One2Multi Graph Autoencoder for Multi-view Graph Clustering

Shaohua Fan
Beijing University of Posts and Telecommunications
Beijing, China
fanshaohua92@163.com

Xiao Wang
Beijing University of Posts and Telecommunications
Beijing, China
xiaowang@bupt.edu.cn

Chuan Shi*
Beijing University of Posts and Telecommunications
Beijing, China
shichuan@bupt.edu.cn

Emiao Lu
Tencent
Shenzhen, China
emiaolu@tencent.com

Ken Lin
Tencent
Shenzhen, China
abrahamlin@tencent.com

Bai Wang
Beijing University of Posts and Telecommunications
Beijing, China
wangbai@bupt.edu.cn

ABSTRACT

Multi-view graph clustering, which seeks a partition of the graph with multiple views that often provide more comprehensive yet complex information, has received considerable attention in recent years. Although some efforts have been made for multi-view graph clustering and achieve decent performances, most of them employ shallow model to deal with the complex relation within multi-view graph, which may seriously restrict the capacity for modeling multi-view graph information. In this paper, we make the first attempt to employ deep learning technique for attributed multi-view graph clustering, and propose a novel task-guided One2Multi graph autoencoder clustering framework. The One2Multi graph autoencoder is able to learn node embeddings by employing one informative graph view and content data to reconstruct multiple graph views. Hence, the shared feature representation of multiple graphs can be well captured. Furthermore, a self-training clustering objective is proposed to iteratively improve the clustering results. By integrating the self-training and autoencoder's reconstruction into a unified framework, our model can jointly optimize the cluster label assignments and embeddings suitable for graph clustering. Experiments on real-world attributed multi-view graph datasets well validate the effectiveness of our model.

KEYWORDS

Attributed Multi-view Graph Clustering, Graph Convolutional Network, Graph Autoencoder

ACM Reference Format:
Shaohua Fan, Xiao Wang, Chuan Shi, Emiao Lu, Ken Lin, and Bai Wang. 2020. One2Multi Graph Autoencoder for Multi-view Graph Clustering. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3366423.3380079

* Corresponding author.

1 INTRODUCTION

Graph clustering, aiming to partition a graph into several densely-connected disjoint communities or groups, is a fundamental task in graph analysis [17]. Graph clustering techniques have been widely used in practice, such as group segmentation [5], structure analysis of communicate network [23], and community detection in social networks [21]. Most existing graph clustering methods focus on dealing with only one graph [12, 20]. However, the real world graph data is far more complex. That is, one usually needs to employ multi-view graph, rather than single-view graph, to better represent the real graph data [16], in which each graph view represents one type of relationship among nodes. Taking the academic network as an example, one graph view can indicate the co-author relationship, while another view can be the co-conference relationship. Moreover, the authors can also be associated with representative keywords as their attributes. Such complex graphs are usually termed as an attributed multi-view graph, which models an interaction system in a complementary and comprehensive way and has a great potential for more accurate graph clustering.

Regarding attributed multi-view graph clustering, previous work can be categorized into two types. One type of work lies in graph analysis based methods [1, 14, 25], which aims to maximize the mutual agreement across different views so as to partition a graph into groups. Another stream of work mainly employs graph embedding techniques to learn compact representation of nodes from multi-view graph data [8, 18, 27], and subsequently, traditional clustering methods such as k-means is used. These methods have achieved good performances on many applications. However, such methods discussed before are both recognized as shallow models, which have limited capacity to reveal the deep relations in complex graph data. Moreover, aforementioned methods pay little attention to the node attribute information.

Recently, Graph Neural Network (GNN) [24], a deep nonlinear representation learning framework, has shown the powerful performance on some graph analysis tasks, such as node classification [6] and clustering [15]. However, most GNNs are developed for single view graph. Also, there are some works which extend GNN to multi-view setting [4, 9], while they are designed in the semi-supervised scenario and used for classification task. Despite the great success of GNN in graph analysis, little effort has been made towards exploring GNN for unsupervised attributed multi-view graph clustering task until now.

It is not a trivial task to apply GNN to attributed multi-view graph clustering, which will face two challenges. 1) How to effectively fuse multi-view graph information in unsupervised setting? It is clear that only one view information is not sufficient for accurate graph clustering, since multi-view graph provide rich side information [25]. A straightforward fusion method is to develop a multi2multi model. That is, multiple encoders and decoders are developed, and each encoder and decoder is for each view graph. However, this straightforward method is not effective, due to the introduction of noise containing in different view graphs. More importantly, the multi2multi model only extracts each view representation separately, while the shared representation may be more important for our task. 2) How to make embeddings learned by GNN more suitable for clustering task? Node embedding and clustering are usually two independent tasks. Node embedding aims to reconstruct the original graph, so the learned node embedding is not necessarily suitable for node clustering. Therefore, we need to optimize node embedding and clustering in a uniform way.

Observing real multi-view graph data, we can find that, although multi-view information reflect node relationship from different aspects, they should share some common node characteristics. Moreover, there usually exists one most informative view dominating clustering performance in many scenarios, which also have been confirmed in literatures [3, 13, 22]. For example, in the academic network, co-author and co-conference relation views both reflect authors' research interests, while co-conference view is more informative, due to revealing common research interests, which yields better cluster performance than other views [3].

According to this observation, we propose a novel One2Multi graph autoencoder framework for attributed multi-view graph clustering to solve the above mentioned challenges. The basic idea of the model is that the shared representation can be extracted from the most informative graph view and content data, and then it is employed to reconstruct all views. Following this idea, we design a novel One2Multi graph autoencoder, which consists of one encoder and multi-decoders. Specifically, it exploits both multi-view graph structure and node content to learn node representation, through one multiple layers Graph Convolutional Network (GCN) [6] encoder learning node representation from the most informative view and multiple graph decoders reconstructing all views. Furthermore, a self-training clustering objective is designed to force the current clustering distribution approaching to a target distribution more suitable for clustering task. By jointly optimizing the reconstruction loss and clustering loss, the model can simultaneously optimize node embeddings and clustering, and mutually improve them in a unified framework.

Our major contributions can be summarized as follows:

- To the best of our knowledge, it is the first time to employ graph deep learning techniques to attributed multi-view graph clustering task which has great potential for many applications.
- We propose a novel One2Multi autoencoder framework for attributed multi-view graph clustering. The One2Multi graph autoencoder provides an effective deep framework to integrate both multi-view graph structure and content information. Moreover, the framework jointly optimizes the multi-view graph embedding learning and graph clustering with mutual promotion.
- The extensive experiments on real-world attributed multi-view graphs show that our algorithm outperforms state-of-the-art graph clustering methods.

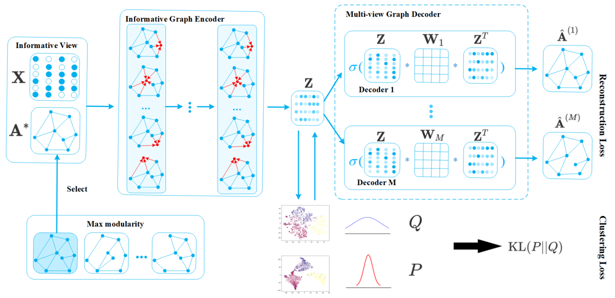


Figure 1: The framework of O2MAC.

2 PROBLEM DEFINITION

In this section, we introduce some notations and definitions that will be used in this paper.

DEFINITION 1. **Attributed Multi-view Graph.** An attributed multi-view graph is represented as $\mathbf{G} = \{\mathbf{V}, \mathbf{E}_1, \cdots, \mathbf{E}_M, \mathbf{X}\}$, where $\mathbf{V} = \{v_i\}_{i=1}^n$ consists of a set of nodes in a graph and $e_{i,j}^{(m)} \in \mathbf{E}_m$ represents a linkage between the node i and j in the m-th graph view. The topological structure of graph $\mathbf{G}$ can be represented by multiple adjacency matrixes $\{\mathbf{A}^{(m)}\}_{m=1}^M$, where $\mathbf{A}_{i,j}^{(m)} = 1$ if $e_{i,j}^{(m)} \in \mathbf{E}_m$, otherwise $\mathbf{A}_{i,j}^{(m)} = 0$. $\mathbf{x}_i \in \mathbf{X}$ indicates the attribute values associated with each node v_i.

Then, we formally define the problem as follows:

DEFINITION 2. **Attributed Multi-view Graph Clustering.** The attributed multi-view graph clustering aims to partition nodes in an attributed multi-view graph into predefined K disjoint clusters $\{C_1, C_2, \cdots, C_K\}$, so that nodes within the same cluster are generally: (1) close to each other in terms of multi-view graph structure while distant otherwise; and (2) close to each other in terms of the node attributes.

3 THE PROPOSED MODEL

In this section, we present the proposed model **One2M**ulti graph **A**utoencoder for multi-view graph **C**lustering (**O2MAC**).

3.1 Overview

The basic idea of the proposed model O2MAC is to develop an One2Multi graph autoencoder to learn node representations from attributed multi-view graph, and then improve the node representations for clustering task by a self-training clustering objective.

Figure 1 shows the overall framework of O2MAC. Our model is mainly composed of two components: One2Multi graph autoencoder and self-training graph clustering. One2Multi graph autoencoder is composed of one informative graph encoder and multi-view graph decoder. With a heuristic metric modularity, we select the most informative view as the input of graph encoder which encodes both graph structure and node content into node representation. Then a multi-view graph decoder is designed to decode the representation for reconstructing all views. Owing to the delicate design of One2Multi graph autoencoder, it not only learns the shared representation, but also absorbs structure characteristics of different views. Furthermore, we use soft labels, generated by the learned embedding itself, to supervise the learning of encoder parameters and cluster centers. The multi-view graph embedding and clustering are optimized in a unified framework, so that we can get an informative encoder which makes the representation more suitable for clustering task.

3.2 One2Multi Graph Convolutional Autoencoder

To represent both multi-view graph structure $\mathbf{A}^{(1)}, \cdots, \mathbf{A}^{(M)}$ and node content $\mathbf{X}$ in a unified framework, we develop a novel One2Multi graph Autoencoder (O2MA) architecture in which one graph convolutional encoder is shared by all views to extract shared representations from one informative graph view and content data, and a multi-decoder is designed to reconstruct multi-view graph data from the shared representation. The most straightforward strategy to integrate multi-view graph information is to develop a multi2multi model, in which multiple encoders are developed to learn a mixed representations from multiple views, then the mixed representation is used to reconstruct multi-view graphs. However, this multi2multi model has several disadvantages. (1) Each view representation is learned separately, which can not extract shared representation well. (2) Mulit-view information introduces much noise, which may be not good for shared representation learning. (3) Learning from all views is also time-consuming.

Informative graph convolutional encoder. Because different graph views represent relationships among the same set of nodes from different aspects and content information is shared by all graph views, there are shared information among views. Moreover, there usually exists one most informative view dominating clustering performance in many scenarios. Therefore, the shared information between the informative view and other views can be extracted from the most informative graph view and content data, and then it can be used to reconstruct all graph views.

Based on this assumption, we take the most informative graph view $\mathbf{A}^* \in \{\mathbf{A}^{(1)}, \cdots, \mathbf{A}^{(M)}\}$ and node content information $\mathbf{X}$ as input to reconstruct all graph views. Please note that we can use prior knowledge to select the informative graph view in many applications. Here, without loss of generality, we provide a heuristic metric, modularity [11], to select the most informative view. Specifically, we first feed each single-view graph adjacency matrix and content information into GCN layers to learn node embeddings respectively, and then perform k-means on the learned embeddings to obtain their clustering indicators. Based on the clustering indicators and adjacency matrix, we compute the modularity score of each graph view, and select the graph view with the highest score as the most informative view. The reason for using modularity is that it provides an objective metric to evaluate clustering structure. And to utilize both graph structure $\mathbf{A}^*$ and node attributes $\mathbf{X}$ in a unified framework, we exploit the GCN layers as a graph encoder. The GCN extends the operation of convolution to graph data in the spectral domain, and learns a layer-wise transformation by a spectral convolution function $f(\mathbf{Z}^{(l)}, \mathbf{A}^*|\mathbf{W}^{(l)})$:

$$\mathbf{Z}^{(l+1)} = f(\mathbf{Z}^{(l)}, \mathbf{A}^*|\mathbf{W}^{(l)}) = \phi(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{Z}^{(l)}\mathbf{W}^{(l)}). \tag{1}$$

$\mathbf{Z}^{(l)}$ is the learned representation by the l-th layer, and $\mathbf{Z}^{(0)} = \mathbf{X} \in \mathbb{R}^{N\times D}$ (N nodes and D features). $\mathbf{W}^{(l)}$ is the filter parameter matrix that we need to learn in the l-th layer. $\tilde{\mathbf{A}} = \mathbf{A}^* + \mathbf{I}$ and $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$. $\mathbf{I}$ is the identity matrix of $\mathbf{A}^*$ and $\phi(\cdot)$ is an activation function.

As shown in the informative graph encoder part of Figure 1, the informative graph encoder $\mathcal{G} = (\mathbf{X}, \mathbf{A}^*)$ is a two-layer GCN and its structure is constructed as follows:

$$\mathbf{Z} = \phi_2(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\phi_1(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}\mathbf{W}^{(0)})\mathbf{W}^{(1)}), \tag{2}$$

where ϕ_1 is the Relu activation function and ϕ_2 is the linear activation function.

Multi-view graph decoder. In order to supervise encoder to extract shared representation by all views, we propose a multi-view graph decoder to reconstruct the multi-view graph data $\hat{\mathbf{A}}^{(1)}, \cdots, \hat{\mathbf{A}}^{(M)}$ from the representation $\mathbf{Z}$.

As shown in the multi-view graph decoder part of Figure 1, our decoder is composed of M view-specific decoders $\{p(\hat{\mathbf{A}}^{(m)}|\mathbf{Z}, \mathbf{W}_m)\}_{m=1}^{M}$ predicting whether there is a link between two nodes in view m, where $\mathbf{W}_m \in \mathbb{R}^{D\times D}$ is the view-specific weights for view m. More specifically, we train a multi-view link prediction layer based on the graph embeddings:

$$\sum_{m=1}^{M} p(\hat{\mathbf{A}}^{(m)}|\mathbf{Z}, \mathbf{W}_m) = \sum_{m=1}^{M} \mathrm{sigmoid}(\mathbf{Z} \cdot \mathbf{W}_m \cdot \mathbf{Z}^T). \tag{3}$$

Reconstruction loss. For multi-view graph autoencoder, we minimize the sum of reconstruction error of each graph view data by:

$$L_r = \sum_{m=1}^{M} L_r^{(m)} = \sum_{m=1}^{M} loss(\mathbf{A}^{(m)}, \hat{\mathbf{A}}^{(m)}), \tag{4}$$

where $L_r^{(m)}$ is the reconstruction loss for view m and L_r is the reconstruction loss for all views. Due to the multi-view architecture of the decoder, the gradients of multi-decoder will propagate through the informative graph encoder during backpropagation process. Therefore, when forward-propagation is processed, the graph encoder will extract the shared representations by all views.

This model can also be viewed as multi-task learning [2]. The multi-view graph decoder provides the multi-task supervised signal for the informative graph encoder to extract shared representation, which makes the shared representation more comprehensive and generalized.

3.3 Self-training Clustering

Aforementioned One2Multi graph convolutional autoencoder can encode attributed multi-view graph into a compact representation. However, the node proximity in embedding space is to preserve local structure of original multi-view graph data, which may be not be guaranteed suitable for clustering. A good data distribution for clustering is that the nodes within the same cluster are gathered densely, and the boundaries between different clusters are distinct. Therefore, it is necessary to introduce other objectives to guide the embedding learning process. Inspired by DEC [26], we employ a self-training clustering objective to utilize "highly confident" nodes as soft labels to supervise the graph clustering process.

Apart from optimizing the reconstruction loss, we input our hidden embeddings into a self-training clustering objective which minimizes the following objective:

$$L_c = \mathrm{KL}(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}, \tag{5}$$

where $KL(\cdot|\cdot)$ is Kullback-Leibler divergence between two distribution, Q is the distribution of the soft labels and q_{ij} is measured by Student's t-distribution [10] to indicate the similarity between node i's embedding z_i and cluster center μ_j:

$$q_{ij} = \frac{(1 + ||z_i - \mu_j||^2)^{-1}}{\sum_{j'}(1 + ||z_i - \mu_{j'}||^2)^{-1}}. \tag{6}$$

It can be seen as a soft clustering assignment distribution of each node. And p_{ij} in Eq.5 is the target distribution defined as:

$$p_{ij} = \frac{q_{ij}^2/f_j}{\sum_{j'} q_{ij'}^2/f_{j'}}, \tag{7}$$

where $f_j = \sum_i q_{ij}$ is the soft cluster frequencies to normalize the loss contribution of each centroid to prevent large clusters from distorting the hidden feature space. As we can see, the target distribution P raises Q to the second power to get a denser distribution. By minimizing KL divergence between Q and P, it will make the distribution of Q more dense. Indeed, in the subsection 4.4, we observe that "highly confident" nodes contribute more to the gradients at the start of KL divergence minimization. The "highly confident" nodes indicate that the nodes have high probability belonging to some cluster which is computed by Eq. 6. This phenomenon can be interpreted as semi-supervised training, which has been demonstrated very effectively in GCN [6].

Overall objective function. We jointly optimize the One2Multi graph autoencoder embedding and clustering learning, and the total objective function is defined as:

$$L = L_r + \gamma L_c, \tag{8}$$

where $\gamma > 0$ is a coefficient that controls the degree of distorting embedded space, and we let $\gamma = 0.1$ for all experiments.

3.4 Optimization

We first pretrain the One2Multi graph autoencoder without the self-training clustering part to obtain a well-trained embedding $\mathbf{Z}$ as described in subsection 3.2. Self-training clustering objective is then performed to improve this embedding. To initialize the cluster centers, we perform standard k-means clustering on the embedded nodes $\mathbf{Z}$ to obtain k initial centroids $\{\mu_j\}_{j=1}^k$. To be specific, there are three kinds of parameters to update: the weights of One2Multi graph autoencoder $W^{(l)}$ and $W_1, \cdots, W_M$, cluster centers μ and target distribution P.

Update O2MAC's weights and cluster centers. Fixing target distribution P and given N samples, the gradients of L_c with respect to cluster center μ_j can be computed as:

$$\frac{\partial L_c}{\partial \mu_j} = 2\sum_{i=1}^{N}(1 + ||z_i - \mu_j||^2)^{-1}(q_{ij} - p_{ij})(z_i - \mu_j). \tag{9}$$

Then given learning rate λ, μ_j is updated by

$$\mu_j = \mu_j - \lambda \frac{\partial L_c}{\partial \mu_j}. \tag{10}$$

The m-th view specific decoder's weights are updated by

$$W_m = W_m - \lambda \frac{\partial L_r^{(m)}}{\partial W_m}. \tag{11}$$

As we can see, the update of W_m only relates to view m reconstruction loss, therefore, the view specific decoder's weights can capture view specific local structure information.

Then the graph encoder's weights are updated by

$$W^{(l)} = W^{(l)} - \lambda\left(\frac{\partial L_r}{\partial W^{(l)}} + \gamma \frac{\partial L_c}{\partial W^{(l)}}\right) = W^{(l)} - \lambda\left(\sum_{r=1}^{M} \frac{\partial L_r^{(m)}}{\partial W^{(l)}} + \gamma \frac{\partial L_c}{\partial W^{(l)}}\right). \tag{12}$$

One can observe that the update of $W^{(l)}$ is related to all views' reconstruction loss, so that the encoder's weights can extract the shared representation by all views.

Update target distribution. The target distribution P which serves as "groundtruth" soft label also depends on predicted soft label. Therefore, to avoid instability in the self-training process, P should be updated using all embedded nodes every T iterations. We update P according to Eq. 6 and Eq. 7. When updating target distribution, the label assigned to v_i is obtained by

$$s_i = \arg\max_j q_{ij}, \tag{13}$$

where q_{ij} is computed by Eq. 6. The training process will be stopped if label assignment change (in percentage) between two consecutive updates for target distribution is less than a threshold δ. And we can obtain clustering results from the last optimized Q.

4 EXPERIMENTS

4.1 Datasets

- **ACM**[1]: This is a paper network from the ACM dataset. We exploit co-paper (two papers are written by same author) relationship and co-subject (two papers contain same subjects) relationship to construct a two view graph. Paper features are the elements of a bag-of-words represented of keywords. And we use papers' research area as ground truth.

[1] http://dl.acm.org

Table 1: The statistics of the datasets.

Dataset	#Node	#Features	#Edges in each view	#Class
ACM	3025	1830	co-paper (29,281) co-subject (2,210,761)	3
DBLP	4057	334	co-author (11,113) co-conference (5,000,495) co-term (6,776,335)	4
IMDB	4780	1232	co-actor (98,010) co-director (21,018)	3

Table 2: Clustering results on three datasets. The '*' indicates the best performance of the baselines. Best results of all methods are indicated in bold. The '-' represents the method run out-of-memory on this dataset.

Method	**ACM**				**DBLP**				**IMDB**			
	Acc	F1	NMI	ARI	Acc	F1	NMI	ARI	Acc	F1	NMI	ARI
LINE	0.6479	0.6594	0.3941	0.3433	0.8689	0.8546	0.6676	0.6988	0.4268	0.2870	0.0031	-0.0090
LINE-avg	0.6479	0.6594	0.3941	0.3432	0.8750	0.8660	0.6681	0.7056	0.4719	0.2985	0.0063	-0.0090
GAE	0.8216*	0.8225*	0.4914*	0.5444*	0.8859	0.8743*	0.6925	0.7410	0.4298	0.4062	0.0402	0.0473
GAE-avg	0.6990	0.7025	0.4771	0.4378	0.5558	0.5418	0.3072	0.2577	0.4442	0.4172*	0.0413*	0.0491*
MNE	0.6370	0.6479	0.2999	0.2486	-	-	-	-	0.3958	0.3316	0.0017	0.0008
PMNE (n)	0.6936	0.6955	0.4648	0.4302	0.7925	0.7966	0.5914	0.5265	**0.4958**	0.3906	0.0359	0.0366
PMNE (r)	0.6492	0.6618	0.4063	0.3453	0.3835	0.3688	0.0872	0.0689	0.4697	0.3183	0.0014	0.0115
PMNE (c)	0.6998	0.7003	0.4775	0.4431	-	-	-	-	0.4719	0.3882	0.0285	0.0284
RMSC	0.6315	0.5746	0.3973	0.3312	0.8994*	0.8248	0.7111*	0.7647*	0.2702	0.3775	0.0054	0.0018
PwMC	0.4162	0.3783	0.0332	0.0395	0.3253	0.2808	0.0190	0.0159	0.2453	0.3164	0.0023	0.0017
SwMC	0.3831	0.4709	0.0838	0.0187	0.6538	0.5602	0.3760	0.3800	0.2671	0.3714	0.0056	0.0004
O2MA	0.8880	0.8894	0.6515	0.6987	0.9040	0.8976	0.7257	0.7705	0.4697	**0.4229**	**0.0524**	**0.0753**
O2MAC	**0.9042**	**0.9053**	**0.6923**	**0.7394**	**0.9074**	**0.9013**	**0.7287**	**0.7780**	0.4502	0.4159	0.0421	0.0564

- **DBLP**[2]: This is an author network from the DBLP dataset. Three views are identified including the co-authorship (two authors have worked together on papers), co-conference (two authors have published papers at the same conference), and co-term (two authors have published papers with the same terms). Author features are the elements of a bag-of-words represented of keywords. To evaluate the method, we use authors' research area as ground truth.
- **IMDB**[3]: This is a movie network from the IMDB dataset. We exploit co-actor (movies are acted by the same actor) relationship and co-director (movies are directed by the same director) relationship to construct a two view graph. Movie features correspond to elements of a bag-of-words represented of plots. To evaluate the method, we use the movies' genre as ground truth.

The detailed descriptions of the datasets are shown in Table 1.

4.2 Baselines

- **LINE** [19]: It is a classical single-view graph embedding method. For single-view methods, we perform the methods on each graph view respectively, and report the best results.
- **GAE** [7]: It is a single-view graph autoencoder method.
- **X-avg**: To exploit multiple views of a network, we apply X method to learn node representations on each single view, then average all learned representations.
- **MNE** [27]: A scalable multi-view network embedding model. For all the multi-view graph embedding/clustering methods, we use the multi-view graph adjacency matrixes as the input.
- **PMNE** [8]: We compare our method with all the three multi-view network embedding models proposed by PMNE, i.e., PMNE (n), PMNE (r), and PMNE (c).
- **RMSC** [25]. A robust multi-view spectral clustering method via low rank and sparse decomposition.
- **PwMC and SwMC** [14]. PwMC is a parameter-weighted multi-view graph clustering method and SwMC is a self-weighted multi-view graph clustering method.
- **O2MA**. A variant of O2MAC, which does not contain clustering loss in the objective function.
- **O2MAC**. Our proposed method for attributed multi-view graph clustering.

Parameter settings and Metrics. For DBLP and IMDB datasets, we train all autoencoder-related models (GAE, O2MAC, O2MA) for 1000 iterations and optimize them with Adam algorithm. As ACM is a smaller dataset, we iterate 250 times for the training. The learning rate λ set as 0.001 for autoencoder-related models. The dimension of all embedding methods is 32. For GAE, it has the same structure with our encoder. For O2MAC, the convergence threshold is set as $\delta = 0.1\%$, and the update intervals $T = 20$. For the rest of the baselines, we retain to the settings described in the corresponding papers. Since all the clustering algorithms depend on the initializations, we repeat all the methods 10 times using random initialization and report the average performance. Moreover, we employ four metrics to validate the clustering results: accuracy (Acc), F-score (F1), NMI and ARI [25].

4.3 Graph Clustering Performance

The results on the three datasets are given in Table 2. As can be seen, the results of O2MAC and O2MA significantly outperform the other

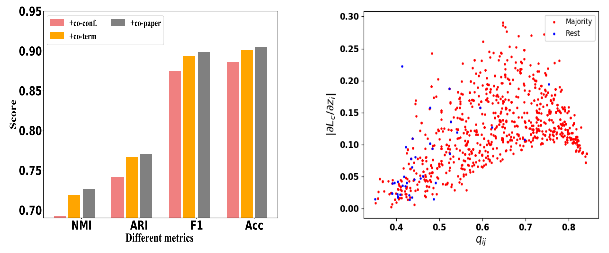
[2] https://dblp.uni-trier.de/
[3] https://www.imdb.com/

Table 3: Clustering results on different input views. The selected views used in our experiments are indicated in bold.

Dataset	Input view	NMI	ARI	Modularity
ACM	co-subject	0.4486	0.4129	0.4896
	co-paper	**0.6515**	**0.6987**	**0.6002**
DBLP	co-term	0.0164	0.0109	-0.0356
	co-author	0.2419	0.1518	0.3599
	co-conf.	**0.7257**	**0.7705**	**0.4392**
IMDB	co-actor	0.0391	0.0411	0.0308
	co-director	**0.0524**	**0.0753**	**0.6516**

baselines on almost all four metrics, indicating the effectiveness of our proposed model. Then, by comparing the results of O2MA and GAE (GAE-avg), we conclude that our proposed One2Multi graph autoencoder is a more effective graph neural network to fuse multi-view graph information. Moreover, compared with O2MA, the better results of O2MAC on ACM and DBLP datasets imply that the self-training clustering objective can further improve the performance after an effective pretraining. While, we also notice that O2MA outperforms O2MAC on IMDB dataset. The reason is that it is difficult to obtain "highly confident" nodes on IMDB. In such situation, these "highly confident" nodes may confine low-confidence nodes to the wrong clusters. To avoid this, we can set $\gamma = 0$ on these datasets, i.e., only use O2MA version.

For baselines, we have the following observations. First, embedding methods significantly outperform other methods, therefore, graph embedding is a promising method to solve graph clustering problem. Second, deep learning method (*i.e.,* GAE) achieves competitive results than other baselines. However, it can only utilize single-view graph and content information. It is possible that a well-designed deep neural network integrating multiple graph views can achieve promising results. Third, the simple multi-view graph average operation, X-avg, does not improve the results, even reduces the performance. Because X-avg is a two-step fusion method, and this mix operation may introduce noise. Therefore, designing an end-to-end fusion model is necessary for clustering task.



(a) Performances of O2MA with additive views.

(b) Gradient visualization at the start of KL divergence minimization.

Figure 2: Model analysis.

4.4 Model Analysis

In this subsection, we analyze various properties of our model. We first validate the necessity of using modularity [11] to select the informative view. We use each graph view as the input of O2MA and reconstruct all graph views on three datasets, and report their results on graph clustering task in Table 3. One can observe that if we feed the graph view with higher modularity value into the encoder, our model will achieve better results. It validates that modularity is a feasible solution to select the informative view.

In our model, we fuse multiple graph views to improve the clustering performance. To further investigate the effect of multiple views on the learned embeddings for the clustering task, we closely examine the performance of O2MA through adding the graph view one by one in DBLP dataset. These three views are co-conference, co-term, and co-paper, and they are added into the model by their order. Figure 2(a) is the four metrics performance of O2MA with additive views. The results with four metrics demonstrate that the performance of our proposed model stably increases as we add views one by one. Therefore, O2MA provides a flexible framework to utilize more graph views.

The underlying assumption of self-training clustering loss is that the highly confident predictions of initial classifier are mostly correct and they will contribute more to the gradients. To verify this assumption for our task, we plot the magnitude of the gradient of L_c with respect to each embedded point, $|\partial L_c/\partial \mathbf{z}_i|$, against its soft assignment, q_{ij}, to a random chosen ACM cluster j (Figure 2(b)). We observe that nodes that are closer to the cluster center (large q_{ij}) contribute more to the gradient (with higher value), and the gradient will propagate through the GCN encoder, which makes the neighborhood nodes have similar embeddings. We also color the majority class of nodes as red, and the rest classes as blue. One can see that most of the nodes in the highly confident area are red and the blue dots are mostly in the low confidence area. Therefore, the highly confident "right" nodes will guide the training process, which in turn helps to improve low confidence ones.

5 CONCLUSION

In this paper, we study the attributed multi-view graph clustering problem, which aims to partition the graph into several non-overlapping clusters with the utilization of its multiple views and attributed information. To solve the challenges in the problem, we propose a novel One2Multi graph autoencoder method for graph clustering, which is called O2MAC. O2MAC utilizes one view graph convolutional encoder and multi-view graph structure decoder to encode the attributed multi-view graphs to a low-dimensional space. Besides that, a self-training clustering loss is introduced to supervise the encoder with highly confident nodes, which makes the feature space more suitable for clustering. The effectiveness of the method is validated by comparing its experimental results with various state-of-the-art algorithms.

ACKNOWLEDGMENTS

This work is supported by the National Key Research and Development Program of China (2018YFB1402600) and the National Natural Science Foundation of China (No. 61772082, 61702296, 61806020). It is also supported by 2018 Tencent Marketing Solution Rhino-Bird Focused Research Program. And Shaohua Fan's research is supported by BUPT Excellent Ph.D. Students Foundation (No. CX2019127).

REFERENCES

[1] Xiao Cai, Feiping Nie, Weidong Cai, and Heng Huang. 2013. Heterogeneous image features integration via multi-modal semi-supervised learning model. In *ICCV*. 1737–1744.

[2] Rich Caruana. 1997. Multitask learning. *Machine learning* 28, 1 (1997), 41–75.

[3] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *SIGKDD*. ACM, 135–144.

[4] Muhammad Raza Khan and Joshua E Blumenstock. 2019. Multi-GCN: Graph Convolutional Networks for Multi-View Networks, with Applications to Global Poverty. *arXiv preprint arXiv:1901.11213* (2019).

[5] Su-Yeon Kim, Tae-Soo Jung, Eui-Ho Suh, and Hyun-Seok Hwang. 2006. Customer segmentation and strategy development based on customer lifetime value: A case study. *Expert systems with applications* 31, 1 (2006), 101–107.

[6] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *ICLR* (2016).

[7] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning* (2016).

[8] Weiyi Liu, Pin-Yu Chen, Sailung Yeung, Toyotaro Suzumura, and Lingli Chen. 2017. Principled multilayer network embedding. In *ICDMW*. IEEE, 134–141.

[9] Yao Ma, Suhang Wang, Chara C Aggarwal, Dawei Yin, and Jiliang Tang. 2019. Multi-dimensional Graph Convolutional Networks. In *SIAM*. SIAM, 657–665.

[10] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[11] Mark EJ Newman. 2006. Modularity and community structure in networks. *Proceedings of the national academy of sciences* 103, 23 (2006), 8577–8582.

[12] Andrew Y Ng, Michael I Jordan, and Yair Weiss. 2002. On spectral clustering: Analysis and an algorithm. In *NIPS*. 849–856.

[13] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. 2011. Multimodal deep learning. In *ICML*. 689–696.

[14] Feiping Nie, Jing Li, Xuelong Li, et al. 2017. Self-weighted Multiview Clustering with Multiple Graphs.. In *IJCAI*. 2564–2570.

[15] S Pan, R Hu, G Long, J Jiang, L Yao, and C Zhang. 2018. Adversarially regularized graph autoencoder for graph embedding. In *IJCAI*.

[16] Meng Qu, Jian Tang, Jingbo Shang, Xiang Ren, Ming Zhang, and Jiawei Han. 2017. An attention-based collaboration framework for multi-view network representation learning. In *CIKM*. 1767–1776.

[17] Satu Elisa Schaeffer. 2007. Graph clustering. *Computer science review* 1, 1 (2007), 27–64.

[18] Yu Shi, Fangqiu Han, Xinwei He, Xinran He, Carl Yang, Jie Luo, and Jiawei Han. 2018. mvn2vec: Preservation and collaboration in multi-view network embedding. *arXiv preprint arXiv:1801.06597* (2018).

[19] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW*. ACM, 1067–1077.

[20] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Attributed Graph Clustering: A Deep Attentional Embedding Approach. *arXiv preprint arXiv:1906.06532* (2019).

[21] Meng Wang, Chaokun Wang, Jeffrey Xu Yu, and Jun Zhang. 2015. Community detection in social networks: an in-depth benchmarking study with a procedure-oriented framework. *Proceedings of the VLDB Endowment* 8, 10 (2015), 998–1009.

[22] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *WWW*. 2022–2032.

[23] Wai-Chiu Wong and Ada Wai-Chee Fu. 2002. Incremental document clustering for web page classification. In *Enabling Society with Information Technology*. Springer, 101–110.

[24] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. 2019. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596* (2019).

[25] Rongkai Xia, Yan Pan, Lei Du, and Jian Yin. 2014. Robust multi-view spectral clustering via low-rank and sparse decomposition. In *AAAI*.

[26] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *ICML*. 478–487.

[27] Hongming Zhang, Liwei Qiu, Lingling Yi, and Yangqiu Song. 2018. Scalable Multiplex Network Embedding.. In *IJCAI*. 3082–3088.