Graph Convolutional Network 기반 집합적 개체 연결

이영훈○, 나승훈
전북대학교
dldudgns73@jbnu.ac.kr, nash@jbnu.ac.kr

Graph Convolutional Networks for Collective Entity Linking

Young-Hoon Lee○, Seung-Hoon Na
Jeonbuk National University

요 약

개체명 연결이란 주어진 문장에 출현한 단어를 위키피디아와 같은 지식 기반 상의 하나의 개체에 연결하는 것을 의미한다. 문장에 나타나는 개체들은 주로 동일한 주제를 가지게 되는데 본 논문에서는 이러한 특징을 활용하기 위해서 개체들을 그래프상의 노드로 표현하고, 그래프 신경망을 이용하여 주변 노드의 정보를 통해 노드 표상을 업데이트한다. 한국어 위키피디아 링크 데이터를 사용하여 실험을 진행한 결과 개발 셋에서 82.09%, 평가 셋에서 81.87%의 성능을 보였다.

주제어: 개체명 연결, 그래프 신경망, 지식 기반

1. 서론

개체명(Named Entity)이란 문장에서 고유한 의미를 가지는 단어 또는 구절을 의미한다. 이러한 개체명은 여러 가지의 개체들을 의미할 수 있게 되는 중의성을 가지게 되는데, 이러한 중의성 문제를 해결하는 개체명 연결(Named Entity Linking)은 주어진 문장에 출현한 단어를 위키피디아[1]와 같은 지식 기반(Knowledge base) 상의 하나의 개체와 연결하여 특정 개체가 무엇인지 식별하는 것을 의미한다. 예를 들어 "거미는 대한민국 가수이다."라는 문장에서 "거미"가 "거미(가수)"를 의미하는지 거미(절지동물)"를 의미하는지 연결하는 작업을 말한다. 문장에 나타나는 개체들은 주로 어떠한 주제를 일관적으로 가리키게 되는데, 이러한 정보들을 얻기 위해서 주변의 문맥뿐만 아니라 동일 문장에 나타나는 다른 개체의 정보를 활용 할 수 있다. 또한 개체들은 그래프로 표현할 수 있는데 각 개체를 노드(Node)로, 개체들 간의 관계를 간선(Edge)으로 표현하게 된다.

본 논문에서는 주어진 문장의 Mention-Context 쌍 정보와 개체의 설명 정보를 이용하여 노드의 표현을 구성하고, 노드들 간의 그래프 정보와 주변 노드의 정보를 활용하여 노드의 표상을 업데이트함으로써 개체명 연결을 수행한다. 또한 주변 노드의 정보를 활용한 모델과 베이스라인 모델의 성능 비교를 통해 그래프 기반의 개체명 연결의 효과성을 보인다.

2. 관련 연구

개체명 연결은 지식 기반 확장이나 정보 추출, 질의응답 등의 여러 태스크에서 특징으로 사용되는 등 여러 자연언어 처리 분야에서 중요한 요소로 작용하고 있다.

개체명 연결을 해결하기 위해 Mention-Context와 개체 간의 유사도를 이용하여 랭킹 문제로 해결한 [2,3]와 같이 주변 문맥을 파악하여 특징을 추출하는 지역적 특징(Local Feature)을 이용한 방법이나, 문장 내 다른 개체들의 정보를 함께 사용하는 전역적 특징(Global Feature)을 이용한 방법 등이 시도되었다.

특히 [4]에서는 지식기반 상의 트리플 <개체1-관계-개체2> 정보를 이용하여 노드를 표현하였고, [5]는 문장 내의 개체들이 그래프상에서 일관된 주제로 연결되도록 하여 Collective 하게 학습하여 문제를 해결하는 등 개체를 노드로, 관계를 간선으로 표현하여 개체 모호성을 해결하는 시도도 있었다.

3. 모델

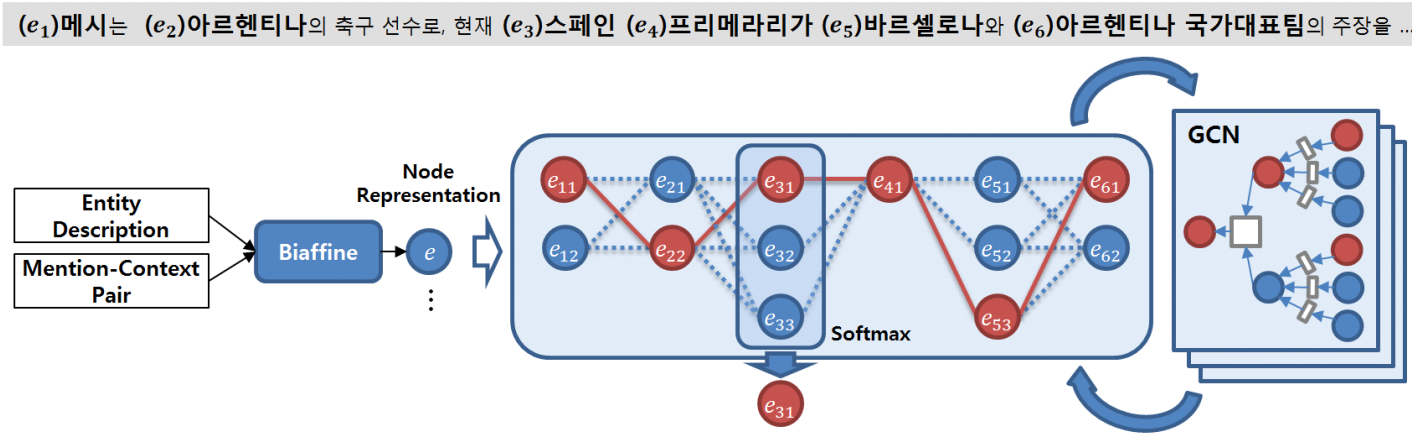
실험에 사용된 모델은 [그림 1]과 같다. 모델은 크게 개체를 노드로 표현하는 부분과 주변 노드의 정보를 활용하여 노드 표상을 업데이트하는 부분으로 나누어져 있다.

3.1 Node Representation

모델은 문장 단위의 입력으로 이루어져 있으며, 각 Mention에 해당하는 후보 개체들을 이용하여 각 개체의 설명 정보와 Mention-Context 정보를 Biaffine 연산을 통해 노드의 표상을 얻어내게 된다. Mention-Context는 [6]에서 사용한 span representation을 이용하여 구성하였고, 다음의 수식과 동일하다.

$$h_t^c = BiLSTM(c_t)$$
$$s_i = \left[h^c_{START(i)}; h^c_{END(i)}; w_i^{HeadAtt}; \phi(i)\right] \quad (1)$$

문서 $C = \{c_1, \ldots, c_n\}$를 GloVe를 통해 임베딩을 구성하게 되고 양방향 LSTM을 통해서 인코딩하게 된다. i는 전체 문장에 존재하는 Mention 중 i번째 Mention을 의미하고, $START(i)$와 $END(i)$는 각각 i번째 Mention의 시작점과 끝점을 나타낸다. $\phi(i)$는 span의 길이를 임베딩한 특징 벡터이다. 수식(1)에서 사용한 head-finding



[그림 1] Collective Entity Linking 모델의 전체 구조

attention은 다음과 같고 $FFNN$은 feed forward 네트워크를 의미한다.

$$\alpha_t = v_\alpha^T FFNN_\alpha(h_t)$$
$$s_{i,t} = \frac{\exp(\alpha_t)}{\sum_{k=START(i)}^{END(i)} \exp(\alpha_t)} \quad (2)$$
$$w_i^{HeadAtt} = \sum_{t=START(i)}^{END(i)} s_{i,t} w_t$$

이어서 각 개체의 개체 설명을 문서와 동일하게 GloVe를 통해 인코딩하고, 양방향 LSTM을 통해 구성하게 된다. 이때, 각 i번째 Mention에는 다중의 후보 개체가 존재하기 때문에 i번째 Mention의 j번째 개체 설명인 D_{ij}를 인코딩하고, Biaffine 연산을 통해 각 노드의 표현인 e_{ij}를 얻는다.

$$h_{ij}^d = BiLSTM(D_{ij})$$
$$e_{ij} = h_{ij}^d U s_i^T + b \quad (3)$$

3.2 GCN(Graph Convolution Network)

주변 노드의 정보를 활용하여 얻어진 노드 표현 e_{ij}를 업데이트하기 위하여 GCN[7]을 사용하게 된다. GCN은 그래프 구조의 데이터를 입력으로 받아 컨볼루션 연산과 유사하게 공유되는 파라미터를 통하여 특징을 추출하는 네트워크이다. 레이어가 높아질수록 더 넓은 범위의 주변 노드의 정보를 이용하여 업데이트하게 된다. 모델은 l번째 layer에 해당하는 각 노드 벡터 $H^{(l)}$와 인접 행렬(adjacency matrix)에 해당하는 $A^{(l)}$와 파라미터로 구성되어있다.

$$H^0 = e$$
$$H^{(l+1)} = \sigma(AH^{(l)}W^{(l)} + b^{(l)}) \quad (4)$$

Biaffine 연산을 통해 얻은 노드 표현 e가 첫 번째 레이어의 입력으로 들어가게 되고, σ는 비선형함수로 ReLU를 사용하였다. 최종적으로 업데이트된 각각 노드의 표현에 선형연산을 취하고 각 후보 개체 차원에 softmax를 통해 최종 점수를 얻게 된다.

4. 실험

4.1 학습 및 평가 데이터

실험에 사용된 데이터는 한국어 위키피디아의 문서 정보를 이용하였다. 위키피디아의 문서 중 개체가 링크(Hyper-Link)되어 있는 문서를 사용하였으며, 개체 표현이 포함되어 있는 문장을 Context로, 개체의 표현을 Mention으로 사용하였다. 개체의 설명은 위키피디아의 가장 첫 문장에 해당하며 주로 개체의 축약적인 정보들을 가지고 있는 문장이다. 학습 셋과 검증 셋, 테스트 셋은 각각 10만 개, 1만 개, 2만 개를 사용하였다. 동일한 Mention을 가지는 개체들을 통해 개체 사전을 구축하고 Mention에 대해서 후보 개체를 구성하였다.

4.2 확장 모델

• **Model 1.** Model 1은 Mention-Context 쌍의 Span 정보만을 이용하여 Mention 표상을 얻고 Biaffine 연산을 통해 노드 표현을 구성한 모델이다. 이때, $score_{ij}$ 는 i번째 Mention의 j번째 후보 개체의 점수를 의미한다.

$$s_i = [h_{START(i)}^c; h_{END(i)}^c]$$
$$score_{ij} = softmax(Biaffine(s_i, h_{ij}^d))$$

• **Model 2.** Mention-Context 쌍의 Span 정보와 Head-Finding Attention, Span 길이 임베딩을 이용하여 Mention 표상을 얻고 Biaffine 연산을 통해 노드 표현을 구성한 모델이다.

$$s_i = [h_{START(i)}^c; h_{END(i)}^c; w_i^{HeadAtt}; \phi(i)]$$
$$score_{ij} = softmax(Biaffine(s_i, h_{ij}^d))$$

• **Model 3.** Biaffine 연산을 통해 얻은 노드 표현을 단순 Neighborhood Aggregation[8]을 통해 노드를 업데이트하여 최종 노드 표현을 얻는 모델이다. 여기서 $H^{(l)}$는 GCN의 l번째 레이어를 의미하고, A_{diag}는 대각 성분 1이고, 이외의 모든 성분이 0인 행렬인 대각행렬(diagonal matrix)를 의미한다.

$$H^{(l+1)} = \sigma(\frac{(A - A_{diag})H^{(l)}W^{(l)}}{|N|} + A_{diag}H^{(l)}B^{(l)} + b^{(l)})$$

• **Model 4.** 섹션 3에서 설명한 모델에 해당하며, Biaffine 연산을 이용해 얻은 노드 표현을 GCN을 통해 주변 노드의 정보를 활용해 업데이트하는 모델이다.

$$H^{(l+1)} = \sigma(AH^{(l)}W^{(l)} + b^{(l)})$$

4.3 실험 평가

개체 사전으로부터 실험 데이터의 Mention에 해당하는 정답을 포함한 5개의 개체를 이용하여 후보 개체를 구성하였다. 개체 사전의 개체가 5개 미만일 경우, 부족한 후보 개체만큼 랜덤으로 개체를 선택하여 후보를 구성하였다. 후보 개체 중에서 정답 개체의 점수가 가장 높은 값을 가지게 되면 정답으로 가정하고 이에 대한 정확도(Accuracy)를 측정하였다.

표 1. 각 확장 모델의 정확도

모델	개발 셋	평가 셋
Model 1	71.00	70.65
Model 2	73.77	73.24
Model 3	81.39	81.01
Model 4	**82.09**	**81.87**

[표 1]은 각 확장 모델별 성능을 나타낸다. 단순 baseline 모델에 head-finding attention을 적용하였을 때, 약 2.5%의 성능 향상을 보였고, 주변 노드의 정보를 이용해 노드를 업데이트한 Model 3과 Model 4는 그렇지 않은 Model 1과 Model 2와 비교하였을 때, 향상된 성능을 보여주었다.

5. 결론

본 논문에서는 한국어 위키피디아의 링크를 이용하여 개체 연결 데이터를 구축하고 실험에 사용하였다. Mention-Context 쌍과 개체 설명의 Biaffine 연산을 통해 노드의 표상을 얻고, GCN을 이용하여 주변 노드의 정보까지 활용하여 노드의 표상을 업데이트하여 성능이 향상됨을 보였다. 추후 연구에서는 두 노드 간의 Relation을 활용하여 노드 표현을 구성하고, 후보 개체에 정답 개체가 존재하지 않는 즉, NIL을 해결하는 개체 연결 모델을 연구할 예정이다.

참고문헌

[1] https://ko.wikipedia.org

[2] Sun, Y., Lin, L., Tang, D., Yang, N., Ji, Z., & Wang, X., Modeling mention, context and entity with neural networks for entity disambiguation., In Twenty-Fourth International Joint Conference on Artificial Intelligence., 2015

[3] 이영훈, 나승훈, "위키피디아 링크 데이터를 이용한 Neural Network Model 기반 한국어 개체명 연결", 제30회 한글 및 한국어 정보처리 학술발표 논문집, 2018.

[4] Cetoli, A., Akbari, M., Bragaglia, S., O'Harney, A. D., & Sloan, M., Named Entity Disambiguation using Deep Learning on Graphs., 2018

[5] Cao, Y., Hou, L., Li, J., & Liu, Z., Neural collective entity linking., 2018

[6] Zhang, R., Santos, C. N. D., Yasunaga, M., Xiang, B., & Radev, D., Neural coreference resolution with deep biaffine attention by joint mention detection and mention clustering., 2018

[7] Kipf, T. N., & Welling, M., Semi-supervised classification with graph convolutional networks., 2016

[8] Hamilton, W. L., Ying, R., & Leskovec, J. Representation learning on graphs: Methods and applications., 2017