딥러닝을 이용한 전이 기반 한국어 품사 태깅 & 의존 파싱 통합 모델

민진우 ○†, 나승훈†, 신종훈††

전북대학교†, ETRI††
Jinwoomin4488@gmail.com, nash@jbnu.ac.kr, jhsin82@etri.re.kr

A Transition based Joint Model for Korean POS Tagging & Dependency Parsing using Deep Learning

Jin-Woo Min○†, Seung-Hoon Na†, Jong-Hoon Sin††
Chonbuk National University†, ETRI††

요 약

형태소 분석과 의존 파싱은 자연어 처리 분야에서 핵심적인 역할을 수행하고 있다. 이러한 핵심적인 역할을 수행하는 형태소 분석과 의존 파싱에 대해 일괄적으로 학습하는 통합 모델에 대한 필요성이 대두 되었고 이에 대한 많은 연구들이 수행되었다. 기존의 형태소 분석 & 의존 파싱 통합 모델은 먼저 형태소 분석 및 품사 태깅에 대한 학습을 수행한 후 이어서 의존 파싱 모델을 학습하는 파이프라인 방식으로 진행되었다. 이러한 방식의 학습을 두 번 연이어 진행하기 때문에 시간이 오래 걸리고 또한 형태소 분석과 파싱이 서로 영향을 주지 못하는 단점이 존재하였다. 본 논문에서는 의존 파싱에서 형태소 분석에 대한 전이 액션을 포함하도록 전이 액션을 확장하여 한국어 형태소 분석 & 의존파싱에 대한 통합모델을 제안하였고 성능 측정 결과 세종 형태소 분석 데이터 셋에서 F1 97.63%, SPMRL '14 한국어 의존 파싱 데이터 셋에서 UAS 90.48%, LAS 88.87%의 성능을 보여주어 기존의 의존 파싱 성능을 더욱 향상시켰다.

주제어: 형태소 분석, 품사 태깅, 의존 파싱, LSTM, 통합모델

1. 서론

형태소 분석과 의존 파싱은 다양한 자연언어처리 분야에서 핵심적인 역할을 수행하고 있다. 한국어 형태소 분석은 CRF, SVM 등 기존의 기계학습 방법이 주를 이루었다[1-3]. 최근 다양한 딥러닝 모델들[4,5]을 이용한 형태소 분석 연구들이 진행되고 있는데 딥러닝 방식은 별도의 자질 추출 단계를 필요로 하지 않고 최소의 자질로부터 복잡한 자질까지 스스로 학습하는 방식이다.

형태소 분석은 음절 단위 형태소 분석 방식과 단어 단위 형태소 분석 방식으로 나뉠 수 있으며 음절 단위 형태소 분석은 입력 문장을 음절 단위로 하여 형태소 시작과 해당 형태소로 이어짐을 나타내는 [B,I]태그가 포함된 품사 태그를 부착한다. 형태소 기반 방식은 분할 된 형태소에 직접 바로 태그를 부여하는 방식이다[6].

또한, 의존 파싱 분야에서도 딥러닝을 이용한 연구들이 많이 이루어지고 있다. 딥러닝을 이용한 의존 파싱 연구는 크게 전이 기반 방식[7-11,14]과 그래프 기반 방식[12,13]으로 나뉘어 연구되고 있다. 전이 기반 방식은 입력에 대한 버퍼와 스택으로부터 자질 벡터들을 얻은 후 딥러닝 신경망을 통해 전이 액션을 결정하는 방식으로 해당 딥러닝 모델은 초기에는 FNN[7]에 기반을 두었으나 최근 들어 LSTM과 같은 RNN 계열의 모델들[8-11]이 주를 이루고 있다. 그래프 기반 방식은 전역적 탐색 방식으로 지배소와 의존소에 대한 점수를 뉴럴 신경망을 통해 계산한다.

본 논문에서는 전이 기반 의존 파싱 방식에서 형태소 분석 액션도 처리 할 수 있도록 전이 액션을 확장하여 형태소 분석 & 의존 파싱 통합 모델을 제안하고 세종 형태소 분석 데이터 셋과 SPMRL' 14 의존 파싱 데이터 셋에 적용하여 각각 형태소 분석 성능 F1 97.80%, 의존 파싱 성능 UAS 90.48%, LAS 88.87%를 보여주어 기존의 의존 파싱 성능을 더욱 향상시켰다.

2. 관련 연구

한국어 품사 태깅에 대한 다양한 연구가 진행되었다. 음절 기반 한국어 형태소 분석은 주로 순차 태깅 기반으로 연구가 진행되었는데 [1,3]에서는 각각 기계학습 모델인 CRF, Structural SVM을 적용하였고 [4]에서는 품사태깅, 개체명 인식 등 순차 태깅 문제에서 최고의 성능을 보이고 있는 딥러닝 모델인 Bi-LSTM CRF를 적용하였다. Sequence-to-Sequence 모델은 임의 길이의 한 종류의 시퀀스를 다른 한 종류의 시퀀스로 변환하는 딥러닝 모델로 기계번역 분야에서 탁월한 성능을 보여주고 있다. [5]에서는 입력문장을 해당 형태소와 품사 태그로 번역하는 모델로

표 1. 전이 액션 별 스택 및 버퍼 정보의 갱신 과정

M_t	C_t	S_t	B_t	Action	M_{t+1}	C_{t+1}	S_{t+1}	B_{t+1}
M	c, C	S	B	$Split(t)$	$(t, c), M$	C	S	B
M	c, C	S	B	$Merge$	M	C	S	B
W_{st}, M	<SP>, C	S	$(\_, u), B$	$Word$	M	C	S	$(\mathrm{wt}(W_{st}), u), B$
M	C	S	$(\mathrm{wt}(W_{st}), u), \mathrm{B}$	$Shift$	M	C	$(\mathrm{wt}(W_{st}), \mathrm{u}), S$	B
M	C	$\mathrm{u}, \mathrm{v}, S$	B	$LeftArc(l)$	M	C	$g_l(v, u), S$	B
M	C	$\mathrm{u}, \mathrm{v}, S$	B	$RightArc(l)$	M	C	$g_l(u, v), S$	B

$[W_{st} = \{(t_1, c_1), \cdots, (t_k, c_k)\}$, k: 해당 어절 내 형태소의 개수, c:형태소, t:품사태그]

보고 Sequence-to-Sequence 모델을 한국어 형태소 분석 및 품사 태깅 문제에 적용하는 연구가 진행되었다.

[6]에서는 중국어 형태소 분할 문제를 전이 기반 방식으로 적용하여 현재 음절을 현재 형태소에 부착하는 액션, 현재 형태소를 결정짓고 해당 음절을 새로운 형태소의 시작음절로 분할하는 2가지 액션으로 적용하여 기존의 성능을 향상시켰다. 또한 [18]에서는 전이 기반으로 중국어 단어 분할 및 의존 파싱에의 통합 모델에 대한 연구를 진행하였는데 [6]에서의 단어 분할을 위한 전이 액션과 의존 파싱을 위한 전이 액션을 적절하게 조합하는 방식을 사용하였다.

딥러닝을 이용한 의존 파싱 연구에는 Biaffine Attention을 적용한 그래프 기반 방식이 있고 이를 이용하여 [12]에서는 영어 데이터셋에서 적용하여 현재 시스템에서 최고 성능을 보여주었다. [13]에서는 한국어와 같은 형태학적으로 복잡한 언어에 적용할 수 있도록 문자기반 Biaffine Attention 모델로 확장하여 한국어 의존 파싱 문제에서 기존 최고 성능을 개선시켰다.

전이 기반 신경망 모델은 다양한 자연어 처리 분야에 적용되어 왔고 의존 파싱 문제에서도 탁월한 성능을 보여주고 있다. 그중에서 Stack LSTM[9-11]은 스택과 버퍼 내의 정보를 Stack LSTM을 통해 인코딩 한 후 다음 전이 액션을 결정하는 방식이다. [10]에서는 한국어와 형태학적으로 복잡하여 같이 복잡하여 단어 표상을 직접적으로 얻어내기 힘든 언어에 대해서 음절 혹은 형태소를 LSTM혹은 CNN을 통해 합성하여 단어의 표상을 얻어내는 방법들이 연구되었다. 음절 태그의 표상과 형태소의 표상들을 LSTM을 통해 얻은 후 결합하여 단어 표상을 얻어내는 하이브리드 합성 방법을 제안하였다. [11]에서는 순환 컨트롤러를 사용하여 전이액션을 결정하는 방식으로 Stack LSTM을 확장한 연구도 있었다.

Sequence-to-Sequence 모델을 이용하여 구문 구조를 [15]에서와 같이 선형화하여 시퀀스로 변환한 후 모델의 출력으로 하여 입력 문장으로부터 출력을 생성하는 방식으로 학습한다. 또한, 한국어 의존 파싱에서 어텐션 메커니즘을 이용하여 입력열에 대한 위치를 출력하는 포인터 네크워크를 이용하여 입력 어절에 대한 중심어를 찾고 해당 전이 액션을 결정하는 연구도 있었다[16].

형태소 분석 및 의존 파싱에 대한 통합 모델에 대한 연구도 진행되었는데 [17]에서는 Stack Propagation 방식을 사용하여 형태소 분석에 대한 학습을 수행한 후 파싱에 대한 학습을 수행 할 때 형태소 분석 결과에 대한 품사의 Explicit 표상을 사용하는 것 아닌 형태소 분석 단계에서 딥러닝 모델을 거친 히든 상태인 Implicit 표상을 사용하는 방식이다. 위의 방식은 형태소 분석이 잘못되었을 때 발생하는 형태소 분석 오류를 최소화 할 수 있다.

본 논문에서는 형태소 분석을 처리 할 수 있도록 전이 액션을 추가하고 [14]의 전이 기반 의존 파싱 모델을 확장하여 한국어 형태소 분석 및 의존 파싱 통합모델을 정의하고 제안 모델에 대한 실험을 진행하였다.

3. 통합 모델

3.1. 전이액션의 확장

[14]에서의 $Shift$, $Left\_Arc$, $Right_Arc$의 기존 파싱 액션 세 가지에 액션은 $Split$, $Merge$, $word$의 세 가지 품사 태깅 액션을 추가하여 총 6가지의 액션으로 구분되며 파싱 액션을 위한 버퍼와 스택인 B, S이외에 형태소 분석을 위한 버퍼와 스택 C, M을 별도로 두며 전이 액션 별 스택 및 버퍼 정보의 갱신 과정은 다음 표 1로 설명한다. 표 1의 c, <SP>, t는 각각 음절, 공백 음절, 품사 태그를 의미하고 u, v는 단어 혹은 부분 트리의 ROOT 노드를 나타내고 해당 어절의 시작 형태소와 형태소의 품사로부터 마지막 k번째 형태소와 품사까지의 집합을 $W_{st} = \{(s_1, t_1), \cdots, (s_n, t_k)\}$로 정의하며 M에서 W_{st}로부터 어절의 시작 형태소와 끝 형태소의 품사를 결합인 어절 태그를 생성하는 함수는 $\mathrm{wt}(W_{st})$, 어절 태그가 부여되지 않은 버퍼의 TOP노드는 튜플 (_, u)로 표현한다.

먼저 형태소 분석 액션인 $Split$ Action은 현재 버퍼가 가리키고 있는 음절을 새로운 형태소를 시작으로 하여 해당 음절(형태소)를 스택 M에 PUSH하는 액션으로 새로운 형태소에 품사 t를 부여하는 역할도 수행한다. $Merge$ Action은 단순히 현재 스택이 가리키고 있는 형태소에 해당 음절을 추가하는 액션으로 실제로 하는 동작은 버퍼의 Focus를 다음 음절로 이동하는 역할만을 하게 된다. $Word$ Action은 공백 음절을 만나면 자동적으로 수행되며 어절의 경계를 의미하므로 M내의 해당 어절에 대한 모든 형태소들에 대한 품사가 결정되어지기 때문에 wt함수로 M에서 어절에 대한 어절 태그를 생성하는 액션이다. 또한, 현재 버퍼의 Focus를 공백 음절에서 다음 음절로 이동하는 역할도 수행한다.

다음으로 의존 파싱에 대한 전이 액션은 [8]의 Arc-standard 방식의 $Shift, Left_Arc, Right_Arc$로 구성된다. $Shift$ Action

표 2. 통합 전이 액션의 실행 예

Action	M	C	S	B
$init$	[]	[고,향,은,<SP>,서,울,이,다,<SP>]	[]	[고향은,서울이다,root]
$Split(NNG)$	[고]	[향,은,<SP>,서,울,이,다,<SP>]	[]	[고향은,서울이다,root]
$Merge$	[고]	[은,<SP>,서,울,이,다,<SP>]	[]	[고향은,서울이다,root]
$Split(JX)$	[고,은]	[<SP>,서,울,이,다,<SP>]	[]	[고향은,서울이다,root]
$Word$	[고,은]	[서,울,이,다,<SP>]	[]	[고향은,서울이다,root]
$Shift$	[고,은]	[서,울,이,다,<SP>]	[고향은]	[서울이다,root]
$Split(NNP)$	[고,은,서]	[울,이,다,<SP>]	[고향은]	[서울이다,root]
$Merge$	[고,은,서]	[이,다,<SP>]	[고향은]	[서울이다,root]
$Split(VCP)$	[고,은,서,이]	[다,<SP>]	[고향은]	[서울이다,root]
$Split(EF)$	[고,은,서,이,다]	[<SP>]	[고향은]	[서울이다,root]
$Word$	[고,은,서,이,다]	[]	[고향은]	[서울이다,root]
$Shift$	[고,은,서,이,다]	[]	[고향은,서울이다]	[root]
$Left\_Arc$(tpc)	[고,은,서,이,다]	[]	[서울이다]	[root]
$Shift$	[고,은,서,이,다]	[]	[서울이다, root]	[]
$Right\_Arc(root)$	[고,은,서,이,다]	[]	[root]	[]

는 현재 파서 버퍼가 가리키고 있는 단어를 스택에 추가하는 액션이다. $Left\_Arc\ Action$, $Right_Arc\ Action$은 스택 내의 두 Top 노드들 간에 의존성을 형성하는 액션으로 두 노드 사이의 지배소(head)와 의존소(dep)를 결정하여 하나로 병합된 파스 트리를 다시 스택에 추가하는 액션이다. 파스 트리를 생성하는 함수는 $g_l(v,u)$로 여기서 l은 의존 관계 레이블을 나타내며 괄호의 앞 요소(v)가 지배소를 나타내고 뒷 요소(u)는 의존소를 표현한다.

위의 표 2는 전이 액션의 실행 과정을 보여준다. 표 2에서 보듯이 입력은 공백을 포함한 한국어 원문이고 기본적으로 음절 단위로 전이 액션을 결정하게 된다. 만약 버퍼 C가 가리키고 있는 음절이 "공백"이 아닌 경우에는 파싱 액션이 아닌 품사 태깅 액션을 결정하게 되는 반면 "공백"인 경우에는 어절의 경계가 결정되기 때문에 $Word$액션이 자동으로 수행되어 어절태그가 생성된 후 해당 시점에 파싱 액션을 결정하도록 하여 형태소 분석 액션과 파싱 액션이 특정 상황에서만 발생하도록 제한하였다.

3.2. 버퍼와 스택의 TOP 노드 표상

두 버퍼 C,B에 대한 표상은 다음 두 입력 임베팅 벡터열 $\boldsymbol{x}=\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\}$, $\boldsymbol{z}=\{\boldsymbol{z}_1,\cdots,\boldsymbol{z}_n\}$로부터 각각 LSTM을 통해 은닉벡터로 얻어지게 되는데 형태소 분석을 위한 음절 임베딩 벡터 $\boldsymbol{x}_i$를 얻을 때 i번째 음절을 c_i라 하고 다음 표 3과 같은 자질 유형을 사용한다.

표 3. 입력으로 사용되는 자질 유형

음절 자질	Explanation
Unigram	c_{i-1}, c_i, c_{i+1}
Bigram	$c_i c_{i+1}$
Trigram	$c_i c_{i+1} c_{i+2}$

반면, 파싱을 위해 사용되는 음절 벡터열 $\boldsymbol{z}$는 별도의 자질 추출 없이 해당 음절의 임베딩으로 구성되며 임베딩을 얻는 과정은 수식 (1),(2)로 정리한다.

$$\boldsymbol{x}_i = lookup([uni(i); bi(i); tri(i)]) \quad (1)$$
$$\boldsymbol{z}_i = lookup(c_i) \quad (2)$$

위의 수식에서 uni, bi, tri함수는 각각 표 3에 대응되어 n-gram 자질을 추출하는 함수이고 $lookup$함수는 임베딩 lookup Table을 보고 해당 임베딩 벡터를 얻어내는 함수이다. 입력열 $\boldsymbol{x}$ 와 $\boldsymbol{z}$로부터 은닉열 $\boldsymbol{h}=\{\boldsymbol{h}_1,\cdots,\boldsymbol{h}_n\}$, $\boldsymbol{k}=\{\boldsymbol{k}_1,\cdots,\boldsymbol{k}_n\}$를 얻는 과정은 수식 (3)과(4)로 보여준다.

$$\{\boldsymbol{h}_1,\cdots,\boldsymbol{h}_n\} = LSTM(\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\}) \quad (3)$$
$$\{\boldsymbol{k}_1,\cdots,\boldsymbol{k}_n\} = LSTM(\{\boldsymbol{z}_1,\cdots,\boldsymbol{z}_n\}) \quad (4)$$

형태소 분석 버퍼와 스택 C, M과 파싱 버퍼와 스택 B, S의 해당 상태 벡터들을 각각 $\mathbf{r}(C)$, $\mathbf{r}(M)$, $\mathbf{r}(B)$, $\mathbf{r}(S)$라 정의한다. S_0, S_1을 각각 버퍼 혹은 스택의 TOP, 2번째 TOP 노드로 표현하고 해당 노드의 입력 음절에 대한 위치를 얻어내는 함수는 $cpos$이며 각 상태 백터는 다음 식 (5)~(8)을 통해 얻어진다[1].

$$\mathbf{r}(C) = \boldsymbol{h}_{cpos(C_0)} \quad (5)$$
$$\mathbf{r}(M) = \boldsymbol{h}_{cpos(M_0)} \quad (6)$$
$$\mathbf{r}(B) = [\boldsymbol{k}_{cpos(B_0)}; lookup(wt(B_0))] \quad (7)$$
$$\mathbf{r}(S) = [\boldsymbol{k}_{cpos(S_0)}; \boldsymbol{k}_{cpos(S_1)}; feats(S_0); feats(S_1)] \quad (8)$$

[1] 엄밀하게 정의하면 각 노드는 음절을 포함한 튜플의 형태로 존재하여 튜플의 음절을 얻어내는 함수인 char를 이용하여 $\boldsymbol{h}_{cpos(char(C_0))}$가 정확한 수식이지만 편의상 위의 형태를 사용한다. 식 (5)~(8) 동일.

식 (5),(6)에서 보듯이 형태소 버퍼와 스택의 상태 백터 $\mathbf{r}(C)$, $\mathbf{r}(M)$는 단순히 입력 열 $\boldsymbol{x}$로부터 LSTM을 통해 얻어진 은닉열 $\boldsymbol{h}$에서 TOP노드에 해당하는 위치의 은닉 벡터의 값을 취하게 된다. 식 (5)가 수행되는 과정을 표 2로 예를 들면 처음 $Merge$ Action 이 수행되는 3번 째 줄 버퍼 C 의 TOP노드는 "은" 을 나타내고 문장 내 음절 열에서 해당 음절은 세 번째에 위치하고 있으므로 LSTM을 통해 얻어진 은닉열 $\boldsymbol{h}$에서 세 번째 은닉 상태인 $\boldsymbol{h}_3$를 취하게 되는 것이다.

파서 버퍼와 스택의 상태 백터인 $\mathbf{r}(B)$, $\mathbf{r}(S)$로 넘어가서 식 (7)로 얻어지는 $\mathbf{r}(B)$는 TOP노드에 해당하는 은닉 열 뿐만 아니라 함수 $\mathrm{wt}(a)$를 통해 a노드의 시작 형태소와 끝 형태소의 품사인 어절태그를 생성한 후 $lookup$함수를 통해 변환된 임베딩 벡터를 결합하여 상태 벡터를 얻는다. 마지막으로 식 (8)으로 얻어지는 $\mathbf{r}(S)$는 $\mathbf{r}(B)$와 동일한 과정을 통해 얻어지는 스택 S의 TOP에 위치하는 두 파스 트리의 루트의 어절 S_0, S_1의 은닉 표상 뿐 아니라 다음 표 4와 같이 노드 S_0, S_1에 대한 각각 6개의 자질 $feats(S_0)$, $feats(S_1)$을 합한 총 12 개의 자질들을 포함한다.

표 4. 파서 스택의 파스트리에 대한 자질 벡터

$feats(S_i)$
$S_i.child_1.label$
$S_i.child_1.sibling_{-1}.label$
$S_i.child_{-1}.label$
$S_i.child_{-1}.sibling_1.label$
$S_i.child_2.$label
$S_i.child_{-2}.$label

여기서 $child$는 해당 노드의 자식 노 드로 $child_{-i}, child_j$는 각각 해당 노드의 왼쪽 i번째 자식 오른쪽 j번째 자식을 의미하며 $sibling$은 해당 노드의 형제 노드로 $child$와 동일한 표기법을 사용하여 label은 상위 노드와의 의존 관계 레이블을 의미한다.

3.3. 전이 기반 품사태깅 & 의존 파싱 통합모델

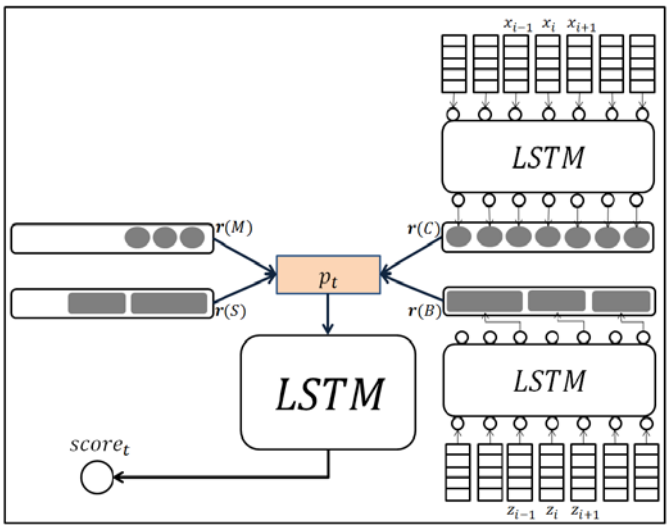

그림 1. 형태소 분석 & 의존 파싱 통합모델 구조

본 모델은 형태소 분석 및 품사 태깅에 대한 학습을 수행한 후 이어서 의존 파싱 모델을 학습하는 파이프 라인 방식과 달리 품사 태깅과 의존 파싱을 동시에 학습하는 Joint 방식이며 모델의 전체적인 도식도는 위의 그림 1와 같다. 그림 1에서 보듯이 첫번째 단계로 음절 단위로 LSTM을 통해 얻어진 은닉벡터들이 버퍼 C, B에 채워지게 된다. 그 후, 전이 액션은 버퍼와 스택의 상태를 결합하여 딥러닝 신경망을 통해 결정하게 되는 구조로 3.2절에서 정의한 $\mathbf{r}(C)$, $\mathbf{r}(M)$, $\mathbf{r}(B)$, $\mathbf{r}(S)$의 상태 벡터들을 각각 아래의 식 (9)와 같이 연결하여 파서 상태 표상을 얻는다.

$$\boldsymbol{p}_t = [\mathbf{r}(C); \mathbf{r}(M); \mathbf{r}(B); \mathbf{r}(S)] \quad (9)$$

식 (9)를 통해 얻어지게 되는 파서 상태 표상 집합을 $\boldsymbol{p} = \{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_t\}$로 표현하며 식 (10)에서와 같이 LSTM 신경망의 입력으로 하여 LSTM을 거쳐 다음 전이 액션으로의 점수 $\boldsymbol{score}_t$를 계산한다. $LSTM_t$ 함수는 LSTM을 통해 얻어진 은닉열로부터 t번째 은닉 벡터를 취하도록 하는 함수이다.

$$\boldsymbol{score}_t = \boldsymbol{W} \cdot \left(LSTM_t(\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_t\})\right) + \boldsymbol{b} \quad (10)$$

얻어진 $\boldsymbol{score}_t$는 softmax층으로 연결되어 확률로 변환한 후 전이 확률 중에 최대가 액션을 다음 전이 액션으로 하여 버퍼와 스택의 다음 상태를 결정한다. 파이프라인 방식과 제안 모델의 차이점은 해당 Action을 결정하기 위해 형태소 분석과 의존 파싱의 버퍼와 스택의 상태를 모두 고려하게 된다. 다시 말해서 품사 태깅 액션을 결정할 때 파싱 결과를 이용하고 파싱 액션을 결정할 때 품사 태깅 결과를 이용하여 서로가 작용하게 되어 보다 나은 결과를 예측하게 하는 방식이다.

3.4. 기학습 파라미터의 사용

3.2 절에서 설명한 바와 같이 형태소 분석을 위한 음절 표상은 LSTM을 통해 얻어지게 된다. 본 논문에서 사용한 SPMRL'14 데이터 셋은 정답 형태소가 부착된 2만 7천여 문장으로 형태소 분석은 의존 파싱에 비해 보다 많은 학습 데이터를 필요로 한다. 파싱 데이터 셋은 일반적으로 형태소 분석 데이터에 비해 부족하기 때문에 형태소 분석에 대한 학습이 완전하게 이루어지지 않는 문제가 발생한다.

본 논문에 대한 연구를 수행하면서 전이 기반으로 형태소 분석 만을 하는 모델에 대한 연구도 수행하였는데 모델의 구조는 음절 열로부터 LSTM을 통해 은닉 상태를 얻어내는 과정과 버퍼와 스택으로부터 상태벡터를 얻고 상태벡터를 LSTM을 통해 전이 액션을 결정하는 두 단계로 구분하며 [1]과 동일한 약 25만여 문장의 세종 형태소 분석 데이터부터 학습을 진행하였다. 위의 형태소 분석 모델의 학습을 통해 얻어진 첫번째 단계의 LSTM의 모든 파라미터 집합을 $\mathbf{mW}$라 하고 본 모델에서의 형태소 분석 데이터 부족 문제를 해결하기 위해 파라미터 집합 $\mathbf{mW}$를 식 (1)의 LSTM의 파라미터의 초기 값으로 사용하여 데이터 부족 문제를 보완하였다.

4. 실험

4.1. 실험 셋팅

본 논문에서는 학습 집합으로 SPMRL'14 한국어 의존 파싱 데이터 셋의 약 2만 7천 문장을 이용하였으며 또한 형태소 분석 실험을 위한 평가 집합으로는 [1]과 동일한 세종 형태소 분석 데이터 셋의 약 25만 여 문장을 사용하였으며 25만 문장 중 5만 문장을 테스트 문장 집합으로 하여 평가를 실시하였다. 제안 모델의 학습률은 0.0005, 모든 히든 레이어의 Dropout 비율은 0.8로 설정하였다.

4.2. 실험 결과

표 5와 6에는 각각 형태소 분석 실험 결과와 의존파싱 실험 결과가 제시되어 있다.

표 5. 세종 형태소 분석 실험 결과

	F1(morph)
CRF * [2]	97.61%
CRF(BIES) *	97.75%
Bi-LSTM CRF *	96.96%
SVM [3]	98.03%
Seq2Seq [5]	97.15%
전이기반 *	97.77%
Segment&Tag&Parsing Joint Model *	97.63%

(*는 평가셋이 동일)

표 6. SPMRL'14 의존 파싱 실험 결과

	UAS	LAS
나승훈[9] Stack LSTM	89.10	87.34
나승훈[11] Stack LSTM + 컨트롤러	89.94	88.36
민진우[14] SynTaxNet	90.33	88.69
POSTag & Parsing Joint Model	90.48	88.87

표 5에서 보듯이 동일 데이터 셋에서 기존의 CRF의 성능보다 미약하게 높은 성능을 보여주고 있으나 BIES 표기법을 사용한 CRF나 전이기반 형태소 분석 모델에 비해서는 다소 떨어지는 결과를 보여준다. 3.4절에서 기학습 파라미터를 사용하여 보완하였으나 좀 더 효과적인 방안이 필요하다.

표 6에는 SPMRL'14 한국어 의존 파싱 데이터셋에 대한기존의 방식과 본 모델에 대한 실험 결과가 제시 되어 있다. 본 모델의 베이스 라인 모델인 [14]의 SynTaxNet은 F1 measure 97.61%의 CRF로 자동 형태소 분석결과를 이용한 모델로 Joint모델의 성능이 더 높은 것을 확인 할 수 있다.

5. 결론

본 논문은 전이 기반 의존 파싱 모델에 형태소 분석을 처리할 수 있도록 전이액션을 추가 확장하여 한국어 형태소 분석 및 의존 파싱 통합모델에 대한 실험을 진행하였고 의존 파싱에서 기존의 성능을 향상시켰다. 하지만 이 방식은 형태소 분석 데이터의 부족으로 별도의 데이터로부터 미리 형태소 분석에 대한 학습 파라미터를 End-to-End 방식이 아닌 방식으로 학습이 진행되었다. 차후 End-to-End 방식으로 학습할 수 있는 방안에 대해 모색할 예정이다.

또한, 형태소 분석에서는 전이 기반 형태소 분석 모델에 비해서는 다소 떨어지는 결과를 보여주고 있는데 이러한 문제점을 보완하는 연구에 대해 진행할 예정이다.

감사의 글

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 정보통신·방송 연구개발사업의 일환으로 수행하였음. [R7119-16-1001, 지식증강형 실시간 동시통역 원천기술 개발]

참고문헌

[1] 나승훈, 양성일, 김창현, 권오욱, 김영길. "CRF 에 기반한 한국어 형태소 분할 및 품사 태깅." HCLT 2012.

[2] Seung-Hoon Na . Conditional Random Fields for Korean Morpheme Segmentation and POS Tagging. ACM Transactions on Asian and Low-Resource Language Information Processing , 14(3), 2015

[3] 2015이창기. "Structural SVM 을 이용한 한국어 띄어쓰기 및 품사 태깅 결합 모델." 정보과학회논문지: 소프트웨어 및 응용 40.12 (2013): 826-832.

[4] 김혜민, 윤정민, 안재현, 배경만, 고영중. 품사 분포와 Bidirectional LSTM-CRFs를 이용한 음절 단위 형태소 분석기, HCL 2016

[5] 이건일, 이의현, 이종혁. "Sequence-to-sequence 모델을 이용한 한국어 형태소 분석 및 품사 태깅." 한국정보과학회 학술발표논문집 (2016)

[6] Zhang, Meishan, Yue Zhang, and Guohong Fu. "Transition-Based Neural Word Segmentation." ACL (1). 2016.

[7] 이창기, 김준석, 김정희. "딥 러닝을 이용한 한국어 의존 구문 분석." 제 26 회 한글 및 한국어 정보처리 학술대회 (2014): 87-91.

[8] 이건일, 이종혁. "순환 신경망을 이용한 전이 기반 한국어 의존 구문 분석." 정보과학회 컴퓨팅의 실제 논문지 21.8 (2015): 567-571.

[9] 나승훈, 김강일, 김영길, "Stack LSTM을 이용한 전이 기반 한국어 의존 파싱," KCC 2016

[10] 나승훈, 신종훈, 김강일, "Stack LSTM 기반 한국어 의존 파싱을 위한 음절과 형태소의 결합 단어 표상 방법", HCLT 2016

[11] 나승훈, 이건일, 신종훈, 김강일, "순환 컨트롤러를 이용한 Stack LSTM기반 한국어 의존 파싱," KCC 2016

[12] T. Dozat, C. D. Manning, “Deep Biaffine Attention for Neural Dependency Parsing,” ICLR 2017
[13] 나승훈, 이건일, 신종훈, 김강일, "Deep Biaffine Attention을 이용한 한국어 의존 파싱", 한국정보과학회 학술발표논문집, 2017.6
[14] 민진우, 나승훈, "전이기반 순환유닛을 이용한 SyntaxNet 기반 한국어 의존 파싱", 한국정보과학회 학술발표논문집, 2017.6
[15] 황현선, 이창기, Sequence-to-Sequence 모델을 이용한 한국어 구구조 구문 분석, HCLT 2016
[16] 박천음, 이창기, “멀티 태스크 학습 기반 포인터 네트워크를 이용한 한국어 의존 구문 분석,” KCC 2016
[17] Zhang, Yuan, and David Weiss. "Stack-propagation: Improved representation learning for syntax." arXiv preprint arXiv:1603.06598 (2016).
[18] Kurita, Kawahara, Kurohashi, Neural Joint Model for Transition-based Chinese Syntactic Analysis, 2017