoong, C. Kim, M. Cheon, H. Park, H. Yoon, M. Choi, J. Kim, and J. Kim, "Defining chunks and chunking using its corpus and Bi-LSTM/CRFs in Korean," *Journal of KIISE*, Vol.47, No.6, pp.587-595, 2020 (in Korean).

[17] J. Kim, "Basic Units and their Tags for Korean Partial Parsing," Korea Maritime and Ocean University, Department of Computer Engineering, Technical Report KMU-NLP-TR-2000-006, 2000 (in Korean).

[18] The National Institute of the Korean Language, The 21 Century Sejong Plan, 2012 (in Korean).

[19] Y. Park and J. Seo, "Correction method for Korean dependency parsing using projectivity and re-searching," *Korean Journal of Cognitive Science*, Vol.22, No.4, pp.419-447, 2011 (in Korean).

[20] Y. Choi and K. Lee, "Korean dependency parser using higher-order features and stack-pointer networks," *Journal of KIISE*, Vol.46, No.7, pp.636-643, 2019 (in Korean).

[21] J. Lim and H. Kim, "Korean dependency parsing using the self-attention head recognition model," *Journal of KIISE*, Vol.46, No.1, pp.22-30, 2019 (in Korean).

[22] J. Li and J. Lee, "Korean transition-based dependency parsing with recurrent neural network," *KIISE Transactions on Computing Practices*, Vol.21, No.8, pp.567-571, 2015 (in Korean).

[23] C. Jeong, J. Shin, J. Lee, and C. Ock, "Transition-based Korean dependency analysis system using semantic abstraction," *Journal of KIISE*, Vol.46, No.11, pp.1174-1185, 2019 (in Korean)

[24] S. Na, K. Kim, and Y. Kim, "Stack LSTMs for transition-based Korean dependency parsing," *Proceedings of Korea Computer Congress*, pp.732-734, 2016.

[25] S. Hong, S. Na, J. Shin, and Y. Kim, "BERT and ELMo for contextualized word embeddings in Korean Dependency Parsing," *Proceedings of Korea Computer Congress*, pp.491-493, 2019 (in Korean).

[26] C. Park, C. Lee, J. Lim, and H. Kim, "Korean dependency parsing with BERT," *Proceedings of Korea Computer Congress*, pp.530-532, 2019 (in Korean).

[27] C. Park and C. Lee, "Korean dependency parsing using pointer networks," *Journal of KIISE*, Vol.44, No.8, pp.822-831, 2017 (in Korean).

[28] K. Park and Y. Mun, "Two-phase shallow semantic parsing based on partial syntactic parsing," *The Transaction of the Korean Information Processing Society*, Vol.17-B, No.1, pp.85-92, 2010 (in Korean).

[29] Y. Choi and K. Lee, "Head-percolation rules of constituent-to-dependency conversion in Korean," *Proceedings of the 30th Annual Conference on Human and Cognitive Language Technology*, pp.514-519, 2018.

[30] H. Won and P. M. Ryu, "Semi-automatic generation of universal dependency corpus from Sejong phrase structured corpus," *Proceedings of KCC*, pp.1430-1432, 2019.

[31] T. Kim, P. Ryu, H. Kim, and H. Oh, "Unified methodology of multiple POS taggers for large-scale Korean linguistic GS set construction," *Journal of KIISE*, Vol.47, No.6, pp.596-602, 2020.

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

[33] X. Ma, Z. Hu, J. Liu, N. Peng, G. Neubig, and E. Hovy, "Stack-pointer networks for dependency parsing," *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp.1403-1414., 2018.

[34] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp.1532-1543, 2014.

[35] K. Jeon, H. Seo, Y. Nam, and J. Kim, "Performance improvement of chunking using cascaded machine learning methods," *Proceedings of the 30th Annual Conference on Human and Cognitive Language Technology*, pp.107-109, 2011 (in Korean).

[36] K. Ahn and Y. Seo, "Chunking of contiguous nouns using noun semantic classes," *The Journal of the Korea Contents Association*, Vol.10, No.3, pp.10-20, 2010 (in Korean).

[37] 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, 2006 (in Korean).

[38] M. Kim, S. Kang, and J. Lee, "Text chunking by rule and lexical information," *Proceedings of the 30th Annual Conference on Human and Cognitive Language Technology*, pp.103-109, 2000 (in Korean).

[39] J. Lim and H. Kim, "Korean dependency parsing using token-level contextual representation in pre-trained language model," *Journal of KIISE*, Vol.48, No.1, pp.27-34, 2021.

남궁 영

https://orcid.org/0000-0001-7405-0498
e-mail : ynamgoong@g.kmou.ac.kr
2015년 고려대학교 컴퓨터정보학과(학사)
2020년 한국해양대학교 컴퓨터공학과(석사)
2020년 ~ 현 재 한국해양대학교 컴퓨터공학과 박사과정
관심분야 : Natural Language Processing, Chunking, Dependency Parsing

김 재 훈

https://orcid.org/0000-0001-8655-2591
e-mail : jhoon@kmou.ac.kr
1986년 계명대학교 전계계산학과(학사)
1988년 한국과학기술원 전산학과(석사)
1996년 한국과학기술원 전산학과(박사)
1988년~1997년 한국전자통신연구원 선임연구원
2001년 ~ 2002년 Information Sciences Institute USC 방문연구원
2007년 ~ 2008년 Beckman Institute UIUC 방문연구원
1997년 ~ 현 재 한국해양대학교 컴퓨터공학과 및 해양인공지능융합전공 교수
관심분야 : Natural Language Processing, Information Retrieval, Corpus Linguistics, Sentiment Analysis