문맥 윈도우의 크기와 위치 변화를 이용한 한국어 기반 명사구 인식

전수영[0] 강인호 김길창
한국과학기술원 전산학과
(syjeon[0], ihkang, gckim)@csone.kaist.ac.kr

Korean BaseNP Identification using the variation of context length and position

Su Young Jeon[0] In-Ho Kang Gil Chang Kim
Dept. of Computer Science, KAIST

요 약

한국어의 비재귀 명사구 즉 기반 명사구(baseNP)를 인식하는 알고리즘을 제시한다. 본 논문에서는 한 개의 주어진 학습 알고리즘에 대해 문맥 윈도우의 크기와 문맥 윈도우의 위치를 달리해 가면서 학습시킨다. 이러한 방법을 통해 서로 다른 정보를 바탕으로 한 기반 명사구 인식을 수행할 수 있으며, 그 결과 서로 다른 여러 개의 결과들을 생성할 수 있다. 본 논문에서는 이렇게 얻어진 여러 개의 인식 결과들을 적절한 방법으로 결합하여 한국어에서 91% 이상의 높은 기본명사구 인식 정확도를 얻어낼 수 있다. 15만 단어 규모의 국어정보베이스의 말뭉치를 사용했으며, 학습 알고리즘으로는 메모리 기반 학습 알고리즘(*memory-based learning*)을 이용하여 실험하였다.

1. 서 론

기반 명사구란 내부에 다른 명사구를 포함하지 않은 명사구, 즉 비재귀 명사구를 의미한다. 기반 명사구 인식은 많은 대규모 자연 언어 처리 응용분야와 정보 추출에서 기본적인 요소로 사용되며, 부분 파싱(partial parsing)의 첫 단계로 필요한 요소이다[1].

(Tjong Kim Sang , 2000)은 여러 개의 학습 알고리즘들(ALLiS, IGTREE, MacEnt, MBL, SNoW)을 사용해서 나온 결과를 결합함으로써 영어의 기반 명사구 인식 성능을 향상시킬 수 있음을 보였다[2]. 그리고 (In-Ho Kang, 2001)는 한 개의 학습 알고리즘만으로 학습 데이터 상의 문맥 윈도우의 위치와 문맥 윈도우의 크기를 변화시킴으로써 서로 다른 여러 개의 결과들을 얻어냈다. 그런 후 이를 결합하는 것만으로 마치 여러 개의 학습 알고리즘을 적용해서 나온 결과들을 결합하는 것과 유사한 성능 향상이 가능함을 보였다. 영어의 기반 명사구 인식 정확도를 약 1% 더 향상시킬 수 있었다.

본 논문에서는 (In-Ho Kang,2001)의 기법을 한국어 기반 명사구 인식에 적용해 보았으며 그 결과 높은 정확도를 얻을 수 있었다.

본 논문의 구성은 다음과 같다. 2장에서 기반 명사구 인식에 관한 관련 연구들을 살펴보고, 3장에서 문맥 윈도우 변화 방법 과 결합 방법에 대해 설명한다. 그리고 4장에서 본 논문의 실험 결과를 분석하고, 5장에서 결론을 맺는다.

2. 관련 연구

Ramshaw 와 Marcus(1995) 는 기반 명사구 인식을 일종의 태깅 작업으로 정의하였다. 각각의 단어는 기반 명사구에 속하는 것(I)이나, 기반 명사구에 속하지 안는 것(O)으로 구분되며, 어떤 기반 명사구의 바로 뒤에 나오는 기반 명사구의 첫 단어는 태그 (B)를 할당받는다[3].

(In-Ho Kang, 2001)은 위의 IOB 대신에 위치정보를 줄 수 있도록 FLOIS 태그를 사용하였다. (1) 기반명사구의 첫 단어는 F, (2) 기반 명사구의 마지막 단어는 L, (3)기반 명사구에 속하지 안는 단어는 O, (4)기반 명사구 안의 단어는 I, (5)한 개의 단어로 된 기반 명사구는 S 태그를 할당한다. 메모리 기반 학습 알고리즘을 사용하여 많은 다른 결과들을 만들어 낸 후, 이들을 결합하는 모델을 제시하였고, Penn treebank를 사용해 실험한 결과 영어의 기반 명사구 인식에서 93.58% 정확율과 92.80%의 재현율을 보였다

(양재명 , 2000)은 한국어 기반 명사구 인식에 규칙 기반 학습을 적용하였다. 초기 예측이 표시된 학습 말뭉치에 대해 규칙 템플릿을 이용함으로써 학습 말뭉치를 목표 말뭉치에 가장 가깝도록 변환하는 일련의 규칙들을 얻어내는 방식이다. 국어정보베이스를 이용해 실험한 결과 91.8% 정확율과 90.7% 재현율을 보였다[4].

3. 문맥 윈도우 변화 방법과 결합 방법

3.1 한국어 기반 명사구 인식의 특징

영어의 기반 명사구 인식과 한국어의 기반 명사구 인식은 서로 상당한 차이를 보인다.

첫째, 영어의 경우는 한정사를 포함하여 명사구의 지배 성분으로 끝나는 명사구의 앞부분으로써, 후치 수식 구문을 제외한 부분을 대상으로 한다. 대부분 관사와 소유격을 이용하여 그 시작 위치를 파악할 수 있으므로 시작 위치의 인식이 끝 위치의 인식보다 비교적 쉬운 편이나 인식 난이도의 차이는 그리 크지 않다.

반면에 한국어의 경우는 지배 성분 후위의 원칙에 의하여, 한국어 명사구에는 후치 수식 구문이 존재하지 않는다. 즉 동일한 의미를 가지는 문장에서, 영어에서의 후치 수식 구문은 한국어에서 전치 수식 구문으로 사용된다. 따라서 명사구의 시작 위치를 찾기가 어려운 특성을 가진다. 그러나 한국어 명사구의 대부분은 바로 뒤에 조사가 붙는 특성이 있으므로 끝 위치를 찾기는 비교적 쉬운 편이다[5].

둘째, 영어의 경우는 품사 태깅을 각각의 단어에 대해 수행하지만, 한국어는 단어가 아닌 각 형태소에 대해서 품사 태그를 할당한다. 즉 한국어의 경우 한 개의 단어가 실질 형태소와 형식 형태소를 포함하여 둘 혹은 그 이상의 형태소를 가진다[4]. 이러한 차이점은 기반 명사구

인식시에 가장 중요한 기여 요소인 품사 정보를 영어와 한국어는 서로 달리 본다는 뜻이다. 예를 들어 아래의 두 문장을 비교해 보자.

(1) The student was saying.

(2) 그 학생은 말하고 있었다.

문장(1)에서는 각각의 단어{ the, student, was, saying, .}가 태그 부착의 기본 단위가 되고, 문장 (2)에서는 { 그, 학생, 은, 말, 하, 고, 있, 었, 다, .}처럼 각각의 형태소가 태그 부착의 기본 단위가 된다. (1)과 (2)의 기반 명사구는 [The student] 와 [그 학생] 이다. 영어의 경우는 student 앞의 The 를 보고 시작 위치를 결정하고 동사 was를 보고 끝 위치를 결정할 수 있다. 그러나 한국어의 경우 "그" 라는 정보 만으로는 시작 위치 결정이 명확하지 않다. 예를 들어 "[바로 그 학생]은 말하고 있다" 와 같이 앞에 다른 수식어가 붙을 수 있기 때문이다. 한편 끝 위치 판별은 비교적 명확한 편인데, 기반 명사구 바로 다음에 "은"이라는 조사(형식 형태소)가 나오기 때문이다. 그러므로 본 논문에서는 조사, 어미, 접사 같은 형식 형태소의 경우는 자세하게 어휘 정보까지 보도록 하고, 실질 형태소는 일부를 제외한 나머지 대부분의 경우 품사 정보 만을 본다.

본 논문은 코퍼스로 15만 단어 규모의 국어정보베이스를 이용한다. 총 322,057 형태소이며 이는 54개의 태그셋으로 품사 태그 부착되어 있다. 그리고 기반 명사구 인식 문제를 일종의 태그 부착 문제로 고려하여 { I, L, O, S }의 4가지 태그를 이용한다. (1) 기반명사구의 첫 형태소를 포함하여 기반 명사구 안의 형태소인 경우 I, (2) 기반명사구의 마지막 형태소는 L, (3)기반 명사구에 속하지 않는 형태소는 O (4)한 개의 형태소로 된 기반 명사구에는 S 를 할당한다. 이렇게 세분화된 청크 태그를 이용하면 위치 정보를 고려 할 수 있다는 장점이 있다. 특히 한국어의 경우 기반 명사구의 마지막 형태소를 잘 찾아낸다는 특성에 착안하여 L이라는 태그를 별도로 사용하였다. 한국어 기반 명사구 인식은 시작 위치 및 내부 형태소의 결정이 어렵고 많은 오류가 시작위치의 오판에서 비롯하기 때문에 학습이 잘되는 끝 위치를 별도의 태그 L로 분리해 학습하는 것이 효율적이다. 반면에 시작 위치 판별이 어렵다는 점을 고려해서 청크 태그 F는 제외하였다. 또한 청크 태그를 지나치게 세분화할 경우 코퍼스가 아주 크지 않다면 학습의 어려움이 있게 된다는 단점이 있으므로 더 이상의 세분화는 좋지 않다.

예를 들어, 문장(3)에 청크 태그를 부착하면 문장(4)와 같다. ((3),(4)는 국어정보베이스의 문장)

(3)
[한국/nq]+의/jcm [세종/nq+기지/ncn]+는/jxt [서남극/ncn 남/ncn 쉐틀란드/nq]+의/jcm [킹조지섬/nq]+에/jca 있/paa+다/ef+./sf

(4)
한국/nq/S+의/jcm/O 세종/nq/I+기지/ncn/L+는/jxt/O 서남극/ncn/I 남/ncn/I 쉐틀란드/nq/L+의/jcm/O 킹조지섬/nq/S+에/jca/O 있/paa/O+다/ef/O+./sf/O

3.2 문맥 윈도우 변화

기반 명사구 인식 시스템은 고려 대상이 되는 문맥으로부터 다양한 정보를 얻어 올바른 클래스를 결정하게 된다. 그러므로 보는 문맥의 크기와 위치를 달리 함으로써 클래스 결정시에 고려하는 정보가 달라지게 되고 결과도 다르게 나온다. 목표 형태소의 오른쪽 문맥은 왼쪽문맥 보다는 기반 명사구의 끝 위치 판별에 필요한 정보를 더 많이 제공한다. 마찬가지로 목표 형태소의 왼쪽 문맥은 오른쪽 문맥보다 기반 명사구의 시작 위치 판별을 위한 정보를 더 많이 제공한다.

본 논문에서는 문맥 윈도우의 크기와 위치를 변화시키는 것으로 서로 다른 여러 개의 인식기를 얻어낼 수 있다.

문맥 윈도우에 대해서는 C_{lp} 표기법을 사용한다. l 은 문맥 윈도우의 크기(length)를 의미하고, p 는 포커스 형태소의 문맥 윈도우 내에서의 위치(position)를 의미한다. 예를 들어, 문맥 윈도우 크기 즉 l이 4인 경우에 대해, 포커스 위치를 변화시키면 "$C_{4,0}$" " $C_{4,1}$" " $C_{4,2}$" " $C_{4,3}$" 와 같이 총 4개의 형태를 얻을 수 있다.

이와 같이 서로 다른 문맥 윈도우 크기 각각에 대해 모든 가능한 위치를 고려함으로써 클래스 결정시에 각기 다른 정보들을 고려할 수 있다.

3.3 결합 방법

본 논문의 기반 명사구 인식기는 2단계 처리 과정으로 이루어진다. 첫 단계에서는 학습 데이터의 품사 정보와 어휘 정보를 고려 하며, 두 번째 단계에서는 품사와 이전 단계에서 예측된 ILOS태그 정보를 이용하여 클래스를 결정하게 된다. 이와 같이 2단계로 구현함으로써 첫 단계에서 예측된 클래스를 주변 문맥의 품사정보와 ILOS태그 정보를 보고 필요할 경우에는 현재의 인식 결과를 수정하는 방식이다. 이와 같이 2단계를 모두 거치고 나면 각기 서로 다른 문맥정보를 보고 결정된 서로 다른 결과들을 얻을 수가 있다. 본 논문에서는 이 결과들을 간단한 다수 투표 방식을 이용해 결합한다. 즉 $C_{1,0}$ 부터 $C_{8,7}$ 까지의 결과들을 보고 가장 많이 나온 청크 태그를 택하는 방식이다. 이러한 방식을 적용한 결과 가장 성능이 좋게 나온 어느 한 개의 결과보다 더 나은 최종 인식 결과를 얻어낼 수 있었다.

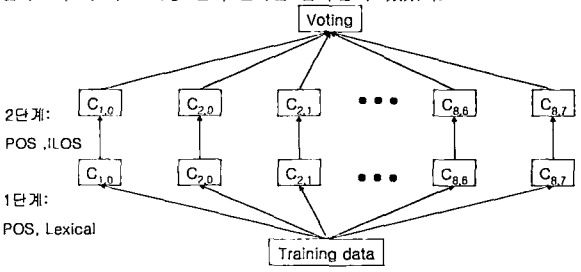


그림1. 여러 개의 서로 다른 결과를 결합하는 방식.

4. 실험

실험에 사용한 코퍼스는 (양재형, 2000) 과 동일한 국어정보베이스를 사용한다. 학습 데이터(8903문장, 240,510 형태소)와 시험 데이터(3180문장, 81,547형태소)역시 동일하게 하였으나, 본 논문의 경우 2단계 처리 과정을 거치므로 학습 데이터의 90% (8013문장)를 1단계 학습을 위해 사용하고, 나머지 10%(890 문장)를 2단계 학습을 위해 사용한다. 그리고, TiMBL(Daelemans et al., 2000)이라는 메모리 기반 학습 소프트웨어 패키지를 이용한다[6].

문맥 윈도우의 크기는 1부터 8까지를 고려하고, 각각의 크기에 대해 포커스 형태소의 모든 가능한 위치를 고려하므로 결과적으로 총 36개의 학습된 인식기를 얻는다.(그림 1참조)

첫 단계 실험에서는 문맥 윈도우 안의 각 형태소의 품사 태그 정보와 어휘 정보를 이용한다. 단 모든 품사에 대해 어휘 정보를 보는 경우 자료 희소성의 문제가 발생하여 단순히 품사 태그 정보만 보는 경우보다 성능이 떨어지는 현상이 나타난다. 또한 기반 명사구를 결정하는데 있어서 모든 어휘를 다 볼 필요는 없으므로 기반 명사구 인식에 영향력을 갖는 경우에만 어휘를 보는 방식을 채택하였다. 어휘 정보를 함께 보는 품사는 표1과 같다.

표1. 어휘정보를 보는 품사

sf, sl, sr, sd, se, su, sy	nbu, nbn
pvd, pad, px	maj
Jcs, jcc, jcv jcj, jcr, jco, jcm, jca, jct	jsc, jst, jsf
ecc, ecs, ecx, etn, etm , ef	xp, xsn, xsv, xsm, xsa

예를 들어 문맥 윈도우 크기가 5 이고 위치는 3인 경우 ($C_{5,3}$)에 1단계 학습 데이터는 다음과 같은 형태이다.

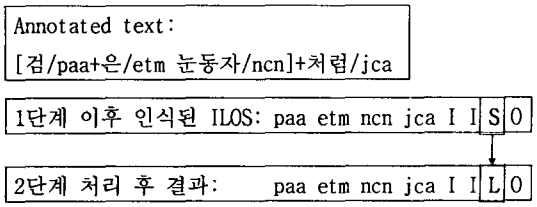
(POS-3, POS-2, POS-1, POS+0 POS+1, LEX-3, LEX-2, LEX-1, LEX+0, LEX+1, targetILOS)

여기서 POS+0와 LEX+0이 포커스가 되고 그것을 중심으로 앞으로 3개의 형태소를 보고 뒤로 1개의 형태소를 문맥 정보로 본다는 형태이다. 단 앞에서도 지적했듯이 어휘(LEX)정보의 경우는 표1에서 제시된 품사일 경우에만 포함시키도록 한다.

두 번째 단계에서는 문맥 윈도우 안의 각 형태소의 품사 태그 정보와 1단계에서 예측되었던 ILOS태그 정보를 이용한다. 즉 두 번째 단계의 학습은 첫 번째 단계에서 얻어진 ILOS태그를 가능한 한 정답에 맞게 수정하는 것을 학습하는 단계로 볼 수 있다. 문맥 윈도우 크기가 5이고 위치가 3인 경우($C_{5,3}$) 학습 데이터는 다음과 같은 형태이다.

(POS-3, POS-2, POS-1, POS+0 POS+1, ILOS-3, ILOS-2, ILOS-1, ILOS+0, ILOS+1, targetILOS)

여기서 포커스인 POS+0 에 대해 1단계에서 제시된 ILOS 태그가 ILOS+0(잘못 인식된 태그라고 가정) 이었지만 2단계의 처리 이후에 1단계에서 잘못 인식되었던 ILOS+0 값을 수정해서 새로운 ILOS태그를 할당할 수 있게 된다. 실제 실험에서 적용된 예를 보이면 아래와 같다. 1단계에서 잘못 인식된 L을 S로 수정한다.



기반 명사구 인식의 성능은 정확율, 재현율, f-rate로 측정된다. 재현율은 시험 말뭉치에 있는 모든 기반 명사구에 대해 인식기가 제시한 기반 명사구의 비율을 말하고, 정확율은 인식기가 제시한 기반 명사구들에 대해 바르게 인식된 것의 비율을 의미한다.

표2는 문맥 윈도우 크기 1부터 8까지에 대해 각각 모든 가능한 포커스 위치변화를 가지고 실험하여 나온 총 36개의 결과 중의 일부이다. $C_{4,2}$ 에서 가장 좋은 결과인 f-rate=90.28%를 얻을 수 있었다. 그리고 같은 문맥 윈도우 크기에 대해서 앞 문맥 보다는 뒤 문맥을 보지 안는 경우($C_{1,0}$, $C_{2,1}$, $C_{3,2}$, $C_{4,3}$...)에 성능이 대폭 감소하는 현상이 나타났다. 이러한 점은 한국어에서 대부분의 경우 기반 명사구의 바로 다음에 조사가 뒤따른다는 경향 때문이다. 즉 이러한 뒤 문맥을 고려하지 않으면 한국어에서는 기반 명사구의 끝 위치 판별이 어려워지기 때문이다.

그리고 각각의 결과들을 개별적으로 볼 때는 최고가 90.28%이다. 하지만 이 36개의 결과들을 단순히 다수 투표(simple majority vote) 방식으로 결합하기만 해도 크게 성능이 향상됨을 볼 수 있다.

SMV(simple majority vote) 결과 91.46%로 향상되었고, 이는 개별 결과 중 최고값인 90.28%보다도 높고 (양재형,2000)의 91.25%보다도 높은 값이다. 표3은(양재형,2000)과 본 논문의 결과를 비교한 것이다.

5. 결론

본 논문은 여러 개의 인식기를 결합하여 한국어 기반 명사구 인식에서 보다 나은 성능을 얻어 낼 수 있음을 보였다.

한 개의 주어진 학습 알고리즘을 이용하여 문맥 윈도우의 크기와 위치를 변화 시켜가며 학습하는 방식으로 여러 개의 다른 인식기를 생성할 수 있었다. 이렇게 생성된 각각의 인식기 결과는 서로 다른 학습 정보를 바탕으로 기반 명사구 인식을 수행한 것이다. 그러므로 이들을 결합할 경우 한 개의 인식기로는 보지 못했던 정보까지 고려할 수가 있었다.

그리고 한국어와 영어는 기반 명사구 인식에 있어서 서로 다른 특성을 가지고 있으므로 이러한 특성을 반영하기 위해 어휘 정보를 선별해서 보았고 , 청크 태그 역시 영어와는 차별을 두었다.

국어정보베이스 말뭉치에 대해 메모리 기반 학습 소프트웨어 패키지인 Timbl을 이용해 실험한 결과 f-rate=91.46%라는 높은 결과를 얻을 수 있었다.

본 논문에서는 최종 결합 방식으로 simple majority vote를 사용하였다. 그러나 만약 weighted vote을 이용하여 앞뒤 문맥의 가중치 차이와 문맥 윈도우의 크기와 위치에 따른 가중치 차이까지 고려한다면 앞으로 더 나은 성능 향상을 얻을 수 있다.

표2.실험 결과의 일부

	태깅정확도	정확율	재현율	f-rate
c1,0	78.87%	44.41%	51.77%	47.80%
c2,0	88.61%	80.99%	86.54%	83.67%
c2,1	86.53%	52.67%	64.45%	57.97%
c3,0	89.71%	83.41%	87.03%	85.18%
c3,1	95.38%	90.09%	89.99%	90.04%
c3,2	87.49%	54.25%	66.84%	59.89%
c4,0	89.40%	83.45%	86.83%	85.11%
c4,1	95.45%	89.56%	89.79%	89.68%
c4,2	95.60%	90.23%	90.33%	**90.28%**
c4,3	87.21%	55.10%	67.46%	60.66%
c5,1	95.30%	88.67%	89.30%	88.98%
c5,2	95.68%	89.98%	89.70%	89.84%
c5,3	95.64%	89.93%	90.07%	90.00%
c6,1	95.22%	88.88%	88.92%	88.90%
c6,2	95.69%	89.78%	89.99%	89.88%
c6,3	95.57%	89.81%	89.60%	89.70%
c7,2	95.67%	90.11%	89.78%	89.94%
c7,3	95.69%	89.98%	89.97%	89.97%
c7,4	95.60%	89.92%	89.67%	89.79%
c8,3	95.61%	89.53%	89.56%	89.55%
c8,4	95.63%	89.68%	89.56%	89.62%
c8,5	95.64%	90.02%	89.66%	89.84%
SMV	**96.04%**	**91.74%**	**91.18%**	**91.46%**

표3. 결과비교

	정확율	재현율	f-rate
SMV	91.74%	91.18%	91.46%
양재형2000	91.80%	90.70%	91.25%

6. 참고문헌

[1] (Ih-Ho Kang, 2001) In-Ho Kang, Yeojin Lee and GilChang Kim. Improving accuracy in base noun phrase identification with the variation of context length and position. NLPRS.

[2] Erik F. Tjong Kim Sang, Walter Daelemans, Herver Dejean, Rob Koeling, Yuval Krymolowski, Vasin Punyakanok, and Dan Roth. 2000. Applying system combination to base noun phrase identification. In Proceedings of COLING2000.

[3] L. Ramshaw and M. Marcus. 1995. Text chunking using transformation-based learning. In Proceedings of the Third Workshop on Very Large Corpora.

[4] 양재형. 2000. 규칙 기반 학습에 의한 한국어의 기반 명사구 인식. 정보 과학회 논문지 제 27권 제10호.

[5] 이신목, 강인호, 김길창. 방향성을 이용한 한국어 비재귀 명사구 인식 모델. 2001 제13회 한글 및 한국어 정보처리

[6]Walter Daelemans, Jakub Zavel, Ko ban der Sloot, and Antal van den Bosch. 2000. Timbl: Tilburg memory based learner, version 4.1, reference guide. In ILK Technical Report-ILK01-04, Available from http://ilk.kub.nl/downloads/pub/papers/ilk0104.ps.gz