하이퍼그래프 희소성에 따른 하이퍼그래프 임베딩 방법 성능 평가

정소빈 [1], 강윤석 [2], 김상욱 [3*]
[1] 한양대학교 인공지능학과
[2] 미시간대학교 정보대학
[3] 한양대학교 컴퓨터소프트웨어학과

sobin98@hanyang.ac.kr, dyskang@umich.edu, wook@hanyang.ac.kr

Evaluating the Performance of Hypergraph Embedding Methods According to Hypergraph Sparsity

So-Bin Jung[1], David Y. Kang[2], Sang-Wook Kim[3*]
[1]Dept. of Artificial Intelligence, Hanyang University
[2]School of Information, University of Michigan
[3]Dept. of Computer Science, Hanyang University

요 약

실세계에서는 두개 이상의 객체들이 서로 관계를 맺고있다. 단 두 객체 간의 관계만 표현하는 그래프와는 달리 여러 객체들 간의 관계를 표현하는 하이퍼그래프는 그룹 상호작용을 잘 표현할 수 있다. 이러한 강점으로 하이퍼그래프를 활용한 응용들이 많이 제안되고 있다. 하이퍼그래프 임베딩은 하이퍼그래프의 구조를 이용하여 노드를 저차원 벡터로 표현하는 방법이다. 이렇게 표현된 벡터들은 노드 분류, 커뮤니티 탐지, 링크예측 등 광범위한 응용에 활용된다. 하지만 하이퍼그래프는 그래프보다 희소성 문제가 훨씬 더 심해 데이터 셋의 희소성이 하이퍼그래프 임베딩 방법의 성능에 큰 영향을 미칠 수 있다. 따라서, 본 논문에서는 희소성에 따른 하이퍼그래프 임베딩 방법들의 성능을 분석하고자 한다. 우리는 8 개의 실세계 데이터셋을 이용한 실험을 통해 데이터가 희소할수록 하이퍼그래프 임베딩 방법들의 성능이 감소하는 것을 확인하였다.

1. 서론

그래프는 실세계의 객체와 객체 간의 관계를 나타내는 자료구조로 노드는 객체, 엣지는 두 노드 간의 짝상호관계를 나타낸다 [1-3]. 그러나, 그래프는 실세계의 관계들을 올바르게 표현하는데 있어서 한계가 존재한다. 실세계에는 두개뿐만이 아니라 세개 이상의 객체들이 그룹상호관계를 맺는 경우가 존재한다 [4-6]. 하지만, 짝상호관계를 표현하는 엣지를 사용하는 그래프는 이러한 그룹 상호관계를 정확하게 표현하기 어렵다.

하이퍼그래프는 그래프의 일반화된 자료구조로 노드와 하이퍼엣지로 이루어져 있다. 이때 하이퍼엣지는 두개 이상의 노드들의 집합으로, 집합 내 노드들의 그룹상호관계를 나타낸다. 하이퍼엣지 덕분에 하이퍼그래프는 실세계 모든 관계를 정보의 손실 없이 표현할 수 있다 [7,8]. 하이퍼그래프의 표현의 강점으로 인해, 최근 하이퍼그래프를 활용한 응용들이 많이 연구되어왔다 [9-13]. 그 중, 하이퍼그래프 임베딩 방법은 하이퍼그래프 구조를 이용하여 노드를 저차원 임베딩 공간에서 벡터로 표현한다 [10,14,16]. 이렇게 표현된 벡터들은 노드 분류[9-13], 커뮤니티 탐지[14-15], 링크 예측[16,17] 등 광범위한 응용에 활용된다. 여러 문헌에서 다양한 하이퍼그래프 임베딩 방법이 제안되었으며 실험을 통해 그래프를 활용할 때보다 하이퍼그래프를 활용했을 때 더 높은 성능을 보임을 확인하였다 [9,12,13].

실세계 그래프는 희소하다는 특징이 있다 [18]. 희소한 특징이란 가능한 모든 엣지 수에 비해 실제 존재하는 엣지의 수가 적음을 의미한다. 이는 연결 구조 정보의 부족으로 임베딩 시 노드들 간의 관계를 정확하게 학습하는 것을 방해하고, 응용기술의 정확도를 떨어뜨리는 근본적인 원인이 된다 [19]. 하이퍼그래프는 하이퍼엣지 사이즈가 임의적이기 때문에, 가능한

* 교신 저자

모든 엣지의 수가 매우 크다. 그래프 희소성에 대한 식 (1)과 하이퍼그래프 희소성에 대한 식 (2)를 통해 수식적으로 살펴봤을 때, 노드의 수가 늘어날수록 식 (1)은 희소성이 2 차 식으로 증가하지만, (2)는 지수적으로 증가한다. 따라서 하이퍼그래프에서의 희소성은 더 심한 정보 부족 문제를 야기하고 이는 하이퍼그래프 임베딩 방법에 큰 영향을 미칠 수 있다. 본 논문에서는 하이퍼그래프 희소성에 따른 하이퍼그래프 임베딩 방법들의 성능을 비교하고 분석한다.

$$s_G = \frac{|E|}{\binom{|V|}{2}} \quad (graph\ G\ =\ (V,E)) \qquad (1)$$

$$s_H = \frac{|E\prime|}{2|V\prime|} \quad (hypergraph\ H\ =\ (V',E')) \qquad (2)$$

2. 하이퍼그래프 임베딩 방법

하이퍼그래프 임베딩 방법들은 하이퍼그래프의 구조를 이용해 노드를 저차원 벡터로 표현한다. HGNN [9]과 HNHN[10]은 그래프 확장 방법을 이용해 하이퍼그래프를 일반 그래프로 변환하여 그래프 합성곱 네트워크(graph convolutional networks, 이하 GCN)기반 임베딩을 적용한 방법이다. HGNN 은 clique expansion [20]을 이용해 일반그래프로 변환하고, HNHN 은 star expansion [21]을 이용해 이분그래프로 변환한다. UniGNN [11]은 그래프와 하이퍼그래프에 대한 일반화된 메세지 전파 함수를 제안해, 일반 그래프에서 사용된 임베딩 방법들 [22-24]을 하이퍼그래프에 적용한 방법이다. AllSet [12]은 두 가지 다른 멀티 셋 함수를 정의했다. 셋 인코더를 활용하는 프레임 워크를 제안하여 하이퍼그래프의 노드를 임베딩 시킨 방법이다. TriCL [13]은 대조적 학습을 이용하는데 노드 수준, 하이퍼 엣지 수준, 멤버십 수준의 대조로 이루어진 삼방향 대조 손실을 활용해 하이퍼그래프의 노드를 임베딩 시킨 방법이다.

3. 성능 평가

데이터셋. 우리는 [13]에서 사용한 8 개의 실세계 하이퍼그래프 데이터셋을 이용한다. 8 개의 데이터셋에 관련된 통계는 <표 1>에 제시한다.

<표 1> 데이터셋 통계

	# Nodes	# Hyperedges	# Features	# Classes
Cora - C	1434	1579	1433	7
Citeseer	1458	1,079	3,703	6
Pubmed	3840	7,963	500	3
Cora-A	2388	1,072	1,433	7
DBLP	41302	22,363	1,425	6
Mushroom	8124	298	22	2
NTU2012	2012	2,012	100	67
ModelNet40	12311	12,311	100	40

실험방법. 우리는 다운 스트림 태스크로 노드 분류를 수행한다. 노드 분류는 주어진 노드의 클래스 레이블을 예측하는 문제로, 임베딩의 성능을 평가하는데 있어서 많이 사용되는 태스크 중 하나이다. 우리는 데이터를 학습/검증/테스트 샘플로 무작위로 분할했고 이때의 비율은 10%/10%80%로 설정했다. 이때, 각 데이터셋에서 전체 하이퍼 엣지 중 사용할 하이퍼 엣지의 비율 $\boldsymbol{\theta}$%($\boldsymbol{\theta}$={20,40,60,80,100})을 선택해 희소성의 정도를 조절한다. 즉, $\boldsymbol{\theta}$값이 감소할수록 하이퍼그래프의 희소성이 증가한다는 것을 나타낸다.

우리는 2 장에서 언급한 5 가지 하이퍼그래프 임베딩 방법들의 성능을 평가한다: HGNN[9], HNHN[10], UniGNN[11], AllSet[12], TriCL[13]. 추가로 일반그래프 희소성에 따른 임베딩 방법의 성능도 확인하여 하이퍼그래프 희소성 문제의 심각성을 확인한다. 이를 위해 우리는 주어진 하이퍼그래프를 clique expansion 을 이용하여 일반그래프로 변환하고, 변환된 그래프에 GCN 을 수행한다 (CEGCN).

실험결과. <표 2>는 하이퍼그래프 임베딩 방법들의 노드 분류 정확도를 보여준다. 실험결과를 정리하면 다음과 같다. 첫째, 그래프가 희소해질수록 노드 분류 정확도가 떨어지는 정도가 일반 그래프를 사용할 때보다 하이퍼그래프를 사용할 때가 더 크다. 이는 하이퍼그래프가 그래프보다 희소성에 민감하다는 사실을 의미한다. 둘째, 그래프가 희소할수록 성능은 낮아지지만 임베딩 방법들의 순위는 유지된다. TriCL 은 모든 데이터셋에서 가장 좋은 성능을 보인다. 또한, 희소성이 증가할수록 정확도가 감소하는 정도가 다른 하이퍼그래프 임베딩 방법들에 비해 제일 작다. 그 다음은 AllSet 으로 3 개의 데이터셋(Cora-C, Cora-A, Pubmed)을 제외하면 두번째로 좋은 성능을 보이고 있다.

4. 결론

본 논문에서, 우리는 희소성 정도에 따른 하이퍼그래프 임베딩 방법의 성능을 분석하였다. 실험을 통해, 하이퍼그래프 희소성 문제가 일반 그래프 희소성 문제보다 심각함을 확인하였다. 또한, 임베딩 방법들에서 희소한 데이터 셋일수록 성능이 낮아지는 것을 확인하였다. 이러한 결과는 하이퍼그래프 임베딩시에 임베딩 방법 뿐만이 아니라 희소성 또한 성능에 영향을 미친다는 것을 확인할 수 있다.

사사

이 논문은 정부(과학기술정보통신부)의 재원으로 (1) 정보통신기획평가원의 지원과 (No. 2020-0-01373, 인공지능대학원지원(한양대학교), No.RS-2022-00155586, 실세계의 다양한 다운스트림 태스크를 위한 고성능 빅 하이퍼그래프 마이닝 플랫폼 개발(SW 스타랩)) (2) 한국연구재단의 지원을 받아 수행된 연구임 (No.2018R1A5A7059549)

<표 2> 하이퍼그래프 임베딩 방법들의 노드 분류

	θ	CEGCN	HGNN	HNHN	UniGNN	Allset	TriCL		θ	CEGCN	HGNN	HNHN	UniGNN	Allset	TriCL
Cora-A	**100**	71.692	77.841	77.032	77.382	76.761	**81.651**	**DBLP**	**100**	85.220	90.090	89.690	90.080	90.100	**91.200**
	80	69.852	76.301	74.843	75.762	73.871	**80.201**		**80**	84.520	89.210	88.660	89.180	89.380	**90.760**
	60	66.372	72.802	70.993	72.432	70.093	**78.421**		**60**	83.870	88.110	87.490	87.990	88.320	**89.980**
	40	64.592	70.451	68.884	69.732	66.942	**76.551**		**40**	83.200	86.880	86.110	86.620	87.130	**88.120**
	20	61.082	65.762	65.571	65.682	63.072	**73.891**		**20**	82.540	84.900	84.110	84.720	85.230	**87.600**
Citeseer	**100**	60.392	67.232	67.282	68.531	68.671	**72.121**	**Mushroom**	**100**	93.200	99.680	99.710	99.810	99.810	**99.830**
	80	59.552	66.972	66.581	67.352	66.802	**71.351**		**80**	94.410	99.430	99.510	99.730	**99.780**	99.760
	60	59.352	65.652	65.562	66.102	66.482	**70.241**		**60**	94.140	99.150	99.430	99.640	**99.720**	99.620
	40	58.472	63.942	63.772	65.302	65.361	**69.351**		**40**	92.320	98.990	99.340	99.580	99.600	**99.650**
	20	58.093	61.972	62.102	63.652	63.912	**68.891**		**20**	93.920	98.760	99.180	99.430	99.520	**99.540**
Pubmed	**100**	79.990	83.220	83.590	83.960	82.910	**84.290**	**NTU2012**	**100**	67.262	73.612	73.002	71.383	**75.372**	75.172
	80	79.580	82.750	82.870	82.830	82.460	**83.920**		**80**	66.522	72.932	72.632	71.182	74.242	**74.802**
	60	79.300	81.950	81.700	81.900	81.930	**83.440**		**60**	66.222	72.692	72.352	70.302	73.622	**74.542**
	40	78.770	81.010	80.760	80.780	81.030	**82.540**		**40**	66.172	71.582	71.602	69.812	73.142	**73.942**
	20	76.981	79.750	79.430	79.520	79.680	**81.110**		**20**	65.681	70.552	70.572	69.002	72.212	**72.552**
Cora-C	**100**	67.851	78.381	76.010	77.691	76.251	**82.130**	**ModelNet40**	**100**	86.850	94.910	96.930	95.170	96.830	**97.080**
	80	66.331	76.651	73.331	75.861	74.901	**80.580**		**80**	86.870	94.830	96.560	95.050	96.770	**97.070**
	60	64.701	74.221	70.991	73.541	72.491	**78.751**		**60**	87.170	94.930	96.350	94.920	96.790	**97.220**
	40	64.141	71.201	68.381	70.761	70.051	**76.320**		**40**	87.610	94.970	95.820	94.710	96.770	**97.090**
	20	63.001	67.331	64.631	67.511	66.641	**73.561**		**20**	89.580	94.910	95.150	94.550	96.330	**96.580**

참고문헌

[1] A. Broder et al., “Graph structure in the web”, Computer Networks, 33, 1, 309–320, 2000.

[2] M. Girvan and M. Newman., “Community structure in social and biological networks”, Proc. Natl. Acad. Sci. U.S.A, 99, 12, 7821, 2002.

[3] R. Milo et al., “Network motifs: simple building blocks of complex networks”, Science, 298, 5594, 824–827, 2002.

[4] A. Benson et al., “Simplicial closure and higher-order link prediction”, Proc. Natl. Acad. Sci. U.S.A, 115, 48, 11221–11230, 2018.

[5] A. Benson, D. Gleich, and J. Leskovec, “Higher-order organization of complex networks”, Science, 353, 6295, 163–166, 2016.

[6] A. Benson, R. Kumar, and A. Tomkins, “Sequences of sets”, KDD, 2018, 1148-1157.

[7] P. Bonacich, A. Holdren, and M. Johnston, “Hyper-edges and multidimensional centrality”, Social Networks, 26, 3, 189-203, 2004.

[8] C. Berge, “Hypergraphs”, North Holland(Amsterdam), 2013.

[9] Y. Feng et al., “Hypergraph neural networks”, AAAI, 2019, 3558-3565.

[10] Y. Dong, W. Sawin, and Y. Bengio, “Hnhn: Hypergraph networks with hyperedge neurons”, ICML, 2020.

[11] J. Huang and J. Yang, “Unignn: a unified framework for graph and hypergraph neural networks”, IJCAI, 2021, 2563-2569.

[12] E. Chien et al., “You are allset: A multiset function framework for hypergraph neural networks”, ICLR, 2022.

[13] D. Lee and K. Shin, “I’m me, we’re us, and i’m us: Tri-directional contrastive learning on hypergraphs”, AAAI, 2023, 8456-8464.

[14] I. Chien, C. Lin, and I. Wang, “Community detection in hypergraphs: Optimal Statistical Limit and Efficient Algorithms”, AISTATS, 2018, 871-879.

[15] P. Chodrow, N. Veldt, and A. Benson, “Generative hypergraph clustering: from blockmodels to modularity”, Sci. Adv. 7, 28, 710, 2021.

[16] N. Yadati et al., “NHP: Neural hypergraph link prediction”, CIKM, 2020, 1705-1714.

[17] H. Hwang et al., “AHP: Learning to negative sample for hyperedge prediction”, SIGIR, 2022, 2237–2242.

[18] Y. Li et al., “Learning signed network embedding via graph attention”, AAAI, 2020, 4772–4779.

[19] H. Yoo et al., “Directed network embedding with virtual negative edges”, WSDM, 2022, 1291–1299.

[20] L. Sun, S. Ji, and J. Ye, “Hypergraph spectral learning for multi-label classification”, KDD, 2008, 668–676.

[21] J. Zien, M. Schlag, and P. Chan, “Multilevel spectral hypergraph partitioning with arbitrary vertex sizes”, IEEE TCAD, 18, 9, 1389–1399, 1999.

[22] T. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks”, ICLR, 2017.

[23] P. Veličković et al., “Graph attention networks”, ICLR, 2018.

[24] K. Xu et al., “How powerful are graph neural networks?”, ICLR, 2019.