Graph Neural Networks을 이용한 한국어 의존 구문 분석

민진우○, 홍승연, 이영훈 나승훈

전북대학교

Jinwoomin4488@gmail.com, hongsy034@hanmail.net, dldudgns73@jbnu.ac.kr, nash@jbnu.ac.kr

Graph Neural Networks for Korean Dependency Parsing

Jin-Woo Min○, Seung-Yean Hong, Young-Hoon Lee, Seung-Hoon Na

Jeonbuk National University

요 약

구문 분석은 문장의 구조를 분석하는 자연어처리 분야로 그래프 기반 방법과 전이 기반 방법으로 나뉘어 연구되어 왔다. 본 논문에서는 그래프 기반 방식에서 높은 성능을 보이고 있는 Deep Biaffine 어텐션 모델에 별도의 High-Order 정보 추출 없이 Graph Neural Network(GNNs)을 이용하여 High-Order 정보를 학습할 수 있도록 확장한 Deep Biaffine 어텐션 GNNs을 적용하여 한국어 세종 구문 분석 셋에서 UAS : 94.44%, LAS : 92.55%의 성능을 달성하였으며 Dual Decomposition을 통해 전이 기반 한국어 구문 분석 모델과 결합하여 추가적인 성능 향상을 보였다.

주제어: 구문 분석, Deep Biaffine, 그래프 기반, GNN

1. 서론

의존 구문 분석은 지배소(Head)와 의존소(Modifier)의 관계에 따라 문장의 구조를 분석하여 파스 트리를 생성하는 자연어 처리 분야로 의존 구문 분석은 전이 기반 방식과 그래프 기반 방식의 두 갈래로 나뉘어 연구되어 왔다. 전이 기반 방식은 스택과 버퍼의 지역적 정보에 의존하여 전이 액션을 결정하는 방식이고 그래프 기반 방식은 문장 내의 모든 단어 쌍의 지배소와 의존소의 점수를 계산하여 구문 분석을 수행하는 방식이다.

본 논문에서는 BERT[2]기반 인코더와 그래프 기반 방식에서 가장 높은 성능을 보이고 있는 Deep Biaffine 어텐션[3] 모델에 별도의 High-Order 자질 추출 없이 High-Order 정보를 학습할 수 있도록 확장한 Deep Biaffine GNNs[5] 모델을 적용하여 한국어 세종 구문 분석 셋에서 UAS : 94.44%, LAS : 92.55% 로 최고 성능을 달성하였으며 이에 전이 기반 한국어 구문 분석 모델과 Dual Decomposition을 통해 결합하여 UAS : 0.04%, LAS : 0.06%의 추가적인 성능향상을 보였다.

2. 관련 연구

그래프 기반 방식에서 가장 높은 성능을 보이고 있는 방식은 Deep Biaffine 어텐션[3] 으로 영문권에서 최고 성능을 달성하고 있다. 한국어 구문 분석에서도 [7-10]에서와 같이 형태학적으로 복잡한 한국어에 적용하기 위해서 단어를 구성하는 형태소로부터 단어를 합성하는 방식으로 Deep Biaffine 모델을 적용하여 성능향상을 이루었다.

최근 자연어 처리 연구 동향에서 ELMo[1], BERT[2] 등 문맥을 고려한 사전 학습된 언어모델을 이용하여 응용 테스크에 활용하는 연구가 주를 이루고 있다[12]. 구문 분석에서도 확장된 언어 모델을 통해 성능향상을 가져왔는데 [8]에서는 ELMo와 멀티헤드 어텐션을 적용하여 성능향상을 하였으며 [9,10]는 Bert를 한국어 토큰 단위로 학습한 모델을 이용하여 한국어 구문 분석에서 최고성능을 달성하였다.

Stack Pointer Network[6]는 모델로 스택의 지배소에서 의존소를 포인팅하는 새로운 방식의 전이 기반 모델로 스택의 표상에 GrandParent, Sibling등의 High-Order 정보를 결합하여 성능 향상을 보였으며 [5]에서는 GNNs을 이용하여 명시적인 자질 추출 없이 High-Order 정보를 학습하는 모델을 제안하여 영문 구문 분석 데이터에서 성능 향상을 달성하였다.

3. GNN 기반 Deep Biaffine 모델

3.1 Deep Biaffine 어텐션 모델

Deep Biaffine 어텐션 모델에서 단어 표상을 위해 단어를 구성하고 있는 형태소 단위의 양방향 LSTM 기반 합성 방식을 사용하였으며 얻어진 단어 표상 $\boldsymbol{w}_i$과 어절의 시작 형태소와 끝 형태소의 태그를 결합한 어절태그 $\boldsymbol{t}_i$ 그리고 Bert기반 인코더를 통해 얻어진 어절의 마지막 토큰에 해당하는 출력 $\boldsymbol{b}_i$을 결합하여 다음 수식과 같이 단어 표상 $\boldsymbol{x}_i$을 얻는다.

$$\boldsymbol{x}_i = [\boldsymbol{w}_i; \boldsymbol{t}_i; \boldsymbol{b}_i;] \quad (1)$$

얻어진 입력열을 다음 수식과 같이 양방향 LSTM을 통해 인코딩 한 후 얻어진 은닉 표상에 대하여 각각 MLP를 적용하여 지배소에 대한 표상과 의존소에 대한 표상을 얻는다.

$$\{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_n\} = LSTM(\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}) \quad (2)$$
$$\boldsymbol{h}_i = MLP^{head}(\boldsymbol{z}_i), \boldsymbol{d}_i = MLP^{dep}(\boldsymbol{z}_i) \quad (3)$$

Deep Biaffine 어텐션 모델은 1) 의존성을 결정, 2)의존 관계 결정의 두 단계로 결정한다.

$$\sigma(i,j) = Softmax_i\left(\boldsymbol{h}_i^T \mathrm{W} \boldsymbol{d}_i + \boldsymbol{b}_1^T \boldsymbol{h}_i + \boldsymbol{b}_2^T \boldsymbol{d}_j\right) \quad (4)$$

의존성을 결정하기 위해서 위의 수식에서 얻어진 지배소와 의존소를 위의 수식 (5)와 같이 Biaffine Attention을 수행하여 문장 내의 모든 의존소에 대한 지배소의 스코어를 얻어 구문 분석을 수행하는 방식이다.

3.2 GNN을 통한 High Order 자질

High Order 자질은 구문 분석의 성능 향상을 위해 사용되는 방법으로 [5]에서는 [4]에서의 Graph Attention Network를 이용하여 어텐션 스코어를 이용하여 트리의 노드를 학습하는 방법으로 사용하였다.

$$\alpha_{ij}^t = \sigma(i,j) = P^t(i|j) \quad (5)$$

어텐션 스코어 $\sigma(i,j)$는 그래프에서의 간선 가중치로 설정하였으며 위의 수식 (5)와 같이 GNN 각층에서의 모든 노드 간의 간선 가중치로 표현한다. 최종 GNN 층을 τ라 할때, 각 GNN 층을 t로 표현하면 GNN을 통한 $\boldsymbol{h}_i$, $\boldsymbol{d}_i$을 다음 수식과 같이 업데이트 된다.

$$\begin{cases} \mathbf{h}_i^t = g\left(W_1 \sum_{j \in N(i)} \alpha_{ji}^t \mathbf{h}_j^{t-1} + B_1 \mathbf{h}_i^{t-1}\right) \\ \mathbf{d}_i^t = g\left(W_2 \sum_{j \in N(i)} \alpha_{ij}^t \mathbf{d}_j^{t-1} + B_2 \mathbf{d}_i^{t-1}\right) \end{cases} \quad (6)$$

$$\begin{cases} \mathbf{h}_i^t = g\left(W_1 \sum_{j \in N(i)} \alpha_{ij}^t \mathbf{d}_j^{t-1} + B_1 \mathbf{h}_i^{t-1}\right) \\ \mathbf{d}_i^t = g\left(W_2 \sum_{j \in N(i)} \alpha_{ji}^t \mathbf{h}_j^{t-1} + B_2 \mathbf{d}_i^{t-1}\right) \end{cases} \quad (7)$$

식 (6)은 각 GNN 층 t에서의 지배소에 대한 표상 $\mathbf{h}_i^t$을 이용하여 GrandParent(조부모)에 대한 정보를 반영하고 의존소에 대한 표상 $\mathbf{d}_i^t$을 이용하여 GrandChild(손자)에 대한 정보를 반영하고. 식 (7)은 같은 지배소를 공유하는 Sibling(형제) 노드의 정보를 반영하는 수식을 표현하고 있다.

식 (8)은 (6), (7)를 결합한 형태로 형제(Sibling), 조부모(GrandParent), 손자(GrandChild)를 모두 업데이트 하는 방식이며 식 (9)는 식 (8)의 비동기(asynchronized) 방식으로 먼저 업데이트 된 지배소를 이용하여 다음 의존소를 업데이트 하는 방식이다.

$$\begin{cases} \mathbf{h}_i^t = g\left(W_1 \sum_{j \in N(i)} \left(\alpha_{ji}^t \mathbf{h}_j^{t-1} + \alpha_{ij}^t \mathbf{d}_j^{t-1}\right) + B_1 \mathbf{h}_i^{t-1}\right) \\ \mathbf{d}_i^t = g\left(W_2 \sum_{j \in N(i)} \left(\alpha_{ij}^t \mathbf{h}_j^{t-1} + \alpha_{ji}^t \mathbf{d}_j^{t-1}\right) + B_2 \mathbf{d}_i^{t-1}\right) \end{cases} \quad (8)$$

$$\begin{cases} \mathbf{h}_i^{t-\frac{1}{2}} = g\left(W_1 \sum_{j \in N(i)} \left(\mathrm{a}_{ji}^t \mathbf{h}_j^{t-1} + \mathrm{a}_{ij}^t \mathbf{d}_j^{t-1}\right) + B_1 \mathbf{h}_i^{t-1}\right) \\ \mathbf{d}_i^t = g\left(W_2 \sum_{j \in N(i)} \left(\mathrm{a}_{ij}^t \mathbf{h}_j^{-\frac{1}{2}} + \mathrm{a}_{ji}^t \mathbf{d}_j^{t-1}\right) + B_2 \mathbf{d}_i^{t-1}\right) \end{cases} \quad (9)$$

여러층의 GNNs을 거친 출력층 τ에서의 최종 지배소와 의존소의 표상을 각각 $\mathbf{h}_j^\tau$, $\mathbf{d}_i^\tau$로 표현하며 이를 Biaffine 어텐션을 적용한 가 최종 트리에 대한 점수가 된다. 학습 과정에서는 각 층에서의 손실(loss)을 layer-wise loss를 표현하여 각 층에서의 layer-wise loss를 모두 더한 값을 L_2라 하고 최종 층 에서의 의존소와 의존관계에 대한 손실의 합을 L_1하며 최종 손실은 다음과 같이 표현된다.

$$L = \lambda_1 L_1 + \lambda_2 L_2$$

위 손실 수식에서 하이퍼 파라미터 λ_1, λ_2를 각각 [5]에서와 같이 1, 0.5로 설정하였으며 layer-wise loss에서는 의존 관계에 대한 loss는 포함하지 않는다.

4. 실험

4.1 실험 셋팅

본 연구에서는 [6-9]과 동일한 세종 구문 분석 데이터로 학습셋 53,842 문장과 평가셋 5,817 문장셋으로 구성되어 있다. 국어 정보 처리 시스템 경진대회의 1,500 문장을 최종 평가셋으로 하여 [10]과 동일하게 별도의 개발셋 없이 학습하였다. UAS(Unlabeled Attatch Score), LAS(Labeled Attatch Score)를 평가지표로 사용하며 BERT 모델은 [10]와 동일한 ETRI BERT 모델을 사용하였으며 논문에서 사용한 모델의 하이퍼 파라미터는 다음 표와 같다.

표 1. 모델의 하이퍼 파라미터

Hyper-parameter	value
형태소 임베딩	100
품사 임베딩	50
BERT 임베딩	768
Dropout rate	0.33
Char RNN Hidden Dimension	512
RNN Hidden Dimension	512
RNN Layers	3
GNN Layers	2
Optimizer	Adam
학습률	0.001
활성 함수	gelu

4.2 실험 결과

베이스라인 모델로는 [7]과 동일한 Deep Biaffine 모델을 사용하였으며 GNN에 사용되는 파라미터를 제외하고 모든 파라미터를 제안 모델과 동일하게 설정하였으며 기존 Deep Biaffine 모델과 식 (6), (7), (8), (9)의 4종류의 GNN 기반 모델의 비교 성능을 제시한다.

표 2. Deep Biaffine GNNs 모델 실험 결과

	UAS	LAS
Deep Biaffine 모델	91.85	89.72
Deep Biaffine + GNN(6)	91.85	89.63
Deep Biaffine + GNN(7)	91.75	89.60
Deep Biaffine + GNN(8)	91.92	**89.76**
Deep Biaffine + GNN(9)	**91.93**	89.74
[BERT] Deep Biaffine 모델	94.40%	92.45%
[BERT] Deep Biaffine + GNN(6)	94.38%	92.41%
[BERT] Deep Biaffine GNN(7)	94.40%	92.46%
[BERT] Deep Biaffine GNN(8)	94.43%	92.51%
[BERT] Deep Biaffine + GNN(9)	**94.44%**	**92.55%**

표 2에서 보듯이 식 (6), (7)의 방식은 기존 Biaffine 모델에 비해 낮은 성능을 보여주고 있으나 모든 High Order 정보를 고려하는 (8), (9) 방식에서는 기존 Biaffine 모델에 비해 UAS : 0.08, LAS : 0.04 높은 성능을 보여주고 있으며 BERT 인코더 모델에서는 UAS : 0.04, LAS : 0.1 가량 높은 성능을 보이고 있다.

표 3. 기존 모델과의 비교

	UAS	LAS
Deep Biaffine[7]	91.78%	89.76%
멀티헤드 + ELMo [8]	92.85%	90.65%
Biaffine + BERT small + Elmo [9]	93.46%	92.85%
Biaffine + BERT [10]	94.04%	92.00%
Arc-Hybrid 전이 기반	91.36%	89.17%
BERT Arc-Hybrid 전이 기반	94.19%	92.12%
GNN Biaffine	91.93%	89.74%
BERT GNN Biaffine	94.44%	92.55%
Dual Decomposition	**94.49%**	**92.61%**

위의 표 3은 기존 연구, 그리고 Arc-Hybrid[11] 전이 기반 모델과의 성능을 비교한다. 현재 세종 구문 분석 셋에서 가장 높은 성능을 보이고 있는 [9] Biaffine + BERT 모델에 비해 UAS : 0.4%, LAS : 0.55% 높은 성능을 보이고 있으며 여기에 BERT Arc-Hybrid 전이 기반 모델과 Dual Decomposition[13] 으로 결합하여 UAS : 0.04%, LAS : 0.06%의 추가적인 성능 향상을 달성하였다.
.

5. 결론

본 논문에서는 BERT 기반 인코더와 기존의 Deep Biaffine 어텐션 모델에 인접한 노드로부터 High-Order 정보를 학습할 수 있는 GNNs을 적용하여 세종 구문 분석 데이터 셋에서 기존 최고 성능에 비해 성능을 달성하였다.향후 연구로는 BERT이외에 RoBERTa, XLNet등 다양한 언어모델을 적용할 예정이다.

참고문헌

[1] Peters, Matthew E., et al. "Deep contextualized word representations." arXiv preprint arXiv:1802.05365 (2018).

[2] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[3] Dozat, Timothy, and Christopher D. Manning. "Deep biaffine attention for neural dependency parsing." arXiv preprint arXiv:1611.01734 (2016).

[4] Petar Velikovi, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Li, and Yoshua Bengio. 2018. Graph attention networks. In International Conference on Learning Representations.

[5] Ji, Tao, Yuanbin Wu, and Man Lan. "Graph-based Dependency Parsing with Graph Neural Networks." Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019.

[6] Ma, X., Hu, Z., Liu, J., Peng, N., Neubig, G., & Hovy, E. (2018). Stack-pointer networks for dependency parsing. arXiv preprint arXiv:1805.01087.

[7] 나승훈, 이건일, 신종훈, 김강일, “Deep Biaffine Attention을 이용한 한국어 의존 파싱”, KCC 2017

[8] 박성식, 오신혁, 김홍진, 김시형, 김학수, “ELMo와 멀티헤드 어텐션을 이용한 한국어 의존 구문 분석”, HCLT 2018

[9] 홍승연, 나승훈, 신종훈 김영길, "BERT와 ELMO 문맥화 단어 임베딩을 이용한 한국어 의존 파싱”, KCC 2019

[10] 박천음, 이창기, 임준호, 김현기, "BERT를 이용한 한국어 의존 구문 분석", KCC 2019

[11] Kiperwasser, Eliyahu, and Yoav Goldberg. "Simple and accurate dependency parsing using bidirectional LSTM feature representations." Transactions of the Association for Computational Linguistics 4 (2016): 313-327.

[12] BERT를 이용한 한국어 자연어처리: 개체명 인식, 감성분석, 의존 파싱, 의미역결정

[13] Koo, Terry, et al. "Dual decomposition for parsing with non-projective head automata." Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2010.