Connectionism 을 이용한 부분 구문 인식기의 구현

정한민*, 여상화, 김태완, 박동인
시스템공학연구소 자연어정보처리연구부 기계번역연구실

An Implementation of Syntactic Constituent Recognizer Using Connectionism

Hanmin Jung, Sanghwa Yuh, Taewan Kim, Dong-In Park
Machine Translation Lab., Natural Language Information Processing Department, SERI

{jhm, shyuh, twkim, dipark}@seri.re.kr

요약

본 논문은 구문 분석의 검색 영역 축소를 통한 구문 분석기의 성능 향상을 목적으로 connectionism 을 이용한 부분 구문 인식기의 설계와 구현을 기술한다. 본 부분 구문 인식기는 형태소 분석된 문장으로부터 명사-주어부와 술어부를 인식함으로써 전체 검색 영역을 여러 부분으로 나누어 구문 분석 문제를 축소시키는 것을 목적으로 하고 있다. Connectionist 모델은 입력층과 출력층으로 구성된 개선된 퍼셉트론 구조이며, 입/출력층 사이의 노드들을, 입력층 사이의 노드들을 연결하는 연결 강도(weight)가 존재한다. 명사-주어부 및 술어부 구문 태그를 connectionist 모델에 적용하며, 학습 알고리즘으로는 개선된 백프로퍼게이션 학습 알고리즘을 사용한다. 부분 구문 인식 실험은 112 개 문장의 학습 코퍼스와 46 개 문장의 실험 코퍼스에 대하여 85.7%와 80.4%의 정확한 명사-주어부 및 술어부 인식을, 94.6%와 95.7%의 명사-주어부와 술어부 사이의 올바른 경계 인식을 보여준다.

Keyword: connectionism, syntactic constituent, neural network

1. 서론

형태소 분석 결과의 직접적인 구문 분석기로의 적용은 문장 전체를 검색 영역으로 요구하여 많은 수행 시간과 파싱 결과들의 생성을 초래한다. 이러한 문제를 해결하고자 구문 분석 전 단계에서 특정한 구문 요소를 인식하여 검색 영역을 분할/축소하려는 부분 구문 인식 기법이 생겨나게 되었다[Lavie94][Lyon95][Nasukawa95].

특히, [Lyon95]는 신경망의 패턴 매칭 능력을 자연어의 구문 요소 인식에 이용하였다. 이 시스템은 양성과 음성 정보의 동시 모델링, 트라이그램의 통계 정보를 이용하고 있다. 그렇지만, 인식 대상이 주어부로 한정되어 있으며 전체 문맥이 아닌 트라이그램만을 통계 정보로 이용한다는 한계를 가지고 있다. 또한, 다른 구문 요소를 위한 시스템의 확장성을 고려하지 못하여 유연성이 없으며 별도의 트라이그램을 위한 금지 테이블이 있어야 한다는 문제점이 있다.

일반 문장의 구문 범주는

[pre-subject] [NP-SBJ] [VP] [post-vp]

와 같이 나누어질 수 있는데, 본 시스템에서의 부분 구문 인식 대상은 명사-주어부(NP-SBJ) 및 술어부(VP)로 한다. Pre-subject 로는 "In July,", "Despite recent declines in yields,"등의 PP 가 올 수 있으며, post-vp 로는 ".", "." 등의 문장 부호나 문장 부호들의 나열이 올 수 있다. 본 시스템에서는 명사-주어부 및 술어부의 범주를 combined 코퍼스[Marcus93]에서 사용한 구문 범주로 정의한다.

다음은 위의 일반 구문 범주로 분리된 문장들을 보여준다.

"[pre-subject] Previously, [NP-SBJ] Mr. Vitulli, 43 years old, [VP] was general marketing manager of Chrysler Corp.'s Chrysler division [post-vp] . "
"[NP-SBJ] I [VP] draw a blank [post-vp] . "
"[NP-SBJ] PS of New Hampshire, Manchester, N.H., [VP] values its internal reorganization plan at about $2.2 billion [post-vp] . "

본 논문은 다음과 같이 전개된다. 2 장에서는 시스템 구성도를, 3 장에서는 connectionist 모델을 설명한다. 4 장은 실험 및 결과 분석을 보여주며 결론이 이어진다.

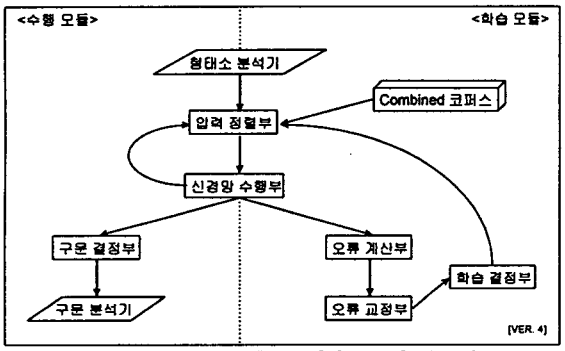
2. 시스템 구성도

부분 구문 인식기는 크게 수행 모듈과 학습 모듈로 나누어진다.

수행 모듈은 형태소 분석기와 구문 분석기를 연결하는 역할을 한다. 형태소 분석된 결과는 입력 정렬부에 의하여 입력층에 적용된다. 이때, 각 단어의 문장 내 위치와 품사 태그는 입력층에서의 위치 결정에 이용된다. 다음은 입력층으로부터 출력 결과를 계산하는 단계를, 구문 결정부는 이 계산 결과를 이용하여 명사-주어부와 술어부를 결정하여 구문 분석기로 넘기는 단계를 수행한다.

학습 모듈은 combined 코퍼스로부터 추출된 구문 태깅 결과를 이용하여 connectionist 모델을 학습시킨다. 학습 방법은 개선된 백프로퍼게이션 학습 알고리즘을 이용하여 허용 오차를 기준으로 학습 종료를 결정한다.

그림 2.1 은 부분 구문 인식기의 시스템 구성을 보여준다.



<그림 2.1> 부분 구문 인식기의 시스템 구성도

3. Connectionist 모델

3.1 모델 설계 원칙

Connectionist 모델의 설계 원칙은 다음과 같다.

첫째, 수행 시간, 주기억 장치에의 부담을 고려하여야 한다. 즉, connectionist 모델의 구조는 간단하여야 하여 입력 패턴에 대한 연산 횟수를 줄여야 한다.

둘째, 문맥 정보를 충분히 반영하여야 한다. [Lyon95]에서는 문맥 정보로서 바이그램(bigram)과 트라이그램(trigram)을 사용하고 있으나, 이는 문장 전체의 문맥을 충분히 반영하지 못한다는 문제점을 가진다. 전체 문맥을 반영하기 위해서는 connectionist 모델 내에서 문장 길이 및 각 단어 사이의 연관성을 이용하도록 설계되어야 한다. 각 단어 사이의 연관성은 형태소 분석 결과로부터 얻어진 품사 태그를 이용하여 구할 수 있다.

셋째, 가능한 모든 정보를 학습에 이용하여야 한다. 코퍼스로부터의 학습 시에 출력층의 각 노드에 대한 변별력있는 활성값을 얻기 위하여 입력 패턴의 흥분(excitation)과 억제(inhibition) 양상에 변화를 주어야 한다.

넷째, 결과에 대한 대책이 있어야 한다. 부분 구문 인식기는 구문 분석기의 전단계 역할을 하여 구문 분석의 성능을 향상시키고자 하는 목적을 가지고 있으나 상황에 따른 대처를 위하여 플러그 인/아웃 기능을 가지고 있어야 한다. 즉, 미흡한 부분 구문 인식 결과를 배제하기 위한 별도의 노력이 필요하지 않도록 하여야 한다. 이를 위하여 부분 구문 인식으로 얻어진 구문 태그들을 형태소 분석 결과 내의 부가적인 정보 필드가 되도록 한다.

3.2 입/출력

Connectionist 모델의 입력과 출력은 모듈의 종류 - 수행 모듈과 학습 모듈 - 에 따라 다르다.

수행 모듈의 입력은 형태소 분석기를 거쳐 품사 태깅된 단어들의 나열로서, 각 단어에는 품사 태그가 부여되어 있다. 출력은 형태소 분석 결과에 부분 구문 인식기에 의해 명사-주어부와 술어부의 구문 태그가 추가된 형태를 가진다.

아래는 수행 모듈의 입/출력 예를 보여준다(품사 태그는 Penn Treebank 에서 정의된 48 개와 복합 단위 인식[정한민 96]을 위한 3 개를 포함한 51 개이며 번호로 부여된다, 구문 태그의 숫자는 문장 내에서의 단어 번호이다).

[입력 문장]

Four of the five surviving workers have asbestos-related diseases, including three with recently diagnosed cancer.

[형태소 분석 결과 / 수행 모듈 입력]

(2 Four) (6 of) (3 the) (2 five) (29 surviving) (13 workers) (31 have) (7 asbestos-related) (13 diseases) (38 ,) (29 including) (2 three) (6 with) (20 recently) (30 diagnosed) (12 cancer) (37 .)

[부분 구문 인식 결과 / 수행 모듈 출력]

(OPEN-NP-SBJ 1) (2 Four) (6 of) (3 the) (2 five) (29 surviving) (13 workers) (CLOSE-NP-SBJ 7) (OPEN-VP 7) (31 have) (7 asbestos-related) (13 diseases) (38 ,) (29 including) (2 three) (6 with) (20 recently) (30 diagnosed) (12 cancer) (CLOSE-VP 17) (37 .)

학습 모듈은 학습을 위한 코퍼스로 combined 코퍼스 [Marcus93]를 이용한다. Combined 코퍼스는 품사 태그와 구문 태그가 결합된 형태이므로 connectionist 모델의 입력으로 사용하기에 용이하다. 학습 모듈의 입력은 각 단어의 위치에 해당되는 품사 태그와 connectionist 모델에서 사용하는 4 개의 구문 태그(OPEN-NP-SBJ, CLOSE-NP-SBJ, OPEN-VP, CLOSE-VP)이며, 이들의 흥분과 억제를 통하여 학습을 수행한다. 학습 모듈의 출력은 출력층의 NP-SBJ 와 VP 의 2 개 노드 중 입력층에서의 흥분/억제 양상에 따라 한 노드가 활성화되는 형태이다.

3.3 Connectionist 모델 구조

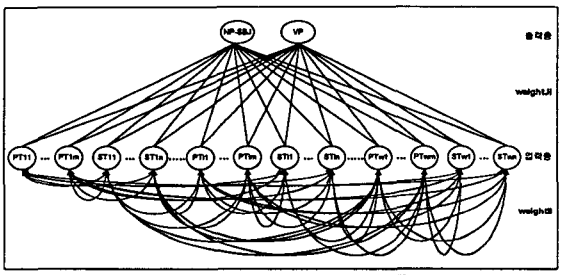
부분 구문 인식을 위한 connectionist 모델은 은닉층없이 입력층과 출력층으로만 구성된 개선된 퍼셉트론 구조이며, 입/출력층 노드들 사이 및 입력층의 노드들 사이에서 연결강도(weight)를 가진다. 은닉층을 사용한 one-layer 퍼셉트론 구조를 사용하지 않은 이유는 실험 결과 인식율에서 현재의 모델과 차이가 거의 없으며 시간 복잡도만 크게 증가하기 때문이다.

입력층은 문장의 각 단어에 부여된 품사 태그를 위한 노

드들과 구문 태그를 위한 노드들로 구성된다. 현재 하나의 단어에 대하여 품사 태그 노드 수는 51개이며 구문 태그 노드 수는 4개이다. 입력층의 전체 노드 수는 한 문장의 최대 단어 수를 w로 정의할 때, (51+4)*w개가 된다. 입력층의 흥분/억제화는 입력 문장의 단어 수에 따라 유동적으로 일어난다. 즉, 입력 문장의 단어 수에 의해 흥분/억제되는 노드 수가 결정된다. 예를 들어, 입력 문장의 단어 수가 w이며 각 단어가 품사적 모호성을 가지고 있지 않을 경우에는 w+4개의 노드(4는 구문 태그의 수)에서 흥분/억제가 일어난다.

출력층은 NP-SBJ와 VP의 2개 노드로 구성된다. 입력에 대한 계산 결과에 따라 이들 중 하나의 노드가 활성화된다(활성값은 시그모이드 함수에 의하여 0부터 1사이의 값을 가진다).

본 connectionist 모델의 입력층은 단어가 품사적 모호성을 가진 문장들도 처리할 수 있도록 설계되었다. 한 단어에 대하여 여러 품사 태그가 부여된 경우에는 해당 단어 노드의 품사 태그들이 동시에 흥분된다. 이에 대한 실험은 실제 형태소 분석기와의 결합 후에 이루어질 예정이다.



* PT: 품사 태그 노드, ST: 구문 태그 노드
* m: 품사 태그 노드 수, n: 구문 태그 노드 수, w: 최대 단어 수
<그림 3.1> Connectionist 모델 구조

연결 강도는 입/출력층 사이의 노드를 연결하는 연결 강도 weightJI와 입력층의 노드들을 연결한 양방향 형태의 연결 강도 weightII로 구성된다. WeightJI는 완전 연결된 형태를 가지며 그 수는 2*w*(m+n)개이다(2는 출력층의 노드 수). WeightII는 품사 태그 노드와 구문 태그 노드 사이에서만 연결되어 있으며 그 수는 양방향을 고려하여 2*m*n개이다.

3.4 학습

Connectionist 모델의 학습은 개선된 백프로퍼게이션 학습 알고리즘을 사용한다. 학습은 다음과 같은 과정으로 진행된다.

첫째, 기본 흥분/억제값을 입력층의 해당 노드에 부여한다. 이때, 학습은 출력층의 활성화를 조절할 수 있도록 입력 패턴의 흥분/억제 양상에 변화를 줄 수 있도록 한다. 입력 패턴의 변화는 4개의 구문 태그가 어느 위치에서 흥분 또는 억제되느냐로부터 시작된다.

다음은 입력 패턴의 흥분/억제 양상 변화에 따른 출력층의 활성화 예를 보여준다(밑줄친 부분은 흥분 상태, 이탤릭체 부분은 억제 상태, f(x)는 시그모이드 함수이다).

[문장]
No price for the new shares has been set.

[Combined 코퍼스]
(OPEN-NP-SBJ 1) (3 No) (12 price) (6 for) (3 the) (7 new) (13 shares) (CLOSE-NP-SBJ 7) (OPEN-VP 7) (32 has) (30 been) (30 set) (CLOSE-VP 10) (37 .)

[출력층의 NP-SBJ 노드 활성화를 위한 입력 패턴]
입력층 -> (OPEN-NP-SBJ 1) (3 No) (12 price) (6 for) (3 the) (7 new) (13 shares) (CLOSE-NP-SBJ 7) *(OPEN-VP 7)* (32 has) (30 been) (30 set) *(CLOSE-VP 10)* (37 .)
출력층 -> NP-SBJ 노드 : MaxValue(f(x)), VP 노드 : MinValue(f(x))

[출력층의 VP 노드 활성화를 위한 입력 패턴]
입력층 -> *(OPEN-NP-SBJ 1)* (3 No) (12 price) (6 for) (3 the) (7 new) (13 shares) *(CLOSE-NP-SBJ 7)* (OPEN-VP 7) (32 has) (30 been) (30 set) (CLOSE-VP 10) (37 .)
출력층 -> NP-SBJ 노드 : MinValue(f(x)), VP 노드 : MaxValue(f(x))

둘째, 입력 패턴에 입력 노드들 사이의 연결 강도 weightII를 적용하여 흥분/억제값을 재조정한다. 학습은 combined 코퍼스의 각 문장에 대하여 NP-SBJ와 VP 노드를 활성화시키도록 두 가지로 입력 패턴을 변화시킨다. 입력 패턴은 연결 강도 weightII를 반영하기 위하여 각 흥분/억제 노드의 흥분값과 억제값을 재조정한다.

흥분/억제값의 재조정 알고리즘은 다음과 같다.

```
for (입력 문장에 대응하는 범위 내의 모든 노드들에 대하여) {
  if (현재의 노드가 구문 태그 노드 STi인 경우) {
    if (STi>0)          // 흥분된 구문 태그 노드
      Value(STi) += Σ(weightII[i][j]*PTj), where PTj>0 and 1<j<w
    else if (STi<0)     // 억제된 구문 태그 노드
      Value(STi) -= Σ(weightII[i][j]*PTj), where PTj>0 and 1<j<w
  }
  else if (현재의 노드가 0이 아닌 값을 가진 품사 태그 노드 PTi인 경우)
    for (현재 단어 범위 내에 있는 모든 구문 태그 노드들에 대하여)
      Value(PTi) += Σ(weightII[i][j]*STj), where STj!=0 and 1<j<w
}
```

셋째, 입력 패턴과 연결 강도 weightJI를 결합하여 각 출력층의 노드에 대한 출력값을 계산한다. 계산 결과는 0부터 1사이의 출력값을 가지는 시그모이드 함수 $f(x) = 1/(1+e^{-x})$을 적용하여 그 값을 출력층의 출력값으로 한다.

넷째, 입력 패턴의 목표 출력과 실제 출력값의 차로부터 오차 δ를 계산하여 출력층의 오프셋과 연결 강도 weightJI, weightII를 변경한다. 오차 δ의 계산식은 다음과 같다.

$\delta = (Ti - Oi)*(e^{-x}/(1+e^{-x})^2)$
Ti는 i번째 출력층 노드의 목표 출력값, Oi는 i번째 출력층 노드의 실제 출력값

다섯째, 오류 허용 범위(현재는 -0.0005 ~ 0.0005) 내로 모든 입력 패턴의 계산 결과가 수렴하면 학습을 종료한다.

3.5 수행

학습된 connectionist 모델에 형태소 분석 결과(현재는 combined 코퍼스로부터 추출한 품사 태깅된 문장들을 사용)를 적용하는 수행 모듈은 구문 태그의 입력 패턴에의

적용 과정을 포함한다. 이는 형태소 분석 결과로부터의 입력 패턴이 학습에 사용한 입력 패턴과 달리 어떤 구문 태그도 포함하고 있지 않는 품사 태그의 나열이기 때문이다. 입력 패턴에 적용되는 구문 태그는 학습 모듈에서와 같이 OPEN-NP-SBJ, CLOSE-NP-SBJ, OPEN-VP, CLOSE-VP 의 4 개로 구성된다.

현재는 실험의 편의상 입력 문장들 중에서

첫째, 명사-주어부와 술어부 사이에 임의의 단어나 구(예: 부사구)가 있는 문장

둘째, 명사-주어부와 술어부가 도치된 문장

셋째, 주어부 또는 술어부가 생략된 문장(예: 명령문)

은 배제하고 있다(이는 현재의 connectionist 모델로 처리할 수 없다는 의미가 아니다, 전체 코퍼스 172 문장 중 8.1% 인 14 문장이 배제되었다).

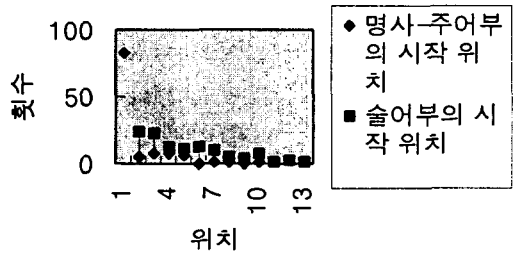
구문 태그들의 입력 노드에의 적용 알고리즘은 다음과 같다.

```
for (i=0; i<(the number of words-2); i++)
    for (j=i+1; j<(the number of words-1); j++)
        for (k=j+1; k<the number of words; k++)
            set_input_layer_proc(i, j, j, k);
```

4. 실험 및 결과 분석

4.1 코퍼스

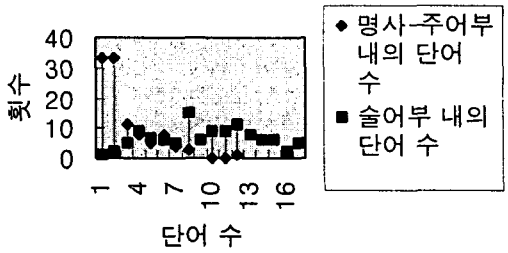
본 connectionist 모델에서 실험을 위하여 사용한 코퍼스는 Penn Treebank[Marcus93]의 "Wall Street Journal"에서 추출한 158 문장(112 개의 학습 코퍼스와 46 개의 실험 코퍼스)이다.



<그림 4.1> 명사-주어부 및 술어부의 시작 위치(학습 코퍼스)

학습 코퍼스는 한 문장의 평균 단어 수가 14.64 개인 112 개의 문장들로 이루어져 있다.

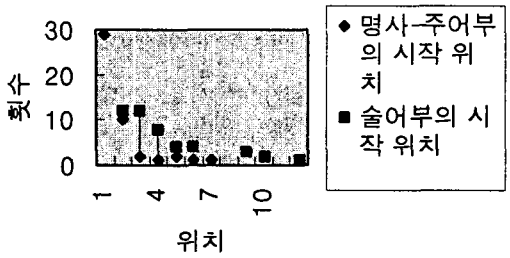
그림 4.1 은 학습 코퍼스에서의 명사-주어부 및 술어부의 시작 위치를 보여주며, 그림 4.2 는 학습 코퍼스에서의 명사-주어부 및 술어부 내의 단어 수를 보여준다.



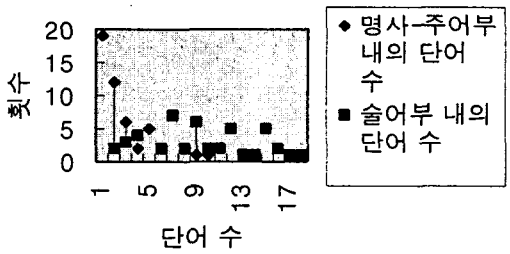
<그림 4.2> 명사-주어부 및 술어부 내의 단어 수(학습 코퍼스)

실험 코퍼스는 46 문장들로 이루어져 있으며 한 문장의 평균 단어 수가 13.89 개인 품사 태그 만이 부여된 형태를 가진다.

그림 4.3 과 4.4 는 실험 코퍼스에 대한 분석을 보여준다.



<그림 4.3> 명사-주어부 및 술어부의 시작 위치(실험 코퍼스)



<그림 4.4> 명사-주어부 및 술어부 내의 단어 수(실험 코퍼스)

4.2 실험 및 결과 분석

Connectionist 모델의 학습 코퍼스와 실험 코퍼스에 대한 실험 결과는 표 4.1 과 같다.

<표 4.1> Connectionist 모델의 실험 결과

항목 \ 코퍼스	학습 코퍼스	실험 코퍼스
문장 수	112	46
올바른 명사-주어부 및 술어부 인식	96(85.7%)	37(80.4%)
올바른 명사-주어부 인식	101(90.2%)	39(84.8%)
올바른 술어부 인식	106(94.6%)	44(95.7%)
올바른 명사-주어부와 술어부의 경계 인식	106(94.6%)	44(95.7%)

표 4.1 의 실험 결과는 올바른 명사-주어부의 인식보다

올바른 술어부의 인식 결과가 높다는 것을 보여준다. 이는 명사-주어부의 경우에 pre-subject로 올 수 있는 형태와 길이가 다양한데 반하여 술어부의 경우에는 post-vp로 올 수 있는 형태와 길이가 문장 부호나 그 나열로 제한되므로 분별력이 높아질 수 있기 때문이다. 코퍼스 상에서 pre-subject가 나타나는 비율은 학습 코퍼스의 경우에는 28문장으로 25%, 실험 코퍼스의 경우에는 17 문장으로 36.9%이다. 이런 결과에서 보듯이 명사-주어부의 높은 인식율을 얻기 위해서는 pre-subject를 포함하는 다양한 형태의 코퍼스에 대한 학습이 필요하다.

올바른 명사-주어부와 술어부의 경계 인식에 대한 실험은 95% 안팎의 높은 인식율을 보여준다. 이 정보만을 구문 분석기에서 이용하더라도 전체 검색 영역을 반으로 줄이는 효과를 얻을 수 있어서 성능 향상에 기여할 것으로 예측된다.

부분 구문 인식기를 구문 분석기와 연계할 경우에 잘못된 인식 결과가 구문 분석기에 미치는 영향을 살펴보아야 한다. 즉, 구문 분석기가 그 인식 결과를 이용하여 잘못된 구문 분석 결과를 생성하는 만큼 구문 분석기의 성능이 떨어질 수도 있기 때문이다. 그렇지만, 잘못된 인식 결과의 대부분은 구문 문법에 의하여 필터링될 것으로 예측되며, 이에 대한 실증적인 결과는 구문 분석기와의 연계된 실험을 통하여 얻을 수 있다.

5. 결론

본 논문은 우수한 분류 능력, 시스템 확장 및 학습을 통한 영역 변경의 용이성, 실시간 프로세싱의 장점을 가진 connectionism을 이용하여 구문 분석 문제를 계산학적으로 다룰 수 있는 크기로 축소시키는 문제를 기술하였다.

본 connectionist 모델은 전체 문맥을 고려할 수 있도록 문장 내 단어들에 부여된 품사 태그와 학습/수행 시에 전체 단어 정보를 활용할 수 있도록 설계되었다. 또한, 수행 시간의 단축을 위하여 입력층에 구문 태그 노드를 도입하였다. 실험을 통하여 명사-주어부 및 술어부의 인식 및 경계 인식에서도 높은 인식율을 보여, 보다 정밀한 학습 알고리즘과 구문 태그의 확장이 적용된다면 구문 분석기의 성능 향상에 큰 도움이 될 수 있음을 보였다.

앞으로의 작업은 다음과 같다.

첫째, weightII에 대한 별도의 학습 알고리즘을 도입하여 95% 이상의 학습율을 얻도록 한다.

둘째, 명사-주어부나 술어부의 생략, 도치 등의 효율적인 처리를 할 수 있도록 한다.

셋째, 구문적 깊이가 깊은 문장들에 대한 내포적 구문 인식 및 보다 많은 구문 요소의 인식이 가능하도록 한다.

넷째, 형태소 분석기와 결합시켜 connectionist 모델에서 품사적 모호성을 가진 단어들을 위한 동시 흥분 기법의 유용성을 실험하도록 한다.

다섯째, 구문 분석기와의 결합을 통하여 connectionist 모델의 유용성을 증명하도록 한다.

참고 문헌

[정한민 96] 정한민, 여상화, 채영숙, 김태완, 박동인, "효율적인 영한 번역을 위한 복합 단위 인식기 설계", 한국정보과학회 가을학술대회 논문집, 1996

[Lavie94] A. Lavie, "An Integrated Heuristic Scheme for Partial Parse Evaluation", Proceedings of ACL, 1994

[Lyon95] C. Lyon and B. Dickerson, "A Fast Partial parse of Natural language Sentences Using a Connectionist Method", cmp-lg/9503023, 1995

[Marcus93] M. Marcus, B. Santorini and M. Marcinkiewicz, "Building a Large Annotated Corpus of English: the Penn Treebank", Computational Linguistics, Vol. 19, 1993

[Nasukawa95] T. Nasukawa, "Robust Parsing Based on Discourse Information: Completing partial parses of ill-formed sentences on the basis of discourse information", Proceedings of ACL, 1995