Hyper2vec: Biased Random Walk for Hyper-network Embedding

Jie Huang[1,2], Chuan Chen[1,2(✉)], Fanghua Ye[1,2], Jiajing Wu[1,2(✉)], Zibin Zheng[1,2], and Guohui Ling[3]

[1] School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
{chenchuan,wujiajing}@mail.sysu.edu.cn

[2] National Engineering Research Center of Digital Life, Sun Yat-sen University, Guangzhou, China

[3] Tencent Technology, Shenzhen, China

Abstract. Network embedding aims to obtain a low-dimensional representation of vertices in a network, meanwhile preserving structural and inherent properties of the network. Recently, there has been growing interest in this topic while most of the existing network embedding models mainly focus on normal networks in which there are only pairwise relationships between the vertices. However, in many realistic situations, the relationships between the objects are not pairwise and can be better modeled by a hyper-network in which each edge can join an uncertain number of vertices. In this paper, we propose a deep model called Hyper2vec to learn the embeddings of hyper-networks. Our model applies a biased 2^{nd} order random walk strategy to hyper-networks in the framework of Skip-gram, which can be flexibly applied to various types of hyper-networks.

1 Introduction

In the real world, there are a variety of systems that are composed of objects connected in a particular way, which can be represented as networks. Recently, a series of network embedding models [3,7,9] have been proposed to obtain a low-dimensional representation of vertices in the networks. However, the relationships between the objects may be not pairwise. For instance, giving a set of articles, the number of authors in each article is not fixed at 2, so it is not suitable to model the relationships between the authors using a normal network. A hyper-network in which each edge can connect an uncertain number of vertices [1] can better depict the relationships between the authors instead.

However, most embedding algorithms cannot be applied to hyper-networks directly. Till now, only a few embedding models for hyper-networks have been proposed and these models are not particularly applicable for general cases. A conventional approach to deal with hyper-networks is to convert them into normal networks, which will lose some important structural information of the original hyper-networks.

Based on the above considerations, we propose a general hyper-network embedding model called **Hyper2vec**, which is based on biased 2^{nd} order random walks in hyper-networks. In this model, a 1^{st} order random walker moves to the next vertex based on the current vertex only, and a 2^{nd} order random walker moves to the next vertex based on both the current and the previous vertices. Furthermore, we utilize a degree biased random walk to correct the negative bias or introduce a positive bias for random walks.

2 Notations and Definitions

A **hyper-network** is defined as a hypergraph $G(V, E)$ with the vertex set V and the hyperedge set E. A **hyperedge** $e \in E$ indicates the relationship of an uncertain number of vertices, which is a subset of V. A weighted hyper-network defined as $G(V, E, w)$ is a hypergraph that has a weight $w(e)$ associated with each hyperedge e. A hyperedge e is incident with a vertex v if $v \in e$, and if $v \in e$, then $h(v, e) = 1$, otherwise $h(v, e) = 0$.

For a vertex $v \in V$, and a hyperedge $e \in E$, the degree of v is defined as $d(v) = \sum_{e \in E} w(e)h(v, e)$, and the degree of e is defined as $\delta(e) = |e| = \sum_{v \in V} h(v, e)$.

3 Hyper2vec: The Proposed Method

In order to preserve the high-order proximity of hyper-networks, we propose an efficient and scalable biased 2^{nd} order random walk model in the framework of Skip-gram.

The 1^{st} order random walk strategy decides the probability of moving to the next vertex based on the current vertex. To generalize random walks in hyper-networks, we adopt the transition rule described in [12]. Given the current vertex v, we first randomly select a hyperedge e incident with v based on the weight of e, and then pick the next vertex $x \in e$ uniformly at random. The unnormalized transition probability of each walk of the 1^{st} order model can be given as follows:

$$\pi_1(x|v) = \sum_{e \in E} w(e) \frac{h(v, e)}{d(v)} \frac{h(x, e)}{\delta(e)}. \tag{1}$$

To shuttle between homophily and structural equivalence when doing prediction tasks, it's feasible to introduce a 2^{nd} order random walk model [3]. We divide the neighbors of vertex v into three categories, the previous vertex u, the neighbors of the previous vertex u, and the others. The 2^{nd} order search bias is defined as follows:

$$\alpha(x|v, u) = \begin{cases} \frac{1}{p}, & \text{if } x = u \\ 1, & \text{if } \exists e \in E, x \in e, u \in e \\ \frac{1}{q}, & \text{others} \end{cases}. \tag{2}$$

The unnormalized transition probability of the 2^{nd} order random walk model is defined as $\pi_2(x|v, u) = \alpha(x|v, u)\pi_1(x|v)$. The relationship between parameters p,

q, and 1 leads to the tendency of **Depth-First Search (DFS)** which encourages outward exploration and **Breadth-First Search (BFS)** which obtains a local view of the start vertex.

Previous studies have shown that DFS and BFS search strategy will introduce some negative bias. For instance, it is observed that BFS introduces a bias towards big degree vertices [6]. On one hand, we need to correct the negative bias [4]. On the other hand, we need to introduce some positive bias to capture some special characteristics of networks or balance the update opportunities for vertices [2,11].

Based on the above considerations, we propose a degree biased random walk model, which has two sampling strategies, **Big-Degree Biased Search (BDBS):** the walker tends to choose a neighbor vertex with bigger degree, and **Small-Degree Biased Search (SDBS):** the walker tends to choose a neighbor vertex with smaller degree.

We define a biased 2^{nd} order random walk by introducing a bias coefficient $\beta(x)$ based on the degree of x with a parameter r. To make the mechanism more flexible, parameter r should not only guide the strategy of BDBS or SDBS but also control the degree of influence. To achieve the above aim, we define degree bias $\beta(x) = d(x) + r$ $(r > 0)$ for BDBS. Obviously, $\beta(x)$ grows linearly with $d(x)$, and as r increases, the degree of BDBS tendency declines. With a similar principle of BDBS, we combine SDBS $(r < 0)$ to produce the final formula as follows:

$$\beta(x) = \begin{cases} d(x) + r, & r > 0 \\ \frac{1}{d(x) - r}, & r < 0 \\ 1, & r = 0 \end{cases}. \tag{3}$$

Combined with the 2^{nd} order random walk model, the biased 2^{nd} order transition probability $p_2'(x|v,u)$ is calculated as follows:

$$p_2'(x|v,u) = \frac{\alpha(x|v,u) \cdot \beta(x) \cdot \sum_{e \in E} w(e) \frac{h(v,e)}{d(v)} \frac{h(x,e)}{\delta(e)}}{Z}, \tag{4}$$

where Z is a normalizing factor. Based on the transition probability $p_2'(x|v,u)$, starting from each vertex in the hyper-network, we can generate a set of random walks. We produce the final embedding results via Skip-gram model. Skip-gram [5] is a language model that maximizes the co-occurrence probability among the words that appear within a window in a sentence, which is replaced by a random walk in our model.

4 Experiments

In this section, we conduct classification experiments on DBLP and IMDb datasets. DBLP[1] is a computer science bibliography. We abstract each author as a vertex and each article as a hyperedge. We select some conferences from

[1] http://dblp.uni-trier.de/xml/.

three different research fields: KDD, WWW from data mining, NIPS, ICML from machine learning, and CVPR, ICCV from computer vision. We use the most frequent fields as the author's label. Authors with degree less than 3 are filtered out, and finally, the hyper-network contains 8329 vertices and 14677 hyperedges. IMDb[2] dataset contains the information about movies and actors. We use a subset of IMDb dataset in our experiments. We abstract each actor as a vertex, and choose the top three actors in each movie to form a hyperedge. We use the top three frequent genres as the actor's labels. Actors with degree less than 3 are filtered out, and finally, the hyper-network contains 4396 vertices and 3875 hyperedges.

Table 1. Results of classification on DBLP and IMDb

Embedding algorithm	DBLP		IMDb	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1
DeepWalk	0.8023	0.7875	0.4798	0.2844
Node2vec	0.8074	0.7929	0.4783	0.2887
LINE	0.7450	0.7242	0.4624	0.2411
HHE	0.4444	0.2081	0.4381	0.0902
HGE	0.5411	0.4764	0.4410	0.2128
Hyper2vec	**0.8180**	**0.8039**	**0.4953**	**0.3101**

We compare the performance of Hyper2vec against normal network embedding algorithms, including DeepWalk [7], Node2vec [3] and LINE [9] and hyper-network embedding algorithms, including HHE [13] and HGE [10]. For normal network embedding algorithms, we use clique expansion [8] to transform a hyper-network into a normal network. Logistic regression is used as the outer classifier and the Micro-F1 and Macro-F1 scores of 5-fold cross-validation are shown in Table 1. From the results, we can draw a conclusion that Hyper2vec performs better than the baselines in classification tasks.

Acknowledgment. The paper was supported by the National Key Research and Development Program (2016YFB1000101), the National Natural Science Foundation of China (11801595, 61503420), the Natural Science Foundation of Guangdong (2018A030310076), the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (2016ZT 06D211) and the CCF Opening Project of Information System.

[2] https://www.imdb.com/.

References

1. Berge, C.: Hypergraphs: Combinatorics of Finite Sets, vol. 45. Elsevier, Amsterdam (1984)
2. Feng, R., Yang, Y., Hu, W., Wu, F., Zhuang, Y.: Representation learning for scale-free networks. arXiv preprint arXiv:1711.10755 (2017)
3. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: KDD, pp. 855–864. ACM (2016). https://doi.org/10.1145/2939672.2939754
4. Kurant, M., Markopoulou, A., Thiran, P.: On the bias of BFS. arXiv preprint arXiv:1004.1729 (2010)
5. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
6. Najork, M., Wiener, J.L.: Breadth-first crawling yields high-quality pages. In: WWW, pp. 114–118. ACM (2001). https://doi.org/10.1145/371920.371965
7. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: KDD, pp. 701–710. ACM (2014). https://doi.org/10.1145/2623330.2623732
8. Sun, L., Ji, S., Ye, J.: Hypergraph spectral learning for multi-label classification. In: KDD, pp. 668–676. ACM (2008). https://doi.org/10.1145/1401890.1401971
9. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: large-scale information network embedding. In: WWW, pp. 1067–1077. International World Wide Web Conferences Steering Committee (2015). https://doi.org/10.1145/2736277.2741093
10. Yu, C.A., Tai, C.L., Chan, T.S., Yang, Y.H.: Modeling multi-way relations with hypergraph embedding. In: CIKM, pp. 1707–1710. ACM (2018)
11. Zeng, Z., Liu, X., Song, Y.: Biased random walk based social regularization for word embeddings. In: IJCAI, pp. 4560–4566 (2018)
12. Zhou, D., Huang, J., Schölkopf, B.: Learning with hypergraphs: clustering, classification, and embedding. In: NIPS, pp. 1601–1608 (2007)
13. Zhu, Y., Guan, Z., Tan, S., Liu, H., Cai, D., He, X.: Heterogeneous hypergraph embedding for document recommendation. Neurocomputing **216**, 150–162 (2016). https://doi.org/10.1016/j.neucom.2016.07.030