공학석사 학위논문

구묶음을 반영한 한국어 의존구조 분석

Korean Dependency Parsing Reflected Chunking

지도교수 김 재 훈

2020년 2월

한국해양대학교 대학원

컴퓨터공학과

남궁 영

본 논문을 남궁 영의 공학석사 학위논문으로 인준함.

위원장 : 박 휴 찬 (인)

위 원 : 류 길 수 (인)

위 원 : 김 재 훈 (인)

2019 년 12 월 26 일

한국해양대학교 대학원

목 차

List of Tables

List of Figures

Korean dependency parsing reflected chunking

Namgoong, Young

Department of Computer Engineering
Graduate School of Korea Maritime and Ocean University

Abstract

In natural language processing, syntactic parsing is to analyze relationship between sentence components. The parsing can resolve semantic as well as syntactic ambiguity by determining the relationship. On the other hand, in Korean parsing, usually there are a lot of components (or morphemes) in an input sentence, and these can cause high complexity and low accuracy in parsing. To alleviate this problem, we propose Korean parsing reflected chunking. Chunking is to identify constituents called chunks which are a sequence of words (or morphemes) playing a syntactic and semantic role in a given sentence. We can decrease the number of the input components of the parser by chunking. Moreover, chunking groups morphemes with auxiliary meaning like functional or grammatical meaning, so we can just focus on the head word in chunks.

The purpose of this paper is therefore threefold. The first is to define Korean chunks. The second is to build Korean dependency corpus reflected chunking, which is for experiments, according to the chunk definition. The corpus can be automatically converted from the existing Korean dependency corpus. The third is to develop a Korean dependency parser reflected chunking. The parser has been experimentally evaluated in parsing Korean text, achieving UAS and LAS of 86.48% and 84.56% respectively. The parser outperforms the Korean parser which is not reflected chunking by 3.5%p and 4.11%p, and has been shown to be better than the existing one in performance. The parser can also analyze semantic as well as syntactic structure.

In the future, the study on chunking in Korean should be conducted consistently for establishing linguistic concepts. An error analysis on the chunking and parsing is required for performance improvement. Furthermore, the difference in vector representation according to the ratio between content chunks and function chunks in a sentence still remains as an interesting subject.

KEY WORDS: Korean parsing, Dependency parsing, Partial parsing, Korean chunking, Korean chunks

구묶음을 반영한 한국어 의존구조 분석

남궁 영

한국해양대학교 대학원
컴퓨터공학과

초록

자연언어처리에서 구문분석은 문장 구성 성분들의 관계를 파악하는 과정을 말한다. 구문분석을 통해 문장의 구조를 결정함으로써 의미적 중의성을 해소할 수 있다. 한국어 구문분석은 구문분석기의 입력이 되는 문장의 성분 수가 많아 이로 인해 분석의 복잡도가 높고 정확도가 낮은 현상을 보인다. 이에 대한 해결방안으로 본 논문에서는 구묶음을 반영한 한국어 구문분석을 제안한다. 구묶음은 형태소분석된 문장에 대해 문법적, 의미적으로 하나의 역할을 하는 연속된 형태소들을 하나의 말덩이로 묶는 작업을 말한다. 구묶음을 수행하면 구문분석의 입력이 되는 문장 성분의 수가 줄어들며, 문장 내에서 보조적인 역할을 하는 요소들이 하나의 말덩이로 묶이므로 말덩이 내의 중심어에 대해서만 의존 관계를 파악할 수 있어 구문분석의 효율성이 증진된다.

따라서 본 논문에서는 구묶음을 반영한 구문분석을 수행하기 위해 한국어에 대해 구묶음과 말덩이를 정의하고 이에 기반하여 구묶음을 수행한다. 또한, 구묶음 수행 결과를 바탕으로 기존의 한국어 의존구조 말뭉치로부터 구묶음을 반영한 의존구조 말뭉치를 구축한다. 이러한 작업을 기반으로 하여 궁극적으로 구묶음을 반영한 구문분석과 기존의 구문분석을 비교하고 분석함으로써 한국어처리에 있어 구묶음의 유효성과 필요성을 보이는 데 그 의의가 있다.

실험 결과 어절 단위로 정확도를 측정했을 때, 구묶음을 반영한 경우는 UAS 기준 86.48%, LAS 기준 84.56% 였으며, 기존 방식의 경우 UAS 기준 82.98%, LAS 기준 80.45%로, 구묶음을 반영한 경우가 각각 3.5%p, 4.11%p 상승한 결과를 보였다.

구묶음을 반영한 구문분석은 정확도나 효율성 면에서 기존의 방법보다 나은 결과를 보였으며, 구문적인 관점뿐만 아니라 의미적인 요소도 함께 분석할 수 있는 방법이다. 따라서 한국어처리에서도 지속적으로 구묶음을 반영한 구문분석에 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 구문분석에 미치는 영향에 관한 연구도 흥미 있는 주제로 남아있다.

KEY WORDS: 구문분석, 의존구조 분석, 부분 구문분석, 구묶음, 말덩이

제 1 장 서 론

한국어처리의 분석 단계는 크게 형태소분석 및 구문분석 등으로 이루어진다. 형태소분석 단계에서는 원시 말뭉치를 의미 있는 가장 작은 말의 단위인 형태소로 분석하고 각각에 해당하는 품사 정보를 부착한다. 구문분석 단계에서는 형태소분석의 결과를 바탕으로 문장 구성 성분들의 관계를 파악하게 된다. 이를 통해 문장의 구조를 결정함으로써 의미적 중의성을 해소할 수 있다.

자연언어처리에서 구문분석은 문장을 바라보는 관점에 따라 크게 구 구조 분석(constituency parsing)과 의존구조 분석(dependency parsing)으로 나뉜다. 구 구조 분석은 문장을 구성 성분들의 결합으로 분석하는 방법이다. 반면 의존구조 분석은 문장 구성 성분 간의 지배소(head)와 의존소(modifier)의 관계를 파악함으로써 문장의 구조를 분석하는 방법이다. 따라서 문장을 구성하는 요소의 위치에 제약이 적고 생략에도 유연하게 대처할 수 있어 한국어 구문분석에 적합하다.

하지만 의존구조 분석은 문장에서 의존 관계를 결정해야 할 노드 수가 많을수록 구문분석기의 계산 복잡도가 높아지고 분석에 있어 모호성이 증가한다. 특히 한국어 의존구조 분석에서는 지배소를 결정할 때 방향성 문제가 발생하는데, 한국어 의존구조 분석의 원칙(나동렬, 1994) 중 하나인 지배소 후위 원칙을 각 노드에 엄격하게 적용할 경우, 다음과 같이 구문적 중심어와 의미적 중심어가 불일치하는 문제가 발생한다.

Figure 1.1 Examples of dependency parsing.

Figure 1.1 (a)에서 서술어 '먹었다'의 목적어는 의미적으로 '사과'이지만, 기존의 방식대로 의존구조를 분석하면 '개를'이 '먹었다'의 의존소가 된다. Figure 1.1 (b)의 예문 역시 문장 전체의 의미적 중심어는 용언 '하다'를 어간으로 갖는 '할'이지만, 기존의 방식대로 구조를 분석하면 'ROOT'를 지배소로 갖는 노드는 문장의 가장 마지막에 있는 보조 용언 '있다'가 된다.

이와 같은 문제를 해결하기 위해 문장 내의 형태소들을 하나의 의미 있는 구성 성분인 말덩이(chunk)로 구묶음한 뒤 구문을 분석할 수 있다(Abney, 1991; Abney, 1996; 김재훈, 2000a). 말덩이란 인간이 한 번에 받아들이는 언어의 단위로, 문법적 및 의미적으로 하나의 기능을 수행하며, 연속성, 비중첩성, 비재귀성이라는 특징을 가진다(김재훈, 2000b). 또한, 형태소분석된 문장에 대해 말덩이 단위를 인식하고 표지를 부여하는 과정을 구묶음(chunking)이라고 한다(Abney, 1991; 박의규 & 나동열, 2006).

말덩이 단위로 구문을 분석하면 구문분석이 문장 성분(sentence constituent) 단위로 이루어지므로 Figure 1.2와 같이 구문분석을 수행할 노드 수가 적어져 구문분석기의 속도 및 정확도가 향상될 수 있다.

Figure 1.2 Examples of dependency parsing with chunking.

또한, 기존의 방식과 달리 Figure 1.2에서처럼 구문적 중심어와 의미적 중심어가 일치하므로, 한국어의 지배소 후위 원칙을 위배하지 않으면서 의미적으로도 어색하지 않은 구문분석이 이루어질 수 있다. 이는 구문분석 단계에서 구문분석 이후에 이루어지는 의미 분석(semantic analysis)까지 함께 수행할 수 있다는 면에서 의의가 있다.

따라서, 본 논문에서는 구묶음을 반영한 의존구조 분석 방법을 제안하고, 이를 어절 단위의 분석 방법과 비교한다. 이를 통해 구묶음을 반영한 의존구조 분석의 유효성을 검증하고 효율적인 한국어 구문분석 방안에 대해 고찰하는 데 본 논문의 의의가 있다.

본 논문의 구성은 다음과 같다. 2장에서는 구문분석 및 구묶음에 관련된 기존 연구들을 살펴본다. 3장에서는 구묶음을 반영한 한국어 의존구조 분석을 위해 구묶음의 기본 단위인 말덩이에 대해 정의하고 이를 기반으로 말뭉치를 구축하여 구묶음을 반영한 의존구조 분석을 제안한다. 4장에서는 3장에서 제안한 구묶음을 반영한 의존구조 분석을 수행하고 그 결과를 분석하여 기존 방법과의 비교를 진행한다. 5장에서는 결론 및 향후 연구에 대해 논한다.

제 2 장 관련 연구

구문분석은 문장 구성 성분들의 관계를 파악하는 과정을 말한다. 구문분석을 통해 문장의 구조를 결정함으로써 의미적 중의성을 해소할 수 있다. 이 장에서는 구문분석의 한 방법론인 의존구조 분석과 관련된 연구를 살펴보고, 본 논문에서 한국어 의존구조 분석을 위해 제안하는 방법인 구묶음과 관련된 기존 연구를 소개한다. 또한, 구문분석기를 학습하고 평가하는 데 필요한 구문분석 말뭉치에 대해 살펴본다.

2.1 의존구조 분석

구문분석의 한 방법론인 의존구조 분석은 문장 성분 간의 지배소와 의존소의 관계를 파악함으로써 문장의 구조를 분석하는 방법이다. 이는 문장을 구성하는 요소의 위치에 제약이 적고 문장 구성 성분의 생략에도 유연하게 대처할 수 있어 최근 한국어 구문분석에서 많이 연구되고 있다. 한국어 의존구조 분석은 일반적으로 세 가지 특징을 가지고 있다(나동렬, 1994). 첫 번째는 지배소가 의존소의 뒤에 위치한다는 지배소 후위 원칙(head final constraint), 두 번째는 문장 내의 모든 성분이 각각 하나의 지배소를 가지며 순환 없이 트리 구조를 이룬다는 지배소 유일의 원칙(single head constraint), 세 번째는 의존 관계 사이의 간선이 서로 교차하지 않는다는 투영성의 원칙(projective constraint)이다. 특히, 문장 요소의 위치가 가변적이고 생략이 잦은 한국어의 언어학적 특성으로 인해 의존구조 분석은 한국어 구문분석에도 많이 적용되고 있다.

의존구조 분석방법으로는 크게 전이 기반 분석방법과 그래프 기반 분석방법이 있으며, 최근 심층학습이 주목을 받으면서 이들 방법에 신경망 기

법들을 적용한 방법들이 제안되었다(Chen & Manning, 2014; Pei *et al.*, 2015). 이 장에서는 의존구조 분석에 신경망 모델을 적용한 방법을 중심으로 기존 연구들을 살펴본다.

2.1.1 의존구조 분석 방법론

(1) 전이 기반 구문분석

전이 기반 구문분석(Nivre, 2003)은 두 단어의 의존 관계를 결정할 전이 행위(action)를 결정함으로써 입력 문장에 대한 의존 트리를 구성해 나가는 방법으로 결정적 의존구조 분석 방법(deterministic dependency parsing)이다. 전이 기반 구문분석기의 기본 구조는 입력 문장을 위한 버퍼, 전이 행위를 결정할 단어들이 저장될 스택, 그리고 결정된 전이 행위들의 집합으로 이루어져 있으며, 이 전이 행위들의 집합으로 입력 문장에 대해 구문분석된 결과를 얻을 수 있다. 전이 행위들은 보통 left arc, right arc, shift, reduce가 있으며, 전이 행위를 언제, 어떻게 결정짓느냐에 따라 Arc-standard(Nivre, 2004), Arc-eager(Nivre, 2003), Arc-hybrid(Kuhlmann *et al.*, 2011), Arc-swift(Qi & Manning, 2017) 등의 방법이 제안되었다.

의존구조 분석에 심층학습을 적용한 시도로는 단어, 품사, 의존 관계 등의 자질을 입력으로 하여 전이 행위를 출력하는 과정을 순방향 신경망(feed-forward neural network) 모델로 구성한 연구가 있다(Chen & Manning, 2014). 이후, 기존의 의존구조 분석 시스템에서 전이 행위를 결정하기 위해 stack LSTM을 이용한 연구가 진행되었다(Dyer *et al.*, 2015). 이는 스택, 버퍼, 전이 행위의 상태 정보를 stack LSTM을 통해 인코딩하고, 이를 입력으로 하여 단일 신경망을 통해 다음 전이 행위를 출력하는 방법이다. SyntaxNet은 순환 신경망을 사용하지 않고 feed-forward 신경망에 beam search 방식을 이용하여 의존 관계를 결정할 때 생기는 모호성 문제를 해결 했다(Andor *et al.*, 2016). 이 외에도 Bi-LSTM 모델을 이용하여 입력 단어들의 표상을 얻은 후 다층 신경망을 이용해 전이 행위에 대한 점수를

구하는 방법이 연구되었다(Kiperwasser & Goldberg, 2016).

전이 기반 방법은 구문분석의 복잡도가 입력 문장의 길이에 선형적이라는 장점이 있다. 하지만 지역적 정보에 의존하여 전이 행위를 결정하므로 문장 내에서 먼 거리에 있는 단어와의 관계를 분석하기 어려우며, 전이 행위를 한번 잘못 결정하면 이후에도 영향을 미친다는 한계점이 있다.

(2) 그래프 기반 구문분석

그래프 기반 구문분석(McDonald *et al.*, 2005a)은 입력 문장의 모든 단어를 노드로 하는 그래프를 구성하고, 각 간선에 의존 관계 점수를 부여한 뒤 최대 신장 트리를 찾는 방법으로 비결정적 의존구조 분석 방법(non-deterministic dependency parsing)이다. 의존 트리 $G=(V,A)$에서 노드를 V, 간선을 A로 하는 방향성 그래프라고 할 때, G의 점수는 식 (2.1)과 같이 나타낼 수 있다(Kübler *et al.*, 2009).

$$score(G)=\sum_{(w_i,r,w_j)\in A}\lambda(w_i,r,w_j) \tag{2.1}$$

위 식에서 $\lambda(w_i,r,w_j)$는 부분 트리에 대한 점수를 나타내며 보통 가중치 벡터 w와 특징함수 $f(w_i,r,w_j)$의 합성곱으로 표현된다(McDonald *et al.*, 2005a). 이때, w는 입력 문장 $S=w_0w_1...w_n$의 한 단어로 w_i와 w_j는 각각 지배소와 의존소가 되며, r은 둘 사이의 관계명이 된다.

그래프 기반 구문분석에서 입력 문장 S에 대해 최종적으로 결정되는 구문분석 트리는 식 (2.1)의 값을 최대로 하는 트리가 된다. 이는 $G\in\varsigma_S$일 때, 식 (2.2)와 같이 표현할 수 있다.

$$\begin{aligned} parse(S) &= \underset{G=(V,A)\in\varsigma_S}{argmax}\ score(G) \\ &= \underset{G=(V,A)\in\varsigma_S}{argmax}\sum_{(w_i,r,w_j)\in A}\lambda(w_i,r,w_j) \end{aligned} \tag{2.2}$$

이때 각 단어 사이의 의존 관계에 대한 가중치나 확률을 부여할 때 심층 학습을 적용할 수 있으며, 트리 전체가 가장 높은 점수를 갖게 하는 의존 관계들을 결정하기 위해 Collins(Collins, 2002), Eisner(Eisner, 1996), Chu-Liu-Edmonds(Chu & Liu, 1965; Edmonds, 1967) 등이 제안한 알고리즘이 사용된다.

그래프 기반 구문분석에 심층학습을 적용한 사례로는 기계학습 분야에서 고안된 주의 집중 메커니즘(attention mechanism)을 그래프 기반 구문분석에 적용한 경우가 있다(Kiperwasser & Goldberg, 2016). 이는 Bi-LSTM 모델을 통해 각 입력 단어의 표상을 구하고, 이를 지배소가 될 수 있는 단어들의 표상과 결합하여 다층 신경망 모델의 입력으로 한 뒤, 각 단어 간 의존 관계의 점수를 구할 때 주의 집중 메커니즘을 이용하는 방법이다. 여기서 나아가 의존 관계 점수를 부여할 때 기존의 주의 집중 메커니즘 대신 bilinear attention(Luong, 2015)으로부터 확장된 biaffine attention(Dozat & Manning, 2017)을 이용한 연구도 진행되었다.

(3) 포인터 네트워크 기반 구문분석

포인터 네트워크 모델은 기존의 전이 기반이나 그래프 기반의 구문분석과 달리 신경망의 출력이 해당 입력 단어의 의존소 또는 지배소의 위치를 가리키게 하는 방법이다. 의존소 또는 지배소 관계를 결정하기 위해 주의 집중 방법이 사용되며, 네트워크의 출력 결과만으로 의존 관계를 파악할 수 있다는 특징이 있다. 가장 먼저 제안된 기본적인 포인터 네트워크 모델(Vinyals *et al.*, 2015)은 기계 번역의 Sequence-to-Sequence 모델과 마찬가지로 인코더와 디코더로 구성되어 있다. 인코더에서는 입력 문장을 순차적으로 인코딩하며, 디코더에서는 입력 단어의 지배소가 되는 단어의 위치를 출력한다. 포인터 네트워크는 인코더를 통해 문장 전체를 함축한 결과를 토대로 지배소를 결정하게 되므로, 일반적인 전이 기반 방법과 달리 문장 전체의 정보를 반영할 수 있다.

스택-포인터 네트워크(Ma *et al.*, 2018)는 포인터 네트워크를 전이 기반

구문분석에 맞게 확장한 모델이다. 이는 의존 관계를 결정할 때 지엽적인 정보만 반영하는 전이 기반 구문분석의 단점을 포인터 네트워크의 특징인 인코더와 주의 집중 매커니즘을 통해 보완하였다. 스택-포인터 네트워크는 포인터 네트워크와 달리 입력 단어에 대한 의존소를 찾아 순차적으로 부분 트리를 구축해 나가는 방법이다. 이때, 하나의 지배소는 여러 개의 의존소를 가질 수 있으므로 내부 스택을 통해 지배소가 될 단어를 저장하여 여러 번 의존소를 찾을 수 있게 하였다. 이러한 과정을 통해 문장 전체의 정보뿐만 아니라 현재까지 생성된 부분 트리에 대한 정보를 반영할 수 있다. 최근에는 의존소를 찾는 부분 트리 형성 과정을 스택에 있는 단어의 순서가 아니라 문장의 구성 방향과 같이 좌측부터 차례로 찾아 나가는 left-to-right 방법의 포인터 네트워크가 제안되었다(Fernández-González & Gómez-Rodríguez, 2019). 이는 내부 스택 없이 간단히 이전 어절과 이후 어절을 이용하여 의존소를 결정하며, 스택-포인터 네트워크에서는 $2n-1$ 이었던 전이 행위 결정 횟수를 n으로 줄이면서도 높은 성능을 보인다.

2.1.2 한국어 의존구조 분석

한국어 구문분석에도 심층학습을 이용한 구문분석이 활발히 이루어졌다. 한국어 역시 전이 기반 방법의 의존구조 분석에 심층학습을 이용하여 자질 튜닝 작업에 들어가는 시간과 노력을 줄이는 연구가 먼저 시도되었다(이창기 외, 2014). 이후 순환 신경망을 이용하여 문장 내에서 먼 거리에 있는 단어들의 의존 관계를 고려하는 방법이 연구되었으며(이건일 & 이종혁, 2015), 의존구조 분석을 위한 순환 신경망의 입력으로 또 다른 순환 신경망을 이용해 한국어 음절, 형태소 등의 표상을 얻어 내는 연구도 진행되었다(나승훈, 2016).

그래프 기반의 한국어 구문분석으로는 biaffine attention을 이용한 구문분석에 한국어를 표현하기 위한 단어 표상층을 결합한 Deep biaffine attention 모델이 연구되었으며(나승훈 외, 2017), 이외에도 ELMo(Peters *et al*., 2018)나 BERT(Devlin *et al*., 2018), XLNet(Yang *et al*., 2019)과 같은 문

맥을 고려한 단어 표상을 이용하여 구문분석을 수행하기도 하였다(홍승연 외, 2019; 박천음 외, 2019; 김민석 외, 2019). 또한, biaffine attention 모델에 추가적인 자질 없이 GNNs(Graph Neural Networks)를 통해 고차원 정보를 학습하는 방법도 연구되었다(민진우 외, 2019a).

포인터 네트워크를 이용한 연구도 활발히 진행되었으며, 이를 이용하여 멀티 태스크 학습 기반으로 의존 관계와 의존 관계명을 동시에 예측하는 연구가 이루어졌다(박천음 & 이창기, 2017). 또한, 한국어를 비롯한 형태적으로 복잡한 언어(morphologically rich language)에 적합하도록 음절, 형태소, 품사 등의 요소를 적절히 이용해 입력 표상을 확장한 실험들이 이루어졌다(홍승연 외, 2018; 차다은 외, 2018; 안재현 & 고영중, 2018). 스택-포인터 네트워크에서 이전 단계의 트리 정보를 반영해주기 위해 부모 노드 외에도 형제 노드와 조부모 노드인 고차원 정보를 함께 반영하는 연구도 수행되었으며(최용석 & 이공주, 2019), 멀티헤드 어텐션(multi-head attention)을 적용한 실험도 수행되었다(김홍진 외, 2019).

이외에도 전이 기반 방법과 그래프 기반 방법을 통합한 모델(민진우, 2019b)이나, 그래프 기반의 biaffine attention 모델과 포인터 네트워크 모델을 앙상블하는 방법(한장훈 외, 2019; 조경철 외, 2019)으로 구문분석 문제를 해결하려는 노력이 이어졌다.

2.2 구묶음

구묶음(chunking, shallow parsing, partial parsing)은 문장의 구성성분 단위를 인식하는 작업이다. 실생활에서 사용되는 문장들은 보통 그 길이가 길거나 생략, 도치 등으로 불완전한 문장이 대부분이다. 이러한 문장에 대해 바로 구문분석을 하면 정확하지 않은 결과를 출력하거나 시스템의 계산 복잡도가 높아질 수 있다. 이러한 경우, 구묶음을 통해 먼저 세부 단위로 분석한 후, 구문분석을 수행하면 결과적으로 구문분석의 정확도와 효율을 높이고 분석 속도를 개선할 수 있다. 이 장에서는 본 논문에서 제안한 구문분석 방법론 중 하나인 구묶음과 관련된 기존 연구들을 살펴본다.

2.2.1 구묶음과 말덩이

자연언어처리에 있어 구묶음 또는 부분 구문분석(partial parsing)은 구문분석 이전에 수행되는 전처리 단계이다(Abney, 1991). 이는 문장 내에서 구문적으로 단일한 역할을 수행하는 형태소들을 하나의 단위로 묶어 구문분석의 입력 성분 수를 줄이고 분석 결과의 모호성을 해소하는 등 구문분석의 문제들을 완화하는 역할을 한다(Abney, 1996; 김재훈, 2000a). 본 논문에서 구묶음은 입력 문장을 말덩이 단위로 묶는 작업을 말한다. 말덩이는 구문적인 의미에서의 문장 구성성분(constituent)으로, 의미를 나타내는 중심어와 이에 문법적 역할을 더해 주는 기능어들로 이루어지며, 문장 내에서 다른 말덩이들과 겹치지 않고 연속적으로 존재한다는 성질을 지니고 있다(Abney, 1991). Abney는 유한상태 오토마타를 이용해 단계적으로 기본구를 찾고 이들을 결합하여 더 큰 단위의 구를 이루어 나가며 구문트리를 형상하고자 하였다(Abney, 1996). Ramshaw와 Marcus의 연구(Ramshaw & Marcus, 1995)에서는 Brill의 변형 기반 학습(transformation-based learning)을 이용하여 자동 학습 기법으로 구묶음 인식을 수행하였다. 이는 구묶음을 표지 부착 문제로 간주하여 구묶음 표지가 부착된 말뭉치로 학습한 뒤 다른 말뭉치에 대해 자동적으로 구묶음을 인식할 수 있게 한 것으로, 주로

명사구와 동사구에 대해 높은 인식률을 보였다. 이 외에도 구묶음 표지와 문법 패턴을 이용(Bourigault, 1992; Voutilainen, 1993)하거나 유한상태 오토마타(Grefenstette, 1996)를 이용하여 명사구를 인식하는 연구가 수행되었다. 이후에는 Argamon *et al*.(1998), Veenstra(1998), Daelemans *et al*.(1999) 등을 중심으로 메모리 기반의 학습이 이루어졌다.

2.2.2 한국어 구묶음

한국어 구묶음에 대한 연구는 그 중요성에 비해 많은 연구가 활발히 이루어지지는 않았지만, 2000년대 초반까지 꾸준히 진행되어 왔다. 2000년 이전에는 형태소분석과 구문분석의 중간 단계로 구문 요소의 형성 단계를 설정하고, 형태소로부터 구문 요소를 형성하는 연구가 진행되었다(안동언, 1987). 또한, 구문분석 과정에서 과도하게 발생하는 연산량과 모호성을 줄이기 위해 부분적 어절 결합을 이용한 구문분석기를 구현한 연구가 수행되었다(김창제 외, 1995). 이외에도 간단한 문맥 자유 문법이나 연어 패턴 정보 등의 규칙을 통해 명사구와 동사구를 인식하고자 하는 노력이 이어졌다(Yoon *et al*., 1999). 영어권과 마찬가지로 변형 기반 학습을 이용하여 얻은 일련의 규칙들을 통해 한국어 기반 명사구를 인식하는 연구도 수행되었다(양재형, 2000). 이후 한국어처리에 있어 부분 구문분석의 역할 및 필요성을 제시하고 이에 대한 방법론 및 다양한 응용 분야에 대한 연구가 진행되었으며(김재훈, 2000), 하나의 기능적 역할이나 구문적 역할을 수행하는 결합된 형태소를 구문 형태소로 정의하고, 이를 구문분석의 기본 입력 단위로 간주함으로써 형태소 및 구문 모호성을 축소하기 위한 구문 단위 형태소의 필요성을 보였다(황이규 외, 2000). 이후 문장에서의 기본 구를 인식하기 위해 기계학습을 적용한 연구가 진행되었으며(황영숙 외, 2002), 규칙 기반의 한국어 부분 구문분석기를 구현하고, 이를 포함한 구문분석과 기존의 방식을 비교한 연구도 수행되었다(이공주 & 김재훈, 2003). 또한, 기존에 명사열 위주로 수행되었던 구묶음을 비롯하여 보조용언과 의존 명사에 대한 구묶음 방식을 제안하고, 이를 활용한 구문분석

에서 의존 관계를 보다 정확하게 추출할 수 있음을 보였다(박의규 & 나동열, 2006). 이외에도 특정 의존 명사를 포함하는 보조 용언을 구성하는 말덩이를 중심으로 명확한 기준을 언어학적인 방법으로 제시하고 말뭉치 구축 오류를 방지할 해결방안을 제안하였다(김태웅 외, 2006). 최근에는 구문분석을 위한 전처리 단계로서 한국어 문장의 모든 구성 성분에 대해 구묶음을 수행하기 위한 말덩이의 기준 및 그 표지를 제시하고, 심층학습을 이용해 구묶음을 수행하였다(남궁영 외, 2019).

2.3 의존구조 분석 말뭉치

말뭉치란 언어 연구를 위해 텍스트를 컴퓨터가 읽을 수 있는 형태로 모아 놓은 언어 자료를 말하며 코퍼스(corpus)라고도 한다. 말뭉치는 원시 텍스트의 모음일 수도 있지만, 대부분은 이용하려는 작업에 따라 해당 작업에 필요한 문법적인 주석이 달린 경우가 많다. 구문분석 말뭉치 역시 원시 텍스트와 함께 구문분석에 필요한 통사적인 정보가 포함된 말뭉치를 말한다.

말뭉치를 구축하는 일은 주석에 대한 상세한 지침이 필요하며 생성된 말뭉치의 효용성 및 신뢰성을 입증할 수 있어야 하므로, 시간적, 금전적으로 큰 비용과 노력이 드는 작업이다. 구문분석은 구 구조 분석이 먼저 활발히 연구되었기 때문에 이와 관련된 말뭉치가 의존구조 말뭉치보다 상대적으로 많다. 따라서 의존구조 말뭉치는 의존구조 분석만을 목적으로 새로이 구축되기도 하지만, 기존의 구 구조 분석 말뭉치를 의존구조 분석용으로 변환하여 사용하는 경우도 많다. 이 장에서는 언어의 종류와 관계없이 일관된 형식의 의존구조 분석 말뭉치 구축을 위한 UD(Universal Dependency) 프로젝트에 대해 설명하고, 한국어 구문분석 말뭉치에 대해 알아본다.

2.3.1 UD 말뭉치

UD(Universal Dependency)는 언어의 종류에 관계없이 일관적인 주석을 가지고 말뭉치를 구축하기 위한 작업 기준을 말한다. 이를 통해 다국어 구문분석기를 개발하거나 언어 간 확장을 용이하게 함으로써 구문분석 및 언어 처리에 관한 연구를 촉진하는 것을 목적으로 한다.

UD에서는 의존구조 말뭉치를 표기하기 위한 통일된 양식을 제공하기 위해 현재 CoNLL-U(Zeman *et al*., 2017) 형식을 채택하고 있으며, 이는 원문과 함께, 10개의 열로 이루어져 있다. 그 중 HEAD 열은 해당 단어의 지배소 순번을 가리키며, DEPREL 열은 이에 해당하는 의존 관계명을 기술한 열이다. UD에서 의존 관계명은 언어의 종류와 관계없이 일관된 주석을 적용하기 위해 언어의 유형론적 관점에서 연구한 내용(de Marneffe *et al*., 2014)을 바탕으로 보완한 50개의 관계명을 사용한다.

한국어에서도 UD의 지침을 따르기 위한 표준안을 구축하는 연구가 이어지고 있으나, 아직 통일된 지침은 없는 실정이다.

2.3.2 한국어 의존구조 말뭉치

세종 구문분석 말뭉치는 21세기 세종 계획의 일환으로 구축된 구 구조 기반의 한국어 구문분석 말뭉치다(김홍규 외, 2007; 홍윤표, 2009). 현재 공식적으로 구축된 한국어 의존구조 분석 말뭉치는 없으며, 한국어 의존구조 분석을 연구하기 위해서는 연구자마다 각각 자체 기준으로 기존의 세종 구문분석 말뭉치를 의존구조로 변환해서 사용하고 있다.

세종 구문분석 말뭉치를 의존구조로 변환하는 연구로 중심어 규칙을 적용한 사례가 있으며(Choi & Palmer, 2011), 문장에서 기능어 및 기호가 다른 단어의 지배소가 되는 문제를 해결하기 위하여, 일부 경우에 중심어 전위 규칙을 적용한 연구가 있다(최용석 & 이공주, 2018). 해당 연구에서는 의존구조로 변환할 때 UD의 CoNLL-U형식을 따르고자 하였으며, 현재

UD와 관련된 한국어를 위한 통일된 규정안이 없어 최대한 세종 말뭉치의 구조를 유지하면서 구문분석에 필요한 최소한의 필드에 초점을 맞추어 말뭉치를 생성하였다. 이 외에도 구 구조 방식으로 구축된 Google UD Treebank, Penn Korean Treebank, KAIST Treebank를 각각 UD 양식에 맞춰 의존구조로 변환한 사례가 있다(Chun *et al*., 2018).

제 3 장 구묶음을 반영한 한국어 의존구조 분석

한국어는 문장 내 구성 요소 간의 이동과 생략이 자유로워 구문을 분석할 때 중의성 문제가 발생할 수 있다. 또한, 의존구조 분석을 할 때 Figure 1.1과 같이 구문적 중심어와 의미적 중심어가 불일치하게 되어 의존 관계 결정에 방향성 문제를 안고 있다. 이러한 문제를 해결하기 위해 Figure 1.2와 같이 구문적으로 하나의 역할을 수행하는 형태소들을 하나로 묶은 뒤에 구문분석을 수행할 수 있다(김재훈, 2000a; 박의규 & 나동열, 2006). 이러한 과정을 구묶음이라고 하며, 이렇게 묶인 하나의 덩어리를 말덩이라고 한다(Abney, 1991). 구묶음을 반영하여 구문분석을 하게 되면, 구문분석의 입력 성분 수를 줄일 수 있으며, 구문적 중심어와 의미적 중심어가 일치하게 되므로 구문분석 이후에 이루어지는 의미분석의 역할까지 함께 수행할 수 있게 된다.

이 장에서는 효율적인 한국어 구문분석을 위해 구묶음을 반영한 한국어 구문분석에 대해 설명한다. 3.1에서는 한국어에 있어 구묶음과 말덩이의 개념에 대해 정의하고, 말덩이의 종류 및 표지를 소개한다. 이를 바탕으로 3.2에서는 심층학습 모델을 이용하여 한국어 구묶음을 수행한다. 3.3에서는 3.2의 결과를 토대로 기존 말뭉치에 구묶음을 적용해 구묶음을 반영한 한국어 의존구조 말뭉치를 구축하는 방안에 관해 서술한다. 끝으로 3.4에서는 구묶음을 반영한 한국어 구문분석에 관해 기술한다.

3.1 한국어 말덩이

말덩이는 보통 어떠한 고정된 형식을 가지며, 하나의 내용어와 그 주변의 기능어들로 구성된다(Abney, 1991). 인지 심리학에서는 사람이 한 번에 받아들이는 언어의 구조(performance structures)가 있으며, 이는 자연스럽게 발화되는 단위나, 또는 내용어(syntactic head, content word)를 기준으로 분절되는 말덩이 등으로 표상된다고 기술한다(Gee & Grosjean, 1983).

한국어 역시 형태소나 어절과 같은 문법적인 기능만 고려한 단위가 아닌 사람이 한 번에 인지하는 언어의 단위가 있다. 이를 말덩이라고 한다.

3.1.1 한국어 말덩이의 정의

말덩이란 인간이 한 번에 받아들이는 언어의 단위로, 문법적, 의미적으로 하나의 기능을 가진 구를 말한다(Abney, 1995). 한국어에서 말덩이는 문법적, 의미적으로 같은 역할을 하는 형태소들의 묶음으로 Figure 3.1과 같이 나타낼 수 있다.

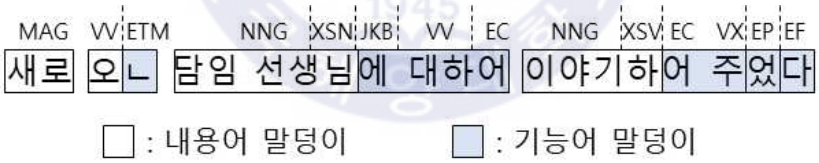

Figure 3.1 An example of a chunked sentence.

Figure 3.1에서 볼 수 있듯이 말덩이는 크게 내용어 말덩이(content chunk)와 기능어 말덩이(function chunk)로 나누어진다. 내용어 말덩이는 말덩이 내에 반드시 의미적 중심어를 가지며, 기능어 말덩이는 의미적 중심어를 가지지 않고 그 자체로 하나의 덩어리를 형성한다. 내용어 말덩이는 그 자체로 하나의 문장 성분을 이루거나, 하나의 내용어 말덩이에 하나 이상의 기능어 말덩이가 모여 하나의 문장 성분을 이룬다. 이를 식으로

표현하면 다음과 같다.

$$\text{sentence constituent} = [\text{content chunk}]^{+}[\text{function chunk}]^{*}$$

따라서 완전한 구묶음 이후의 문장은 Figure 3.2와 같이 한국어의 7가지 문장 성분[1] 으로 표현이 가능하다.

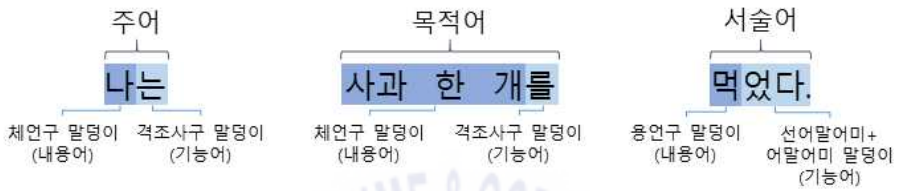

Figure 3.2 An example of chunks and sentence constituents in a chunked sentence.

한국어 말덩이는 다음과 같은 특징들을 가진다.

- 완전한 구묶음 이후의 문장은 한국어의 7가지 문장 성분으로 표현할 수 있다.
- 서술어를 제외한 문장 성분들은 모두 하나의 내용어로 이루어져 있거나 하나의 내용어에 하나의 기능어가 첨가된 형식으로 이루어진다.
- 서술어는 본용언을 내용어로 하며, 기능어로는 하나 이상의 보조용언, 연결어미, 선어말어미, 종결어미 등의 말덩이가 함께 올 수 있다.
- 내용어 말덩이는 반드시 의미적 중심어(semantic head)를 가진다.
- 기능어 말덩이에 속하는 형태소들은 의미적 중심어를 가지지 않고 그 자체로 하나의 덩어리를 형성한다.
- 병렬구문은 의미적 중심어가 여러 개 존재하므로 각각의 구절을 하나의 말덩이로 간주한다.

1) 주어, 서술어, 목적어, 보어, 관형어, 부사어, 독립어

- 동일한 구가 연속해서 등장할 경우 최장일치를 기본으로 한다.
- 말덩이를 이루는 구성 성분들은 연속적이어야 한다(no discontinuous).
- 문장 내의 하나의 형태소는 반드시 하나의 말덩이에 속하며, 서로 다른 말덩이에 중복하여 구묶음 되지 않는다. 즉, 비중첩성을 지닌다(no center-embedded).
- 말덩이 내의 구문 구조는 선형으로, 트리 구조를 형성하지 않고 비재귀성을 지닌다(no recursive).

3.1.2 말덩이 종류 및 표지

말덩이는 크게 내용어 말덩이와 기능어 말덩이가 있으며, 3.1.1에서 정의한 사항과 특징에 따라 내용어 말덩이 6개, 기능어 말덩이 11개인 총 17개의 말덩이로 분류된다(남궁영 & 김재훈, 2018a). 말덩이는 같은 기능을 하는 연속된 형태소들의 묶음으로, 각 형태소는 다른 성분들과 함께 구묶음이 되지 않을 경우 기본적으로 자신의 품사명과 같은 말덩이에 속할 수 있으며 이를 **표준 말덩이**라 한다. 각 말덩이의 종류 및 표지는 Table 3.1과 같으며, 각 말덩이에 대한 설명은 (남궁영 & 김재훈, 2018a)에 기술된 바와 같다.

Table 3.1 Types and labels of chunks.

말덩이 종류	말덩이 표지	
내용어 말덩이	체언구(NX), 본용언구(PX), 지정사구(CX), 부사구(AX), 관형사구(MX), 독립어구(IX)	
기능어 말덩이	격조사구(JKX), 관형격조사구(JMX), 보조사구(JUX), 접속조사구(JCK), 호격조사구(JVX),	보조용언구(PUX), 선어말어미구(EPX), 연결어미구(ECX), 전성어미구(ETX), 종결어미구(EFX), 문장부호구(SYX),

3.2 한국어 구묶음

구묶음은 형태소분석, 개체명 인식과 함께 순차 표지 부착 문제로 해결할 수 있다. 구묶음은 크게 말덩이를 인식하는 단계와 이에 해당하는 말덩이 표지를 부착하는 단계로 이루어진다(Abney, 1996). 최근에는 자연언어 처리에 심층학습 기법이 적용되면서 신경망을 이용하여 이 단계를 한 번에 해결하고, 또한, 자질 추출에 드는 노력을 경감하면서도 우수한 결과를 보이고 있다(Lample *et al*., 2016). 이 장에서는 초기 구묶음 말뭉치 구축을 위해 반자동 방식으로 한국어 구묶음을 수행한 사례를 보이고, 심층학습 모델 중 하나인 Bi-LSTM/CRFs 모델(Huang *et al*., 2015)을 이용하여 한국어 문장 내의 모든 구성 요소에 대해 구묶음을 수행한다.

3.2.1 구묶음 말뭉치 구축

3.1.1절과 3.1.2절에서 말덩이에 대해 기술한 내용을 바탕으로 구묶음을 수행할 수 있다. 구묶음은 형태소분석된 문장에 대해 수행한다. 따라서 세종 형태 분석 말뭉치를 바탕으로 (남궁영 & 김재훈, 2018a)에서 정의한 기준 및 그 표지를 중심으로 구묶음을 수행하여 말덩이 표지가 부착된 구묶음 말뭉치를 구축할 수 있다. 이는 언어 정보 부착 시스템(Noh *et al*., 2018)을 활용하여 반자동 형식으로 수행할 수 있으며, 그 결과를 Figure 3.3과 같이 CoNLL 형식으로 변환하여 구묶음 시스템의 학습 및 평가에 이용한다. 이때, ‘text’는 원문이며, 각 열은 왼쪽부터 차례대로 순번, 형태소, 품사, 띄어쓰기 여부, 말덩이 표지를 나타낸다.

```
# text = 아마 그런 사람은 없으리라 본다.
1       아마        MAG     1       B-AX
2       그런        MM      1       B-NX
3       사람        NNG     0       I-NX
4       은          JX      1       B-JUX
5       없          VA      0       B-PX
6       으리라      EC      1       B-PUX
7       보          VV      0       I-PUX
8       ㄴ다        EF      0       B-EFX
9       .           SF      0       B-SYX
```

Figure 3.3 An example of a chunked sentence in Korean chunk corpus.

3.2.2 심층학습을 이용한 한국어 구묶음

이 절에서는 3.2.1절의 구묶음 말뭉치를 이용하여 심층학습 모델을 이용한 한국어 구묶음을 수행한다. 구묶음을 위해 순차 표지 부착 문제에서 뛰어난 성능을 보이는 Bi-LSTM/CRFs 모델을 이용한다. 이는 Figure 3.4와 같이 기존의 순환 신경망(RNN, Recurrent Neural Network)에서 장기 의존성을 보완하기 위해 고안된 장단기 기억(LSTM, Long-Short Term Memory) 계층을 양방향으로 이용하고, 출력 계층에서 CRFs(Conditional Random Fields)를 통해 가장 적합한 표지를 선택하는 방식이다.

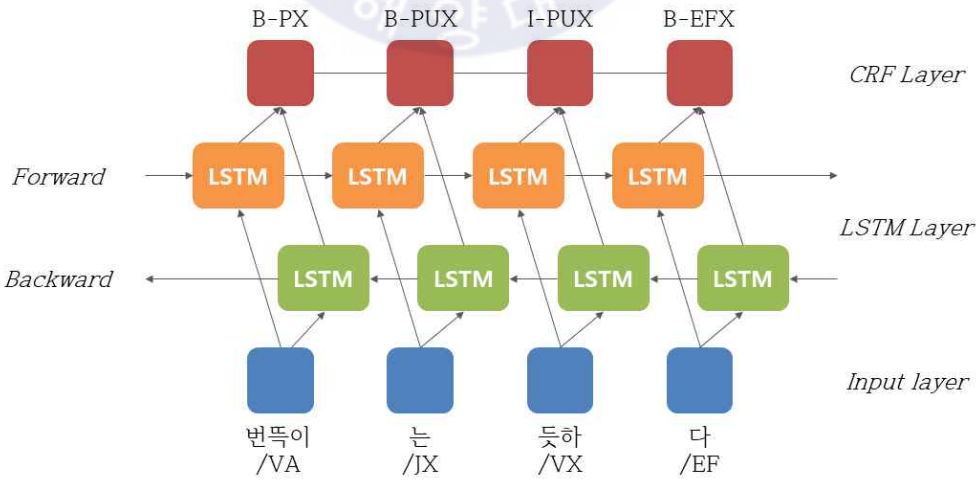

Figure 3.4 The structure of the Bi-LSTM/CRFs model for Korean chunking.

구묶음은 형태소분석된 문장에 대해 수행되며, 같은 형태의 형태소라도 품사에 따라 구묶음 결과가 달라지는 등 품사 태그에 영향을 많이 받는다. 따라서, 모델의 입력은 Figure 3.4에서와 같이 각 형태소와 그에 상응하는 품사를 결합한 것을 한 단위로 표현하여 사용한다.

인식한 말덩이의 표지를 표기하기 위한 방법으로는 Figure 3.4의 최종 출력에서 표기한 바와 같이 순차 표지 부착 문제에 보편적으로 사용되는 IOB 형식(Ramshaw & Marcus, 1995)을 이용한다. 이때 구묶음 작업에 있어서 문장 내에 있는 형태소는 모두 임의의 말덩이로 구묶음 되므로, 하나의 형태소는 반드시 하나의 말덩이 표지를 가지게 된다(남궁영 외, 2018b). 따라서 어떤 형태소든 말덩이에 속하지 않는 예외가 없으므로 흔히 사용되는 IOB 형식 중 'O' 표지는 사용하지 않는다.

심층학습을 이용한 한국어 구묶음에 관한 실험 환경 및 결과는 4장에서 자세히 기술한다.

3.3 구묶음을 반영한 한국어 의존구조 말뭉치 생성

이 절에서는 구묶음을 반영한 한국어 의존구조 말뭉치를 생성하는 방안에 대해 논한다. 일반적으로 대량의 말뭉치를 구축하는 일에는 시간적, 비용적 측면에서 큰 노력이 필요하다(NIKL, 2010; Marcus *et al.*, 1993). 따라서 본 논문에서는 말덩이 기반의 의존구조 말뭉치를 구축하기 위해 기존의 의존구조 말뭉치로부터 변환하는 알고리즘을 기술한다.

3.3.1 구묶음을 반영한 의존구조 말뭉치

일반적으로 의존구조 말뭉치를 구축할 때 2.3.1절에서 언급한 UD의 CoNLL-U 형식을 따르는 것이 보편적이다. 구문분석 말뭉치에 적용되는 UD의 CoNLL-U 형식은 Figure 3.5와 같이 10개의 열[2]로 이루어지며, 이 중 HEAD(지배소) 열은 해당 토큰의 지배소 순번을, DEPREL 열은 지배소

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

와의 의존 관계 표지를 기술한다.

#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	_	_

Figure 3.5 An example of original Korean dependency corpus.

(최용석 & 이공주, 2018)의 유연한 중심어 후위 원칙의 말뭉치는 CoNLL-U 형식을 따르며 Figure 3.5와 같이 구성되어 있다. 세종 구문분석 말뭉치와 마찬가지로 한 행, 즉 구문분석의 한 단위가 되는 토큰은 주로 어절 단위이며, 띄어쓰기가 있는 경우 문장 부호나 쌍이 있는 기호를 분리하여 하나의 토큰으로 취급한다.

반면, 말덩이 기반의 의존구조 말뭉치는 한 행이 내용어 말덩이와 기능어 말덩이로 이루어진 문장 성분 단위로 이루어져 있다. 말덩이 기반의 의존구조 말뭉치 역시 기본적으로 CoNLL-U 형식을 따르며 Figure 3.6과 같이 이루어져 있다.

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

Figure 3.6 An example of Korean dependency corpus reflected chunking.

기존과 다른 점은 언어 형식을 기입하는 FORM 열에 말덩이의 특성을 반영하여 내용어 말덩이(conts)와 기능어 말덩이(func)로 나누어 표기한다는

점이다. 또한, 말덩이 표지를 기입하는 CHUNKTAG 열을 추가한다. 지면 관계상 한국어에는 잘 사용되지 않는 FEATS, DEPS, MISC 열은 제외하고 기술하였다. 기존의 의존구조 말뭉치와 구묶음을 반영한 의존구조 말뭉치의 특징을 비교하면 Table 3.2와 같다.

Table 3.2 The comparison table between original dependency corpus and the one reflected chunking.

	한국어 의존구조 말뭉치	구묶음을 반영한 의존구조 말뭉치
행 단위	어절*	문장 성분
말덩이 표지	×	○
내용어/기능어 분리	×	○
어절 경계	○	×**

* 일부 기호는 분리해서 표현
** 원문과 대조하여 어절 경계 복원 가능

3.3.2 말뭉치 변환 과정 및 알고리즘

이 장에서는 (최용석 & 이공주, 2018)의 말뭉치를 말덩이 기반의 의존구조 말뭉치로 변환할 때 핵심이 되는 변환 과정에 관해 설명하고, 실제 말뭉치 변환에 사용한 알고리즘을 소개한다.

의존구조 말뭉치를 구묶음을 반영한 의존구조 말뭉치로 변환하는 전체 과정은 Figure 3.7과 같다.

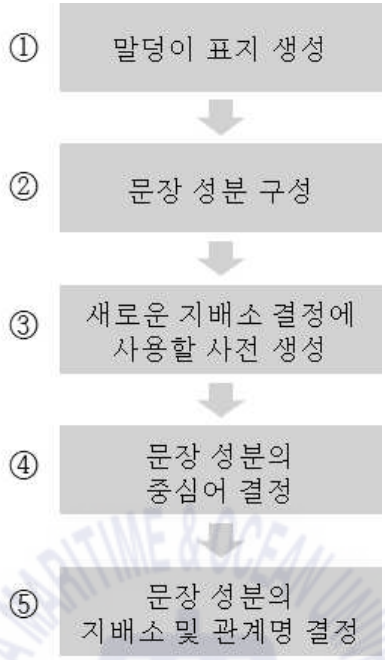

Figure 3.7 A whole process of converting Korean dependency corpus to the one reflected chunking.

변환 과정을 설명하는 데 사용할 예문은 Figure 3.5의 문장인 “프랑스의 세계적인 의상 디자이너 엠마누엘 웅가로가 실내 장식용 직물 디자이너로 나섰다.”이며, 변환 과정을 직관적으로 설명하기 위해 Figure 3.5와 Figure 3.6 중 ID, FORM, XPOSTAG, CHUNKTAG, HEAD 열만을 간략히 Table 3.3과 Table 3.4로 나타내어 변환 방법을 기술한다.

Table 3.3 An example of the original Korean dependency corpus.

ID	FORM	XPOSTAG	HEAD
1	프랑스의	NNP+JKG	4
2	세계적인	NNG+XSN+VCP+ETM	4
3	의상	NNG	4
4	디자이너	NNG	6
5	엠마누엘	NNP	6
6	웅가로가	NNP+JKS	11
7	실내	NNG	8
8	장식용	NNG+XSN	9
9	직물	NNG	10
10	디자이너로	NNG+JKB	11
11	나섰다.	VV+EP+EF+SF	0

Table 3.4 An example of Korean dependency corpus reflected chunking.

ID	FORM		XPOS	CHUNK	HEAD
	cont	func			
1	프랑스	의	NNP+JKG	NX+JMX	3
2	세계 적 이	ㄴ	NNG+XSN +VCP+ETM	CX+ETX	3
3	의상 디자이너 엠마누엘 웅가로	가	NNG+NNG +NNP+NNP +JKS	NX+JKX	5
4	실내 장식 용 직물 디자이너	로	NNG+NNG +XSN+NNG +NNG+JKB	NX+JKX	5
5	나서	었 다 .	VV+EP +EF+SF	PX+EPX +EFX+SYX	0

① 말덩이 표지 생성

말덩이 기반의 의존구조 말뭉치로 변환하기 위해서는 먼저 문장에 대해 구묶음이 수행되어야 한다. 이때, 3.2.1절에서 기술한 순차 표지 부착 모델을 이용하여 구묶음을 수행한다(남궁영 외, 2019). 예문에 대해 구묶음을 수행하면 Figure 3.8과 같이 예측된 말덩이 열을 얻을 수 있다. Figure 3.8은 위에서부터 차례로 원문, 형태소 단위의 문장, 말덩이 표지를 나타낸다.

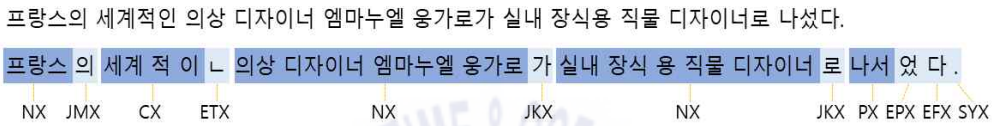

Figure 3.8 The result of chunking.

② 문장 성분 구성

예측된 말덩이 열을 토대로 문장 성분을 구성할 수 있다. 말덩이 기반 의존구조 말뭉치에서 한 행은 문장 성분으로 이루어지며, 문장 성분은 하나의 내용어 말덩이에 0개 이상의 기능어 말덩이로 이루어진다. 따라서 ①에서 얻은 말덩이 표지를 통해 간단히 입력 문장을 문장 성분 단위로 표현할 수 있다.

③ 새로운 지배소 결정에 사용할 사전 생성

문장 성분 단위로 재구성된 노드는 기존과는 다른 ID를 갖게 된다. 따라서 기존의 ID와 새로운 ID의 관계 정보를 저장할 필요가 있다. ②의 결과를 이용해 Table 3.5와 같이 ID 관계 정보를 가진 사전을 만들 수 있다. 또한, Table 3.6과 같이 사전의 값(value)과 키(key)를 반전시킨 역ID사전을 만들면, 기존의 ID 중 문장 성분의 중심어에 해당하는 토큰를 선정했을 때, 해당 토큰의 HEAD(지배소) 정보를 새로운 ID에 전파할 수 있다.

Table 3.5 The ID dictionary.

new_ID (문장 성분)	old_ID (토큰)
1	1
2	2
3	3, 4, 5, 6
4	7, 8, 9, 10
5	11

Table 3.6 The Reversed ID dictionary.

old_ID (토큰)	new_ID (문장 성분)
1	1
2	2
3	3
…	…
11	5

예를 들어, Table 3.3의 1번 토큰(‘프랑스의’)이 변환된 말뭉치에서 어떤 HEAD를 가지게 되는지 기술하면 다음과 같다. Table 3.3에서 1번 ID에 해당하는 HEAD는 4번이다. 4번은 Table 3.6의 역ID사전에서 3번 ID를 값(value)으로 가진다. 즉, HEAD에 해당하는 토큰(‘디자이너’)이 변환된 말뭉치에서 3번 ID의 문장 성분에 있음을 알 수 있다. 따라서, 최종적으로 변환된 말뭉치에서 우리가 찾고자 하는 토큰(‘프랑스의’)이 속해있는 문장 성분의 HEAD 즉, 지배소는 3번 토큰으로 결정할 수 있다.

④ 문장 성분의 중심어 결정

말뭉치 변환 과정에서 핵심이 되는 단계이다. ③에서 예를 든 것과 같이 변환 전과 후가 같은 토큰으로 이루어진 경우, 해당 토큰이 곧 문장 성분의 중심어가 되므로 중심어를 결정할 필요가 없다. 하지만 일반적으로 변환된 말뭉치에서 한 문장 성분은 Table 3.4의 3번 ID처럼 기존 말뭉치의 여러 토큰에 해당하는 경우가 대부분이다. 이럴 경우, 이 중 중심어가 되는 토큰을 선정하여 ③의 예시와 같은 과정을 거치면 해당 문장 성분의 최종 HEAD를 결정할 수 있다.

문장 성분 내의 중심어를 선정하는 규칙은 명료하다. 문장 성분을 이루고 있는 내용어 토큰들 중 그 HEAD가 해당 토큰들 내에 없는 것이 중심어가 된다. 즉, 문장 성분 내에서 의존소를 가지지만 지배소는 가지고 있지 않은 토큰이 해당 문장 성분의 중심어가 된다.

이를 Table 3.4의 3번 문장 성분을 예로 들어 설명하면 다음과 같다. 3번 문장 성분을 이루는 토큰들은 기존 말뭉치에서 [3, 4, 5, 6]의 ID에 해당하며 각각 [4, 6, 6, 11]을 HEAD로 가졌다. 이 HEAD 번호 중 4와 6은 해당 토큰에 이미 존재하지만, 11은 그렇지 않다. 즉, 4와 6을 HEAD로 가지는 [3, 4, 5]번은 이 토큰 리스트 내에서 지배소를 가지지만, 11번을 HEAD로 가지는 6번 토큰은 해당 문장 성분 내에서 다른 토큰들의 수식만 받을 뿐 지배소를 가지고 있지 않다. 따라서 6번 토큰을 이 문장 성분의 중심어로 선정하여 ③의 예시와 같은 과정을 거치게 되면 해당 문장 성분의 새로운 HEAD도 결정할 수 있게 된다. 이는 말덩이로 이루어진 문장 성분이라는 단위가 의미적으로는 물론 구문적으로도 한 덩어리를 이루기에 가능한 방법이다.

⑤ 문장 성분의 지배소 및 관계명 결정

이상의 과정에서 역ID사전을 통해 말덩이 기반으로 변환된 의존구조 말뭉치의 HEAD 즉, 지배소를 결정하였으며, 문장 성분을 이루는 토큰이 여러 개일 경우 중심어를 선정한 뒤 해당 문장 성분의 지배소를 결정하였다. 말뭉치를 구성하는 항목은 HEAD 외에도 UPOSTAG, DEPREL 등이 있다. 이러한 정보들은 각 문장 성분의 중심어가 결정되었을 때, 해당 토큰이 원래 갖고 있던 요소들을 따르면 된다. 즉, Table 3.4의 1번 문장 성분(‘프랑스의’)은 기존 의존구조 말뭉치의 UPOSTAG인 PROPN과 DEPREL인 nmod를 그대로 갖게 된다. Table 3.4의 3번 문장 성분의 경우, 중심어에 해당하는 토큰(‘웅가로’)의 기존 UPOSTAG인 PROPN과 DEPREL인 nsubj을 갖게 된다.

이러한 과정을 통해 Figure 3.5의 의존구조 말뭉치를 변환하면 Figure 3.6과 같은 말덩이 기반의 의존구조 말뭉치를 구축할 수 있다. 이상에서 설명한 바와 같이 구묶음을 반영한 말뭉치로 변환하는 알고리즘을 기술하면 Figure 3.9와 같다.

```
def To_Chunk_Dependency_Corpus(dependency_corpus):
    # 문장 성분 단위로 분리
    toConst = To_Constituent(dependency_corpus)

    # look-up 사전 생성
    idDict = ID_Dictionary(toConst) # {new_ID: old_ID}
    idDict_reversed = ID_Dictionary_Reversed(idDict)
                                    # {old_ID: new_ID}

    # 문장 성분 내의 중심어 선정
    for old_id in idDict[new_ID]:
            # 토큰의 head에 해당하는 id가
            # 문장 성분 내에 없으면 이 토큰을
            # 해당 문장 성분의 중심어(content)로 선정
        if not old_id.HEAD in idDict[new_ID]:
            content_list = Add_to_Content_List(old_id)

    # 선정한 중심어를 토대로 역ID사전을 이용하여
    # 문장 성분의 최종 지배소 및 관계명 결정
    for old_id in content_list:
        new_head = idDict_reversed[old_ID]
        new_relation = old_id.DEPREL
        new_upostag = old_id.UPOSTAG
```

Figure 3.9 The conversion algorithm from the original Korean dependency corpus to Korean dependency corpus reflected chunking.

3.3.3 말뭉치 변환 결과 분석

구묶음을 반영한 대량의 의존구조 말뭉치를 구축하기 위해, 3.3.2에서 기술한 알고리즘을 기존의 의존구조 말뭉치에 적용하였다. 기존의 말뭉치는 세종 구문분석 말뭉치를 (최용석 & 이공주, 2018)의 의존구조 변환 도

구를 이용하여 생성한 것으로 총 62,345문장으로 이루어져 있다. 이를 기반으로 본 논문의 변환 알고리즘을 적용하여 말덩이 기반 의존구조 말뭉치를 생성한 뒤 띄어쓰기 및 형태소분석 오류가 전파된 13,053문장을 제외하고 정제된 문장들을 모은 결과, 생성된 말뭉치는 총 49,292문장이었다. Table 3.7은 기존의 말뭉치와 말덩이 기반 의존구조 말뭉치를 정량적으로 비교한 내용이다. 동등한 비교를 위해 기존의 말뭉치 중 정제된 말뭉치만을 집계에 이용했다.

Table 3.7 The statistics of Korean dependency corpus, the refined version and Korean dependency corpus reflected chunking.

	의존구조 말뭉치 (원본)	**의존구조 말뭉치 (정제 후)**	**구묶음을 반영한 의존구조 말뭉치**
문장 수	62,345	**49,292**	**49,292**
표지 종류 수	45	45	18
형태소 수	1,566,560	**1,054,859**	**1,054,859**
말덩이 수	•	•	**804,854**
행 단위	어절*	어절*	문장 성분
전체 행 수	713,238	**526,378**	**377,301**
의존 관계 태그 수	50	50	50

* 일부 기호는 분리해서 표현

전체 문장 수와 형태소 수는 각각 49,292개와 1,054,859개로 두 말뭉치 모두 동일하며, 구묶음을 반영한 의존구조 말뭉치에서는 그 기본 단위인 말덩이 수가 804,854개로 집계되었다. 비교 항목 중 '전체 행 수'는 구문분석의 입력이 되는 한 단위인 노드의 개수와 일치하며, 이는 기존 말뭉치에서는 어절 단위가 되고 구묶음을 반영한 말뭉치에서는 문장 성분 단위가 된다. 따라서 구묶음을 반영한 의존구조 말뭉치의 경우 기존과 비교했을 때 같은 문장 수에 대해 입력 노드 수가 현저히 줄어드는 것을 확인할 수 있다.

3.4 구묶음을 반영한 한국어 의존구조 분석

이 절에서는 3.3절까지 기술한 내용을 바탕으로 구묶음을 반영한 한국어 구문분석에 사용한 모델에 관해 설명한다. 3.4.1절에서는 본 논문에서 구문분석 모델로 사용하는 스택-포인터 네트워크 모델(Ma *et al.*, 2018)에 대해 설명하고, 구묶음을 반영한 한국어 구문분석에 적용한다. 3.4.2절에서는 모델의 입력이 될 한국어 문장 성분의 표현 방법에 대해 기술한다.

3.4.1 의존구조 분석 모델

스택-포인터 네트워크 모델은 의존구조 분석에 적합하도록 포인터 네트워크를 확장한 모델이다. 이는 포인터 네트워크와 내부 스택을 결합한 형태로 인코더와 디코더로 구성되어 있으며, 인코더에서 전체 문장을 함축한 뒤 디코더에서 깊이 우선 방식으로 root에서부터 의존구조 트리를 구성해 나가는 하향식(top-down) 모델이다. 이때 하나의 중심어는 여러 의존소를 가질 수 있으므로 내부 스택이 의존소를 탐색할 지배소의 순서를 저장하여 하나의 지배소가 여러 번 의존소를 찾을 수 있도록 한다. 또한, 각 단계에서 찾은 의존소가 유일한 정답임을 보장하기 위해 의존소를 찾아가는 순서를 사전 정의하게 된다. 의존소 순회 방법은 inside-out, left-to-right, right-to-left가 있으며, 각각 중심어의 왼쪽 기준으로 가장 가까운 의존소부터, 중심어의 왼쪽 기준으로 가장 먼 의존소부터, 중심어의 오른쪽 기준으로 가장 먼 의존소부터 방문하는 방법이다(최용석 & 이공주, 2019). 스택-포인터 네트워크는 각 단계에서 스택의 최상위 노드의 의존소가 될 자식 노드를 선택하게 된다. 이러한 방식은 구문분석을 할 때 문장의 일부가 아니라 현재 단계 이전까지 형성된 부분 트리를 고려하여 의존관계를 결정할 수 있게 한다. 따라서 기존의 전이 기반 구문분석에서 의존 관계를 결정할 때 문장 일부만 보는 문제를 완화할 수 있다.

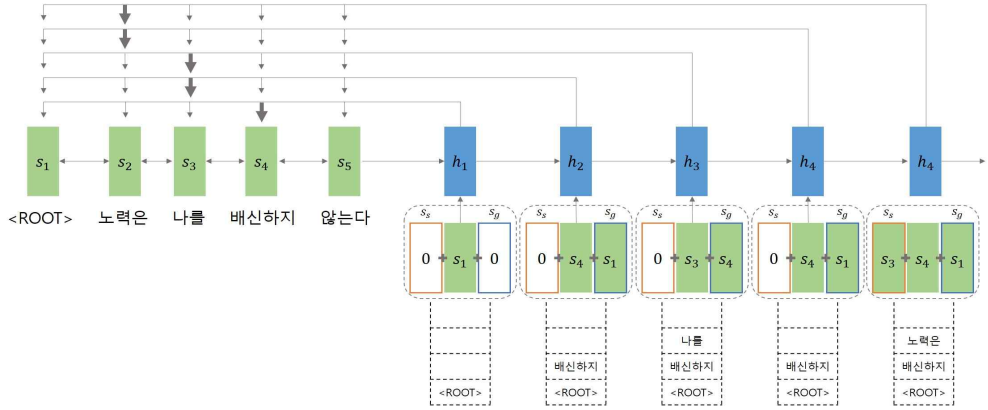

Figure 3.10 The structure of stack-pointer network for Korean dependency parsing.

스택-포인터 네트워크는 Figure 3.10과 같이 인코더와 디코더로 이루어진 sequence-to-sequence 모델의 형태로 이루어져 있다. 이는 입력열의 위치에 해당하는 단어를 통해 출력열을 이룰 단어의 조건부 확률을 학습할 수 있는 신경망 모델이다. 학습할 때 인코더에서는 순환 신경망을 통해 입력 단어에 대한 은닉 표상(encoder hidden state; s_i)을 생성한다. 내부 스택은 root가 있는 상태로 초기화된다. 디코더는 매 단계 t 마다 현재 스택의 최상위 노드에 해당하는 입력 표상 벡터를 받아 디코더의 은닉 표상(decoder hidden state; h_t)을 생성한다. 이를 통해 식 3.1과 같은 방식으로 주의 집중 벡터(a^t)를 계산한다. 이때, 주의 집중 점수(e^t)를 계산하는 $score$함수는 h_t와 s_i 간의 주의 집중 정도를 잘 표현할 수 있는 방식이면 어떤 함수든 가능하다(Luong *et al.*, 2015). 해당 논문에서는 biaffine attention 방법을 이용하였다.

$$e^t = score(h_t, s_i) \tag{3.1}$$
$$a^t = softmax(e^t)$$

포인터 네트워크 기반의 구문분석기는 이와 같은 방식으로 계산된 주의 집중 점수를 통해 의존소가 될 단어의 위치 c를 출력하며, 이를 통해 c의 위치에 있는 단어 w_c를 지배소 단어인 w_h의 의존소로 결정하면서 두 단어 w_h와 w_c사이의 의존 관계를 결정하게 된다. 그 뒤 w_c를 스택에 삽입한 후 다음 단계로 넘어간다. 만일 w_h가 자기 자신의 위치를 출력하게 되어 $c = h$가 되면 w_h의 의존소를 모두 찾았음을 뜻하며, w_h를 스택에서 제거하고 다음 단계로 넘어간다.

평가 시에는 입력 문장의 단어가 빠짐없이 포함된 완전한 구문분석 트리를 생성할 수 있어야 한다. 이를 위해 디코딩 단계에서는 구문분석 트리를 구성할 단어들의 후보로 이루어진 열을 갖고 있다. 디코딩의 매 단계마다 현재 지배소 단어의 의존소를 선택하며, 선택된 단어는 후보 단어 열에서 제거됨으로써 다른 단어의 의존소가 되는 경우를 방지하게 된다.

본 논문에서는 구묶음을 반영한 한국어 구문분석을 위해 기존의 스택-포인터 네트워크를 이에 맞게 변형하여 적용한다. 한국어 구문분석에 스택-포인터 네트워크를 사용한 기존의 연구들(차다은 외, 2018; 최용석 & 이공주, 2018; 홍승연 외, 2018)은 Figure 3.10에서처럼 네트워크의 입력 단위가 어절 단위이다. 이와 달리 한국어 구문분석에 구묶음을 반영하면 네트워크의 입력 단위는 3.3.1절에서 설명한 바와 같이 문장 성분(sentence component)이 된다. 이를 바탕으로 스택-포인터 네트워크를 이용해 구문분석을 수행하면 Figure 3.11과 같이 표현할 수 있으며, Figure 3.10과 비교했을 때 입력 단어의 수가 줄어든 것을 확인할 수 있다.

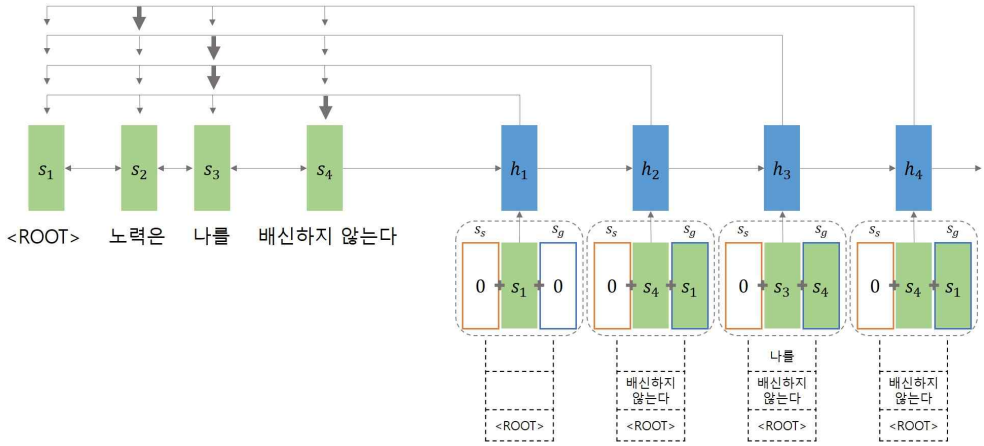

Figure 3.11 The structure of stack-pointer network for Korean dependency parsing with chunking.

3.4.2 입력 데이터 표상 구조

구묶음을 반영한 한국어 의존구조 분석을 하기 위해서 본 논문에서는 문장 성분을 구문분석의 단위로 이용한다. 문장 성분은 내용어 말덩이와 기능어 말덩이로 이루어져 있다. 내용어 말덩이는 의미적으로 문장 성분의 중심어가 되며, 기능어 말덩이는 해당 문장 성분이 문장 내에서 가지는 문법적인 역할을 결정짓도록 한다. 이러한 특징을 살려 구문분석에 반영하기 위해 본 논문에서는 내용어 말덩이와 기능어 말덩이의 표상을 각각 나타내 결합하는(concatenate) 방식으로 문장 성분을 표현한다.

이때, 각각의 말덩이는 여러 형태소가 결합된 형태이기 때문에 말덩이 전체를 하나의 벡터로 표현하기보다 형태소 표상들의 집합으로 표현하면 미등록어 문제를 일부 해소할 수 있다. 이를 위해 말덩이를 이루고 있는 형태소 표상들에 합성곱 신경망(Convolutional Neural Network; CNN)을 적용하여 하나의 말덩이를 표현한다.

기본적으로 각 형태소 표상은 형태소와 품사 정보를 이용하여 표현하며, 이때 발생하는 미등록어 문제를 해소하기 위해 본 논문에서는 문자

단위 표상도 함께 결합하여 사용한다. 이때도 말덩이 표상을 생성할 때와 마찬가지로 합성곱 신경망을 이용하여 하나의 형태소 표상을 나타낸다.

이상에서 설명한 구문분석 입력 단위의 표상 과정을 나타내면 Figure 3.12와 같다.

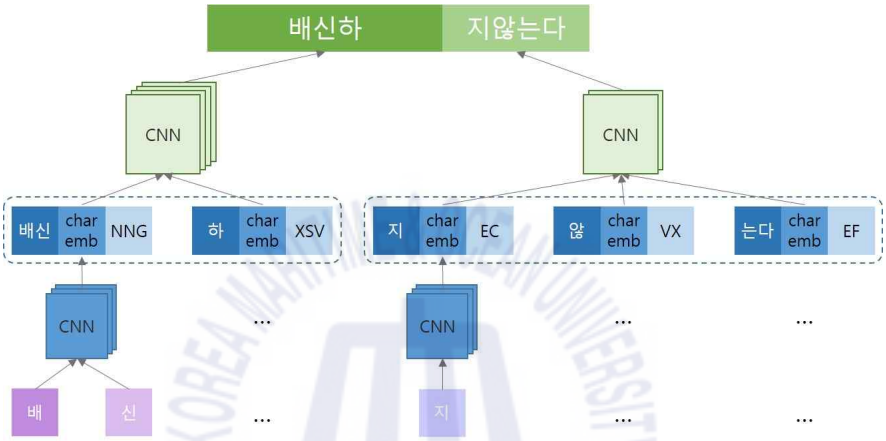

Figure 3.12 An example of the representation of a sentence component as an input for Korean dependency parsing reflected chunking.

제 4 장 실험 및 평가

이 장에서는 본 논문에서 제안한 방법에 대한 실험을 기술한다. 4.1절에서는 심층학습을 이용한 한국어 구묶음에 관한 실험을 다루며, 4.2절에서는 구묶음을 반영한 한국어 의존구조분석에 관한 실험을 다룬다. 각 절은 실험에 사용한 데이터에 대한 통계와 모델 매개변수 등 실험 환경을 소개하고 평가 방법 및 결과 분석을 하는 순으로 기술한다.

4.1 한국어 구묶음

이 절에서는 3.2절에서 소개한 심층학습을 이용한 한국어 구묶음에 대한 실험을 진행한다. 모델은 순차 표지 부착에 좋은 성능을 보이는 Bi-LSTM/CRFs 모델을 사용하였으며, 한국어 문장 내의 모든 구성 요소에 대해 구묶음을 수행한다.

4.1.1 실험 환경

(1) 실험 데이터

실험에 사용된 말뭉치는 3.2.1절에 기술한 구묶음 말뭉치이며, 이 중 13,113 문장을 실험에 사용하였다. 학습에 사용된 문장은 10,490개이고 검증에 사용된 문장은 1,311개이며, 평가에 사용된 문장은 1,312개이다. 실험에 사용한 문장 및 형태소 개수는 Table 4.1과 같다.

Table 4.1 The number of sentences and morphemes used in chunking.

(단위: 개)

	문장 수	형태소 수
학습말뭉치	10,490	163,641
검증말뭉치	1,311	20,191
평가말뭉치	1,312	20,416
전체말뭉치	13,113	204,248

(2) 모델 파라미터

실험에 사용한 모델의 활성 함수는 ReLU(Nair & Hinton, 2010)를 이용하고 학습기로는 RMSprop(Hinton *et al.*, 2012)을 이용하며, 모델의 각종 hyper-parameter들은 실험적으로 조절해가며 평가에 사용하였다.

4.1.2 실험 결과

(1) 평가 방법

평가 방법은 구묶음과 마찬가지로 개체의 경계를 찾고 해당 개체에 부착된 표지의 적합성을 판별하는 개체명 인식 시스템 평가에 이용되는 지표를 사용하였다. 개체명 인식 시스템을 평가하는 방법에는 대표적으로 MUC에 사용된 방법(Chinchor & Sundheim, 1993; Chinchor & Robinson, 1998)과 이를 기반으로 평가 방식에 따라 세분화하여 측정한 SemEval에 사용된 방법(Segura-Bedmar *et al.*, 2013)이 있다. 본 논문에서는 이러한 방법을 구묶음 시스템을 평가하는 데 이용하였으며, Table 4.2에 설명한 네 가지 경우에 대해 정밀도(precision), 재현율(recall), F1-점수(F1-score)를 각각 측정하였다.

Table 4.2 Evaluation schemes for performance evaluation.

평가 방법	설 명
경계/표지 일치 (strict)	시스템이 예측한 말덩이의 **경계 및 표지**가 모두 정답과 일치하는 경우
경계 일치 (exact)	표지의 일치 여부와 관계없이 말덩이의 **경계**를 잘 인식한 경우
부분 경계 일치 (partial)	표지의 일치 여부와 관계없이 시스템이 예측한 말덩이의 경계와 정답의 **경계가 일부** 겹치는 경우
표지 일치 (type)	말덩이의 경계 일치 여부와 관계없이 시스템이 예측한 **표지**가 일치하는 경우

(2) 평가 결과

Table 4.2의 평가 방법을 바탕으로 각 경우에 따른 구묶음 시스템의 성능을 측정한 결과 말덩이의 경계와 표지가 모두 일치하는 경우의 F1-점수는 97.02였다. 전체 평가 방식에 대한 실험 결과는 Table 4.3과 같다.

Table 4.3 The results according to the evaluation schemes.

	경계/표지	경계	부분경계	표지
정밀도	**97.26**	97.69	97.69	97.54
재현율	**96.78**	97.21	97.21	97.07
F1-점수	**97.02**	97.45	97.45	97.30

4.2 구묶음을 반영한 한국어 의존구조 분석

이 절에서는 3.4절에서 소개한 구묶음을 반영한 한국어 구문분석에 대한 실험을 진행한다. 실험은 기존의 구문분석과 구묶음을 반영한 구문분석을 각각 진행하였으며, 그 결과를 비교한다. 두 방식의 구문분석 모두 스택-포인터 네트워크를 이용하여 동등한 방식의 의존구조 분석을 진행하였으며, 구묶음을 반영한 구문분석의 경우 모델의 입력 표상을 3.4절에서 언급한 바와 같이 입력 단위에 맞게 변화를 주었다.

4.2.1 실험 환경

(1) 실험 데이터

기존의 구문분석을 위한 실험 데이터는 3.3.1절에서 언급한 바와 같이 세종 구 구조 구문 트리(NIKL, 2010)를 (최용석 & 이공주, 2018)의 중심어 전파 규칙에 따라 의존구조로 변환한 데이터를 사용하였다. 구묶음을 반영한 구문분석을 위한 실험 데이터는 동등한 데이터를 가지고 3.3의 방법을 통해 문장 성분 단위로 변환한 의존구조 말뭉치를 사용하였다.

실험에 사용한 구묶음 말뭉치는 형태소 오류가 있거나 입력 단위 수가 1인 문장 등을 제외한 총 41,643문장을 사용하였다. 이 중 학습말뭉치는 33,313문장이고, 검증말뭉치와 평가말뭉치는 각각 4,165문장을 이용하였다. 입력 성분 수는 문장 성분 단위로, 학습말뭉치, 검증말뭉치, 평가말뭉치에 각각 269,793, 47,641, 42,073개로 집계되었다. 이를 정리하면 Table 4.4와 같다.

Table 4.4 The metadata of Korean dependency corpus reflected chunking.

	문장 수 (문장)	입력 성분 수(개)
학습말뭉치	33,313	269,793
검증말뭉치	4,165	47,641
평가말뭉치	4,165	42,073
전체말뭉치	41,643	359,507

기존 구문분석과 구묶음을 반영한 구문분석에 사용한 데이터의 전체 문장 수는 같으며, 형태소 수, 어절 수, 문장 성분 수 등을 비교하면 Table 4.5와 같다.

Table 4.5 The statistics of the Korean dependency corpus and Korean dependency corpus reflected chunking.

(단위: 개)

	Chunk data set	Sejong data set
전체 문장 수	41,643	
입력 성분 수	359,507 (문장 성분)	460,052 (어절)
평가데이터 입력 성분 수	42,073 (문장 성분)	54,146 (어절)
문장당 평균 입력 단위 수	**8.63**	**11.04**
전체 형태소 수	1,009,707	
문장당 평균 형태소 수	24.24	

이때, 문장당 평균 입력 단위 수를 보면 구묶음을 반영한 경우는 8.63이고 그렇지 않은 경우는 11.04이다. 일반적으로 자연언어처리에서 전이 기반 구문분석의 복잡도는 $O(N)$이고, 그래프 기반 구문분석의 복잡도는 $O(N^3)$이다. 이때 N은 입력 단위의 수이다(Nivre, 2008; Nivre & McDonald, 2008). 따라서 구묶음을 반영한 경우가 분석에 있어 좀 더 빠른 속도를 보인다.

(2) 모델 파라미터

기존 구문분석을 수행하기 위해 사용된 모델 파라미터는 기본적인 스택-포인터 네트워크의 파라미터를 유지하며 한국어의 특성에 맞게 변형한 모델(최용석 & 이공주, 2019)을 본 논문의 실험 상황에 맞게 적용하였다(Table 4.6). 형태소 표상은 세종 말뭉치를 이용하여 GloVe(Pennington *et al.*, 2014)를 통해 표현한 값을 사용하였으며, 음절과 품사 표현은 사전 학습 없이 무작위 정수로 초기화하였다.

구묶음을 반영한 구문분석에 사용된 모델 파라미터는 기본적으로 Table 4.6과 같으며, 문장 성분 표상을 위해 내용어 말덩이와 기능어 말덩이의 결합 비율이 2:1이 될 수 있도록 합성곱 신경망의 필터 수를 각각 기존의 2/3, 1/3으로 조절하였다.

Table 4.6 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

4.2.2 실험 결과

(1) 평가 방법

평가 척도는 의존구조 분석을 평가하기 위해 주로 사용되는 UAS(Unlabeled Attachment Score)와 LAS(Labeled Attachment Score)를 이용하여 측정하였다. 이때, UAS는 전체 입력 성분 중 의존 관계인 HEAD를

정확히 찾은 비율로, 식 (4.1)과 같이 나타낼 수 있다.

$$\text{UAS} = \frac{n(correct\ HEAD)}{n(input\ \ unit)} \tag{4.1}$$

LAS는 전체 입력 성분 중 HEAD는 물론이고 의존 관계명인 RELATION까지 모두 정확히 찾은 비율을 뜻한다. LAS는 보통 식 (4.2)와 같이 구할 수 있다.

$$\text{LAS} = \frac{n(correct\ \ HEAD\&RELATION)}{n(input\ \ unit)} \tag{4.2}$$

평가 단위는 크게 두 가지가 있는데, 입력 성분 단위(word-based)와 문장 단위(sentence-based) 방식이 있다. 입력 성분 단위로 평가하는 방법은 micro-average라고도 불리며, 단순히 전체 문장 중에 의존 관계를 잘 찾은 입력 성분 수를 측정하는 방법이다. 문장 단위로 평가하는 방법은 macro-average라고 불리며, 문장마다 의존 관계를 잘 찾은 입력 성분의 비율을 측정한 다음 이를 전체 문장 수로 나누는 방법이다. 본 논문에서는 UAS와 LAS마다 각각 입력 성분 단위와 문장 단위의 측정방식을 통해 구문분석의 성능을 평가하였다.

(2) 평가 결과

4.2.1절에서 설명한 평가 척도와 단위를 기준으로 구묶음을 반영한 경우와 그렇지 않은 경우를 단순히 측정하여 비교하면 Table 4.7와 같다.

Table 4.7 The evaluation results.

Evaluation unit	Metric	Chunk	Original
word-based (micro-average)	UAS	83.06%	82.98%
	LAS	80.40%	80.45%
sentence-based (macro-average)	UAS	86.45%	85.78%
	LAS	83.87%	83.23%

이는 각각 그 입력 단위가 구묶음의 경우 문장 성분, 기존의 경우 어절로 비교 기준이 다르다. 따라서 입력 단위를 통일하여 비교할 필요가 있다. 보통 문장 성분은 여러 어절로 이루어지므로, 어절을 기준으로 하여 비교를 진행한다. 이에 대한 분석은 다음 절에서 기술한다.

(3) 결과 분석

한 문장 성분은 하나의 지배소를 가리키며, 문장 성분 내의 어절들은 선형적이므로 중심어를 제외한 나머지 요소들은 모두 중심어를 수식하는 형태가 된다. 즉, 문장 성분의 중심어만 지배소를 갖게 되며, 나머지 어절들은 모두 이 중심어를 지배소로 갖게 되므로, 이미 중심어를 찾은 것과 마찬가지이다. 따라서 구묶음을 반영한 구문분석의 경우, 전체 어절 중 지배소를 잘못 찾은 경우를 제하면 어절 단위 평가를 할 수 있다.

예를 들어 설명하면 다음과 같다. Table 4.8의 2번 행은 Table 4.9의 2~4번 행까지에 해당하는 부분이다. 이를 어절 단위로 변환하면 Table 4.10과 같다. 이때, '놓고', 와 '나면'은 2번 행의 '해'와 함께 같은 문장 성분을 이루고 있으며, 모두 '해'를 중심어로 가지게 되므로 이미 지배소를 잘 찾은 것이나 다름없다. 따라서 Table 4.9의 2~4행이 지배소를 잘못 찾을 경우 그 횟수가 그대로 반영되지만, Table 4.8에서 2번 행이 지배소를 잘못 찾은 경우 그 횟수는 1회에 그친다.

Table 4.8 An example of a dependency sentence reflected chunking.

ID	FORM(cont)	FORM(func)	HEAD	DEPREL
1	김장	만	2	obj
2	하	아_놓_고_나_면	4	advcl
3	이미	-	4	advmod
4	겨울_이	다_.	0	root

Table 4.9 An example of an original dependency sentnece.

ID	FORM	HEAD	DEPREL
1	김장만	2	obj
2	해	6	advcl
3	놓고	2	aux
4	나면	3	aux
5	이미	6	advmod
6	겨울이다.	0	root

Table 4.10 An example of a converted sentence from Table 4.8.

ID	FORM	HEAD	DEPREL
1	김장만	2	obj
2	해	4	advcl
-	놓고	-	advcl
-	나면	-	advcl
3	이미	4	advmod
4	겨울이다.	0	root

이상에서 설명한 바와 같이 어절을 기준으로 하여 두 방법론을 비교할 경우, 오답 개수를 측정하여 비교하면 간단하다. 식 (4.3)은 어절 단위 정확도를 측정하기 위한 식이다.

$$\frac{n(correct\,word)}{n(word)} = \frac{n(word) - n(incorrect\,word)}{n(word)} \tag{4.3}$$

이를 이용하여 기존의 경우 UAS와 LAS를 구하면 식 (4.4)와 같으며, 구묶음을 반영한 경우는 식 (4.5)와 같다.

$$\begin{aligned} \text{UAS:} \quad & \frac{54{,}146 - 7{,}323}{54{,}146} \times 100 = 86.48\% \\ \text{LAS:} \quad & \frac{54{,}146 - 8{,}362}{54{,}146} \times 100 = 84.56\% \end{aligned} \tag{4.4}$$

$$\begin{aligned} \text{UAS:} \quad & \frac{54{,}146 - 9{,}218}{54{,}146} \times 100 = 82.98\% \\ \text{LAS:} \quad & \frac{54{,}146 - 10{,}583}{54{,}146} \times 100 = 80.45\% \end{aligned} \tag{4.5}$$

따라서 구묶음을 반영한 경우가 기존보다 UAS 기준 3.5%p 상승했으며, LAS 기준 4.11%p가 증가한 것을 알 수 있다.

제 5 장 결론 및 향후 연구

본 논문에서는 한국어 구문분석을 효율적으로 수행하기 위한 방법론으로 구묶음을 반영한 한국어 의존구조 분석을 제안하였다. 구묶음은 입력 문장을 말덩이 단위로 인식하고 표지를 부여하는 과정을 말하며, 이를 통해 형태소분석된 입력 문장을 의미적, 문법적으로 하나의 역할을 수행하는 말덩이들로 표현할 수 있다. 말덩이는 내용어 말덩이와 기능어 말덩이로 이루어지는데, 한국어에서는 이들의 조합으로 하나의 문장 성분을 표현할 수 있다. 구묶음을 반영한 한국어 구문분석에서는 이러한 문장 성분이 하나의 입력 단위가 된다.

기존에 연구된 한국어 구문분석은 입력 단위로 형태소 또는 어절 단위를 사용하였다. 이렇게 하면 입력열을 이루는 노드 수가 형태소 수 또는 띄어쓰기 수만큼이 되며, 보조 용언, 조사 상당 어구와 같이 한국어 문장 내에 다른 보조적인 역할을 하는 요소를 포함한 모든 노드에 대해 통사적 관계를 결정지어야 한다. 이로 인해 구문분석의 복잡도가 높아지며 분석 결과에 영향을 미치는 요인으로 작용한다. 하지만 구묶음을 수행한 뒤 구문분석을 진행하면 구문분석의 입력 단위가 문장 성분이 된다. 이는 여러 형태소 또는 어절이 하나의 문장 성분을 이루게 됨을 뜻하므로 결과적으로 구문분석의 입력 노드 수를 감소시키는 역할을 한다. 이로 인해 구문분석의 복잡도가 줄어들고 정확도를 향상할 수 있게 된다.

따라서 본 논문에서는 구묶음을 반영한 한국어 의존구조 분석을 수행하여 기존의 방식보다 향상된 결과를 보여주었다. 본격적인 구문분석을 수행하기 위해 한국어에 있어 말덩이에 대한 정의를 내리고 이에 입각하여 구축된 구묶음 말뭉치를 이용하여 한국어 구묶음을 수행하였다. 구묶음을

수행한 결과를 토대로 기존의 세종 말뭉치에서 구묶음을 반영한 의존구조 말뭉치로 변환하는 알고리즘을 통해 다량의 구묶음 기반의 의존구조 말뭉치를 구축하였으며, 이를 통해 의존구조 분석을 진행하였다. 구문분석 모델로는 스택-포인터 네트워크를 이용하며, 41,643문장에 대해 어절 단위로 측정하였을 경우 UAS 기준 86.48%, LAS기준 84.56%의 결과를 보였다. 이는 기존의 방식으로 구문분석을 수행하였을 때의 결과인 UAS 82.98%, LAS 80.45%보다 각각 약 3.5%p, 4.11%p의 성능 향상을 보여주었다.

구묶음은 한국어 구문분석을 문법적, 의미적 관점에서 좀 더 효율적으로 수행할 수 있게 해 준다. 이러한 구묶음과 관련된 연구가 지속적으로 수행되기 위해 본 논문에서 사용한 구문분석기 외에 다양한 종류의 분석기를 통해 결과를 비교하고 분석할 수 있을 것이다. 또한, 한국어 구문분석에 있어 내용어와 기능어의 비중을 달리해가며 결과를 분석해 보는 것도 유효할 것이다. 그리고 구묶음을 반영한 구문분석의 원활한 수행을 위하여 기반 기술인 부분 구문분석기의 오류 분석 및 성능 향상이 지속해서 이루어져야 할 것이며, 구묶음을 반영한 말뭉치의 효용성에 관한 연구도 다각도에서 검증되면 한국어 구문분석에 더욱 이바지하는 바가 클 것으로 생각한다.

참고문헌

Abney, S. P. (1991). “Parsing by chunks”, *Principle-based Parsing, eds*. Berwick, R. Abney, S. and Tenny, C., Kluwer Academic Publishers.

Abney, S. P. (1995). “Chunk and dependencies: Bringing processing evidence to bear on syntax”, *Computational Linguistics and the Foundations of Linguistic Theory,* pp. 145-164.

Abney, S. P. (1996). “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.

Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov S. and Collins, M. (2016). “Globally normalized transition-based neural networks”, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 2422-2452.

Argamon, S., Dagan, I. and Krymolowski, Y. (1998). “A memory-based approach to learning shallow natural language patterns”, *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics*, pp. 67-73.

Bourigault, D. (1992). “Surface grammatical analysis for the extraction of terminological noun phrases”, *Proceedings of the 15th International Conference on Computational Linguistics*, pp. 977-981.

Chen, D. and Manning C. (2014). “A fast and accurate dependency parser using neural networks”, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 740-750.

Chinchor, N. and Sundheim, B. (1993). “MUC-5 evaluation metrics”, *Proceedings of the 5th Message Understanding Conference*, pp. 69-78.

Chinchor N. and Robinson, P. (1998). “Appendix E: MUC-7 named entity task definition (version 3.5)”, *Proceedings of the 7th Message Understanding Conference*, https://www.aclweb.org/anthology/M98-1028 (accessed 2019.05.03.)

Choi, J. D. and Palmer, M. (2011). “Statistical dependency parsing in Korean: From corpus generation to automatic parsing”, *Proceedings of the 2nd Workshop on Statistical Parsing of Morphologically-Rich Languages*, pp. 1-11.

Choi, K., Han, Y., Han, Y. and Kwon. O. (1994). “KAIST tree bank project for Korean: Present and future development”, *Proceedings of the International Workshop on Sharable Natural Language Resources*, pp. 7-14.

Chu, Y. J. and Liu, T. H. (1965). “On the shortest arborescence of a directed graph”, *Science Sinica*, vol. 14, pp. 1396−1400.

Chun, J,. Han, N., Hwang, J. D. and Choi, J. D. (2018). “Building universal dependency treebanks in Korean”, *Proceedings of the 11th International Conference on Language Resources and Evaluation*, pp. 2194-2202.

Collins, M. (2002). “Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms”, *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*. pp. 1-8.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K. and Kuksa, P. (2011). “Natural language processing (almost) from scratch”, *Journal of Machine Learning Research,* vol. 12, pp. 2493-2537

Daelemans, W., Buchholz, S. and Veenstra, J. (1999). “Memory-based shallow parsing”, *Proceedings of the Conference on Computational Natural Language Learning*, pp. 53-60.

de Marneffe, M. C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J. and Manning, C. D. (2014). “Universal Stanford dependencies: A cross-linguistic typology”, *Proceedings of the Language Resources and*

Evaluation Conference, vol. 14, pp. 4585-4592.

Devlin, J., Chang, M., Lee, K. and Toutanova, K. (2018). "BERT: Pre-training of deep bidirectional transformers for language understanding", *arXiv*:1810.04805

Dozat, T. and Manning, C. (2017). "Deep biaffine attention for neural dependency parsing", *The International Conference on Learning Representations.*

Dyer, C., Ballesteros, M., Ling, W., Matthews, A. and Smith, N. A. (2015). "Transition-based dependency parsing with stack long short-term memory", *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pp. 334-343.

Edmonds, J. (1967). "Optimum branchings", *Journal of Research of the National Bureau of Standards*, vol. 71B, no. 4, pp. 233–240.

Eisner, J. (1996). "Three new probabilistic models for dependency parsing: An exploration", *Proceedings of The 16th International Conference on Computational Linguistics*. pp. 340-345.

Fernández-González, D. and Gómez-Rodríguez, C. (2019). "Left-to-right dependency parsing with pointer networks", *arXiv*:1903.08445.

Gee, J. P. and Grosjean, F. (1983). "Performance structures: A psycholiguistic and linguistic appraisal", *Cognitive Psychology*. vol. 15, no. 4, pp. 411-458.

Grefenstette, G. (1996). "Light parsing as finite state filtering", *Proceedings of the Workshop on Extended Finite State Models of Language*. pp. 20-25.

Han, C., Han, N., Ko, E., Palmer, M. and Yi, H. (2002). "Penn Korean treebank: Development and evaluation", *Proceedings of the 16th Pacific Asia Conference on Language, Information and Computation.* pp. 69-78.

Hinton, G., Srivastava, N. and Swersky, K. (2012). Slides of Neural Networks for Machine Learning – Lecutre 6e. https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

Huang, Z., Xu, W. and Yu, K. (2015). "Bidirectional LSTM-CRF models for

sequence tagging", *arXiv*:1508.01991.

Hwang, Y., Chung, H., Park, S., Kwak, Y. and Rim, H. (2002). "Improving the performance of Korean text chunking by machine learning approaches based on feature set selection", *Journal of Korea Information Science Society: Software and Applications,* vol. 29, no. 9/10, pp. 654-668.

Johansson R. and Nugues, P. (2007). "Extended constituent-to-dependency conversion for English", *Proceedings of the 16th Nordic Conference of Computational Linguistics*, pp. 105-112.

Jurafsky, D. and Martin. J. H. (2018). "Dependency Parsing", *Speech and Language Processing, 3rd ed. draft,* Prentice Hall Publishers.

Kim, J. (2000). "Partial parsing", *Korea Information Processing Society Review*, vol. 7, no. 6, pp. 83-96.

Kiperwasser, E. and Y. Goldberg (2016). "Simple and accurate dependency parsing using bidirectional LSTM feature representations", *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 313-327.

Kübler, S., McDonald R. and Nivre. J. (2009). *Dependency Parsing*. Morgan & Claypool Publishers.

Kuhlmann, M., Gomez-Rodrıguez, C. and Satta, G. (2011). "Dynamic programming algorithms for transition-based dependency parsers", *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp 673–682.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K. and Dyer, C. (2016). "Neural architectures for named entity recognition", *arXiv*:1603.01360v3.

Lee, K. and Kim, J. (2003). "Implementing Korean partial parser based on rules", *Korea Information Processing Society Transactions: Part B*, vol 10, no. 4, pp. 389-396.

Li, Z., Cai, J., He, S. and Zhao, H. (2018). "Seq2seq dependency parsing", *Proceedings of the 27th International Conference on Computational Linguistics*,

pp. 3203–3214

Loung, T., Pham, H. and Manning C. D. (2015). “Effective approaches to attention-based neural machine translation”, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412-1421.

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

Marcus, M. P., Santorini, B and Marcinkiewicz, M. A. (1993). “Building a large annotated corpus of English: The Penn treebank”, Computational Linguistics, vol. 19, no. 2, pp. 313-330.

McDonald, R., Crammer, K. and Pereira, F. (2005a). “Online large-margin training of dependency parsers”, *Proceedings of 43rd Annual Meeting of the Association for Computational Linguistics*, pp. 91-98.

McDonald, R., Pereira, F., Ribarov, K., and Hajic, J. (2005b). “Non-projective dependency parsing using spanning tree algorithms”, *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pp. 523-530.

McDonald, R., Nivre, J., Quirmbach-Brundage, Y., Goldberg, Y., Das, D., Ganchev, K., Hall, K., Petrov, S., Zhang, H., Täckström, O., Bedini, C., Castelló, N. B. and Lee, J. (2013). “Universal dependency annotation for multilingual parsing”, *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics,* pp. 92-97.

Nair, V. and Hinton, G. (2010). “Rectified linear units improve restricted boltzmann machines”, *Proceedings of the 27th International Conference on Machine Learning*. pp. 807-814.

NIKL, The National Institute of the Korean Language, (2010). “21st century sejong project”, 2010

Nivre, J. (2003). “An efficient algorithm for projective dependency parsing”, *Proceedings of the 8th International Workshop on Parsing Technologies*, pp. 149-160.

Nivre, J. (2004). “Incrementality in deterministic dependency parsing”, *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*. pp. 50-57.

Nivre, J. (2008). “Algorithms for deterministic incremental dependency parsing”, *Journal of Computational Liguistics*, vol. 34, no. 4, pp. 513-553.

Nivre, J. and McDonald, R. (2008). “Integrating Graph-Based and Transition-Based Dependency Parsers”, *Proceedings of Association for Computational Linguistics*, pp. 950-958.

Noh, K., Kim, C., Choi, M., Yoon, H. and Kim, J. (2018). “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. 42, no. 7, pp. 580-586.

Park, E. and Ra, D. (2006). “Processing dependent nouns based on chunking for Korean syntactic analysis”, *Korean Journal of Cognitive Science*, vol. 17, no. 2, pp. 119-138.

Pei, W., Ge, Tao and Chang, B. (2015). “An effective neural network model for graph-based dependency parsing”, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pp. 313-322

Pennington, J., Socher, R. and Manning, C. D. (2014). “GloVe: Global Vectors for word representation”, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532-1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. and Zettlemoyer, L. (2018). “Deep contextualized word representations”, *arXiv*:1802.05365

Ramshaw L. and Marcus, M. (1995). “Text chunking using transformation-based learning”, *Third Workshop on Very Large Corpora*, pp. 82-94

Segura-Bedmar, I., Martínez, P. and Zazo, M. H. (2013). “SemEval-2013 Task 9 : Extraction of drug-drug interactions from biomedical texts (DDIExtraction 2013)”, *Proceedings of the 2nd Joint Conference on Lexical and Computational Semantics and the 7th International Workshop on Semantic Evaluation*, pp. 341-350.

Veenstra, J. (1998). “Fast NP chunking using memory-based learning techniques”, *Proceedings of the Joint Conference on New Methods in Language Processing and Computational Natural Language Learning*, pp. 1-8.

Vinyals, O., Fortunato, M. and Jaitly, N. (2015). “Pointer network”, *Advances in Neural Information Processing Systems*, pp. 2692-2700.

Voutilainen, A. (1993). “NPTool, a detector of English noun phrases”, *Proceedings of the Workshop on Very Large Corpora, Association for Computational Linguistics*, pp. 48-57.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. and Le, Q. V. (2019). “XLNet: Generalized autoregressive pretraining for language understanding”, *arXiv*:1906.08237

Yoon, J., Choi, K. and Song, M. (1999). “Three types of chunking in Korean and dependency analysis based on lexical association”, *Proceedings of the 18th International Conference on Computer Processing Languages*, pp. 59-65.

Zeman, D. *et al*. (2018). “CoNLL 2017 Shared Task: Multiligual parsing from raw text to universal dependencies”, *Proceedings of the 11th International Conference on Language Resources and Evaluation.*

김민석, 신창욱, 오진영, 차정원 (2019). “XLNet을 이용한 한국어 구문분석”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 540-542.

김재훈 (2000a). “부분 구문분석 방법론”, 정보처리학회지, 제7권, 제6호, pp. 83-96.

김재훈 (2000b). *한국어 부분 구문분석의 단위와 그 표지*, *한*국해양대학교 컴퓨터공학과, 기술문서, KMU-NLP-TR-2000-006.

김창제, 정천영, 김영훈, 서영훈 (1995). “부분적인 어절 결합을 이용한 효율적인 한국어 구문 분석기”, *한국정보과학회 학술발표논문집*, 제22권, 제2호, pp. 597-600.

김태웅, 조희영, 서형원, 김재훈 (2006). “의존명사를 포함하는 보조용언의 구 묶음”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 279-284.

김홍규, 강범모, 홍정하 (2007). “21세기 세종계획 현대국어 기초말뭉치: 성과와 전망”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 311-316.

김홍진, 오신혁, 김담린, 김보은, 김학수 (2019). “멀티헤드 어텐션과 포인터 네트워크 기반의 음절 단위 의존 구문 분석”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 546-548.

나동렬 (1994). “한국어 파싱에 대한 고찰”, *정보과학회논문지*, 제 12권, 제 8호, pp. 33-46.

나승훈, 김강일, 김영길 (2016). “Stack LSTM을 이용한 전이 기반 한국어 의존 파싱”, *한국정보과학회 학술발표논문집*, pp. 732-734.

나승훈, 이건일, 신종훈, 김강일 (2017). “Deep Biaffine Attention을 이용한 한국어 의존 파싱”, *한국정보과학회 학술발표논문집*, pp. 584-586.

남궁영, 김재훈 (2018a). *한국어 기저구 표지 정의*, 한국해양대학교 컴퓨터공학과, 기술문서, KMOU-NLP-2018-002.

남궁영, 김창현, 천민아, 박호민, 윤호, 최민석, 김재훈 (2018b). “구문 분석을 위한 한국어 말덩이 정의”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 409-412.

남궁영, 김창현, 천민아, 박호민, 윤호, 최민석, 김재균, 김재훈 (2019). “Bi-LSTM/CRF 모델을 이용한 한국어 구묶음”, *한국정보과학회 학술발표논문집*, pp. 631-633.

민진우, 홍승연, 이영훈, 나승훈 (2019a). “Graph Neural Networks을 이용한 한국어 의존 구문 분석”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp.

537-539.

민진우, 나승훈, 신종훈, 김영길 (2019b). “Dual Decomposition을 이용한 전이기반 및 그래프 기반 의존 파서 통합 모델”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 25-29.

박의규, 나동열 (2006). “한국어 구문분석을 위한 구묶음 기반 의존명사 처리”, *인지과학*, 제17권, 제2호, pp. 119-138.

박천음, 이창기 (2017). “포인터 네트워크를 이용한 한국어 의존 구문 분석”, *한국정보과학회논문지*, 제44권, 제8호, pp. 822-831.

박천음, 이창기, 임준호, 김현기 (2019). “BERT를 이용한 한국어 의존 구문 분석”, *한국정보과학회 학술발표논문집*, pp. 530-532.

안동언 (1987). *기계번역을 위한 한국어 해석에서 형태소로부터 구문요소의 형성에 관한 연구*, 한국과학기술원, 전산학과, 석사학위논문.

안재현, 고영중 (2018). “의존 관계명 태그 분포를 이용한 한국어 의존 구문 분석”, *정보과학회 컴퓨팅의 실제 논문지*, 제24권, 제9호, pp. 487-492.

안휘진, 박찬민, 서민영, 이재하, 손정연, 김주애, 서정연 (2018). “Deep Bi-affine Network와 스택 포인터 네트워크를 이용한 한국어 의존 구문 분석 시스템”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 689-691.

양재형 (2000). “규칙기반 학습에 의한 한국어의 기반 명사구 인식”, 정보과학회 논문지: 소프트웨어 및 응용, 제27권, 제10호, pp. 1062-1071.

원혜진, 류법모 (2019). “세종구구조말뭉치 기반 Universal Dependency 말뭉치 반자동 생성 연구”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 545-547.

이건일, 이종혁 (2015). “순환 신경망을 이용한 전이 기반 한국어 의존 구문 분석”, *정보과학회 컴퓨팅의 실제 논문지*, 제21권, 제8호, pp. 567-571.

이찬영, 김진웅, 김한샘 (2018). “Universal Dependency 관계 태그셋의 한국어 적용”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 334-339.

이창기, 김준석, 김정희 (2014). “딥 러닝을 이용한 한국어 의존 구문 분석”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 87-91.

조경철, 김주완, 김균엽, 박성진, 강상우 (2019). “다양한 앙상블 알고리즘을 이용한 한국어 의존 구문 분석”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 543-545.

차다은, 이동엽, 임희석 (2018). “Stack-Pointer Network를 이용한 한국어 의존 구문 분석”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 685-688.

최용석, 이공주 (2018). “한국어 구절 구문 코퍼스의 의존 구문 구조 트리로의 변환에서 중심어 전파 규칙”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 514-519.

최용석, 이공주 (2019). “고차원 정보와 스택-포인터 네트워크를 이용한 한국어 의존 구문 파서”, *정보과학회논문지*, 제46권, 제7호, pp. 636-643.

한장훈, 박영준, 정영훈, 이인권, 한정욱, 박서준, 김주애, 서정연 (2019). “순차적 구문 분석 방법을 반영한 포인터 네트워크 기반의 한국어 의존 구문 분석기”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 533-536.

홍승연, 나승훈, 신종훈, 김영길 (2018). “Bidirectional Stack Pointer Network를 이용한 한국어 의존 파싱”, *한국정보과학회 언어공학연구회 학술발표논문집*, pp. 19-22.

홍승연, 나승훈, 신종훈, 김영길 (2019). “BERT와 ELMo 문맥화 단어 임베딩을 이용한 한국어 의존 파싱”, *한국정보과학회 학술발표논문집*, pp. 491-493.

홍윤표 (2009). “21세기 세종 계획 사업 성과 및 과제”, *새국어생활*, 19(1):5-33.

황영숙, 정후중, 박소영, 곽용재, 임해창 (2002). “자질집합선택 기반의 기계학습을 통한 한국어 기본구 인식의 성능향상”, *정보과학회논문지: 소프트웨어 및 응용*, 제29권, 제9-10호, pp. 654-668.

황이규, 이현영, 이용석 (2000). “형태소 및 구문 모호성 축소를 위한 구문단위 형태소의 이용”, *한국정보과학회논문지:소프트웨어 및 응용*, 제27권, 제7호, pp.784-793.

감사의 글

논어 학이(學而)편에 이런 구절이 나옵니다.

- 젊은이들은 집에 들어가서는 부모님께 효도하고 나가서는 어른들을 공경하며, 말과 행동을 삼가고 신의를 지키며, 널리 사람들을 사랑하되 어진 사람과 가까이 지내야 한다.

이렇게 행하고서 남은 힘이 있으면 그 힘으로 글을 배우는 것이다. -

제 배움이 이러한 길로 나아갈 수 있도록 이끌어 주시고 함께 해 주신 할아버지, 할머니, 아버지, 어머니, 진이, 홍이, 한국해양대학교 자연언어처리실험실 선배, 동기, 후배님, 그리고 조교님들, 고려대학교 서화회, 별빛항해 선후배님들, 재언 선배, 예진, 보경, 민경, 영석, 경현 님, 해금이까지 모두 이 지면을 통해 고마운 마음을 전합니다.

또한, 현시대 자연언어처리 분야를 함께 공부하고 이끌어 갈 학회에서 만난 수많은 연구자와 책과 강의를 통해 만나 뵀던 선구자분들, 부족한 논문을 끝까지 지도해 주시고 아낌없이 조언을 주신 류길수 교수님, 박휴찬 교수님께도 감사의 말씀 드립니다.

그리고,

학문에서뿐 아니라 사람으로서 삶을 살아갈 본보기상이 되어주신 김재훈 지도 교수님께 이 지면을 통해 고마운 마음을 올립니다. 살아가면서 안고 있던 궁금증과 고민들을 선생님을 보며 해결하고 감탄하고 배웠습니다.

끝으로, 저를 자랑스럽게 바라봐 주심으로 인해 제가 가는 길에 힘과 용기를 주신 아버지, 어머니, 진이, 홍이에게 다시 한번 감사의 말씀 올립니다.

사랑합니다.