순차적 구문 분석 방법을 반영한 포인터 네트워크 기반의 한국어 의존 구문 분석기

한장훈[0], 박영준, 정영훈, 이인권, 한정욱, 박서준, 김주애, 서정연
서강대학교, 자연어 처리 연구실

hanjh04@naver.com, yeongjoon1227@gmail.com, hoon2j@gmail.com, md98765@naver.com, immerhjw@gmail.com, bakseo3060@gmail.com, juaeKim@sogang.ac.kr, seojy@sogang.ac.kr

Korean Dependency Parsing Using Sequential Parsing Method Based on Pointer Network

Janghoon Han[0], Yeongjoon Park, Younghoon Jeong, Inkwon Lee, Jungwook Han, Seojun Park, Juae Kim, Jeongyeon Seo
Sogang University, Natural Language Processing Lab

요 약

의존 구문 분석은 문장 구성 성분 간의 의존 관계를 분석하는 태스크로, 자연어 이해의 대표적인 과제 중 하나이다. 본 논문에서는 한국어 의존 구문 분석의 성능 향상을 위해 Deep Bi-Affine Network와 Left to Right Dependency Parser를 적용하고, 새롭게 한국어의 언어적 특징을 반영한 Right to Left Dependency Parser 모델을 제안한다. 3개의 의존 구문 분석 모델에 단어 표현을 생성하는 방법으로 ELMo, BERT 임베딩 방법을 적용하고 여러 종류의 모델을 앙상블하여 세종 의존 구문 분석 데이터에 대해 UAS 94.50, LAS 92.46 성능을 얻을 수 있었다.

주제어: 의존 구문 분석, Left to Right Dependency Parsing, BERT, Ensemble

1. 서론

의존 구문 분석은 문장 성분의 관계를 파악하여 문장의 구조를 이해하는 작업으로, 자연어 이해에 기반이 되는 과제이다. 전통적인 의존 구문 분석 연구는 전이 기반 방식과 그래프 기반 방식으로 구분된다. 전이 기반 방식은 버퍼와 스택으로부터 전이 액션을 결정하는 지역적 탐색 방식이다. 그래프 기반 방식은 가능한 모든 의존 구문 트리를 고려하는 전역 탐색 방식으로 의존 구문 분석에 접근한다.

포인터 네트워크[1]는 인코더와 디코더로 이루어져 있다. 인코더는 입력을 LSTM을 거쳐 문장 정보를 함축하고 함축한 정보를 디코더로 전달한다. 이 후 디코더에서 어텐션(Attention) 방식을 사용하여 의존관계를 구한다. [2]는 이러한 [1]의 연구를 활용하여 내부 스택(Internal Stack)을 사용한 스택-포인터 네트워크 모델을 제안하였다. 한편 Left to Right 포인터 네트워크[3]는 [2]와는 다르게 스택을 사용하지 않고 문장의 왼쪽부터 오른쪽으로 순차적으로 구문분석을 한다. 따라서 [2]보다 구조가 간단하고 속도가 빠르다는 장점이 있다.

본 논문에는 Left to Right 포인터 네트워크를 한국어에 적용하였다. 또한 서술어가 뒤쪽에 위치하는 한국어의 특성을 반영하여 오른쪽부터 순차적으로 구문분석을 하는 Right to Left Pointer Network 모델을 제안하였다. 또한 제안 모델 외에도 [4]의 Deep Bi-affine 모델을 추가하여 3개의 의존 구문 분석 모델에 ELMo[5], BERT[6]를 입력 임베딩으로 하여 앙상블을 수행하였고 최종적으로 UAS 94.50, LAS 92.46 성능을 얻었다.

2. 관련 연구

기존의 의존 구문 분석은 크게 그래프 기반 모델과 전이 기반 모델로 구분된다. 나승훈 외[7]의 연구에서는 그래프 기반 방식에 속하는 Deep Bi-affine Network를 한국어에 적용하여 확장한 문자 기반의 Bi-affine 모델을 제안하였다. 안휘진 외 [8]의 연구에서는 한국어 어절의 특성을 고려한 자질을 활용하여 스택-포인터 네트워크를 한국어에 적용하였다.

사전 학습된 단어 표현은 자연어 처리 태스크를 해결하는데 중요한 역할을 하였다. 그 중 ELMo는 대용량의 말뭉치로부터 독립된 여러 층의 Bi-LSTM 통해 문맥 표현(contextualized embedding)을 학습한다. 이는 단순 단어 임베딩이 담고 있지 않은 문장 내의 문맥적 정보를 표현할 수 있다는 장점이 있다. BERT는 여러 개의 층으로 이루어진 양방향 트랜스포머 인코더이다. 사전 학습 된 BERT 모델 위에 출력 층을 추가하여 fine-tuning을 하는 방식으로 사용된다. ELMo와는 달리 BERT는 모든 층의 문맥 정보를 반영하여 학습한다. 박천음 외[9]의 연구에서는 한국어 의존구문분석을 해결하기 위해 BERT를 이용한 어텐션 기반 의존 구문 분석 모델을 제안하였다.

3. 제안 방법

3.1 의존 구문 분석 방법

본 장에서는 제안시스템에 사용된 세가지 의존 구문 분석 방법을 설명한다.

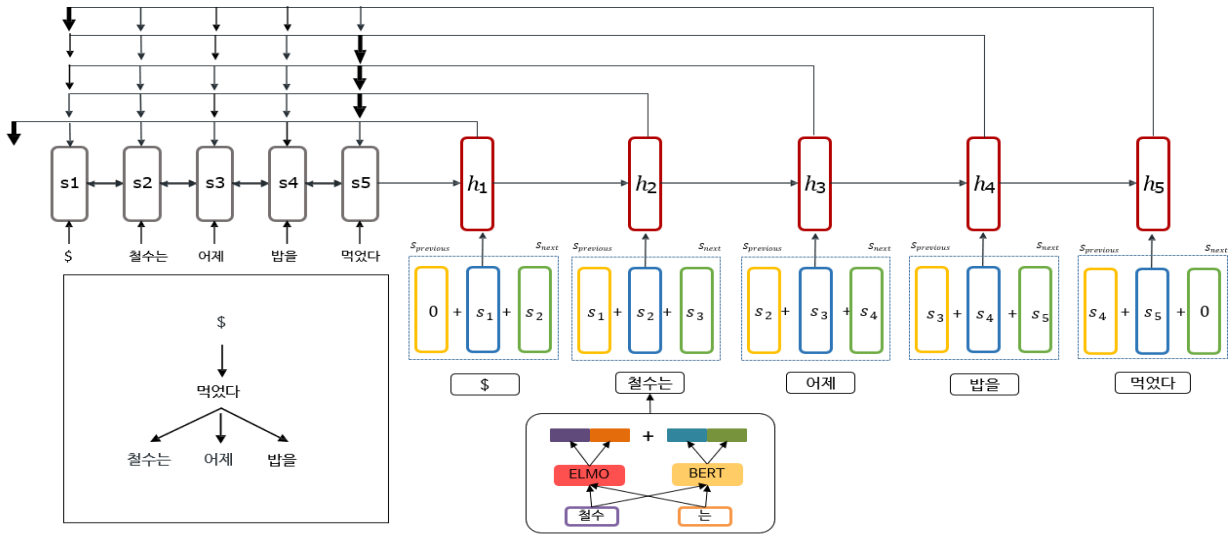

그림 1. Left to Right Dependency Parser의 디코더 모델

3.1.1 Deep Bi-Affine Network

Deep Bi-affine Network는 그래프 기반 의존구문 분석 시스템으로, 어텐션을 이용하여 의존성(arc)과 의존 관계(label)를 결정한다. 먼저 각 어절의 형태소로부터 단어 표상(word embedding) 얻은 후에 Bi-LSTM 을 적용하여 은닉 표상 $\boldsymbol{r_t}$을 얻는다. 여기에 MLP 계층을 적용하여 $\boldsymbol{h_t^{arc-dep}}$, $\boldsymbol{h_t^{arc-head}}$를 생성하고 이 과정을 문장 전체에 반복한다. 마지막으로 아래와 같은 biaffine attention 연산을 수행하여 어절 별로 head가 될 확률을 계산한다.

$$s_t^{arc} = H^{arc-head} U^{arc} h_t^{arc-dep} + H^{arc-head} u^{arc} \quad (1)$$

$$s_t^{rel} = H^{rel-head} U^{rel} h_t^{rel-dep} + H^{rel-head} u^{rel} \quad (2)$$

의존 관계($label$)의 경우에는 가능한 의존성(arc)에 대해서 모든 의존 관계의 확률 분포가 존재하기 때문에 ($arc * label * arc$) 사이즈를 갖는 학습 가중치(U^{rel})를 사용한다.

3.1.2 Left to Right Dependency Parser

Left to right Dependency Parser는 스택-포인터 네트워크를 기반으로 한 모델이다. 인코더와 디코더 두 부분으로 구성되며 인코더는 Bi-LSTM을 이용하여 각 어절의 은닉 표상을 생성한다.

디코더에서 두 모델의 차이점이 발생한다. 스택-포인터 네트워크는 매 time step 마다 스택의 맨 위에 있는 어절에 대한 의존소를 예측하는 반면, Left to Right Dependency Parser 모델은 문장의 왼쪽부터 순차적으로 어절의 지배소를 예측한다. 이렇게 하면 2n-1이었던 절차가 n으로 줄어들 뿐만 아니라 더 높은 성능을 내는 것으로 보여진다. 예측에는 어텐션 방식이 활용되고, 추가적인 입력자질로는 Grandparent과 Sibling 대신 문장 내에서 이전 어절과 이후 어절의 은닉 표상인 Prev, Next를 사용한다.

3.1.3 Right to Left Dependency Parser

본 논문에서 Left to Right Dependency Parser를 한국어의 특성에 맞게 새롭게 제안한 모델이다. 영어와는 달리 한국어는 지배소가 의존소 뒤에 오는 지배소 후위의 원칙이 있다. 따라서 디코딩 순서를 오른쪽 에서 왼쪽으로 하는 Right to Left Dependency Parser를 제안한다. 그림 1에서 LRparser(Left to right Dependency Parser)의 각 어절에 대한 지배소의 예측순서가 $0, S_5, S_5, S_5, S_1$라면 RLparser(Right to Left Dependency Parser)의 디코더는 반대인 $S_1, S_5, S_5, S_5, 0$ 순으로 의존 구문 트리를 생성한다. 이 결과 항상 지배소가 의존소보다 먼저 생성된다는 장점이 있다.

실험 결과 이 모델을 추가하여 앙상블 할 경우 기존 LRparser와 다른 특성을 지녀 더 좋은 성능을 내는데 도움이 주는것으로 확인된다.

3.2 임베딩 방법

본 장에서는 앞에서 설명한 의존 구문 분석기의 입력이 되는 단어 표현 방법을 제시한다.

3.2.1 ELMo

ELMo는 양방향 LSTM으로 구성된 언어 모델이다. 기존의 단어 임베딩과는 달리 문맥에 따라 달라지는 단어의 정보를 표현할 수 있다. 제안 시스템의 한국어 ELMo의 입출력 단위는 형태소이다. 먼저 형태소를 음절 단위로 분해한 후 CNN을 거쳐 음절 임베딩을 생성한다. 이후 음절 임베딩과 형태소 임베딩을 concat하여 ELMo의 입력표현을 만든다. 생성된 입력은 여러 층으로 이루어진 Bi-LSTM을 통과하고 이전 형태소와 다음 형태소에 대한 크로스 엔트로피를 계산하여 학습한다. 이렇게 사전 학습된 ELMo의 은닉계층을 가중합 하여 결과 값을 얻고 어절의 양 끝 형태소를 concat하여 의존 구문 분석의 임베딩으로 활용한다.

3.2.2 BERT

BERT는 여러 개의 층으로 이루어진 양방향 트랜스포머 인코더이다. Base 모델과 Large 모델 중 Base 모델을 사용하였으며 모델의 레이어의 수를 L, 히든 레이어의 크기를 H, 셀프 어텐션(self-attention)의 수를 A 라고 하였을 때 Base 모델은 L=12 H=768 A=12를 가진다. BERT는 트랜스포머와는 달리 포지션 인코딩 대신 포지션 임베딩을 사용한다. 여기에 세그먼트 임베딩과 토큰 임베딩을 추가하여 입력으로 넣는다. BERT base 모델은 12개의 인코더 블록을 지니고 있으며 각각의 인코더 블록 안에서는 멀티 헤드 어텐션(Multi-Head Attention)이 수행된다.

$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (3)$$

$$\mathrm{MultiHead}(Q,K,V) = \mathrm{Concat}(\mathrm{head}_1,\ldots,head_h)W^O \quad (4)$$

$$\text{Where } \mathrm{head_i} = Attention\left(QW_i^Q, KW_i^k, VW_i^v\right)$$

멀티 헤드 어텐션은 768차원을 가진 단어 벡터를 12개로 나누어 여러 번의 병렬적인 어텐션을 수행하는 방법이다. 어텐션 계산시 셀프 어텐션(3) 방식을 사용한다. 셀프 어텐션은 Q벡터에 대한 모든 K벡터의 어텐션 스코어를 구하고 어텐션 분포를 사용하여 모든 V벡터를 가중합한다. 이때 어텐션 스코어 함수는 Scaled Dot-Production을 사용한다. 이렇게 구한 12개의 셀프 어텐션을 concat하여 멀티헤드 어텐션(4)을 구한다. 이후 position-wise FFNN을 통과하여 인코더 블록의 출력이 된다. 12개의 인코더 블록을 거쳐 사전 학습된 BERT는 출력이 BERT 토큰 단위로 나오게 되는데 본 논문은 마지막 BERT의 4개 층의 출력을 가중합 하여 각 어절의 양 끝 토큰을 concat한 값을 모델의 입력으로 사용하였다.

3.3 앙상블

본 논문에서는 Deep Bi-affine Network, Left to Right, Right to Left 네트워크 모델의 결과를 이용한 앙상블 기법을 제안한다.

각각의 모델은 그래프 기반, 포인터 네트워크 기반 등의 서로 다른 특성을 갖고 있고, BERT, ELMo 임베딩을 서로 다르게 적용하였기 때문에 이 모델들을 앙상블한 결과 높은 수준의 성능을 얻을 수 있었다.

4. 실험

본 논문에서는 ETRI[10]에서 사전 학습한 BERT base 모델을 사용하였다. BERT를 사용하기 위한 형태소 분석으로는 ETRI 형태소 분석기를 사용하였다. ELMo는 4GB의 대용량 뉴스 데이터를 이용하여 사전 학습하였다. 의존 구문 분석모델에는 세종데이터 셋[11][12]을 사용하였으며 총 59,659문장 중 90%인 53,842문장을 학습에 사용하고 나머지 10%, 5,817문장을 평가에 사용하였다. 평가 방법으로는 Unlabeled Attachment Score(UAS)와 Labeled Attachment Score(LAS)를 사용하였다.

표 1. Concat 여부에 따른 성능비교

	Dependency parsing	UAS	LAS
1	LRparser + BERT_ft	93.59	91.34
2	LRparser + BERT_concat	93.70	91.43

[표 1]은 LRparser 모델의 입력 방식에 대한 성능차이를 나타낸다. BERT의 출력은 BERT 토큰 단위로 나오는데 이는 의존 구문 분석 모델의 입력 단위인 어절보다 작은 단위이다. 따라서 어절을 대표하는 토큰을 선택해야 하는데 [표 1]의 BERT_ft는 한 어절 안에서 첫번째 토큰을 어절의 입력으로 하는 경우이고 BERT_concat은 한 어절 안에서 첫 번째와 마지막 토큰을 concat하여 어절의 입력으로 하는 경우이다. 실험결과 첫 번째 토큰만을 사용하는 모델 보다 concat의 방식의 모델이 높은 성능을 보이는 것으로 보아 어절의 양 끝 정보를 확인하는 것이 보다 풍부한 문맥정보를 전달할 수 있다는 것을 알 수 있다.

표 2. 모델에 따른 성능 비교

	Dependency parsing	UAS	LAS
1	Biaffine	92.42	90.31
2	LRparser	92.32	90.19
3	RLparser	92.18	90.07
4	Biaffine + BERT_concat	94.07	91.92
5	Biaffine + ELMo + BERT_concat	94.10	92.03
6	LRparser + BERT_concat	94.14	92.07
7	LRparser + ELMo + BERT_concat	**94.23**	**92.11**
8	RLparser + BERT_ft	94.10	91.97

[표 2]는 모델에 따른 성능이다. 의존 구문 분석 모델인 Biaffine, LRparser, RLparser에 대해 BERT와 ELMo를 추가하여 모델을 구성하였다. 세 가지 의존 구문 분석 모델 모두에서 BERT와 ElMo 이용하여 임베딩을 추가하였을 때 성능이 향상되었으며, 따라서 의존 구문 분석에서도 문맥 정보가 모델의 성능에 향상에 중요한 요인이라는 것을 알 수 있다. 단일모델로는 LRparser 모델에 ELMo와 BERT를 추가한 방법이 UAS 94.23% LAS 92.11 로 성능이 제일 높았다.

표 3. [표 2] 모델로 조합된 앙상블 성능

앙상블에 따른 성능비교	UAS	LAS
모델A (4 ,5 ,6, 7)	94.43	92.40
모델B (4 ,5 ,6 ,7 ,8)	94.50	92.46

[표 2]의 모델 중 성능 높은 5개의 모델을 조합하여 앙상블을 수행하였다. [표 2]의 모델 A는 4개의 모델 (Biaffine + BERT, Biaffine + ELMo +BERT, LRparser + BERT, LRparser + ELMo +BERT)을 앙상블한 결과이다. 모델 A에 RLparser(8)를 추가한 모델 B의 성능이 향상된

것으로 보아 RLparser가 단일모델 성능은 낮았지만 Biaffine, LRparser가 학습하지 못한 정보를 학습한다는 사실을 알 수 있었다.

표 4. 세종코퍼스 의존구문분석 성능 비교

Dependency parsing	UAS	LAS
Biaffine attention-나승훈	91.78	89.76
deep biaffine + 스택 포인터 네트워크- 안휘진	92.17	90.08
ELMo와 멀티헤드 어텐션-박성식[13]	92.85	90.65
MLBiaffine(+ELMO) -홍승연[14]	93.12	91.00
BERT + LSTM deep biaffine-박천음	94.06	92.00
제안모델	**94.50**	**92.46**

[표 2]의 최고 성능 모델 5개로 앙상블한 방법(모델 B)이 UAS 94.50, LAS 92.46로 기존 연구보다 좋은 성능을 보였다.

5. 결론

본 논문에서는 의존 구문 분석 State of the Art 모델인 Deep Biaffine 모델과 LR parser 모델에 BERT와 ELMo를 임베딩으로 하여 학습하였다. 또한 LR parser를 한국어에 맞게 변형한 RL parser를 제안하였고 다양한 실험을 통해 성능이 높은 5개의 모델로 앙상블한 의존 구문 분석 시스템을 제안하였다. 실험 결과 세종 데이터셋에 대하여 UAS 94.50%, LAS 92.46% 의 좋은 성능을 보였다.

감사의 글

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학지원사업의 연구결과로 수행되었음(2015-0-00910)

이 논문은 한국전자통신연구원에서 공개한 한국어 언어모델(korBERT)를 사용함(No.2013-2-00131, 휴먼 지식증강 서비스를 위한 지능진화형 WiseQA 플랫폼 기술 개발)

참고문헌

[1] O. Vinyals, M. Fortunato and N. Jaitly, Pointer Networks, Neural Information Processing Systems (NIPS), pp.2692-2700, 2015

[2] X. MA, Z. Hu, J. Liu, Stack-Pointer Networks for dependency Parsing, ACL, 2018.

[3] FERNÁNDEZ-GONZÁLEZ, Daniel; GÓMEZ-RODRÍGUEZ, Carlos. Left-to-Right Dependency Parsing with Pointer Networks. arXiv preprint arXiv:1903.08445, 2019.

[4] T. Dozat and C. D. Manning. Deep biaffine attention for neural dependency parsing. arXiv preprint arXiv:1611.01734, 2016.

[5] PETERS, Matthew E., et al. Deep contextualized word representations. arXiv preprint arXiv:1802.05365, 2018.

[6] J. Devlin, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805, 2018.

[7] 나승훈, et al. "Deep Biaffine Attention을 이용한 한국어 의존 파싱", KCC, pp. 584-586, 2017.

[8] 안휘진, et al. "Deep Bi-affine Network와 스택 포인터 네트워크를 이용한 한국어 의존 구문 분석 시스템", HCLT, pp. 689-691, 2018.

[9] 박천음, et al. "BERT를 이용한 한국어 의존 구문 분석", 한국정보과학회 학술발표논문집, pp, 530-532. 2019

[10] http://aiopen.etri.re.kr/

[11] 이창기, et al. "딥 러닝을 이용한 한국어 의존 구문 분석", HCLT, pp. 87-91, 2014.

[12] 국립국어원. 21세기세종계획. 2012.

[13] 박성식, et al. "ELMo와 멀티헤드 어텐션을 이용한 한국어 의존 구문 분석", HCLT, pp. 8-12, 2018.

[14] 홍승연, et al. "BERT와 ELMo 문맥화 단어 임베딩을 이용한 한국어 의존 파싱", 한국정보과학회 학술발표논문집, pp. 491-493, 2019

[15] A. Vaswani, et al. Attention Is All You Need. Neural Information Processing Systems (NIPS), pp. 5998-6008, 2017

[16] 박찬민, et al. "한국어 ELMo 임베딩을 이용한 의미역 결정", 한국정보과학회 학술발표논문집, pp. 608-610. 2019