



저작자표시-비영리-동일조건변경허락 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.
- 이차적 저작물을 작성할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



동일조건변경허락. 귀하가 이 저작물을 개작, 변형 또는 가공했을 경우에는, 이 저작물과 동일한 이용허락조건하에서만 배포할 수 있습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

工學博士學位論文

지배소 후위 언어를 위한
효율적 구문분석



2013年 月

昌原大學校 大學院

컴퓨터공학과

吳 珍 瑛

工學博士學位論文

지배소 후위 언어를 위한
효율적 구문분석

Efficient Parsing for Head final Language

指導教授 車損遠

이 論文을 工學博士學位論文으로 提出함.

2013年 月

昌原大學校 大學院

컴 퓨 터 工學科

吳 珍 瑛

吳珍瑛의 博士學位 論文을 認准함.

審 查 委 員 長 김 한 경 ①

審 查 委 員 이 중 근 ①

審 查 委 員 이 수 현 ①

審 查 委 員 한 요 섭 ①

審 查 委 員 차 정 원 ①

2013年 月 日

昌原大學校 大學院

목차

그림 목차	iv
표 목차	vi

I. 서론	1
1. 구문분석을 위한 실용적인 모티브	2
2. 애매성(Ambiguity)	5
3. 의존문법	6
4. 한국어 특징	8
5. 세종 구문부착 코퍼스	12
6. 논문의 구성	14
II. 관련연구	16
1. 그래프 기반 구문분석	18
2. 트랜지션 기반 구문분석	20
3. 통합 구문분석	22
III. 다단계 구단위화를 이용한 한국어 의존구조 분석	24
1. 문제정의	24
2. 구문태그	26
3. 다단계 구단위화 방법	31

4.	Conditional Random Fields	34
5.	비어휘 다단계 구 단위화 시스템	36
5.1.	실험환경.....	36
5.2.	실험 결과.....	38
5.3.	오류 분석.....	39
6.	공기정보(Co-occurrence)를 이용한 의미제약.....	41
6.1.	공기 정보.....	42
6.2.	실험 결과.....	44
IV.	문장 구조 Frame	50
1.	문장 구조 자질의 필요성	50
2.	필수격.....	51
3.	다단계 구단위화의 문장 구조 자질	52
4.	Frame 설정	62
5.	Frame 적용	64
5.1.	자질 학습.....	64
5.2.	분석 가이드.....	66
6.	Frame 실험	67
6.1.	문장 구조 설정	67
6.2.	자질 학습 성능	69
6.3.	분석 가이드 성능	70
7.	Frame 모델 성능 향상.....	71

8. 성능 비교	75
V. 결론 및 향후 연구	77



그림 목차

<그림 I-1> 의존문법을 이용한 구문구조 예	3
<그림 I-2> 세종코퍼스 구구조 형태	13
<그림 I-3> 세종코퍼스 의존구조 형태	14
<그림 II-1> 그래프 기반 구문분석 예제	18
<그림 II-2> 트랜지션 기반 구문분석 예제	20
<그림 III-1> 제안 시스템 구조도	26
<그림 III-2> 구문태그 예제	27
<그림 III-3> 구문태그 자질의 예	28
<그림 III-4> 의사결정나무	30
<그림 III-5> 다단계 구단위화를 이용한 구문분석의 예	32
<그림 III-6> 구문분석 자질의 예	33
<그림 III-7> 기능표지가 없는 어절 예	40
<그림 III-8> 복잡한 문형 예	41
<그림 III-9> 의미정보가 필요한 문장 예	42
<그림 III-10> 공기정보 예제	43
<그림 III-11> 동사 묶음 처리 예	44
<그림 IV-1> 오류 전파 예 1	53
<그림 IV-2> 오류 전파 다단계 분석 예 1	54

<그림 IV-3> 오류 전파 예 2.....	55
<그림 IV-4> 오류 전파 다단계 분석 예 2.....	56
<그림 IV-5> Frame 예제	59
<그림 IV-6> 규칙 예.....	60
<그림 IV-7> 하향식 분석 예.....	61
<그림 IV-8> 다단계 구 단위화에서 Frame 설정 예.....	63
<그림 IV-9> Frame 자질의 예	64
<그림 IV-10> 자질 학습 구조도.....	65
<그림 IV-11> Frame 특징.....	66
<그림 IV-12> 구문분석 규칙.....	67
<그림 IV-13> 구문분석과 확장 문장 구조 학습 방법.....	73

표 목차


<표 I.1> 영어의 문법 관계.....	8
<표 I.2> 주요 구문태그 집합.....	8
<표 III.1> 기본자질.....	29
<표 III.2> 조합된 자질들.....	31
<표 III.3> 비어휘 시스템 결과.....	38
<표 III.4> 기본 실험 성능.....	46
<표 III.5> 기능어 형태소 정보 추가 실험 성능.....	47
<표 III.6> 공기정보 빈도 학습.....	49
<표 III.7> 공기정보 클래스 학습.....	49
<표 IV.1> 비어휘 시스템의 각 문장 길이별 성능.....	57
<표 IV.2> 구문태그의 의존구조 길이 통계.....	58
<표 IV.3> 4개 자질 실험과 6개 자질 실험의 Frame 설정 성능.....	68
<표 IV.4> Frame 자질 학습 실험 성능.....	69
<표 IV.5> Frame 분석 가이드 실험 성능.....	70
<표 IV.6> 정답 Frame의 구문분석 성능.....	71
<표 IV.7> 확장 문장 구조 Frame 설정 성능.....	74
<표 IV.8> 확장 문장 구조 모델 적용.....	74

<표 IV.9> 성능 비교	76
----------------------	----



제 I 장

서론



구문분석은 문장 안에서 문장성분들의 관계를 찾는 과정이다. 품사태깅과 더불어 구문분석은 자연어처리 분야에서 필수 단계 중 하나이다. 응용 프로그램에서 언어 분석에 대한 요구가 증가하면 할수록 구문분석에 대한 요구는 증가한다. 예를 들어 인터넷 문서와 같이 대용량의 문서에 대한 정보추출의 경우 단순히 문자열 패턴으로 정보를 추출하는 것이 아니라 구문분석을 이용하면 사용자의 요구에 맞는 정보를 좀 더 정확하게 추출할 수 있다. 이 경우에 반드시 필요한 것이 처리 속도, 안정성, 그리고 성능이다. 품사태깅과 달리 구문분석은 각 문장 성분들 간의 관계를 조사하기 위해서 일반적으로 $O(n^3)$ 의 처리 시간이 소요되기 때문에 문장이 길수록 속도는 더

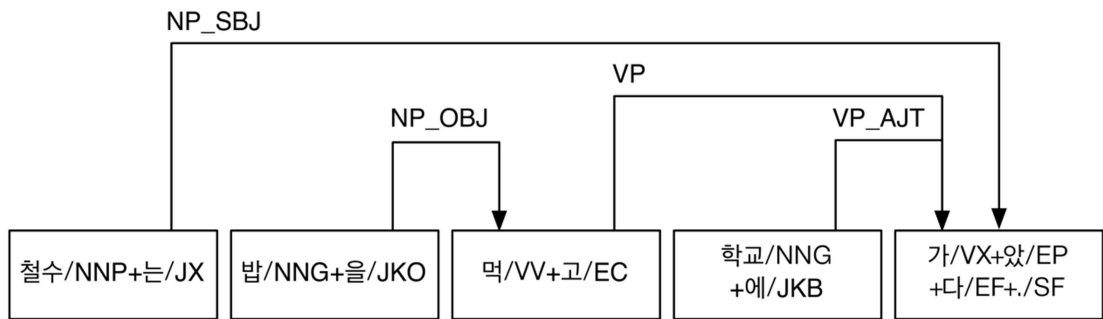
속 느려지게 된다. 다시 말해 문자열 패턴 방법과 비교하여 좀더 정확한 결과를 추출하는 장점은 있지만, 구문분석은 언어 분석의 기초단계에 해당하므로 한 문장의 처리 시간이 길어지면 상위 응용 프로그램의 대기시간이 늘어난다.

또한, 구문분석은 문장 전체에 대한 분석결과를 출력하므로 문장이 복잡해질수록 완전한 분석결과를 출력하는 것이 힘들어진다. 더욱이 인터넷 문서와 같이 비문이 많은 문서에서는 분석을 성공하지 못하는 경우도 많이 발생한다. 성능은 모든 시스템에서 중요하지만 특히 구문분석에서는 구문분석의 오류가 응용 프로그램에 직접 영향을 미치기 때문에 더욱 중요하다.

영어권에서는 구문분석 연구가 성숙하여 구문분석 프로그램을 사용하여 다양한 연구 결과를 보이고 있다. 그러나 한국어의 경우는 다양한 연구에서 사용할 수 있는 고속, 고성능 구문분석기가 존재하지 않는다. 따라서 영어권에 비해서 사용할 수 있는 자원이 줄어들어 연구의 깊이와 결과가 좋지 못한 상황이다.

1. 구문분석을 위한 실용적인 모티브

<그림 I-1>은 의존문법을 이용한 구문구조를 나타낸다. 먼저 단어 자체와 품사가 있다. 예를 들어 ‘칠수-는’은 ‘NNP+JX’의 품사를 가진다. ‘NNP’는 고유명사를 나타내고 ‘JX’는 보조사를 나타낸다. 또한 ‘밥을’은 ‘밥/NNG+을/JKO’로 분석된다.



<그림 I-1> 의존문법을 이용한 구문구조 예

여기서 ‘NNG’는 보통명사이고 ‘JKO’는 목적격 조사를 나타낸다. 구문구조에는 문장성분을 나타내는 정보도 함께 포함하고 있다. ‘철수/NNP+는/JX’는 ‘NP_SBJ’가 된다. 이것은 명사구(NP)이면서 주어(SBJ)를 나타낸다. ‘밥/NNG+을/JKO’은 ‘NP_OBJ’가 된다. 이것은 명사구(NP)이면서 목적어(OBJ)가 된다는 것을 의미한다. ‘먹/VV+고/EC’는 VP가 된다. 이것은 동사구를 나타낸다.

또한 구문구조에는 문장성분 간의 관계를 담고 있다. 예문에서 ‘밥을’은 ‘먹고’의 목적어이고 ‘학교에’는 ‘갔다’와 연결되는 부사어이다. 그리고 ‘갔다’는 ‘철수는’을 주어로 갖는 동사이다.

이러한 구문구조와 문장성분을 이용하여 동사-필수격 형태로 변환한다면 문장의 의미를 이해하는데 도움을 준다.

구문분석은 많은 자연어 처리 응용의 성능향상에 도움을 준다. 여기에는 정보추출, 정서분석, 정보검색 그리고 통계적 기계번역이 포함된다.

정보추출(Information Extraction)에서는 문서로부터 정보를 추출하고 이들 사이의

관계를 찾아 데이터베이스에 입력한다. 예를 들어 많은 자연재해를 다루고 있는 문서에서 자연재해와 이 재해가 발생한 년도, 위치, 피해액들과의 관계를 구문분석기를 이용해 찾을 수 있다.

상품에 대한 사용자의 평을 찾는 정서분석(Sentiment Analysis)에서 정서분석의 대상과 이를 평가하는 용언(‘좋다’, ‘싫다’)들과 관계 역시 구문분석기와 사전을 이용하여 쉽게 찾을 수 있다.

고성능 정보검색 역시 구문분석기와 관련이 있다. ‘창원에 있는 일식집을 찾아줘.’라고 하는 질의가 있을 때, 기존의 정보검색은 ‘창원’과 ‘일식집’이 있는 문서를 모두 찾아서 보여주지만 구문분석기를 사용하면 ‘일식집’ 중에서 ‘창원’에 위치한 것을 찾아줄 수 있어 성능을 높일 수 있다.

기계번역(Machine Translation)의 경우는 좀 더 좋은 응용의 예이다. 기계번역에서 가장 문제가 되는 것은 정렬이다. 병렬 문장이 주어져 있을 경우, 입력 언어의 단어나 구에 대응하는 번역 언어의 단어나 구를 찾는 것이다. 영어와 한국어의 경우에는 어순이 많이 다르기 때문에 순서대로 대응되지 않는다. 구문분석을 사용하면 문장에서의 관계에 따라 이 대응 순서를 일치시키는데 도움을 준다.

영어: Apple announced iphone.

한국어: 애플이 아이폰을 출시했다.

영어: Speaker said that Apple announced iphone yesterday.

한국어: 대변인은 애플이 아이폰을 어제 출시했다고 말했다.

입력 문장을 그대로 비교하면 두 문장은 어순이 매우 다르다. 영어는 ‘주어-동사-목적어’ 어순이지만 한국어는 ‘주어-목적어-동사’ 어순이다. 그렇지만 이 문장들을 구문분석을 하고 나면 문제는 달라진다. 두 문장 모두 동사인 ‘announce’의 주어가 ‘Apple’이고 목적어는 ‘iphone’이다. 동일하게 한국어 문장에서도 ‘출시하다’의 주어는 ‘애플’이고 목적어는 ‘아이폰’이 된다. 따라서 구문구조에서 동일하게 적용할 수 있다.

2. 애매성(Ambiguity)

자연어처리 시스템에서 가장 중요한 문제는 애매성 해소이다. 애매성 해소는 후보가 여러 개 있을 경우에 이 중에서 적당한 하나를 선택하는 것이다.

- 품사 태깅에서 애매성: 예를 들어 ‘나는’은 ‘날/동사+는/어미’, ‘나/대명사+는/조사’, ‘나/동사+는/어미’로 후보가 될 수 있다. 이 중에서 문맥에 맞는 후보를 선택해야 한다.
- 개체명 인식에서 애매성: 개체명 인식에서 ‘창원대학교’는 조직과 장소로 모

두 사용될 수 있다. 문맥에 따라 개채명이 조직인지 장소인지를 판별해야 한다.

- 구문분석에서 애매성: ‘철수는 빵을 먹은 영희가 집에 갔다고 생각했다.’라는 문장에서 ‘먹(eat)’의 주어가 ‘철수’인지 ‘영희’인지 구별해야 한다.

애매성을 해결하기 위해 많은 연구들이 있었다. 이들에 대해서는 다음 장에서 다룬다.

3. 의존문법

문장을 구성하는 요소끼리의 의존관계를 파악함으로써 문장을 분석하는 것을 의존문법이라고 한다. 원래 Dependency라는 말은 언어학자인 David Hays가 처음 사용하였다. 영어에서 지배소(Governor)와 피지배소(Dependent)는 단어 단위이지만 한국어에서는 어절 단위일 수도 있고 형태소 단위일 수도 있다. 그렇지만 문장 성분이라는 입장에서 보면 어절이 더 적당해 보인다.

구 구조문법은 구성요소(Constituent)가 모여서 더 큰 구성요소를 만들고 문장까지 구성해가는 방식이다. 그에 반해 의존문법은 구성요소들 간의 의존관계들의 집합이다. 따라서 만들어질 수 있는 의존관계는 문장의 길이(구성요소 수)가 n 일 때 $\frac{n(n-1)}{2}$ 가 된다. 그렇지만 한국어에서는 다음에서 설명할 지배소 후위 원칙과 지배소 유일

원칙에 의해 그 수는 적어진다.

한국어에서 의존문법이 유용한 이유는 다음과 같다.

- 1) 한국어가 부분 어순 자유 언어이기 때문이다. 한국어는 조사나 어미가 발달해 있고 이로 인해 동사를 제외한 다른 성분의 이동이 자유롭다. 의존문법은 규칙 추가 없이 이러한 자유어순을 표현하는데 적합하다.
- 2) 생략 등으로 인한 불연속성에 견고하다.

- 의존관계 제약

의존관계는 두 구성요소 사이의 관계를 정의한 것이다. 예를 들어 ‘철수가 밥을 먹었다.’에서 ‘밥을’-‘먹었다’ 사이에는 목적어 관계가 있다. Marie-Catherine de Marneffe et al[1] 등은 영어에서 <표 I.1>과 같이 48개의 문법 관계를 정의하였다.

세종 구문 부착 코퍼스에서는 의존관계를 구문표지와 기능표지로 나누어서 표시하고 있다. 이것은 한 어절이 지배소가 될 때와 피지배소가 될 때 적용하기에 편하다. 지배소가 되면 구문표지를 보면 되고 피지배소가 될 때는 기능표지를 보면 된다.

주요 한국어 구문태그 집합은 <표 I.2>와 같다. 전체 구문태그는 부록에 기술하였다.

<표 I.1> 영어의 문법 관계

Argument Dependencies	설명
nsubj	문장 주어
csbj	절 주어
dobj	목적어
iobj	간접 목적어
pobj	전치사의 목적어
Modifier Dependencies	설명
tmod	시제 수식
appos	동격 수식
det	한정사
prep	전치 수식

<표 I.2> 주요 구문태그 집합

구문표지		기능표지	
S	문장	SBJ	주어
NP	체언구	OBJ	목적어
VP	용언구	CMP	보어
VNP	긍정 지정사구	MOD	체언 수식어
AP	부사구	AJT	용언 수식어
DP	관형사구	CNJ	접속어

4. 한국어 특징

한국어는 교착어(Agglutinative Language), 후위언어(Postpositional Language)이며 주어-목적어-동사의 순서를 가지는 비구성 언어(non-Configurational Language)이다. 한국어에서 자주 나타나는 조사, 어미, 선어말 어미 등은 문장에서 명사구, 동사구의 기능

을 나타낸다. 또한 동사구를 명사나 형용사구로 변경시키기도 한다. 이 장에서는 구문분석과 관련된 한국어 특징에 대해서 살펴본다.

- 어절은 여러 개의 형태소로 이루어져 있다.

한국어에서 어절은 명확하게 구분되는 여러 개의 형태소로 이루어져 있다. 예를 들어 다음과 같은 문장을 보자.

(1) 나는 학교에 갔다.

이 문장에서 “갔다”는 다음 예문 (2)와 같이 3개의 형태소로 이루어져 있다.

(2) 가/동사 + 았/과거시제 선어말 어미 + 다/종결 어미

- 후위 언어이다.

한국어는 조사, 어미, 선어말 어미가 발달된 후위언어이다. 영어와 같이 어순에 의해서가 아니라 이러한 기능 형태소들이 명사의 구문적 기능, 동사의 시제나 상, 어절간의 수식관계 등 문법관계를 결정한다. 다음과 같은 예를 보자.

(3) 철수-가 사과-를 먹-었-다.

예문 (3)에서 ‘가’는 주격을 나타내고, ‘를’은 목적격을 나타낸다. 그리고 ‘었’은 동사의 시제가 과거임을 나타낸다.

- 어순이 비교적 자유롭다.

한국어는 기본적으로 SOV(주어-목적어-동사) 언어이다. 그럼에도 불구하고 영어와 비교해서 동사를 제외하고는 어순이 비교적 자유롭다. 예문 (4)를 보자.

(4) a. 그가 빵을 먹었다. (SOV)

b. 빵을 그가 먹었다. (OSV)

예문 a와 b가 모두 가능하다.

- 이중 주어, 이중 목적어가 있다.

한국어에서는 때때로 이중 주어와 이중 목적어가 나타난다. 예문 (5)를 보자.

(5) 나-는 그-가 좋다.

이 경우에 “나-는”과 “그-가”는 모두 주어가 된다. 이 경우에 어느 것이 문장의 주어인지 판별하는 것은 어려운 일이다. 한국어에서는 다음 예문 (6)과 같이 이중 목적어도 가능하다.

(6) 그 사람을 책을 주어라.

- 필수 성분이 생략될 수 있다.

한국어에서는 또한 필수격이 생략되는 경우가 빈번히 일어난다. 다음 예문 (7)을 보자.

- (7) a. 점심을 드셨어요?
b. 당신은 점심을 드셨어요?

예문에서 b는 a와 같이 사용할 수 있다. 또한 예문 (8)과 같은 대등문에서는 동사도 다른 필수격과 같이 생략될 수 있다.

(8) 나는 밥과 빵을 먹었다.

5. 세종 구문부착 코퍼스

본 논문에서 사용하는 세종코퍼스[27]는 약 6만 문장을 제공하고 있는 대용량 코퍼스이다. 기본으로 <그림 1-2>와 같이 각 문장이 구구조 방식으로 표현 되어있다. 구구조는 구성성분 단위로 나누어서 분석하는 것인데 괄호로 내포된 구성 성분을 포함하여 트리 형식으로 표현한다.

본 연구는 구구조 세종코퍼스를 <그림 1-3>와 같이 의존구조로 바꾸는 작업을 수행하여 실험하였다. 또한 이외의 세종코퍼스 분석 오류도 수정하였다.

수정한 것은 크게 다음과 같다.

(1) 보조동사, 본동사

<그림 1-2>의 마지막 세 어절의 구조를 보면 ‘알 수 있다.’가 구로 묶여 있고 나머지 앞의 어절이 ‘알 수 있다.’구를 수식하는 것으로 표현되어 있다. 이 문장처럼 동사구가 있을 경우 마지막 동사를 문장동사로 간주하고 지배소를 가지게 구성되어있다.

예를 들어 ‘꽃무늬 디자인이 주류를 이루고 있다.’라는 문장에서 ‘이루고’는 본동사이고 ‘있다.’는 보조동사이다. 세종코퍼스에서 ‘디자인이’라는 어절이 ‘있다’라는 보조동사를 지배소로 가지도록 구축되어 있다.

이런 보조동사, 본동사 문서에 대해서 모두 본동사를 지배소로 가지도록 수정하였다.

(2) 구 패턴

<그림 I-3>과 같이 9의 마지막 형태소가 ‘르’이고 다음 10번 어절에 ‘NNB’ 품사태그, 11번 어절에 ‘있’ 또는 ‘없’ 형태소가 있을 경우 해당 구 패턴을 묶어서 ‘알’ 어절을 지배소로 가지도록 수정하였다.

(3) 이외 오류 수정

세종 구문코퍼스는 오랜 기간을 통해서 개발되었다. 오랜 시간을 거치면서 일관성 문자, 구문 부착 오류 등이 다수 포함되어 있다. 본 논문을 작성하기 전에 총 9만 8천여 문장의 코퍼스(문장 당 평균 14.5어절)를 정제 하였다. 이외에도 용언이 없는 문장을 제거하였다. 이 문장들은 대부분 제목으로 길이가 짧은 문장이다. 그리고 오류가 포함된 문장들도 수정 또는 삭제 하였다.

; 가장 돈을 많이 받는다는 미국의 프로선수들 연봉과 비겨보면 알 수 있다.
(S (VP (NP_AJT (NP (VP_MOD (AP 가장/MAG)
(VP_MOD (NP_OBJ 돈/NNG + 을/JKO)
(VP_MOD (AP 많이/MAG)
(VP_MOD 받/VV + 는다는/ETM))))
(NP (NP_MOD 미국/NNP + 의/JKG)
(NP 프로/NNG + 선수/NNG + 들/XSN)))
(NP_AJT 연봉/NNG + 과/JKB))
(VP 비기/VV + 어/EC + 보/VX + 면/EC))
(S (NP_SBJ (VP_MOD 알/VV + 르/ETM)
(NP_SBJ 수/NNB))
(VP 있/VV + 다/EF + .SF)))

<그림 I-2> 세종코퍼스 구구조 형태

; 가장 돈을 많이 받는다는 미국의 프로선수들 연봉과 비겨보면 알 수 있다.

1	4	AP	가장/MAG
2	4	NP_OBJ	돈/NNG+을/JKO
3	4	AP	많이/MAG
4	6	VP_MOD	받/VV+는다는/ETM
5	6	NP_MOD	미국/NNP+의/JKG
6	7	NP	프로/NNG+선수/NNG+들/XSN
7	8	NP_AJT	연봉/NNG+과/JKB
8	9	VP	비기/VV+어/EC+보/VX+면/EC
9	10	VP_MOD	알/VV+르/ETM
10	11	NP_SBJ	수/NNB
11	11	VP	있/VV+다/EF+./SF

<그림 I-3> 세종코퍼스 의존구조 형태

6. 논문의 구성

본 논문에서는 의존문법을 이용하여 한국어 구문분석을 시도한다. 각 시도에 대해서 여기서 간단히 설명한다.

- 비어휘 모델: 형태소 정보를 사용하지 않고 품사, 품사태그와 구문태그만을 사용하여 구문분석을 진행한다.
- 비어휘 모델 + 형태소 자질, 보조용언: 기능어 형태소를 품사와 병기함으로써 분별성을 높였다. 또한 보조용언을 묶어서 처리함으로써 처리 단계를 줄일 수 있다.

- 비어휘 모델 + 형태소 자질, 보조용언 + 공기관계: 두 성분의 관계 찾기 문제를 CRF를 이용한 꼬리달기 문제로 전환하여 처리한다. 여기서 의미 제약을 줄 수 있는 방법이 없다. 따라서 본 논문에서는 성분들 사이에 의미제약은 두 의미형태소의 공기관계를 사용한다.
- 비어휘 모델 + 형태소 자질, 보조용언 + 공기관계 + Frame 자질학습: 의미정보를 이용해 제약을 가하더라도 복잡한 구문구조를 가진 문장을 처리하는데 한계가 있다. 따라서 본 논문에서는 문장구조 정보를 자질로 추가하여 실험하였다.
- 비어휘 모델 + 형태소 자질, 보조용언 + 공기관계 + Frame 분석가이드: 문장구조 정보를 규칙으로 처리한 방법이다.

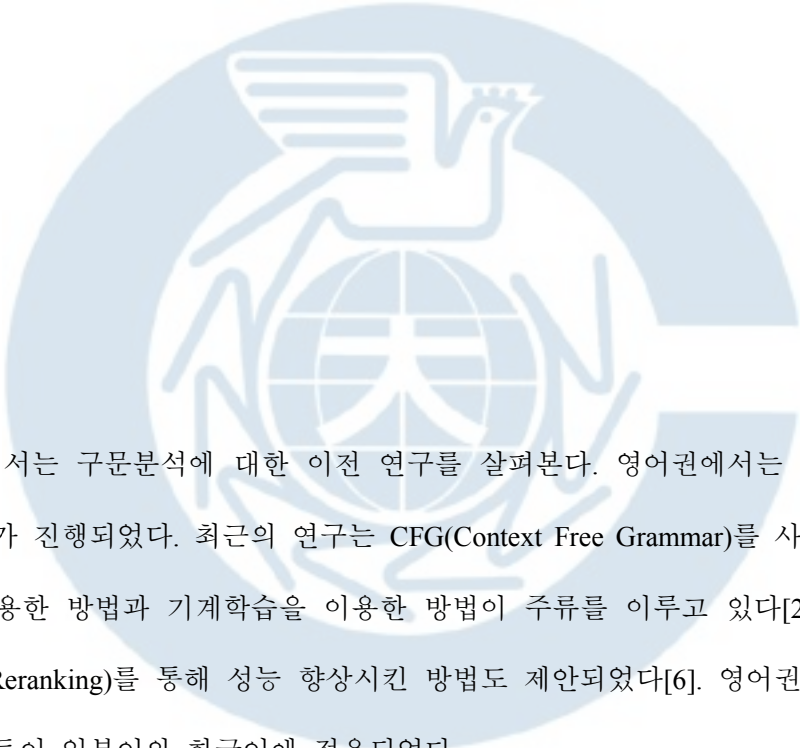
본 논문의 구성은 다음과 같다.

II장에서는 이전에 개발된 구문분석 시스템에 대해서 시스템의 특징과 성능을 알아본다. III장에서는 제안하는 전체 시스템의 구조와 특징을 기술한다.

IV은 본 논문이 제안하는 문장 구조 Frame에 대한 기술하고 V장에서는 결론과 향후 과제를 다룬다.

제 II 장

관련 연구



이 장에서는 구문분석에 대한 이전 연구를 살펴본다. 영어권에서는 오래 전부터 많은 연구가 진행되었다. 최근의 연구는 CFG(Context Free Grammar)를 사용하고 통계 모델을 이용한 방법과 기계학습을 이용한 방법이 주류를 이루고 있다[2,3,4,5]. 또한 재순위화(Reranking)를 통해 성능 향상시킨 방법도 제안되었다[6]. 영어권에서 개발된 많은 방법들이 일본어와 한국어에 적용되었다.

어순이 비교적 자유로운 일본어에서도 의존 구조를 이용하는 방법이 많이 제안되었다. 의존 구조의 애매성을 해소하기 위해 통계적 방법과 기계학습을 이용한 다양

한 방법이 제안되었다. 예를 들어 최대 우도 추정(Maximum Likelihood Estimation)[7], 결정 트리(Decision Tree)[8], 최대 엔트로피 모델(Maximum Entropy Model)[9,10], 지지 기반 기계(Support Vector Machine)[11] 등이다.

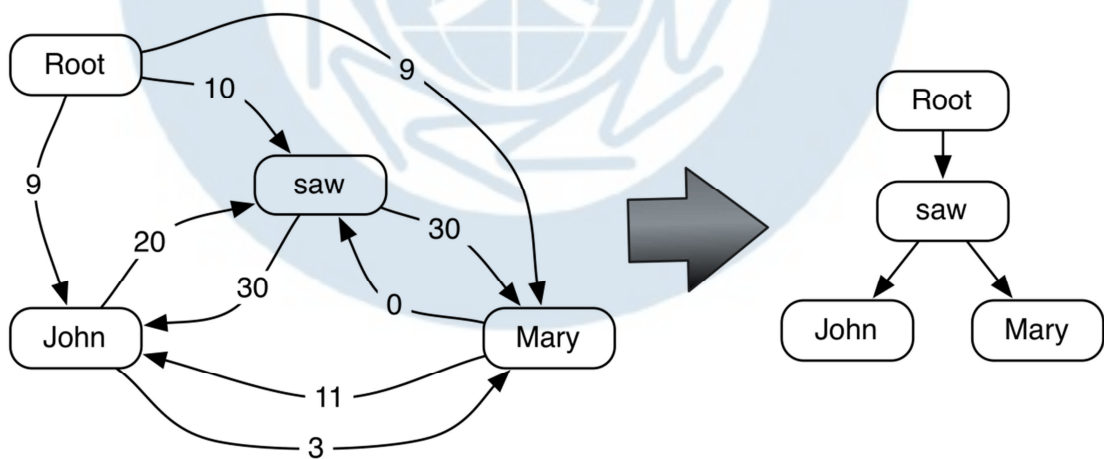
한국어에서도 다양한 시도가 있었다. 한국어 구문분석은 단일화 문법(Unification Grammar), 핵심어 중심 구구조 문법(HPSG: Head-Driven Phrase Structure Grammar), 어휘 기능 문법(LFG: Lexical Functional Grammar), 결합 범주 문법(CCG: Combinatorial Categorical Grammar)을 이용한 시스템들이 제안되었다[12,13,14,15,16]. 최근에는 거의 모든 연구가 의존 문법을 기반으로 하고 있다. 또한 일본어와 마찬가지로 의존 구조의 애매성을 해소하기 위해 다양한 통계 방법과 기계학습을 이용하는 방법들이 제안되었다[17,18]. 초기의 한국어에 대한 연구는 학습 코퍼스의 부족으로 연구실 수준의 연구에 머물렀지만 최근에는 한국어정보베이스(KIB: Korean Language Information Base), 세종 구문 코퍼스 등이 제작되면서 대용량 코퍼스를 이용하는 연구가 활기를 띠고 있다[17,18].

어순이 비교적 자유로운 언어에서 두 단어간의 연결 관계를 결정짓는 의존문법이 상대적으로 많이 연구되고 있다. 그 중 최근에는 트랜지션 기반[19] 구문 분석과 그래프 기반[20]의 구문분석이 활발하다. 이 두 방법의 큰 차이점은 추론방법, 학습, 자질 표현이다.

1. 그래프 기반 구문분석

그래프 기반 구문분석은 문장을 방향성 있는 트리로 표현하여 단어간의 간선에 가중치를 부여하고 최대 신장트리를 이용하여 가장 높은 점수의 트리를 찾아내는 방법을 제안하였다. 문장의 전체를 고려하여 가중치가 높은 트리를 구성한다는 점이 장점이지만 트랜지션 방법에 비해 속도는 느리다. 이후에 온라인 학습 방법[20]과 Esiner[21] 알고리즘에서 High-order 자질을 사용한 논문[22] 등 그래프 기반의 여러 방법을 제안하였다.

그래프 기반의 구문분석의 큰 특징은 전역적 학습 및 추론을 한다는 점이다. 구문 분석에서 전역적 학습 방법의 의미는 문장의 모든 어절 사이의 관계를 고려하여 분석을 하는 방법이다.



<그림 II-1> 그래프 기반 구문분석 예제

<그림 II-1>는 그래프 기반 구문분석 예제이다. 왼쪽 그래프와 같이 4개의 노드에 대해 모든 아크를 연결한다. 아크의 가중치는 학습 모델로서 얻게 된다.

한국어는 문장의 마지막 어절이 루트가 되지만 영어에서 문장 중간의 노드가 루트가 될 수 있기 때문에 따로 루트 노드를 만든다. 화살표 방향은 지배소에서 피지배소 방향으로 표시되어 있다. 루트는 피지배소가 되는 노드가 없기 때문에 화살표는 한 방향만 존재하고 이외의 노드는 양방향으로 아크를 생성한다.

최종적으로 모든 아크의 가중치를 고려하여 최대 신장트리 알고리즘(Maximum Spanning Tree)을 이용하여 오른쪽 그래프와 같은 결과를 얻게 된다. 다시 말해 현재 구성되어있는 아크 중 가장 가중치가 높은 아크 집합을 찾는 것이다.

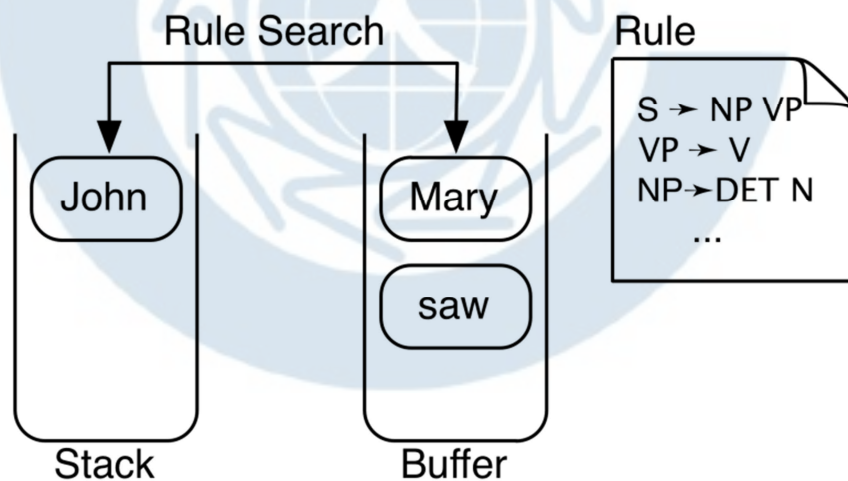
최대 신장트리 알고리즘을 사용하기 때문에 각 노드에 연결된 모든 아크를 고려하게 된다. 그러므로 구문분석 시 다른 방법에 비해 문장 구조에 대한 고려를 많이 하게 되는 것이다.

하지만 그래프 기반 구문분석의 단점은 학습 시에도 <그림 II-1> 같은 그래프를 구성하고 자질을 생성하기 때문에 학습 코퍼스 양이 작다면 표현되는 자질의 양이 적어지게 되고 분석 성능 또한 떨어지게 된다. 자질을 생성시 현재 노드와 연결되어 있는 노드간에 자질을 생성하기 때문이다.

2. 트랜지션 기반 구문분석

그래프 기반 구문분석과 달리 트랜지션 기반 구문분석은 지역적 학습 방법이다.

트랜지션 방법은 Shift-reduce 방법으로서 간단한 문법을 적용하여 높은 구문분석 성능을 보였다. 두 개의 노드 사이의 관계에 따라 의존구조를 한번에 결정해 나가기 때문에 빠른 속도를 보장하였다. 하지만 트랜지션은 해당 노드 사이의 관계만을 고려하기 때문에 전체 문장의 구조를 반영하지 않는 것이 단점이다. 이러한 단점을 보완하기 위해서 만들어진 구조를 다시 검색하여 의존 관계를 재정의 하는 논문도 제안하였다[23]. 이외에도 SVM을 이용하여 두 노드 사이의 관계를 정의하는 연구가 있었다[11,24,25].



<그림 II-2> 트랜지션 기반 구문분석 예제

<그림 II-2>은 트랜지션 기반 구문분석의 기초 방법인 Shift-reduce의 예제이다.

두 개의 스택을 사용하여 ‘Shift’와 ‘Reduce’라는 연산을 함으로써 문장을 해석해 나간다. Stack은 입력문의 각 노드를 Shift 연산으로 옮겨 오는 것이고 Buffer는 분석할 노드를 저장하고 있다. 각 스택의 상위 노드 사이에 규칙이 적용되는지를 확인하면서 분석해 나가는 방법이다. 규칙이 적용된다면 Reduce 연산으로서 규칙의 상위 개념과 바뀌어지고 최종적으로 Buffer에 노드가 남지 않으면 분석을 완료한다.

트랜지션 기반 구문분석은 결정적(Deterministic) 구문해석 알고리즘이라 한다. 두 노드의 규칙을 찾을 때, 여러 규칙으로 정의 가능하다면 그 중에서 하나의 규칙만을 선택하는 것이다. 따라서 Buffer의 마지막 노드에 대해서 규칙을 검색하여도 일치하는 것이 없다면 구문분석 오류가 된다. 하지만 이전의 상태로 돌아가(backtracking) 다시 분석해 올 수 없음을 뜻한다.

반대로 그래프 기반 구문분석은 비결정적(Non-deterministic) 구문해석 알고리즘이다. 모든 노드를 연결하는 아크의 각각을 확인하는 것으로 구성할 수 있는 모든 후보를 생성 후, 가장 높은 가중치를 선택하는 것이라 할 수 있기 때문이다.

결정적 구문해석 알고리즘은 분석의 속도가 빠르다는 것이 장점이지만 단점으로 오류 전파의 문제가 있다.

트랜지션의 오류는 루트 노드와 멀어질수록 다시 말해 트리의 깊이가 깊어 질수록 성능이 높고 반대로 루트 노드와 가까워 질수록 성능이 떨어지는 특징이 있다.

즉, 수식하는 어절에 대해서는 비교적 성능이 높지만 문장 구조를 이루는 어절의 성능이 떨어진다는 의미이다.

가장 큰 원인이 바로 오류 전파이다. 분석 초기에 오류가 발생하면 문장의 구조를 잘못 판단하게 된다. 초기 오류가 이후의 단계에 영향을 주기 때문에 문장의 성능이 떨어지는 것이다. 반대로 그래프 방법에서는 노드와 연결되는 모든 노드들의 가중치를 계산하고 분석 구조를 잡아가기 때문에 문장의 성능이 트랜지션에 비해서 높다.

3. 통합 구문분석

위 두 방법은 오류의 패턴에도 차이가 있다. 트랜지션 방법은 지역적인 학습 방법을 이용하여 길이가 짧은 의존구조에 대한 성능이 높고 분석 처리 속도가 빠른 반면 그래프 방법은 전역적 학습 방법을 통하여 길이가 긴 의존구조의 성능이 높다.

본 연구에서는 트랜지션 기반의 방법을 기초로 한 구문분석에 그래프 기반 방법의 장점을 통합한 구문분석을 제안한다.

이전 통합모델[26] 연구에서는 트랜지션과 그래프 기반의 각 시스템 결과를 자질로 추가하여 성능을 높인 연구이다. 서로의 장단점을 통합한 것으로 그래프 기반 시스템에 트랜지션 기반의 구문분석 시스템 결과를 자질로 사용한 것의 성능이 높다.

그래프 기반 구문분석은 전역적 학습 방법을 사용하므로 전체 구조를 고려하는

것에 유리하고 트랜지션은 오류전파의 문제와 자질을 추가하여 기계학습으로서 모델을 생성하기 때문에 그래프 기반의 장점을 자질로서 확실하게 사용할 수 없었기 때문이다. 그래서 그래프 기반 시스템을 기준으로 하는 시스템의 성능이 높다.

본 연구에서는 트랜지션의 빠른 구문분석 처리 속도와 성능을 기반으로 하여 문장 구조를 표현하는 자질을 추가하고 성능을 보완하였다. 기존 연구와 달리 그래프 기반의 시스템 결과가 아닌 문장 전체의 관점으로서 자질을 생성하여 적용하도록 정의하였다.



제 III 장

다단계 구단위화를 이용한 한국어 의존구조 분석



본 장에서는 논문에서 해결하려는 문제를 정의하고 제약사항을 설명한다. 또한 제안 시스템의 각 부분에서 대해서 설명한다.

1. 문제정의

본 연구에서는 구문분석의 기본 단위를 어절 단위로 한다. 따라서 문장은 어절들의 집합이 된다.

즉 문장은 $S = \langle s_1, s_2, \dots, s_m \rangle$, 의존구조는 $D = \langle dep(1), dep(2), \dots, dep(m-1) \rangle$ 와 같이 표시한다. 여기서 $dep(i) = j$ 는 어절 s_i 가 어절 s_j 를 지배소로 갖는다는 것을 의미한다. 이러한 구조에서 기존의 연구에서와 같이 다음과 같은 세 가지의 제약을 D 가 만족한다고 가정한다.

1. 한국어는 지배소 후위 언어이다.

즉, 지배소는 피지배소보다 항상 뒤에 위치한다.

한국어는 수식어구가 수식하는 어절의 앞쪽에 위치한다. 반대로 영어는 앞쪽 또는 뒤쪽에 위치하기 때문에 지배소 후위 언어가 아니다.

예를 들어 ‘나는 빨간 꽃을 보았다.’ 라는 문장에서 ‘꽃을’ 수식하는 ‘빨간’ 어절이 앞쪽에 위치한다. 이처럼 모든 어절의 수식하는 어절이 앞쪽에 위치하므로 후위 언어라 한다.

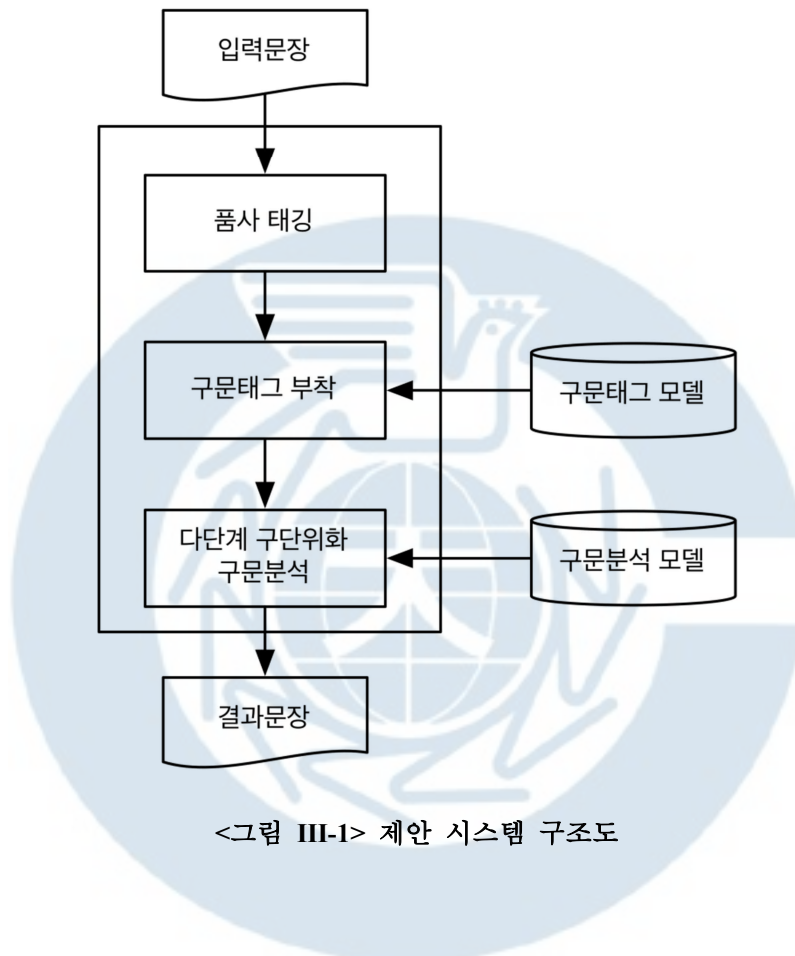
2. 교차 의존 구조는 없다.

‘빵을 칠수는 영희가 먹었다고 생각한다.’ 와 같은 문장은 처리대상에서 제외된다. 즉, ‘빵을’ 어절은 ‘먹었다고’를 지배소로 가지고 ‘칠수는’은 ‘생각한다’를 지배소로 가진다. 이 두 의존 구조 사이에 교차 구조가 생기는데 이를 제외한다.

3. 각 어절의 머리는 유일하다.

문장을 이루는 각각의 어절이 지배소를 하나씩만 가진다는 의미이다.

전체적인 구문분석 시스템은 품사 태깅과 구문태그 부착, 구문분석 이렇게 3개의 단계를 거치도록 설계하였다. 전체 시스템 구조는 <그림 III-1>과 같다.



2. 구문태그

구문태그는 문장에서 어절이 가지는 문장성분을 찾는 문제이다. <그림 III-2>는 구문태그의 예를 보여준다.

구문분석에 있어서 구문태그는 영향력이 큰 자질이다. 예를 들어 <그림 III-2>의 두 번째 어절인 ‘스포츠에’가 ‘NP_AJT’ 구문태그를 가지는데 구문표지인 ‘NP’는 체언구를 뜻하고 기능표지인 ‘AJT’는 현재 어절의 지배소에 영향을 받은 것으로 용언을 수식한다는 뜻이다. 구문태그 각각에 대한 자세한 설명은 부록에 실었다.

형태소 분석	구문태그
물론/MAG	AP
스포츠/NNG+에/JKB	NP_AJT
있/VV+어서/EC+도/JX	VP
이것/NP+은/JX	NP_SBJ
예외/NNG+가/JKC	NP_CMP
아니/VCN+다/EF+./SF	VP

<그림 III-2> 구문태그 예제

● 구문태그를 위한 자질

구문태그를 위한 자질의 기본은 형태소 분석기의 결과로서, 문장을 이루는 어절들의 모든 형태소 중에서 각 단계의 모델 생성에 있어 많은 영향을 주는 것을 선택하였다[28].

<그림 III-3>은 구문태그에 대한 자질 예이다. 구문태그는 5개의 자질을 사용하며, 자질들을 생성함에 있어서 기호에 대한 품사는 추가하지 않았다

자질1은 현재 어절의 첫 번째 형태소의 품사 자질이다. 자질3은 마지막 형태소

앞의 품사이며, 자질2는 자질1과 자질3의 자질 사이에 'XSA, XSV, VCP'¹ 중 하나의 품사가 있을 경우에 추가된다. 이것은 문장 분석에 있어 용언에 대한 영향을 고려하였을 때, 어절을 이루는 품사가 많을 경우 위의 자질이 추가되지 않는 것을 막기 위해 2번째 자질로 추가하였다. 자질4는 마지막 품사에 대한 자질이다. 여기에서 4번 자질은 어절을 이루는 형태소가 단일일 경우 1번 자질이 4번에도 추가된다. 이는 구문태그를 예측하는 모델 결과를 분석해보았을 때, 어미 또는 조사가 없는 어절에 대한 오류가 가장 많았다. 이를 위해 4번 자질에 반복 추가함으로써, 오류를 줄이고자 하였다.

형태소 분석	1	2	3	4	5
물론/MAG	MAG	-	-	MAG	-
스포츠/NNG+에/JKB	NNG	-	-	JKB	있/VV
있/VV+어서/EC+도/JX	VV	-	EC	JX	-
이것/NP+은/JX	NP	-	-	JX	-
예외/NNG+가/JKC	NNG	-	-	JKC	아니/VCN
아니/VCN+다/EF+./SF	VCN	-	-	-	-

<그림 III-3> 구문태그 자질의 예

그리고 5번째 자질은 다음어절 첫 번째 형태소에 대한 형태소와 품사로서 이루어진 자질이다. 다음 어절이 조용사(XSA, XSV)가 붙어서 형용사, 또는 동사가 되는 경

¹ XSV는 동사 조용사, XSA는 형용사 조용사, VCP는 긍정 지정사이다.

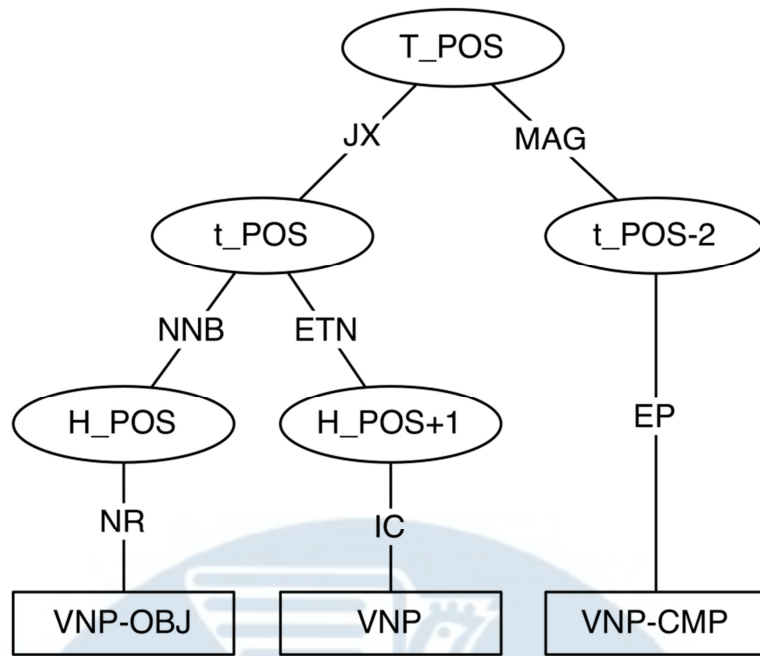
우에도 5번 자질에 추가하였다. 예를 들어 다음 어절이 ‘입장/NNG+하/XSV+ㄴ 다 /EF+./SF’ 일 경우 이전 어절의 5번 자질에는 ‘입장하/VV’가 추가된다.

생성된 기본 자질들은 다양한 조합을 통해서 새로운 자질로 만들 수 있다. 이 경우, 대부분의 연구에서는 연구자의 사전지식이 중요한 요소가 된다. 본 연구에서는 의사결정나무(Decision Tree)를 사용하여 최적의 자질 조합을 생성하였다.

<표 III.1> 기본자질

자질이름	자질내용
H_POS±n	어절의 첫 형태소의 품사 앞, 뒤 n개씩
T_POS±n	어절의 끝 형태소의 품사 앞, 뒤 n개씩
t_POS±n	어절의 끝에서 두 번째 형태소의 품사 앞, 뒤 n개씩
H_POS	자신의 첫 형태소 품사
T_POS	자신의 끝 형태소 품사
t_POS	자신의 끝에서 두 번째 형태소의 품사

최적의 자질 조합이란 <표 III.1>의 기본자질들을 다양한 조합을 통해서 새로운 자질을 만드는 것이다. 입력된 학습 코퍼스의 문장 별로 각 어절을 중심으로 n 개의 문맥정보를 정의하고 기본자질에서 생성되는 모든 조합을 의사결정나무에 입력하게 된다.



<그림 III-4> 의사결정나무

<그림 III-4>는 조합된 자질입력으로 생성된 의사결정나무의 한 부분이다. 의사결정나무의 결과를 이용하여 어절 구문태그 예측을 위한 최적 조합자질을 생성한다.

의사결정나무에서 뿌리(Root)에서 가까울수록 정보량이 많은 자질이 된다. 이 의사결정나무에서 깊이 우선탐색(Depth-first Search)으로 조합된 자질을 생성한다.

<표 III.2>은 조합된 자질들을 보여준다. <표 III.2>에서 위의 4개(1,2,3,4)는 조합된 자질들을 보여주고 아래의 3개(5,6,7)는 이 자질들에서 최상위 자질인 'T_POS'를 제거한 자질들을 보여준다.

<표 III.2> 조합된 자질들

번호	조합된 자질
1	T_POS
2	T_POS t_POS-2
3	T_POS t_POS H_POS
4	T_POS t_POS H_POS+1
5	t_POS-2
6	t_POS H_POS
7	t_POS H_POS+1

3. 다단계 구단위화 방법

다단계 구단위화(Cascaded Chunking) 방법은 [33]에서 영어를 위해 처음 제안되었다. 이 방법은 [34]과 [35]에 의해 일본어에 적용되어 좋은 성능을 보였다.

본 연구에서는 한국어의 특성에 맞게 이를 변형하여 적용한다[29]. <그림 III-5>는 다단계 구단위화의 과정을 보여준다. <그림 III-5>에서 ‘D’는 의존소에 대한 표시이다. 각 단계에 ‘D’ 표지를 부착할 수 있는 어절은 바로 다음 어절이 지배소일 경우이다. 1단계에서 ‘닭과’, ‘모든’, ‘한’, ‘자리에’에 ‘D’ 표지가 부착되었다. 그렇지 않은 어절에는 ‘-’ 표지를 부착한다.

그리고 다음단계로 넘어갈 때 앞의 어절 표지가 ‘-’이고 현재 어절의 표지가 ‘D’인 경우 삭제된다. <그림 III-5>에서 1단계 ‘자리에’ 어절이 삭제되지 않은 이유는 ‘한’ 어절이 의존소 표지 ‘D’를 가지고 있기 때문이다.

예를 들어 두 번째 단계에서는 ‘닭과’, ‘모든’, ‘한’ 어절이 삭제되어 ‘달걀의 것이

자리에 모였다.’의 문장에 대해 구문분석을 다시 수행한다. 이런 과정은 한 어절이 남을 때까지 반복한다.

일반적인 구문분석 방법의 시간 복잡도가 $O(n^3)$ 임에 비하여 구단위화 기법은 $O(n^2)$ 이므로 매우 빠르다. 또한 구문 요소의 결합이 아니라 레이블링 문제이므로 입력 문장에 대해서 매우 강건하다. 즉, 분석되지 않은 문장이 없다.

문장 단계	닭과	달걀의	모든	것이	한	자리에	모였다.
초기	-	-	-	-	-	-	-
1 단계	닭과 D 삭제	달걀의 -	모든 D 삭제	것이 -	한 D 삭제	자리에 D	모였다. -
2 단계		달걀의 D 삭제		것이 D		자리에 D	모였다. -
3 단계				것이 D 삭제		자리에 D	모였다. -
4 단계						자리에 D 삭제	모였다. -
5 단계							모였다 - 종료

<그림 III-5> 다단계 구단위화를 이용한 구문분석의 예

의존관계를 위한 자질은 각 어절에서 기본적으로 4 개의 자질을 기반으로 사용한다. 그리고 추가적인 자질은 이후 장에서 다룬다. <그림 III-6>은 자질 예이다.

자질들을 생성함에 있어서 기호 또는 종결어미 품사는 추가하지 않는다. 단 기호 중에서 구문분석 시 주요 자질이 될 수 있는 것은 추가한다. 예를 들어 ‘,’와 따옴표(“”)와 같은 기호는 추가한다.

자질 1 은 현재 어절의 첫 번째 형태소의 품사 자질이다. 자질 2 는 마지막 형태소 품사이다. 단, 첫 번째 형태소가 ‘NNG’ 또는 ‘XR’이면서 두 번째 형태소가 ‘XSA, XSV, VCP’인 경우 동사 자질로 수정하였다. 예를 들어 어절의 형태소 분석이 ‘입장/NNG+하/XSV+다/EF+./SF’인 경우 ‘NNG+XSV’이므로 ‘입장하/VV+다/EF+./SF’로 어절을 변경하여 자질을 추가한다. 그래서 자질 1 에는 ‘VV’ 자질이 들어가고 자질 2 는 ‘.’가 된다. 구문태그 결과만을 자질로 사용하였을 경우 구문태그 자질의 오류에 대해서 구문분석 예측 확률이 낮아 품사에 대한 자질을 추가하였다.

형태소 분석	1	2	구문태그	
			구문표지	기능표지
물론/MAG	MAG	-	AP	-
스포츠/NNG+에/JKB	NNG	JKB	NP	AJT
있/VV+어서/EC+도/JX	VV	JX	VP	-
이것/NP+은/JX	NP	JX	NP	SBJ
예외/NNG+가/JKC	NNG	JKC	NP	CMP
아니/VCN+다/EF+./SF	VCN	-	VP	-

<그림 III-6> 구문분석 자질의 예

4. Conditional Random Fields

기계학습은 크게 비지도 학습(Unsupervised Learning)과 지도 학습(Supervised Learning)으로 나뉘지는데 CRFs는 지도학습에 속한다[30]. 레이블이 부착된 학습 데이터에서 노드를 대표할 수 있는 자질(Feature)을 추출한다. 자질은 학습하는 데이터의 목적에 맞게끔 연구자의 직관에 따라 작성하는 것으로 각 노드에 1개 이상의 자질을 정의하며, 자질에 따라 성능의 차가 크다.

입력열 $X = x_1 x_2 \dots x_n$, 상태열 $T = t_1 t_2 \dots t_n$ 이 주어지고 가중치 $\Lambda = \{\lambda \dots\}$ 가 주어졌을 때, CRFs에서는 조건 확률로 식 (1)과 같이 정의된다.

$$P(T|X) = \frac{1}{Z(X)} \exp \left(\sum_{i=1}^n \sum_k \lambda_k f_k(t_{i-1}, t_i, x, i) \right) \quad (1)$$

여기서 $Z(X)$ 는 확률 값으로 만들어 주는 정규화 값이고 $f_k(t_{i-1}, t_i, x, i)$ 는 자질 함수이다. 예를 들어 자질함수는 식 (2)와 같이 정의 할 수 있다.

$$f_1(t_{i-1}, t_i, x, i) = \begin{cases} 1, & \text{if } t_i = \text{PERSON and } x_{n-1} = \text{Mr.} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

f_1 자질 함수를 적용하면 현재 노드를 기준으로 앞의 노드가 'Mr.'이면서 현재 노

드의 t_i 이 'PERSON'이라면 자질 값에 1을 주고 이외에는 0값을 가지게 된다. 이렇게 f_k 개의 자질 함수를 정의하고 값을 정하게 된다.

또한 λ_k 는 각 자질에 대한 가중치를 나타낸다. k 는 k 번째 자질이며, 자질 함수는 현재 시간에 대해 관측열 x_i , 상태변이 $t_{i-1} \rightarrow t_i$ 에 대해서 전이의 양상을 측정할 수 있다. 매개변수들은 주어진 입력열과 이에 대응하는 상태열에 대한 조건부 확률이 최대화하는 최대 우도(Maximum Likelihood)에 의해서 추정된다. 훈련 집합에 대해서 다음과 같은 로그 유사도(Log-likelihood)를 계산한다.

$$L(\Lambda) = \sum_l \log P_{\Lambda}(t_l|x_l) = \sum_l \left(\sum_{i=1}^n \sum_k \lambda_k f_k(t_i, x, i) - \log Z_{x_l} \right) \quad (3)$$

식 (3)를 최대로 하도록 학습한다. 일반적으로 CRFs는 IIS(Improved Iterative Scaling)나 GIS(Generalized Iterative Scaling)[31]를 사용하여 학습한다.

또한 학습 데이터의 과적합(Overfitting) 문제를 해결하기 위해서 가우스 사전 평활[32]을 적용한다.

본 연구에서 CRF++²을 사용하였다. 학습에 사용한 자질은 <그림 III-6>의 기본 자질과 실험에 맞게 자질을 추가하여 사용한다.

² <http://crfpp.googlecode.com/svn/trunk/doc/index.html>

5. 비어휘 다단계 구 단위화 시스템

구문분석은 대용량의 정보를 이용하여 서버에서 처리되는 것이 일반적이다. 그렇지만 모바일 환경이 일반화된 현재에서 모바일 단말기에서 구문분석 요구도 증가하였다. 본 장에서는 모바일 단말기에서도 처리가 가능한 비어휘 모델에 대해서 설명한다. 본 모델은 어휘 정보를 사용하지 않기 때문에 매우 작은 모델로 받아들일 수 있는 성능을 보인다.

비어휘 모델은 <그림 III-6>과 같이 구문분석 자질에 어휘자질을 사용하지 않고 비어휘로 자질을 구성하여 모델 사이즈를 줄이면서 도메인 의존성을 낮추었다.

5.1. 실험환경

먼저 실험환경에 대한 설명이다. 수정된 세종 코퍼스 61,533문장을 10-Fold Cross-validation을 수행하였다. 한 문장의 평균 길이는 약 11이다.

제안한 시스템의 성능 평가를 위해 아크-정확도와 아크-재현율을 결합한 $F_1 - measure$ 와 문장 정확도(Exact-Matching)를 사용하였다. 평가 척도는 식 (4)와 같다. 본 연구에서는 구문 분석을 레이블링 문제로 해결하기 때문에 아크-정확도와 아크-재현율이 같다.

$$\begin{aligned}
& \text{아크 - 정확도}(\text{Arc Precision}, AP) \\
&= \frac{\text{구문 분석 파스트리에서 올바른 아크의 수}}{\text{구문분석 파스트리에서 모든 아크의 수}}, \\
&= \text{아크 - 재현율}(\text{Arc Recall}, AR) \\
&= \frac{\text{구문 분석 파스트리에서 올바른 아크의 수}}{\text{정답 파스트리에서 모든 아크의 수}}, \quad (4) \\
&F_1 - \text{measure} = \frac{2 \cdot AP \cdot AR}{AP + AR}, \\
&\text{Exact - Matching} = \frac{\text{정확히 분석된 문장의 수}}{\text{문장의 수}}
\end{aligned}$$

본 논문에서는 어절과 문장 성능으로 각각 평가 하였다. 어절의 성능은 식 (4)와 같이 각 어절에 대해서 지배소가 올바르게 연결되었는지를 확인한다. 문장 성능은 문장의 모든 어절의 지배소가 올바르게 연결된 경우 문장 정확도가 높아진다.

그리고 구문분석의 정확도 평가 방법에는 두 가지가 있다. LAS(Labeled Attachment Score)와 UAS(Unlabeled Attachment Score)이다. 본 논문에서는 UAS로 평가 한다.

예를 들어 <그림 1-1>과 같이 5개의 어절로 이루어진 문장에서 아크는 각 어절 간의 화살표로 표시된 것이고 아크의 관계는 화살표에 표시된 구문태그 이다.

‘철수/NNP+는/JX’ 어절은 ‘가/VX+있/EP+다/EF+./SF’ 어절과 아크가 생기고 아크의 관계는 ‘NP_SBJ’가 된다. 두 가지 평가 방법에 표시된 ‘Label’은 아크 관계를 뜻한다.

그러므로 LAS방법은 구문분석 정확도를 평가할 때 어절 사이의 아크가 올바른지, 아크의 관계가 정확한지를 확인하는 것을 의미한다. UAS는 아크만 올바르게 연결되

어도 정확도가 높아진다.

구문분석에서 비중이 높은 자질 중 하나가 구문태그이다. 하지만 구문태그에 오류가 있다고 해서 아크가 무조건 오류인 것은 아니므로 LAS와 UAS의 값의 차이가 난다.

5.2. 실험 결과

비어휘 시스템의 결과는 <표 III. 3>과 같다. 평균 6,153문장을 테스트 하였으며 모델크기는 평균 2.3MB이다. 모델의 크기가 작으면서 어휘 자질을 사용하지 않기 때문에 언어에 의존적이지 않다.

<표 III. 3> 비어휘 시스템 결과

	문장 수	어절 성능	문장 성능
1	6,154	86.25	40.36
2	6,154	86.23	39.28
3	6,154	86.35	40.48
4	6,153	86.18	38.65
5	6,153	86.26	39.69
6	6,153	86.61	40.83
7	6,153	86.25	40.22
8	6,153	86.12	38.60
9	6,153	86.38	39.88
10	6,153	86.32	40.13
평균		86.30	39.81

5.3. 오류 분석

기본이 되는 비어휘 시스템에서 발생하는 오류를 분석하여 IV장부터 이를 극복하기 위한 자질을 제안하고 효과를 실험 결과로서 보이고자 한다.

- 기능표지가 없는 어절 오류

구문태그에서도 기능표지는 지배소를 결정하는데 중요한 자질이 된다. 이런 기능표지가 없는 어절은 어미 또는 조사가 없는 어절과 같다.

(1) 도춘방 전국부녀연합회 부녀연구중심 부소장은 전한다

어절에 어미 또는 조사가 없는 연속된 명사에서도 오류가 많이 발생한다. 문장 (1)에서 ‘도춘방 전국부녀연합회 부녀연구중심’은 기능표지가 없이 명사만 연결되어 있다. 이 어절들의 지배소는 어절의 순서에 따라 또는 문맥에 따라 수식구조가 달라진다. <그림 III-7>에서 2번 어절과 3번 어절이 같은 자질 구조를 가지고 있다. 분석 시 현재 어절 자질 뿐만 아니라 문맥정보를 사용하지만 인접한 어절이므로 문맥정보도 비슷하다. 그렇다면 1번 어절이 지배소를 결정할 때 2번 어절을 지배소로 가질 수 있는 확률이 높아진다.

#	형태소 분석	1	2	3	구문태그	지배소
1	도춘방/NNP	NNP	-	NP	-	4
2	전국/NNG+부녀/NNG+연합회/NNG	NNG	NNG	NP	-	3
3	부녀/NNG+연구/NNG+중심/NNG	NNG	NNG	NP	-	4
4	부소장/NNG+은/JX	NNG	JX	NP	SBJ	5
5	전하/VV+ㄴ 다/EF+./SF	VV	-	VP	-	5

<그림 III-7> 기능표지가 없는 어절 예

- 대등구 또는 나열형 문장의 오류

(2) 컴퓨터공학과를 졸업한 박씨는 한국 전산인협회 사무국장, 한국대 창원대 개론 강의를 맡아 왔다.

문장 (2)와 같이 ‘,’ 또는 ‘나, 와, 랑’이 포함된 문장은 나열형인지 대등구 문장인지 문장의 구조를 파악하여 분석해야 정확한 분석이 가능하다.

나열형의 문장인 경우 나열되는 어절간에 지배소 관계가 형성 되어야 하지만 대등구인 경우 두 문장이 합쳐지는 구조를 형성해야 하므로 오류 발생률이 높다.

- 복잡한 문형 오류

문장에 주어와 동사가 여러 개 있는 문장은 복잡도가 높아지므로 분석 오류가 많다. 중심축이 되는 주어와 동사의 연결에 오류가 생긴다면 문장의 전체적인 분석에 오류가 생길 가능성이 높아진다.

#	형태소 분석	구문태그	정답	시스템
1	나/NP+는/JX	NP_SBJ	7	3
2	과자/NNG+를/JKO	NP_OBJ	3	3
3	먹/VV+은/ETM	VP_MOD	4	4
4	영희/NNP+가/JKS	NP_SBJ	6	6
5	집/NNG+에/JKB	NP_AJT	6	6
6	가/VV+왔/EP+다고/EC	VP_CMP	7	7
7	생각/NNG+하/XSV+왔/EP+다/EF+./SF	VP	7	7

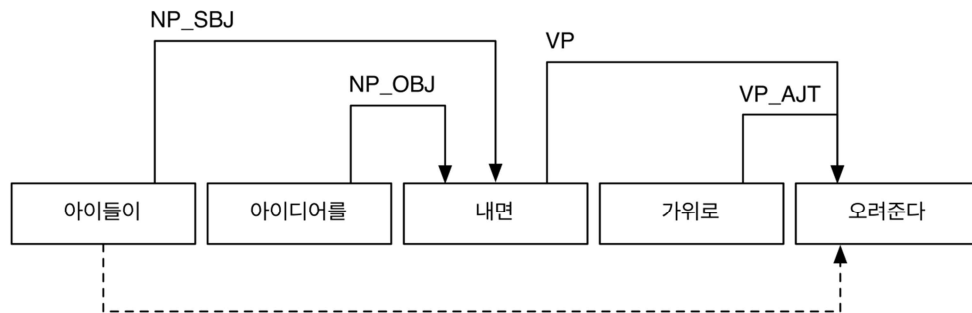
<그림 III-8> 복잡한 문형 예

<그림 III-8>에서 문장의 첫 번째 주어인 1번 어절은 7번 어절을 지배소로 가진다. 하지만 시스템 결과는 3번 어절을 지배소로 분석한다. 2번 어절이 제거되고 나면 1번 어절과 3번 어절이 인접하게 되는데 문장 전체를 확인할 수 없기 때문에 3번 어절을 지배소로 가지게 되어 해당 단계부터 오류가 발생하게 된다.

6. 공기정보(Co-occurrence)를 이용한 의미제약

본 장에서는 공기정보를 이용한 구문분석 방법에 대해 설명한다. 공기정보 시스템은 비어휘 다단계 구단위화 방법에서 발생하는 오류를 해결하기 위하여 의미정보를 도입하는 것이다.

앞의 어절의 표지가 ‘-’ 이면서 현재 어절의 표지가 ‘D’ 인 어절을 제거하며 분석할 경우, 단어들 간의 의미정보 제약을 줄 수 있는 방법이 없다.



<그림 III-9> 의미정보가 필요한 문장 예

예를 들어 <그림 III-9>의 문장에서 ‘아이들이’ - ‘내면’ - ‘오려준다.’ 어절이 연결되는 것인지 ‘아이들이’ - ‘오려준다’ 인지에 대한 판단은 의미정보를 파악해야 좀더 정확한 분석이 가능하다.

이러한 문제를 해결하는 방법은 의미정보를 줄 수 있는 Thesaurus 또는 Ontology 등을 구축하는 것인데 이는 시간과 노력이 많이 들어가는 작업이다. 본 논문에서는 의미정보에 대한 해결책으로 공기정보를 사용한다.

6.1. 공기 정보

조사와 어미가 없는 어절의 경우 문맥자질이 비슷하여 발생하는 오류를 공기정보를 추가하여 어휘자질의 효과를 주고자 한다. 공기정보는 구문코퍼스에서 피지배소와 지배소 쌍으로 생성한다. 단, 쌍의 각 어절은 동사와 의존명사가 포함되어 있지 않으며 조사 또는 어미가 없는 어휘 정보만을 잘라 구성한다.

#	형태소 분석	1	2	구문태그		공기 정보	지배소
				구문표지	기능표지		
1	그러나/MAJ	NP	JX	AP	-	0	9
2	이/NP+들/XSN	NNG	JKO	NP	-	0	4
3	캐릭터/NNG	VV	ETM	NP	-	1	4
4	상품/NNG	NNP	JKS	NP	-	1	5
5	대부분/NNG+이/JKS	NNG	JKB	NP	SBJ	0	6
6	외국/NNG+산/XSN +이/VCP+라/EC+ㄴ/ETM	VV	EC	VNP	MOD	0	7
7	점/NNG+은/JX	VV	EP	NP	SBJ	0	9
8	문제/NNG+로/JKB	NNG	JKB	NP	AJT	1	9
9	지적/NNG+되/XSV+ㄴ다 /EF+./SF	VV	-	VP	-	0	9

<그림 III-10> 공기정보 예제

<그림 III-10>의 문장에서 3~5번째 어절인 ‘캐릭터 상품 대부분이’에 대한 공기 정보는 ‘캐릭터-상품’과 ‘상품-대부분’이 된다. ‘캐릭터’가 ‘상품’ 어절을 지배소로 가지고 있고, ‘상품’이 ‘대부분이’라는 어절을 지배소로 가지기 때문에 이러한 쌍을 구성하게 된다.

이렇게 정의한 공기정보는 <그림 III-10>과 같이 정의된 자질 4개(자질1, 자질2, 구문표지, 기능표지) 이외에 한 개의 공기 정보 자질을 추가하여 공기정보 값에 따라 자질을 부여 받아 학습하게 된다.

공기정보는 빈도 또는 클래스로 학습한다. 클래스로 할 경우 현재 어절의 어휘와 지배소 어절의 어휘가 공기정보 값이 있으면 1, 그렇지 않으면 0값으로 자질을 부여

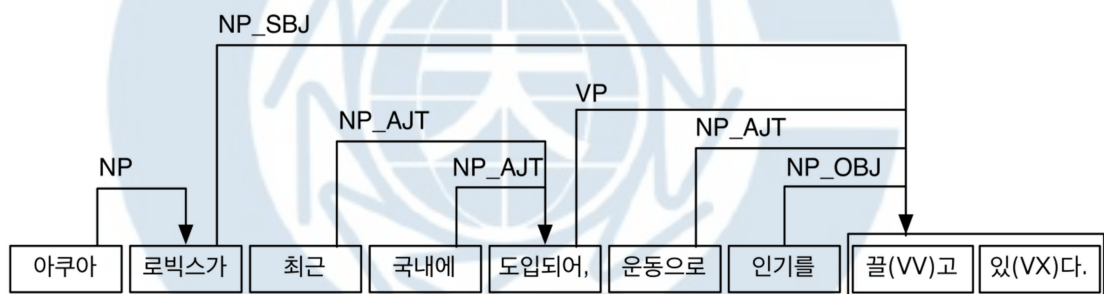
받는다. 다음 실험에서 자세히 설명한다.

6.2. 실험 결과

● 기본 실험

기본 실험은 <그림 III-6>에서 정의한 것을 학습하여 분석한 것을 말한다. 단, 분석 시 동사 관련하여 몇 가지를 처리하고 분석한다.

첫 번째는 보조동사, 본동사 관련이다. 문장에서 동사가 많을수록 문장 복잡도는 높아지고 문장의 성능이 떨어지므로 구와 같이 묶어서 처리하도록 수정하였다.



<그림 III-11> 동사 묶음 처리 예

<그림 III-11>와 같이 ‘끝/VV+고/EC’, ‘있/VX+다/EF+./SF’ 두 어절을 묶음으로 하여 처리한다. 두 어절을 묶음으로서 자질도 통합하도록 하였다.

지배소와 관련되는 기능표지가 중요하므로 ‘끌고’ 어절의 기능태그를 ‘있다’ 어절의 기능태그로 교체하였다. 반대로 구문표지는 ‘끌고’ 어절의 구문표지를 그대로 사

용한다.

<그림 III-11>에서 ‘끌고 있다’는 같은 구문태그이기 때문에 변화가 없지만 ‘나와 /VV+아야/EC’(VP) + ‘하/VX+ㄴ다고/EC’(VP_CMP) 처럼 구문태그가 다를 경우 묶음이 되면서 ‘나와/VV+아야/EC+하/VX+ㄴ다고/EC’(VP_CMP)가 된다.

문장에 이런 묶음이 많아지면 노드의 수가 줄어들게 되고 보조동사를 지배소로 가지는 오류가 사라지게 되므로 좀더 효율적인 분석이 가능하다.

동사 묶음 이외에도 구 묶음의 패턴을 이용하여 노드 수를 줄이고자 하였다. 이것은 고정된 의존구조 형식을 가진 패턴을 묶음으로서 노드 수를 줄여 분석의 효율을 높이하고자 한다.

추가된 패턴은 앞의 어절에 ‘ㄴ’ 또는 ‘을/를’이 있고 현재 어절에는 ‘NNB’ 품사, 다음 어절 첫 형태소가 ‘있/없’인 경우 세 어절을 묶음으로 하였다.

이 또한 묶음이 되면서 자질 또한 변경이 된다.

<표 III. 4>는 기본 실험의 성능을 나타낸다. 비 어휘 시스템에 비해서 어절 성능이 0.1정도 떨어진다. 하지만 묶음 처리가 구문분석에서 효율적으로 활용되기 때문에 이 시스템을 기본으로 다른 실험을 진행한다.

<표 III.4> 기본 실험 성능

	문장 수	어절 성능	문장 성능
1	6,154	86.2	40.32
2	6,154	86.21	39.58
3	6,154	86.36	40.62
4	6,153	86.08	38.83
5	6,153	86.22	39.61
6	6,153	86.62	40.7
7	6,153	86.05	40.11
8	6,153	86.24	38.71
9	6,153	86.35	39.67
10	6,153	86.29	39.75
평균		86.26	39.79

- 기능어 형태소 정보 추가 실험

기본 실험은 어휘자질을 사용하지 않은 모델이다. 비 어휘 모델은 분석 모델 크기가 어휘 모델에 비해 현저히 작으며 성능도 크게 떨어지지 않는다.

하지만 어휘 자질은 조사 또는 어미가 없는 어절에 대해서 성능을 높일 수 있다. 어휘자질 없이는 품사열과 문맥자질이 비슷할 가능성이 높기 때문에 의존구조를 정하기가 쉽지 않다. 또는 어휘 자질을 씌으로써 문장의 올바른 구조를 잡는데 도움을 준다.

본 연구에서는 어절의 모든 어휘자질을 넣는 것은 모델사이즈가 너무 커지고 자질 데이터 부족(Sparseness) 문제로 성능이 떨어질 수 있으므로 기능어 형태소 정보를 추가하였다. 기능어 형태소 정보는 어절의 마지막 어미 또는 조사 어휘를 의미한다.

한국어는 조사와 어미가 발달한 언어로 ‘나는 밥가 먹다.’와 같은 문장에서 ‘가’와 같이 정문에서 문맥상 어울리지 않는 조사, 어미가 존재하지 않는다. 이러한 것을 자질에 추가함으로써 연결되지 않아야 할 의존구조를 보완하고자 하였다.

<표 III. 5>는 기능어 형태소 정보를 추가한 실험의 성능을 나타낸다. 기본에 기능어 형태소 정보를 추가한 것으로 성능이 향상함을 알 수 있다.

<표 III. 5> 기능어 형태소 정보 추가 실험 성능

	문장 수	어절	문장
1	6,154	86.79	42.74
2	6,154	86.69	41.01
3	6,154	86.67	41.44
4	6,153	86.43	40.03
5	6,153	86.28	40.01
6	6,153	86.88	41.93
7	6,153	86.61	42.08
8	6,153	86.6	40.44
9	6,153	86.71	41.9
10	6,153	86.16	40.29
평균		86.58	41.19

- 공기정보 자질 추가

10-Fold Cross-validation이므로 9 Fold를 학습 및 공기정보 구축에 쓰고 1 Fold를 평가한다. 공기정보를 문서에 나오는 쌍의 빈도로 구성하는데 임계값(3) 이상의 빈도 값을 피지배소 어절의 자질에 대입할 경우 성능은 <표 III. 6>와 같다. 임계값을 설정한 것은 노이즈를 제거하기 위한 것이다.

<표 III. 7>는 빈도 값을 자질에 추가하지 않고 임계값 이상인 쌍에 대해서 클래스로 나누어 학습하였다. 클래스는 0 또는 1로 구분한다. 0은 공기정보 데이터에 없다는 의미로 자질에 0을 붙이고 공기정보 데이터에 임계값 이상의 값이 존재한다면 자질에 1을 부착한다.

<표 III. 7>의 성능이 더 좋은 이유는 CRFs에서는 자질의 숫자 값을 가중치로 학습하지 않고 하나의 문자열로 취급하기 때문에 클래스로 학습한 것의 성능이 더 높다. 그러므로 이후의 공기정보 자질은 클래스로 학습한 것을 사용한다.

<표 III.6> 공기정보 빈도 학습


	문장 수	어절 성능	문장 성능
1	6,154	86.74	42.27
2	6,154	86.36	40.62
3	6,154	86.5	41.32
4	6,153	86.42	40.11
5	6,153	86.38	40.27
6	6,153	86.82	41.57
7	6,153	86.35	41.28
8	6,153	86.5	40.14
9	6,153	87.15	42.92
10	6,153	86.21	40.06
평균		86.54	41.06

<표 III.7> 공기정보 클래스 학습

	문장 수	어절 성능	문장 성능
1	6,154	86.91	42.8
2	6,154	86.58	41.05
3	6,154	86.78	41.66
4	6,153	86.46	39.87
5	6,153	86.69	40.6
6	6,153	86.98	41.74
7	6,153	86.94	42.61
8	6,153	86.83	40.74
9	6,153	87.26	43.28
10	6,153	86.31	40.34
평균		86.77	41.47

제 IV 장

문장 구조 (Frame)



본 장에서는 기존의 다단계 구 단위화 방법에서 나타나는 오류를 보완하기 위해 추가한 자질에 대해 설명한다. 문장에서 주요 요소가 되는 어절을 정의하고 자질 (Frame)을 추가함으로써 문장 성능을 높이하고자 한다.

1. 문장 구조 자질의 필요성

앞서 설명한 것과 같이 구문분석은 문장 안에서 문장성분들의 관계를 찾는 과정이다. 자연어처리 분야 필수 단계 중 하나이며 상위 응용프로그램에서 구문분석에

대한 요구도 증가한다. 즉, 문장의 구조를 파악 함으로서 의미 있는 데이터를 추출하고자 하는 것으로 문장의 성능이 중요하다.

한국어는 지배소 후위 언어이므로 마지막 어절이 루트(Root)가 되고 루트를 지배소로 가지는 어절의 정확도가 높을수록 문장의 정확도가 높아진다. 이외의 어절들은 수식하는 역할의 어절일 확률이 높기 때문이다.

<그림 III-8>와 같이 1번 어절이 3번 어절을 지배소로 가지면서 의미적으로 ‘영희’가 아닌 ‘나’ 주체가 ‘과자’를 먹은 것이 되어버린다. 수식 어절의 오류도 물론 중요하지만 문장의 주요 구조의 오류가 발생하는 것은 상위 응용 프로그램에서 더 큰 문제를 야기 할 수 있다.

2. 필수격

문장에서 가장 핵심적인 성분은 서술어이다. 격 문법(Case Grammar)은 서술어를 바탕으로 어절의 의미적인 관계를 기술 하는 것이다. 격은 어절과 어절 사이의 의미관계를 뜻한다. 즉, 어절과 어절 사이의 관계가 ‘주격/목적격/...’과 같이 정의 하는 것을 격이라 한다.

격들은 핵심적 성분인 서술어가 필요로 하는 격들을 지정한 것이다. 그 중에서도 필수적인 관계에 대해서 정의 한 것을 필수격(Obligatory case)이라 하고 이외의 관계는 임의격(Optional case)이라 한다. 다시 말해 필수격은 서술어가 반드시 가져야 하는

격을 의미한다. 예를 들면 ‘가다’라는 동사는 ‘움직이다’는 뜻으로 해석될 경우 ‘~이 ~에/로 가다.’라는 필수격을 가지게 된다.

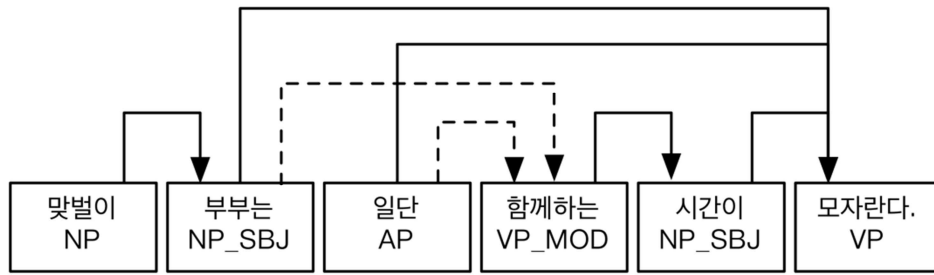
하지만 필수격은 문장 의미에 따라 격의 구조가 달라질 수 있으며, 서술어의 의미에 따라 달라진다. ‘가다’도 ‘경과하다’라는 의미일 경우 다른 필수격을 가지게 된다.

하지만 많은 수의 서술어 필수격을 문법으로 정의하고 적용하는 것은 어려운 문제이지만 필수격이 구문분석의 유용한 자질이라는 것은 틀림없는 사실이다.

본 논문에서는 기계학습 모델로서 필수격이 될 확률이 높은 어절을 정의하고 Frame 자질을 부여한다. 또한 Frame 자질을 추가함으로써 문장의 성능을 높이려 한다.

3. 다단계 구단위화의 문장 구조 자질

본 논문의 다단계 구단위화는 트랜지션 기반의 구문분석이다. 대신 <그림 II-2>와 같이 각 스택의 상위 노드간의 관계를 파악하기 위해 규칙을 검색하는 것과는 다르다. 규칙 대신에 CRFs로 학습한 모델을 적용하여 표지를 부착한 다음, 앞의 어절의 표지가 ‘-’이면서 현재 어절이 ‘D’ 표지인 어절을 제거 한다. 제거 하는 것이 곧 스택에 ‘Shift’ 연산이 되는 것과 같다.



<그림 IV-1> 오류 전파 예 1

그렇기 때문에 이것 또한 트랜지션의 오류 전파 문제가 존재한다. <그림 IV-1>는 오류 전파의 예를 표현하는 것이다. 실선으로 표시한 것은 분석되어야 할 정답을 표현한 것이다. 점선은 시스템 결과 중 오류만을 표시한 것이다.

‘함께하는’ 어절은 ‘시간이’라는 어절의 수식 어절이다. 하지만 ‘부부는’ 어절과 ‘일단’ 어절이 ‘함께하는’을 지배소로 가지면서 전체적인 문장의 구조가 달라진 것을 확인할 수 있다.

<그림 IV-2>는 <그림 IV-1>의 오류 전파 발생을 단계별로 자세히 보여주고 있다. ‘맞벌이 부부는 일단 함께하는 시간이 모자란다.’라는 입력 문장을 다단계 구 단위화로 분석해나가는 것을 나타내는 것인데 1, 2단계에서 오류가 발생하게 되어 전체 문장 의미가 다르게 진행되는 것을 표현하고 있다.

주격 역할을 하는 ‘부부는’ 어절과 부사인 ‘일단’ 어절은 동사를 지배소로 가지는 확률이 높다. 조사 또는 어미가 없는 명사 또는 체언 수식어인 ‘MOD’ 기능태그가 붙은 어절을 제외하고는 대부분의 어절이 동사를 지배소로 가질 확률이 높다.

그래서 예제와 같이 1,2 단계에서 바로 뒤 어절이 동사인 경우 앞의 두 어절 모두 인접한 동사인 ‘함께하는’ 어절을 지배소로 가지게 되는 것이다.

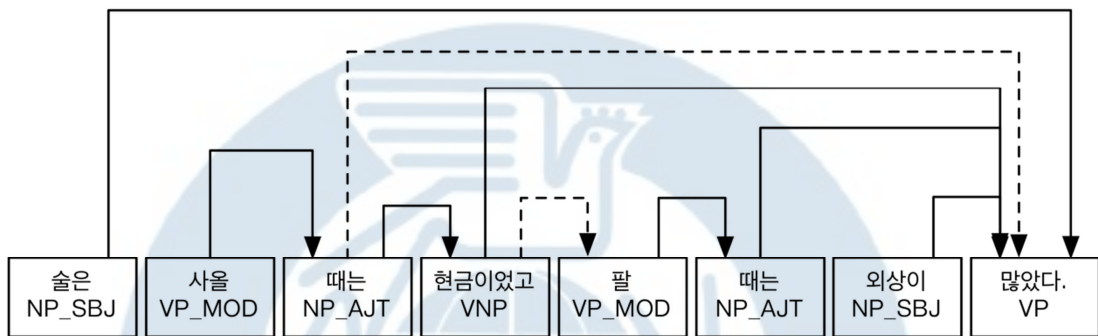
예제 입력 문장에서 오류전파로 인한 문장 의미가 달라지게 된다.

문장 단계	맞벌이	부부는	일단	함께하는	시간이	모자란다.
초기	-	-	-	-	-	-
1 단계	맞벌이 D 삭제	부부는 -	일단 D 삭제	함께하는 D	시간이 D	모자란다. -
2 단계		부부는 D 삭제		함께하는 D	시간이 D	모자란다. -
3 단계				함께하는 D 삭제	시간이 D	모자란다. -
4 단계					시간이 D 삭제	모자란다. -
5 단계						모자란다. - 종료

<그림 IV-2> 오류 전파 다단계 분석 예 1

다음과 같이 반대의 경우도 있다.

<그림 IV-3>에서 두 번째 ‘때는’ 어절이 ‘현금이었고’를 지배소로 가져야 정답이다. 하지만 <그림 IV-4>의 다단계 출력을 보면 1 단계에서 ‘때는’이 ‘.’ 표지이고 ‘현금이었고’는 ‘D’ 표지가 붙어서 ‘현금이었고’인 어절이 제거되게 된다. 첫 단계에서 지워져서는 안될 어절이 지워짐으로 해서 구문분석 구조가 바뀌게 된 것이다.



<그림 IV-3> 오류 전파 예 2

<그림 IV-2>와 <그림 IV-4>는 공통적으로 문장의 구조가 정답 구문분석 구조와 달라진 것을 보여주고 있다. <그림 IV-2>는 문장의 어절이 인접한 동사 어절을 지배소로 가질 확률이 높아서 발생하는 오류 패턴이었다. 그리고 <그림 IV-4>은 지배소가 될 수 있는 어절이 먼저 제거 됨으로써 단계에서 인접한 다른 어절에 오류를 전파하게 되는 예이다.

이러한 오류 전파는 각 분석 단계에서 어절의 지배소가 문장의 구조를 고려하여 옳은 것인지 판단하게 하는 자질을 추가함으로써 성능이 향상 될 수 있다.

문장 단계	술은	사울	때는	현금이었고	팔	때는	외상이	많았다.
초기	-	-	-	-	-	-	-	-
1 단계	술은 -	사울 D 삭제	때는 -	현금이었고 D 삭제	팔 D	때는 -	외상이 D 삭제	많았다. -
2 단계	술은 -		때는 -		팔 D 삭제	때는 D		많았다. -
3 단계	술은 -		때는 -			때는 D 삭제		많았다. -
4 단계	술은 -		때는 D 삭제					많았다. -
5 단계	술은 D 삭제							많았다. -
6 단계								많았다. - 종료

<그림 IV-4> 오류 전파 다단계 분석 예 2

이러한 오류 전파는 문장 길이가 길어질수록 발생할 확률이 높아지고 성능이 떨어진다. <표 IV.1>은 문장을 이루는 어절의 수로서 문장의 길이를 측정하여 해당 길이별 시스템 성능을 측정한 결과이다. 어절의 수가 많아질수록 즉 문장의 길이가 길어질수록 어절의 성능뿐 아니라 문장의 성능이 가파르게 낮아지는 것을 확인 할 수 있

다.

문장의 길이가 길어질수록 동사와 같은 격률 어절의 수가 많아지고 이에 영향을 받는 어절의 수도 많아진다. 이들 사이의 어절에 애매성을 판단하기 어려워지면 전체적인 문장 구조에 오류가 발생하는 것이다.

<표 IV.2>는 구문태그를 기준으로 의존 구조 길이의 통계를 확인한 것이다. 의존 구조 길이는 빈도를 기준으로 ‘2 이하’, ‘3 이상’의 백분율이다. 표의 왼쪽은 의존 구조 길이가 긴 구문태그의 정보이고, 오른쪽은 의존 구조 길이가 짧은 구문태그 나열이다. 구문태그는 ‘구문표지_기능표지’로 표현한다. ‘*_SBJ’는 기능표지가 ‘SBJ’인 것들의 모든 통계를 뜻한다.

<표 IV.1> 비어휘 시스템의 각 문장 길이별 성능

길이	1~9어절			10~19어절			20~29어절			30어절 이상		
	#	어절	문장	#	어절	문장	#	어절	문장	#	어절	문장
1	3,099	92.01	66.63	2,212	86.13	18.26	660	82.46	2.12	183	80.15	0.55
2	2,998	92.13	67.65	2,334	86.09	16.24	629	82.26	1.59	193	80.58	0.0
3	3,056	92.44	68.82	2,293	86.04	16.22	625	82.63	2.56	180	79.83	0.0
4	2,961	92.02	66.57	2,411	85.82	16.38	601	82.24	1.83	180	81.08	0.56
5	3,007	91.83	66.68	2,359	85.97	17.97	609	83.22	1.97	178	79.72	0.56
6	3,032	92.41	68.77	2,321	86.19	17.97	640	82.94	1.56	160	81.17	0.0
7	3,044	92.12	67.9	2,343	85.82	17.03	581	82.74	1.38	185	80.1	0.54
8	3,059	91.98	66.56	2,325	85.6	14.19	584	82.47	1.54	185	80.59	0.0
9	3,019	91.99	66.55	2,366	86.24	18.43	595	82.39	1.51	173	80.48	0.0
10	3,033	92.21	67.99	2,353	86.02	17.0	618	82.26	1.13	149	79.87	0.0

<표 IV.2> 구문태그의 의존구조 길이 통계

구문태그	빈도	의존구조 길이		구문태그	빈도	의존구조 길이	
		2 이하	3 이상			2 이하	3 이상
NP_INT	603	36.82	63.18	*_MOD	132,299	94.24	5.76
*_INT	612	36.93	63.07	VNP_MOD	7,178	94.36	5.64
VNP	6,165	46.16	53.84	VP_MOD	87,032	94.46	5.54
IP	1,397	52.97	47.03	NP	72,729	95.26	4.74
IP_*	1,407	53.09	46.91	*_CMP	10,656	95.99	4.01
VNP_AJT	405	53.58	46.42	VP_OBJ	2,767	97.07	2.93
NP_SBJ	84,263	57.25	42.75	DP_*	20,359	97.57	2.43
*_SBJ	85,675	57.65	42.35	DP	20,346	97.59	2.41
VP_AJT	1,548	59.82	40.18	NP_CMP	5,539	99.44	0.56

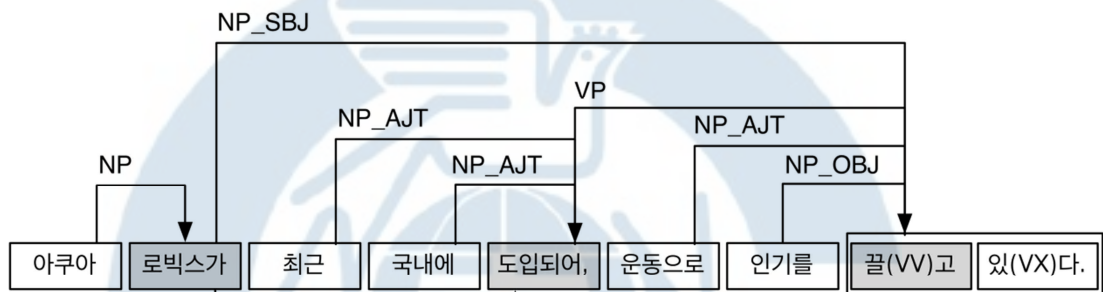
감탄사(IP), 독립어(INT), 부사(AJT), 주격(SBJ)과 같은 구문태그는 길이가 긴 반면 체언 수식어(MOD), 보어(CMP), 관형사(DP) 구문태그는 의존길이가 짧은 것을 알 수 있다. 문장의 구조를 표현할 수 있는 어절의 의존구조 길이가 길다는 것을 알 수 있다.

세종코퍼스의 구의 평균은 약 1.4어절이다. 평균적으로 약 2개의 어절이 구를 이룬다는 것이다. 어절이 구를 넘어서 지배소를 가진다는 것은 루트 노드와 좀더 가깝다는 것을 의미하며 문장 구조의 의미를 담고 있는 어절이 될 가능성이 높아진다.

<그림 IV-5>의 예제는 기존의 시스템에서 문장구조 자질을 추가할 경우 어떤 효과를 주게 되는지를 표현하였다. 문장에서 ‘로봇스가’ 어절이 ‘끌고’를 지배소로 가져야 정답이다. 하지만 시스템은 ‘도입되어,’를 지배소로 가져 오류가 발생한다. 이때

의존구조 길이가 긴 어절인 ‘로빅스가’와 ‘도입되어,’ 어절을 Frame으로 설정한다면 문장의 중요한 정보를 갖고 있는 어절을 알게 되므로 분석에 대한 오류를 줄여 줄 수 있다.

즉, 기존의 모델은 어절의 전부가 Frame인 것이고 제안하는 모델은 그 중에서 구문분석에 긍정적인 효과를 줄 수 있는 문장 구조 어절에만 Frame 자질을 추가하여 문장 분석 성능을 높이하고자 하는 것이다.



<그림 IV-5> Frame 예제

- 문장 구조 자질 정의

기본적인 구문분석 기법에는 여러 가지가 있다. 그 중에서 큰 범주로 보면 상향식 구문분석(Bottom-up Parsing)과 하향식 구문분석(Top-down Parsing)으로 나뉘어 진다.

하향식 구문분석은 개시기호(S)부터 시작하여 차례로 규칙을 적용하여 비 종단비호를 치환하는데 이것이 입력된 문장이 얻어질 때까지 반복하는 방법이다. 그리고

반대로 비 종단기호에서 적용되는 규칙으로 치환하여 개시기호가 얻어지면 분석이 완료되는 것을 상향식 구문분석이라 한다. 본 논문의 시스템은 상향식 구문분석이다.

종단 기호는 입력된 문장의 형태소가 되는 것이고 비 종단 기호는 형태소를 제외한 품사 또는 구문태그와 같은 기호들을 뜻한다. <그림 IV-6>에서 종단기호는 ‘I, girl, telescope, ...’ 등이고 비 종단 기호는 ‘S, NP, VP, ...’가 된다. 그리고 각 규칙은 오른쪽 또는 왼쪽 기호가 각각 왼쪽 또는 오른쪽 기호로 치환된다는 것을 말한다. (1)의 규칙은 개시기호 S는 N과 VP로 치환된다. 또는 반대로 NP와 VP는 S 기호로 치환됨을 말한다.

(1) S → NP VP	(8) N → I girl telescope
(2) NP → DET N	(9) V → saw
(3) NP → NP PP	(10) DET → the
(4) VP → V NP	(11) PREP → with
(5) VP → VP PP	
(6) PP → PREP NP	
(7) NP → N	

<그림 IV-6> 규칙 예

<그림 IV-6>규칙을 이용하여 “I/N saw/V the/DET girl/N with/PREP the/DET telescope./N”의 문장을 하향식 구문분석으로 한다면 <그림 IV-7>과 같다. 개시 기호부터 시작하여 왼쪽 비 종단 기호부터 치환해 가며 입력된 문장을 완성해 나가는 것이다. 오른쪽의 번호는 <그림 IV-6>에서 적용되는 규칙 번호를 뜻한다. 치환하는 기호의 위치에 따라 우측유도(Rightmost derivation) 또는 좌측유도(Leftmost derivation)라

고 한다. 유도 방향에 따라 결과가 달라지게 된다.

또한 어떤 시점에서 분석이 실패하게 되면 다른 치환 규칙이 적용되는 시점까지 되돌아 가서(backtracking) 분석을 계속하게 된다.

S	→	NP VP	(1)
	→	N VP	(7)
	→	I VP	(8)
	→	I V NP	(4)
	→	I V NP PP	(3)
	→	I saw NP PP	(9)
	→	I saw DET N PP	(2)
	→	I saw the N PP	(10)
	→	I saw the girl PP	(8)
	→	I saw the girl PREP NP	(6)
	→	I saw the girl with NP	(11)
	→	I saw the girl with DET N	(2)
	→	I saw the girl with the N	(10)
	→	I saw the girl with the telescopes	(8)

<그림 IV-7> 하향식 분석 예

(4)규칙이 적용된 4번째 결과에서 (7)규칙을 먼저 적용하게 되면 전체 문장의 분석이 이루어 지지 않기 때문에 다시 4번째 결과로 돌아와서 (3)규칙을 적용하게 되는 것이다.

하향식 구문분석은 개시기호로부터 문장을 생성하는 것과 같다. 분석이 완료될 때까지 규칙을 무한정 반복하기 때문에 일반적으로 분석 속도는 느리다. 하지만 하

향식 구문분석은 문장의 전체 구조를 세부 구조로 나누어 가면서 분석하는 방법이기 때문에 문장 구조 자질을 추출하는 것에 있어서는 하향식 구문분석과 같은 관점에서 설계해야 한다.

본 논문에서는 문장 구조 자질에 대해서 다음과 같은 실험을 한다.

- 문장 구조 자질 학습: 문장 구조 자질을 기존 실험 모델로서 이용하여 설정하고 자질에 추가하여 학습 및 분석하는 방법이다. 모델에서 부착되는 지표를 다른 처리 없이 사용한다.
- 분석 가이드: 문장 구조 자질 학습과 같이 기존 실험 모델로서 문장 구조를 설정하고, 문장 구조 자질을 고려한 분석 규칙에 따라 구문분석을 하는 방법이다.
- 확장 문장 구조 자질: 위의 문장 구조 모델은 기존의 지역적 학습 방법으로 생성된 자질이므로 하향식 구문분석의 관점으로서 문장 구조 자질 모델을 확장하여 실험한다.

4. Frame 설정

다단계 구 단위화는 각 단계마다 ‘D’ 또는 ‘-’ 표지를 부착한다. ‘-’표지는 적어도

의존구조 길이가 2이상인 것을 의미한다. 그렇다면 ‘-’는 Frame의 후보가 될 수 있다는 것을 의미한다.

따라서 첫 번째 단계에서 ‘D’ 표지 다음에 오는 ‘-’ 표지의 어절을 Frame이라고 설정한다. 문장의 구조는 다단계 구 단위화 단계를 진행하면서 변하는 정보가 아니므로 첫 번째 단계에서 Frame을 설정하는 것이다.

예를 들어 <그림 IV-5>의 첫 번째 단계를 표현하면 <그림 IV-8>과 같다.

‘로빅스가’ 어절은 앞 어절인 ‘아쿠아’가 ‘D’ 표지이면서 자신이 ‘-’ 표지이므로 Frame으로 설정된다. 마찬가지로 ‘도입되어,’와 ‘끌고 있다.’어절도 Frame이 된다. 문장의 끝부분의 ‘끌고 있다’는 구 패턴 묶음으로 처리되어 하나로 처리한다.

문장 단계	아쿠아	로빅스가	최근	국내에	도입되어,	운동으로	인기를	끌고	있다.
초기	-	-	-	-	-	-	-	-	-
1 단계	아쿠아	로빅스가	최근	국내에	도입되어,	운동으로	인기를	끌고 있다.	
	D	-	-	D	-	-	D	-	
		Frame			Frame			Frame	

<그림 IV-8> 다단계 구 단위화에서 Frame 설정 예

5. Frame 적용

5.1. 자질 학습

형태소 분석	1	2	구문태그		Frame
			구문표지	기능표지	
아쿠아/NNG	NNG	-	NP	-	0
로빅스/NNG+가/JKS	NNG	JKS	NP	SBJ	1
최근/NNG	NNG	-	NP	AJT	0
국내/NNG+에/JKB	NNG	JKB	NP	AJT	0
도입/NNG+되/XSV+어/EC+,/SP	VV	SP	VP	-	1
운동/NNG+으로/JKB	NNG	JKB	NP	AJT	0
인기/NNG+를/JKO	NNG	JKO	NP	OBJ	0
끝/VV+고/EC	VV	EC	VP	-	1
있/VX+다/EF+,/SF	VX	-	VP	-	0

<그림 IV-9> Frame 자질의 예

첫 번째 방법은 CRFs 자질에 Frame을 추가하는 것이다.

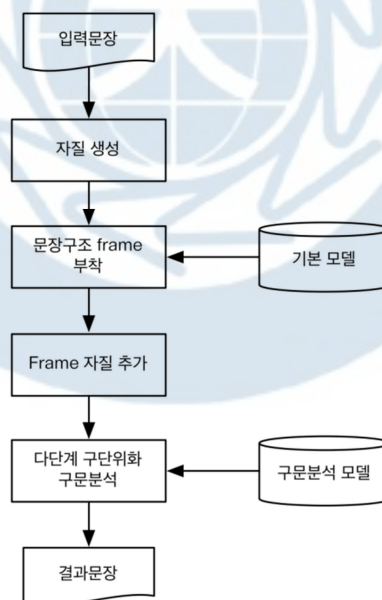
<그림 IV-9>와 같이 첫 번째 단계에서 설정된 Frame을 기본 실험 자질에 추가하여 CRFs를 학습한다. Frame인 것은 1 그렇지 않은 어절은 0을 넣어 학습한다.

기본 실험 자질은 Frame을 설정할 때 사용한 모델에서 정의한 자질을 뜻한다.

<그림 IV-9>은 비어휘 다단계 구 단위화 시스템의 결과에 Frame 자질을 추가한 것을 표현한다. 즉, 기본 실험 자질은 품사태그 자질 1~2, 구문태그(구문표지, 기능표지) 이렇게 4개의 자질에 Frame 자질을 추가한 것이다.

본 논문에서는 앞에서 설명한 자질을 추가하여 기본 실험 자질을 구성한다. 4개의 자질과 기능어 형태소 자질, 공기정보 자질까지 추가하여 6개를 사용한다. 그리고 6개의 자질 뒤에 Frame 자질을 추가하여 다단계 구 단위화 분석 시에는 총 7개의 자질을 사용한다. 분석 시 첫 분석 단계에서는 6개의 자질로, 이후의 단계에서는 7개의 자질로서 분석하게 된다.

자질 학습 방법 구조도는 <그림 IV-10>와 같다.

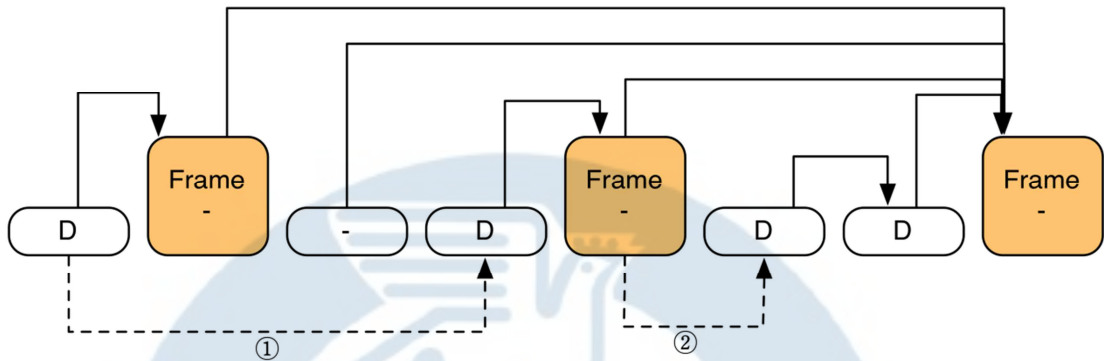


<그림 IV-10> 자질 학습 구조도

5.2. 분석 가이드

구문분석 시 Frame의 정보를 이용하여 제거되는 어절을 제한하는 것을 말한다.

먼저 몇 가지 Frame에 관한 규칙을 정의 하였다.



<그림 IV-11> Frame 특징

<그림 IV-11>은 Frame이 추가된 구문분석의 특징을 그림으로 표현한 것이다.

- (1) $t_i = 'D'$ 는 바로 뒤를 수식한다
- (2) $t_{i-1} = '-'$ and $t_i = 'D'$ 이면 'D'를 현재 단계에서 삭제한다.
- (3) $t_{i-1} = 'D'$ and $t_i = '-'$ 이면 '-'는 Frame이다.
- (4) Frame이 아닌 어절은 인접한 Frame 내부의 어절을 의존구조로 가지거나 외부 Frame만 가능하다.

<그림 IV-11>의 ①과 같이 인접한 Frame이 아닌 다른 어절을 지배 소로 가지므로 해당 의존구조는 허용되지 않는다.

(5) Frame은 Frame만 지배소로 가질 수 있다.

<그림 IV-11>의 ②와 같이 Frame이 다른 Frame이 아닌 어절을 지배소로 가지므로 허용되지 않는다.

<그림 IV-12> 구문분석 규칙

분석 가이드 방법 <그림 IV-12>에서 (1)~(5)까지 정의한 것을 위배하는 의존구조가 있을 시, 해당 의존구조는 무시하고 분석하는 것을 말한다.

6. Frame 실험

6.1. 문장 구조 설정

먼저 첫 번째 단계에서 Frame을 설정하는 성능을 측정하였다. Frame 성능 측정은 정확도(Precision), 재현율(Recall), F_1 - *measure* 로 확인하였다. 구문분석은 문장의 모든 어절에 레이블링을 해야 하므로 정확도와 재현율이 같지만 Frame은 모든 어절에 해당하는 것이 아니므로 각각에 대한 성능을 표기하였다.

Frame의 성능이 높을수록 문장의 구조 정보를 효과적으로 사용할 수 있으므로 성능이 매우 중요하다.

<표 IV. 3>은 4개 자질 기본 실험과 추가 자질(기능어 형태소 자질, 공기정보)을 추가한 6개 자질 기본 실험을 모델로 첫 번째 단계에서 Frame을 설정할 경우의 성

능을 나타낸다. 이것은 구문분석의 성능이 아닌 Frame의 성능만을 측정한 것이다. 4개 자질 실험에 비해 6개 자질 실험의 Frame 설정 성능이 높음을 알 수 있다.

Frame은 문장 구조 의미를 가진 자질이기에 때문에 분석 첫 단계에서 오류가 발생할 경우 오류 전파가 심각해진다. 그러므로 Frame의 성능이 전체 구문분석의 성능과 깊은 연관관계가 있다.

<표 IV.3> 4개 자질 실험과 6개 자질 실험의 Frame 설정 성능

	4개 자질 실험			6개 자질 실험		
	Precision	Recall	F1	Precision	Recall	F1
1	0.95	0.89	0.92	0.95	0.91	0.93
2	0.95	0.9	0.92	0.95	0.91	0.93
3	0.94	0.9	0.92	0.95	0.92	0.93
4	0.94	0.9	0.92	0.95	0.91	0.93
5	0.95	0.9	0.92	0.95	0.91	0.93
6	0.95	0.9	0.92	0.95	0.92	0.93
7	0.95	0.89	0.92	0.95	0.91	0.93
8	0.95	0.9	0.92	0.95	0.91	0.93
9	0.95	0.9	0.92	0.95	0.91	0.93
10	0.95	0.9	0.92	0.95	0.91	0.93

구문분석을 위한 여러 자질을 추가할 경우 문장구조를 파악하는 것의 성능이 향상됨을 확인 할 수 있다. 그러므로 이후 실험에서는 Frame 자질을 설정할 때 기본 실험 모델이 아니라 6개 자질 모델을 사용하여 설정한다.

6.2. 자질 학습 성능

자질 학습 성능은 <표 IV.4>와 같다. 이 성능은 동사 묶음 처리를 하고 기능어 형태소와 공기정보(클래스)를 자질로 사용한 실험이다.

문장 구조 자질인 Frame을 추가하면서 기대했던 문장 성능이 향상되었다. 즉, 지역적 학습으로 문장 구조에 오류가 많았던 트랜지션의 단점을 보완하였다.

<표 IV.4> Frame 자질 학습 실험 성능

	문장 수	어절성능	문장성능
1	6,154	86.65	42.75
2	6,154	86.58	42.33
3	6,154	86.73	42.59
4	6,153	86.61	41.52
5	6,153	86.77	41.36
6	6,153	86.95	42.84
7	6,153	86.62	42.68
8	6,153	86.77	41.74
9	6,153	87.1	43.44
10	6,153	86.51	42.68
평균		86.73	42.39

6.3. 분석 가이드 성능

앞의 5.2에서 정의한 특징을 만족할 때에만 구문구조가 생길도록 구현하였다. 기본 모델은 6개 자질 모델을 사용하였다.

<표 IV.5> Frame 분석 가이드 실험 성능

	문장 수	어절성능	문장성능
1	6,154	81.62	34.27
2	6,154	81.7	33.23
3	6,154	81.9	34.74
4	6,153	81.83	33.51
5	6,153	81.58	33.11
6	6,153	81.94	34.13
7	6,153	81.19	34.28
8	6,153	81.58	33.14
9	6,153	81.31	33.33
10	6,153	81.09	33.17
평균		81.57	33.69

분석 가이드 성능은 <표 IV.5>와 같다. 자질 학습 성능에 비해서 많이 떨어진다. 이것은 앞에서 정의한 Frame 특징을 만족하지 않는 어절이 많다는 것을 의미한다.

Frame 자질 학습 모델과 같이 다단계 구 단위화의 첫 단계에서 Frame을 설정하고 분석 가이드에서 설정한 Frame 특징으로 분석 하는 것인데 Frame에 오류가 있다면 정의한 특징에 맞지 않는 구조가 생성된다.

정의한 Frame 특징에 문법적인 오류가 있는 것은 아니지만 설정한 Frame의 오류에 따른 전체적인 분석 구조가 변경되기 때문에 성능이 다른 실험에 비해 현저히 떨어지는 것을 확인하였다.

7. Frame 모델 성능 향상

<표 IV. 4>의 실험에서 Frame을 정답으로 설정하고 구문분석을 수행할 경우 <표 IV. 6>의 성능과 같다. 즉 Frame이 첫 번째 단계에서 잘못 설정 될 경우 문장의 구조가 잘못 지정되는 것으로 오히려 오류를 생성할 수 있는 것이다.

<표 IV. 6> 정답 Frame의 구문분석 성능

	문장 수	어절성능	문장성능
1	6,154	88.55	46.56
2	6,154	88.34	45.6
3	6,154	88.52	46.39
4	6,153	88.44	45.31
5	6,153	88.54	44.68
6	6,153	88.82	46.6
7	6,153	88.29	46.19
8	6,153	88.31	44.92
9	6,153	88.67	46.69
10	6,153	88.36	46.17
평균		88.48	45.91

지금까지 설명해온 Frame 설정 모델은 트랜지션 기반의 자질 생성 및 학습으로 만들어진 것이다. Frame은 문장의 주요 어절을 찾는 것인데 기존의 모델보다 좀더 Frame에 맞는 모델 생성이 필요하다.

- 확장 문장 구조 자질

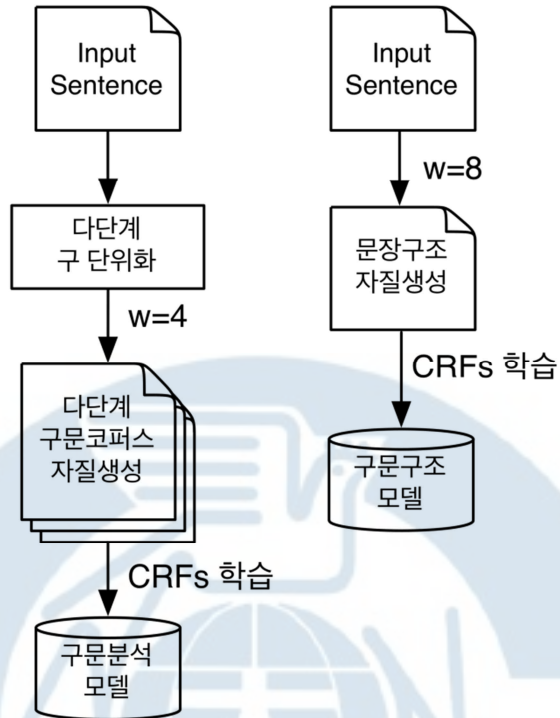
확장 문장 구조 자질은 기존의 CRFs 학습 모델로서 좀더 문장 구조를 파악하기 위해 문맥정보를 늘려 학습한다.

<그림 IV-13>은 구문분석 모델과 확장 문장구조 모델의 학습 차이점을 표현하고 있다. 기존의 구문분석 모델은 입력된 문장을 다단계 구 단위화로 단계를 모두 생성한다. 그리고 자질을 부착하여 CRFs를 학습하게 된다. 그리고 학습 시 문맥정보 크기를 4로하여 자질을 생성한다.

확장 문장구조 모델에서는 다단계 구 단위화를 거치지 않고 입력된 문서에서 자질을 생성하여 학습한다. 다른 점은 문맥정보 크기를 8로 하는 것이다. 구문분석 모델에 비해 좀더 넓은 문맥정보를 학습함으로써 문장 구조를 좀더 높은 성능으로 설정하도록 하였다.

구문분석에서 문맥정보를 확대하여 학습하면 자질의 수가 많아지기 때문에 자질 부족현상 때문에 표지 오류가 많아지므로 늘리는 것이 오히려 성능이 떨어지게 된다.

대신 Frame는 문장구조 자질이므로 넓은 문맥자질 조합이 필요하게 된다.



<그림 IV-13> 구문분석과 확장 문장 구조 학습 방법

<표 IV. 7>은 문맥정보를 확장한 문장구조 자질의 성능이다. 학습한 모델로서 Frame을 설정할 경우 평가 문서로 성능을 측정한다. <표 IV. 3>에 비해 F1 성능이 향상 되었다.

문장 구조 모델을 생성하여 <표 IV. 4>의 실험에 적용한 결과는 <표 IV. 8>과 같다. Frame 성능이 오르면 전체 구문성능이 향상 되는 것을 확인 할 수 있다. Precision 성능이 높은 모델을 정의하면 더 향상된 구문분석 성능을 기대할 수 있다.

<표 IV.7> 확장 문장 구조 Frame 설정 성능

	Precision	Recall	F1
1	0.95	0.94	0.94
2	0.95	0.94	0.94
3	0.95	0.94	0.94
4	0.95	0.94	0.94
5	0.95	0.94	0.94
6	0.95	0.95	0.95
7	0.95	0.94	0.94
8	0.95	0.94	0.94
9	0.95	0.94	0.94
10	0.95	0.94	0.94

<표 IV.8> 확장 문장 구조 모델 적용

	문장 수	어절성능	문장성능
1	6,154	86.86	43.18
2	6,154	86.76	42.44
3	6,154	87.04	43.0
4	6,153	87.05	42.35
5	6,153	87.07	41.96
6	6,153	87.34	43.38
7	6,153	86.84	43.17
8	6,153	87.08	42.22
9	6,153	87.32	43.8
10	6,153	86.92	43.3
평균		87.03	42.88

즉, 문장 구조 자질의 성능이 높을수록 문장 전체의 구조가 정확해지며 이를 고려한 구문분석의 성능에 반영된다는 것을 확인 할 수 있다.

8. 성능 비교

기존에 연구되어온 구문분석의 성능을 비교하였다. <표 IV.9>는 기존연구의 실험 결과이다.

한국어는 제안한 본 시스템을 제외한 나머지는 세종코퍼스가 아닌 KIB 코퍼스로 실험하였다. KIB코퍼스는 의존구조 구성 단위가 세종코퍼스와 다르다.

세종코퍼스는 어절단위로 의존구조를 생성하는 반면 KIB는 형태소를 기준으로 의존구조를 표현하고 있다.

코퍼스가 다르기 때문에 절대적 비교는 할 수 없지만 정확도로서 비교하고자 표를 기술 하였다. 문장 정확도가 없을 경우 표시하지 않았다.

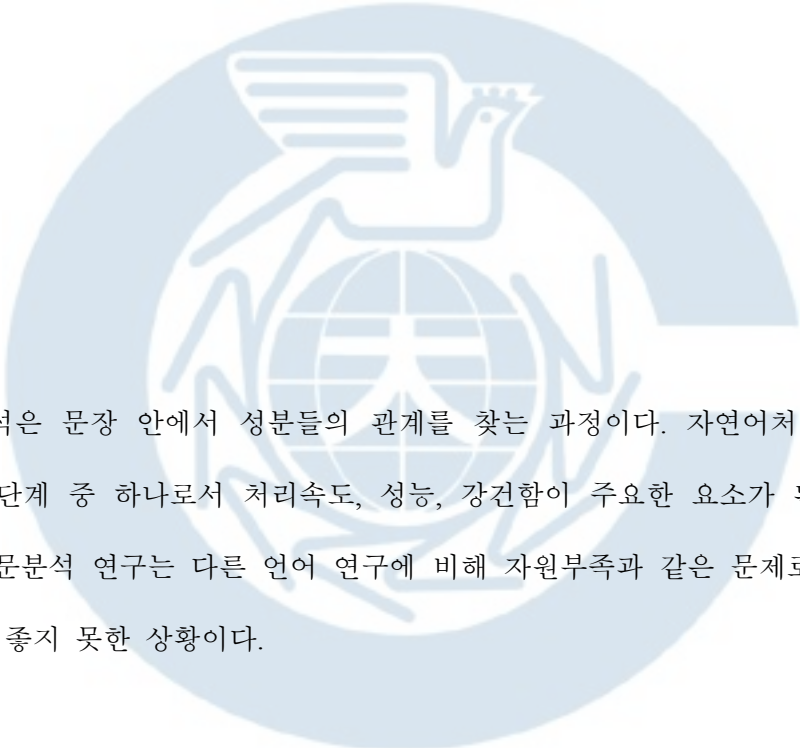
통합모델 MST_{Malt} 와 $Malt_{MST}$ 는 아래첨자로 쓴 방법의 결과를 자질로 추가한 것을 의미한다. MST는 그래프 기반 구문분석이고, Malt는 트랜지션 기반의 구문분석이다. 즉 MST_{Malt} 는 트랜지션 기반 구문분석의 결과를 자질로 사용한 시스템으로 기반 시스템은 그래프 기반 시스템을 뜻한다. 그리고 이 두 통합 모델은 LAS로 평가한 것이고 다른 시스템은 UAS 평가를 하였다.

<표 IV.9> 성능 비교

구분	의존관계 정확도(%)	문장 정확도(%)	코퍼스	비고
제안방법	87.03	42.88	세종	한국어
Lee[18]	88.42	35.13	KIB	한국어
Lex.(Chung)[17]	86.74	34.04	KIB	한국어
Unlex.(Chung) [17]	85.62	32.25	KIB	한국어
Kudo[11]	89.09	46.17	Kyoto Univ.	일본어
Kudo[32]	89.29	47.53	Kyoto Univ.	일본어
Yamada[24]	90.30	38.40	Penn Treebank	영어
Unlex.(Klein & Manning 2003)[3]	85.7	28.8	Penn Treebank	영어
Charniak 2000[4]	89.5		Penn Treebank	영어
Petrov & Klein 2007[5]	90.1		WSJ, Brown	영어
Charniak 2005[6]	91.4		Penn Treebank	영어
통합모델 MST _{Malt} [26]	82.53		CoNLL-X	다국어
통합모델 Malt _{MST} [26]	82.01		CoNLL-X	다국어

제 VI 장

결론 및 향후 연구



구문분석은 문장 안에서 성분들의 관계를 찾는 과정이다. 자연어처리 분야에서 필수적인 단계 중 하나로서 처리속도, 성능, 강건함이 주요한 요소가 된다. 하지만 한국어 구문분석 연구는 다른 언어 연구에 비해 자원부족과 같은 문제로 연구 깊이와 결과가 좋지 못한 상황이다.

본 논문에서는 처리속도, 성능, 강건함을 가지면서 기존의 시스템에 비해 문장 성능을 높이기 위한 문장 구조 자질을 제안한다. 기존에는 다단계 구단위화(Cascaded Chunking) 방법을 사용하여 강건함을 획득하였고 어휘를 사용하지 않은 자질 학습으로 모델의 사이즈가 작으며, 고속으로 처리할 수 있었다.

구문분석은 자연어처리의 기본 시스템이 되는 결과이기 때문에 상위 응용프로그램에서 적절한 데이터를 추출하려면 성능이 중요하다. 구문분석의 결과를 이용하게 되면 좀더 적은 데이터를 사용하여 사용자에게 맞는 데이터를 가공할 수 있기 때문이다.

구문분석의 성능은 어절(또는 형태소)성능과 문장 성능으로 평가한다. 문장의 성능을 높이기 위해서는 문장을 이루는 모든 어절의 아크가 올바르게 연결되어야 한다.

문장의 길이가 길어질수록 문장의 성능이 떨어지는데 이것은 문장의 큰 구조를 파악하기 힘들기 때문이다. 특히 트랜지션 기반의 구문분석 방법은 지역적 학습 방법과 오류전파 문제가 있으므로 그래프 기반 구문분석에 비해 문장구조 성능이 낮음을 기술하였다.

본 논문에서는 문장 성능을 향상시키기 위해 문장구조 자질을 적용하였다.

문장구조 자질은 문장에서 주요 요소가 되는 어절을 정의하는 것으로 자질로 정의하여 분석을 하는 방법이 성능향상에 도움을 준다는 것을 확인하였다. 즉, 지역적 학습 모델을 이용하여 분석을 하지만 문장구조 자질을 고려하여 전체 문장 구조 정보를 이용하는 것과 같다.

이외에도 기존 비어휘 다단계 구 단위화 시스템의 오류분석을 통하여 기존 문제점을 파악하고 구 패턴 묶음, 기능어 형태소 자질, 공기정보 자질을 추가하여 성능향상을 이룰 수 있었다.

또한 문장 구조 자질을 시스템 내부에서 정의하여 분석 시 성능 향상이 되는 것 뿐만 아니라 정답 문장 구조자질을 이용하면 성능이 큰 폭으로 증가하는 것으로 보아 잠재적인 성능 향상을 확인 할 수 있다.

그것을 바탕으로 문장 구조 설정 성능 향상을 위해 넓은 문맥자질로서 학습한 모델을 실험하였다. 이로서 문장 구조 자질이 문장 성능 향상에 큰 영향을 주는 것을 확인 할 수 있다.

향후 연구로는 문장 구조 자질과 분석 가이드 방법을 통합한 모델로서 구문분석 성능 향상 연구와 확장 문장 구조 자질을 위한 효과적인 학습 방법을 추가적으로 연구할 예정이다.

또한 KIB(Korean Language Information Base) 코퍼스를 사용하여 평가하고 다른 시스템과 직접 평가를 해 볼 예정이다. KIB는 의존구조 구성단위가 ‘형태소’이므로 이를 ‘어절’단위로 변경하여야 한다. 그리고 세종코퍼스와 달리 구문태그의 기능태그가 없기 때문에 KIB 코퍼스에 맞는 자질을 추가적으로 연구할 예정이다.

참고문헌

1. Marie-Catherine de Marneffe, Bil MacCartney and Christopher Manning, “*Generating typed dependency parses from phrase structure parses*”, LREC, pp. 449-454, 2006
2. Eugene Charniak, “*Statistical parsing with a context-free grammar and word statistics*”, AAAI, pp. 598-603, 1997
3. Dan Klein and Christopher Manning, “*Accurate Unlexicalized Parsing*”, ACL, pp. 423-430, 2003
4. Eugene Charniak, “*A Maximum-Entropy-Inspired Parse*”, NAACL, pp. 132-139, 2000
5. Slav Petrov and Dan Klein, “*Improved Inference for Unlexicalized Parsing*”, HLT-NAACL, pp. 404-411, 2007
6. Eugene Charniak and Mark Johnson, “*Coarse-to-fine n-best parsing and MaxEnt discriminative reranking*”, ACL, pp. 173-180, 2005
7. Masakazu Fujio and Yuji Matsumoto, “*Japanese Dependency Structure Analysis based on Lexicalized Statistics*”, EMNLP, pp. 87-96, 1998
8. Msahiko Haruno, Satoshi Shirai and Yoshifumi Ooyama, “*Using Decision Trees to Construct a Practical Parser*”, Machine Learning, pp. 505-511, 1998
9. Kiyotaka Uchimoto, Satoshi Sekine and Hitoshi Isahara, “*Japanese Dependency*

- Structure Analysis Based on Maximum Entropy Models*", EACL, pp. 196-203, 1999
10. Kiyotaka Uchimoto, Masaki Murata, Satoshi Sekine and Hitoshi Isahara, "*Dependency model using posterior context*", IWPT, 2000
 11. Taku Kudo and Yuji Matsumoto, "*Japanese Dependency Structure Analysis based on Support Vector Machines*", EMNLP/VLC, pp. 18-25. 2000
 12. Jang Chul Geum and Gil Changs Kim, "*Implementation of HPSG Parsing Mechanism for syntactic parsing for Korean*", KIISE, pp. 139-142, 1998
 13. Hui Sung Jung, Jae Hoon Kim, Jin Sun Lee, Soo Yeon Chun and Moon Jun Park, "*Design of Korean-English machine translation system (KoEng)*", Workshop of Machine Translation, pp. 87-96, 1989
 14. Jae Hyung Yang, "*A study on the Korean language analysis system based on HPSG*", Master' s thesis, Dept. of Computer Engineering Seoul National University, 1990
 15. D. H. Yoon and Y. T. Kim, "*Analysis techniques for Korean sentence based on Lexical Functional Grammar*", IWPT, pp. 369-78, 1989
 16. Jeongwon Cha, Geunbae Lee and Jong-Hyeok Lee, "*Morpho-syntactic categorial modeling of Korean*", ACH, pp. 431-453, 2002
 17. Hoojung Chung, "*Statistical Korean Dependency Parsing Model based on the surface Contextual Information*", Ph.D. dissertation, 2004
 18. Yong-Hun Lee and Jong-Hyeok Lee, "*Korean Parsing using Online Learning*", KCC, pp. 299-304, 2010
 19. Joakim Nivre, "*An efficient Algorithm for projective Dependency Parsing*", IWPT, pp.

149-160, 2003

20. Ryan McDonald, Koby Crammer and Fernando Pereira, “*Online Large-margin training of Dependency Parsers*”, ACL, pp. 91-98, 2005
21. Jason Eisner, “*Three new probabilistic models for dependency parsing: An exploration*”, CoNLL, pp. 340-345, 1996
22. Ryan McDonald and Fernando Pereira, “*Online Learning of Approximate Dependency Parsing Algorithms*”, EACL, pp. 81-88, 2006
23. Joakim Nivre, “*Non-Projective Dependency Parsing in Expected Linear Time*”, ACL/IJCNLP, pp. 351-359, 2009
24. Hiroyasu Yamada and Yuji Matsumoto, “*Statistical dependency analysis with support vector machines*”, IWPT, pp. 195-206, 2003
25. Joakim Nivre, Johan Hall, Jens Nilsson, Gulsen Eryigit and Svetoslav Marinov, “*Labeled pseudo-projective dependency parsing with support vector machine*”, CoNLL, pp. 221-225, 2006
26. Joakim Nivre and Ryan McDonald, “*Integrating Graph-based and Transition-based Dependency Parsers*”, ACL/HLT, pp. 950-958, 2008
27. 세종계획 21, <http://www.sejong.or.kr/>
28. Jin Young Oh, Yo-Sub Han, Jungyeul Park and Jeong-Won Cha, “*Predicting Phrase-Level Tags Using Entropy Inspired Discriminative Models*”, ICISA, 2011
29. Jin Young Oh and Jeong-Won Cha, “*High Speed Korean Dependency Analysis Using Cascaded Chunking*”, The Korea Society for Simulation, Vol. 19, 2010

30. John Lafferty, Andrew McCallum and Fernando Pereira, “*Conditional random fields: Probabilistic models for segmenting and labeling sequence data*” , ICML, pp. 282-289, 2001
31. Stephen Della Pietra, Vincent Della Pietra and John Lafferty, “*Inducing features of random fields*” , PAMI, pp. 380-393, 1997
32. Adam Berger, Vincent Della Pietra and Stephen Della Pietra, “*A maximum entropy approach to natural language processing*” , Computational Linguistics, pp. 39-71, 1996
33. Steven Abney, “*Parsing By Chunking*”, In Principle-Based Parsing, Kluwer Academic Publishers, 1991
34. Taku Kudo and Yuji Matsumoto, “*Japanese Dependency Analysis using cascaded Chunking*”, COLING, pp. 63-39, 2002
35. Huiwei Zhou, Tong Yu and Degen Huang, “*Japanese Dependency Analysis Based on SVMs and CRFs*”, International journal of mathematics and computers in simulation, pp. 233-237, 2007

ABSTRACT

Efficient Parsing for Head final Language

by Jin-Young Oh

*Dept. of Computer Engineering
Graduate School, Changwon National University
Changwon, Korea*

A syntactic parsing that captures a sentence structure and roles of words in a sentence is a key step of natural language understanding. However, we cannot use the Korean syntactic analyzer because of low performance and low robustness.

We propose new robust, high speed and high performance Korean syntactic analyzer using cascaded chunking method. We treat a parsing problem as a labeling problem. We label syntactic information to each Eojeol at each step using CRFs. CRFs use part-of-speech tag and Eojeol syntactic tag features.

The cascaded chunking is transition based dependency parsing. The main advantage of the cascaded chunking is high speed. However a major disadvantage is error propagation due to the greedy inference strategy and dependencies get longer.

The Graph based dependency parsing is globally learning and exhaustive search algorithms over all arcs.

To integrate two models, we propose new features about sentence structure. We call it as a Frame, which defines key Eojeols in sentence. This means that Frame Eojeols play key roles for sentence structure.

We detect Frame Eojeols at first step, and then we use their information at following steps during the analysis.

In this paper, we show how these results can be exploited to improve sentence level accuracy by using Frame features. And we also use morphemes of functional words, co-occurrence features and phrase-pattern features.

Our experimental results show that frame feature improve sentence accuracy and additional features also improve Eojeol level performance. We are sure that frame features and additional features make performance improvement for the Korean parsing problem.

부록 A. 품사 집합

TAG	POS	TAG	POS
NNG	일반명사	IC	감탄사
NNB	의존명사	VCP	긍정지정사
NNP	고유명사	VCN	부정지정사
NP	대명사	VV	동사
NR	수사	VA	형용사
JKS	주격조사	VX	보조용언
JKC	보격조사	EF	종결어미
JKO	목적격조사	EC	연결어미
JKG	관형격조사	ETN	명사형 전성어미
JKB	부사격조사	ETM	관형형전성어미
JKV	호격조사	EP	선어말어미
JKQ	인용격조사	SF	마침표, 물음표, 느낌표
JC	접속조사	SP	쉼표, 가운뎃점, 콜론, 빗금
JX	보조사	SS	따옴표, 괄호표, 줄표
XPN	명사접두사	SE	줄임표
XSN	명사파생접미사	SO	붙임표(물결, 숨김, 빠짐)
XSB	부사파생접미사	SL	외국어
XSV	동사파생접미사	SH	한자
XSA	형용사파생접미사	SN	숫자
XR	어근	NF	명사추정범주
MM	관형사	NV	용언추정범주
MAG	일반부사	SW	기타기호
MAJ	접속부사	NA	분석불능범주

부록 B. 구문태그 집합

- 구문표지

	범주	사례
S	문장	
Q	인용절	인용부호(“”) 안에 들어 있는 두 개 이상의 문장
NP	체언구	체언(명사, 대명사, 수사)
VP	용언구	용언(동사, 형용사, 보조용언)
VNP	긍정 지정사구	긍정 지정사 ‘이다’
AP	부사구	부사
DP	관형사구	관형사
IP	감탄사구	감탄사

- 기능표지

	범주	사례
SBJ	주어	주격 체언구, 명사 전성 용언구, 명사절 (NP_SBJ, VP_SBJ, S_SBJ)
OBJ	목적어	목적격 체언구, 명사 전성 용언구, 명사절 (NP_OBJ, VP_OBJ, S_OBJ)
CMP	보어	보격 체언구, 명사 전성 용언구, 인용절 (NP_CMP, VP_CMP, S_CMP)
MOD	체언 수식어	관형격 체언구, 관형형 용언구, 관형절 (NP_MOD, VP_MOD, S_MOD)
AJT	용언 수식어	부사격 체언구, 문말어미+부사격조사 (NP_AJT, VP_AJT, S_AJT)
CNJ	접속어	접속격 체언(NP_CNJ)
INT	독립어	체언(NP_INT)

● 기타표지

	범주	사례
RPN	삽입어구	삽입된 성분의 기능표지 위치에 표시(예: NP_PRN)
X	의사구 (pseudo phrase)	인용부호와 괄호를 제외한 나머지의 부호나, 조사, 어미가 단독으로 어절을 이룰 때 그 구문표지 위치 표시(예: X_CMP)
L, R	부호	인용부호나 괄호의 구문표지 위치에 표시. 왼쪽 부호에는 L 을, 오른쪽 부호에는 R 을 표시
Q, U, W, Y, Z	인용절	인용부호(“”)에 이끌려 나온 두 개 이상의 인용절을 대신하여 표기되는 부호



부록 C. 구문분석 예제

본 논문에서 제안하는 시스템으로 구현한 시스템의 웹 서비스 예제 화면이다.

아래의 그림은 ‘나는 밥을 빨리 먹었다.’라는 입력 문장에 대해서 ‘결과 1’은 그래프로 표현한 것이고 ‘결과 2’는 지배소 번호와 구문태그를 출력한다.


Espresso-P: Dependency Parser for Korean

[Help](#)[Parser](#)

☒결과1☐결과2

1문장, 구문분석을 수행했습니다.

1 나는 밥을 빨리 먹었다.



나는 밥을 빨리 먹었다.

1	나는	NP_SBJ	4
2	밥을	NP_OBJ	4
3	빨리	AP	4
4	먹었다.	VP	4

감사의 글

부족함이 많은 제가 논문을 작성할 수 있도록 도움을 준 모든 분들에게 감사합니다.

오랜 연구실 생활 동안 항상 큰 버팀목이 되어주신 차정원 교수님께 우선 감사드립니다. 그리고 묵묵히 뒤에서 응원을 많이 해주신 가족에게 감사합니다.

튼튼한 지원자가 되어주던 연구실을 벗어나려고 하니 많은 시간들이 떠오릅니다.

밤새도록 세미나를 하고, 프로젝트 하며 힘들어 하기도 하고, 맛있는 맛집을 먹으러 다니며 열심히 연구했던 지난날을 떠올리며 앞으로도 저 또한 게을리 보내지 않겠습니다.

감사합니다.



이 력 서

성 명: 오 진 영

생년월일: 1985년 11월 19일

출 생 지: 경상남도 창원군

주 소: 경상남도 창원군 남지읍 학계리 948-3번지

학 력

2004-2007: 창원대학교 공과대학 컴퓨터공학과(B.S.)

2008-2009: 창원대학교 대학원 컴퓨터공학과(M.S.)

2010-2013: 창원대학교 대학원 컴퓨터공학과(Ph.D.)

발표논문

1. 오진영, 차정원, “문장 구조 정보를 이용한 새로운 한국어 구문분석”, 한국정보과학회, 2013
2. Jin Young Oh, Yo-Sub Han, Jungyeul Park, Jeong-Won Cha, “Predicting Phrase-Level Tags Using Entropy Inspired Discriminative Models”, ICISA, 2011
3. 오진영, 차정원, “다단계 구단위화를 이용한 고속 한국어 의존구조분석”, 한국시물레이션학회 논문지, 19권 1호, 2010
4. 성병기, 오진영, 차정원, “LiveTwitter: 트위터 기반 핫이슈 검색시스템”, 제22회 한글 및 한국어 정보처리학회, pp.179~182, 2010
5. 오진영, 차정원, “고성능 비어휘정보 한국어 구문분석”, 한국 컴퓨터 종합학술대회 논문집, pp.295~298, 2010
6. 오진영, 차정원, “엔트로피 지도 CRF를 이용한 한국어 어절 구문태그 예측”, 정보과학회 논문지:컴퓨팅의 실제 및 레터 제15권 제5호, pp.395-399, 2009

7. 오진영, 차정원, “오류 분석을 통한 파서의 성능향상”, 제21회 한글및 한국어 정보처리 학회, pp.213-218, 2009
8. 안유미, 오진영, 차정원, “학습기능을 가진 효율적인 품사 부착 도구 설계 및 구현”, 제21회 한글 및 한국어 정보처리 학회, pp.191-196, 2009
9. 오진영, 차정원, “엔트로피 지도 CRF를 이용한 어절 구문태그 예측”, 추계정보과학회 논문집 제35권 제2호(A), pp.65-66, 2008
10. 오진영, 차정원, “CRF를 이용한 강건한 한국어 의존구조 분석”, 제20회 한글 및 한국어 정보처리 학회, pp.23-28, 2008

