의미 정보를 이용한 이단계 단문분할

박 현 재[†] · 우 요 섭[††]

요 약

단문분할은 한 문장에 용언이 복수개 있을 때 용언을 중심으로 문장을 나누는 방법이다. 기존의 방법은 정형화된 문장의 경우 비교적 효율적인 결과를 얻을 수 있으나, 구문적으로 복잡한 문장인 경우는 한계를 보였다. 본 논문에서는 이러한 한계를 극복하기 위해서 구문 정보만이 아니라, 의미 정보를 활용하여 단문을 분할하는 방법을 제안한다. 정형화된 문장의 경우와 달리 일상적인 문장은 문장 구조의 모호성이나 조사의 생략 등이 빈번하므로 의미 수준에서의 단문분할이 필요하다. 의미 영역에서 단문분할을 하면 기존의 구문 의존적인 방법들에서 발생하는 모호성을 상당수 해소할 수 있게 된다. 논문에서는 먼저 하위범주화 사전과 시소러스의 의미 정보를 이용하여 용언과 보어성분 간의 의존구조를 우선적으로 파악하고, 구문적인 정보와 기타 문법적인 지식을 사용하여 기타 성분을 의존구조에 점진적으로 포함시켜가는 이단계 단문분할 알고리즘을 제안한다. 제안된 이단계 단문분할 방법의 유용성을 보이기 위해 ETRI-KONAN의 말뭉치 중 25,000문장을 수작업으로 술어와 보어성분 간의 의존구조를 태깅한 후 본 논문에서 제안한 방법과 비교하는 실험을 수행하였으며, 이때 단문분할의 결과는 91.8%의 정확성을 보였다.

Two-Level Clausal Segmentation using Sense Information

Hyun-Jae Park[†] · Yo-Seop Woo[††]

ABSTRACT

Clausal segmentation is the method that parses Korean sentences by segmenting one long sentence into several phrases according to the predicates. So far most of researches could be useful for literary sentences, but long sentences increase complexities of the syntax analysis. Thus this paper proposed Two-Level Clausal Segmentation using sense information which was designed and implemented to solve this problem. Analysis of clausal segmentation and understanding of word senses can reduce syntactic and semantic ambiguity. Clausal segmentation using Sense Information is necessary because there are structural ambiguity of sentences and a frequent abbreviation of auxiliary word in common sentences.

Two-Level Clausal Segmentation System(TLCSS) consists of Complement Selection Process(CSP) and Noncomplement Expansion Process(NEP). CSP matches sentence elements to subcategorization dictionary and noun thesaurus. As a result of this step, we can find the complement and subcategorization pattern. Secondly, NEP is the method that uses syntactic property and the others methods for noncomplement increase of growth. As a result of this step, we acquire segmented sentences.

We present a technique to estimate the precision of Two-Level Clausal Segmentation System, and shows a result of Clausal Segmentation with 25,000 manually sense tagged corpus constructed by ETRI-KONAN group. An Two-Level Clausal Segmentation System shows clausal segmentation precision of 91.8%.

※ 본 연구는 정보통신부의 대학기초연구지원사업과 인천대학교의 일부 연구비 지원으로 수행되었음.

† 정 회 원 : 인천대학교 대학원 정보통신공학과

†† 정 회 원 : 인천대학교 정보통신공학과 교수

논문접수 : 2000년 6월 2일, 심사완료 : 2000년 9월 7일

1. 서 론

한국어 문장은 용언을 중심으로 하며 나머지 어휘 성분들은 부분적으로 자유로운 어순적 특성을 가지고 있으며, 여러 개의 용언들을 포함하고 있다. 단문분할은 용언을 기준으로 문장을 여러 개의 단문으로 나누는 방법이다. 자연언어 처리에서 단문분할의 중요성은 기존의 많은 연구들에서 제시되었다. 지금까지 한국어의 단문분할을 위해 사용되어온 방법은 크게 구문 패턴을 이용한 방법[1-4], 구문분석과 공기정보를 이용하는 방법[5], 하위범주화 사전을 이용하는 방법[6,7] 등으로 구분할 수 있다.

구문 패턴을 이용한 방법은 문장의 접속 구조를 찾아내고 문맥 정보를 포함하는 패턴으로 정의한 후 정의된 패턴에 따라 단문분할을 한다[1-4]. 즉 문장에서 문장의 분할이 가능한 모든 지점을 찾아낸 후 문장을 나누는 방법이다. 그러나, 이 방법은 두 가지 이상의 패턴이 적용되었을 때의 해결 방법이 미흡하고 관형절 처리에서 좋은 결과를 나타내지 못하였다.

구문분석과 공기정보를 이용하는 방법[5]은 통계적인 방법으로 단문분할 과정 중 구문분석의 모호성을 해소하기 위해 조사의 격과 공기정보를 이용하였다. 그러나, 적은 말뭉치보다는 많은 말뭉치에서 공기정보를 찾아 공기정보의 신뢰성을 얻을 필요가 있다. 또한, 조사에 의존하여 특정 분야의 말뭉치에서 단문분할을 하였으므로 보조사나 대용조사가 많은 말뭉치에 적용할 때에 좋은 효율성을 기대하기 힘들다. 그리고, 한국어 문장의 경우 조사의 생략이 빈번하므로 단문분할 과정 중에 나타나는 구조적 모호성을 해소하기가 어려운 문제점이 있다.

하위범주화 사전을 이용하여 단문을 분할하는 방법은 수작업에 의한 하위범주화 사전[6,7]을 이용하는 방법과 자동으로 하위범주화 사전을 구축한 하위범주화 사전[8]을 이용하는 방법이 있다. 기존의 수작업에 의한 하위범주화 사전 이용 방법은 적은 양의 용언 패턴을 구축하였으므로 문장에 적용할 때에 정확성이 낮았다. 또한 사람이 직접 기술하여 얻은 정보이므로 구축하는 사람마다 다른 결과를 나타낼 수 있고, 구조적 모호성이 존재하는 문장에 적용될 때에는 올바른 구조를 선택할 수 있는 정확한 기준이 되기 어렵다고 생각된다. 자동으로 하위범주화 사전을 구축하는 방법은 대량의 말뭉치로부터 통계에 기반한 패턴을 추출하여, 구문적인 모호성을 해결하고 단문분할 하고자 하는 방법으로, 이때 조사에 의존하여 패턴을 구축하는 것이 일반적이다. 그러나, 조사의 생략이 빈번한 문장에는 적용하기 힘들며, 자동 구축할 때에 나타나는 패턴 선정의 기준을 정하기가 어렵다.

본 논문에서는 단문분할할 때에 발생할 수 있는 구문적인 모호성 문제와 조사나 패턴에 의존한 단문분할의 한계를 극복하고 다양한 구조의 문장을 정확히 단문분할을 하기 위해 의미 정보를 가지고 있는 대규모의 하위범주화 사전[9], 계층적 개념을 가지고 있는 시소러스[10], 기타 한국어의 구문적인 특징을 사용하여 단문분할을 하는 이단계 단문분할 방법(Two-Level Clausal Segmentation System, TLCSS)을 제안한다. 대규모의 하위범주화 사전과 시소러스를 이용하여 문장 안의 용언이 가질 수 있는 핵심성분인 보어를 가진 후보를 모두 추출한 후, 후보들 중에서 하위범주화 사전의 의미 정보와 시소러스의 개념 정보를 이용하여 최적 후보를 선택하며, 특정한 구문적인 특성을 적용하여 보어성분을 확장한다. 그리고, 후처리 과정에서 남은 비핵심성분을 처리하는 이단계의 방법을 제안한다.

논문의 구성은 2장에서는 이단계 단문분할에서 사용하는 한국어 용언의 하위범주화 사전과 명사의 시소러스의 의미정보에 대해 간략히 기술하고, 3장에서는 이단계 단문분할 알고리즘을 제안하며, 4장에서는 본 논문에서 제안한 이단계 단문분할의 실험 및 결과를 보이고, 끝으로 5장에서는 결론과 앞으로의 연구방향을 제시한다.

2. 의미 정보

본 논문에서는 단문분할 시 발생하는 구조적 모호성을 해소하기 위해 하위범주화 사전과 시소러스의 의미정보를 이용한다. 하위범주화 사전과 시소러스의 규모 및 정확도는 본 논문에서 제안하는 이단계 단문분할 방법에 있어 중요하다. 그러나 이들 정보의 구축 방법이나 평가실험 등을 본 논문에서 상세히 기술하기는 어려우므로, 관련 문헌[9,10]을 참고하도록 하고, 여기서는 하위범주화 사전과 시소러스의 개략적인 구조와 의미정보에 대해 기술한다.

먼저 하위범주화 사전[9]은 말뭉치에서 추출된 19,000여개의 술어에 대해 25,000여개의 패턴이 정의된 것을

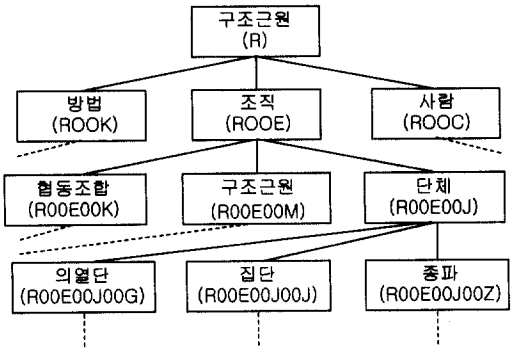
사용하였다. 이 사전은 41개의 동사 표준패턴과 17개의 형용사 표준패턴을 정의하고, 4개의 의미역(Semantic Role)을 기준으로 하여 반자동적인 방식으로 구축된 사전이다. 대규모의 사전이므로 일부 실험에서 사용되는 소규모의 하위범주화 사전과 같이 영역 의존적인 정보를 담고 있지 않으며, 범용 사전이므로 본 논문의 실험에 적합하다고 판단된다. 하위범주화 사전은 문형 slot과 의미 slot을 가지고 있고 구조는 (그림 1)과 같다.

```
붙다
[패턴번호] : 1153
[문  형] : 1. 이[AGT] 2.을 [ACC]
[의  미] : 1. 인간    2. 놀이기구, 악기
[의미코드] : 1. R00G00BOOUO1a01w
             2. ROOGOOAOOBO9r00J01SOOA,
                ROOGOOAOOBO9r00JO1m
```

(그림 1) 하위범주화 사전의 예

시소러스는 기존의 120,000개의 명사 계층 사전[10]을 본 논문에서 사용하는 하위범주화 사전과 부합되도록 수정하여 사용하였다. 시소러스 명사의 의미간 상하위 관계 파악이 용이하게 하기위해 의미 코드를 부여하였다. 의미 코드가 부여된 시소러스 구조의 예는 (그림 2)와 같다.

시소러스는 국어사전의 어의문을 추적하여 작성된 것을 수작업으로 정리하여 작성된 것으로, WordNet과 같은 동의어 수준의 사전이 아니라, 개념의 계층구조 사전이다. 동일한 어휘의 상의어가 복수개 있을 수 밖에 없으므로 시소러스의 구조는 네트워크 형태가 된다. 따라서 접두어식으로 의미코드를 부여할 때, 상의어가 복수개 존재하는 어휘의 경우는 그 어휘를 뿌리로 하는 별도의 계층을 다시 만들고, 두 계층구조를 연결하는 상의어 테이블을 별도로 두도록 하였다[9]. 이와 같이, 의미slot과 가지고 있는 하위범주화 사전과 개념 코드를 가지고 있는 계층적인 시소러스를 단문분할할 때에 정확한 결과를 얻기 위해 사용한다. 하위범주화 사전의 의미slot와 문장에 나타나는 명사의 시소러스 의미 코드를 비교하는 것으로 하위범주화 사전을 이용하여 문장의 보어성분을 찾아내어 문장의 구조를 파악할 때에 의미레벨의 비교를 함으로써 구조의 모호성을 줄일 수 있다. 의미slot과 의미코드 비교는 상하위 비교를 기본으로 하며 이 때 명사와 용언의 문장 내에서의 의미(sense)를 찾아낼 수 있다. 이러한 선택제약(Selectional Restriction) 방법[11, 12]을 본 논문에서 제안하는 이단계 단문분할에 사용한다.

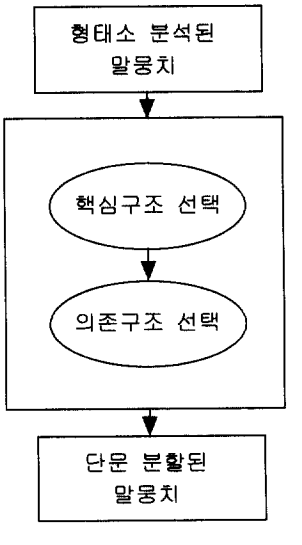


(그림 2) 시소러스 구조의 예

3. 단문분할 과정

3.1 이단계 단문분할 알고리즘

본 논문은 (그림 3)과 같이 형태소 분석된 말뭉치로부터 이단계의 과정을 거쳐 문장을 단문으로 분해하는 알고리즘을 제안한다.



(그림 3) 이단계 단문분할 구성도

형태소 분석은 단문분할은 물론 의미분석에 이르기까지 첫 단계이면서 가장 중요한 단계가 된다[11].

즉, 형태소 분석 과정의 오류나 모호성은 본 논문의 실험과정에 있어서도 정확도에 큰 영향을 주게 된다. 알고리즘에 초점을 맞추기 위해, 논문에서는 형태소 분석의 영향에 대한 설명은 제외하였으며, 실험과정에서는 수작업으로 검증된 형태소 분석 말뭉치[10]를 사용하였다.

이단계 단문분할 과정은 용언들이 문장에서 이끄는 보어를 찾아내는 핵심구조 선택 과정과 보어에 보어가 아닌 성분을 포함시키는 의존구조 확장 과정으로 단문분할을 수행한다. 이는 복잡한 문장을 단문분할하거나 구문분석을 할 때에는, 먼저 용언과 보어간의 의존관계를 찾고, 교차수식이 되지 않는다는 한국어 특성을 고려하여 핵심성분 간의 부분 의존관계를 먼저 파악하는 것이 구문 분석의 탐색공간을 크게 줄이고, 구문적인 오류나 모호성을 감소시켜 준다고 생각된다. 단문분할에 쓰이는 용언은 동사(VV), 형용사(VJ)와 용언화 접사를 포함하여 문장에서 용언과 같은 역할을 하는 성분으로, 보어는 용언이 문장에서 이끌 수 있는 필수성분으로 정의하며, 보어 이외의 체언을 비핵심성분으로 정의한다.

3.2 핵심구조 선택 알고리즘

본 논문에서는 단문분할을 하기 위해 문장의 핵심어(Head)인 용언을 추출한 후 문장에서의 용언이 이끄는 보어를 찾아내고 찾아낸 여러 가지 후보들 중에서 최적의 후보를 선정하기 위해 하위범주화 사전과 시소러스의 의미정보를 사용한다. 또한, 보어를 찾을 때에 지배성분의 왼쪽방향 뿐만 아니라, 오른쪽 방향의 일정범위의 성분들까지 비교함으로써 간단한 관형절의 경우도 단문분할이 가능하게 하였다. 핵심구조 선택 알고리즘은 (그림 4)와 같다. CountSentence는 한 문장의 길이를 나타내며, CountHead는 한 문장의 지배성분의 개수를 나타내는 변수이다.

먼저 문장 안에서 지배성분을 추출한다. *isHeadVerbal*() 는 문장 내에서 성분이 지배성분인지를 판단하는 과정이다. 지배성분을 본 논문에서는 동사(VV)와 형용사(VJ)인 용언과 용언화 접사('되/SF', '하/SF', '스럽/SF', '답/SF')를 포함하여 문장에서 용언으로 사용될 수 있는 요소를 말한다. 보조용언(VX)은 왼쪽의 가장 가까운 용언에 포함시키는 방법으로 지배성분에서 제외시킨다. *isHeadVerbal*()가 존재하는 경우 문장성분을 HeadVerbals[]에 넣는다. HeadVerbals[]은 문장성분 중 지배성분을 넣는 배열이다. *isDoubleQuote*() 는 지배성분을 찾는 과정 중 문장에 큰따옴표(")가 존재하는지를 판단하는 과정으로 문장에 큰따옴표가 존재하는 경우는 *findNextDoubleQuote*()과정을 거쳐서 문장내의 큰따옴표의 범위를 찾아낸다. *selectProcess*()는 큰따옴표내의 범위에서만 문장과 상관없이 단문분할 과정을 하는 과정이다.

문장 속에서 지배성분을 찾아낸 후 지배성분의 보어성분을 찾는다. *SearchSubcatDict*()는 하위범주화 사전에서 지배성분의 각각의 패턴을 찾아내어서 배열 SubcatforHeadverbal[]에 넣는다. *matchSubcatandThesaurus*()는 지배성분의 하위범주화 사전의 문형slot의 조사와 문장의 조사를 비교한다. 또한, 문장의 명사들의 시소러스의 의미성분과 하위범주화 사전의 의미slot과 상하위의 계층적 비교를 한 후 일치할 경우에는 보어성분으로 선택하여 배열MatchedSubcatforHeadVerbal[]에 넣는다. 만약에 조사가 없는 체언의 경우 체언의 시소러스에서의 의미 정보와 하위범주화 사전의 의미slot을 비교 한 후 일치할 때에는 보어성분으로 선택한다. 이 과정에서 비교하는 대상은 우선 지배성분의 왼쪽에 위치한 체언(명사(NN) 대명사(NP), 수사(NX), 의존 명사(NX))만을 대상으로 제한하였다. 이와 같은 과정을 *SearchSubcatDict*()에서 찾은 지배성분의 CountHead만큼 반복한다. 관형절과 같이 보어가 용언의 뒤에 있는 경우는 의존구조 확장 단계에서 처리한다.

하위범주화 사전과 시소러스의 의미 정합에 있어 상하위 계층 비교 이외에 개념 거리를 도입하는 방법도 고려할 수 있다. 앞서의 (그림 2)에서 시소러스의 노드를 접두어식으로 계층 인코딩한 것은 시소러스 네트워크의 활용에 있어 상하위 또는 인접 정도를 계산적으로 쉽게 파악하기 위한 것이다. 이렇게 어휘와 코드만을 DB에 담아두면 임의의 두 어휘(코드)간의 상하위 여부와 깊이, 형제 관계 여부 특정 노드로부터의 거리 차이 및 경로까지 단순한 스트링 매칭으로 결정할 수 있다. 그러나 형제노드까지 의미 정합에 사용하여 실험한 결과에서 그 유용성을 거의 찾을 수 없었다. 시소러스의 활용 방법에 따라서는 개념 거리를 정의하고 사용하는 것이 필요할 수도 있겠지만, 하위범주화 사전과의 정합에 있어서는 상위 개념인 하위범주화 사전 의미와 하위 개념인 시소러스 의미 사이의 상하위 비교만이 가치있다고 가정하고 본 논문을 기술한다.

```
//핵심 구조 선택 알고리즘
int CountSentence;

void selectProcess(0, CountSentence){
 int CountHead=0;
 int CountMatchedSubcat=0;
 int HeadVerbals[MaxWordsinSentence];
 SubcatItem SubcatforHeadVerbal[MaxSubItem];
 SubcatItem MatchedSubcatforHeadVerbal[MaxSubItem];

 for(int i=0, j=0 ; i<CountSentence ; i++ ){
   if(isHeadVerbal(i)){
     HeadVerbals[CountHead++]=i;
   }
   if(isDoubleQuote(i)){
     j=findNextDoubleQuote(i);
     selectProcess(i,j);
     i=j+1;
   }
 }

 for(int i=0; i<CountHead; i++){
   SubcatforHeadverbal[]=SearchSubcatDict(i);
   for(int j=0; j<length of SubcatforHeadVerbal[]; j++){
     MatchedSubcatforHeadVerbal[CountMatchedSub
     cat++]=matchSubcatandThesaurus(SubcatforHe
     adVerbal[j], i);
   }
   SelectCandidate(MatchedSubcatforHeadVerbal[],Count MatchedSubcat);
 }
}
```

(그림 4) 핵심구조 선택 알고리즘

selectCandidate()는 추출된 후보 중 가장 일치 성분이 많은 후보를 최적 후보로 선택하는 과정으로 동일한 개수인 경우는 주어의 가중치를 낮게 하여 후보를 선택한다. 최적 후보로 선택하는 과정을 문장의 지배성분의 수(CountHead)만큼 반복하여 모든 지배성분에 대해 최적 후보를 결정한다. 핵심구조 선택 알고리즘에 의해 지배성분의 보어 성분을 찾아내고 보어성분에 포함되지 않는 경우는 비핵심성분으로 분류한다. 문장에 큰따옴표("")등 인용부호가 있는 경우, 지배성분은 인용부호 내의 범위에서만 핵심성분인 보어성분을 추출하는 제약 조건을 두어 핵심구조 선택 알고리즘을 적용한다.

3.3 의존구조 확장 알고리즘

핵심구조 선택 알고리즘에서 선택된 용언의 보어에 비핵심성분을 포함시키는 단계를 거쳐 단문을 추출하는 과정으로 다음과 같은 알고리즘을 제안한다.

의존구조 확장 알고리즘은 (그림 5)와 같다. 의존구조 확장 알고리즘은 보어성분 확장하는 과정으로 비핵심성분을 문장의 특징을 고려한 후 보어성분에 선택한다. 지배성분의 보어성분을 제외한 나머지 성분을 문장의 끝인 오른쪽부터 찾아 나가면서 처리한다.

먼저, *isComplement*()는 문장성분이 지배성분의 보어성분인지를 판단하여 보어 성분인 경우 Current Complement로 채택한다. 지배 성분이 관형절의 형태를 포함하는 경우는 *ornProcesstoRight*()에서 하위범주화 사전의 문형 slot과 의미 slot을 비교한 후 명사성분을 보어성분에 포함시킨다. *isOrn*()는 문장의 지배성분이 간단한 관형절 형태 '지배성분 + [~는/EM,~ㄴ/EM,~ㄹ/EM,~되/SF]' 의 형태와 같은 경우를 검사하는 불린 함수이다.

```
// 의존 구조 확장 알고리즘
int CountSentence;

void expandProcess(0, CountSentence){
 int CurrentComplement;
 int ConjunctionFlag;

 for(int i=CountSentence; i>0; i--){
   if(isComplement(i)){
     CurrentComplement=i;
   }else if(isOrn(i)){
     for(intj=i+1;j<CountSentence;j++){
       ornProcesstoRight(i,j);
     }
     }else {
       if(CounjuctionFlag){
         searchConjunctionNounPhrase(i, CountSentence);
       }else if(searchComplexNounPhrase(i, CountSentence))
         searchVerbalforNoun(i, CountSentence);
       }
     }

     if(isConnector(i)){
       ConjunctionFlag=TRUE;
     }
   }

   for(int i=0; j<CountSentence;i++){
     if(isAux(i))
       searchVerbalforAux(0,i);
     else if(isAdverb(i))
       searchVerbalorAdjectforAdverb(i,CountSentence);
     else if(isAdject(i))
       searchNounforAdject(i,CountSentence);
     else if(isAaverb(i))
       searchVerbalforInjection(i,CountSentence);
     else if(isSymbol(i))
       SymbolProcess();
     execpProcess();
   }

 postProcessing();
}
```

(그림 5) 의존구조 확장 알고리즘

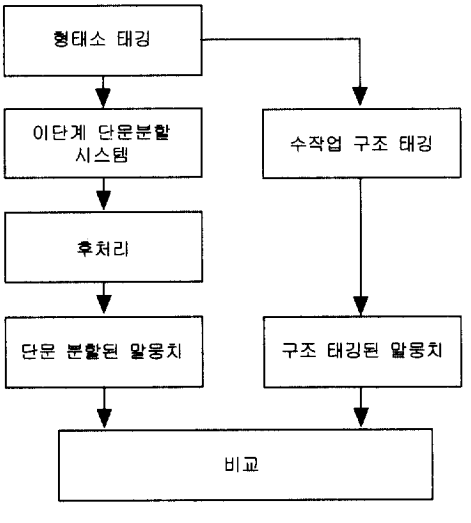
isConnector()는 비핵심성분이 ',', '그리고' 등의 접속어, 또는 '과/와/이나' 등을 포함하는 명사구를 판단하여 ConjunctionFlag를 결정한다. ConjunctionFlag가 TRUE인 경우 *searchCounjunctionNounPhrase*()에서 접속어를 처리한다. 예를 들어 ','로 연결되는 체언의 경우 앞의 성분이 보어로 선택된 경우 ','의 뒤에 있는 체언성분을 앞뒤의 체언과 같은 보어성분으로 선택한다. *searchComplexNounPhrase*()는 보어성분으로 채택

되지 않은 명사인 경우 바로 뒤의 명사가 보어성분인 경우 복합명사 사전과 비교하고 복합명사인 경우 *searchVerbalforNoun*()에서 CurrentComplement의 복합명사로 보어성분에 포함시킨다.*searchVerbalor AdjectforAdverb*(), *searchNounforAdject*(), *searchVerbalforInjection*(), *SymbolProcess*()는 부사(AD), 관형사(AJ)와 감탄사(IJ), 부호인 경우 가까운 보어성분에 포함시키는 과정이다. *execpProcess*()는 그 외의 예외적인 처리를 위한 부분이다. *postProcessing*()는 지배성분이 주어(AGT, CHD)인 보어성분을 포함하지 못하는 경우 문장의 맨 앞에 주어가 존재하면 포함시키는 과정과, 용언이 하위범주화 사전에 등록되지 않는 경우 가장 인접한 주어를 보어로 선택하여 포함하는 과정이다.

4. 실험 및 고찰

4.1 실험 방법

지금까지 제안한 이단계 단문분할의 타당성을 알아보기 위해서 말뭉치에 적용하는 실험을 하여 효용성을 검증한다. 실험 방법은 (그림 6)과 같다. 본 실험에 사용된 말뭉치는 ETRI-KONAN그룹의 말뭉치 중 25,000문장을 사용하였다. ETRI-KONAN그룹의 말뭉치는 여러 분야의 문장들이 포함되어 있으며 또한 구어체 문장들도 다수가 포함되어 있다[10]. 이단계 단문분할 시스템을 거쳐 나온 단문분할된 말뭉치 결과를 평가하기 위해 명사의 개념 태깅과 용언과 명사간의 의존 관계 등을 수작업으로 태깅한 말뭉치와 비교함으로써 정확도를 측정한다.



(그림 6) 단문분할 실험 과정

4.2 실험 결과

4.2.1지배성분 추출

본 논문에서는 단문의 지배성분을 용언과 용언화 접사가 붙어 용언을 역할을 하는 성분으로 정의한다. 논문에서 사용하는 말뭉치 25,000문장에서 용언과 용언화 접사를 가지고 있는 성분을 추출하였다.

〈표 1〉 실험에 사용된 지배성분

동사,형용사	70762개
용언화 접사를 포함한 성분	41764개
지배성분	112526개

실험에서 사용한 말뭉치는 ETRI-KONAN 그룹의 말뭉치는 <표 1>과 같이 동사와 형용사는 한 문장 당 2.83개, 용언화 접사를 포함한 성분은 1.67개로 한 문장 당 4.5개의 지배성분을 가지고 있고, 평균 8.02개의 어절수를 가지고있는 문장을 대상으로 실험을 수행하였다.

4.2.2 핵심구조 선택 과정

핵심구조 선택 과정은 문장에서 용언이 포함할 수 있는 보어성분을 추출하는 과정이다. 실험에서 사용한 하위범주화 사전은 19,000여개의 용언을 대상으로 25,000여 개의 패턴으로 구성되어 있다. 하위범주화 사전은 12만개의 계층적 의미코드를 부여한 의미 정보를 가지고 있는 시소러스의 정보를 포함하고 있다. 이러한 시소러스의 의미정보를 이용하여 문장내의 명사의 상하위 관계를 파악함으로써 보어성분의 의미정보를 파악할 수도 있는 부수적 효과가 있다.

추출된 용언을 대상으로 보어성분을 찾기 위해 본 논문에서 제안한 핵심구조 선택 알고리즘에 의해 실험을 하였다. 핵심성분인 보어를 선택하는 과정을 말뭉치 중에서 (그림 7)의 예문에 적용하여 단문분할하는 과정과 결과를 기술한다.

어제 밤에 집에서 바다로 고기를 잡으러 나가셨던 부모님께서 파도에 휩쓸려 돌아가셨습니다.

(그림 7) 예제 문장

(그림 7)의 예문에서 용언 '나가다'의 경우 하위범주화 사전의 각각의 패턴에 대한 후보를 생성된 것은 (그림 8)과 같다. 후보를 생성할 때에는 하위범주화 사전의 의미 정보와 문장의 체언들의 의미를 시소러스에서 찾아서 비교함으로써 의미 매칭된 경우를 보어성분으로 채택하게 된다. 이런 과정을 거친 후보들 중에서 가장 많이 보어 성분을 가지고 있는 경우 최적후보로 선택하게 된다. 만약 후보들이 보어성분의 수가 일치할 경우는 조사의 일치 여부를 판단하여 조사까지 매칭된 후보를 선택한다. '나가다'의 경우는 하위범주화 사전 700번인 '집에서 바다로 나가다'가 최적후보로 선택되게 된다. 용언으로 선택된 '잡다', '휩쓸리다', '돌아가다'와 같은 경우도 '나가다'와 같이 최적 후보를 선택한다. 이와 같은 핵심성분 선택 과정의 예문의 결과는 (그림 9)와 같다.

SubNum	후 보		
685	-	-	바다로 나가다.
686	-	-	바다로 나가다.
700	-	집에서	바다로 나가다.
701	-	집에서	- 나가다.
702	-	집에서	- 나가다.

(그림 8) 후보 생성

핵심 구조 추출
고기를 잡다.
집에서 바다로 나가다.
휩쓸리다.
부모님께서 돌아가다.

(그림 9) 핵심구조 선택 과정 결과

실험에서 사용한 112,526개의 지배성분 중 하위범주화 사전에 미등록 된 지배성분을 살펴보면 486개(0.43%)가 하위범주화 사전에 미등록되어 있고 그 중에 331개는 사역형과 피동형 형태이었다. 예제에서는 '휩쓸리다'의 경우 하위범주화 사전에 등록되어 있지 않는 용언이다. 사역, 피동형 동사가 출현한 경우는 하위범주의 변형이 필요한 부분이고, 변형 규칙에 의해 일부 처리[9]할 수 있지만 실험에서는 제외하였다.

본 실험에서는 최적 후보의 보어성분들의 정확성을 평가하기 위해 반자동적인 방법으로 의미 태깅된 25,000문장과 비교함으로써 정확도를 평가하였다. 의미 태깅된 보어성분과 핵심구조 선택 과정의 결과인 최적 후보를 비교한 결과는 <표 2>와 같다.

<표 2>의 결과는 미등록된 지배성분을 제외한 지배성분을 대상으로 하였다. 완전 매칭의 경우는 지배성분의 하위범주화 사전 패턴 의미성분과 문장의 성분과 모두 일치한 경우이고, 부분 매칭은 일부 의미성분이 문장의 성분과 일치하지 않는 경우이다. 부분 매칭의 경우는 하위범주화 사전과 시소러스가 충분한 정보를 담고있지 못하기 때문으로 대규모 어휘를 대상으로 한 경우에는 불가피한 것으로 판단된다. 또한, 의미 정보를 단문분할에 이용함으로써 용언과 명사의 어휘 의미를 단문분할과 동시에 파악할 수 있는 장점도 있다.

〈표 2〉 의미 태깅된 문장과의 정확도

완전 매칭	80%
부분 매칭	20%

4.2.3 의존구조 확장 과정

핵심구조 선택 과정에서 추출된 최적 후보에서 선택된 보어성분 이외의 성분 및 관형절과 예외 처리를 하는 과정으로 본 논문에서 제안한 의존구조 확장 알고리즘에 의해 실험을 하였다.

먼저 복합명사의 경우는 보어성분 왼쪽에 인접한 하나의 보어성분이 아닌 명사와 붙여서 복합명사 사전과 비교한 후 복합명사 사전에 등록되어 있는 명사의 경우 보어성분이 아닌 명사를 보어성분으로 포함시킨다. 복합명사 처리 후 한국어 문장에서 많이 나타나는 구문적인 특징인 관형절 처리를 하였다. 본 실험에서 사용된 말뭉치의 지배성분 중 12,154개인 10.8%가 관형절의 형태를 가지고 있다. 또한, 최적후보 선택의 경우 후보가 주격성분을 가지고 있지 않을 때는 오른쪽으로 가까운 2개의 문장 성분 중 관형절의 형태가 있는 경우에는 관형절 형태의 지배성분의 오른쪽 성분을 하위범주화의 주격성분과 의미 매칭을 하여 매칭된 경우에는 보어성분으로 선정하도록 하였다. 관형절 처리의 과정을 거친 경우 관형절 형태의 문장 중84%(10,209개)가 올바른 단문분할 결과를 보여 관형절 처리전의 정확도보다 8.9%의 향상을 보였다. 관형절 처리 후 남은 비핵심성분을 의존구조 확장 알고리즘에 의해 예문

을 처리한 결과는 (그림 10)과 같다.

```
의존 구조 추출
부모님께서 어제 밤에 고기를 잡다.
부모님께서 어제 밤에 집에서 바다로 나간다.
휩쓸리다.
부모님께서 돌아가다.
```

(그림 10) 의존구조 확장 결과

단문분할 과정의 결과 가장 많은 오류를 나타내는 것은 용언이 이끄는 보어성분 중 먼 거리에 있는 주어 성분을 올바르게 찾지 못하는 경우였다. 이런 오류를 줄이기 위해 후처리 과정을 수행한다. 후처리 과정으로 용언이 주어(AGT,CHD)인 보어성분을 포함하지 못하는 경우 문장의 맨 앞에 주어가 존재하면 포함시키는 과정과, 용언이 하위범주화 사전에 등록되지 않는 경우 가장 인접한 주어를 보어로 선택하여 포함하는 과정 등이다. 예문의 경우 '휩쓸리다'에 주어인 '부모님께서'가 보어로 포함이 되었다. 이런 후처리 과정의 결과 2.6%의 향상을 보였다.

본 논문의 예문의 이단계 단문분할 시스템의 단문분할 결과는 (그림 11)과 같다. 이단계 단문분할 시스템 결과는 <표 3>과 같다.

```
단문 분할 결과
부모님께서 어제 밤에 고기를 잡다.
부모님께서 어제 밤에 집에서 바다로 나가다.
부모님께서 파도에 휩쓸리다.
부모님께서 돌아가다.
```

(그림 11) 단문분할 결과

〈표 3〉 이단계 단문분할 과정의 결과

전체 용언 112526개 중		
후처리 전	100,373개	정확도 89.2%
후처리 후	103,389개	정확도 91.8%

본 논문은 의미 정보를 단문분할에 이용함으로써 특정 분야가 아닌 조사가 생략된 일반적인 말뭉치에서도 정확한 단문분할 결과를 얻을 수 있었다. 또한, 단문분할의 결과로 단문 단위의 문장들을 얻을 수 있는 것은 물론, 구문분석과정을 거치지 않고도 단문내에서 주요 성분들간의 의존관계를 파악할 수 있었다. 핵심구조 선택 단계과정에서 하위범주화 사전과 시소러스의 매칭을 통해 지배성분과 보어의 의미도 파악이 가능하였다.

본 실험과정에서의 실패 원인을 분석해 보면, 첫째 핵심구조 선택 과정에서는 하위범주화 사전에 용언이 포함 되지않은 경우(0.43%)가 있었으므로 올바른 결과를 얻을 수 없는 경우가 있었으며, 둘째로 하위범주화 사전과 시소러스의 정확한 매칭이 안 되는 경우가 존재하여 보어성분의 선택이 되지 않는 경우와 셋째로 먼 거리에 보어성분이 있는 경우에 정확히 보어성분을 선택하지 못하는 경우가 일반적인 실패 원인이었다. 기타 형태소 태깅이 잘못된 경우와 하위범주화 사전의 용언의 패턴이 부족한 경우 등도 나머지 실패의 원인이 되었다.

단문분할의 성공률을 향상시키기 위해서는 이러한 실패 요인들을 해결하는 경험적인 규칙을 의존구조 확장 단계와 후처리 단계에서 추가할 필요가 있을 것으로 생각된다.

5. 결 론

본 논문은 한국어 문장을 단문으로 분리하는 이단계 단문분할 시스템을 설계하고 구현하였다. 제안된 이단계 단문분할 시스템은 단문분할 과정에서 구문 정보뿐만 아니라, 의미 정보를 이용한 선택 제약 방법을 사용함으로써 단문분할할 때에 나타날 수 있는 모호성을 줄여 보다 정확한 단문분할이 가능하다.

본 논문에서는 핵심구조 선택 단계와 의존구조 확장 단계로 수행되는 이단계 단문분할 알고리즘을 제안하였다. ETRI-KONAN의 말뭉치 중에서 25,000문장을 대상으로 실험을 하였고 단문분할을 한 결과 91.8%의 성공률을 보였다. 본 논문에서 제안된 방법은 의미 정보를 단문분할에 이용함으로써 용언과 명사의 어휘 의미를 단문분할과 동시에 파악할 수도 있는 장점도 있었다.

본 논문에서 사용하는 하위범주화 사전의 구문 정보와 의미 정보의 정확성은 단문분할 과정의 정확도에 큰 영향을 주기 때문에, 앞으로 하위범주화 사전의 보완 작업을 통해 시소러스와의 의미 정보의 비교의 정확성을 높이고 단문분할할 때에 구문 정보나 경험적인 규칙을 적용함으로써 단문분할 결과의 정확성을 향상

시켜 나가는 연구를 지속하고자 한다. 의미slot과 의미코드 비교는 상하위 비교만을 하였으나, 일치되지 않는 경우에는 의미코드를 이용하여 개념 거리를 계산하여 일치 여부를 판단하는 것을 고려하는 문제와 단문 분할 결과를 하위범주화 사전에 피드백함으로써 하위범주화 사전을 보완하는 연구를 진행중이다. 이러한 연구를 통해 최종적으로 문장내의 명사의 의미를 보다 정확히 찾아내는 WSD(Word Sense Disambiguation) 연구에 기여할 것을 기대한다.

참 고 문 헌

[1] 이호, 백대호, 임해창, "분류 정보를 이용한 단어 의미 중의성 해결", 한국정보과학회논문지 제24권 제7호, pp.779-789, 1997.

[2] 양단희, 송만석, "말뭉치로부터 격틀 구축에 필요한 학습 데이터 추출", 제 10회 한글 및 한국어 정보처리 학술발표 논문집, pp. 287-292, 1998.

[3] 김광진, 송영훈, 이정현, "복합문에서의 단문 추출 시스템의 설계 및 구현", 산업과학기술연구소 논문집, 제22권, pp.373-380, 1994.

[4] 박성배, "문장분할을 이용한 한국어 분석", 서울대학교 컴퓨터공학과 석사학위논문, 1996.

[5] 이현아, 이종혁, 이근배, "단문분할을 통한 명사구 색인 방법", 한국정보과학회논문지, 제24권 제3호, pp. 302-311, 1997.

[6] Yorick Wilks and Mark Stevenson, "Sense Tagging : Semantic Tagging with a Lexicon," *Proc. of the SIGLEX Workshop*, pp.74-78, 1997.

[7] Ralph Grishman, Catherine Macleod, Adam Meyers, "Complex Syntax : Build a Computational Lexicon," *Proc. of COLING-94*, pp.268-272, 1994.

[8] 김나리, "패턴 정보를 이용한 한국어 구문 분석", 서울대학교 컴퓨터공학과 박사학위논문, 1997.

[9] 우요섭, 윤덕호, 양승현, 김영섭, "시소러스와 술어 패턴을 이용한 의미역 부착 한국어 하위범주화 사전의 구축", 한국정보과학회논문지, pp.364-372, Jun. 2000.

[10] 서영훈 외, "토큰 기반 한국어 분석기 개발- 한국어 의미 분석 사전 및 하위범주화 사전구축", 한국전자통신연구원 보고서, 1998.

[11] Yorick Wilks and Mark Stevenson, "The Grammar of Sense : Using part-of-speech tags as a first step in semantic disambiguation," *Natural Language Engineering, Volume 4*, pp.135-143, Jun. 1998.

[12] Adam Kilgarriff, "What is word sense disambiguation good for?" *Proc. Natural Language Processing Pacific Rim*, pp.209-214, Dec. 1997.

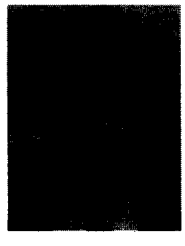
박 현 재

e-mail : now_hj@hanmail.net

1999년 인천대학교 정보통신공학과 졸업(학사)

1999년~현재 인천대학교 정보통신공학과 석사과정 재학중

관심분야 : 인공지능, 데이터베이스 정보검색



우 요 섭

e-mail : yswooo@lion.inchon.ac.kr

1986년 한양대학교 전자통신공학과

1988년 한양대학교 대학원 전자통신공학과(공학석사)

1992년 한양대학교 대학원 전자통신공학과(공학박사)

1992년~1994년 인천대학교 정보통신공학과 조교수

1994년~현재 (시립)인천대학교 정보통신공학과 부교수

관심분야 : 한국어 정보처리, 멀티미디어 정보검색, VOD