

최신 방향 네트워크 임베딩 방법들의 성능 평가

유현식*, 이연창*, 김상욱*

*한양대학교 컴퓨터소프트웨어학과

hsyoo32@hanyang.ac.kr, lyc0324@hanyang.ac.kr, wook@hanyang.ac.kr

Evaluation of Directed Network Embedding Methods

Hyunsik Yoo*, Yeon-Chang Lee*, Sang-Wook Kim*

*Dept. of Computer and Software, Hanyang University

요 약

방향 네트워크 임베딩 문제는 주어진 방향 네트워크의 노드들을 그들 간의 비대칭 관계를 보존할 수 있는 저 차원 벡터들로 표현하는 것이다. 최근, 이 문제에 대한 다양한 방법들이 제안되어왔다. 본 논문에서 우리는 네 가지 실세계 방향 네트워크 데이터셋과 세 가지 에지 예측 시나리오를 이용한 실험을 통해, 최신 방향 네트워크 임베딩 방법들의 성능을 종합적으로 비교 분석한다.

1. 서론

네트워크 임베딩은 주어진 네트워크의 노드들을 저 차원의 벡터들 (이하 임베딩) 로 표현하고, 그 임베딩들에 네트워크의 구조적 특성 (예: proximity) 이 반영되도록 학습하는 것을 목표로 한다. 이러한 임베딩들은 에지 예측, 노드 분류 등의 다양한 기계 학습 애플리케이션을 수행하는 데 유용한 특징으로 활용될 수 있다 [1].

본 논문에서, 우리는 에지 방향을 활용하는 네트워크 임베딩에 초점을 맞춘다. 예를 들어, 인스타그램에서 i 가 인플루언서 j 를 팔로우하더라도, j 는 i 의 존재조차 몰라서 i 를 팔로우하지 않을 수 있다. 이와 같은 노드들 간의 비대칭 관계 (asymmetric relationship) 는 네트워크에서 i 가 j 를 에지로 가리키고 j 는 i 를 에지로 가리키지 않는 것으로 표현될 수 있다.

이러한 노드들 간의 비대칭 관계를 임베딩에 반영하기 위해 다양한 네트워크 임베딩 (이하 방향 네트워크 임베딩) 방법들이 제안되어 왔다. 대표적인 예로는 APP [2], ATP [3], NERD [4], GravityAE [5] 및 GravityVAE [5] 가 있다. 대부분의 방향 네트워크 임베딩 방법들은 노드 i 에서 노드 j 로의 방향 에지들이 주어지면, 각 에지에서 노드들의 역할에 따라 다음과 같이 구분한다: (1) i -- 특정 노드를 가리키는 소스 노드; (2) j -- 특정 노드에 의해 가리킴을 받는 타겟 노드. 그 후, 그 방법들은 각 노드에 대해 소스 노드로서의 특성을 보존하는 소스 임베딩과 타겟 노드로서의 특성을 보존하는 타겟 임베딩을 학습한다.

그러나, 우리는 이러한 방향 네트워크 방법들의 성능이 종합적으로 비교 분석되지 않고 있다는 것을 확

인했다. 따라서, 본 논문에서는 다양한 실세계 방향 네트워크 데이터셋 및 다양한 에지 예측 평가 시나리오를 활용한 실험들을 수행한다. 이를 통해, 우리는 에지 방향을 임베딩에 반영하기 위해 가장 효과적인 방법에 대해 논의하고자 한다.

2. 방향 네트워크 임베딩 방법들

방향 네트워크 임베딩 방법들은 크게 (1) 행렬 분해 기반 방법, (2) 딥러닝 기반 방법 및 (3) 랜덤 워크 기반 방법으로 분류될 수 있다. 먼저, 행렬 분해 기반 [3] 및 딥러닝 기반 [5] 방법들은 모든 노드 쌍들의 비대칭 근접성이 보존되도록 임베딩들을 학습한다. 구체적으로, 그 방법들은 소스 노드와 타겟 노드 간의 비대칭 근접성을 행렬의 형태로 나타낸다. 그 후, 행렬 분해 기술 (예: singular value decomposition) 혹은 딥러닝 기술 (예: graph autoencoder) 을 사용하여, 노드들 간 비대칭 근접성을 근사하는 임베딩들을 얻는다.

반면, 랜덤 워크 기반 방법들 [2, 4] 은 샘플링 된 노드 쌍들의 비대칭 근접성이 보존되도록 임베딩들을 학습한다. 구체적으로, 각 노드에 대해, 해당 (시드) 노드로부터 출발한 랜덤 워크가 방문한 다수의 노드들을 샘플링한다. 그 후, 각 시드 노드와 샘플링된 노드들 간의 비대칭 근접성을 최대화하는 동시에, 시드 노드와 무작위로 선택된 노드들 간의 비대칭 근접성을 최소화하는 임베딩들을 얻는다.

3. 실험

본 장에서는 앞서 언급한 다섯 가지 최신 방향 네트워크 임베딩 방법들 (APP [2], GravityAE [5],

GravityVAE [5], NERD [4] 및 ATP [3]) 간 성능 비교를 수행한다. 우리는 각 방법의 하이퍼 파라미터들에 대해 그리드 검색을 통해 찾은 최적의 값을 사용했다.

데이터셋. 우리는 네 가지 실세계 방향 네트워크 데이터셋들을 사용했다: Gnutella (GNU), Wiki-Vote, JUNG/Javax (JUNG) 및 Edinburgh Associative Thesaurus (EAT). <표 1>은 데이터셋들의 통계를 보여준다.

<표 1> 데이터셋 통계

데이터셋	GNU	Wiki-Vote	JUNG	EAT
노드 수	6,301	7,115	6,120	23,132
에지 수	20,777	103,689	50,535	312,320

실험 방법. 우리는 다양한 에지 예측 시나리오에서 각 임베딩 방법의 성능을 측정한다. 에지 예측의 목표는 각 임베딩 방법이 주어진 네트워크에서 일부 제거된 방향 에지들을 얼마나 정확하게 예측할 수 있는지를 평가하는 것이다. 평가를 위해, 주어진 네트워크의 에지들을 트레이닝 (80%) 및 테스트 (20%) 셋으로 분할한다. 그 후, 각 트레이닝/테스트 셋에 존재하는 에지들을 긍정 정답 (positive example) 으로, 이와 동일한 수의 무작위로 샘플링된 (주어진 네트워크에 존재하지 않는) 에지들을 부정 정답 (negative example) 으로 간주한다. 그 후, 트레이닝 셋에 대해 각 임베딩 방법을 수행하여 노드들의 임베딩들을 얻고, 이러한 임베딩들을 기반으로 로지스틱 회귀 분류기를 학습시킨다. 최종적으로, 학습된 분류기를 통해 테스트 셋의 각 에지가 긍정 정답인지 부정 정답인지 분류한다. 분류 정확도 측정을 위해, 우리는 area under curve (AUC) 를 이용한다.

더 나아가, 우리는 에지 방향을 고려하는 에지 예측 시나리오에서도 각 임베딩 방법의 성능을 평가한다. 구체적으로, 우리는 각 임베딩 방법이 주어진 방향 네트워크에 존재하는 각 에지의 방향을 얼마나 정확하게 예측할 수 있는지를 평가한다. 이를 위해, 우리는 긍정 정답과 반대 방향을 갖는 에지를 부정 정답으로 샘플링한다. 여기서, 우리는 긍정 정답과 반대 방향을 가진 부정 정답의 비율 (k %) 에 따라, 세 가지 에지 예측 시나리오를 구성한다: (1) $k=0$, (2) $k=50$, (3) $k=100$. 먼저, $k=0$ 시나리오는 앞서 언급한 에지 예측 시나리오와 동일하다. 다음으로, $k=100$ 시나리오는 단방향의 긍정 정답에 대해서만 반대 방향을 갖는 에지를 부정 정답으로 샘플링한다. 마지막으로, $k=50$ 시나리오는 일부 단방향의 긍정 정답과 반대 방향의 에지를 전체 부정 정답의 50%로 샘플링하고, 무작위의 존재하지 않는 에지를 나머지 50%로 샘플링한다.

실험 결과. <표 2>는 앞서 언급한 세 가지 에지 예측 시나리오에 대한 최신 방향 네트워크 임베딩 방법들의 성능을 보여준다. 결과를 요약하자면, 우리는 한 가지 방법이 다른 모든 방법들을 일관적으로 능가하는 경우는 없다는 것을 확인하였다. 다시 말해, 가장 성능이 좋은 방법은 데이터 셋 및 예측 시나리오에 따라 달라진다: Wiki-Vote 에서 $k=0$ 및 $k=50$ 시나리오와 JUNG 에서 모든 시나리오에서는 GravityVAE; GNU, JUNG 및 EAT 에서 $k=0$ 시나리오에서는 NERD; 나머

지 모든 경우에 대해서는 ATP.

ATP 의 경우, 모든 데이터셋에서 $k=0$, $k=50$ 그리고 $k=100$ 순으로 높은 정확도를 보인다. 이러한 결과는 ATP 로 얻은 임베딩은 노드 간 비대칭 근접성을 정확하게 보존한다. 모든 방향 네트워크 임베딩 방법들은 원래 에지 방향을 고려하도록 설계되었음에도 불구하고, 이러한 경향은 ATP 에서만 관찰된다.

<표 2> 다섯 가지 방법들의 에지 예측 정확도

데이터셋	평가 종류	APP	Gravity AE	Gravity VAE	NERD	ATP
GNU	$k=0$	0.617	0.634	0.723	0.773	0.758
	$k=50$	0.606	0.648	0.750	0.809	0.813
	$k=100$	0.634	0.710	0.822	0.851	0.877
Wiki-Vote	$k=0$	0.823	0.871	0.906	0.901	0.824
	$k=50$	0.676	0.878	0.905	0.890	0.891
	$k=100$	0.686	0.922	0.950	0.897	0.966
JUNG	$k=0$	0.939	0.946	0.955	0.955	0.951
	$k=50$	0.950	0.944	0.968	0.963	0.968
	$k=100$	0.930	0.976	0.991	0.979	0.990
EAT	$k=0$	0.772	0.836	0.839	0.864	0.855
	$k=50$	0.701	0.791	0.815	0.825	0.882
	$k=100$	0.630	0.838	0.851	0.802	0.915

4. 결론

방향 네트워크 임베딩 방법들은 노드들 간의 연결 정보뿐만 아니라 에지 방향 정보를 임베딩에 함께 반영한다. 본 논문에서 우리는 다양한 실세계 방향 네트워크 데이터셋을 이용한 실험을 통해, 최신 방향 네트워크 임베딩 방법들 APP, GravityAE, GravityVAE, NERD, ATP 의 성능을 다양한 에지 예측 평가 시나리오에서 종합적으로 비교 분석하였다. 실험을 통해, 우리는 데이터셋 및 예측 시나리오에 따라 가장 우수한 성능을 보이는 방법이 다르다는 것을 확인했다. 이러한 결과는 향후 일관적으로 우수한 정확도를 보일 수 방법에 대한 연구가 필요함을 보여준다.

Acknowledgement

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No.2018R1A5A7059549, No.2020R1A2B5B03001960)

참고문헌

- [1] Aditya Grover et al., "Node2vec: Scalable Feature Learning for Networks," In *Proc. of ACM KDD*, 2016.
- [2] Chang Zhou et al., "Scalable Graph Embedding for Asymmetric Proximity," In *Proc. of AAAI*, 2017.
- [3] Jiankai Sun et al., "ATP: Directed Graph Embedding with Asymmetric Transitivity Preservation," In *Proc. of AAAI*, 2019.
- [4] Megha Khosla et al., "Node Representation Learning for Directed Graphs," In *Proc. of ECML PKDD*, 2019.
- [5] Guillaume Salha et al., "Gravity-Inspired Graph Autoencoders for Directed Link Prediction," In *Proc. of ACM CIKM*, 2019.