CommDGI: Community Detection Oriented Deep Graph Infomax

Tianqi Zhang
tqzhang18@fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
China

Yun Xiong*
yunx@fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
Shanghai Institute for Advanced Communication and Data Science
China

Jiawei Zhang
jiawei@ifmlab.org
IFM Lab, Department of Computer Science, Florida State University
FL, USA

Yao Zhang
yaozhang18@fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
China

Yizhu Jiao
yzjiao18@fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
China

Yangyong Zhu
yyzhu@fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
Shanghai Institute for Advanced Communication and Data Science
China

ABSTRACT

Graph Neural Networks(GNNs), like GCN and GAT, have achieved great success in a number of supervised or semi-supervised tasks including node classification and link prediction. These existing graph neural networks can effectively encode neighborhood information of graph nodes through their message aggregating mechanisms. However, there are some unsupervised and structure-related tasks like community detection, which is a fundamental problem in network analysis that finds densely-connected groups of nodes and separates them from others in graphs. It is still difficult for these general-purposed GNNs to learn the needed structural information in these particular problems. To overcome the shortcomings of general-purposed graph representation learning methods, we propose the Community Deep Graph Infomax (CommDGI), a graph neural network designed to handle community detection problems. Inspired by the success of deep graph infomax in self-supervised graph learning, we design a novel mutual information mechanism to capture neighborhood as well as community information in graphs. A trainable clustering layer is employed to learn the community partition in an end-to-end manner. Disentangled representation learning is applied in our graph neural network so that the model can improve interpretability and generalization. Throughout the whole learning process, joint optimization is applied to learn the community-related node representations. The experimental results show that our algorithm outperforms state-of-the-art community detection methods.

CCS CONCEPTS

• **Computer systems organization** → **Neural networks**; • **Information systems** → *Clustering*; • **Computing methodologies** → *Knowledge representation and reasoning*.

KEYWORDS

Graph Neural Networks, Community Detection, Graph Clustering, Self-supervised Learning

ACM Reference Format:
Tianqi Zhang, Yun Xiong, Jiawei Zhang, Yao Zhang, Yizhu Jiao, and Yangyong Zhu. 2020. CommDGI: Community Detection Oriented Deep Graph Infomax. In *The 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3340531.3412042

1 INTRODUCTION

With the boom of rich data available for researchers in real-world networks, it is vital to analyze attributed graphs which tend to contain more helpful information. For the past few years, graph neural networks [9, 14, 27] have shown their striking ability to tackle a series of representation learning related tasks such as node classification and link prediction [33] in attributed graphs.

Despite their great success, the representations learned from these models are not a panacea for every downstream task. Most of the real-world networks display the community structure. Community detection, also known as graph clustering, can find such structural feature and serve as a crucial analysis tools to have an insight into complex networks or graphs [4, 22, 32]. A common definition of community detection is to partition graph nodes into

*Corresponding author.

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3412042

several disjoint groups, where internal nodes are more similar to each other than external nodes. Spontaneously, with the development of network analysis, using graph neural network to exploring communities in attributed graphs becomes a new trend for researchers [7, 12, 25].

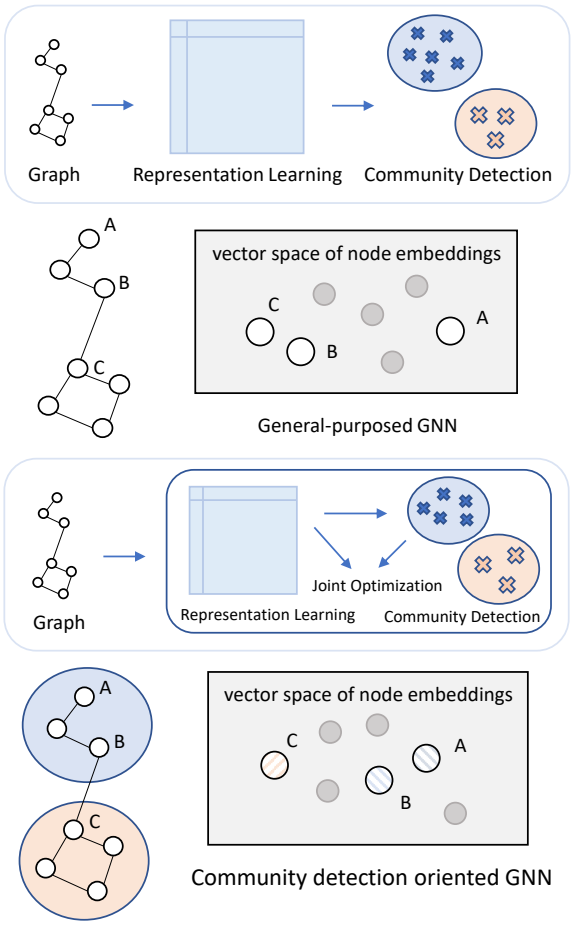

Figure 1: An example to illustrate the shortcoming of previous representation-learning-based community detection methods.

Community detection is generally defined as an unsupervised task, where no label information can be used to train a graph neural network. A natural way to do community detection is to learn node embeddings in an unsupervised manner and then perform clustering algorithms on the node embeddings. Recently, some unsupervised graph neural networks [23, 28] have achieved fairly high performance on node classification tasks, like Deep Graph Infomax (DGI) [28]. DGI extracts self information of graph as well as each node representation and then maximize the mutual information between graph representation and node representations. DGI can perform self-supervised learning through introduceing contrastive training [19, 20] methods, which help to learn effective node representations to deal with a number of downstream tasks. Simple clustering methods like K-means can be performed on the learned representations to do the community detection. However, this straightforward two-step approach brings about a problem that the clustering step can hardly optimize the node embedding learning.

Figure 1 is an example to illustrate the shortcoming of general-purposed graph neural networks and two-step approaches on community detection tasks. In a general-purposed graph neural network, community detection is performed sequentially so that the encoder tend to cast the neighbor nodes to the closer vector space. However, community detection problem requires more higher-order structural information and the nodes in one community without direct edge may be more similar to each other than the neighbor nodes. It is necessary to perform community detection problems in a community detection oriented graph neural network.

Some end-to-end graph neural network frameworks have been proposed lately to replace two-step community detection approaches, most of them are graph autoencoder [15] based and focus on the adjacency matrix reconstruction [8]. The disadvantage of these methods is that they directly leverage an edge-related clustering loss to learn the data representation with high cluster cohesion but lack an effective way to capture the higher-order structural relationship between nodes and communities, which may still lead to the under-expression of node representations.

In this paper, we still study community detection problems with graph neural networks. But we want to capture community-related characteristics more comprehensively with the help of community oriented optimization. It is challenging due to the following reasons:

- **Graph learning without labels**: Generally, community detection tasks are performed without labels which means the neural work itself is expected to generate some self-supervised information which can encourage the output of the encoder to have desired characteristics.
- **Community-independent representations**: The representations learned by general-purposed graph neural networks cannot capture clustering and community structural information. It is necessary but challenging to combine the clustering process with the graph neural network to better handle the community detection problems.
- **Entangled representations**: The representations learned by graph neural networks under matrix reconstruction objectives are entangled and has poor interpretability.
- **End-to-end learning**: Unifying the graph and clustering learning is challenging but useful to a community oriented graph neural network. The clustering assignment can help update the node representation in a community-related way.

In order to overcome the challenging issues mentioned above, we propose a new community oriented graph neural network, Community Deep Graph Infomax (CommDGI). To encode node structural and community-aware representations in our framework, mutual information maximization [2] is used to capture local and global structural information. Unlike DGI, our model proposes a new mutual information maximization paradigm. Graph mutual information, similar to the one used in DGI, is calculated between node and graph while community mutual information is calculated between node and community (sub-graph). These two kinds of mutual

information are maximized together to encode the graph and clustering feature of each node. We also use a trainable clustering layer in our graph neural network so that the graph neural network can be trained in an end-to-end manner for community detection tasks, and the classical community detection optimization goal modularity [21] is employed to learn more structural feature with mutual information together through the soft K-means assignment in clustering layer.

To improve the generalization and interpretability of entangled representations, we propose a disentangled learning method in our graph neural network as well. Recent studies [13, 18] have shown that disentangled representation learning is able to bring enhanced generalization ability and be more interpretable especially in some scenarios where extracting information beyond the local neighborhood is needed [17]. Our model assumes that the representation of nodes can be split into multiple channels and through the disentangled operation, the whole model can learn more latent information.

Furthermore, the improvements mentioned above can be unified with joint optimization, and the representation of nodes as well as the community assignment can be learned together.

Our main contributions are summarized as follows:

- We propose a novel graph neural network Deep Community Graph Infomax (CommDGI) for community detection. The proposed model can effectively encode node attributes in a self-supervised manner with the idea of DGI's graph mutual information maximization. To our knowledge, it is a pretty new attempt to combine community detection with mutual information maximization.
- We add a trainable clustering layer into the graph neural network. We employ modularity objective and design a new community detection objective on attributed graphs named community mutual information to help learn the community-related node representations.
- We use disentangled representation while training and aggregating node information in the clustering layer, which can improve the interpretablity of the whole model.
- Our model can learn the community partition directly and use multiple objectives to optimize the node representation. The clustering step and node representation learning step are optimized together.
- Experiments on real-world datasets demonstrate the superiority of CommDGI in comparison with the state-of-the-art community detection methods.

2 PRELIMINARY

Definition 1. (Attributed Graphs): An attributed graph is defined as $G = (V, E, X)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of n nodes with $|V| = n$, $E \subseteq V \times V$ is the edge set, and $X = \{x_1, x_2, \ldots, x_n\}$ are the attribute values where $x_i \in \mathbb{R}^d$, x_i is an attribute vector associated with v_i, and d is the dimension of attribute vectors. In addition, adjacency matrix $A \in \{0, 1\}^{n \times n}$ is used to describe the observed graph structure. If an edge $(v_i, v_j) \in E$, $A_{ij} = 1$; otherwise, $A_{ij} = 0$. Throughout this paper, we restrict our discussions on undirected and unweighted attributed graphs.

Table 1: Descriptions of frequently-used notations in this framework

Notation	Description
A_{ij}	Adjacency matrix value between node v_i and node v_j
v_i	Node i in Graph G
x_i	Original features of node v_i
h_i	Learned representations of node v_i
h_{ik}	Learned representations of node v_i in the k-th channel
μ_k	The cluster center of community k
μ_{kk}	The k-th channel of the cluster center of community k
r_{ik}	The degree to which node v_i is assigned to cluster k
$R = \{r_{ik}\}$	The entire assignemnt matrix
d_i	The degree of node v_i
s	The summary vector of graph G
D	Degree matrix
$\mathcal{D}$	Discriminator
$\mathcal{E}$	Encoder
$\mathcal{R}$	Readout function

Problem Definition: In this paper, we consider the community detection problems in attributed graphs. Given an attributed graph G and the number of communities K, the community detection method in this paper aims to find a function $f : V \to \{1, 2, \ldots, K\}$ such that for all nodes satisfying $f(v_i) = k$ belong to the k-th community. The partition of function f should follow these principles: (1) The nodes in the same group are closely connected to each other while the nodes in different groups are the opposite. (2) The nodes in the same community tend to have similar attribute values while the nodes in different communities may have relatively diverse attribute values even they are neighbors in the graph level. (3) Function f can properly preserve the node attribute and structural information of the attributed graph G. Finally, we can find out disjoint groups of nodes and their induced subgraphs, i.e., communities.

3 THE PROPOSED MODEL

In this section, we go into the details of the proposed model CommDGI. For the convenience of elaboration, we provide the frequently-used notations through this paper in Table 1. We will detail each component of CommDGI and summarize the complete procedure of CommDGI in pseudo code at the end of this section.

3.1 Graph Infomax Layer

To assign nodes into different communities, we should encode the attributes of each node first. We construct the graph infomax layer with a general unsupervised learning graph neural network. The idea of this layer is similar to DGI that we maximize the graph MI between the graph-level representation and node-level representations.

In order to capture node-level information, our encoder $\mathcal{E}$ is a Graph Convolutional Network (GCN) [14] model with the following propagation rule:

$$H = \mathcal{E}(X, A) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}X\Theta). \quad (1)$$

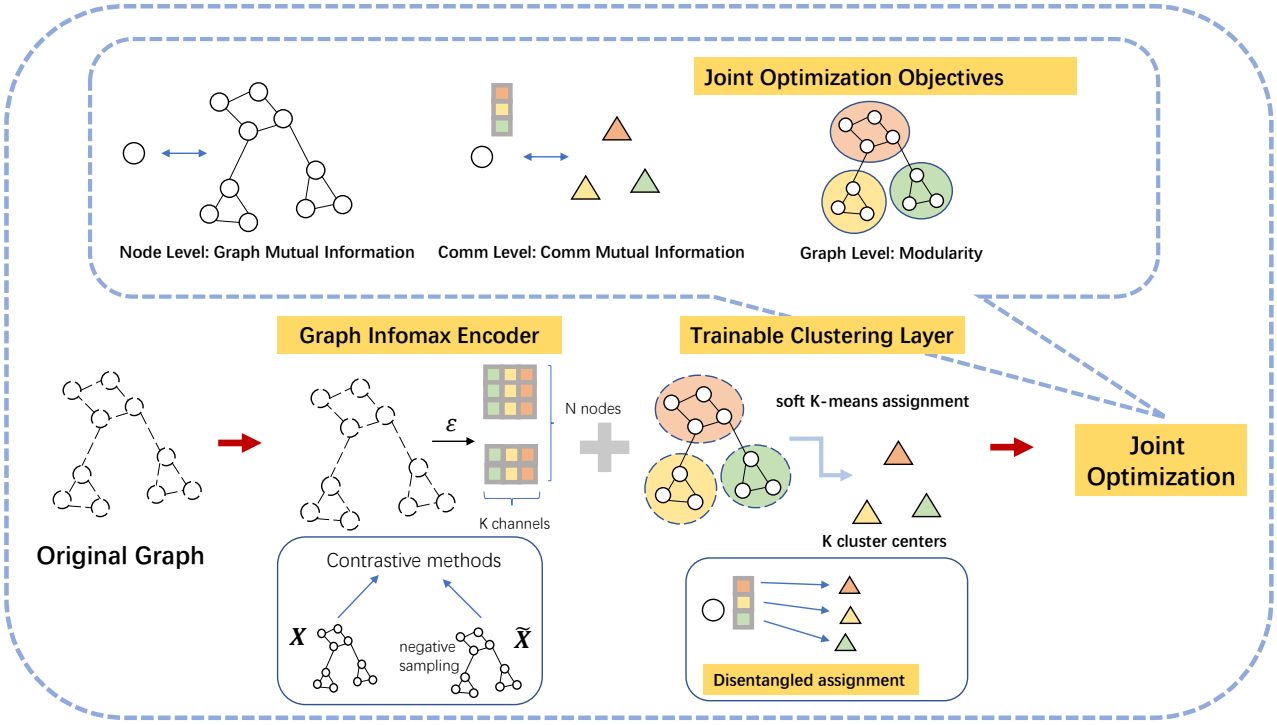

Figure 2: The conceptual framework of Community Deep Graph Infomax (CommDGI). Given an attributed graph, CommDGI learns a hidden node representation through a deep graph infomax encoder, and manipulates it with a trainable clustering layer, which is optimized together with the graph encoder and perform clustering assignment during training. We choose the objectives from three scales including node level(graph MI), community level(community MI) and graph level(modularity).

Here, $\hat{A} = A + I_n$ is the adjacency matrix with inserted self-loop and $\hat{D}$ is degree matrix where $\hat{D}_{ii} = \sum_j \hat{A}_{ij}$. For the nonlinearity, σ is the parametric ReLU(PReLU) function [10] and Θ is a learnable linear transformation applied to every node. $H = \{h_1, h_2, \ldots, h_n\}$ represents higher-level representations $h_i \in \mathbb{R}^{d'}$ of each node v_i.

We apply contrastive methods here to help urge the learned node representations to better capture the structural similarities of different nodes in the attributed graph. The negative sampled adjacency matrix $\tilde{A}$ is the same as original adjacency matrix A while the original features X are randomly shuffled in respective rows to get negative sampled features $\tilde{X}$. We can learn negative sampled node representations $\tilde{H}$ with $\mathcal{E}(\tilde{X}, \tilde{A})$.

In order to acquire graph-level representation, we leverage a readout function $\mathcal{R} : \mathbb{R}^{n \times d'} \rightarrow \mathbb{R}^{d'}$ to summarize the node representations into a graph-level representation $s = \mathcal{R}(\mathcal{E}(X, A))$. The choice of $\mathcal{R}$ in our CommDGI will be discussed in the parameter settings (Section 4.3).

We use a discriminator to help maximize the local mutual information, $\mathcal{D} : \mathbb{R}^{d'} \times \mathbb{R}^{d'} \rightarrow \mathbb{R}$, such that $\mathcal{D}(h_i, S)$ represents the probability score assigned to this patch-summary pair (should be higher for patches contained within the summary). Specifically, we implement the discriminator $\mathcal{D}$ with a dot-product of two vectors followed by a sigmoid function.

The graph MI objective uses a noise-contrastive type objective [20] with a standard binary cross-entropy (BCE) loss between the samples from the joint (positive examples) and the product of marginals (negative examples). We propose the follow graph MI loss:

$$\mathcal{L}_{graph} = \frac{1}{N+M}\left(\sum_{i=1}^{N} \mathbb{E}_{(X,A)}[\log \mathcal{D}(h_i, s)] + \sum_{j=1}^{M} \mathbb{E}_{(\tilde{X},\tilde{A})}[\log(1 - \mathcal{D}(\tilde{h}_j, s))]\right). \tag{2}$$

Here, N and M represent the number of positive samples and negative samples.

3.2 Trainable Clustering Layer and Community-oriented Objectives

In this clustering layer, we implement a differentiable K-means clustering [30], which produces a soft assignment of the nodes to clusters, along with the cluster centers in embedding space.

In community detection problems, the cluster assignment of nodes is equivalent to assigning the nodes to communities. When we update the model, the node assignment of clustering layer is trained and this is the community assignment we want to learn

from the whole framework. We first introduce the forward pass of this layer.

Considering a node v_i and h_i is its representation, μ_k is the cluster center of cluster k. $R = \{r_{ik}\}$ is the assignment matrix of the soft K-means, where r_{ik} is the degree to which node v_i is assigned to cluster k. In the classic hard K-means, r_{ik} is a binary number, but in this layer we let it be a real number satisfying $\sum_k r_{ik} = 1$. Specifically, we use a soft-min assignment to assign each point to the cluster centers based on distance. We use norm $\|\cdot\|$ as negative cosine similarity due to its strong empirical performance. δ is an inverse-temperature hyperparameter which defines how hard to employ the clustering process. We can optimize the cluster centers via an iterative process analogous to the typical K-means updates by alternately setting:

$$\mu_k = \frac{\sum_i r_{ik} h_i}{\sum_i r_{ik}}, \forall k = 1, \ldots, K, \tag{3}$$

$$r_{ik} = \frac{exp(-\delta \|h_i - \mu_k\|)}{\sum_{z=1}^{K} exp(-\delta \|h_i - \mu_z\|)}, \forall k = 1, \ldots, K. \tag{4}$$

These iterates converge to a fixed point where μ remains the same between successive updates. The backward pass of this layer can be effectively approximated without unrolling the entire iteration trajectory [30].

Having obtained the cluster assignment R and the cluster centers μ in a differentiable manner, we need a way to differentiably interpret the clustering as a soft solution to the optimization problem and differentiate a relaxation of the objective value of the graph optimization problem in terms of that solution. We propose a community oriented MI in our framework. Community MI can encourage the representations to encode more community-related characteristic of nodes in the graph learning.

Cluster centers μ, node representations h_i and node assignment r_{ik} are already known in the layers mentioned above. We define the Community MI objective as following equation:

$$\mathcal{L}_{community} = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{E}_{(H,\mu)} [\log \mathcal{D}(r_{ik} h_i, \mu_k)]. \tag{5}$$

Community MI in CommDGI takes node representations as the local information and cluster centers as the community-level information. We still use a discriminator $\mathcal{D}$ here to represent the probability score assigned to every patch-summary pair.

Similar to the intention of graph MI maximization, community MI maximization estimation encourages the framework to learn more community-level information of each node while updating so that the learned embeddings of CommDGI can be community detection oriented and different from the general-purposed graph neural networks.

Modularity is a classical objective in community detection optimization which estimates the quality of a partition of the network in communities. The general expression of modularity is:

$$Q = \frac{1}{2m} \sum_{ij} \sum_{k=1}^{K} (A_{ij} - \frac{d_i d_j}{2m}) \hat{r}_{ik} \hat{r}_{jk}, \tag{6}$$

where m is the number of edges of the network, the sum runs over all pairs of nodes v_i and v_j, A_{ij} is the element of adjacency matrix, d_i is the degree of node i and $\hat{r}_{ik}$ is 1 if node v_i is assigned to community k and 0 if not. This measures the number of edges within communities compared to the expected number if edges were placed randomly.

We can apply the modularity objective into our clustering layer as:

$$\mathcal{L}_{modularity} = \frac{1}{2m} \mathrm{Tr}[R^T (A_{ij} - \frac{d_i d_j}{2m}) R], \tag{7}$$

which is the expected value of a partition sampled according to assignment R.

Modularity is a sort of distance between the actual network and an average over random networks, ignoring altogether the distribution of the relevant community variables, like the fractions of edges within the clusters, over all realisations generated by the configuration model. If the distribution is not strongly peaked, the values of the community variables measured on the original graph may be found in a large number of randomised networks, even though the averages look far away from them. So we pay more attention to the significance of the modularity maximization in our model than to the modularity value itself.

By comparison, Graph MI and community MI pay a lot of attention on the node attributes learning while modularity is more edge-related. By employing the modularity objective in our joint learning, CommDGI can capture more edge information and graph-level partition information than other graph neural network commnity detection approaches.

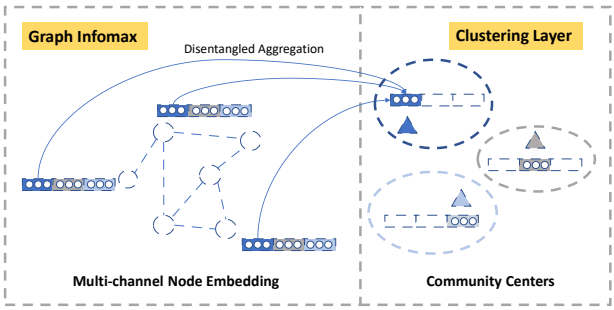

Figure 3: An example to illustrate the disentangled operation of CommDGI.

3.3 Disentangled Learning

The existing graph neural networks generally take a holistic approach to representation learning: the representation learned for a node describes the node's neighborhood as a perceptual whole. Therefore, the nuances between the different parts of the neighborhood are ignored. However, the formation of a graph always display a community structure which indicates the node in a real-world network usually connects with others for various reasons (e.g., company, interest, family), and therefore possesses a neighborhood consisting of several different components.

In the community detection problem, a disentangled representation can be more explicable. We can naturally assume that the node

in the same community tend to have the similar representation on a particular aspect. And the determined aspect of representation of different communities are diverse. A node may be assigned a particular community due to its feature on a particular component (we call it a special channel in this paper).

The feasibility of disentangled representation has been proved in DisenGCN [18]. In this paper, we propose a simplified version of disentangled representation learning aiming to effectively improve the community detection related generalization of our graph neural network.

We define the representation of node v_i as $h_i = \{h_{i1}, h_{i2}, \ldots, h_{iK}\}$ where $h_i \in \mathbb{R}^{d'}$ and $h_{ik} \in \mathbb{R}^{\frac{d'}{K}}$ represents the representation on the k-th channel related to community k. The representation of node v_i, h_i, is a concat result of $\{h_{i1}, h_{i2}, \ldots, h_{iK}\}$.

The number of channels is determined by the number of communities before graph learning. We rewrite the node assignment formula of the clustering layer after applying the disentangled operation:

$$\mu_{kk} = \frac{\sum_i r_{ik} h_{ik}}{\sum_i r_{ik}}, \forall k = 1 \ldots K \tag{8}$$

where μ_{kk} donates the k-th channel of the cluster center representation of community k. The rest channels of μ_k are defined by the mean value of each node representation. In another word, we only aggregate the information related to community k into the community center embedding.

Through this disentangled representation aggregation, the model's community MI can be updated in a different way and our CommDGI can learn the disentangled information of nodes and communities, which is able to capture more specific community-related features beyond the traditional learning pattern.

Algorithm 1 Community Deep Graph Infomax

Require: Graph G with n nodes; number of communities K; Number of iterations $iter$; Target distribution update interval T.
Ensure: Node embeddings H, final community detection partition results R.

```
for l = 1 to iter − 1 do
    Update the node representations H by Eq.(1);
    Initial the cluster centers μ by K-means++ [1];
    Aggregate the node representations H to community centers
    μ in a disentangled manner by Eq.(8);
    if l%T == 0 then
        Calculate the target community assignment R by Eq.(4);
    end if
    Calculate three objective loss, graph MI, community MI and
    modularity by Eq.(2) Eq.(5) Eq.(7);
    Update the whole framework by maximizing Eq.(9);
end for
Get the clustering results with final optimization by Eq.(10).
```

3.4 Joint Optimization

We jointly optimize the graph representation learning and clustering layer, and define our total objective function as:

Table 2: Benchmark Graph Datasets

Dataset	#Nodes	#Links	#Communities	#Features
Cora	2,708	5,429	7	1,433
Citeseer	3,327	4,732	6	3,703
Pubmed	19,717	44,378	3	500

$$\mathcal{L} = \alpha \mathcal{L}_{graph} + \beta \mathcal{L}_{community} + \mathcal{L}_{modularity}, \tag{9}$$

where $\mathcal{L}_{graph}$ is the graph MI loss, $\mathcal{L}_{community}$ is the community MI loss and $\mathcal{L}_{modularity}$ is the modularity loss. α and β are the coefficients that control the balance of three objectives. It is worth mentioning that we could gain our community detection result directly from the last optimized result, and the label estimated for node i could be obtained as:

$$c_i = \arg\max_k r_{ik}, \tag{10}$$

which is the most likely assignment from the last soft K-means assignment distribution R.

Our method is summarized in Algorithm 1. Our algorithm has the following advantages:

- **Unsupervised Graph Learning**: The graph infomax layer can effectively aggregate the node information by contrastive training and graph MI.
- **Clustering Oriented Representations**: The proposed unsupervised learning clustering component manipulates the embedding to improve the clustering performance.
- **Disentangled Learning**: The disentangled node learning and aggregation can be leveraged to more comprehensively describe the node relationship with every community.
- **Joint & End-to-end Optimization**: The framework jointly optimizes the three parts of the loss functions, learns the embedding and performs community detection in a unified framework.

4 EXPERIMENTS

4.1 Datasets

We use three standard citation networks, Cora, Citeseer and Pubmed, which are widely-used for assessment of attributed graph analysis in our experiments. Detailed information of the datasets is summarized in Table 2, and publications in the datasets are categorized by the research sub-fields.

4.2 Comparison Models

We compared a total of eight algorithms with our method in the experiments. The community detection algorithms cover the approaches that only use node attributes or adjacency matrix information, and also include approaches that integrate both. Deep graph representation learning-based community detection algorithms were also compared.

Detailed community detection methods used in the experiment are listed as follows(summarized in Table 3):

- **K-means++**: K-means is one of the most widely-used clustering algorithm, and we use the K-means++, which employs

Table 3: Comparision of different models

Method	Node Attributes	Edge Information	Two step	End-to-end
K-means++	√	×	√	×
DeepWalk	×	√	√	×
ComE	√	√	√	×
CDE	√	√	×	√
vGraph	√	√	×	√
DAEGC	√	√	×	√
AGC	√	√	×	√
CommDGI	√	√	×	√

the K-means with careful seeding, as a community detection baseline algorithm here. This method only makes use of node attributes as shown in Table 3.

- **DeepWalk** [24]: DeepWalk is a widely-used structural representation learning methods based on random walk.
- **ComE** [6]: ComE is a node representation learning methods based on community embedding learning.
- **CDE** [16]: CDE is a community structure embedding method to encode community structures based on nonnegative matrix factorization optimization.
- **vGraph** [26]: vGraph is a probabilistic generative model to learn community membership and node representation collaboratively.
- **DAEGC** [29]: DAEGC is a goal-directed graph clustering approach employing an attention network to encode the importance of the neighboring nodes to a target node and decoder is trained to reconstruct the graph structure.
- **AGC** [34]: AGC is an adaptive graph convolution method for attributed graph clustering that exploits high-order graph convolution to capture global cluster structure and adaptively selects the appropriate order for different graphs.
- **CommDGI**: CommDGI is the proposed model of this paper, which is a community detection oriented graph neural network.

4.3 Evaluation Metrics and Parameter Settings

In our experiment, we will target on the community detection tasks in attributed graphs and concern the performance of all the comparison community detection methods. We use four clustering metrics to evaluate the community detection result: clustering accuracy (ACC), adjusted rand index (ARI), F-score and normalized nutual information (NMI). A better community detection model should lead to the higher values for all the metrics.

All the existing links in the citation networks are used to construct the adjacency matrix, and sub-field labels are used as the ground-truth community assignment.

For end-to-end baseline methods, we carefully choose the best metrics for each model. For embedding-based baseline methods, we apply a K-means algorithm with 300 iterations after the node representation learning to achieve the community assignment of each node. For ComE algorithm, we leak some label information to learn the better node representations.

For our community detection framework CommDGI, we use an one-layer graph convolutional network as the encoder of graph infomax layer in the experiment. A multi-layer GCN can also work well in our framework but may not improve the learned representation too much with the increase of training iterations. The learnable linear transformation of the encoder computed the node features with $d'_{ij} = 64$ in each disentangled channels and we calculate the average result of the patched node representations as the readout function $\mathcal{R}$ to get the graph-level representation. The initial cluster centers are chosen by the idea of K-means++, which can help accelerate the convergence of the model. Clustering coefficient δ is set to 30 here. Objective parameters are set as $\alpha = 2$ and $\beta = 5$. We train our model for 1000 iterations using the Adam optimizer with the learning rate of 0.001 and the weight decay of 0.2.

4.4 Experiment Result

In the community detection task, we compare the performance of eight different community detection methods. Table 4 shows the performance of our model and other seven baseline methods evaluated by NMI, ACC, F-score and ARI, where the bold values indicate the best performance. The framework we proposed in this paper, CommDGI, performs better than all the other methods on three datasets by NMI, F-score and ARI metrics simultaneously, which shows its effectiveness in the community detection task. CommDGI incorporates the community-oriented objectives and disentangled representation into consideration and thus combine the node representation learning tasks with clustering tasks.

The ACC of CommDGI beats other algorithms on Citeseer and Pubmed datasets while is a bit lower than DAEGC on Cora dataset. The explanation for this result is that our model pays more attention on the community structural information learning and the other metrics are the direct metrics of clustering performance evaluation. The labels of three evaluated datasets is actually the label of subfield categories thus our CommDGI approach may not encode the most related classification information in this aspect.

We can observe from the result table that methods using both the adjacency matrix information and node attributes of the graph generally work better than those using only one kind of information. On the Cora dataset, for example, ComE, DAEGC, AGC and our method outperform all the baselines using only one kind of information. This observation demonstrates that both the graph structure and node attributes contain useful information for community detection, and illustrates the significance of capturing the combination of two-kind information.

In addition, we can observe that the graph neural network methods like GAEGC, AGC and our method perform much better than other two-step methods. Specially, vGraph is a generative model and is different to other optimization-based model we discuss in this paper. This observation demonstrates that the joint optimization of graph representation learning and clustering is effective and can better encode the community-related graph information on attributed graphs.

4.5 Ablation Study

We perform an ablation study to understand the importance of each design choice of our framework based on the NMI result of

Table 4: Performance comparison with different community detection methods.

		K-means++	Deepwalk	ComE	CDE	vGraph	DAEGC	AGC	CommDGI
Cora	NMI	0.221	0.249	0.366	0.275	0.345	0.528	0.536	**0.579**
	ACC	0.323	0.538	0.438	0.415	0.287	**0.704**	0.689	0.698
	F-score	0.392	0.418	0.435	0.386	0.305	0.682	0.656	**0.684**
	ARI	0.229	0.290	0.324	0.301	0.312	0.496	0.462	**0.502**
Citeseer	NMI	0.191	0.145	0.250	0.299	0.103	0.397	0.411	**0.419**
	ACC	0.416	0.362	0.441	0.583	0.293	0.672	0.670	**0.684**
	F-score	0.404	0.340	0.448	0.452	0.294	0.636	0.625	**0.647**
	ARI	0.286	0.127	0.281	0.329	0.067	0.410	0.402	**0.414**
Pubmed	NMI	0.230	0.212	0.300	0.201	0.224	0.266	0.316	**0.357**
	ACC	0.415	0.614	0.415	0.579	0.260	0.671	0.697	**0.699**
	F-score	0.481	0.533	0.552	0.571	0.332	0.659	0.687	**0.692**
	ARI	0.249	0.251	0.232	0.257	0.185	0.278	0.281	**0.292**

Table 5: The NMI results of ablation experiment on three datasets.

Framework	Cora	Citeseer	Pubmed
Only Graph MI Optimization	0.465	0.342	0.268
+ Community MI	0.552	0.380	0.319
+ Modularity	0.579	0.399	0.325
+ Disentangled Operation	0.579	0.419	0.357
Two-step DGI	0.530	0.340	0.288

three datasets with diverse characteristics. Specifically, there are five main aspects we hope to learn:

- the importance of using DGI as the backbone unsupervised graph neural network architecture;
- the effectiveness of adding community MI on CommDGI, compared to the single graph MI objective design in DGI;
- the effectiveness of adding classical community detection objective Modularity;
- the improvement of using a disentangled representation aggregation in our framework;
- the advantage of CommDGI compared to performing the community detection with two-step model.

We can observe from Table 5 that our framework can already obtain a relatively high performance with a single graph MI optimization compared to some previous methods which illustrates the significant power of DGI on dealing with unsupervised graph representation learning tasks.

Examining the row 2 and row 3 of this table, we can see both community MI and Modularity objectives clearly contribute the superior performance of CommDGI over previous incomplete models. In addition, row 4 show that disentangled operation plays an important role in CommDGI's performance on Citeseer and Pubmed datasets but doesn't obviously improve the result on Cora dataset. We can learn from this observation that the disentangled representations tend to be helpful in more complex attributed networks which have more nodes or features. In the small-scale network, entangled node representations can achieve the similar result to the disentangles representations.

The last two rows of table 5 show that the combination of graph neural network and community detection can achieve better performance than sequential methods.

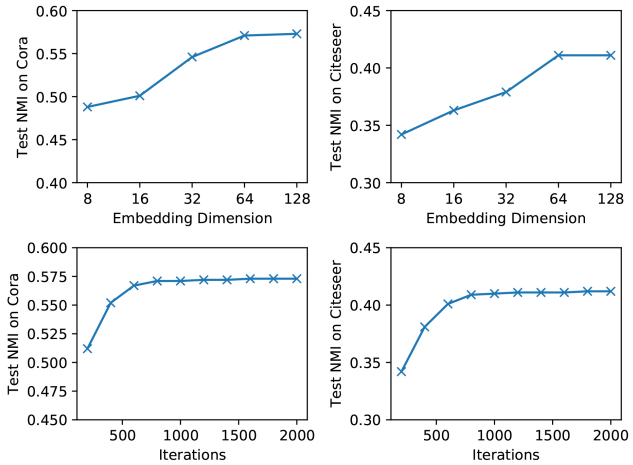

Figure 4: Community Detection NMI results with different embedding dimensions in one channel and different iterations. Our model chooses 64 as a standard setting.

4.6 Parameter Analysis

In this part, we examine the influence of three key parameters in our framework: embedding dimensions, training iterations and the

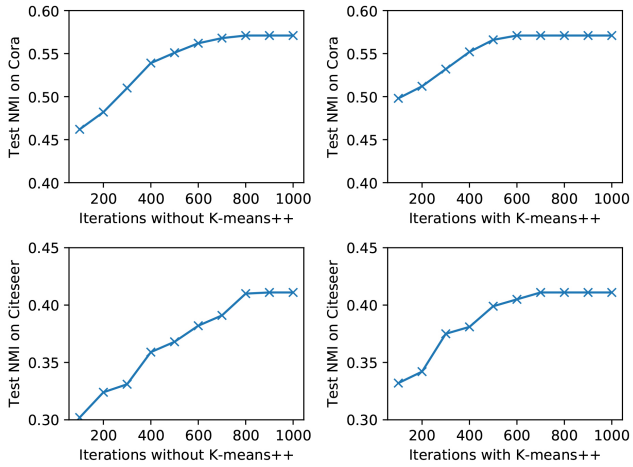

Figure 5: Rate of convergence study with K-means++ clustering center choice.

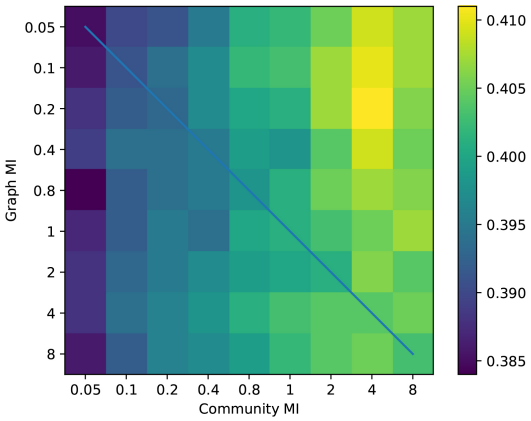

Figure 7: Objective parameter experiment on Citeseer Dataset (The deeper the color, the higher the NMI result).

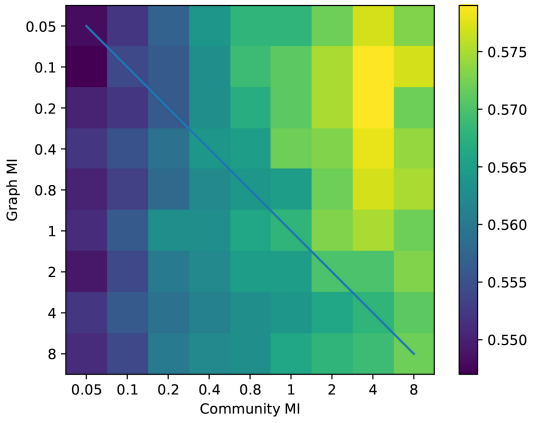

Figure 6: Objective parameter experiment on Cora Dataset (The deeper the color, the higher the NMI result).

coefficient of graph MI, community MI and Modularity. We also study the rate of convergence of the model with K-means++ cluster center choice and the random center choice.

The two lines of sub-figures in Figure 4 show the dimension and iteration sensitivity experiment results on the Cora and Citeseer datasets. The sub-figures in the first line indicate that setting a higher embedding size d in one channel can provide a higher performance on both datasets. A chosen dimension bigger than 64 will still achieve high performance but require much more training time, which may make the model hard to update on Pubmed dataset.

In general, our model can reach the peak of NMI result on Cora and Citeseer datasets in 1000 iterations, more iterations may slightly improve the model performance but not obvious in our iteration experiments shown in Figure 4.

Figure 5 shows the rate of convergence of the model with K-means++ cluster center choice and the rate of convergence of the random choice. With K-means++ cluster center choice, we can obtain a stable result in about 500 iterations and also achieve a higher initial value, which will considerably improve the effectiveness of our community detection framework on large-scale networks.

The parameter α and β donate the importance of graph MI, community MI and modularity. We choose 0.05,0.1,0.2,0.4,0.8,1.0,2.0,4.0 and 8.0 for each parameter and conduct the experiment. Figure 6 and figure 7 display the NMI result tendency of different coefficient choice in our framework. We can observe from the table that our algorithm is robust to different parameter settings. Specially, setting a bigger parameter of community MI and a smaller parameter of graph MI and Modularity will lead to the best performance.

5 RELATED WORK

5.1 Mutual Information Estimation

The infomax principle [3] first proposes the idea of maximizing mutual information(MI) between the inputs and outputs of neural networks but is difficult to calculate the MI between high dimensional continuous variables. Mutual Information Neural Estimation (MINE) [2] realizes the estimation of MI on deep neural networks through training a statistics network as a classifier of samples coming from the joint distribution of two random variables and their product of marginals. An KL-based formulation of MI is used in MINE. Deep InfoMax (DIM) [11] trains an encoder to maximize the mutual information between a global representation and local parts of the input (such as patches of an image). Deep graph infomax (DGI) [28] adapt ideas from DIM to the graph domain and obtains remarkable performance in an unsupervised manner. Contrastive methods and mutual information maximization are used to update DGI model and learn the node representations.

5.2 Deep Community Detection Methods

Deep community detection aims to combine clustering task with deep graph learning. Deep embedding clustering (DEC) [31] designs a KL-divergence loss to make the representation learned by autoencoder surround the cluster centers closer. Improved deep embedding clustering adds a reconstruction loss to the objective of DEC as a constraint to help autoencoder learn a better data representation. Deep attentional embedded graph clustering (DAEGC) [29] applies a graph attentional autoencoder to measure the significance of the neighborhood nodes and employs the KL-divergence loss from DEC to help learn the assignment of graph clustering (community detection). This kind of methods rely on adjacency matrix reconstruction objective to update the graph neural networks, which may ignore the high-order community characteristic and the characteristic of the data itself.

To better capture data characteristic, some GCN-based methods have been proposed recently. ClusterNet [30] propose an alternative decision-focused learning approach that integrates a differentiable method for common graph optimization problems as a layer in learned system. GCN is used to encode node information, and a differentiable version of k-means clustering is used to learn the cluster partition. Structural Deep Clustering Network(SDCN) [5] applies a GCN module, consisting of multiple graph convolutional layers, to learn the GCN-specific representation. SDCN also applies an autoencoder module to learn the autoencoder-specific representation from the raw data and uses a dual self-supervised module to uniformly guide these two modules. These GCN-based models can learn more node characteristic but the representations are still entangled and not community-aware.

6 CONCLUSION

In this paper, we propose a community oriented graph neural network CommDGI. Because community detection task is naturally unsupervised, we use several self-training clustering objective so that the model can generates soft labels from soft K-means assignments to supervise the embedding updating. The graph MI, community MI and Modularity objective are jointly optimized to simultaneously obtain both graph embedding and community detection result. Disentangled representations are used to make the model more generally and capable of handling the complex attributed networks. A comparison of the experimental results with various state-of-the-art algorithms validate CommDGI's community detection performance.

ACKNOWLEDGMENTS

This work is funded in part by the National Natural Science Foundation of China Projects No. U1636207 and No. U1936213. This work is also partially supported by NSF through grant IIS-1763365 and by FSU. This work is funded by Ant Financial through the Ant Financial Science Funds for Security Research.

REFERENCES

[1] David Arthur and Sergei Vassilvitskii. 2007. k-means++: the advantages of careful seeding. (2007), 1027–1035.
[2] Mohamed Ishmael Belghazi, Sai Rajeswar, Aristide Baratin, Devon R Hjelm, and Aaron Courville. 2018. MINE: Mutual Information Neural Estimation. *arXiv: Learning* (2018).
[3] Anthony J Bell and Terrence J Sejnowski. 1995. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation* 7, 6 (1995), 1129–1159.
[4] Vincent D Blondel, Jeanloup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008, 10 (2008), 10008.
[5] Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. 2020. Structural Deep Clustering Network. arXiv:arXiv:2002.01633
[6] Sandro Cavallari, Vincent W Zheng, Hongyun Cai, Kevin Chenchuan Chang, and Erik Cambria. 2017. Learning Community Embedding with Community Detection and Node Embedding on Graphs. (2017), 377–386.
[7] Zhengdao Chen, Xiang Li, and Joan Bruna. 2017. Supervised Community Detection with Line Graph Neural Networks. arXiv:arXiv:1705.08415
[8] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. 2017. Improved Deep Embedded Clustering with Local Structure Preservation. (2017), 1753–1759.
[9] William L Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. (2017), 1024–1034.
[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. (2015), 1026–1034.
[11] R Devon Hjelm, Alex Fedorov, Samuel Lavoiemarchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. (2019).
[12] Xin Huang, Hong Cheng, and Jeffrey Xu Yu. 2015. Dense community detection in multi-valued attributed networks. *Information Sciences* 314 (2015), 77–99.
[13] Hyunjik Kim and Andriy Mnih. 2018. Disentangling by Factorising. *arXiv: Machine Learning* (2018).
[14] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
[15] Thomas N. Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. arXiv:arXiv:1611.07308
[16] Ye Li, Chaofeng Sha, Xin Huang, and Yanchun Zhang. 2018. Community Detection in Attributed Graphs: An Embedding Approach. (2018), 338–345.
[17] Ninghao Liu, Qiaoyu Tan, Yuening Li, Hongxia Yang, Jingren Zhou, and Xia Hu. 2019. Is a Single Vector Enough?: Exploring Node Polysemy for Network Embedding. (2019), 932–940.
[18] Jianxin Ma, Peng Cui, Kun Kuang, Xin Wang, and Wenwu Zhu. 2019. Disentangled Graph Convolutional Networks. (2019), 4212–4221.
[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. (2013), 3111–3119.
[20] Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. (2013), 2265–2273.
[21] M E J Newman. 2006. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America* 103, 23 (2006), 8577–8582.
[22] M E J Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical Review E* 69, 2 (2004), 026113–026113.
[23] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. Graph Representation Learning via Graphical Mutual Information Maximization. *arXiv: Learning* (2020).
[24] Bryan Perozzi, Rami Alrfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. (2014), 701–710.
[25] Yiye Ruan, David Fuhry, and Srinivasan Parthasarathy. 2013. Efficient community detection in large networks using content and links. (2013), 1089–1098.
[26] Fanyun Sun, Meng Qu, Jordan Hoffmann, Chinwei Huang, and Jian Tang. 2019. vGraph: A Generative Model for Joint Community Detection and Node Representation Learning. (2019), 514–524.
[27] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).
[28] Petar Velickovic, William Fedus, William L Hamilton, Pietro Lio, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax. (2019).
[29] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Attributed Graph Clustering: A Deep Attentional Embedding Approach. (2019), 3670–3676.
[30] Bryan Wilder, Eric Ewing, Bistra Dilkina, and Milind Tambe. 2019. End to end learning and optimization on graphs. *arXiv: Learning* (2019).
[31] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. (2016), 478–487.
[32] Jaewon Yang and Jure Leskovec. 2015. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* 42, 1 (2015), 181–213.
[33] Muhan Zhang and Yixin Chen. 2018. Link Prediction Based on Graph Neural Networks. arXiv:arXiv:1802.09691
[34] Xiaotong Zhang, Han Liu, Qimai Li, and Xiaoming Wu. 2019. Attributed Graph Clustering via Adaptive Graph Convolution. (2019), 4327–4333.