동적 어텐션을 이용한 지식그래프 임베딩

황민성 [1] 황지영 [2‡]
[1] 한국과학기술원 전기및전자공학부 [2] 한국과학기술원 전산학부
hminsung@kaist.ac.kr, jjwhang@kaist.ac.kr

Knowledge Graph Embedding with Dynamic Attention

Minsung Hwang[1] Joyce Jiyoung Whang[2‡]
[1]Electrical Engineering, KAIST [2]School of Computing, KAIST

요 약

지식그래프(Knowledge Graph)는 우리가 알고 있는 지식들을 방향성이 있는 그래프로 표현한 것이다. 지식그래프는 여러 개체(Entity)들과 그 개체들 사이의 관계(Relation)로 이루어져 있는데, 이러한 지식그래프 상에서 다양한 문제를 해결하기 위해서는 지식그래프 임베딩(Embedding) 기법을 통해 개체와 관계들을 연속적인 공간의 벡터로 나타내야 한다. 최근 연구에 따르면, 지식그래프가 아닌 일반 그래프의 임베딩 모델 중, 어텐션(Attention)을 기반으로 한 그래프 신경망(Graph Neural Network) 모델의 경우, 정적(Static) 어텐션 문제가 발생할 수 있음이 밝혀졌다. 본 논문에서는 그래프 신경망을 사용한 어텐션 기반의 지식그래프 임베딩에서도 정적 어텐션 문제가 발생하는지 확인하고, 이 문제를 해결하기 위해 동적(Dynamic) 어텐션 기법을 도입한다.

1. 서 론

지식그래프는 우리가 알고 있는 여러가지 지식들을 방향성이 있는 그래프로 표현한 것이다. 지식 그래프는 여러 지식들의 개체와 관계들을 노드와 링크를 가진다. 지식그래프는 각 사실관계를 삼중항(Triplet) 으로 나타낸다. 각 삼중항은 시작 개체(Head entity)와 도착 개체(Tail entity), 그리고 두 개체 사이의 관계로 이루어져 있다. 지식그래프 임베딩은 이러한 각각의 개체, 관계들을 연속적인 공간의 벡터로 표현하는 방법들을 의미한다. 지식그래프 임베딩을 통해 두 개체 사이의 관계나 삼중항의 사실관계 확인 등의 문제를 해결할 수 있다. 따라서 지식그래프를 적절하게 임베딩하는 것은 중요하고, 필수적인 과정이다.

지식그래프의 임베딩을 구하는 방법은 거리 환산 방법(Translation distance), 의미 일치 방법(Semantic matching) 등 여러가지가 있지만, 본 논문에서는 그래프 신경망을 사용한 어텐션 기반의 지식그래프 임베딩 기법을 개선하는 방법을 사용하였다.

일반 그래프 임베딩 모델의 경우, 어텐션 기반의 그래프 신경망[1]은 이웃 노드들로 부터 임베딩을 취합할 때, 각각의 이웃 노드에 가중치를 부여하여 취합한다. 일반 그래프에서 뿐만 아니라, 지식그래프에서도 어텐션 기법을 적용하여 임베딩을 진행 할 수 있다. 가장 잘 알려진 방법중 하나는 지식그래프 삼중항의 두 개체의 임베딩과 관계의 임베딩을 모두 한 줄로 연결(Concatenate)하여 어텐션을 계산하는 방법[3]이다. 이 경우, 일반 그래프에서 얻어낸 어텐션과 유사한 어텐션을 얻어 낼 수 있다.

이러한 고전적인 어텐션 기법은 충분히 효과적이지만, 일부 상황에서는 정적 어텐션 문제로 인해 효과가 비교적 떨어지는 상황이 발생한다. 정적 어텐션 문제란, 일반적인 그래프 어텐션 기법을 통해 얻어진 어텐션의 순위가 일정하게 되는 문제이다. 이 정적 어텐션 문제가 큰 문제가 되지 않는 경우도 있지만, 많은 경우에 대부분의 노드에서 특정 노드의 정보를 가장 강하게 취합하게 되어, 간단한 예측조차 틀리게 예측하는 상황이 발생한다.

정적 어텐션 문제를 해결하기 위해 고안된 것이 동적 어텐션 기법[2]이다. 기존 어텐션 기법에서 비선형 함수의 위치를 이동하는 동적 어텐션 기법을 통해 어텐션을 계산할 수 있다. 이 경우, 어텐션의 순위가 달라지게 되어 정적 어텐션 문제가 발생하지 않는다.

앞서 언급한 어텐션 기반의 지식그래프 임베딩 기법 또한 그래프 어텐션 기법과 유사한 어텐션 기법을 사용하기 때문에, 정적 어텐션 문제가 발생할 것으로 예상하였다. 따라서 본 논문에서는 어텐션 기반의 지식그래프 임베딩에서도 정적 어텐션 문제가 발생하는지 확인함과 동시에, 동적 어텐션 기법을 통해 성능이 향상되는지 확인한다.

2. 표기법

단방향 그래프 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$는 노드의 집합 $\mathcal{V}$와 링크의 집합 $\mathcal{E}$으로 구성된 그래프이다. 링크 집합 $\mathcal{E}$의 원소

‡ 교신저자 (Corresponding Author)
이 성과는 과학기술정보통신부의 재원으로 한국연구재단의 지원(2022R1A2C4001594)과 정보통신기획평가원의 지원(No. 2022-0-00369, (4세부) 전문지식 대상 판단결과의 이유/근거를 설명가능한 전문가 의사결정 지원 인공지능 기술개발)을 받아 수행된 연구임.

(j,i)는 노드 j에서 노드 i 로의 링크를 의미한다. 본 논문에서 이웃 노드 $\mathcal{N}_\mathrm{i}$는 노드 i 로 향하는 링크를 가지고 있는 노드의 집합이다.

지식그래프는 $\mathcal{G}_{\mathrm{kg}} = (\mathcal{V}, \mathcal{R}, \mathcal{E})$ 로 표기한다. $\mathcal{V}$와 $\mathcal{R}$ 은 각각 지식그래프의 개체, 관계의 집합이다. 지식그래프의 삼중항은 $(\mathrm{e_h}, \mathrm{r}, \mathrm{e_t}) \in \mathcal{E}$ 로 나타낸다. 각 그래프에서의 임베딩 벡터들은 볼드체로 표기한다. 노드 임베딩 $\mathbf{h_i}$, 개체 임베딩 $\mathbf{e_h}, \mathbf{e_t}$, 관계임베딩 $\mathbf{r}$ 으로 표현한다. 볼드체로 표현된 대문자 $(\mathbf{W})$ 는 선형 레이어에서 가중치(Weight)를 나타내는 행렬이다. 볼드체로 표현된 소문자$(\mathbf{a})$는 벡터를 의미한다. $\mathbf{a}\cdot\mathbf{b}$는 두 벡터 a,b의 내적을 의미한다.

3. 관련 연구

3.1 그래프 신경망(Graph Neural Network)

그래프 신경망 기법은 그래프 내 노드들의 임베딩을 계산할 때, 이웃 노드들의 임베딩을 취합하여 자신의 임베딩을 업데이트 하는 기법이다. 여기서 이웃 노드들의 임베딩을 취합하는 방법에 따라 그래프 신경망 기법의 종류가 나누어 진다. 식 (1)은 주변노드들의 임베딩을 취합하여 업데이트하는 과정을 나타낸다.

$$\mathbf{h'}_\mathrm{i} = \sigma(\boldsymbol{h}_i, aggregate(\{\boldsymbol{h}_j | j \in \mathcal{N}_i\}) \cdots (1)$$

σ는 비선형 활성화 함수를 의미한다. $aggregate$는 집합의 임베딩들을 취합해주는 함수이다.

3.2 그래프 어텐션 네트워크(GAT)

그래프 어텐션 네트워크[1]에서는 이웃 노드의 임베딩을 취합할 때, 각 노드의 어텐션을 계산하여 다른 가중치로 취합한다. 식 (2) 는 그래프 어텐션 네트워크에서 어텐션을 계산하는 방법이다.

$$e(\boldsymbol{h}_i, \boldsymbol{h}_j) = \sigma(\boldsymbol{a}^T \cdot \boldsymbol{W}[\boldsymbol{h}_i || \boldsymbol{h}_j]) \cdots (2)$$

이 어텐션 값은 소프트맥스(Softmax) 함수를 통해 모든 이웃에 대해 표준화(Normalize) 된다.

$$\alpha_{ij} = softmax_j\left(e(\boldsymbol{h}_i, \boldsymbol{h}_j)\right) \cdots (3)$$

이렇게 일반화된 어텐션 값을 이용해 이웃 노드의 임베딩을 취합하여 임베딩을 업데이트 한다.

$$h'_i = \sigma(\sum_{j \in N_i} \alpha_{ij} \cdot \boldsymbol{W}\boldsymbol{h}_j) \cdots (4)$$

3.3 동적 그래프 어텐션 네트워크(GATv2)

3.2에서 설명한 그래프 어텐션 네트워크에서는 각 노드에 대해 계산된 어텐션 값이 일정한 순위를 가지는 정적 어텐션 문제가 발생한다.

정적 어텐션 문제를 해결하기 위해 고안된 모델이 동적 그래프 어텐션 네트워크[2]이다. 정적 어텐션 문제는 식 (1)에서 두 개의 선형변환 층이 연속적으로 계산되면서 발생한다. 동적 그래프 어텐션 네트워크에서 계산하는 어텐션은 3.2에서 계산한 어텐션에서 비선형 함수 밖으로 한개의 선형 레이어를 이동하여 계산한다. 식 (5)는 동적 어텐션을 계산하는 방법이다.

$$e(\boldsymbol{h}_i, \boldsymbol{h}_j) = \boldsymbol{a}^T \cdot LeakyReLU(\boldsymbol{W}[\boldsymbol{h}_i || \boldsymbol{h}_j]) \cdots (5)$$

이후 그래프 어텐션 네트워크과 똑같이 소프트맥스 함수를 통해 표준화 해준 뒤, 임베딩을 취합한다.

3.4 지식그래프 임베딩

지식그래프 임베딩은 지식그래프의 각 개체, 관계를 연속적인 공간의 벡터로 표현하는 것이다. 벡터로 표현된 개체와 관계들을 이용해 개체 사이의 관계 예측, 삼중항의 사실관계 확인 등에 사용할 수 있다.

지식그래프 임베딩은 크게 세가지 과정으로 진행된다. (i)개체와 관계를 벡터로 표현한다. (ii) 스코어를 정의한다. (iii) 개체, 관계 임베딩을 학습한다. 지식그래프 임베딩 기법에는 거리 환산, 의미 일치, 그래프 신경망 등 여러 종류가 있다.

3.5 어텐션 기반 지식그래프 임베딩(KBAT)

어텐션 기반 지식그래프 임베딩은 그래프 신경망 기반의 지식그래프 임베딩 방법 중 하나이다. 이 모델에서는 인코딩 과정에서 개체의 임베딩을 업데이트할 때, 개체가 포함된 모든 삼중항의 어텐션을 계산한다. 삼중항의 어텐션은 다음과 같이 계산할 수 있다.

$$\boldsymbol{c}_{ijk} = \boldsymbol{W}_1[\boldsymbol{e_h} || \boldsymbol{e_t} || \boldsymbol{r}] \cdots (6)$$

$$b_{ijk} = LeakyReLU(\boldsymbol{w}_2 \boldsymbol{c_{ijk}}) \cdots (7)$$

$$\alpha_{ijk} = softmax_{jk}(b_{ijk}) \cdots (8)$$

그래프 어텐션 네트워크와 유사하게, 계산된 어텐션을 통해 각 개체의 임베딩을 업데이트 한다. 어텐션 레이어를 두 번 적용하여 최종 임베딩을 얻어내고, 스코어를 정의하여 학습을 진행한다.

4. 동적 어텐션 기법의 지식그래프 임베딩 적용

동적 그래프 어텐션 네트워크에서 사용한 방법을 어텐션 기반 지식그래프 임베딩 기법에 동일하게 적용하여 모델의 어텐션 층을 수정하여 동적 어텐션이 적용된 KBATv2모델을 생성한다.

어텐션 기반 지식그래프 임베딩 모델의 식에서 어텐션 계산 파트를 수정하여, 두 개의 선형 층 사이에 비선형 함수인 LeakyReLU 함수를 적용하여준다.

$$\boldsymbol{c}_{ijk} = \boldsymbol{W}_1[\boldsymbol{e_h} || \boldsymbol{e_t} || \boldsymbol{r}] \cdots (9)$$

$$b_{ijk} = \boldsymbol{w_2} LeakyReLU(\boldsymbol{c}_{ijk}) \cdots (10)$$

$$\alpha_{ijk} = softmax_{jk}(b_{ijk}) \cdots (11)$$

결과적으로 두개의 선형변환 층 $\boldsymbol{W_1}, \boldsymbol{w_2}$가 분리된다.

5. 실험

모델의 성능 비교를 위해 지식그래프 데이터에서 링크예측(link prediction)실험을 진행하였다. 지식그래프

내의 삼중항들에서 시작 개체를 다른 개체로 대체하거나, 도착 개체를 다른 개체로 대체하여 각 경우에 대해 Hits@10, Mean Rank(MR), Mean Reciprocal Rank(MRR)를 계산하고 두 결과를 평균내었다.

Hits@10은 예측된 스코어로 줄을 세웠을 때, 상위 10 개의 점수를 가진 예측 삼중항중에 정답이 존재하는 비율을 말한다. Mean Rank는 샘플 삼중항들의 순위를 계산하고, 순위들의 평균을 낸 것이다. Mean Reciprocal Rank는 각 샘플 삼중항들의 순위 중, 가장 높은 순위의 역수의 평균을 구한 것이다. 따라서 Hits@10과 MRR은 높을수록 높은 성능을 의미하고, MR은 낮을수록 높은 성능을 의미한다. 모든 실험은 필터링된(filtered) 스코어로 계산되었다.

표 1 FB15K-237 통계표

Entity	Relation	Number of triplets		
		Train	Valid	Test
14,541	237	272,155	17,535	20,466

표 2 FB15K-237 모델 별 하이퍼파라미터

Model	Learning rate	LeakyRelu alpha	Weight decay
KBAT	**0.001**	0.3	**0.00001**
KBATv2	{**0.001**, 0.005}	{0.3, 0.5, **0.7**}	{**0.00001**, 0.000005}

5.1 실험 데이터

결과 비교를 위해서 실제 지식그래프 데이터 셋인 FB15K-237를 사용하였다. 데이터 셋에 대한 정보는 표 1과 같다. 표 2는 실험에 사용한 각 하이퍼파라미터(Hyperparameter)를 나타낸 것이다. 하이퍼파라미터는 실험데이터에서 검증(Validation) 데이터를 분리하여, 앞서 언급한 세개의 메트릭 (metrics)을 계산하였다. 이후 각 메트릭에서 최고 성능과 비교한 이득(gain)을 계산하여 세개의 이득의 합을 얻어내었고, 이 총 이득이 가장 높은 경우의 하이퍼파라미터를 사용하였다.

실험 결과를 얻어낸 후, 기존 임베딩 모델 KBAT과 결과를 비교하였다. 결과비교에 사용한 식은 다음과 같다. sign함수는 metric이 hits@10, MRR일때는 +1 이고, MR일 때는 -1을 가지는 함수이다[5].

$$Gain(metric) = sign(metric)\frac{score(KBATv2) - score(KBAT)}{score(KBAT)} * 100$$

5.2 실험 결과

실제로 비선형 함수 밖으로 한개의 선형 레이어를 이동시켜서 생성한 KBATv2모델의 경우 기존 모델에 비해서 높은 성능을 나타내었다. FB15K-237의 경우 어텐션의 순위가 일정한 정적 어텐션 문제가 영향을 끼쳤음을 확인 할 수 있고, 동적 어텐션 기법을 적용함으로써 성능을 향상시킬 수 있음을 확인하였다.

표 3 링크예측 결과. 메트릭 별 이득 및 총 이득

	FB15K-237		
metric	Hits@10	MR	MRR
Standard KBAT	0.6215	206.7	0.4640
KBATv2	0.6296	177.9	0.4585
Gain	1.287%	13.93%	-1.185%
Total gain	14.03%		

6. 결론 및 향후 연구

본 논문에서는 어텐션 기반의 지식그래프 임베딩 기법에서 정적 어텐션 문제가 발생하는지 확인하고, 동적 어텐션 기법을 적용하여 성능이 개선되는지 확인하였다. 한 가지 데이터에 대해서는 성능이 개선됨을 확인하였으나, 다양한 데이터에 동일한 기법을 적용해보지 못하였기 때문에, 모든 경우에 정적 어텐션문제가 발생하고, 동적 어텐션 기법이 성능을 개선시키는지 확인하지 못했다.

향후 연구에서는 다양한 데이터에 동일하게 동적 어텐션 기법을 적용하여 실험을 해보고 정적 어텐션 문제가 발생하는지 확인해볼 수 있다. 또한 정적 어텐션 문제가 발생하는지 실제로 계산된 어텐션 값을 분석해볼 예정이다.

참고문헌

[1] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. “Graph Attention Networks,” *International Conference on Learning Representations*, 2018.

[2] S. Brody, U. Alon, and E. Yahav. “How Attentive are Graph Attention Network?,” *International Conference on Learning Representations*, 2022.

[3] D. Nathani, J. Chauhan, C. Sharma, and M. Kaul. “Learning Attention-based Embeddingsfor Relation Prediction in Knowledge Graphs,” *Association for Computational Linguistics*, 2019.

[4] Q. Wang, Z. Mao, B. Wang, and L. Guo, “Knowledge graph embedding: A survey of approaches and applications,” *IEEE Transactions on Knowledge and Data Engineering*, pp. 2724-2743, Sep. 2017.

[5] C. Chung and J. J. Whang. “Knowledge Graph Embedding via Metagraph Learning,” *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.