# 복합 커널을 사용한 한국어 종속절의 의존관계 분석

김상수[1] 박성배[1] 이상조[1] 박세영[2]
경북대학교 컴퓨터공학과
[1]{sskim, sbpark, sjlee}@sejong.knu.ac.kr
[2] seyoung@knu.ac.kr

# Analyzing Dependencies of Korean Subordinate Clauses using a Composite Kernel

Sang-Soo Kim○ Seong-Bae Park Sang-Jo Lee Se Young Park
Department of Computer Engineering
Kyungpook National University

**요 약**

한국어에서 절들의 의존관계를 밝히는 작업은 구문 분석 작업에서 가장 어려운 작업들 중에 하나로 인식되고 있다. 절의 의존관계를 파악하는 일은 표면적으로 나타나는 정보만을 가지고 처리할 수 없고, 의미 정보 같은 추가적인 정보가 필요할 것으로 판단하고 처리해왔다. 본 논문에서는 추가적인 정보를 사용하지 않고, 문장에서 얻을 수 있는 표면적인 정보만을 사용하여 절들 간의 의존관계를 파악하는 방법을 제안한다. 문장에서 얻을 수 있는 표면적인 정보는 문장의 구문 정보(tree structure information)와 어휘 및 거리 정보를 가지고 있는 정적인 정보(static information)로 나누어 볼 수 있다. 본 논문에서는 절들 간의 의존 관계 파악을 위하여 구문 정보 및 어휘정보 등을 하나 이상의 커널의 결합해서 사용하는 복합 커널(composite kernel)을 제안하고, 이 커널에 맞는 다양한 인스턴스 공간의 설정을 제안한다.

실험 데이터는 구문 트리로 표현된 STEP 2000코퍼스를 사용하였다. 실험은 최적화된 인스턴스 공간을 절들 간의 의존관계 파악 및 문장 수준에서 성능을 검정하였다. 관계 인스턴스 공간은 절들 간의 연결을 기준으로 Path-enclosed Tree와 Flattened Path-enclosed Tree로, 하부절(관형절)의 표현 유무로 Complete Tree, Contex-sensitive Tree, Simple Tree로 나누어 각각의 조합으로 실험하여 결정하였다. 그리고 결정된 인스턴스 공간에서 복합커널을 사용한 방법이 좋은 성능을 발휘함을 보였다.

## 1. 서 론

한국어에서 절들의 의존관계를 밝히는 작업은 구문 분석 작업에서 가장 어려운 작업들 중에 하나로 인식되고 있다. 특히 비교적 자유로운 문장의 어순과 동사의 다양한 활용으로 인하여 하나의 문단(paragraph)도 문법적으로 오류 없이 하나의 문장으로 표현할 수 있다. 그리고 절과 의존 관계가 성립되는 대상이 절, 명사구 등이 매우 다양하고 의존 관계 대상들 사이의 거리의 제약도 크게 받지 않는 등의 매우 다양한 형태로 나타난다. 이렇게 됨으로 표면적으로 나타나는 정보-동사의 패턴, 어미, 거리-의 형태만으로는 절들이 어떤 대상과 의존 관계에 있는지 파악하기 매우 힘들다. 따라서 한국어에서 절의 의존관계를 파악하는 일은 표면적으로 나타나는 정보만을 가지고 처리할 수 없고, 의미 정보 같은 추가적인정보가 필요할 것으로 판단하고 처리해왔다.

커널 함수(kernel method)는 각각의 개체의 고유한 표현(the original representation of objects)을 사용하면서, 두 개체들 사이의 유사성의 정도를 밝히는 능력이 좋다고 보고되고 있다. 이러한 능력으로 인하여 최근에는 커널 함수는 자연어처리 분야에서 많은 관심을 가지고 다양하게 적용되고 있다. 다양한 커널들 중에서 특히 파스트리 커널은 파스트리로 구성된 학습 데이터로부터 수직적 구조 정보(hierarchical structure information)를 추출하고, 비교해서 두 트리의 유사성의 정도를 밝히는 것으로 유명하다. 이 방법은 기존의 자질 기반의 방법에서처럼 파스 트리를 자질 벡터로 변환하면서 발생하는 손실을 최소화하면서 정확한 측정이 가능하다[1,2]. 이뿐만 아니라, 최근 연구에서는 한 문제에 여러가지 관점을 가지는 커널들을 다양한 방법으로 결합하여 정보의 손실을 줄이면서 많은 문제에 접근하고 있다.

본 논문에서는 절들 간의 의존관계는 추가적인 정보 없이 구문 트리의 구문 구조 정보와 각 어휘 같은 정적 자질 정보만으로 파악할 수 있다고 가정했다. 이 가정에 따라 문법적 구문 정보를 잘 다루는 파스트리 커널과 정적 자질을 다루는 지수 커널을 결합한 복합 커널을 제안하고, 의존 관계를 잘 표현하는 다양한 인스턴스 공간을 제시한다. 그리고 제안된 커널과 인스턴스 공간을 활용하여 하위절의 의존관계를 파악하였고, 파스트리 및 자질 기반 방법보다 높은 성능을 발휘하는 것을 보인다.

본 논문의 구성은 다음과 같다. 2장에서는 절들 간의 의존관계에 관련된 선행 연구들을 살펴보고, 3장에서는 본 논문에서 제안하는 복합커널 즉, 파스트리 커널과 커널의 조합에 관해서 간략히 설명한다. 4장에서는 복합커널을 사용하여, 의존관계를 표현하는 인스턴스 공간의 구성을 알아보고 5장에서는 본 연구에 사용된 말뭉치 및 실험 결과를 보인다. 마지막으로 6장에서는 결론과 향후 연구에 관해 논하는 것으로 구성되어 있다

## 2. 관련 연구

절들 사이의 의존관계를 인식하는 연구는 주로 절을 인식(clause identification)하는 연구와 문장의 의존 구조를 밝히는 연구에서 주로 있어왔다. 절을 인식하는 연구는 각각의 절의 시작과 끝 지점을 인식하는 문제로서 각각의 절의 시작과 끝을 인식을 통하여 절들의 의존 관계를 파악하려고 했다. 특히 2001년 CoNLL에서는 shard task로 종속절을 인식하는 다양한 연구들이 있었다. Carreas[3]는 Boosting Tree를 사용하였고, Molina[4]는 HMM을 사용하여 종속절을 인식하였다.

문장의 의존구조에 관한 연구에서는 의존 관계를 파악하는 대상이 절이 아니라 하나의 단어 또는 어절 단위로 삼았다. 따라서 절들 사이의 의존 관계를 파악하기 보다는 문장 속에 속한 동사구의 의존 관계를 파악 방향으로 연구가 진행되었다. Uchimoto[5]는 Maximum Entropy Model을 사용하여 다양한 자질(Feature)를 사용하여 의존 구조를 밝혔고, 특히 각각의 자질들이 의존구조에 어떤 영향을 주는지 각각 실험하였다. Kudo[6]는 의존 구조를 이진 분류 문제로 정의하고 Support Vector Machine을 사용하였다. 학습에 사용된 자질은 의존관계에 있는 어절의 단어 문법적 자질-와 POS Tag 등- 뿐만 아니라 의미역(Semantic Role) 같은 의미정보을 사용하였다. 이 뿐만 아니라 각 체언의 조사와 서술어의 어미의 활용(inflection)을 사용했다.

한국어에서는 절의 경계를 인식하는 연구와 구문 분석 과정에서 연구가 진행되어 왔다. 이현주[7]는 문장에서 n-gram을 추출하고 이를 SVM을 통하여 절들의 경계를 인식했다. 특히 중요 성분이 뒤쪽에 출현하는 특징을 이용하여 끝 지점을 먼저 인식하고 이를 바탕으로 시작점을 인식하였다. 서광진[8]은 구문 분석 과정에서는 의존문법을 기반으로 조사와 어미를 구분하여 각각의 어절을 지배소와 의존소로 나누고 이들간의 의존관계를 밝혔다. 정후중[9]은 구문 분석과정에서 의존소 및 지배소의 의존과계 분석을 위하여 코퍼스에서 통계 정보를 추출하여 각 어휘들간의 의존관계를 분석하였다. 김상수와 박성배[10]는 파스트리 커널을 이용하여 의존 관계에 있는 절들간의 대상을 분석하였고, 절들간의 어휘 정보 및 거리 같은 표면적 정보를 사용하여 절들 간의 의존관계를 분석하였다.

파스 트리 커널을 사용하여 절들 사이의 의존관계를 밝히는 연구는 아직 없었지만, 다양한 관계를 추출하는 부분에 적용되고 있다. Min Zhang[1]은 파스트리 커널과 개체(entity) 커널을 결합한 복합 커널로 문장 속에서 각각의 개체들 사이의 관계를 추출하였다. Zelenko[11]는 파스 트리에서 의존관계을 추출하기 위한 커널을 개발했다. 이 커널은 루트(root)에서 밑으로 단말 노드(leaf node)를 재귀적 방법으로 비교했다. Bunescu와 Mooney[12]는 문장에서 두 개체를 추출하고 이들 간의 가장 짧은 패스(shortest path)를 추출하여 의존 그래프(dependency graph)로 표현하였다. 그리고 이를 바탕으로 두 개체들 사이의 관계를 추출하는 A Shortest Path Dependency Kernel을 개발했다.

## 3. 절들 간의 의존 관계 파악을 위한 복합 커널

본 논문에서는 하위절의 의존관계 파악을 위해 파스트리 커널과 지수커널을 복합적으로 사용하는 복합 커널을 제안한다. 따라서 본 장에서는 먼저 파스트리 커널에 대하여 간략하게 설명하고, 본 논문에서 제안하는 복합 커널, 즉 파스트리 커널과 지수 커널과의 조합에 관해서 간단하게 살펴본다.

### 3.1 파스 트리 커널(parse tree kernel)

본 논문에서는 사용한 파스트리 커널은 convolution 커널의 한 종류로 Collins와 Duffy가 제안한 커널을 사용하였다[13].

파스트리 커널에서 벡터의 자질은 각 파스트리에 나타날 수 있는 모든 subtree로 이루어진다. 이때, 각 자질의 값은 subtree의 빈도로 할당된다. 그림 1은 간단한 parse tree에 대해 subtree들의 예를 보여 준다. 하지만 이러한 subtree를 명시적으로 구한다는 것은 불가능하다.

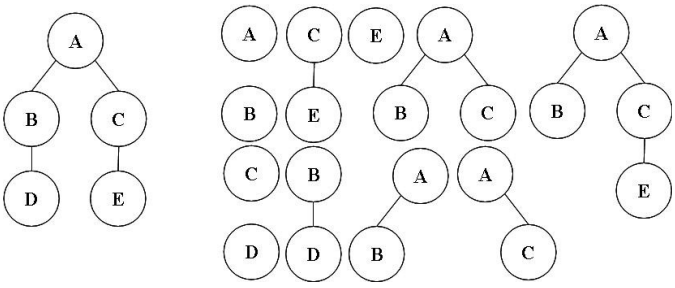



그림 1. Parse tree와 subtree의 예

트리 **T**의 부분 트리인 $subtree_1$, $subtree_2$, ... $subtree_n$이 주어졌다면, 파스트리 **T**는 아래의 벡터로 표현될 수 있다.

$$V_T = (\#subtree_i(T_1),\ \#subtree_2(T_2),\ ...,\ \#subtree_n(T_n))$$

여기서 $\#subtree_i$는 파스트리 T의 *i*번째 부분트리의 빈도수를 의미한다.

비교 대상으로 주어진 파스트리 $\mathbf{T_1}$, $\mathbf{T_2}$의 내적(inner product)은 다음과 같다.

$$< V_{T_1}, V_{T_2} >= \sum_i \#subtree_i(T_1) \cdot \#subtree_i(T_2)$$
$$= \sum_i ( \sum_{n_1 \in N_{T_1}} I_{subtree_i}(n_1)) \cdot ( \sum_{n_1 \in N_{T_2}} I_{subtree_i}(n_2))$$
$$= \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} C(n_1, n_2)$$

여기서 $\mathbf{N_{T1}}$, $\mathbf{N_{T2}}$는 파스트리 $\mathbf{T_1}$,과 $\mathbf{T_2}$에서 나타날 수 있는 하위 트리의 집합을 의미하고, $I_{subtreei}(n_1)$은 $subtree_i$가 루트에 노드 $n$이 나타나면 1을 돌려주고, 이외의 경우에는 0을 돌려준다. $C(n_1,n_2)$는 $n_1$, $n_2$ 노드에 나타날 수 있는 하위트리의 빈도수의 합이고, 이는 아래의 식과 3개의 규칙을 통하여 계산된다.

$$C(n_1, n_2) = \sum_i I_{f_i}(n_1) \cdot I_{f_i}(n_2)$$

1) 만약 $n_1$과 $n_2$가 다르다면, $C(n_1,n_2)$는 0을 돌려준다.
2) 그렇지 않고, n1과 and n2가 pre-terminal이라면, C(n1,n2)는 $1 \times \lambda$을 돌려준다.
3) 위의 2개를 만족하지 않는다면, 다음과 같이 계산된다.

$$C(n_1, n_2) = \lambda \prod_i^{nc(n_1)} (1 + C(ch(n_1, i), ch(n_2, i)))$$

여기서 $nc(n_1)$은 $n_1$노드의 자식의 빈도수를 의미하고, $ch(n_1,i)$는 노드 $n_1$의 $i$번째 자식 노드이다. 그리고 $\lambda$ (.0< $\lambda$<1)는 하위 트리의 크기에 따라 커널의 값을 줄여주기 위한 decay factor이다.

### 3.2 파스트리 커널과 지수 커널의 조합

두 커널의 조합은 아주 다양하게 만들어 질 수 있다. 그 다양한 조합들 중에서 본 논문에서는 아래의 식과 같이 파스트리 커널과 지수 커널을 조합한 복합 커널(composite kernel)을 다음과 같이 구성하여 사용하였다. 여기서 지수 커널을 사용한 이유는 의존관계분석에서 벡터로 표현된 자질을 다루는데 선형 커널보다 더 높은 성능을 보임을 보고되어서 본 논문에서는 선형커널보다 지수커널을 사용하였다.

$$Kc_1 = \alpha K_{tree}(R_1, R_2) + K_{poly}(R_1, R_2)$$
$$= \alpha K_{tree}(\mathrm{tf}_1, \mathrm{tf}_2) + K_{poly}(\mathrm{vf}_1, \mathrm{vf}_2) \quad (1)$$

$$Kc_2 = K_{tree}(R_1, R_2) \times K_{poly}(R_1, R_2)$$
$$= K_{tree}(\mathrm{tf}_1, \mathrm{tf}_2) \times K_{poly}(\mathrm{vf}_1, \mathrm{vf}_2) \quad (2)$$

여기서 $K_{tree}$, $K_{poly}$은 파스트리 커널 및 지수 커널을 의미하고, $\alpha$는 계수(coefficient)이다. $R_1$,$R_2$는 절의 의존관계 인스턴스를 의미하고, 파스트리 형태로 표현되어 있다. tf는 $R_1$, $R_2$에서 추출한 구조 정보(syntactic tree feature), vf는 벡터로 표현되어진 정적인 자질(static feature)을 나타내고 있다. 본 논문에서는 이들 각각의 조합에 대하여 각각 실험을 수행하고 최적의 조합을 구하였다.

## 4. 한국어 종속절의 의존 관계 분석

본 논문에서는 절들 간의 의존관계를 위해 앞 장에서 제안된 복합 커널을 사용한 SVM을 이용하여 의존관계를 분석하였다. 따라서 본 장에서는 의존관계를 파악하기 위한 확률 모델, SVM 사용을 위한 학습 데이터 생성 및 본 논문에서 해결하고자 하는 의존 관계 생성의 대상에 관해서 살펴본다.

### 4.1 확률 모델

본 연구에서는 절들 간의 의존관계를 위해 앞 장에서 제안된 복합 커널을 사용한 SVM을 이용하여 의존관계를 분석하였다. 한 문장에 속한 종속절들의 집합인 $C$는 $\{c1, c2, ..., c_n\}$으로 구성되고, 의존 관계의 패턴 집합 $D$는 $\{Dep(1), Dep(2), \cdots, Dep(n-1)\}$로 정의된다. 이때 $Dep(i)=j$는 절 $Ci$가 $Cj$와 의존관계에 있다는 것을 의미한다. 그림 2는 이들 관계를 간략하게 보여주고 있다. 절 $A$가 절 $C$와 의존관계를 성립한다면 $Dep(A)=C$로 나타낼 수 있다. 이때 절 $A$를 의존절, 절 $B$를 지배절이라고 정의한다.

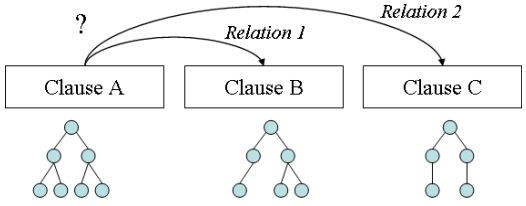


그림 2. 절들 간의 의존관계 정의

입력으로 C가 주어졌을 때 의존 관계 패턴인 D를 결정하는 문제는 P(D|C)가 최대로 되는 D를 찾는 문제로 볼 수 있고 이는 아래의 수식과 같이 정의할 수 있다

$$D_{best} = \arg\max_D P(D \mid C)$$

만약 의존 관계 패턴인 D가 독립이라고 가정하면 P(D|C)는 아래의 식과 같아진다. P(Dep(i)=j|fij)의 의미는 fij가 주어졌을 때 ci와 cj사이의 의존 관계가 존재하는 확률을 의미한다. 여기서 fij는 ci와 cj사이의 의존관계를 판단하는데 사용되는 다양한 언어적 및 표면적인 자질을 의미한다.

$$P(D \mid C) = \prod_{i=1}^{m-1} P(Dep(i) = j \mid \mathrm{f}_{ij}\}, \mathrm{f}_{ij} = \{f_1, ..., f_n\} \in \mathrm{R}^n$$

$$P(Dep(i) = j \mid \mathrm{f}'_{ij}) = \tanh\left( \sum_{k,l; f_{kl} \in SVs} \alpha_{kl} y_{kl} K(f_{kl}, \mathrm{f}'_{ij}) \right)$$

즉, SVM에서 나오는 마진의 값을 sigmoid 함수의 일종인 tanh로 취한 값이다. 이것은 의존관계의 확률을 나타낸다고 볼 수 있다[37].

이렇게 구해진 확률을 바탕으로 전체 문장에 속한 절들의 의존관계를 다음의 제약(constraints)를 바탕으로 폭포수 파싱 모델(cascading parsing)[37]을 사용하여 전체 문장에 속한 절들의 의존관계를 파악하였다.

1) 가장 마지막 절을 제외한 모든 의존절의 지배절은 항상 오른쪽에 나타난다.

2) 모든 의존절은 하나의 지배절만을 가지고, 지배절은 여러개의 의존절을 가질 수 있다.

3) 절들 간의 의존관계는 교차(cross)가 발생하지 않는다.

### 4.2 절들 간의 의존 관계 처리 대상

한국어의 절들은 다른 절과 연결되는 연결절, 명사구를 수식하는 관형절 그리고 문장의 종료를 의미하는 종결절 등의 3종류로 나누어진다. 본 논문에서는 절들의 의존 관계 분석 대상을 연결절이 다른 절과 의존관계가 성립되는 관계만을 대상으로 삼았다. 그 이유는 관형절은 다음에 나타나는 명사구를 수식하고, 종결절은 항상 문장의 마지막에 등장함으로 그 의존관계는 연결절에 비해서 의존 관계 파악에 용이한 편이다. 연결절은 문장 속에서 나타나는 위치 등에 따라 의존 관계가 성립되기보다는 어미의 변화와 문장 속에서 문맥에 따라 다양하게 의존 관계가 성립되어 의존관계 분석이 매우 어려운 작업이다.

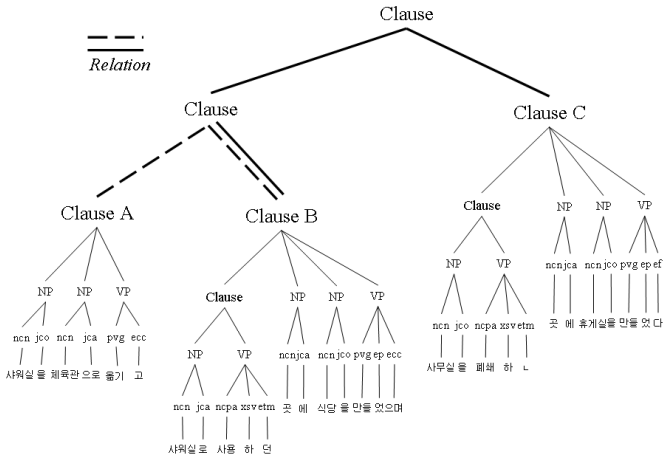



그림 3. 절의 구성 및 의존관계 형성의 예

그림 3은 한 문장에서 절의 구성과 절들 간의 의존관계의 예를 보여주고 있다. 이 문장에서는 모두 5개의 절을 가지고 있다. 이 절들 중에서 2개는 관형절, 마지막 하나는 종결절이다. 따라서 본 논문에서는 연결절인 총 3개의 절만을 대상으로 삼았고, 그 절의 내용은 다음과 같다.

| | |
|---|---|
| **절 A** | **(C** 샤워실을 체육관으로 옮기고**)** |
| **절 B** | **(C (C** 샤워실로 사용하던**)** 곳에 식당을 만들었으며**)** |
| **절 C** | **(C (C** 사무실을 폐쇄한**)** 곳에 휴게실을 만들었다.**)** |

각 절의 구성을 살펴보면 절A와 절B가 하나의 의존관계를 형성하고 있고, 절A와 절B가 결합된 절이 절C와 결합되어있다. 그리고 절 B와 절C는 하위절로 관형절을 하나씩 포함하고 있다.

## 5. 종속절의 의존 관계 분석을 위한 자질 선택

본 연구에서는 절들 간의 의존관계 분석을 위하여 2개의 다른 시각에서 접근하였다. 하나는 절의 문법적 구성에 따른 접근이고, 나머지는 절을 구성하고 있는 어휘 및 거리에 따른 접근이다. 이렇게 나누어 접근한 이유는 의존관계 분석에 문법적 구성도 중요하고, 절을 구성하는 단어 및 거리 등도 모두 중요하기 때문이다. 따라서 그림 4와 같이 그림 3에서 의존관계가 성립되는 절을 추출하여 다음의 자질들을 표현하였다.

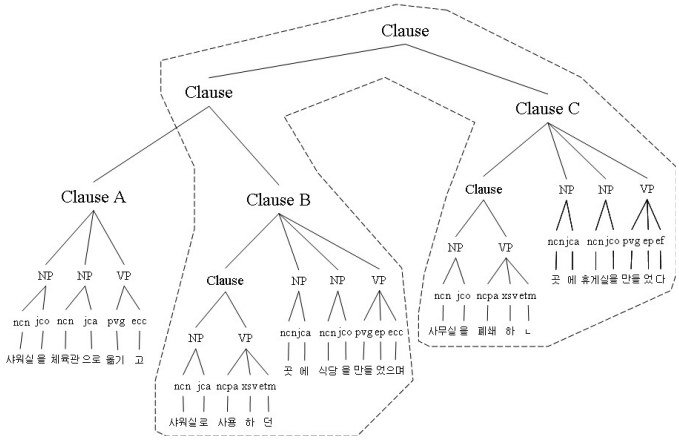



그림 4. 의존 관계 인스턴스 추출

### 5.1 문법적 구성 자질(Syntactic structure feature)

절의 의존관계 인스턴스는 파스 트리 속에 내부적으로 존재한다. 따라서 파스 트리에서 구조 정보를 추출하고, 표현하는 방법을 결정하는 일은 파스 트리를 사용하는데 있어서 아주 중요한 문제가 된다. 너무 많은 정보를 포함하게 되면 노이즈로 작용하여 성능을 저하시키고 너무 적은 정보를 가지고 처리하면 전반적인 성능의 저하를 불러온다. 이 문제는 절의 내부적 표현, 외부적 표현 그리고 부속절의 표현의 3개로 나누어 볼 수 있다.

**• 절의 내부적 표현**

파스 트리에서 하나의 절은 그림 5와 같이 4개의 계층으로 구성되어 있다. 본 논문에서는 마지막 word layer를

제외한 clause, phrase, POS layer만을 사용하였다. 이유로는 word layer에 속하는 각 단어들의 데이터의 희소성의 발생 문제와 정적 자질에서 이 단어들을 사용을 들 수 있다. 그림 2의 예제는 그림1의 'Clause A'이고, 인스턴스 공간 표현에 사용된 각각의 계층을 보여주고 있다.

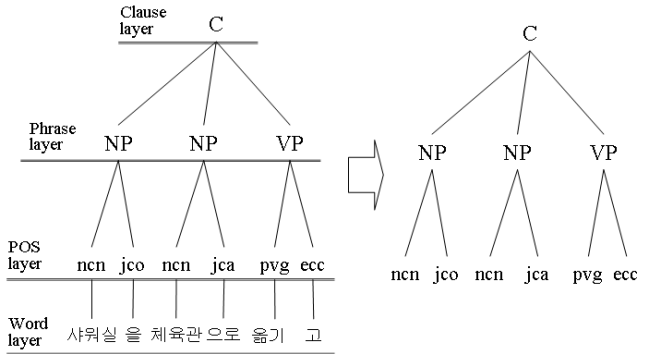



그림 5. 절의 내부적 표현

**• 절의 외부적 표현**

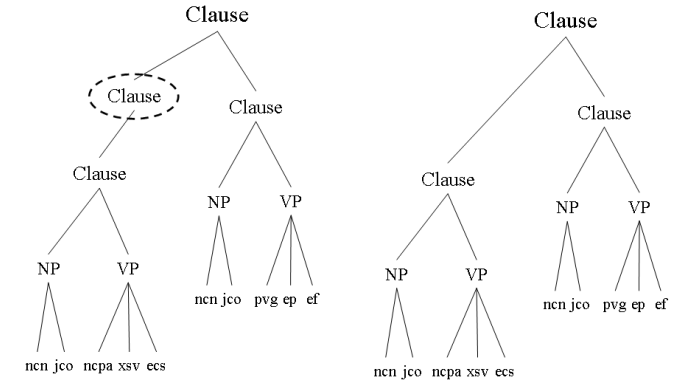
절의 외부적 표현은 절들 간의 수직적인 관계를 표현하는 문제로 바꾸어 볼 수 있다. 즉, 두 절들 사이의 연결되는 단일 노드를 어떻게 표현할 것인가를 결정하는 것이다. 이 경우에는 모든 단일 노드를 사용하는 경우와 제거하는 경우로 나누어 각각의 경우에 대하여 각각 고려하였다[1].

1) Path-enclosed Tree (PT)
- 트리로 표현된 2개의 절을 포함하고 가장 짧은 링크로 연결된 트리를 의미한다. 여기에서 단일 노드들을 모두 포함되어 있다. 그림 6의 (a)를 말한다.

2) Flattened Path-enclosed Tree (FPT)
- PT에서 절을 연결하는 노드들 중에서 중심이 되는 노드 하나를 제외한 모든 노드를 제거한 트리이다. 그림 6의 (b)를 말한다.

**• 부속절의 표현**

본 논문에서는 의존관계 대상을 연결절로 연결되는 절만을 대상으로 삼았다. 따라서 명사구를 수식하는 관형




(a)Path-enclosed Tree (b) Flattened Path-enclosed Tree

그림 6. 절의 외부적 표현

절은 처리대상에서 제외하였다. 그러나 하나의 연결절이 관형절을 포함하고 있을 때, 이들을 어떻게 표현하는 것인가에 따라 다음과 같이 나누어 고려하였다.

**• 부속절의 표현**

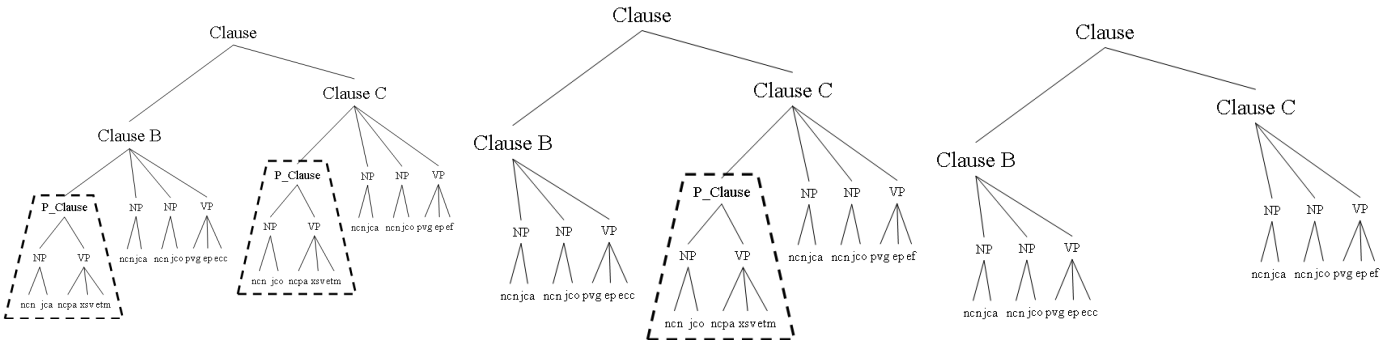
본 논문에서는 의존관계 대상을 연결절로 연결되는 절만을 대상으로 삼았다. 따라서 명사구를 수식하는 관형절은 처리대상에서 제외하였다. 그러나 하나의 연결절이 관형절을 포함하고 있을 때, 이들을 어떻게 표현하는 것인가에 따라 다음과 같이 나누어 고려하였다. 각각의 예는 그림 7의 (a),(b),(c)와 같다.

1) Complete Tree (CT)
- 지배절 및 의존절에 각각 관형절을 사용하는 것이다.

2) Context-Sensitive Tree (CST)
- 의존절의 관형절을 사용하지 않고, 지배절의 관형절을 사용하는 것을 의미한다.

3) Simple Tree (ST)
- 지배절 및 의존절에 나타나는 모든 관형절을 사용하지 않음을 의미한다.

이렇게 각각의 경우의 주로 명사구를 수식하는 관형절이 절들 간의 의존관계에 얼마만큼 영향을 주는지 확인




(a) Complete FPT(CFPT) (b) Context-sensitive FPT(CSFTP) (c) Simple FPT(SFPT)

그림 7. 관계 인스턴스 공간 표현의 예

하기위해서 파악하기 위함이다.

### 5.2 절의 정적 자질(static feature)의 사용

정적 자질은 그림 8과 같이 의존 관계에 있는 절들의 단어 및 POS 테그를 추출하고, 두 절들 간의 거리를 추출하여 벡터로 표현하여 사용하였다. 그림 8은 그림 2에서 절 B와 절 C의 의존 관계를 표현하고 있고, 이들 절간의 거리는 바로 이웃하고 있음으로 1로 나타낼 수 있다.

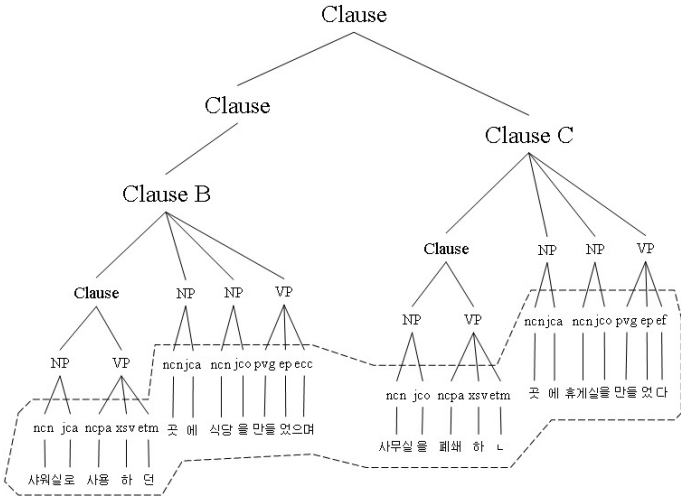



그림 8. 정적 자질의 추출

## 6. 실험 및 분석

본 논문에서 사용한 말뭉치는 STEP2000과제의 결과물인 구문 구조 부착 말뭉치(word, POS, Chunk tag가 표시)를 변형하여 만들었다. 코퍼스는 6,934개의 문장과 26,876개의 절로 구성되어있다. 여기서 90%의 문장인 6,240개를 학습에 사용하였고 나머지 10%에 해당되는 694개의 문장을 실험에 사용하였다. 표1은 실험에 사용한 코퍼스의 정보를 보여주고 있다. 여기에서 학습 및 실험에서 본 논문에서 의존관계 성립되는 연결절의 수는 약 30%를 차지하는 것을 알 수 있다.

표 1. 코퍼스의 구성 정보

| 분 류 | 학습데이터 | 실험데이터 |
|---|---|---|
| 문장의 수 | 6,240 | 694 |
| 모든 절의 수 | 24,226 | 2,650 |
| 관형절 및 종결절의 수 | 15,457 | 1,666 |
| 연결절의 수 | 8,769 | 984 |

본 논문에서는 파스 트리에서 표시되어 있는 절이 얼마나 정확하게 의존 관계대상 절을 인식 및 판단하는 가를 측정하기 위해서 평가 측도로 정확도(accuracy)를 사용하였다. 그리고 사용된 인식율은 의존관계를 형성하는 의존관계가 인식된 것을 의미하고, 절 단위는 한 문장의 모든 절에서 정확하게 지배절을 인식하는 것을 의미한다. 마지막으로 문장단위는 전체 말뭉치에서 모든 의존관계가 올바르게 파악된 문장의 비율을 의미한다.

가장 먼저 관계 인스턴스 공간에 따른 실험 결과를 살펴보면 표2와 같고, SVM은 SVM Light를 사용하였다[14]. 사용된 파라메타는 파스트리 커널(λ=0.4), SVM(C=0.4)을 사용하였고, 지수커널의 지수(dgree)는 2를 적용하였다.

실험은 크게 4개로 나누어 진행하였다. 먼저 앞에서 제안한 2개의 복합 커널 조합 방법에 대하여 실험하였고, 다음으로 절의 외부적 표현에 따른 성능을 평하하였다. 세 번째로는 절간의 의존관계 파악에 다른 방법보다 본 논문에서 제안한 방법이 높은 성능을 보임을 성능 평가해보고, 마지막으로 인스턴스 공간에 따른 성능을 평가하였다.

### 6.1 복합 커널 계수에 따른 성능의 변화

본격적인 실험에 앞서 두 개의 커널을 조합하기위한 계수에 따른 전반적인 성능을 실험해보았다. 먼저 절 외적 표현으로는 PT를 관현절의 표현은 ST를 사용하여 실험하였고, 사용한 커널의 조합은 식(1)을 사용하였다. 계수의 값을 0.1 만큼 변화하면서 실험을 수행하였다. 그 결과로는 값이 0.3일 때 가장 높은 결과를 얻었고, 전반적인 성능의 변화는 그림 9와 같다.

이 결과는 절들 간의 의존관계 형성에 문법적 구조 정보가 약 30%, 어휘 및 거리정보가 약 70%의 영향을 미친다고 볼 수 있다. 그러나 이 결과는 문장의 표면적으로 드러나는 정보를 바탕으로 한 것으로 의미(semantic) 정보가 사용하지 않아서 정확하지는 않다. 그러나 이는 본 논문에서 문법적 구조 정보를 사용하는 것이 유용하다는 것을 의미한다.

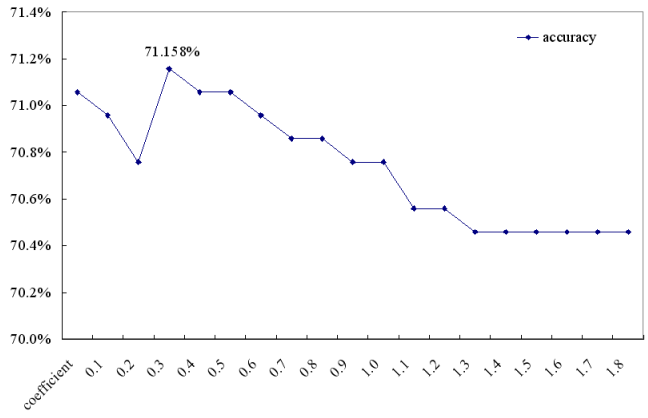



그림 9. 선형 조합에서 계수(coefficient) 변화에 따른 정확도

### 6.2 커널 조합에 따른 성능 변화

두 개의 커널을 조합하기위한 계수에 따른 전반적인 성능을 실험해보았다. 먼저 절 외적 표현으로는 PT를 관현절의 표현은 ST를 사용하여 실험하였다.

실험의 대상은 의존관계 대상을 인식, 인식된 절들을 대상으로 의존 대상 절 하나를 선택했을 경우와 각각 하나의 절을 선택했을 때 문장 전체가 정확하게 인식되는

경우를 각각 측정하였다. 실험에서 기준점은 나타난 절의 바로 다음 절을 의존관계의 대상으로 삼았을 경우를 의미한다. 즉 i번째의 절은 i+1번째의 절과 의존관계를 성립하는 경우이다.

표 2의 결과에 따르면 복합 커널 B가 A보다 높은 성능을 보였다. 여기서 복합커널 A는 식(1)에 따른 커널의 조합이고 계수 값은 앞선 실험에서 밝혀진 0.3을 사용하였고, 복합커널 B는 식(2)에 따른 결과 값이다. 여기서 커널의 다양한 조합에 따라서, 그 값이 조금씩 변화하는 것을 알 수 있다. 따라서 두 개 이상의 커널을 결합하여 사용할 때, 다양한 실험을 통해서 최적의 조건을 찾아야 한다.

표 2. 커널 조합에 따른 결과

| 관계 인스턴스 공간 | 절단위(Accuracy) % |
|---|---|
| 기준점(Base Line) | 57.50 |
| 복합 커널 A | 71.15 |
| 복합 커널 B | 74.14 |

### 6.3 복합 커널과 다른 모델과의 성능 평가

본 논문에서 제안한 복합 커널과 다른 모델과의 성능을 비교하였고, 그 결과는 표 3과 같다. 이 결과에 따르면, 기준점, 파스트리 커널만 사용한 것과 자질기반 SVM 보다 복합커널이 전반적으로 더 높은 성능을 보임을 알 수 있다. 이는 하나의 정보보다 복합 커널을 사용하여 보다 많은 정보를 사용함으로 보다 높은 성능을 얻을 수 있음을 보여준다.

여기서 파스트리 커널은 어휘 정보가 포함된 파스트리만을 추출하고, 이를 파스 트리 커널을 사용해서 실험한 결과 값이고, 자질 기반 SVM은 복합 커널에서 정적 자질만을 추출하여 사용한 값이다.

복합 커널(FPT)와 같은 공간에서 자질을 추출하여 사용하였다. 특히 FPT기반의 복합커널이 PT 기반의 복합커널보다 더 놓은 성능을 보였다. 이유로는 절들 간의 외부 연결에 단일 노드가 노이즈로 작용함을 알 수 있다.

표 3. 절의 외부적 연결 표현에 따른 성능 평가(단위:%)

| 관계 인스턴스 공간 | 인식률 | 절단위 | 문장단위 |
|---|---|---|---|
| 기준점(Base Line) | 57.50 | 57.50 | · |
| 파스트리 커널[10] | 89.12 | 61.89 | 62.19 |
| Feature_based SVM | 56.46 | 70.05 | 61.03 |
| 복합커널(PT) | 98.51 | 68.37 | 64.36 |
| **복합커널(FPT)** | **91.41** | **74.14** | **72.62** |

### 6.4 인스턴스 표현 공간에 따른 성능 평가

마지막으로 인스턴스 표현 공간에 따른 실험의 결과는 표 4와 같다. 이 실험은 절 외부 표현은 FPT를 사용하였고, 관형절의 변화에 따른 값을 바탕으로 관형절이 얼마나 많은 영향을 주고 있는지 알 수 있는 실험이다.

이 실험에서 'A'와 'B'의 구분은 A는 제약을 따르지만 cascading chunk 모델을 사용하지 않은 경우이고 'B'는 cascading chunk 모델을 사용하여 파싱한 경우를 보여주고 있다. 이 실험에서 cascading chunk 모델의 유용성과 complete tree의 경우에 가장 높은 성능을 보임을 알 수 있다. 그러나 context-sensitive tree와 큰 값을 보여주고 있지는 않아서, 유의미한 결과를 가지는지 더 보강 연구가 필요할 것으로 보인다.

표 4. 내부절의 표현에 따른 성능 평가(단위:%)

| 관계 인스턴스 공간 | | 인식률 | 절단위 | 문장단위 |
|---|---|---|---|---|
| 기준점(Base Line) | | 57.50 | 57.50 | · |
| complete tree | A | 95.32 | 77.42 | 73.37 |
| | ***B*** | ***98.62*** | ***83.31*** | ***78.13*** |
| context-sensitive tree | A | 95.48 | 80.56 | 77.21 |
| | B | 98.43 | 82.12 | 77.55 |
| simple tree | A | 96.35 | 68.69 | 71.42 |
| | B | 91.34 | 74.14 | 72.62 |

마지막으로 표 3와 4에서 문장단위의 성능이 전반적으로 60%이상의 성능을 보이는데, 이는 실험대상의 문장이 대체로 1-2개의 의존 관계를 가지는 문장이 많고, 입력대상이 절 단위로 인식된 상황이어서 높은 것으로 보인다.

## 6. 결론 및 향후 연구

본 논문에서는 한국어 문장에서 문법적 구조 정보와 정적 정보를 사용할 수 있는 복합 커널을 제안하고, 이 커널에 맞는 적합한 의존 관계 인스턴스 공간을 제안하였다. 여기에 사용된 방법은 구조 정보를 담당하는 파스트리 커널과 정적 정보를 담당하는 지수 커널을 결합하였고, 최적화된 관계 인스턴스 설정을 통하여 SVM을 사용하여 절들 간의 의존관계를 파악하였다.

최적화된 관계 인스턴스 공간을 결정하기 위해서 절들 간의 연결을 PT와 FPT 계열로 나누었고, 내부절을 CT, CST, ST로 나누어 각각 실험하고, 그 성능을 보였다. 그리고 결정된 인스턴스 공간에서 복합 커널을 사용한 본 연구의 방법이 좋은 성능을 발휘함을 보였다.

향후 연구에는 코퍼스 외부 자원(의미정보, 시소러스)을 활용하고, 다양한 자질 선택 및 커널의 개선을 통하여 성능을 개선하는 많은 연구가 있어야 할 것이다.

### Acknowledgements

이 논문은 2007년도 정부(과학기술부)의 재원으로 한국과학재단의 지원을 받아 수행된 연구임 (R01-2006-000-11196-0)

## 참고 문헌


[1] M.Zhang, J.Zhang, J.Su, and G.-D.Zhou, "A Composite Kernel to Extract Relations between Entities with Both Flat and Structured Features," In *Proceedings of COLING- ACL'2006*, pp. 825-832.

[2] R.Bunescu and R.Mooney, "A Shortest Path Dependency Kernel for Relation Extraction," In *Proceedings of EMNLP'2005*, pp. 724-731.

[3] X.Carreras and L. Marquez, "Boosting Trees for Clause Splitting", In *Proceedings of CoNLL'2001*, pp.73-75.

[4] A.Molina and F.Pla, "Clause Detection using HMM," In *Proceedings of the CoNLL'2001*, pp.73-75.

[5] K.Uchimoto, S.Sekine and H.Isahara, "Japanese Dependency Structure Analysis Based on Maximum Entropy Models," Proceedings of EACL'1999, pp.196-203.

[6] T.Kudo and Y.Matsumoto, "Japanese Dependency Analysis using Cascaded Chunking," In *Proceedings of the ACL'2002*, pp.1-7.

[7] H.-J. Lee, S.-B. Park, S.-J. Lee, and S.-Y Park, "Clause Boundary Recognition Using Support Vector Machines," In *Proceedings of PRICAI'2006*, pp. 505-514.

[8] 서광진, 어절 사이의 의존관계를 이용한 한국어 구문 분석기, 한국 과학기술원 석사학위 논문, 1993.

[9] H.Chung, *Statistical Korean Dependency Parsing Model based on the Surface Contextual Information*, 고려대학교 박사학위 논문, 2004.

[10] S-S Kim, S-B Park, and S-J Lee, "Analyzing Dependencies of Korean Subordinate Clauses using parse tree kernels," In *Proceedings of CICLing 2007*, pp. 218–.228.

[11] D.Zelenko, C.Aone and A.Richardella, kernel methods for relation extraction, *Journal of Machine Learning Research*, no.2, pp.1083-1106, 2003.

[12] R.C.Bunescu and R. J.Mooney, "A Shortest Path Dependency Kernel for Relation Extraction," In *Proceedings of EMNLP'2005,* PP.724-731.

[13] M.Collins and N.Duffy, "Convolution Kernels for Natural Language," In *Proceedings of NIPS'2001,* pp.625-632.

[14] T.Joachims, "Learning to Classify Text Using Support Vector Machines," Dissertation, Kluwer, 2002.

[15] T.Kudo and Y.Matsumoto, "Japanese Dependency Structure Analyisis Based on Support Vector Machines," In *Proceedings of joint Sigdata Conference on Empirical Methods in Natural Language Processing And Very Large Corpora*, pp.18-25, 2000.