VilLain: Unsupervised Node Embedding on Hypergraphs via Virtual Label Propagation (Online Appendix)

Anonymous Author(s)

ABSTRACT

In this supplementary document, we (1) discuss desirable properties in detail and (2) provide experimental settings.

1 DESIRABLE PROPERTIES

In this section, we discuss some desirable properties for learning node embeddings in hypergraphs which VilLain is designed to have:

P1. No labels (i.e., unsupervised): Manual labeling can be time-consuming, expensive, and vulnerable to noises. Moreover, many real-world hypergraphs are unlabelled. These facts motivate us to learn node embeddings without requiring semantic labels or even the number of unique labels.

P2. No features: Despite their usefulness, node attributes are not present in a large number of real-world hypergraphs, and thus we aim to obtain expressive node embeddings without requiring such side information.

P3. No augmentations: Many graph contrastive learning methods adopt data augmentation for unsupervised learning [7, 10, 11]. However, the quality of their output embeddings is highly dependent on the way and degree of data augmentation. Moreover, data augmentation may pose a risk of change in the semantic of the original graph. Thus, we aim to avoid such structural manipulation.

P4. No negative samples: In many unsupervised methods, node embeddings are trained so that positive samples and negative samples are distinguished from each other. This, however, may require a large number of negative examples, which are time-consuming to process, and may suffer from performance degradation due to false negatives. Thus, relieving the reliance on negative samples is desirable.

In Table 1, we review existing node embedding methods regarding the above desired properties. DeepWalk [6] and Node2Vec [3] train the skip-gram model to learn node embeddings, and to this end, Node2Vec uses negative samples, while DeepWalk adopts the hierarchical softmax. DGI [7], GRACE [10], and GMI [5] use node features and negative samples for contrastive learning. GRACE also augments the input graph with a risk of changing the semantic of the original graph. More importantly, all these methods are designed for graphs, and thus they cannot fully utilize the higher-order information of hypergraphs. For node embedding on hypergraphs, LBSN [9] and Hyper2Vec [4] rely negative sampling; and HGNN [2], HNHN [1], and HyperGCN [8] are typically trained in a (semi-)supervised manner assuming an enough number of labels.

Table 1: VilLain satisfies four properties desirable for node embedding. Specifically, it does not require node labels, node features, data augmentation, or negative sampling.

		DeepWalk [6]	Node2Vec [3]	DGI [7]	GRACE [10]	GMI [5]	LBSN [9]	Hyper2Vec [4]	HGNN [2]	HNHN [1]	HyperGCN [8]	**VilLain**
Form of Input Data		Graph					Hypergraph					
Requires	Node Labels	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗
	Node Features	✗	✗	✓	✓	✓	✗	✗	✓	✓	✓	✗
	Data Augmentation	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗
	Negative Sampling	✗	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗

Table 2: Github links to the baseline source codes.

Method	Source Code
Deepwalk & Node2vec	https://github.com/benedekrozemberczki/karateclub
DGI	https://github.com/PetarV-/DGI
GRACE	https://github.com/CRIPAC-DIG/GRACE
GMI	https://github.com/zpeng27/GMI
Hyper2vec	https://github.com/jeffhj/NHNE
HGNN	https://github.com/iMoonLab/HGNN
HNHN	https://github.com/twistedcubic/HNHN
HyperGCN	https://github.com/malllabiisc/HyperGCN

2 EXPERIMENTS

In this section, we provide additional information of experimental settings and results that are not covered in the main paper.

2.1 Experimental Settings

We provide detailed settings of the experiments that we conducted.

Datasets: For all datasets, we use the largest connected component of the original hypergraph. We process the huge Amazon dataset by filtering out nodes with 10 most frequently appeared labels. Then, we randomly sample 1% of the nodes from each label.

Implementation: We summarize the github links of the source codes we used to conduct experiments on baselines in Table 2.

CIKM '22, October 17–22, 2022, Atlanta, Georgia, USA

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

REFERENCES

[1] Yihe Dong, Will Sawin, and Yoshua Bengio. 2020. HNHN: Hypergraph networks with hyperedge neurons. *arXiv preprint arXiv:2006.12278* (2020).

[2] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *AAAI*.

[3] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*.

[4] Jie Huang, Chuan Chen, Fanghua Ye, Jiajing Wu, Zibin Zheng, and Guohui Ling. 2019. Hyper2vec: Biased random walk for hyper-network embedding. In *DASFAA*.

[5] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. Graph representation learning via graphical mutual information maximization. In *WWW*.

[6] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*.

[7] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. In *ICLR*.

[8] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. 2019. HyperGCN: a new method of training graph convolutional networks on hypergraphs. In *NeurIPS*.

[9] Dingqi Yang, Bingqing Qu, Jie Yang, and Philippe Cudre-Mauroux. 2019. Revisiting user mobility and social relationships in lbsns: a hypergraph embedding approach. In *WWW*.

[10] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131* (2020).

[11] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph contrastive learning with adaptive augmentation. In *WWW*.