ISBN: 978-981-18-7950-0
2023 the 13th International Workshop on Computer Science and Engineering (WCSE 2023)
doi: 10.18178/wcse.2023.06.024

HOCD: Overlapping Community Detection based on Hypergraph Convolutional Neural Networks

Dengdi Sun [1,4], You Shi[2], Bin Luo[2] and Zhuanlian Ding [3+]

[1] School of Artificial Intelligence, Anhui University, Hefei, 230601, China.

[2] School of Computer Science and Technology, Anhui University, Hefei, 230601, China.

[3] School of Internet, Anhui University, Hefei, 230039, China.

[4] Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, 230026, China.

Abstract. Community structure is one of the universal topological attributes in complex networks. Finding community structure is the basic task of complex network analysis. Community detection is designed to divide the network into multiple substructures, which plays an important role in understanding the network and revealing the potential function of the network. As the amount of data increases, the network becomes more complex. In particular, nodes in a network may have multiple identities, such as node features and whether communities overlap. In recent years, many community detection methods based on deep learning have made great progress. However, the problem of these works is that the representation ability of models is limited. How to design an efficient and powerful representation model is very important. We have noticed that the hypergraph model can detect the higher-order relationship in the data. We use the powerful representation ability of the hypergraph convolutional network and apply it to the task of community detection, and finally realize our idea, and our method has achieved the best performance in the experiment.

Keywords: Overlapping community detection, Hypergraph convolution neural networks.

1. Introduction

In recent years, graph convolutional neural network has become an important research direction in the field of artificial intelligence and big data due to its strong representation ability, good performance and interpretability. It has been widely used in recommender system, bioinformatics, social networks, computer vision, natural language processing and other fields, and it has achieved good results. A graph structure is a typical data structure [1-2], whose vertices represent nodes and edges represent the connections between nodes, while the hypergraph structure is used to build higher-order dependencies between data.

The study of graph structure by neural network has attracted great attention of researchers. In [3] and [4], neural networks on graphs were first introduced to apply recurrent neural networks to graphs. To generalize convolutional networks to graphs, the method is divided into spectral and non-spectral methods. For spectral methods, the convolution operation is represented in the spectral domain of the graph. [5] introduced the first graph CNN, which used graphic Laplace basis as an analogy to the Fourier transform. In [6], spectral filters can be parameterized using smoothing coefficients to be spatially localized. In [7], spectral filters are further approximated using the Chebyshev expansion of the graph Laplacian. Then, in [8], the Chebyshev polynomials are reduced to first-order polynomials to form an efficient hierarchical propagation model. For spatial methods, the convolution operation is defined in the spatial closure node group. In [9], the power of the transition matrix is used to define the neighborhood of nodes. [10] generalize convolution in the spatial domain using a local path operator in the form of a Gaussian mixture model. In [11], attention mechanisms are introduced into graphs to build attention-based architectures to perform node classification tasks on graphs. [12] proposed a learnable Hypergraph Laplacians module for updating the hypergraph topology during training.

The hypergraph structure is used to build higher-order dependencies between data. Hypergraph learning was first proposed in [13] as a propagation process over hypergraph structures. Transductive reasoning on

+ Corresponding author. Tel:+86 551 65108582.
E-mail address: dingzhuanlian@163.com

hypergraphs aims to minimize the difference between tones with strong connections on hypergraphs. In [14], hypergraph learning is further applied to video object segmentation. [15] used a hypergraph structure to model image relationships and perform inductive reasoning on image ranking. To further improve the structure of hypergraphs, the learning of hyperedge weights has attracted attention, which has a great impact on data correlation modeling.

There are also some other ways to create a hypergraph from a given graph, such as Normalized cut, etc. But, as we do not want to create an accurate hypergraph and obviously this means of producing is NP-Hard, we ignore to describe them more. By using HGCN as a bridge to the family of graph neural networks, it is possible to establish linkages with other frameworks, such as MoNet [16], and develop higher-order counterparts of these variants to handle non-pairwise relations.

Our work is to design a novel Hypergraph Convolution Networks for overlapping community detection. The main contributions of our work are as follows:

- First to apply hypergraph neural networks to task of overlapping community detection.
- Conduct a comprehensive evaluation of our model and show that it outperforms existing overlapping community detection methods.

2. Related Works

2.1. Hypergraph Convolutional Neural Networks

In recent years, graph-based convolutional neural networks have attracted a lot of attention, first proposed by [17]. Some classical methods [8] [18] [19] are specialized neural network structures that can operate on the structure of graphs. A graph neural network is a method for learning deep models or embedding graph-structured data. The goal of embedding methods [20-23], is to learn vector representations of nodes in the graph, which are then used for downstream tasks. The biggest advantage of the graph convolutional neural network over the traditional neural network is that it can use the neural network model to encode the graph structure of different input data. Graph convolutional neural networks haveshown great advantages in representation learning due to their ability to exploit graph structures.

The traditional graph convolutional neural network approach relies on pairwise relationships between data, which means that each edge on the graph can only link two vertices. However, in many practical application scenarios, the structure of the data may be beyond the existence of pairwise relationships. For example, suppose that the edge represents the article, and the point is the author of the article. In a simple graph, one author can only connect to one article, thus easily losing multiple authors of the same article. Because a simple graph can only be two nodes and one line, and an article can only have two authors; but for a hypergraph, more than two nodes can be connected on an edge. The paired relationship has been unable to meet our actual needs very well. But hypergraphs can satisfy such a requirement. At this stage, hypergraph convolutional neural networks have become a new research direction.

2.2. Community Detection

Community structure is one of the ubiquitous topological properties in complex networks and finding community structure is the basic task of complex network analysis. Community detection aims to divide the network into multiple subgraphs, and it also plays an important role in understanding the network and revealing the potential functions of the network. A graph neural network is a powerful model for processing graph-structured data, which can extract and represent features from the graph structure. Network data can be regarded as graph data, so the use of graph neural network model to solve the task of community detection is a new research direction of community detection. Community-level descriptions are able to capture important information about the global structure of the graph. Such descriptions are useful in many practical applications, such as identifying users with similar interests in social networks [24] or proteins with similar functions in biochemical networks [25].

2.3. Community Detection Based on Deep Learning

At present, many deep learning methods have been well applied in community detection, and the problem of community detection on graphs has been fully confirmed in the research literature. We note that most of the current deep learning-based work is Tasks based on link prediction and node classification have not received enough attention in the unsupervised community detection task. Some methods [26-28] have been developed based on this, but these works are cantered on non-overlapping Community detection is carried out, however the real communities are often mostly overlapping communities. Overlapping community detection algorithms can be roughly classified into non-negative matrix factorization [29-31] probabilistic inference [32-35], and heuristic methods [36-39]. Deep learning for graphs can be broadly classified into two categories: graph neural networks and node embedding.

GNNs [8] [10] [40] is a specialized neural network that can operate on graph-structured data, and the goal of embedding approaches is to learn vector representations of nodes in a graph for the downstream tasks. Although the embedding method can detect the disjoint community structure well, it is no longer suitable for the detection of overlapping communities, which is also proved by our experiments.

Our work is to design an overlapping community detection model based on hypergraph convolutional neural networks, and we note that our work is the first to apply hypergraph neural networks to task of overlapping community detection.

3. Methodology

Here, we introduce our model for overlapping community detection based on hypergraph neural network convolutions (HOCD). Its core idea is to combine the powerful representation ability of hypergraph convolutional neural network with overlapping community detection based on Bernoulli-Poisson probability distribution.

3.1. Definitions

Most of the existing work operates on graphs of pairwise relations. The definition of a simple graph is $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ represents the set of vertices of the graph, and the edge set is represented as $E \subseteq V \times V$. The adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N \times M}$ of a graph is expressed as a pairwise. In a simple graph, each edge can only connect two vertices. But in many practical applications, the relationship between vertices is a complex unpaired relationship. To represent this unpaired relationship, we refer to a practical graph structure - the hypergraph structure. In a hypergraph, an edge can connect more than two vertices. We set $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ to be a hypergraph with N vertices and M hyperedges, each hyperedge $\varepsilon \in \boldsymbol{E}$ is assigned in a positive weight W_ε, and all positive weights are stored in a diagonal matrix. Except for simple graphs that define adjacency matrices, hypergraph can be represented by an incidence matrix $\boldsymbol{H} \in \mathbb{R}^{N \times M}$, then incidence matrix $\boldsymbol{H}$ can be defined as:

$$H_{i,j} = \begin{cases} 1, if\ \ node\ \ v_i\ \ \in\ \ hyperedge\ \ \varepsilon_j \\ 0, if\ node\ \ v_i\ \ \notin\ \ hyperedge\ \varepsilon_j \end{cases}, \tag{1}$$

3.2. Hypergraph Convolutional Network

In fact, the hypergraph convolutional neural network is a special case of the graph convolutional neural network, and we will demonstrate this process later. The main obstacle to defining convolution operators in hypergraphs is the transition probability between two vertices, which can be used to propagate the embedding (or feature) of each vertex in a graph neural network. To achieve this, we hold two assumptions: (1) There should be more propagation between those vertices linked by common hyperedges. (2) Hyperedges with larger weights should have larger confidence in propagation. Then the process can be defined as:

$$x_i^{(l+1)} = \sigma\left(\sum_{j=1}^{N}\sum_{\epsilon=1}^{M} H_{i\epsilon}H_{j\epsilon}W_{\epsilon\epsilon}x_j^{(l)}\boldsymbol{P}\right), \tag{2}$$

where $X_i^{(l)}$ is the embedding of the of the l-th layer, $\sigma(\cdot)$ is a non-linear activation function like ReLU, and $\boldsymbol{P}$ is the trainable weight matrix in the l-th layer convolution. Considering the scale change of $x_j^{(l)}$, Thus, we impose a symmetric normalization and arrive at our final formulation:

$$\boldsymbol{X^{(l+1)}} = \sigma\left(\boldsymbol{D^{-1/2}HWB^{-1}H^{\mathrm{T}}D^{-1/2}X^{(l)}P}\right), \tag{3}$$

here, we recall that $\boldsymbol{D}$ and $\boldsymbol{B}$ are the degree matrices of the vertex and hyperedge in a hypergraph, respectively.

3.3. Overlapping Community Detection

The purpose of overlapping community detection is to assign nodes to C communities. This assignment can be represented as a non-negative community attribute matrix This assignment can be represented as a non-negative community attribute matrix $\boldsymbol{F} \in \mathbb{R}_{\geq 0}^{N \times C}$, where $\boldsymbol{F}_{uc}$ represents the strength of node u's membership in community c. There may be cases where a node is not assigned to any community, and there may be cases where a node belongs to multiple communities. This is the reason why the overlapping community is complicated. Although the community is very intuitive to us, it is difficult to find a unified identity of the community. We usually think that the community is a set of nodes, which is related to other nodes in the same group, this group of nodes is more likely to be assigned to the same community, in other words, the probability that the nodes in the group are paired will be greater. Intuitively, we can use the framework of probabilistic inference to solve the problem of overlapping community detection, which can be attributed to inferring the unobserved attribute matrix from the observed graph structure. Once we posit a community-based generative model $p(\boldsymbol{G}|\boldsymbol{F})$ for the hypergraph, detecting communities boils down to inferring the unobserved affiliation matrix $\boldsymbol{F}$ given the observed hypergraph $\boldsymbol{G}$.

Besides the traditional probabilistic view, one can also view community detection through the lens of representation learning. The community affiliation matrix F can be considered as an embedding of nodes into $\mathbb{R}_{\geq 0}^{C}$, with the aim of preserving the graph structure. Given the recent success of representation learning for graphs [41], a question arises: "Can the advances in deep learning for graphs be used to design better community detection algorithms?". We proved that simply combining existing node embedding approaches with overlapping K-means does not lead to satisfactory results. Instead, we propose to combine the probabilistic and representation learning, and learn the community affiliations in an end-to-end manner using GNN. We used the two-layer hypergraph convolutional neural network [42] as the basis of our model. The HGCN is defined as：

$$\boldsymbol{F} \coloneqq HGCN_{\theta}(\boldsymbol{H}, \boldsymbol{X}) = ReLU\big(\widehat{\boldsymbol{H}} ReLU\big(\widehat{\boldsymbol{H}} \boldsymbol{X} \boldsymbol{P}^{(1)}\big) \boldsymbol{P}^{(2)}\big), \tag{4}$$

We have tested that only use a single HGCN model in our experiments, but simply use one of the branch cannot extract the complete information of the given data. And, some deeper model has been considered in our work, but none of them led to any noticeable improvements. In addition, we also considered Jumping Knowledge Network [8] and GarphSage [40] architectures, but they did not perform well enough on some datasets, so we did not adopt these solutions in the end.

3.4. Bernoulli-Poisson Model

The Bernoulli-Poisson (BP) model [34] [39] [43] is a graph generative model that allows for overlapping communities. According to the BP model, the graph is generated as follows. Given the affiliations $F \in R_{\geq \mathbb{0}}^{N \times C}$, adjacency matrix entries $\boldsymbol{A}_{uv}$ are sampled i.i.d. as

$$\boldsymbol{A}_{uv} \sim Bernoulli\left(\log\left(1 - \exp\left(-\boldsymbol{F}_{\mathrm{u}} \boldsymbol{F}_{\mathrm{v}}^{\mathrm{T}}\right)\right)\right), \tag{5}$$

where F_{u} is the row vector of community affiliations of the node u (the u's row of the matrix F). Intuitively, the more communities' nodes u and v have in common, the more likely they are to be a connected by an edge.

Instead of treating the affiliation matrix $\boldsymbol{F}$ as a free variable over which optimization is performed, we generate F with a HGCN:

$$\boldsymbol{F} \coloneqq HGCN_{\theta}(\boldsymbol{H}, \boldsymbol{X}). \tag{6}$$

A ReLU non-linearity is applied element-wise to the output layer to ensure non-negativity of $\boldsymbol{F}$.

The negative log-likelihood of the Bernoulli-Poisson model is:

$$-\log p\,(\boldsymbol{H} \mid \boldsymbol{F}) = -\sum_{(u,v)\in E} \log\left(1 - \exp\left(-\boldsymbol{F}_{\boldsymbol{u}} \boldsymbol{F}_{\boldsymbol{v}}^{\mathrm{T}}\right) + \sum_{(u,v)\notin E} \boldsymbol{F}_{\boldsymbol{u}} \boldsymbol{F}_{\boldsymbol{v}}^{\mathrm{T}}\right., \tag{7}$$

Real-world graphs are usually extremely sparse, which means that the second term in Eq. (7) will provide a much larger contribution to the loss. We counteract this by balancing the two terms, which is a standard technique in imbalanced classification [44].

$$\mathcal{L}(\boldsymbol{F}) = -E_{(u,v)\sim P_E}[\log(1-\exp(\boldsymbol{F_u}\boldsymbol{F_v}^{\mathrm{T}})] + E_{(u,v)\sim P_N}[\boldsymbol{F_u}\boldsymbol{F_v}^{\mathrm{T}}] \tag{8}$$

where P_E and P_N denote uniform distributions over edges and non-edges respectively.

Instead of directly optimizing the affiliation matrix $\boldsymbol{F}$, as done by traditional approaches [34] [45], we search for neural network parameters $\theta^{\star}$ that minimize the (balanced) negative log-likelihood:

$$\theta^{\star} = arg\min_{\theta} \mathcal{L}\left(HGCN_{\theta}(\boldsymbol{H},\boldsymbol{X})\right) \tag{9}$$

Using a HGCN for community prediction has several advantages. First, due to an appropriate inductive bias, the HGCN gives similar community affiliation vectors for neighbouring nodes, which greatly improves the quality of predictions compared to simpler models. Also, the above formulas can well integrate the node features into the model. Finally, with the formulation from Eq. (6), we can also predict the community of nodes that were not seen during training step. Fig.1 shows the overview of our HOCD model.

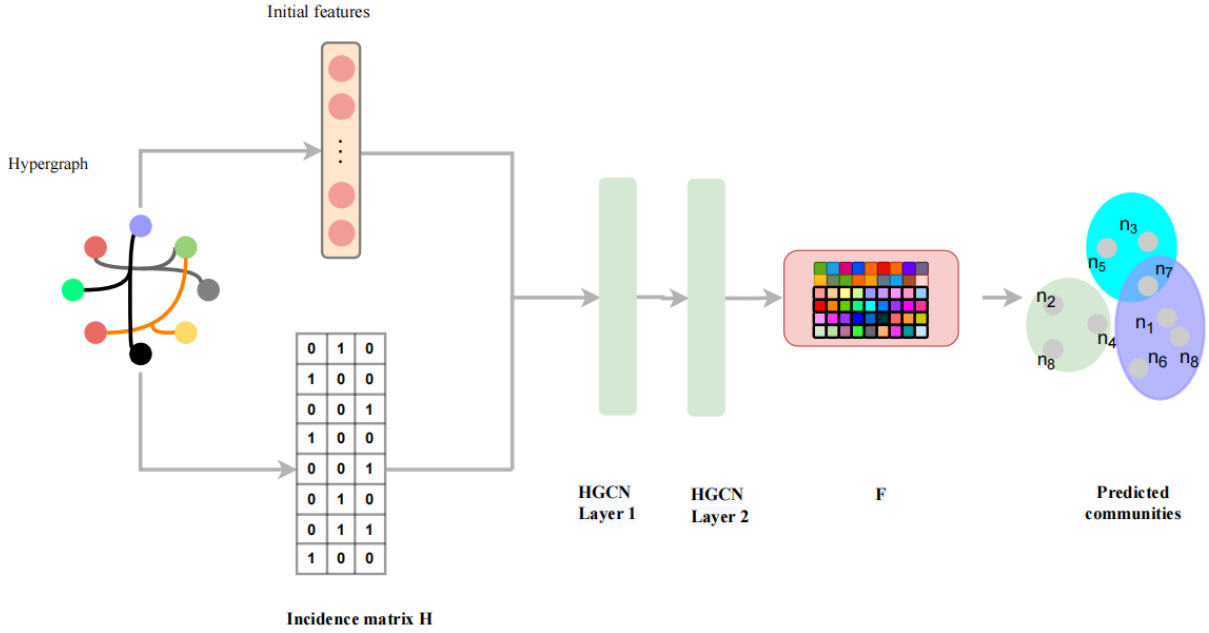

Fig.1: The overview of HOCD model.

4. Experiments

In this section, we will evaluate the performance of our model on a few datasets, we use six social network datasets for a fair comparison with the previous method i.e., GCN.

4.1. Datasets

We use the following datasets, the Facebook dataset is a small collection of ego-networks from the Facebook graph, and the community structures within these datasets are overlapping communities. Larger graph datasets (10K+ nodes) with reliable ground-truth overlapping community information and node attributes are not openly available, which hampers the evaluation and development of new methods. The parameters of some datasets are shown in Table 1.

Table 1: Overview of the datasets (K stands for 1000)

Datasets	Network type	N	M	D	C
Facebook 348	Social	224	46	21	14
Facebook 414	Social	150	1.7K	16	7
Facebook 686	Social	168	1.6K	9	14
Facebook 698	Social	61	270	6	13
Facebook 1684	Social	786	14.0K	15	17
Facebook 1912	Social	747	30.0K	29	46

4.2. Recovery of Ground-truth

We evaluate our model by examining the results of our method in comparison with ground-truth. We choose three overlapping community detection methods based on different paradigms: probabilistic inference, non-negative matrix factorization (NMF), and deep learning methods.

Among the methods given, BigCLAM [34], EMP [35], and SnetOC [33] are based on Bernoulli-Poisson distribution models, CESNA is an extension of BigCLAM, and CESNA additionally models node attributes. SNMF and CDE are non-negative matrix factorization (NMF) methods for overlapping community detection. Previous work also implemented two neural graph embedding-based methods, DeepWalk and Graph2Gauss, to compute node embedding for all nodes in each graph. Graph2Gauss considers both node features and graph structure, while DeepWalk only considers node features. These two methods use non-exhaustive overlapping (NEO) k-means to cluster nodes. Previous work expressed the methods of DeepWalk and Graph2Gauss as DW/NEO and G2G/NEO.

To compare the detected communities with the ground truth, we first need to convert the predicted continuous community affiliations **F** into binary community assignments. We assign node u to community c if its affiliation strength $\boldsymbol{F}_{uc}$ is above a fixed threshold τ, If the membership of the node u to the community c is greater than the threshold τ we set, we will assign the node u to the c community. For different datasets, we dynamically adjust the threshold τ to get the result, and then use it in further experiments without additional tuning. We found that popular metrics for quantifying agreement between true and detected communities, such as Jaccard and F1-scores [30] [35] [45] can give arbitrarily high scores for completely uninformative community assignments. Instead, we use overlapping normalized mutual information (NMI) [39], as it is more robust and meaningful. Table 2 shows the recovery of the ground-truth community of each method.

Table 2: Recovery of ground-truth community, measured by NMI (in %).

Dataset	BigCLAM	CESNA	EPM	SNetOC	CDE	SNMF	DW/NEO	G2G/NEO	NOCD	**HOCD**
Facebook 348	26.0	29.4	6.5	24.0	24.8	13.5	31.2	17.2	36.4	**44.4**
Facebook 414	48.3	50.3	17.5	52.0	28.7	32.5	40.9	32.3	**59.8**	59.7
Facebook 686	13.8	13.3	3.1	10.6	13.5	11.6	11.8	5.6	21.0	**25.4**
Facebook 698	45.6	39.4	9.2	44.9	31.6	28.0	40.1	2.6	41.7	**51.0**
Facebook 1684	32.7	28.0	6.8	26.1	28.8	13.0	37.2	9.9	26.1	**49.3**
Facebook 1912	21.4	21.2	9.8	21.4	15.5	23.4	20.8	16.0	35.6	**41.4**

4.3. Ablation Study

Our model achieved superior performance in community recovery. Intuitively, it makes sense to use a HOCD for the reason laid out in the previous section. However, we wonder that whether it is possible to achieve comparable result with other models. To resolve this problem, we considered the following two baselines. Table 3 shows the comparison of our model with other methods in community recovery. i.e., single feature-based HGCN or graph-based HGCN method, GNN-based approach, Multi-layer perception (MLP) and even simpler model Free Variable (FV).

Table 3: The comparison of HGCN-based method with other baselines in community recovery.

Datasets	HOCD(OURS)	MLP	GNN	Free variable
Facebook 348	44.4 ±1.8	36.4 ±2.0	11.7 ±2.7	25.7 ±1.3
Facebook 414	59.7 ±0.9	59.8 ±1.8	22.1 ±3.1	49.2 ±0.4
Facebook 686	25.4 ±1.2	21.0 ±0.9	1.5 ±0.7	13.5 ±0.9
Facebook 698	51.0 ±1.0	41.7 ±3.6	1.4 ±13	41.5 ±1.5
Facebook 1684	49.3 ±0.8	26.1 ±13	17.1 ±2.0	22.3 ±1.4
Facebook 1912	41.4 ±0.2	35.6 ±1.3	17.5 ±19	18.3 ±1.2

For both the HGCN-based model and GNN-based model, we use a 2-layer hypergraph and graph convolutional neural network with hidden size of 128 and the output (second) layer of size C (number of communities to detect). We apply batch normalization after the first graph convolution layer. Dropout with 50% keep probability is applied before every layer. We add weight decay to both weight matrices with regularization strength $\lambda = 10^{-2}$. The feature matrix X (or A, in case we are working without attribute) is normalized such that every row has unit L_2-norm. For MLP-based model. We found the MLP model to perform best with the same configuration as described above for the GCN model (i.e., same regularization

strength, hidden size, dropout, and batch norm). For Free Variable model, we considered two initialization strategies for the free variable model: (1) Locally minimal neighbourhoods (the strategy used by the BigCLAM and CESNA models.) (2) Initializing F to the output of an untrained GCN.

We found the first strategy to consistently provide better results. As for the training step for HGNN-bases model, GNN-based model, and MLP-based model, we train both models using Adam optimizer [46] with default parameters. The learning rate is set to 10^{-3}. We use the following early stopping strategy: Every 50 epochs we compute the full training loss. We stop optimization if there was no improvement in the loss for the last $10 \times 50 = 500$ iterations, or after 5000 epochs, whichever happens first. For free variable model, we use Adam optimizer with learning rate $5 \cdot 10^{-2}$. After every gradient step, we project the $\boldsymbol{F}$ matrix to ensure that it stays non-negative $\boldsymbol{F}_{uc} = \max\{0, \boldsymbol{F}_{uc}\}$. We use the same early stopping strategy as for the GNN and MLP models.

4.4. Hyperparameter Sensitivity Analysis

There is an important hyperparameter to evaluate the results. Fig.2 shows the performance of NMI on different datasets using different thresholds. When the threshold is set to 0.5, we can see three out of six datasets achieve the best results. It is generally in line with a trend of first increase and then decrease. For facebook 348 dataset, facebook 686 dataset and facebook 698 dataset, the performance reaches the best when the threshold is set to 0.3, 0.1 and 0.6 respectively. When the threshold is set to 1, the performance of all datasets is in the lowest level. This is not an unexplainable phenomenon, because it is very difficult for us to make our predicted probability reach to 1.

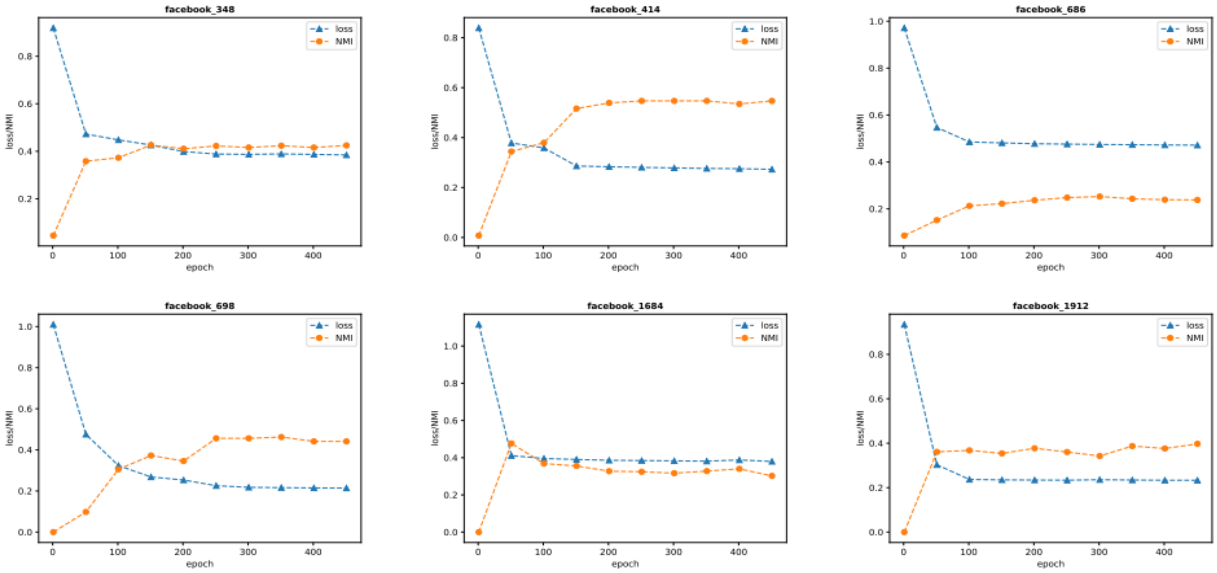

Fig. 2: The performance of NMI on different datasets using different thresholds.

5. Conclusion

We propose an accurate, flexible and scalable model HOCD, a graph neural network model for overlapping community detection. Experimental results show that the model is efficient. In addition to strong empirical results, our work raises interesting follow-up questions. We will continue the study of hypergraph convolution in overlapping community detection later. There are other variants in the graph convolutional neural network family. We are going to add graph attention network (GAT) to hypergraph convolution in the neural network, rather than adding to the hypergraph convolutional neural network, in other words, the hypergraph convolutional neural network has an innate attention mechanism [12]. In the hypergraph convolution, for a given vertex, the incoming and the outgoing information streams are all assigned different importance, and the goal of hypergraph attention is to learn a dynamic association matrix. Thus, a dynamic transition matrix between vertices can be better revealed. To summarize, the results obtained in this paper provide strong evidence that deep learning for graphs deserves more attention as a framework for overlapping community detection.

Acknowledgements

This work was supported by NSFC (62076005, 61906002, U20A20398), NSF of Anhui Province (2008085MF191, 2008085QF306, 2008085UD07), and the University Synergy Innovation Program of Anhui Province (GXXT-2021-002).

References

[1] D. C. G. Pedronette, F. M. F. Gon¸calves, and I. R. Guilherme, "Unsupervised manifold learning through reciprocal knn graph and connected components for image retrieval tasks," Pattern Recognition, vol. 75, pp. 161–174, 2018.

[2] D. C. G. Pedronette and R. d. S. Torres, "Image re-ranking and rank aggregation based on similarity of ranked lists," Pattern Recognition, vol. 46, no. 8, pp. 2350–2360, 2013.

[3] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in Proceedings. 2005 IEEE international joint conference on neural networks, vol. 2, no. 2005, 2005, pp. 729–734.

[4] Mikolov, Tomas, Martin Karafiát, Lukáš Burget, Jan Honza Černocký and Sanjeev Khudanpur. "Recurrent neural network based language model." Interspeech (2010).

[5] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," arXiv preprint arXiv:1312.6203, 2013.

[6] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," arXiv preprint arXiv:1506.05163, 2015.

[7] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," Advances in neural information processing systems, vol. 29, 2016.

[8] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," arXiv preprint arXiv:1609.02907, 2016.

[9] J. Atwood and D. Towsley, "Diffusion-convolutional neural networks," Advances in neural information processing systems, vol. 29, 2016.

[10] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," Advances in neural information processing systems, vol. 30, 2017.

[11] P. Veli˘ckovi´c, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," arXiv preprint arXiv:1710.10903, 2017.

[12] J. Zhang, Y. Chen, X. Xiao, R. Lu, and S.-T. Xia, "Learnable hypergraph laplacian for hypergraph learning," in ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2022, pp. 4503–4507.

[13] D. Zhou, J. Huang, and B. Sch¨olkopf, "Learning with hypergraphs: Clustering, classification, and embedding," Advances in neural information processing systems, vol. 19, 2006.

[14] Y. Huang, Q. Liu, and D. Metaxas, "video object segmentation by hypergraph cut," in 2009 IEEE conference on computer vision and pattern recognition. IEEE, 2009, pp. 1738–1745.

[15] Y. Huang, Q. Liu, S. Zhang, and D. N. Metaxas, "Image retrieval via probabilistic hypergraph ranking," in 2010 IEEE computer society conference on computer vision and pattern recognition. IEEE, 2010, pp. 3376–3383.

[16] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 5115–5124.

[17] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," IEEE transactions on neural networks, vol. 20, no. 1, pp. 61–80, 2008.

[18] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," Advances in neural information processing systems,vol. 30, 2017.

[19] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in International conference on machine learning. PMLR, 2018, pp. 5453–5462.

[20] A. Bojchevski and S. G¨unnemann, “Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking,” arXiv preprint arXiv:1707.03815, 2017.

[21] A. Grover and J. Leskovec, “node2vec: Scalable feature learning for networks,” in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855–864.

[22] T. N. Kipf and M. Welling, “Variational graph auto-encoders,” arXiv preprint arXiv:1611.07308, 2016.

[23] B. Perozzi, R. Al-Rfou, and S. Skiena, “Deepwalk: Online learning of social representations,” in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 701–710.

[24] M. E. Newman and M. Girvan, “Finding and evaluating community structure in networks,” Physical review E, vol. 69, no. 2, p. 026113, 2004.

[25] N. J. Krogan, G. Cagney, H. Yu, G. Zhong, X. Guo, A. Ignatchenko, J. Li, S. Pu, N. Datta, A. P. Tikuisis et al., “Global landscape of protein complexes in the yeast saccharomyces cerevisiae,” Nature, vol. 440, no. 7084, pp. 637–643, 2006.

[26] S. Cavallari, V. W. Zheng, H. Cai, K. C.-C. Chang, and E. Cambria, “Learning community embedding with community detection and node embedding on graphs,” in Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 377–386.

[27] J. J. Choong, X. Liu, and T. Murata, “Learning community structure with variational autoencoder,” in 2018 IEEE international conference on data mining (ICDM). IEEE, 2018, pp. 69–78.

[28] L. Yang, X. Cao, D. He, C. Wang, X. Wang, and W. Zhang, “Modularity based community detection with deep learning.” in IJCAI, vol. 16, 2016, pp. 2252–2258.

[29] D. Kuang, C. Ding, and H. Park, “Symmetric nonnegative matrix factorization for graph clustering,” in Proceedings of the 2012 SIAM international conference on data mining. SIAM, 2012, pp. 106–117.

[30] Y. Li, C. Sha, X. Huang, and Y. Zhang, “Community detection in attributed graphs: An embedding approach,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1, 2018.

[31] F. Wang, T. Li, X. Wang, S. Zhu, and C. Ding, “Community discovery using nonnegative matrix factorization,” Data Mining and Knowledge Discovery, vol. 22, no. 3, pp. 493–521, 2011.

[32] P. Latouche, E. Birmel´e, and C. Ambroise, “Overlapping stochastic block models with application to the french political blogosphere,” The Annals of Applied Statistics, vol. 5, no. 1, pp. 309–336, 2011.

[33] A. Todeschini, X. Miscouridou, and F. Caron, “Exchangeable random measures for sparse and modular graphs with overlapping communities,” arXiv preprint arXiv:1602.02114, 2016.

[34] J. Yang and J. Leskovec, “Overlapping community detection at scale: a nonnegative matrix factorization approach,” in Proceedings of the sixth ACM international conference on Web search and data mining, 2013, pp. 587–596.

[35] M. Zhou, “Infinite edge partition models for overlapping community detection and link prediction,” in Artificial intelligence and statistics. PMLR, 2015, pp. 1135–1143.

[36] E. Galbrun, A. Gionis, and N. Tatti, “Overlapping community detection in labeled graphs,” Data Mining and Knowledge Discovery, vol. 28, no. 5, pp. 1586–1610, 2014.

[37] D. F. Gleich and C. Seshadhri, “Vertex neighborhoods, low conductance cuts, and good seeds for local community methods,” in Proceedings of the18th ACM SIGKDD international conference on Knowledge discovery and data mining, 2012, pp. 597–605.

[38] Y. Li, K. He, D. Bindel, and J. E. Hopcroft, “Uncovering the small community structure in large networks: A local spectral approach,” in Proceedings of the 24th international conference on world wide web, 2015, pp. 658–668.

[39] A. F. McDaid, D. Greene, and N. Hurley, “Normalized mutual information to evaluate overlapping community finding algorithms,” arXiv preprint arXiv:1110.2515, 2011.

[40] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, and S. Jegelka, “Representation learning on graphs with jumping knowledge networks,” in International conference on machine learning. PMLR, 2018, pp. 5453–5462.

[41] H. Cai, V. W. Zheng, and K. C.-C. Chang, “A comprehensive survey of graph embedding: Problems, techniques, and applications,” IEEE Transactions on Knowledge and Data Engineering, vol. 30, no. 9, pp. 1616–1637, 2018.

[42] S. Bai, F. Zhang, and P. H. Torr, “Hypergraph convolution and hypergraph attention,” Pattern Recognition, vol.

110, p. 107637, 2021.

[43] A. Todeschini, X. Miscouridou, and F. Caron, “Exchangeable random measures for sparse and modular graphs with overlapping communities,” arXiv preprint arXiv:1602.02114, 2016.

[44] H. He and E. A. Garcia, “Learning from imbalanced data,” IEEE Transactions on knowledge and data engineering, vol. 21, no. 9, pp. 1263–1284, 2009.

[45] J. Yang, J. McAuley, and J. Leskovec, “Community detection in networks with node attributes,” in 2013 IEEE 13th international conference on data mining. IEEE, 2013, pp. 1151–1156.

[46] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.