# Similarity-based methods or conventional ones, which is better for graph embedding?

류진수 [1], Masoud Rehyani Hamedani [2], 김상욱 [3*]
[1] 한양대학교 인공지능학과 석사과정
[2] 한양대학교 컴퓨터소프트웨어학과 연구조교수
[3] 한양대학교 컴퓨터소프트웨어학과 교수

jinsu97@hanyang.ac.kr, masoud@hanyang.ac.kr, wook@hanyang.ac.kr

# Similarity-based methods or conventional ones, which is better for graph embedding?

Jin-Su Ryu[1], Masoud Rehyani Hamedani [2], Sang-Wook Kim[3*]
[1]Dept. of Artificial Intelligence, Hanyang University
[2,3]Dept. of Computer Science, Hanyang University

**요 약**

그래프 임베딩 방법은 그래프 구조를 이용하여 그래프의 노드를 저차원 임베딩 공간에서 벡터로 매핑하여 각 노드를 벡터로 표현하는 것을 목표로 한다. 다양한 방법들이 제시되었지만 기존의 방법들은 그래프에서 노드 간의 유사성을 잘 보존할 수 없어 다양한 기계 학습에 대해 부정확한 벡터를 생성하였다. 이러한 문제를 해결하기 위해 노드 사이의 유사성을 이용한 방법이 제안되었다. 본 논문에서, 우리는 여섯 가지 실세계 데이터셋을 사용하여 세 가지 기계 학습 작업시 그래프 임베딩 방법들의 성능을 비교하여 유사성 기반의 그래프 임베딩 방법의 우수성을 확인했다.

## 1. 서론

그래프 임베딩 방법(이하 임베딩 방법)은 그래프 구조를 이용하여 그래프의 노드를 저차원 임베딩 공간에서 벡터로 매핑하며 이를 통해 노드 간의 이웃 유사성, 의미 정보 및 커뮤니티 구조를 보존한다[1-2]. 이 벡터는 link prediction [1-2], graph reconstruction [3], node classification [4-5]을 비롯한 광범위한 다운스트림 기계 학습 작업에서 활용할 수 있다.

여러 문헌에서 다양한 임베딩 방법이 제안되었으며 DeepWalk [1], DWNS [6], Gravity [7], node2vec [2] 및 NetMF[8]와 같은 기존 임베딩 방법은 노드의 암시적 및 로컬 이웃 보기에만 의존하며, 전역 노드 간의 유사성을 고려하지는 않았다. 한편, 유사성 기반 임베딩 방법은 노드 간의 전역 유사성을 임베딩 프로세스에서 활용하는 새로운 관점을 도입했다. VERSE[4], SimNet [9] 및 FREDE [5]는 이 유사성을 임베딩 프로세스에서 활용하였다.

우리는 본 논문에서는 여섯 가지의 실세계 데이터셋과 세 가지의 기계 학습 작업에 대한 광범위한 실험을 통해 유사성 기반 임베딩 방법과 기존 임베딩 방법의 효과를 비교한다. 우리의 실험 결과는 후자의 방법들보다 전자의 방법들이 훨씬 능가한다는 것을 보여준다. 우리가 아는 한, 이것은 문헌에서 이 두 임베딩 기술의 첫 번째 광범위한 비교이다.

## 2. 임베딩 방법들

DeepWalk (DWK) [1]는 언어 모델링의 skip-gram 모델에서 영감을 얻었으며, 짧은 랜덤 워크를 활용하여 그래프를 정렬된 선형 노드 시퀀스로 변환한다. 대상 노드의 조건에 따라 이웃 노드의 벡터를 관찰할 가능성을 최대화한다. node2vec (N2V) [2]은 이전 노드를 다시 방문하고 새 노드를 방문하는 것이 모두 매개변수에 의해 제어되는 편향된 랜덤 워크를 적용하여 DeepWalk 를 확장한다. NetMF (NMF) [8]는 SVD (singular value decomposition)를 DeepWalk 의 저위 연결 행렬에 적용하여 벡터를 얻는다. DWNS (DNS) [6]는 적대적 훈련에 기반한 2 인용 게임 학습 프로세스를 통해 DeepWalk 의 견고성을 개선하려고 한다. Gravity (GRV) [7]는 Newton 의 만유인력 이론에서 영감을 받았으며 임베딩을 위해 그래프 변형 자동 인코더를 사용한다. 이러한 방법들은 임베딩시 노드의 암시적 및

로컬 이웃 보기를 활용한다. 그래서 그래프에서 노드 사이의 유사성을 잘 보존할 수 없으므로 다양한 기계 학습 작업에 대해 부정확한 벡터를 생성한다[4, 9].

이러한 문제를 완화하기 위해 유사성 기반 임베딩 방법은 노드의 로컬 이웃 보기를 고려하는 대신, 그래프의 전역 유사성을 임베딩 프로세스에서 활용하는 그래프 임베딩에 대한 새로운 관점을 도입했다. 유사성 기반 임베딩 방인 VERSE (VRS) [4]는 RWR 과 벡터 내적의 normalized softmax 를 각각 사용하여 얻은 임베딩 공간의 벡터와 그래프에서 노드의 유사성 분포 사이의 KL distance 를 최소화하려고 시도한다. SimNet (SNT) [9]의 개념은 VERSE 와 매우 유사하지만 그래프의 노드 유사성 점수를 포함하는 행렬에 SVD 를 적용한다. FREDE (FRD) [5]는 NetMF[8]에서 영감을 받았으며, SVD 및 FD 행렬 스케치를 그래프에서 노드 쌍의 대략적인 RWR 점수를 포함하는 행렬에 적용하여 유사성 행렬의 공분산을 보존하려고 시도한다.

## 3. 실험

**데이터셋.** Amazon [10]은 Amazon 웹 사이트에서 자주 공동 구매하는 제품의 그래프다. CoCit [4-5]은 데이터 마이닝, 데이터베이스 및 기계 학습 분야의 논문 인용 그래프다. Cora [11]는 컴퓨터 과학 논문의 인용 그래프다. 노드 레이블은 연구 주제를 나타낸다. DBLP [12]는 2006 년 이전에 발표된 데이터 마이닝 및 데이터베이스 분야 논문의 인용 그래프다. Linux [13]는 Linux 커널 소스 코드(버전 3.16)에서 만든 그래프로, 노드(즉, 파일) $u$에서 $v$로의 링크는 $u$이 $v$을 포함함을 나타낸다. Live[10]는 LiveJournal 웹사이트에 있는 블로거 간의 사회적 우정 그래프다. 우리는 <표 1>에 요약된 실세계 데이터세셋을 사용한다.

<표 1> 데이터셋 통계

| | \|V\| | \|E\| | #Labels | Type |
|---|---|---|---|---|
| Amazon | 30,000 | 62,642 | 71 | undirected |
| CoCit | 44,034 | 195,361 | 15 | directed |
| Cora | 23,166 | 91,500 | 70 | directed |
| DBLP | 21,177 | 124,065 | 11 | directed |
| Linux | 30,837 | 213,954 | - | directed |
| Live | 11,755 | 160,046 | 7,086 | undirected |

**기계 학습 작업.** 우리는 다음 세 가지 기계 학습 작업을 사용한다. Graph Reconstruction[3]에서는 인접 노드가 임베딩 공간에 가깝게 위치해야 한다는 사실을 기반으로 그래프의 인접 행렬을 재구성하려고 한다. Node Classification [4-5]에서는 노드의 레이블을 예측하는 것을 목표로 한다. 마지막으로 Link Prediction [4-5]에서 우리는 그래프에서 누락된 링크를 예측하는 것을 목표로 한다.

**Graph Reconstruction.** 임의의 노드 $v$에 대해 $v$의 벡터와 다른 노드의 벡터의 내적을 계산하고 해당 결과를 내림차순으로 노드를 정렬한 후, 그래프에서 $v$의 링크와 동일한 노드 수를 선택한다. 정확도는 모든 노드에 대해 각 노드에서 선택된 노드와 실제 그래프에서 해당 노드가 인접한 노드인지 일치 여부의 평균 비율로 계산한다.

<표 2> Graph Reconstruction 정확도

| | Amazon | CoCit | Cora | DBLP | Linux | Live |
|---|---|---|---|---|---|---|
| DWK | 0.1716 | 0.0845 | 0.1249 | 0.0977 | 0.0605 | 0.5211 |
| DWN | **0.5361** | 0.1918 | 0.2152 | 0.1311 | 0.0634 | 0.3176 |
| GRV | 0.0041 | 0.0002 | 0.0012 | 0.0024 | 0.0001 | 0.0207 |
| N2V | 0.1699 | 0.0867 | 0.1226 | 0.0990 | 0.0619 | 0.5230 |
| NMF | 0.0416 | 0.0019 | 0.0285 | 0.0118 | 0.0027 | 0.2235 |
| FRD | 0.0223 | 0.0277 | 0.0410 | 0.0446 | **0.0878** | 0.2454 |
| SNT | 0.0438 | - | - | - | - | 0.3397 |
| VRS | 0.4446 | **0.2214** | **0.2476** | **0.1738** | 0.0636 | **0.7031** |

-: SNT(SimNet)은 directed 그래프에 대해 임베딩이 불가능함

**Node Classification.** 이 작업은 모든 노드에 대해 레이블이 지정된 데이터 세트인 CoCit 및 Cora 에만 적용할 수 있다. 두 개의 데이터 세트 각각을 사용하여 테스트 세트로 노드의 $q$% ($q$ = {1, 3, 5, 7, 9})를 임의로 선택하고 훈련 세트로 남은 노드를 선택한다. 그런 다음 노드의 벡터에 로지스틱 회귀를 적용한다. 정확도 측정을 위해 Macro-F1 를 평가 지표로 사용한다.

<표 3> CoCit 데이터 Node Classification 정확도

| | 1% | 3% | 5% | 7% | 9% |
|---|---|---|---|---|---|
| DWK | 0.1213 | 0.1063 | 0.1182 | 0.1154 | 0.1123 |
| DWN | 0.1361 | 0.1410 | 0.1447 | 0.1406 | 0.1444 |
| GRV | 0.2292 | 0.2221 | 0.2152 | 0.2184 | 0.2184 |
| N2V | 0.1007 | 0.1011 | 0.1080 | 0.1095 | 0.1092 |
| NMF | 0.0423 | 0.0409 | 0.0395 | 0.0398 | 0.0410 |
| FRD | 0.1185 | 0.1043 | 0.1013 | 0.1028 | 0.1019 |
| VRS | **0.3062** | **0.3011** | **0.3019** | **0.3134** | **0.3179** |

<표 4> Cora 데이터 Node Classification 정확도

| | 1% | 3% | 5% | 7% | 9% |
|---|---|---|---|---|---|
| DWK | 0.1536 | 0.1797 | 0.1903 | 0.1887 | 0.1879 |
| DWN | 0.2646 | 0.2968 | 0.3033 | 0.3108 | 0.3153 |
| GRV | 0.3476 | 0.3324 | 0.3137 | 0.3222 | 0.3260 |
| N2V | 0.1709 | 0.1756 | 0.1792 | 0.1751 | 0.1721 |
| NMF | 0.0185 | 0.0580 | 0.0660 | 0.0625 | 0.0611 |
| FRD | 0.1535 | 0.1565 | 0.1487 | 0.1583 | 0.1544 |
| VRS | **0.4386** | **0.5050** | **0.5168** | **0.5177** | **0.5216** |

**Link Prediction.** 각 데이터 세트에서 테스트 세트로 링크의 10%를 무작위로 제거한다. 임의의 노드 $u$와 $v$에 대해 Average(Avg.), Hadamard(Had.) 및 Weighted L2(L2.) 연산자[2]를 적용하여 얻은 링크($u$, $v$)의 벡터에 로지스틱 회귀 분류기를 통해 학습한다. 학습 시각 테스트 훈련 세트의 기존 링크는 양성 샘플로 간주되고 그래프에 존재하지 않는 동일한 수의 링크는 음성 샘플로 무작위로 선택된다. Biased negative samples (BNS) link prediction 은 directed 그래프를 각 훈련 및 테스트 세트에 대해 링크의 $q$%($q$ = {25, 50, 75})를 무작위로 선택하고 반대 방향의 링크를 음성 샘플로 간주한다. 다른 (100-$q$)% 음성 샘플은 위와 같이 무작위로 선택된다. 우리는 본 논문에서 $q$ = {25, 75}일 때도 유사한 상황이 관찰되므로, $q$ = 50 일 때의 결과를 제시한다. 우리는 정확도 측정을 위해 area under the curve (AUC)를 평가 지표로 사용한다.

<표 5> Amazon 과 Live 데이터 Link Prediction 정확도

| | Amazon | | | Live | | |
|---|---|---|---|---|---|---|
| | Avg. | Had. | L2. | Avg. | Had. | L2. |
| DWK | 0.5920 | 0.9488 | 0.7559 | 0.7614 | 0.8908 | 0.7348 |
| DWN | **0.6316** | 0.9893 | 0.9956 | 0.7489 | 0.9110 | **0.9428** |
| GRV | 0.5234 | 0.5395 | 0.5325 | 0.6420 | 0.6339 | 0.6221 |
| N2V | 0.5824 | 0.9482 | 0.7615 | 0.7627 | 0.8881 | 0.7337 |
| NMF | 0.4983 | 0.9949 | **0.9969** | 0.7627 | 0.9184 | 0.9346 |
| FRD | 0.5652 | 0.9321 | 0.6794 | **0.7898** | 0.9424 | 0.7233 |
| SNT | 0.5277 | 0.6280 | 0.9254 | 0.7671 | 0.9432 | 0.9222 |
| VRS | 0.5546 | **0.9971** | 0.9964 | 0.7897 | **0.9631** | 0.9359 |

<표 6> CoCit 과 Cora 데이터 Link Prediction 정확도

| | CoCit | | | Cora | | |
|---|---|---|---|---|---|---|
| | Avg. | Had. | L2. | Avg. | Had. | L2. |
| DWK | 0.6299 | 0.6715 | 0.5773 | 0.6723 | 0.7571 | 0.6690 |
| DWN | 0.6922 | 0.7263 | 0.5759 | 0.6889 | 0.8378 | 0.6388 |
| GRV | 0.5994 | 0.5937 | 0.5664 | 0.6585 | 0.6537 | 0.5691 |
| N2V | 0.6327 | 0.6646 | 0.5706 | 0.6728 | 0.7523 | 0.6642 |
| NMF | **0.7295** | 0.6562 | **0.6772** | 0.6934 | 0.7071 | 0.6772 |
| FRD | 0.6933 | **0.8978** | 0.6468 | 0.6227 | 0.8641 | 0.5346 |
| VRS | 0.6359 | 0.5061 | 0.4856 | **0.7537** | **0.9722** | **0.8978** |

<표 7> DBLP 과 Linux 데이터 Link Prediction 정확도

| | DBLP | | | Linux | | |
|---|---|---|---|---|---|---|
| | Avg. | Had. | L2. | Avg. | Had. | L2. |
| DWK | 0.7644 | 0.7628 | 0.6703 | **0.8632** | 0.6381 | **0.8626** |
| DWN | 0.7630 | 0.7591 | 0.6246 | 0.8456 | 0.7526 | 0.7191 |
| GRV | 0.7044 | 0.7027 | 0.6273 | 0.7591 | 0.7495 | 0.7075 |
| N2V | 0.7656 | 0.7819 | 0.6821 | 0.8527 | 0.6427 | 0.8609 |
| NMF | 0.7507 | 0.6947 | 0.6912 | 0.7999 | 0.8212 | 0.8215 |
| FRD | 0.6953 | 0.9047 | 0.6181 | 0.7594 | 0.7699 | 0.7060 |
| VRS | **0.7912** | **0.9661** | **0.8936** | 0.8529 | **0.9665** | 0.8215 |

<표 8> CoCit 과 Cora 데이터 BNS Link Prediction 정확도

| | CoCit | | | Cora | | |
|---|---|---|---|---|---|---|
| | Avg. | Had. | L2. | Avg. | Had. | L2. |
| DWK | 0.5581 | 0.5704 | 0.5366 | 0.5874 | 0.6284 | 0.5861 |
| DWN | 0.5909 | 0.6095 | 0.5374 | 0.5936 | 0.6746 | 0.5676 |
| GRV | 0.5504 | 0.5481 | 0.5338 | 0.5811 | 0.5776 | 0.5291 |
| N2V | 0.5620 | 0.5700 | 0.5316 | 0.5863 | 0.6264 | 0.5810 |
| NMF | **0.6149** | 0.5753 | **0.5886** | 0.7999 | 0.6012 | 0.5910 |
| FRD | 0.5904 | **0.6908** | 0.5714 | 0.7594 | 0.6832 | 0.5097 |
| VRS | 0.5827 | 0.4990 | 0.5025 | **0.8529** | **0.7385** | **0.7122** |

<표 9> DBLP 과 Linux 데이터 BNS Link Prediction 정확도

| | DBLP | | | Linux | | |
|---|---|---|---|---|---|---|
| | Avg. | Had. | L2. | Avg. | Had. | L2. |
| DWK | 0.6334 | 0.6221 | 0.5835 | **0.6781** | 0.5635 | **0.6761** |
| DWN | 0.6323 | 0.6334 | 0.5613 | 0.6755 | 0.6275 | 0.6074 |
| GRV | 0.6091 | 0.6085 | 0.5689 | 0.6290 | 0.6243 | 0.6022 |
| N2V | 0.6326 | 0.6255 | 0.5899 | 0.6723 | 0.5685 | 0.6749 |
| NMF | 0.6272 | 0.5899 | 0.5594 | 0.6482 | 0.6257 | 0.6581 |
| FRD | 0.5978 | 0.7015 | 0.5594 | 0.6226 | 0.6257 | 0.5993 |
| VRS | **0.6505** | **0.7401** | **0.6989** | 0.6755 | **0.7267** | 0.6500 |

## 4. 결론

본 논문에서 처음으로 여섯 가지의 실세계 데이터를 이용하여 노드 사이의 유사성을 고려하지 않는 임베딩 방법인 DeepWalk, DWNS, Gravity, node2vec 및 NetMF 와 노드 사이의 유사성을 기반으로 한 임베딩 방법인 VERSE, SimNet, 및 FREDE 의 성능을 세 가지 기계 학습 작업 실험을 통해 성능을 비교하였다. 우리는 다양한 데이터와 기계 학습 작업을 통해 광범위한 실험을 진행하여 유사성 기반의 그래프 임베딩 방법의 우수성을 확인하였다. 하지만 기계 학습 작업의 종류와 데이터셋에 따라 우수한 성능을 가지는 방법이 다르다는 것을 확인할 수 있다. 이를 통해 기계 학습 작업과 데이터셋에 무관하게 우수한 성능을 보여 줄 수 있는 유사성 기반 임베딩 방법에 대한 연구가 필요한 것을 확인할 수 있다.

## 사사

이 논문은 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2022-0-00352, No.RS-2022-00155586, 실세계의 다양한 다운스트림 태스크를 위한 고성능 빅 하이퍼그래프 마이닝 플랫폼 개발(SW 스타랩)).

## 참고문헌

[1] Bryan Perozzi et al., “DeepWalk: Online Learning of Social Representations” In *Proc. of ACM KDD*, 2014, pp. 701–710.

[2] Aditya Grover et al., “node2vec: Scalable Feature Learning for Networks”, In *Proc. of ACM KDD*, 2016, pp. 855–864.

[3] Yuan Yin et al., “Scalable Graph Embeddings via Sparse Transpose Proximities”, In *Proc. of ACM KDD*. 2019, pp. 1429–1437.

[4] Anton Tsitsulin et al., “VERSE: Versatile Graph Embeddings from Similarity Measures”, In *Proc. of WWW*, 2018, pp. 539–548.

[5] Anton Tsitsulin et al., “FREDE: Anytime Graph Embeddings”, The VLDB Endowment 14, 6, 2021, pp. 1102–1110.

[6] Quanyu Dai et al., “Adversarial Training Methods for Network Embedding”, In *Proc. of WWW*, 2019, pp. 329–339.

[7] Guillaume Salha et al., “Gravity-Inspired Graph Autoencoders for Directed Link Prediction”, In *Proc. of ACM CIKM*, 2019, pp. 589–598.

[8] Jiezhong Qiu et al., “Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec.” In *Proc. of ACM WSDM*, 2018, pp. 459–467.

[9] Moein Khajehnejad. “SimNet: Similarity-Based Network Embeddings with Mean Commute Time.”, Plos ONE, 14, 8, 2019. pp. 1102–1110.

[10] Jaewon Yang et al., “Defining and Evaluating Network Communities Based on Ground-Truth”, In *Proc. of* IEEE ICDM, 2012, pp. 745–754.

[11] Lovro Šubelj et al., “Model of Complex Networks based on Citation Dynamics”, In *Proc. of WWW*, 2013, pp. 527–530.

[12] Masoud Reyhani Hamedani et al., “AdaSim: A Recursive Similarity Measure in Graphs”, In *Proc. of* ACM CIKM. 2021, pp. 1528–1537.

[13] Jérôme Kunegis, “KONECT-The Koblenz Network Collection”, In *Proc. of WWW*, 2013, pp. 1343–1350.