국어사전의 명사 뜻풀이말 Parser

허 정[0] 김준수 이수광 옥철영
울 산 대 학 교 전 산 학 과
{orionhj,jskim}@cic.ulsan.ac.kr skl@future.co.kr okcy@uou.ulsan.ac.kr

A Parser for Noun ' s Definition in Korean Dictionary

Jeong Hur [0] Jun-Soo Kim Soo-Kwang Lee Chul-Young Ok
Dept. of Computer Science, Ulsan University

요 약

국어 사전은 자연 언어 처리에서 필요로 하는 많은 정보를 구조적으로 포함하고 있으므로, 사전으로부터 다양한 언어 지식을 자동으로 획득할 수 있는 방법이 필요하다. 본 연구는 이러한 자동 지식 획득을 위한 기본적인 도구로서 국어 사전의 뜻풀이말 파서를 구현하는 것을 목적으로 한다.

이를 위해서 우선 국어 사전의 뜻풀이말을 대상으로 일정한 수준의 구문 부착 말뭉치를 구축하고, 이 말뭉치로부터 통계적인 방법에 기반하여 문법 규칙과 확률을 자동으로 추출한다. 본 연구는 이를 응용한 확률적 차트 파서를 구현하는 것이다.

그 결과 고려대 태거보다 11.61%의 정확률 향상을 보였는데, 이로써 구문 구조 정보가 품사 태깅에도 유용함을 알 수 있었다.

1. 서론

구문 분석은 문장이 통사적으로 어떤 구조를 가지고 있는지를 밝히고 그것에 따라 문구조를 생성해 내는 작업으로서, 이를 위한 여러가지 문법 이론(Grammar Formalism)들이 제시되었다. 그리고 이것을 컴퓨터에서 효율적으로 처리하기 위해 여러가지 구문 분석 기법(Parsing Method)들이 등장했다.

한국어는 자유로운 어순과 불완전한 문장의 허용이라는 구문적 특성 때문에 한국어의 구문 분석에는 단일화에 기반한(Unification-based) 구문 분석과 의존 문법(Dependency Grammar)에 기반한 구문 분석이 주목을 받아왔다. 이와 같은 구문 분석 방법론은 어휘의 자질 정보나 범주와 같은 언어 정보가 구문 분석에서 대단히 중요한 역할을 한다. 그러나 실세계의 모든 언어 현상을 고려하고, 언어의 변화에 능동적으로 대처할 수 있는 견고한 언어 지식을 구축하는 것은 어렵다.

반면에, 최근의 통계적(Statistical) 자연 언어 처리는 대용량의 말뭉치(Corpus)로부터 기계 학습(Machine Learning) 방법에 의해 획득된 통계적 언어 지식을 기반으로 언어 모델(Language Model)을 구축하여, 자연 언어 처리 각 단계에서 발생하는 여러 중의성을 해결하는 연구에서 성공적인 결과[1, 2]를 보이고 있다.

본 연구는 통계적 방법론에 입각하여 국어 사전의 명사 뜻풀이말에 내재된 확률적 문법 규칙을 추출하고, 이를 응용한 파서를 구현하는 것이다

2. 어절 태그 사전과 트리 사전

UPAS[1]에서는 어절 태그 사전(Word-Tag Dictionary)과 트리 사전(Tree Dictionary)이 품사 태깅과 파싱 과정에 각각 사용된다.(그림2-1) 어절 태그 사전은 태거를 이용하여 구축된 품사 부착 말뭉치에서, 트리 사전은 수동으로 구축된 구문 부착 말뭉치에서 자동으로 구축된다.

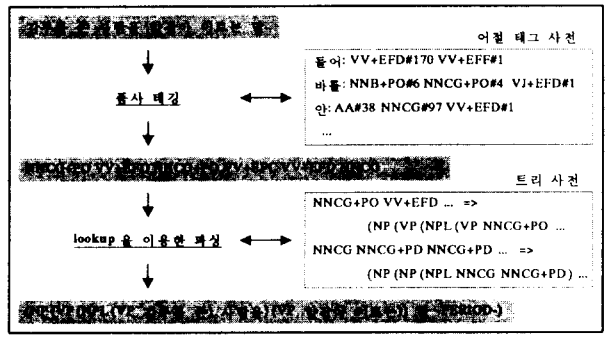


그림 2-1 어절 태그 사전과 트리 사전을 이용한 파싱

2.1 어절 태그 사전

어절 태그 사전(Word-Tag Dictionary)은 그림2-1에서와 같이 어절, 어절의 품사 태그열과 빈도수로 구성된다.

먼저, UMRD[2]에서 76,975 개의 명사 뜻풀이말을 추출한 다음, 고려대 태거를 이용하여 품사 부착 뜻풀이말을 구축하였다. 이로부터 어절 태그사전을 자동으로 구축하였다. 수집된 어절은 총 81,084 개이고 어절 태그열의 종류는 총 1,563 개 였으며 어절당 평균 분석 결과는 1.029 개 였다.

2.2 트리 사전

어절의 품사 태그열과 구별하기 위해 문장내 전체 어절의 품사 태그열을 '문장 태그열'이라고 부르기로 한다. 총 76,975 개 문장을 품사 태깅하였는데, 32,547 가지의 문장 태그열이 존재하였다. 이 중 다빈도 문장 태그열에 해당되는 10,025 개의 문장들에 대해 반자동으로 구문 부착을 하였고, 이 구문 부착 말뭉치로부터 411 개의 문법 규칙과 문법 확률을 추출하여 트리 사전(Tree Dictionary)을 구축하였다.

[1] UPAS(Ulsan university PArSer)

[2] UMRD(Ulsan university' s Machine Readable Dictionary)

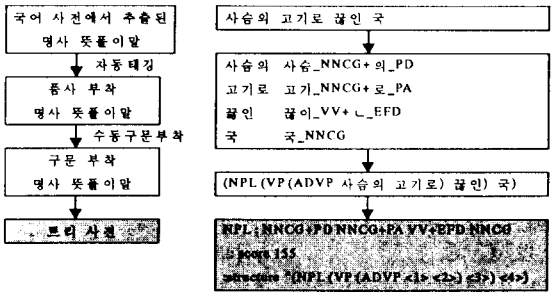


그림 2-2: 트리 사전의 구축 과정의 예

트리 사전은 문법 규칙, 스코어와 구문구조로 구성되며 그림2-2와 같은 과정을 거쳐 생성된다.

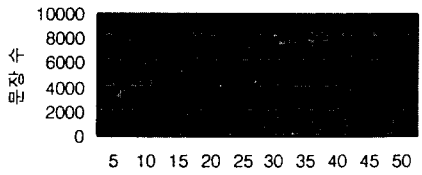


그림 2-3: 문법 규칙의 커버리지(Coverage)

그림2-3은 그림2-2의 과정을 통하여 구축된 문법 규칙에 대한 통계이다. 총 411 개의 문법 규칙 중 50 개(약 12.16%)의 규칙이 총 10,025 개의 문장 중 8,150 개(약 81.29%)의 문장에 적용된다. 이것은 아주 적은 수의 구문 구조가 학습 말뭉치의 많은 문장을 커버함을 알 수 있다.

많은 문장이 적은 수의 문법 규칙에 의해 커버되지만, 동일한 문장 태그열로부터 생성되는 두 개 이상의 구조가 존재할 수 있다. 즉 명사구 해석의 중의성이 발생하는데, 수집된 문법 규칙에서 많이 발생하는 명사구 중의성의 한 예를 보이면 다음과 같다.

유형 : NNCG+PO VV+EFC VV+EFD NNCG+PD NNCG
ㄱ. (NP(VP 수를 (VP 놓아 만든))(NPL 첫먹이의 버선))
ㄴ. (NP(NPL(VP 오라를 (VP 차고 나선)) 포졸의) 위풍)

UPAS는 위와 같이 중의성을 가지는 품사 시퀀스에 대한 구문 구조를 선택하기 위해 부분 트리에 대한 발생확률을 이용한다.

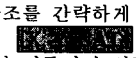
3. 파싱 알고리즘

UPAS는 상향식(Bottom-up)의 확률적 차트 파싱(Probabilistic Chart Parsing) 방법을 사용한다. 파서는 사전확률(Dictionary Probability) 과 문법 확률(Grammar Probability)을 이용하여 각 노드의 스코어를 계산하며, 스코어가 가장 높은 노드를 먼저 선택하여(Best-first 탐색) 부분 트리를 구성하고, 이 부분 트리들 중에서 스코어가 가장 높은 부분 트리만을 대상으로(Viterbi 탐색) 유일한 최적의 파스 트리를 찾아낸다.

3.1 상향식 차트 파싱

차트 파싱은 다이나믹 프로그래밍 기법을 사용하여 수행되어 온 분석 과정을 차트라는 구조물에 저장하여 동일한 작업의 반복적인 수행을 방지함으로써 보다 효율적으로 파싱을 수행할 수 있는 알고리즘이다.[4] 이 알고리즘은 새로운 분석을 수행하기 전에 차트를 살펴보고 이미 동일한 작업이 수행되었는지를 검색한다. 만일 이미 동일한 작업이 수행되었다면 이미 수행된 과정과 결과를 이용한다. 따라서 동일한 분석을 반복하게 하는 백트래킹 기법의 비효율성이 제거된다.

본 연구에서 사용되는 차트는 이차원 배열[X,T]로서 X 좌표는 노드의 문장 위치를 나타내고, Y 좌표는 노드가 표현하는 현재까지 완성된 부분 구조의 문장 범위(Span)을 나타낸다. 알고리즘에서 사용하는 노드의 구조를 간략하게 아래와 같이 표기하기로 한다.



위에서 X, Y 는 노드가 기록되어 있는 차트의 X 좌표와 Y 좌표이며, CAT 은 터미널 심볼 또는 문법 심볼을 나타낸다.

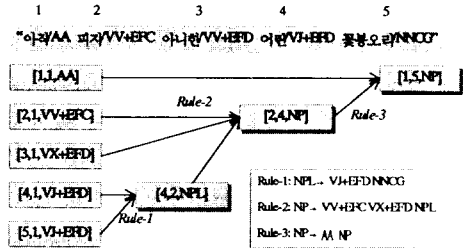


그림 3-1: 파스 트리에 대한 노드 기호의 표현

3.2 스코어 계산

각 어절의 태그열은 스코어 계산에 사용되는 빈도값을 가지고 있고 아래 식에 의해 사전 확률(Dictionary Probability, $P_{tag}(t|w)$)이 계산된다.

$$P_{tag}(t|w) = \frac{\text{Frequency of word w with tag t}}{\text{Frequency of word w}}$$

규칙 X → Y 의 확률($P_{rule}(X \rightarrow Y)$)은 아래의 식과 같이 트리 태깅 말뭉치에서 X 가 Y 를 유도하는 빈도와 넌터미널 X 의 빈도값에 의해 계산된다.

$$P_{rule}(X \rightarrow Y) = \frac{\text{Frequency with which X is expanded as Y}}{\text{Frequency of rules where LHS is X}}$$

파스 트리(T)의 스코어($S_{tree}(T)$)는 아래의 식과 같이 트리를 구성하기 위해 사용된 규칙들의 확률값과 각 어절들의 태그 확률값의 제곱으로 계산된다.

$$S_{tree}(T) = \prod_{R:\,rules\,in\,T} P_{rule}(R) \quad \prod_{t:\,tags\,in\,T} (P_{tag}(t|w))^2$$

어절의 태그 확률값에 제곱을 한 이유는 문법 규칙의 확률값보다 어절 태그열의 확률값에 가중치를 좀 더 두기 위한 것으로, 동등한 가중치를 할당한 경우보다 더 좋은 결과를 나타내었다.

3.3 문법 Factoring

문법 규칙은 공통 접두사로 Factoring 되며 유한 오토마톤(Finite State Automaton)으로 표현될 수 있다. 문법 스코어는 Best-first 탐색 알고리즘을 적용하기 위하여 유한 상태 오토마톤의 각 노드에 저장된다.

그림3-3은 그림3-2의 규칙들에대한 유한 상태 오토마톤을 나타낸다. 하나의 규칙 패스에 이르는 부분 스코어의 합이 그 규칙에 대한 스코어이다. 예를 들면, NP -> ABC 의 스코어는 "20 + 0 + 0 + 5 + 5 = 30"과 같이 된다.

NP → A B C : 30
NP → A B C D : 25
NP → A B F : 20
NPL → A B C : 40
NPL → A E : 25

그림 3-2: 문법 규칙의 예

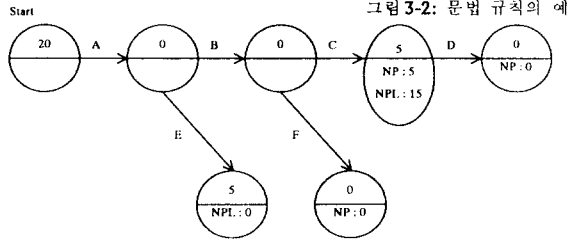


그림 3-3: 그림 3-2 의 문법 규칙에 대한 수정된 오토마톤

3.4 Best-first 탐색

파서는 힙(Heap)을 가지고 있는데, 이곳에 확장(expand)될 Active 노드들이 저장된다. 이 Active 노드들은 그들의 스코어에 따라 순서대로 힙에 저장된다. 따라서 힙의 Top은 베스트 스코어를 가진 Active 노드이며 이 노드가 다음에 확장될 노드이다. Top -NP또는 Top -NPL[3]이 전체문장에 대해서 생성되면 파싱은 더 높은 스코어를 가진 트리를 만들 가능성이 있는 Active 노드가 있는지를 검사한 후 종료된다. 이와 같은 탐색 기법은 일반적인 방식의 차트 파서보다 Active 노드의 수를 많이 감소시킨다.

3.4 Viterbi 탐색

코스트 최소법은 해의 노드와 링크에 적당한 코스트를 두고, 코스트 최소의 패스를 우선해로 하여 선택하는 방법이다.[4] 격자해 중에서 코스트 최소해를 구하는 데는, 부분적 최적해를 유지해 간다고 하는 다이나믹 프로그래밍의 일종으로 Viterbi 알고리즘이 있다.

그림3-3의 문법 오토마톤과 같이 노드와 노드에 적당한 스코어를 계산해 놓고, 스토어 최대 패스를 우선해로 하여 선택하는 방법을 사용한다.

4. 실험 및 평가

이 장에서는 확률적 문법 규칙의 유용성을 다섯 가지 실험을 통해 살펴본다. 실행 환경은 WindowNT 4.0, CPU 333MHz, RAM 250M이며 C 언어로 구현하였다.

(내부실험 1) 확률적 문법 규칙의 정확률 실험
(내부실험 2) 구문 구조 정보를 이용한 품사 태깅 실험
(외부실험 3) 확률적 문법 규칙의 정확률 실험
(외부실험 4) 왼쪽 우선 파싱 방법을 적용한 실험
(외부실험 5) 실험 3 에 왼쪽 우선 탐색을 적용한 실험

표 1. 내부실험 결과

	실험 1	실험 2
Node 수(InActive Edge)	1,944	3,520
Anode 수(Active Edge)	2,024	5,795
실행시간	12.09㎳	15.77㎳
고려대 품사 태거의 정확률	98.47%	84.77%
문법 규칙에 의한 태깅 정확률	99.52%	97.85%
문법 규칙의 정확률	90.19%	96.38%

실험1은 동일한 품사 태그열을 가지는 문장들을 분류한 후 고빈도인 상위 219 개의 분류군(총 문장수: 9,776 개)에서 트리 사전을 구축하고, 이 분류군에서 각각 하나씩의 문장을 추출하여 실험 대상 문장으로 선정한다. 그 결과는 위의 표1과 같은 데, 이것은 205 개[4]의 문장이 9,776 개 문장을 대표할 수 있음을 뜻하는 것이다.

실험2는 중의성 어절의 중요도[3]를 계산하여 총 36 개의 중의성 어절을 선정하였다. 이 어절을 포함하는 뜻풀이말 중 고려대 태거에서 오류를 일으키는 249 개의 문장을 실험 대상 문장으로 선정하여 실험 하였다. 그 결과 고려대 태거의 태깅 정확률보다 11.61% 의 향상을 보였는데, 이는 문맥 자유 구구조로 기술된 확률적 문법 규칙이 어휘의 문맥과 확률을 이용하는 문맥 의존 확률 모델에 표현된 제약사항들을 충분히 표현하고 있음을 보여주는 것이다. 또한 이는 기존의 품사 태깅 시스템이 더 높은 정확률을 얻기 위해서는 부분 또는 완전 구문 분석으로부터 구조 정보를 획득하여 품사 태깅 모형을 확장해야 함을 보여주는 것이다.

[3] 루트 노드에 해당하는 NP, NPL 을 나타낸다.

[4] 문장 길이 2 인 문장들은 제외하였다.

내부 실험의 결과, 태깅 정확률과 문법 규칙의 정확률이 비교적 높게 나온 것은 실험에 사용된 학습 문장의 수와 트리 사전에 등록된 규칙의 수가 많지 않기 때문이다. 그러므로 일반적인 태깅 시스템과 정확률을 비교하는 것은 다소 무리가 있는 듯하다. 그러나 실험 결과로부터 국어 사전의 명사 뜻풀이말과 같은 제한된 언어 사용 분야에서는 적은 양의 말뭉치로부터 습득한 확률적 문법 규칙(Probabilistic Grammar Rules) 이 전체 말뭉치를 대표할 수 있으며, 구문 구조 정보는 태깅시 매우 유용함을 확인할 수 있다.

표 2. 외부실험 결과

	실험 3	실험 4	실험 5
재현률	**51.74%**	51.74%	51.74%
정확률	75.52%	84.77%	**87.47%**
Node 수(InActive Edge)	5,750	5,676	5,516
Anode 수(Active Edge)	9,329	8,388	8,145
실행시간(㎳)	**33.58**	34.08	33.76

실험3에서는 사전 확률과 문법 확률을 사용하여 정확률을 실험하였다.

실험 4 는 문법 확률을 이용하지 않고 사전 확률과, 왼쪽 우선 파싱(Left-to-Right Parsing) 방법을 이용하여 수행하였다. 이 실험은 명사 뜻풀이말은 중심어가 문장의 맨끝에 오므로 중심어 후위의 원칙[5]에 따라 수식 성분을 먼저 탐색하여 부분 트리를 찾는 것이 유용할 것이라는 가정을 기반으로 한다.

실험5는 실험3과 같은 방식으로 하되, 단말 노드 또는 부분 트리를 선택함에 있어서 Best -first 탐색이 아니라 왼쪽 우선(Left-first) 탐색을 적용하였다. 전체 파싱 방법은 사전 확률과 문법 확률이 최대가 되는 최적해를 찾는 방식이다. 실험 결과로부터, 중심어 후위의 원칙에 따라 왼쪽 우선 탐색 방식(실험4)과 명사구 해석의 중의성을 해소하기 위해 문법 확률을 사용하는 방식(실험3)을 혼용하는 것이 가장 적합함을 알 수 있다.

5. 결론 및 향후 연구 과제

본 논문에서는 국어 사전의 명사 뜻풀이말의 구문 분석을 위하여 구문 부착 말뭉치를 구축하고 여기서 자동 추출한 확률적 문법 규칙을 이용하는 상향식의 확률적 차트 파서를 구현하였다.

실험 결과, 태깅 시스템이 더 높은 정확률을 얻기 위해서는 부분 또는 완전 구문 분석으로부터 구조 정보를 획득하여 품사 태깅 모형을 확장해야 함을 뒷받침한다.

앞으로의 과제는 첫째, UPAS 를 이용한 명사 계층망의 자동 구축에 관한 연구이다. UPAS 는 명사의 상의어, 하의어의 자동 추출 뿐만 아니라, 명사와 호응 관계에 있는 수식언의 자동 추출에도 유용하게 사용될 수 있다. 동시에 UPAS 의 출력 결과를 한국어에 맞는 의존 구조로 변환하는 연구가 병행되어야 한다. 둘째, UPAS 시스템이 범용적으로 사용되기 위해서는 문법 규칙에 대한 확장이 필요하다. 국어 사전의 명사 뜻풀이말 뿐만아니라 일반적인 문장을 분석하기 위해서는 문법 규칙을 확장해야 하며, 견고하고 일관성 있는 규칙의 기술 방법이 마련되어야 한다.

6. 참고 자료

[1] 김나리, "패턴정보를 이용한 한국어 구문분석", 서울대 컴공과 박사학위 논문, 1997
[2] 서정연,김창현, "통계적 방법을 이용한 구문 분석", 정보과학회지 제 14 권 제 7 호, 1996
[3] 김재한, "한국어 어휘 중의성 해소를 위한 태깅 시스템", 울산대학교 석사학위논문, 1994
[4] Makoto Nago 저, "자연언어처리", 홍릉과학출판사 1996

[5] 한국어의 특성 중 하나로 수식받는 말이 수식하는 말뒤에 옴을 말한다.