효율적인 영어 구문 분석을 위한 최대 엔트로피 모델에 의한 문장 분할

(Intra-Sentence Segmentation using Maximum Entropy Model for Efficient Parsing of English Sentences)

김 성 동 †

(Sung-Dong Kim)

요 약 긴 문장 분석은 높은 분석 복잡도로 인해 기계 번역에서 매우 어려운 문제이다. 구문 분석의 복잡도를 줄이기 위하여 문장 분할 방법이 제안되었으며 본 논문에서는 문장 분할의 적용률과 정확도를 높이기 위한 최대 엔트로피 확률 모델 기반의 문장 분할 방법을 제시한다. 분할 위치의 어휘 문맥적 특징을 추출하여 후보 분할 위치를 선정하는 규칙을 학습을 통해 자동적으로 획득하고 각 후보 분할 위치에 분할 확률 값을 제공하는 확률 모델을 생성한다. 어휘 문맥은 문장 분할 위치가 표시된 말뭉치로부터 추출되며 최대 엔트로피 원리에 기반하여 확률 모델에 결합된다. Wall Street Journal의 문장을 추출하여 학습 데이타를 생성하는 말뭉치를 구축하고 네 개의 서로 다른 영역으로부터 문장을 추출하여 문장 분할 실험을 하였다. 실험을 통해 약 88%의 문장 분할의 정확도와 약 98%의 적용률을 보였다. 또한 문장 분할이 효율적인 파싱에 기여하는 정도를 측정하여 분석 시간 면에서 약 4.8배, 공간 면에서 약 3.6배의 분석 효율이 향상되었음을 확인하였다.

키워드 : 문장 분할, 최대 엔트로피 모델, 구문 분석, 기계번역

Abstract Long sentence analysis has been a critical problem in machine translation because of high complexity. The methods of intra-sentence segmentation have been proposed to reduce parsing complexity. This paper presents the intra-sentence segmentation method based on maximum entropy probability model to increase the coverage and accuracy of the segmentation. We construct the rules for choosing candidate segmentation positions by a learning method using the lexical context of the words tagged as segmentation position. We also generate the model that gives probability value to each candidate segmentation positions. The lexical contexts are extracted from the corpus tagged with segmentation positions and are incorporated into the probability model. We construct training data using the sentences from Wall Street Journal and experiment the intra-sentence segmentation on the sentences from four different domains. The experiments show about 88% accuracy and about 98% coverage of the segmentation. Also, the proposed method results in parsing efficiency improvement by 4.8 times in speed and 3.6 times in space.

Key words : intra-sentence segmentation, maximum entropy model, parsing, machine translation

1. 서 론

인터넷의 보급으로 기계번역 시스템의 필요성이 급속하게 증가하고 있는 현실에서 실용적인 기계번역 시스템을 위한 구문 분석기는 실생활에 나타나는 다양한 형식의 문장들을 분석할 수 있어야 한다. 이러한 문장들의 분석을 위해서는 매우 많은 규칙이 필요하며 많은 규칙을 이용한 구문 분석은 많은 시간과 공간을 필요로 한다. 본 논문에서는 구문 분석의 적용률을 유지하기 위해 구문 분석을 위한 문법의 크기를 줄이지 않으면서, 구문 분석 대상이 되는 문장의 길이를 줄임으로써 긴 문장을 효율적으로 분석하기 위한 문장 분할 방법을 제안한다. 문서 분류(text categorization)[1]나 문장 경계 찾기(sentence boundary identification)[2,3]에서의 문장 분할(inter-sentence segmentation)과 구분하기 위해서 본 논문에서의 문장 분할은 문장 내부 분할(intra-sentence segmentation)이라 부른다.

· 본 연구는 한성대학교 2004년도 교내연구비 지원과제임

† 종신회원 : 한성대학교 컴퓨터공학부 교수

sdkim@hansung.ac.kr

논문접수 : 2003년 7월 15일

심사완료 : 2005년 4월 1일

문장 분할을 이용하여 구문 분석의 성능을 개선하기 위한 여러 연구가 있었다. [4]에서는 영어 문장을 세 개의 부분으로 분할하여 구문 분석의 복잡도를 줄인다. 이 연구에서는 영어의 평서문이 세 개의 분할로 이루어진다는 사실을 이용하였다: 주어 앞 부분 - 주어부 - 서술부. 또한 [5]에서는 신경망(neural network)의 패턴 매칭 능력을 이용하여 평서문의 세 부분의 위치를 찾는 방법을 소개하였다. 예문 기반 기계 번역(example-based machine translation)에서 효율적인 분석을 위해 구성 성분 경계 분석(constituent boundary parsing) 방법이 제안되었다[6]. 이 연구에서는 원시 문장의 분석을 위해 예문 기반 방법과 원시 언어 분석 방법을 결합하여 예문 기반 방법의 장점을 활용하는데, 원시 문장의 분석의 복잡도를 줄이기 위해 언어학적 패턴(linguistic pattern) 매칭을 도입하였다. 또한 문장 분할을 위해 문장 패턴(sentence pattern)을 이용하여 긴 문장의 구성 형식을 명시하고 이를 이용하여 문장을 분할하는 연구가 있었다[7]. 여기서는 정의된 패턴에 의해 입력 문장을 분할하고, 각각의 분할을 독립적으로 분석하고 그 결과를 합성하여 전체 문장의 구조를 생성하는 3 단계 과정을 거쳐 문장을 분석한다.

위의 방법들은 구문 분석의 복잡도를 줄이기 위해 제안되었는데, 문장 분할의 적용률이 매우 낮기 때문에 실용적인 기계 번역 시스템에 적용하기가 어려우며 문장 패턴을 얻기 위해 매우 많은 사람의 노력을 필요로 한다[7]. 문장 분할을 실용적인 시스템에 적용할 수 있기 위해서는 문장의 유형이나 길이에 관계없이 문장 분할을 할 수 있어야 한다. 본 논문에서는 문장 분할의 후보를 찾고 그 중에서 가장 적절한 분할 위치를 선택하는 학습 가능한 모델을 제시한다. 제안된 모델은 문장 분할 위치가 표시된 말뭉치를 이용하여 분할 위치에 대한 문맥 정보를 학습하고 후보 분할 위치를 찾기 위한 규칙을 생성한다. 그리고 각 후보 분할 위치의 확률은 최대 엔트로피 원리에 기초하여 생성되는 확률 분포에 의해 정해진다. 주어진 사실만을 고려하여 가장 공평한 확률 분포는 최대 엔트로피를 가지는 확률 분포이며[8], 최근에 통계적인 방법의 자연언어 처리의 분야에서 널리 이용되고 있다[9,10]. 본 논문에서 제시된 문장 분할은 영한 기계번역 시스템에서 차트 기반의 문맥 자유 문법 파서를 위한 전 단계의 역할을 한다. 이는 보다 효율적인 구문 분석을 가능하게 함으로써 영한 기계 번역 시스템의 주요 구성 요소로 활용되고 있다.

2장에서는 최대 엔트로피 모델링 방법을 설명하며 3장에서는 어휘 문맥의 구조와 표현 방법을 제시하고 후보 분할 위치 선정 방법을 설명한다. 4장에서는 분할 위치 결정을 위한 확률 분포 생성 방법을 5장에서는 분할 위치 결정 과정을 설명한다. 6장에서는 본 논문에서 제시한 문장 분할 모델의 성능을 제시하고 다른 문장 분할 방법과 비교한다. 7장에서 본 논문을 결론지으며 몇 가지 추후 과제를 제시한다.

2. 최대 엔트로피 모델링

문장 분할을 위해 논문에서 제시한 방법은 통계적인 모델을 생성하는데 매우 강력한 방법으로 알려져 있는 최대 엔트로피 원리를 이용한다[11]. 최대 엔트로피 모델링 방법은 여러 가지 제한 요소들(constraints)을 손쉽게 결합하는 방법을 제공하며 학습 데이타를 분할하거나 필요한 파라미터의 개수를 증가시키지 않고 복잡한 사건의 교집합을 모델화하기 때문에 통계적으로 효율적인 방법이라고 할 수 있다. 또한 이것은 다른 통계적인 방법에 비해 성능이 좋은 모델을 제공하는데, 예를 들어 최대 엔트로피 trigram은 interpolated trigram [12]나 back-off[13]보다 음성 인식에서 좋은 결과를 보였다. 이러한 이유들 때문에 본 논문에서는 최대 엔트로피 원리를 적용한다.

2.1 최대 엔트로피 원리(Maximum Entropy Principle)

본 논문에서는 문장에 있는 후보 분할 위치에 확률을 주는 확률 모델 $p(y|x)$를 생성한다. 여기서 $y \in 0, 1$는 문맥 x를 가질 때 후보 분할 위치인지를 나타내는 확률 변수이다. 이러한 확률 모델은 문맥이 가지는 서로 다른 특성(feature)들의 가중치를 결정하는 로그-선형 모델(log-linear model)을 구축하여 생성할 수 있다. 특성은 2진 값을 갖는 함수 f이며, 분할 위치의 문맥에 대한 정보를 표현한다.

분할 위치가 표시된 크기 N의 말뭉치가 주어질 때, 경험적 분포(empirical distribution) $\tilde{p}(x, y)$는 다음과 같이 정의한다.

$$\tilde{p}(x, y) = \frac{|(x, y)|}{N}.$$

여기서 생성하려는 모델은 $\tilde{p}(x, y)$에 가장 잘 근사하는 확률 모델이며 특성들의 확률에 대한 기대값으로 확률 모델의 근사 정도를 판단한다. $\tilde{p}(x, y)$에 대한 특성 f_i의 기대값은 다음과 같이 표현된다.

$$\tilde{p}(f_i) \equiv \sum_{x,y} \tilde{p}(x, y) f_i(x, y).$$

그리고 확률 모델 $p(y \mid x)$에 대한 f_i의 기대값은 다음과 같다.

$$p(f_i) \equiv \sum_{x,y} \tilde{p}(x) p(y|x) f_i(x, y).$$

여기서 $\tilde{p}$는 말뭉치에서의 x의 경험적 분포이다. 본 논문에서 생성하고자 하는 문장 분할을 위한 확률 모델

은 문장 분할에서 고려되는 특성들이 말뭉치에서 나타내는 통계를 올바르게 반영하는 것이며 이는 모든 특성 $f_i \in F$[1]에 대해서 $p(f_i) = \tilde{p}(f_i)$라는 제한 조건으로 표현될 수 있으며 다음과 같이 나타낼 수 있다.

$$\sum_{x,y} \tilde{p}(x)\,p(y|x)f_i(x,y) = \sum_{x,y} \tilde{p}(x,y)f_i(x,y) \quad (1)$$

특성의 집합 F가 주어졌을 때, 제한 조건 (1)을 만족하는 모든 확률 분포 중에서 가장 공평한 확률 분포 p를 구하는 문제에 대하여 최대 엔트로피 원리가 이용되는데 확률 분포 $p(y|x)$의 공평도(uniformity)는 다음과 같은 조건부 엔트로피로 측정된다.

$$H(p) \equiv -\sum_{x,y} \tilde{p}(x)\,p(y|x)\log p(y|x) \quad (2)$$

최대 엔트로피를 가지는 확률 분포가 학습 데이타에서 얻는 정보를 충실하게 반영하는 가장 공평한 확률 분포가 되며 이를 최대 엔트로피 원리라고 부른다.

2.2 최대 유사도 원리(Maximum Likelihood Principle)

모델을 생성할 때, 지수 분포가 다루기 쉽고 실제 분포에 가까운 분포라는 특징이 있기 때문에[14] 확률 변수 x, y가 지수적으로 분포(exponentially distributed)되어 있다는 가정을 한다. 선형 지수 계열(linear exponential family)의 확률 분포 Q는 다음과 같이 표현된다.

$$Q(f) = \{\ p(y|x) = \frac{1}{Z_\lambda(x)} \exp\left(\sum_i \lambda_i f_i(x,y)\right)\ \} \quad (3)$$

여기서 λ_i는 실수 값을 갖는 특성 파라미터이고 $Z_\lambda(x)$는 확률 분포를 얻기 위한 정규화 상수(normalizing constant)이다.

$$Z_\lambda(x) = \sum_y \exp\left(\sum_i \lambda_i f_i(x,y)\right)$$

문장 분할 확률 모델을 선형 지수 계열의 확률 분포 모델로 가정한 것은 지수 계열의 확률 모델의 집합 Q와 생성하고자 하는 분포 집합의 교집합이 존재하며, 교집합은 최대 엔트로피 확률 분포를 포함하고 더욱 좋은 성질은 이것이 유일하다는 사실[15]을 이용할 수 있기 때문이다. 최대 유사도를 갖는 확률 모델을 구하는 것이 최대 엔트로피를 가지는 확률 모델을 구하는 것과 같은 것이라는 것이 최대 유사도 원리이다. 확률 모델 p에 의해 예측되는 경험적 확률 분포 $\tilde{p}$의 로그 유사도(log likelihood)를 표현해야 하며, 로그 유사도 $L_{\tilde{p}}(p)$는 다음과 같다.

$$L_{\tilde{p}}(p) \equiv \log \Pi_{x,y} p(y|x)^{\tilde{p}(x,y)} = \sum_{x,y} \tilde{p}(x,y)\log p(y|x)$$

최대 유사도의 원리에 의해 제한 조건을 만족하는 선형 지수 계열의 확률 분포를 생성하게 되며 이것이 본 논문에서 생성하려는 문장 분할 확률 모델이다.

3. 어휘 문맥과 후보 분할 위치 선정

이 장에서는 2장에서 설명한 확률 모델에 포함되는 특성에 대하여 설명한다. 후보 분할 위치는 단어가 문장 내에서 가지는 어휘 문맥(lexical context)에 의해 결정된다. 사람이 분할 위치를 표시한 말뭉치로부터 각 단어의 어휘 문맥을 구성하여 이를 모아 학습 데이타를 구축하고 데이타의 학습을 통해 "분할 가능 위치(*SegmentablePosition*)"라는 개념(concept)을 학습한다. 이러한 개념 학습을 통해 후보 분할 위치를 선정하는 규칙을 획득하고 여기에 이용되는 문맥 정보가 확률 모델에 포함되는 특성이 된다.

3.1 후보 분할 위치

문장은 단어, 구, 절 등이 잘 정의된 문법에 의해 결합되어 만들어진다고 볼 수 있으므로 하나의 문장은 이들 구성 요소(constituent)에 대응하는 보다 짧은 세그먼트(segment)로 분할될 수 있다. 여기서 세그먼트는 자연언어를 기술하는데 일반적으로 이용되는 문맥 자유 문법의 비단말 기호에 대응한다. 특정한 세그먼트의 시작 위치 또는 두 세그먼트의 경계가 될 수 있는 단어의 위치가 후보 분할 위치이며 이것은 또한 구성 성분의 경계이며 구나 절의 시작 위치이다.

3.2 안전한 분할

문장 분할에 의해 구문 분석의 복잡도를 줄일 수 있지만 잘못된 문장 분할은 잘못된 분석이나 분석의 실패를 유발할 수 있다. 일반적으로 연관된 단어의 블록으로 문장을 분할하는 분할을 안전한 분할이라 하는데 본 논문에서는 안전한 세그먼트를 생성하는 분할을 안전한 분할로 정의한다. 3.1절에서 설명한 바와 같이 세그먼트의 개념은 문장을 이루는 구나 절과 동등하다. 그래서 세그먼트 $s_{k,l}$은 특정 구문 범주 기호 N^p가 그것을 직접 또는 간접적으로 지배하고(directly or indirectly dominate) 주어진 문법에 의해 이웃한 세그먼트와 결합될 수 있을 때 안전한 세그먼트라고 한다.

> **안전한 세그먼트**
> 세그먼트 $s_{k,l}$은 다음과 같은 조건을 만족할 때 안전하다.
> (1) $N^p \Rightarrow w_k \ldots w_l$, where $N^p \in N$, $k \le l$
> (2) $N^q \Rightarrow s_{i,j} s_{k,l}$, where $N^q \in N$, $j = k-1$, $1 \le i \le j$
> 기호 '$\Rightarrow$'는 직접 또는 간접 지배 관계를 나타낸다.

그림 1은 문장 "The students who study hard will pass the exam"에 대한 불안전한 분할과 안전한 분할

1) F는 모든 후보 특성의 집합을 나타낸다.

을 보여준다. 그림 2에서 (a)는 두 번째 세그먼트가 하나의 구문 범주로 분석될 수 없고 결과적으로 완전한 분석을 할 수 없기 때문에 불안전한 분할이다. 반면, (b)에서는 첫 번째 세그먼트는 명사구(NP)에 대응하고 두 번째 세그먼트는 동사구(VP)에 대응한다. 그러므로 (b)는 올바른 분석 결과를 얻을 수 있는 안전한 분할이다.

(a) The students | who study hard will pass the exam

(b) The students who study hard | will pass the exam

그림 1 영한 번역에서 안전한 분할과 불안전한 분할의 예

3.3 어휘 문맥(lexical contexts)

본 논문에서는 4-단어 크기의 윈도우 내에서의 어휘 문맥을 정의하며 다음의 정보를 포함한다: 단어, 단어의 분할 가능 여부, 단어의 왼쪽 2 단어와 오른쪽 2 단어. 또한 각 단어의 품사, 단어의 왼쪽 2개 단어의 하위 범주화(subcategorization) 정보[2]. 어휘 문맥은 모두 12개의 속성(attribute)들로 구성되며 그림 2는 그 구조를 보여준다.

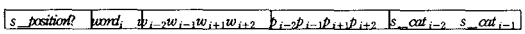

그림 2 어휘 문맥의 구조

어휘 문맥에는 두 가지 종류가 있다. 사람이 분할 위치를 표시한 말뭉치에서 어휘 문맥을 생성하는데, 분할 위치로 표시된 단어에 대해서는 활성 어휘 문맥(active lexical context)을, 그 외의 단어들에 대해서는 비활성 어휘 문맥(inactive lexical context)을 생성한다. 말뭉치에 나타나는 데이타와 이 데이타로부터 생성되는 활성, 비활성 문맥의 예는 그림 3과 같다.

Of course his parents became terribly worried **#when** they saw what was happening to Atzel.
어휘 문맥:
(**1 when** terribly worried they saw ADV VERB PRON VERB 0 1)
(***0 they*** *worried when saw what* VERB CONJ VERB PRON 1 0)

그림 3 말뭉치 데이타와 활성/비활성 어휘 문맥의 예[3]

3.4 후보 분할 위치 선정 규칙의 생성

후보 분할 위치를 선정하기 위한 규칙은 어휘 문맥을 구성하는 속성들의 논리곱(conjunction)으로 정의한다. 규칙을 자동적으로 얻기 위해 본 논문에서는 버전 공간 학습(version space learning)을 이용한다. 버전 공간 학습 방법은 속성의 논리곱 형태로 구성되는 제한된 규칙 공간 내에서 학습 데이타를 하나씩 검사하면서 점근적으로(asymptotically) 최적화 된 해를 찾을 수 있는 학습 방법이다[16]. 활성, 비활성 문맥으로 구성되는 학습 데이타를 이용하여 "분할 가능 위치"의 목표 개념 학습을 통해 후보 분할 위치를 규정할 수 있는 속성 값의 논리곱으로 기술되는 가설 h를 찾게 되며 이것으로부터 규칙을 생성한다. 규칙은 버전 공간 학습을 통해 얻어지며 각각의 규칙은 가장 일반적인 가설과 가장 특수한 가설로 경계가 지어지는 버전 공간이 된다. 규칙 생성 알고리즘의 설명을 위해 충돌 관계와 결합 연산을 소개한다.

충돌 관계(conflict relation)[4]
가설 $h_1 = (a_1,\ldots,a_n)$와 $h_2 = (b_1,\ldots,b_n)$에 대해서 다음의 식 모두를 만족하지 않으면 두 개의 가설 h_1와 h_2는 충돌 관계이다.

(1) $a_i = b_i$

(2) $a_i \neq b_i$ and ($a_i =$ '*' or $b_i =$ '*')

결합 연산(conjoin operation) $\oplus$
$(a_1,\ldots,a_n)$, $(b_1,\ldots,b_n)$, $(c_1,\ldots,c_n)$ 등을 분할 가능 어휘 문맥이라 할 때, 결합 연산은 두 개의 어휘 문맥을 취해서 하나의 어휘 문맥을 생성한다.

$$(a_1,\ldots,a_n) \oplus (b_1,\ldots,b_n) = (c_1,\ldots,c_n)$$

$$c_i = \begin{cases} '*' & \text{if } a_i \neq b_i \\ a_i & \text{if } a_i = b_i \end{cases}$$

적어도 두 개 이상의 분할 가능 위치에서 같은 값을 가지는 속성은 단어가 분할 위치가 되는데 영향을 준다고 가정한다. 따라서 결합 연산의 결과로 생성되는 어휘 문맥의 속성 중 무관 값이 아닌 속성들은 후보 분할 위치 선정에 영향을 미치는 속성으로 간주한다.

규칙 생성을 위한 개념 학습의 과정은 규칙 기술(rule description), 규칙 감축(rule reduction), 그리고 규칙 복구(rule recovery)의 세 단계로 구성된다. "규칙 기술" 단계에서는 활성 문맥 데이타를 관찰하여 그것들을 후보 분할 위치로 선정할 수 있는 규칙을 기술하고, "규칙 감축" 단계에서는 비활성 문맥 데이타를 이용하여 이런 문맥을 가지는 단어를 후보 분할 위치로 선정하는 규칙을 제거한다. 그리고 "규칙 복구" 단계에서 다시 활성 문맥 데이타를 관찰하여 전 단계에서 제거된 규칙 중 활성 어휘 문맥을 가지는 단어를 후보 분할 위치로 선정하는데 필요한 규칙들을 복구한다. 그림 4는 규칙 기술 알고리즘을 보여준다. 규칙 기술 알고리즘은 각 단어의 활성 어휘 문맥을 입력으로 하여 모든 입력을 후보 분할 위치로 선정하는 규칙의 집합을 기술한다. 규칙은 버전 그래프에서 특수화 경계 s 노드와 일반화 경계 g 노드간의 경로로서 정의된다.

2) 단어가 절을 목적어로 취할 수 있는 동사임을 표시하는 이진 값(binary value).

3) 그림에서 기호 '#'은 분할 위치를 나타낸다.

4) 여기서 '*' 은 무관 값(don't care value)을 의미한다.

입력: 단어 w_i의 활성 어휘 문맥의 집합 D_{w_i}.
출력: 한 쌍의 일반화 경계 (g)와 특수화 경계 (s)로 구성되는 규칙의 집합 R_{w_i}.
1. $R_{w_i} = 0$
2. 집합 D_{w_i}에 속하는 모든 lc_i와 lc_j ($i \neq j$)에 대해, 다음을 수행한다.
   (1) $new\_lc = lc_i \oplus lc_j$
   (2) R_{w_i}의 모든 원소 r_k에 대해, 다음을 수행한다.
      a. new_lc와 $r_k g$간의 관계를 결정한다.
      b. if ($new\_lc <_g r_k g$) then
         if ($r_k s == NULL$) then $r_k s \leftarrow new\_lc$
         else new_lc를 버전 그래프에서 일반화 관계를 만족시키는 위치에 삽입
      c. if ($new\_lc >_g r_k g$) then $r_k g \leftarrow new\_lc$
   (3) if (new_lc is *conflict* with all $r_k \in R_{w_i}$) then
      - 새로운 규칙 r'을 다음과 같이 기술한다: r' : $g \leftarrow new\_lc$
      - $R_{w_i} \leftarrow R_{w_i} \cup r'$

그림 4 규칙 기술 알고리즘[5]

유추 가능(inferable) $\vdash$
단어 w_i의 어휘 문맥 lc_{w_i}이 그 단어에 대한 후보 분할 위치 선정 규칙 r_{w_i}에 대하여 다음을 만족하면 lc_{w_i}는 규칙 r_{w_i}에 의해 유추 가능하다고 하고 $r_{w_i} \vdash lc_{w_i}$으로 표현한다.
$\{ (r_{w_i}:n_i,\ r_{w_i}:n_m) \mid r_{w_i}:n_m \geq_g lc_{w_i} \text{ and } r_{w_i}:n_j \geq_g lc_{w_i} \} \neq \emptyset$

입력: 문장 $S = w_1, w_2, \ldots, w_n$과 규칙의 집합 $RuleSet = \{R_{w_1}, R_{w_2}, \ldots, R_{w_n}\}$.
출력: 후보 분할 위치의 집합 $SP = \{sp_1, sp_2, \ldots, sp_m\}$.
1. $SP = \emptyset$
2. S의 모든 단어 w_i에 대하여 다음을 수행한다.
   (1) 어휘 문맥 lc_{w_i}을 생성한다.
   (2) w_i에 대한 후보 분할 위치 선정 규칙 $R_{w_i} = \{r^1_{w_i}, r^2_{w_i}, \ldots, r^k_{w_i}\}$의 모든 규칙 $r^j_{w_i}$에 대하여 if ($r^j_{w_i} \vdash lc_{w_i}$) then $SP \leftarrow w_i$.

그림 5 후보 분할 위치 선정 알고리즘

3.5 후보 분할 위치 선정 방법

학습을 통해 자동적으로 구축한 규칙을 이용하여 주어진 문장에서 후보 분할 위치를 선정하기 위해 '유추 가능' 연산을 정의한다.

그림 5는 문장에서 후보 분할 위치를 선정하는 알고리즘을 보여준다. 주어진 문장의 각 단어의 어휘 문맥을 생성하고 이것이 그 단어에 대한 후보 분할 위치 선정 규칙으로부터 "유추 가능"한 어휘 문맥인지를 판단하여 후보 분할 위치의 여부를 결정한다.

4. 최대 엔트로피 문장 분할 모델의 생성

본 논문에서는 2.2절에서 설명한 최대 유사도 원리를 적용하여 문장 분할을 위한 확률 모델을 생성한다. 확률 모델의 생성 과정은 문장 분할에 고려되는 특성을 선택하는 단계(feature selection)와 선형 지수 계열 확률 모델의 각 특성에 대한 파라미터 λ_i 값을 추정하는 단계(parameter estimation)로 구성된다.

특성 선택 단계에서는 특성이 말뭉치에서 나타나는 빈도수를 기준으로 하는 빈도에 의한 특성 선택(frequency-based feature selection) 방법을 이용한다. 빈도수가 적은 특성들은 신뢰성이 떨어진다고 가정하여 무시한다. 응용 분야에 따라 가장 적절한 컷오프(cutoff) 빈도수 값이 다르며 본 논문에서는 10을 컷오프 빈도수 값으로 정하였다[6]. 분할 위치 결정에 영향을 미치는 요소들은 특성(feature)의 형태로 모델에 포함되고 특성은 이진 값을 갖는 함수(binary-valued function)이며 다음과 같이 표현된다.

$$f:\ E \rightarrow \{0,1\}$$

여기서 E: $X \times Y$(X: 문맥 상황의 집합, Y: {0, 1})이다. Y 값은 분할 위치인지 아닌지를 나타낸다. 이러한 특성이 2장에서 설명한 최대 엔트로피 모델링 과정에서 사용된다.

분할 위치 결정에 영향을 미치는 요인들로서 고려되어야 하는 것들은 다음과 같으며 처음 세 가지는 안전한 분할을 위해, 그리고 마지막 것은 구문 분석 효율 향상의 정도를 최대화 하기 위해 고려된다: 단어의 어휘 문맥, 단어의 위치 정보, 다른 분할 위치가 앞에 존재하는 지의 여부, 분할의 결과로 생기는 세그먼트의 크기. 어휘 문맥적 특성의 예는 다음과 같다. 앞 단어가 "*say*"인 경우 현재 단어가 분할 위치로 적절하다는 것을 반영한다. 이는 "*say*" 동사가 주로 절(clause)을 목적어로 갖기 때문이며 새로운 절을 시작하는 위치는 분할 위치로서 높은 선호도를 가지기 때문이다.

$$f_{lc}(x,y) = \begin{cases} 1 & \text{if } x_{word} = \text{"}that\text{"} \text{ and } x_{w_{i-1}} = \text{"}say\text{"} \text{ and } y = 1 \\ 0 & \text{otherwise} \end{cases}$$

위의 특성은 현재 단어가 "*that*"이고 앞 단어가 "*say*"인가를 확인하는 특성이다. 어휘 문맥적 특성들은 3장에서 설명한 후보 분할 위치 선정 규칙으로부터 추출된다. 3장의 규칙은 후보 분할 위치가 가지는 어휘 문맥의 속성들을 표현하며 자주 나타나는 속성들은 분할 위치 결정에 유력한 증거가 될 수 있다. 어휘 문맥적 특성들은 규칙을 구성하는 버전 그래프상의 일반화 경계 g와 특수화 경계 s간의 경로를 추적하면서 추출된다. 어휘 문맥적 특성 추출을 위한 알고리즘의 기술을 위하여 다음을 정의한다.

활성 속성(active feature)[7] a
$a[n] \neq$ '*'
활성 차이(active difference) 연산[8] $\ominus$
노드 n_i의 활성 속성 집합이 A_i이고 n_j의 활성 속성 집합이 A_j일 때, 노드 n_i, n_j간의 활성 차이는 다음과 같다.
$$n_i \ominus n_j = A_i - A_j$$
연합 조인(union join) 연산[9] $\bowtie$
활성 속성 a를 가진 $f_{(a)}$와 ($a_1, \ldots, a_n$)을 가진 특성 $f_{(a_1,\ldots,a_n)}$에 대해
$$f_{(a)} \bowtie f_{(a_1,\ldots,a_n)} = f_{(a_1,\ldots,a_n,a)}$$

5) 규칙 r_k의 일반화 경계를 $r_k : g$로, 특수화 경계를 $r_k : s$로 표현한다. 그리고 $<_g$ 연산자는 두 어휘 문맥간의 일반화 관계를 비교한다.

6) 대부분의 응용에서 컷오프 값으로 5 또는 10의 값을 이용하여 만족할만한 결과를 얻을 수 있었다.

어휘 문맥적 특성 추출을 위한 알고리즘은 그림 6과 같다. 알고리즘에서 *Combination(set_of_active_attributes)* 함수는 활성 속성 집합의 각 속성들의 조합을 활성 속성으로 가지는 특성을 생성한다.

같은 단어라도 문장에서의 위치에 따라 분할 위치로 선택될 수 있는 선호도가 다르다. 예를 들어, "CRRES *on* the launch pad is a joint NASA-Air Force satellite to study the effects *on* the micro-electronic components"에서 문장 앞에 나오는 "*on*" 보다는 뒤에 나오는 "*on*"이 분할 위치로 더 적절하다. 문장 내에서의 위치를 분할 위치 결정 과정에 반영하기 위해 단어의 위치 정보 특성(regional feature)을 도입한다. 문장을 일정한 개수의 영역(region)으로 나누고 단어가 위치하는 영역 값으로 위치 정보를 계산한다. n개의 단어로 이루어진 문장에서 i번째 단어 w_i의 영역 값은 다음 식으로 계산한다.

$$region\_value = \lceil \frac{i}{n} \times R \rceil$$

여기서 R은 문장의 영역 개수를 의미한다.[10] 위의 문장은 19 단어로 이루어지며 앞의 "on"의 영역 값은 $\lceil \frac{2}{19} \times 4 \rceil = 1$이며, 뒤의 "*on*"의 영역 값은 $\lceil \frac{16}{19} \times 4 \rceil = 4$가 된다. 지역적 특성의 예는 다음과 같다.

$$f_4(x,y) = \begin{cases} 1 & \text{if } x_{word} = \text{"}on\text{"} \text{ and } x_{region} = 4 \text{ and } y = 1 \\ 0 & \text{otherwise} \end{cases}$$

입력: 후보 분할 위치 선정 규칙, $RuleSet = \{r_1, r_2, \ldots, r_m\}$.
출력: 어휘 문맥적 특성 집합, $F_{lc} = \{f_1, f_2, \ldots, f_n\}$.
1. $F_{lc} = \emptyset$
2. 집합 *RuleSet*에 속하는 모든 r_i에 대해, 다음을 수행한다.
   (1) $F'_{lc} = \emptyset$ // 규칙 r_i에서 추출되는 특성의 집합
   (2) $F'_{lc} \leftarrow Combination(r_i$의 모든 활성 속성)
   (3) r_iS까지의 경로상의 모든 노드 n에 대해, 다음을 수행한다.
      a. n과 직접 일반화 n_g의 활성 차이 $n_{at} = n \ominus n_g$
      b. $F_{at} \leftarrow Combination(n_{at})$
      c. 모든 $f \in F_{at}$에 대해, 다음을 수행한다.
         - 모든 $f_{lc} \in F'_{lc}$에 대해 $F \leftarrow f \bowtie f_{lc}$
         - $F'_{lc} = F'_{lc} \cup F \cup F_{at}$
      d. $F_{lc} = F_{lc} \cup F'_{lc}$

그림 6 어휘 문맥적 특성 추출을 위한 알고리즘

다른 분할 위치가 앞에 존재하는지의 여부는 안전한 분할을 위해 고려된다. 영어 문장은 주부와 술부로 이루어지는데 일반적으로 주어부는 길지 않기 때문에 주어에서는 끊어 읽지 않으며, 따라서 말뭉치에서 분할 위치를 표시할 때 첫 분할 위치는 주로 술부에 위치하게 된다. 또한 문장에서 처음 나오는 분할 위치는 사람이 처음으로 문장을 분할하는 위치이므로 다른 분할 위치에 비해 상대적으로 안전하다고 볼 수 있다. 이러한 이유로 문장에서 첫 분할 위치인지의 여부는 안전한 분할을 위해 중요한 요인이 된다. 특성의 예는 다음과 같다.

$$f_{1st}(x,y) = \begin{cases} 1 & \text{if } x_{word} = \text{"}that\text{"} \text{ and } x_{segmentable\_idx} = 1 \text{ and } y = 1 \\ 0 & \text{otherwise} \end{cases}$$

위의 특성은 현재 단어가 "that"이고 첫 후보 분할 위치인가를 확인하는 특성이다.

파라미터 추정 단계에서는 선형 지수 계열의 확률 모델에 포함되는 모든 특성의 가중치 값을 결정한다. 본 논문에서는 [9]에서 제안한 개선된 Iterative Scaling 알고리즘을 이용하여 (3)에 속하는 확률 모델의 가중치 λ_i의 값을 계산한다. 알고리즘은 파라미터 $\lambda_1, \ldots, \lambda_n$의 값을 점진적으로 조정하고 동시에 확률 모델의 로그-유사도 값을 증가시키면서 특정한 조건을 만족시킬 때까지 파라미터 값의 조정을 계속한다. 본 논문에서는 모든 가중치 λ_i와 λ_{i+1}의 값을 소수점 아래 5자리까지 비교하여 차이가 없을 때까지 알고리즘을 실행한다. 다른 종료 조건을 이용하는 것으로는 일정한 회수의 반복을 수행한 후에 알고리즘을 종료시키는 방법, 로그-유사도나 정확도의 값의 변화가 무시할 수 있을 만큼 적을 때 종료하는 방법 등이 있다. 알고리즘의 종료 조건은 계산 시간과 확률 모델의 로그-유사도 값에 영향을 미친다.

5. 분할 위치 결정

각각의 후보 분할 위치는 서로 다른 특성을 가지며 각 후보 분할 위치가 가지는 특성에 따라 분할 확률 값이 다르게 된다. 후보 분할 위치의 분할 확률을 계산하는 과정을 다음의 예를 통해서 설명한다. 문장 "But the chemical company said it is progress on cost controls and be coming more efficient."에서 후보 분할 위치 "*it*"의 문맥 상황을 x_{it}이라 할 때 분할 확률은 다음과 같은 과정을 통해 결정된다. 첫째, 단어 "*it*"가 가지는 특성을 결정한다. "*it*"는 두 번째 영역에 속하므로 f_2의 지역적 특성, 첫 번째 후보 분할 위치를 표현하는 f_{1st} 특성, 그리고 다음과 같은 세 개의 어휘 문맥적 특성을 가진다.

7) 여기서 *a[n]*은 규칙을 정의하는 경로상의 노드 *n*의 속성 *a*의 값이다. 즉, 활성 속성은 노드의 속성 값이 무관 값이 아닌 속성이다.
8) 규칙을 정의하는 경로에 있는 두 개 노드간의 활성 차이는 각 노드가 가진 활성 속성 집합의 차집합이다.
9) 연합 조인 연산은 두 개의 특성을 취하여 이들이 가진 활성 속성의 합집합을 활성 속성으로 가지는 새로운 특성을 생성한다.
10) 본 논문에서는 문장을 4개의 영역으로 구분하고 각 영역을 위한 f_1, f_2, f_3, f_4의 4개의 지역적 특성을 이용한다.

$$f_{lc_1}(x_{it}, y) = \begin{cases} 1 & \text{if } x_{word} = \text{“that”} \text{ and } x_{i-1} = \text{“say”} \text{ and } y = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$f_{lc_2}(x_{it}, y) = \begin{cases} 1 & \text{if } x_{word} = \text{“ ”} \text{ and } x_{pos_{i+1}} = \text{VERB and } y = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$f_{lc_3}(x_{it}, y) = \begin{cases} 1 & \text{if } x_{word} = \text{“it”} \text{ and } x_{w_{i-1}} = \text{“say”} \text{ and } x_{pos_{i+1}} = \text{VERB and } y = 1 \\ 0 & \text{otherwise} \end{cases}$$

둘째, 다음의 식으로 분할 확률 값을 계산한다.

$$p(1|x_{it}) = \frac{1}{Z(x_{it})} \exp(\lambda_{lc_1} + \lambda_{lc_2} + \lambda_{lc_3} + \lambda_{f_2} + \lambda_{f_{1st}})$$

분할 위치 결정 알고리즘은 그림 7과 같다. 알고리즘에서 집합 A는 분할 확률이 각 단어의 임계값 ε_{w_i}보다 큰 후보 분할 위치의 집합이고 B는 그 이외의 후보 분할 위치의 집합이다. 임계값은 다음의 식으로 계산한다.

$$\varepsilon_{w_i} = \frac{w_i\text{가 분할 위치로 표시된 빈도수}}{\text{말뭉치에서의 } w_i\text{의 수}}$$

이것은 특정 단어가 분할 위치가 될 기대값을 의미하며 이 값보다 큰 확률을 가지면 그 단어가 분할 위치로서 적절하다고 간주된다. 모든 후보 분할 위치의 분할 확률이 임계값보다 작은 경우에는 분할의 결과로 생기는 세그먼트의 크기를 함께 고려한다.[11] 이 두 가지 값의 합으로 분할 위치의 점수가 결정되고 가장 큰 점수를 가지는 위치를 분할 위치로 결정한다.

입력: 후보 분할 위치의 집합 $SP\_Set = \{sp_1, sp_2, \ldots, sp_m\}$, 문장의 단어 수 n
출력: 최적의 분할 위치 sp_*.
1. $A = \emptyset$, $B = \emptyset$
2. 모든 $sp_i \in SP\_Set$ 에 대해, 다음을 수행한다.
   (1) 분할 확률 $p(1|sp_i)$를 계산한다.
   (2) $p(1|sp_i) > \varepsilon_{w_i}$이면 $A \leftarrow A \cup \{sp_i\}$
   아니면 $B \leftarrow B \cup \{sp_i\}$
3. $A \neq \emptyset$이면 $sp_* = \arg\max_{sp_i \in A} p(1|sp_i)$
4. $A = \emptyset$이면, 모든 $sp_i \in B$에 대해 다음을 수행한다.
   (1) $size_1 = sp_i.position$, $size_2 = n - sp_i.position$
   (2) $seg\_size_{sp_i} = \max(size1, size2)$
   (3) $score_{sp_i} = p(1|sp_i) + \frac{1}{seg\_size_{sp_i}}$
5. $sp_* = \arg\max_{sp_i \in B} score_{sp_i}$

그림 7 분할 위치 결정 알고리즘.

일정한 길이 이상의 문장은 분석이 용이한 길이의 세그먼트로 분할되어야 하므로 일정한 길이 이상의 세그먼트가 존재하지 않을 때까지 문장 분할을 계속한다.[12]

6. 실험 결과

6.1 말뭉치

말뭉치는 크게 학습 데이타(training data)와 테스트 데이타(test data)로 나뉜다. 학습 데이타는 후보 분할 위치를 분류하는 규칙을 생성하고 최대 엔트로피 확률 모델을 위한 통계를 얻기 위해 사용된다. Wall Street Journal의 문장에서 쉼표(comma)를 포함하지 않는 길이 15 이상의 문장 3,000개를 추출하여 사람이 분할 위치를 표시하고 이로부터 어휘 문맥을 생성하여 학습 데이타를 구축하였다. 테스트 데이타는 4개의 영역에서 추출된 길이 15 이상의 쉼표가 없는 문장 각각 300개씩을 추출하여 구성하였다: Wall Street Journal, 고등학교 영어 교과서, Byte Magazine, Washington Post의 정치 분야. 분할 성능이 학습 데이타와 같은 영역과 다른 영역에서 어떻게 달라지는 가를 확인하기 위하여 여러 분야에서 테스트 데이타를 추출하였다. 또한 문장 길이에 따라 분할 성능이 다르게 나타나고 문장 분할 방법이 어느 정도 길이까지의 문장에 유용하게 적용될 수 있는지를 판단하기 위하여 문장 길이를 기준으로 테스트 데이타를 기준으로 분류하였으며 결과는 표 1과 같다.

표 1 테스트 데이타의 문장 길이 분포

문장 길이	문장 수
15~19	400
20~24	400
25~29	280
30~	120

6.2 후보 분할 위치 선정 규칙과 분할 확률 모델 생성

그림 8은 본 논문에서 제안한 문장 분할을 위한 학습과 확률 모델 생성 과정을 보여준다. Wall Street Journal에서 추출한 3,000 문장에서 5,375개의 활성 어휘 문맥과 40,236개의 비활성 어휘 문맥을 생성하였다. “분할 가능 위치” 개념 학습 결과 모두 9,002개의 노드를 가지는 360개의 버전 그래프가 생성되었고 이로부터 5,851개의 후보 분할 위치 선정 규칙을 생성하였다. 생

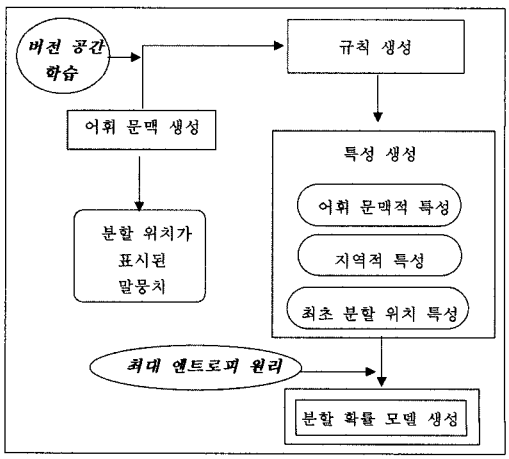

그림 8 문장 분할을 위한 학습과 확률 모델 생성 과정

11) 세그먼트의 크기 값은 0~1의 값으로 정규화한다.
12) 본 논문에서는 문장을 12 단어 이하의 세그먼트들로 분할한다.

성된 규칙으로부터 6,596개의 후보 특성을 생성하여 확률 모델 생성에 이용하였다. 2.3절에서 설명한 빈도수 기반 특성 선택 방법으로 12분 동안에 2,866개의 특성을 포함하는 확률 모델을 생성하였다.

6.3 분할 성능(segmentation performance)

분할 성능은 학습에 의해 자동적으로 생성한 후보 분할 위치 선정 규칙의 적용률(coverage)과 최대 엔트로피 확률 모델에 의한 분할 위치 결정의 정확도(accuracy), 그리고 이 두 가지를 결합한 구문 분석 입장에서의 성능 척도인 분할 기여도(segmentation contribution) 값으로 측정한다. 적용률은 분할 대상이 되는 일정한 길이 이상의 문장 수와 후보 분할 위치가 발견되어 실제로 분할된 문장 수의 비로 정의된다. 그리고 정확도는 분할 확률에 의해 문장을 안전하게 분할하여 분석의 실패를 유발하지 않도록 분할하는 정도를 표현하는 척도이며 실제로 문장 분할이 이루어진 문장 수와 안전하게 분할되어 분할이 분석 실패를 유발하지 않은 문장 수의 비로 정의된다.

$$coverage = \frac{\text{실제로 분할된 문장 수}}{\text{문장 분할 대상의 문장 수}}$$

$$accuracy = \frac{\text{안전하게 분할된 문장 수}}{\text{실제로 분할된 문장 수}}$$

문장 분할의 목적은 구문 분석의 복잡도를 줄여 효율적인 분석을 가능하게 하는 것이다. 따라서 구문 분석의 입장에서 문장 분할을 통해 얼마나 효과를 얻을 수 있는가를 나타내는 척도로서 분할 기여도 값 *SC_value*를 도입한다. 이는 문장 분할 대상이 되는 문장 중에서 안전하게 분할되어 올바르게 분석된 문장의 비율을 표현하며 다음과 같이 계산할 수 있다.

$$SC\_value = \frac{\text{안전하게 분할된 문장 수}}{\text{문장 분할 대상의 문장 수}} = coverage \times accuracy$$

분할 성능을 비교하기 위해 아무런 문맥 정보도 이용하지 않고 경험적인 방법으로 분할 위치를 결정하는 방법을 기준 방법(baseline method)으로 이용하였다. 즉 학습 데이타에서 5번 이상 분할 위치로 표시된 단어를 후보 분할 위치로 선정하고 다음의 분할 확률 $p(w_i)$에 의해 분할 위치를 결정한다.

$$p(w_i) = \frac{w_i\text{가 분할 위치로 표시된 빈도수}}{\text{학습 데이타에서의 } w_i\text{의 개수}}$$

다른 비교 대상으로서 사람이 만든 분할 규칙에 의하여 분할하는 규칙 기반 방법 [17]을 이용한다. 이 방법은 긴 문장 분석에 이용되는 문맥 자유 문법의 관찰을 통해 후보 분할 위치를 규정하는 규칙을 사람이 구축한다. 분할 위치 결정을 위해 후보 분할 위치는 유형별로 분류되고 유형마다 분할 우선 순위가 할당된다. 분할 위치는 후보 분할 위치의 분할 우선 순위와 분할로 생성되는 세그먼트의 크기를 고려하여 결정된다.

표 2는 각각의 문장 분할 방법의 분할 성능을 비교한 것이다. 여기서 FFS(frequency-based feature selection)는 빈도수를 이용하여 선택된 특성을 가지고 분할 확률 모델을 생성하여 이를 이용하여 분할 위치를 결정하는 방법을 나타낸다.

표 2 문장 분할 방법의 분할 성능 비교

	적용률 (%)	정확도 (%)	SC_value
기준 방법	100	77.6	0.776
규칙 기반 방법	85.2	86.5	0.737
FFS	98.3	88.1	0.866

표 2에서 FFS와 기분 방법을 비교해보면 분할의 정확도가 문맥 정보에 의존한다는 것을 보여준다. 문맥 정보를 이용하는 제안된 방법은 비록 적용률이 약간 낮지만 정확도가 더 높으며 두 방법의 분할 기여도로 판단하면 구문 분석의 입장에서 보다 유용하다고 할 수 있다. 규칙 기반 방법은 약 86%의 정확도를 보여 제안된 방법에 비해 약간 정확도가 낮지만 적용률이 많이 떨어지기지 때문에 분석에 대한 기여도가 세 가지 방법 중 가장 낮게 나타났다. 규칙에 의해 적용률을 높이는 것은 많은 사람의 노력을 필요로 하기 때문에 이를 상당히 높이는 것은 매우 어려운 일이다.

표 3은 FFS 방법의 각 테스트 영역에서의 적용률, 정확도, 분할 기여도를 문장 길이별로 보여준다. 0.866의 *SC_value* 평균 값은 15 단어 이상의 약 87%의 문장

표 3 문장 길이에 따른 분할 성능

테스트 영역	문장 길이	적용률 (%)	정확도 (%)	SC_value
Wall Street Journal	15~19	100	94	0.94
	20~24	100	92	0.92
	25~29	98.6	88.4	0.87
	30~	96.7	75.9	0.73
고등학교 영어 교과서	15~19	99	92.9	0.92
	20~24	100	91	0.91
	25~29	97.1	88.2	0.86
	30~	93.3	75	0.7
Byte Magazine	15~19	100	90	0.9
	20~24	98	88.8	0.87
	25~29	97.1	85.3	0.83
	30 ~	96.7	72.4	0.7
Washington Post	15~19	99	90.9	0.9
	20~24	98	86.7	0.85
	25~29	95.7	83.6	0.8
	30 ~	93.3	71.4	0.67
전 체	1200 문장	98.3	88.1	0.866

이 제안된 방법의 문장 분할을 통해 낮은 분석 복잡도를 가지고 올바른 구문 구조를 생성할 수 있다는 것을 의미한다. 또한 테스트 데이타 영역에 따라 약간 차이가 있지만 제안된 문장 분할 방법이 영역과 무관하게 적용될 수 있다는 것을 보여준다.

6.4 분석 효율

구문 분석의 효율은 일반적으로 분석 시간과 분석에 사용된 메모리 양으로 측정한다. 문장 분할을 하지 않았을 경우 20 단어 이상의 문장은 많은 경우에 분석이 종료하지 않는다. 따라서 본 논문에서는 문장 길이가 15에서 20 사이의 문장에 대해서만 문장 분할을 이용한 분석과 그렇지 않은 분석의 분석 효율을 비교한다. 분석 효율 향상은 다음과 같이 정의되는 시간과 공간 면에서의 효율 향상(efficiency improvement: EI)을 측정한다.

$$EI_{time} = \frac{t_{unseg}}{t_{seg}}, \quad EI_{memory} = \frac{m_{unseg}}{m_{seg}}$$

여기서 EI_{time}은 시간 면에서의 효율 향상을, EI_{memory}는 메모리 면에서의 효율 향상을 표현한다. 그리고 t_{unseg}와 m_{unseg}는 분할을 하지 않은 분석에서의 분석 시간과 사용된 메모리 양을 의미하고 t_{seg}와 m_{seg}는 분할을 이용한 분석에서의 분석 시간과 메모리 사용량이다.

구문 분석은 Ultra-Sparc 30 시스템에서 수행되었다. 표 4는 문장 분할을 이용한 구문 분석의 성능 향상을 보여준다. 표에 나타나는 수치는 한 문장을 분석할 때 소요되는 시간과 메모리의 평균값이다. 분할을 이용하지 않은 분석의 경우 Wall Street Journal의 73 문장, 고등학교 영어 교과서의 85 문장, Byte Magazine의 79 문장, Washington Post의 68 문장에 대해서만 분석 결과를 생성하였으며 분할을 이용한 분석과의 비교는 분석이 완료된 문장만을 대상으로 하였다.

표 4 문장 분할을 이용한 분석 효율 향상[13)]

		분할을 이용한 분석	분할을 이용하지 않은 분석	효율 향상
Wall Street Journal	시간 (sec)	4.8	22.1	4.6
	공간 (MB)	1.1	3.9	3.5
고등학교 영어 교과서	시간 (sec)	4.6	19.6	4.3
	공간 (MB)	0.9	3.4	3.8
Byte Magazine	시간 (sec)	5.4	25.1	4.6
	공간 (MB)	1.1	3.7	3.4
Washington Post	시간 (sec)	5.1	29	5.7
	공간 (MB)	1.1	4.3	3.9

13) "분할을 이용한 분석"에서 제시된 수치는 문장 분할에 소요된 시간, 메모리 사용량이 포함된 것임.

문장 분할을 하지 않는 경우에는 20 단어 이상의 문장들은 분석이 어려웠으나 문장 분할을 통해 20 단어 이상의 문장도 분석이 용이해진다. 표 5는 20 단어 이상의 문장들을 분할을 이용하여 분석할 때의 분석 시간과 사용된 메모리 양을 보여준다. 표 5를 보면 문장이 길어질수록 분할을 이용하더라도 분석에 상당히 많은 시간이 소요됨을 알 수 있는데, 이를 해결하기 위해서는 분할 위치의 특성과 생성된 세그먼트간의 병렬성을 이용하는 효율적인 구문 분석 알고리즘에 대한 연구가 필요하다.

6.5 다른 분할 방법과의 비교

이 절에서는 본 논문에서 제안한 문장 분할 방법을 다른 문장 분할 방법과 분할의 적용률과 분석 효율 향상의 정도 면에서 비교한 결과를 설명한다. [6]의 영어-일본어 기계 번역에서 분석 효율 향상을 목적으로 한 "성분 경계 파싱"은 약 350개의 패턴을 이용하여 주로 10 단어 이내의 짧은 문장의 분석을 수행한다. 따라서 본 논문에서 대상으로 하는 15 단어 이상의 긴 문장에 대해서도 올바른 분석 결과를 얻으면서 효율적인 분석을 할 수 있기 위해서는 보다 많은 패턴이 필요하고 이는 많은 사람의 노력을 요구한다. 따라서 실용적 기계 번역을 위해 유용하다고 볼 수 없다. [4,5]에서 제안한 하나의 문장을 세 부분으로 분할하는 방법은 평서문과 명령문도 주어가 없는 평서문으로 간주한다면 적용 대상이 될 수 있다. 이 방법은 비교적 적은 학습 데이타(300 문장)를 가지고 주어의 중심어와 경계를 찾는데 약 97%의 성공률을 보인다. 이 방법은 효율적인 분석이 가능하지만 하나의 주어와 하나의 서술부로만 이루어진 단문(simple sentences)에만 적용할 수 있다. 그러나 대부분의 긴 문장을 이루는 대등 접속문(coordinate sentence)이나 복합문(complex sentence)에는 적용하기 어렵기 때문에 긴 문장 분석을 위한 일반적인 방법으로는 적절하지 않다고 판단된다. [7]에서 제안한 문장 패턴을 이용한 문장 분할 방법은 시간 면에서 1.4배, 공간 면에서 2.4배의 분석 효율 향상을 얻을 수 있었다. 이것은 본 논문에서 제시한 결과에 비해 떨어지며 문장 패턴의 적용률이 36.2%에 불과하여 긴 문장 분석을 위한 일반적인 방법으로는 적절하지 않다.

본 논문에서 제안한 문장 분할 방법은 시간 면에서 약 4.8배, 공간 면에서 약 3.6배의 분석 효율 향상을 이루었으며 단문 뿐만 아니라 대등 접속문이나 복합문에도 적용할 수 있다. 즉, 문장의 종류나 길이에 무관하게 적용할 수 있으며 약 98%의 적용률과 약 88%의 분할 정확도를 보인다. 이는 제안된 문장 분할 방법이 실용적인 영어-한국어 기계 번역 시스템을 위한 분석의 전처리 단계로서 유용함을 의미한다. 또한 학습과 통계를 이

용한 방법은 학습 데이타를 구축하는 것 이외에 다른 사람의 노력을 요구하지 않기 때문에 적용률과 정확도의 향상을 위한 확장이 용이하다.

표 5 20 단어 이상의 문장을 분할을 이용하여 분석하였을 때의 분석 효율

영 역	문장 길이	평균 시간 (sec)	평균 메모리 (MB)
Wall Street Journal	20~24	9.2	2
	25~29	13.5	3.1
	30~	29.5	7
고등학교 영어 교과서	20~24	7.8	1.9
	25~29	13.7	2.9
	30~	28	6.5
Byte Magazine	20~24	10.2	2
	25~29	15.6	3
	30~	30.2	7.1
Washington Post	20~24	10.6	2.1
	25~29	17	3.3
	30~	29.5	6.9

7. 결 론

본 논문에서는 긴 문장의 효율적인 분석을 위한 문장 분할 방법 제시한다. 분할 위치가 표시된 말뭉치로부터 얻은 어휘 문맥 정보를 이용하여 후보 분할 위치를 선정하는 규칙을 버전 공간 학습을 통해 자동적으로 획득하고 어휘 문맥 정보의 통계 정보를 이용하여 생성한 최대 엔트로피 확률 모델을 이용하여 가장 적절한 분할 위치를 결정한다. 말뭉치를 이용한 학습 모델은 성능의 향상을 위한 확장을 용이하게 하고 사람의 노력을 줄일 수 있게 한다.

제안된 방법의 분할 성능은 약 98%의 적용률과 88%의 정확도, 그리고 0.87의 기여도 값을 보였다. 또한 분석의 시간 면에서 약 4.8배, 공간 면에서 약 3.6배의 효율 향상에 기여함을 알 수 있었다. 따라서 제시된 실험 결과는 제안된 문장 분할 방법이 긴 문장의 분석에 유용하게 이용될 수 있으며 실용적인 기계 번역 시스템에 적용할 수 있음을 보여준다.

분할 성능의 향상과 보다 효율적인 구문 분석을 위한 과제는 다음과 같이 정리할 수 있다. 첫째, 분할 정확도의 향상을 위해 분할 위치 결정에 고려되는 다른 유형의 특성에 대한 연구와 불안전한 분할을 구문 분석 이전에 인식하여 복구하는 알고리즘에 대한 연구가 진행되어야 할 것이다. 둘째, 문장 분할의 결과 생성된 세그먼트들의 특성을 이용할 수 있는 분석 알고리즘에 대한 연구는 보다 효율적이고 정확한 구문 분석을 가능하게 할 것이다. 문장을 구성하는 세그먼트들은 결합되어 분석되어야 하지만 독립적으로 분석될 수도 있다. 따라서 보다 빠른 구문 분석을 위해서는 이러한 세그먼트의 특성을 파악하고 이용하는 분석 알고리즘에 대한 연구가 필요하다. 셋째, 대부분의 영어 문장, 특히 분할 대상이 되는 15 단어 이상의 문장은 쉼표 등의 무장 부호가 많이 사용되는데, 이 문장 부호는 문장 분할의 주요 단서가 된다. 따라서 본 논문의 결과와 문장 부호 분석을 이용하는 방법을 결합하여 문장 분석의 정확성과 효율성을 향상시키는 연구 또한 의미 있는 과제가 될 것이다.

참 고 문 헌

[1] J. Lafferty, D. Beeferman, and A. Berger, "Text Segmentation using Exponential Models," In *Second Conference on Empirical Methods in Natural Language Processing*, 1997, Providence, RI.

[2] David D. Palmer and Marti A. Hearst, "Adaptive Multilingual Sentence Boundary Disambiguation," Computational Linguistics, Vol. 23, No. 2, pp. 241-265, 1997.

[3] J. C. Reynar and A. Ratnaparkhi. "A Maximum Entropy Approach to Identifying Sentence Boundaries," In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pp. 16-19, 1997, Washington D.C.

[4] C. Lyon and B. Dickerson, "Reducing the Complexity of Parsing by a Method of Decomposition," In *International Workshop on Parsing Technology*, Sept., 1997.

[5] C. Lyon and R. Frank, "Neural Network Design for a Natural Language Parser," In *International Conference on Artificial Neural Networks*, 1995.

[6] Osamu Furuse and Hitoshi Iida, "Constituent Boundary Parsing for Example-Based machine Translation," In *Proceedings of 1994 Conference on Computational Linguistics*, pp. 105-111, 1994, Kyoto, Japan.

[7] S. D. Kim and Y. T. Kim, "Sentence Analysis using Pattern Matching in English-Korean Machine Translation," In *Proceedings of the 1995 ICCPOL*, pp. 25-28, 1995.

[8] E. T. Jaynes, "Information Theory and Statistical Mechanics," *Physical Review*, Vol. 106, pp. 620-630, 1957.

[9] Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Pietra, "A Maximum Entropy Approach to Natural Language Processing," *Computational Linguistics*, Vol. 22, No. 1, pp. 39-72, 1996.

[10] A. Ratnaparkhi, "A Maximum Entropy Part of Speech Tagger," In E. Brill and K. Church, editors, *Conference on Empirical Methods in Natural Language Processing*, 1996, University of

Pennsylvania.
[11] Eric S. Ristad, "Maximum Entropy Modeling for Natural Language," 1997, Madrid.
[12] F. Jelinek and R. L. Mercer, "Interpolated Estimation of Markov Source Parameters from Sparse Data," In *Workshop on Pattern Recognition in Practice*, 1980, Amsterdam, The Netherlands.
[13] S. M. Katz, "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 35, 1987.
[14] Sheldon M. Ross, "Introduction to Probability Models," Academic Press, 1997.
[15] A. Ratnaparkhi, "A Simple Introduction to Maximum Entropy Models for Natural Language Processing," Technical report, Institute for Research in Cognitive Science, University of Pennsylvania, 1994, IRCS Report 97-08.
[16] Tom M. Mitchell, "Machine Learning," The McGraw-Hill Companies, Inc., 1997.
[17] 김성동, 김영택. "효율적인 영어 구문 분석을 위한 문장 분할", 한국 정보과학회 논문지, Vol. 24, No. 8, pp. 884-890, 1997.

김 성 동

1987년 3월~1991년 2월 서울대학교 컴퓨터공학과 학사. 1991년 3월~1993년 2월 서울대학교 컴퓨터공학과 석사. 1993년 3월~1999년 8월 서울대학교 컴퓨터공학과 박사. 1999년 8월~2001년 2월 서울대학교 컴퓨터신기술공동연구소 특별연구원. 2001년 3월~현재 한성대학교 컴퓨터공학부 조교수. 관심분야는 자연언어처리, 기계번역, 기계학습, 데이타마이닝