Fine-grained Attributed Graph Clustering

Fine-grained Attributed Graph Clustering

Zhao Kang[1,2] Zhanyu Liu[1] Shirui Pan[3] Ling Tian[1,2,*]

Abstract

Graph clustering is a prevalent issue associated with social networks, data mining, and machine learning; its objective is to detect communities or groups in networks. Inspired by the recent success of deep learning (DL), new DL-based graph clustering methods have achieved promising results. However, a deep neural network involves a large number of training parameters. Moreover, existing methods typically select the similarity metric by an ad hoc approach, which considerably affects the resulting output. In this study, we propose a principled graph learning perspective, fine-grained attributed graph clustering. Based on a shallow approach, the proposed method sufficiently exploits both node features and structure information by benefiting from graph convolution. Consequently, a fine-grained graph encoded higher-order relations is automatically learned. Comprehensive experiments on benchmark datasets demonstrate the superiority of the proposed method over state-of-the-art algorithms, including several DL methods.

1 Introduction

Attributed graphs, which comprise a set of nodes associated with individual feature attributes and edges characterizing pairwise relationships, are natural and efficient representations for non-Euclidean data [22]. They are widely used in daily lives and academic studies, such as citation, social, and brain neural networks. Clustering techniques are commonly used to discover communities or groups in graphs. Nevertheless, some classical clustering methods, such as k-means, may be ineffective because they only handle the data features. Moreover, some graph-based clustering methods have been proposed to leverage structural information [29, 35]. Typically, they first learn a compact graph embedding, then implement classic clustering methods upon it. However, they frequently result in suboptimal performance because they ignore informative node features.

To acquire the best of both worlds, some recent attributed graph clustering methods focus on simultaneously capturing the structural relationship and exploiting the node content information. Refer to [6] for a complete review of various techniques, including nonnegative matrix factorization (NMF) [38], random walks [41], and k-medoids [1]. Among them, deep learning (DL)-based methods, such as graph autoencoder (GAE), variational GAE (VGAE) [15], marginalized GAE (MGAE) [34], graph convolutional network (GCN) [14], adversarially regularized graph autoencoder (ARGA) and variational graph autoencoder (ARVGA) [28], produce state-of-the-art results. DL models have a large number of training parameters and are expensive to train. Recently, Zhang et al. [44] propose adaptive graph convolution (AGC), a graph convolution approach that exploits global cluster structure, performing impressively.

Various graph clustering methods generally build the graph for spectral clustering according to hand-crafted rules or some formula, such as the cosine function, inner product, and Euclidean distance [36]. However, analytically describing a similarity measure is challenging [24]. Heuristic graphs might fail to completely reflect the underlying relations and result in suboptimal solutions in spectral clustering [19, 30]. Learning the graph from data is a principle way.

This study aims to automate the construction of the similarity graph matrix. Unlike other applications, we explore both node feature and structural information for attributed graph clustering. Therefore, we learn the similarity graph based on convolution features. Furthermore, this graph is required to best approximate the initial high-order relationships. To differentiate from other graphs, we refer to the similarity graph as fine-grained graph. Consequently, the fine-grained graph produces high-quality clustering.

The main contributions of this study are as follows:

- Unlike existing graph clustering approaches, a fine-grained graph is automatically learned from the data in our proposed model, which results in better clustering performance.
- Both feature and structural information are explicitly explored in the proposed model; their combination is beneficial because they complement each other in generating excellent clustering results.
- Experiments on the *de facto* benchmark datasets demonstrate that the proposed method's performances are superior to those of the existing methods, includ-

*Corresponding author.
[1]University of Electronic Science and Technology of China. {zkang}@uestc.edu.cn.
[2]Trusted Cloud Computing and Big Data Key Laboratory of Sichuan Province.
[3]Monash University.

ing the recent DL-based approaches. In particular, our method also achieves promising performance on overlapping communities detection task.

2 Related Work

Attributed graph clustering considers both node features and structural information [2]. Conventional NMF and spectral clustering have been applied to both node features and graph to achieve consistent clustering [38, 18]. Some generative models have been used to describe their interactions [9, 3]. Popular GCN has also been used to integrate them [14]. For example, a two-layer GCN is used to perform node embedding and an autoencoder (AE) or variational AE is used to reconstruct the adjacency matrix in GAE and VGAE [15]. A three-layer GCN is used to learn node representation, and a marginalized denoising AE is used to reconstruct the node features in MGAE [34]. ARGA and ARVGA [28] apply GAE and VGAE, respectively, to learn node representations and then use generative adversarial networks to match the node representations to a prior distribution. These methods can only capture the neighbors of each node two or three hops away and may fail to capture the global cluster structure of large graphs.

Recently, an attention network is introduced to characterize the importance of neighbors to a node, and an inner product decoder reconstructs the graph structure in deep attentional embedding graph clustering (DAEGC) [33]. GMM-VGAE [10] combines variational graph auto-encoder with Gaussian mixture models to effectively discover the inherent complex data distributions. AGC [44] captures the global cluster structure by exploiting high-order graph convolution. In particular, instead of stacking several layers in GCN, AGC first uses a k-order graph convolution that acts as a low-pass graph filter on node features to achieve smooth embeddings and then implements spectral clustering on graph built by the inner product of the learned features. DAEGC and AGC use the common approach of constructing the similarity graph, which might be inadequate for reflecting the underlying "true" relations among nodes. As demonstrated in spectral clustering, an adequate similarity graph can significantly enhance the clustering performance. To address the above issue, automatically learning similarity from data is the essential approach.

For clustering, a more challenging situation is that there are overlappings between clusters, which means that some nodes belong to several categories [40]. For instance, a particular user could connect to her family and her coworkers in a social network. In the literature, the organization of nodes in different groups is often considered by community detection methods. To solve this problem, some attempts have been made, e.g., BigCLAM [42], CESNA [43], Circles [25], SVI [8], vGraph and its variant vGraph+ [31]. In particular, vGraph learns the cluster membership and node representation collaboratively based on an effective variational inference algorithm. It achieves the state-of-the-art performance on both community detection and node representation learning tasks. Nevertheless, the reported accuracy still has much room for improvement. Thus, characterizing communities with overlapping nodes is still an open challenge [7].

3 Proposed Method

Given a nondirected graph $G = (\mathcal{V}, E, X)$, where $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ represents n nodes, $X = \{x_1, \cdots, x_n\}^\top \in \mathcal{R}^{n\times d}$ is the corresponding feature matrix of the nodes, and E is a set of edges denoted by an adjacency matrix $\widetilde{A} = \{\tilde{a}_{ij}\} \in \mathcal{R}^{n\times n}$. $\tilde{a}_{ij} = 1$ if $(v_i, v_j) \in E$ and $\tilde{a}_{ij} = 0$ otherwise. $A = D^{-\frac{1}{2}}(\widetilde{A} + I)D^{-\frac{1}{2}}$ is a symmetrically normalized adjacency matrix, and D is the degree matrix, in which a renormalization technique is applied by adding a self-loop to each node [34]. I represents the identity matrix with a proper size. From another perspective, A denotes the transition probability matrix of a single-step random walk. The objective of graph clustering is to divide the n nodes of a graph G into m disjoint groups.

Each column of X can be treated as a graph signal. A graph signal is smooth if nearby nodes have similar feature representations. A smooth graph signal contains more low-frequency basis signals than high-frequency ones [26]; therefore, a low-pass filter can be applied to achieve a smooth graph signal, which is called graph convolution. A filtered graph signal will facilitate the downstream clustering task because nearby nodes are more likely to lie in the same group. Specifically, the smoothed signals $\bar{X}$ can be achieved by solving the following optimization problem [26]:

$$\min_{\bar{X}} \|\bar{X} - X\|_F^2 + \frac{1}{2}\operatorname{Tr}\left(\bar{\mathrm{X}}^\top \mathrm{L}\bar{\mathrm{X}}\right), \tag{3.1}$$

where $L = I - A$ is the normalized graph Laplacian. $\bar{X}$ can be obtained by taking the derivative of Eq. (1) w.r.t. $\bar{X}$ and setting it to zero, which yields

$$\bar{X} = (I + \frac{1}{2}L)^{-1}X. \tag{3.2}$$

To get rid of matrix inversion, we approximate $\bar{X}$ by its first-order Taylor series expansion, i.e., $\bar{X} = (I - L/2)X$. Generally, k-th order graph filtering can be written as

$$\bar{X} = (I - \frac{1}{2}L)^k X, \tag{3.3}$$

where k is a positive integer. Graph filtering can filter out undesirable high-frequency noise while preserving the graph geometric features. Afterwards, we can input the inner product of $\bar{X}$ to the spectral clustering algorithm for obtaining the final partitions.

However, this simple approach might underuse the rich structural information because it only uses the fixed raw

graph A, which may not reflect the "true" graph topology, and A might be sparse and noisy owing to the inevitable measurement error. In addition, AGC only learns a feature representation, based on which the simple inner product graph can lead to overcomplicated (with large number of edges) or misleading graphs [23, 13] and may be unable to truthfully reveal the global structure of a given data.

To address these issues, we propose learning a fine-grained graph from data. Recently, the self-expressiveness property of data has shown great success in similarity learning [27]. Mathematically, it can be formulated as follows:

$$\min_S \|X^\top - X^\top S\|_F^2 + \alpha\Theta(S), \tag{3.4}$$

where $\alpha > 0$ is a trade-off parameter and $\top$ represents the transpose operator. The first term is the self-reconstruction error, and the coefficient matrix $S \in \mathcal{R}^{n\times n}$ measures similarities between samples. Θ is a regularization function, which includes the well-known nuclear norm, L_1 norm [12], and Frobenius norm [21]. Instead of using raw node features, we use the filtered $\bar{X}$ to benefit from the smoothed representation.

Even by using Eq. (3.4) to obtain graph S, the raw affinity matrix A is only implicitly used when computing $\bar{X}$. It is desired to exploit this structural information explicitly. In addition, in various real applications, data can display structures beyond simply being low-rank or sparse. Considering these, we ensure that our learned S considers the original relationships and propose a fine-grained attributed graph clustering (FGC) model as follows:

$$\min_S \|\bar{X}^\top - \bar{X}^\top S\|_F^2 + \alpha\|S - f(A)\|_F^2, \tag{3.5}$$

where $f(A)$ is a function of A, which characterizes complex structure relationships. The second term measures the deviation of learned graph S from the original structure. By penalizing the second term, we ensure that S retains some topology information in the original graph. Therefore, the derived graph S could reconstruct the k-order graph convolution $\bar{X}$ and approximate high-order proximity $f(A)$.

In practice, networks are always sparse, i.e., $\mathcal{O}(E) = \mathcal{O}(\mathcal{V})$. Therefore, the original A, which represents the first-order proximity, is usually very sparse and insufficient to completely model the pairwise relations between nodes. In this study, we explore high-order neighbors in a network, which is important in practice [32, 20]. For example, the second-order proximity between node v_i and v_j can be characterized by the probability that a two-step random walk from node v_i to v_j. Intuitively, the probability will be significant if node v_i and v_j share several common neighbors. From this perspective, p-order proximity is the probability that a random walk starts from v_i and reaches v_j with p steps [5]. It can be computed as follows:

$$A^p = \underbrace{A \cdot A \cdots A}_{p}. \tag{3.6}$$

To incorporate high-order proximity in $f(A)$, we define it as follows:

$$f(A) = A + A^2 + \cdots + A^P. \tag{3.7}$$

Therefore, $f(A)$ encodes the information regarding p-order proximity, where $p = 1, 2, \cdots, P$. Consequently, the fine-grained graph, S, will benefit from the complementary information provided by different orders of A.

Compared with existing works in the literature, FGC has several distinct properties. First, a novel regularizer, i.e., the second term in Eq.(3.5) is designed to flexiblely explore the different orders of topology information in the original graph. Second, a graph learning strategy is introduced to tackle the graph clustering problem because the original graph could be noisy or incomplete and is not directly applicable. Third, the proposed objective function (3.5) simultaneously exploits node features and graph structure information, enhancing the performance. Fourth, the proposed model can be easily extended to solve subspace learning, visual analysis [17] tasks, etc. In addition, we want highlight that the adopted graph filtering is basically a simplified version of GCN [39]. Compared to GCN-based approaches, the proposed method is simple and naturally interpretable. In particular, we demonstrate a strategy for traditional shallow models to benefit from deep representation learning.

3.1 Optimization Eq. (3.5) can be easily solved by setting its first-order derivative w.r.t. S to zero, which yields

$$S = (\bar{X}\bar{X}^\top + \alpha I)^{-1}(\alpha f(A) + \bar{X}\bar{X}^\top). \tag{3.8}$$

When the number of nodes is very large, the calculation of the inverse of the entire large matrix $(\bar{X}\bar{X}^\top + \alpha I) \in \mathcal{R}^{n\times n}$ has prohibitively high computational complexity with $\mathcal{O}(n^3)$, making it infeasible for large-scale data. This problem can be alleviated using the Woodbury matrix identity:

$$(A+UBV^\top)^{-1} = A^{-1} - A^{-1}U(B^{-1}+V^\top A^{-1}U)^{-1}V^\top A^{-1}.$$

Then, Eq. (3.8) can be reformulated as follows:

$$\begin{aligned} S &= \big[\frac{1}{\alpha}I - \frac{1}{\alpha^2}\bar{X}(I + \frac{1}{\alpha}\bar{X}^\top\bar{X})^{-1}\bar{X}^\top\big](\alpha f(A) + \bar{X}\bar{X}^\top). \\ &= f(A) + \frac{\bar{X}\bar{X}^\top}{\alpha} - \frac{\bar{X}(I+\frac{1}{\alpha}\bar{X}^\top\bar{X})^{-1}\bar{X}^\top(\alpha f(A)+\bar{X}\bar{X}^\top)}{\alpha^2}. \end{aligned} \tag{3.9}$$

The complexity is reduced from $\mathcal{O}(n^3)$ to $\mathcal{O}(dn^2)$ when $n > d$. Moreover, the cost for calculating $\bar{X}$ is $\mathcal{O}(n^3)$.

Algorithm 1 FGC

Require: Node set $\mathcal{V}$, adjacency matrix A, feature matrix X, order of filter k, trade-off parameter α, polynomial order of proximity matrix P, cluster number m.
Ensure: m clusters.
1: $L = I - A$
2: $\bar{X} = (I - \frac{1}{2}L)^k X$
3: $f(A) = \sum_{i=1}^{P} A^i$
4: Compute S using (3.8) or (3.9)
5: $C = \frac{1}{2}(|S| + |S^\top|)$
6: Do spectral clustering to C

In fact, the graph is often sparse (assuming N denotes the number of nonzero entries in graph Laplacian), we can left multiply X by $(I - \frac{L}{2})$ k times, resulting in $\mathcal{O}(Ndk)$. Further, the complexity of Eq. (3.5) can be reduced to $\mathcal{O}(n)$ time using the idea of the anchor point [11]. Thus, in future, the proposed method can be easily modified to operate with large-scale data.

In Eq. (3.9), the first term is $f(A)$ and the other two terms are functions of $f(A)$ and $\bar{X}\bar{X}^\top$. Therefore, the last two terms perform a correction to the high-order proximity information $f(A)$. The fine-grained graph S basically modifies the original high-order proximity matrix to achieve a better one, which will be responsible for the excellent performance of the proposed algorithm.

Then, we perform classical spectral clustering to divide nodes into m clusters. In particular, $C = \frac{1}{2}(|S| + |S^\top|)$ is first calculated to make the similarity matrix symmetric and nonnegative. Then, spectral embeddings are obtained by computing the eigenvectors corresponding to the m largest eigenvalues of C. Finally, the k-means method is applied to the eigenvectors to achieve cluster assignments. The complete procedures for the proposed model are presented in Algorithm 1. Notably, FGC is an iteration-free method, which is desired in practice [1].

4 Experiments

4.1 Datasets We conduct experiments on several benchmark datasets commonly used for assessing of attributed graph analysis. Cora, Citeseer, and Pubmed [15] are citation networks; the nodes in these networks represent publications and are connected if one cites the other. Wiki [41] is a webpage network; the nodes in this network represent web pages and are connected if one links the other. The features in Cora and Citeseer are binary word vectors, whereas tf-idf weighted word vectors in Pubmed and Wiki. Compared to other recent attributed graph clustering work, we additionally employ Large Cora [16] dataset, having the most edges. For these datasets, clusters represent a collection of publications or web pages. The details of these attributed networks are summarized in Table 1.

[1] The source code is available at https://github.com/sckangz/FGC

Table 1: Dataset statistics.

Dataset	Nodes	Edges	Features	Clusters
Cora	2708	5429	1433	7
Citeseer	3327	4732	3703	6
Pubmed	19717	44338	500	3
Wiki	2405	17981	4973	17
Large Cora	11881	64898	3780	10

4.2 Baselines and Evaluation Metrics We compare FGC with three types of clustering methods. The major difference among these methods is the data information that they adopt for clustering.

- Clustering methods that use only graph structures, including Spectral-g (spectral clustering that only uses the graph structure information), NMF-based community-preserved embedding (M-NMF) [37], deep neural networks for graph representations (DNGR) [4], and DeepWalk [29].
- Clustering methods that use only the node features, including k-means and Spectral-f (spectral clustering that uses only the information of node features to construct a graph matrix via linear kernel).
- Attributed graph clustering methods that use both node features and graph structure information, including ARGA and ARVGA [28], and GAE and VGAE [15], MGAE [34], AGC [44], DAEGC [33], semantic community identification in large attribute networks (SCI) [38], VGAE with Gaussian mixture models (GMM-VGAE) [10].

Three commonly used performance metrics, clustering accuracy (Acc), normalized mutual information (NMI) and macro F1-score (F1), are used to assess the performance of different models; a higher value indicates better performance. For an unbiased comparison, we follow the settings in AGC and directly copy part of the results from AGC and DAEGC. These methods are carefully tuned on each dataset to achieve the best performance. For the newly added Large Cora, we only compared with several representative methods whose code is available. We run each method 10 times on each dataset and report the average results. For FGC, we consider 2-order neighbors by setting $f(A) = A + A^2$ and search for the best parameters, α, and the order of the filter, k.

4.3 Result Analysis The experiment results are summarized in Table 2, wherein the best results are highlighted in bold. The proposed method clearly outperforms other state-of-the-art methods in most of the evaluation measures. In particular,

Table 2: Clustering performance of various methods on various datasets.

Methods	Input	Cora			Citeseer			Pubmed			Wiki			Large Cora		
		Acc%	NMI%	F1%	Acc%	NMI%	F1%	Acc%	NMI%	F1%	Acc%	NMI%	F1%	Acc%	NMI%	F1%
Spectral-g	Graph	34.19	19.49	30.17	25.91	11.84	29.48	39.74	3.46	51.97	23.58	19.28	17.21	39.84	8.24	10.70
DNGR	Graph	49.24	37.29	37.29	32.59	18.02	44.19	45.35	15.38	17.90	37.58	35.85	25.38	-	-	-
DeepWalk	Graph	46.74	31.75	38.06	36.15	9.66	26.70	61.86	16.71	47.06	38.46	32.38	25.74	-	-	-
M-NMF	Graph	42.30	25.60	32.00	33.60	9.90	25.50	47.00	8.40	44.30	-	-	-	-	-	-
k-means	Feature	34.65	16.73	25.42	38.49	17.02	30.47	57.32	29.12	57.35	33.37	30.20	24.51	33.09	9.36	11.31
Spectral-f	Feature	36.26	15.09	25.64	46.23	21.19	33.70	59.91	**32.55**	58.61	41.28	43.99	25.20	29.71	11.65	17.76
ARGA	Both	64.00	44.90	61.90	57.30	35.00	54.60	59.12	23.17	58.41	41.40	39.50	38.27	-	-	-
ARVGE	Both	63.80	45.00	62.70	54.40	26.10	52.90	58.22	20.62	23.04	41.55	40.01	37.80	-	-	-
GAE	Both	53.25	40.69	41.97	41.26	18.34	29.13	64.08	22.97	49.26	17.33	11.93	15.35	-	-	-
VGAE	Both	55.95	38.45	41.50	44.38	22.71	31.88	65.48	25.09	50.95	28.67	30.28	20.49	-	-	-
MGAE	Both	63.43	45.57	38.01	63.56	39.75	39.49	43.88	8.16	41.98	50.14	**47.97**	39.20	38.04	32.43	29.02
AGC	Both	68.92	53.68	65.61	67.00	41.13	62.48	69.78	31.59	68.72	47.65	45.28	**40.36**	40.54	32.46	31.84
DAEGC	Both	70.40	52.80	**68.20**	67.20	39.70	63.60	67.10	26.60	65.90	38.25	37.63	23.64	39.87	32.81	19.05
SCI	Both	41.21	21.57	11.82	33.45	9.77	18.01	44.89	5.99	35.73	32.72	26.38	19.03	26.78	11.31	7.68
GMM-VGAE	Both	71.50	54.43	67.76	67.44	42.30	63.22	**71.03**	30.28	**69.74**	-	-	-	-	-	-
FGC	Both	**72.90**	**56.12**	63.27	**69.01**	**44.02**	**64.43**	70.01	31.56	69.10	**51.10**	44.12	34.79	**48.25**	**35.24**	**35.52**

- FGC evidently outperforms the clustering methods that exploit only the proximity or feature matrix information, a straightforward consequence because FGC completely utilizes the available data by exploiting both proximity matrix and node features, which complement each other and consequently enhance the clustering algorithm. Although DeepWalk and DNGR apply deep AE for representation learning, their results are unremarkable because they neglect the feature information.

- FGC consistently outperforms the GCN-based clustering methods: GAE, VGAE, MGAE, ARGA, and ARVGA. Although the GCN-based clustering methods exploit both the proximity and feature matrix information, they are targeted at better representations, and the downstream graph construction step might have resulted in information loss. By contrast, FGC better utilizes the available information, which automatically outputs a fine-grained graph. MGAE perofrms good on Wiki, probably because Wiki is more densely connected than other datasets, and exploring 3-hop neighbors might be sufficient for feature smoothing. GAE, VGAE, ARGA, and ARVGA exploit only 2-hop neighbors of each node. However, it is inadequate for larger and sparser networks, such as Citeseer and Pubmed, wherein the performance gaps between FGC and MGAE are wide. Although FGC only uses second-order information in the experiment, its performance is promising, which can be attributed to its graph learning approach.

- FGC consistently outperforms the closely related method AGC by a considerable margin on Cora, Citeseer, Wiki, and Large Cora and is comparable on Pubmed. Although AGC aggregates information within k-hop neighbors, it only produces better feature representations for clustering. Unlike AGC, FGC directly outputs a fine-grained graph, which is subsequently used for spectral clustering. In addition, AGC adopts an automatic approach to find a suitable k, which may lead to a local solution. For fairness of comparison, we also tune k in AGC to report its best performance. In terms of ACC, NMI, F1, the results are 68.90, 53.65, 65.60; 68.39, 42.50, 63.76; 69.87, 31.60, 68.79 on Cora, Citeseer, Pubmed, respectively. We can see that our method can still outperform AGC.

Table 3: Computation time of several representative methods (Seconds).

Method	Cora	Citeseer	Pubmed	Wiki	Large Cora
AGC	3.42	40.36	20.77	8.21	29.18
DAEGC	561.69	946.89	50854.15	562.85	9339.67
FGC	4.60	9.49	268.44	8.11	58.76

- FGC clearly outperforms DAEGC and is comparable with GMM-VGAE method performing graph embedding and clustering in a unified framework. In addition, soft labels are used from the clustering to supervise graph embedding. Therefore, they are very complex, and the network might be difficult to train, whereas FGC is simple yet competitive.

Furthermore, we report the computation time of several representative methods given in Table 3. All methods are implemented on Python with an Intel Core i5-8400 CPU and 16GB memory. In particular, we retain the original setting (Tensorlow CPU) for DAEGC. As expected, AGC and FGC are efficient, with several orders of magnitude faster than DAEGC.

Table 4: Result of ablation study.

Method	Cora			Citeseer			Pubmed			Wiki			Large Cora		
	Acc%	NMI%	F1%	Acc%	NMI%	F1%	Acc%	NMI%	F1%	Acc%	NMI%	F1%	Acc%	NMI%	F1%
Baseline	67.61	53.12	56.53	67.02	41.57	62.61	69.90	**32.51**	68.99	53.89	51.27	**46.06**	48.30	26.23	13.96
$f(A) = 0$	69.46	56.10	63.25	66.93	41.73	61.09	64.45	24.63	64.40	**56.30**	**52.44**	44.85	43.46	27.48	31.73
$f(A) = A$	68.57	53.94	59.83	67.72	41.06	59.11	70.14	32.47	69.29	51.60	47.19	43.55	**49.26**	30.15	17.81
$f(A) = A + A^2$	**72.90**	**56.12**	63.27	**69.01**	**44.02**	**64.43**	70.01	31.56	69.10	51.10	44.12	34.79	48.25	**35.24**	**35.52**
$f(A) = A + A^2 + A^3$	71.57	56.10	59.83	67.15	40.29	58.10	**70.32**	30.34	**69.49**	47.53	39.68	34.37	48.31	30.51	24.01
$f(A) = A + A^2 + A^3 + A^4$	71.49	52.55	**63.96**	67.33	40.03	58.23	68.49	27.32	68.21	43.49	35.83	27.17	46.87	26.26	30.91

4.4 Ablation Study To verify the effectiveness of two main components (convolution and high-order proximity) in FGC, we conduct ablation experiments on the five datasets. The ablation study results are summarized in Table 4.

To examine the effect of convolution, we use the following model as a baseline:

$$\min_{S} \|X^\top - X^\top S\|_F^2 + \alpha \|S - f(A)\|_F^2, \tag{4.10}$$

where $f(A) = A + A^2$. In other words, the baseline results in Table 4 are obtained on the basis of raw features. Compared with Table 2, following are the observations:

- The baseline results are satisfactory and outperform other methods, except the most recent AGC and DAEGC, which verifies the advantage of the proposed graph learning approach.
- Compared with AGC, the baseline produces comparable results on Cora, Citeseer, and Pubmed, and performs much better on Wiki and Large Cora. Instead of using a convolution graph such as AGC, the baseline explicitly uses the second-order proximity.
- Compared with DAEGC, the baseline performs better on Pubmed and is comparable on Citeseer. Notably, both DAEGC and baseline exploit features and second-order neighbor information, but DAEGC adopts an inner product decoder to model the similarity graph. The simplicity and intelligible properties of FGC make it appealing for real-world applications.

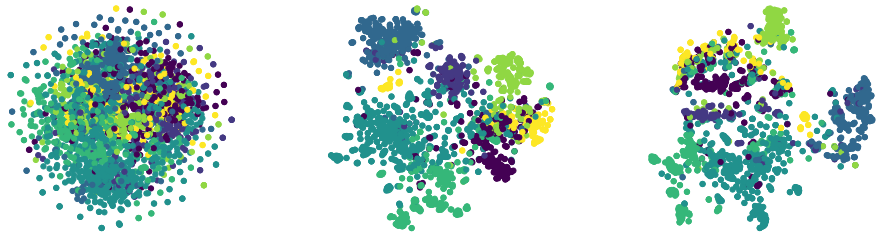
In summary, the performance of the baseline demonstrates the effectiveness of automatic graph learning from data. In addition, the performance of FGC is generally better than the baseline, proving the effectiveness of convolution and making the graph signal smooth. However, the baseline outperforms FGC on Wiki, because the affinity matrix A is very dense and convolution makes the features oversmooth, i.e., the features of nodes in different clusters are mixed and become indistinguishable.

To investigate the impact of high-order topological structure, we assess the performance of FGC with different orders of $f(A)$ (Table 4). In particular, we also set $f(A) = 0$ to assess the benefit of having the node-node proximity preserving term in Eq. (3.5). We draw the following conclusions:

- FGC with original A outperforms the baseline, demonstrating the advantage of smooth features because of the adoption of graph convolution.
- In general, up to second-order proximity generates the best performance. Without exploring the original proximity, we could not achieve a good performance in most cases. The second-order method outperforms the first-order method, which clearly verifies the importance of incorporating high-order information. However, the third and fourth-order methods might deteriorate the performance, probably because of the approach using which we compute high-order information. Computing it directly using adjacency matrices could alter the relationship between nodes. Moreover, the proposed approach also introduces redundant information [45].

4.5 Parameter Analysis There is a trade-off parameter α in model (3.5), which controls the amount of proximity information to be preserved. In addition, there is an implicit parameter k, i.e., the order of the filter, which controls the smoothness of representation. When k increases, nearby node features become similar. However, a very large k will result in oversmoothing, i.e., the features of nodes in different clusters will be mixed and become indistinguishable. To visualize this effect, we apply t-SNE to the raw and filtered node features of Cora (Figure 1). Nearby nodes have similar feature representations as k increases. The data exhibits clear cluster structures at $k = 15$. Nevertheless, the cluster structures disappear when the features are oversmoothed at $k = 60$.

Considering Cora and Large Cora as examples, Figure 2 shows the effects of α and k on clustering performance. A rational result could be achieved with a small range of k and a large range of α. Thus, we can fix k and search for α in practice.



(a) Raw features (b) $k = 15$ (c) $k = 60$

Figure 1: t-SNE demonstration of the raw and filtered node features of Cora dataset.

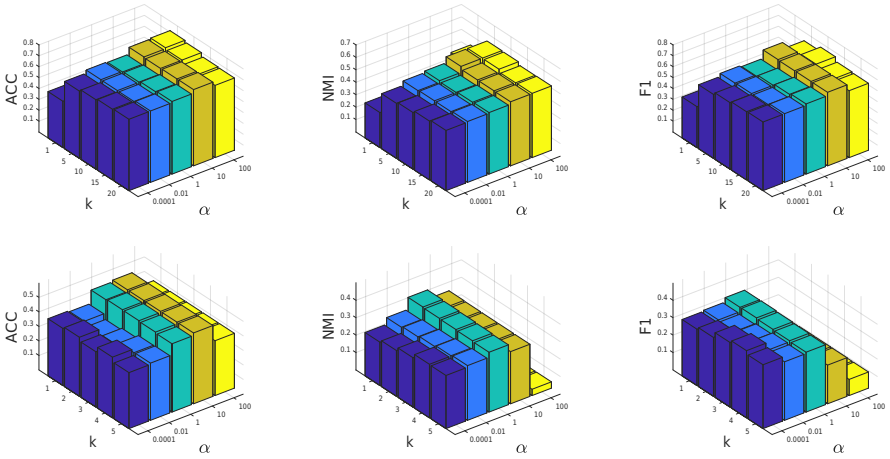


Figure 2: The influence of parameters k and α on results of Cora (1st row) and Large Cora (2nd row) datasets.

4.6 Overlapping Community Detection We examine the performance of the proposed method in a more challenging task: overlapping community detection. Therefore, we replace the k-means step with fuzzy c-means in the proposed method such that each node could be assigned to more than one group. Facebook is chosen as a benchmark dataset, comprising 10 different ego-networks with identified circles. Social circles formed by friends are assumed to be ground-truth communities. Five representative baselines are chosen: BigCLAM [42], CESNA [43], Circles [25], SVI [8], vGraph and its variant vGraph+ [31]. Following vGraph, we use F1 and Jaccard similarity to evaluate the performance of overlapping community detection.

The results are summarized in Table 5. The proposed method outperforms all baseline methods in 7 out of 10 datasets with both metrics because it is capable of leveraging both feature and structure information, which verifies its superiority in overlapping case.

5 Conclusion

In this study, we investigate the attributed graph clustering from a principled graph learning perspective. The main idea is to automatically learn a high-quality graph based on self-expression in a smooth representation. Then, the learned graph is required to preserve some high-order proximity to some extent. Although the proposed method is simple to understand and implement, it yields promising results compared with several state-of-the-art deep learning methods on benchmark datasets. Furthermore, it achieves impressive performance on a more challenging task: overlapping community detection. Therefore, our method is a promising clustering technique in real applications.

6 Acknowledgements

This research work was supported by National Natural Science Foundation of China (NSFC) (61806045, U19A2059), and by Sichuan Science and Technology Program(2020YFG03282021YFG0018).

Table 5: Evaluation on overlapping communities. '-' means the task is not completed in 24 hours.

	F1							Jaccard						
Dataset	BigCLAM	CESNA	Circles	SVI	vGraph	vGraph+	FGC	BigCLAM	CESNA	Circles	SVI	vGraph	vGraph+	FGC
facebook0	0.2948	0.2806	0.2860	0.2810	0.2440	0.2606	**0.3231**	0.1846	0.1725	0.1862	0.1760	0.1458	0.1594	**0.2119**
facebook107	**0.3928**	0.3733	0.2467	0.2689	0.2817	0.3178	0.2928	**0.2752**	0.2695	0.1547	0.1719	0.1827	0.2170	0.1850
facebook1684	0.5041	0.5121	0.2894	0.3591	0.4232	0.4379	**0.5159**	0.3801	0.3871	0.1871	0.2467	0.2917	0.3272	**0.4025**
facebook1912	0.3493	0.3474	0.2617	0.2804	0.2579	0.3750	**0.4436**	0.2412	0.2394	0.1672	0.2010	0.1855	0.2796	**0.3507**
facebook3437	0.1986	0.2009	0.1009	0.1544	0.2087	**0.2267**	0.1725	0.1148	0.1165	0.0545	0.0902	0.1201	**0.1328**	0.0979
facebook348	0.4964	0.5375	0.5175	0.4607	0.5539	0.5314	**0.6260**	0.3586	0.4001	0.3927	0.3360	0.4099	0.4050	**0.5194**
facebook3980	0.3274	0.3574	0.3203	-	0.4450	0.4150	**0.5302**	0.2426	0.2645	0.2097	-	0.3376	0.2933	**0.4299**
facebook414	0.5886	0.6007	0.4843	0.3893	0.6471	**0.6693**	0.5422	0.4713	0.4732	0.3418	0.2931	0.5184	**0.5587**	0.4081
facebook686	0.3825	0.3900	0.5036	0.4639	0.4775	0.5379	**0.5903**	0.2504	0.2534	0.3615	0.3394	0.3272	0.3856	**0.4499**
facebook698	0.5423	0.5865	0.3515	0.4031	0.5396	0.5950	**0.6000**	0.4192	0.4588	0.2255	0.3002	0.4356	0.4771	**0.4970**

References

[1] Esra Akbas and Peixiang Zhao. "Attributed graph clustering: An attribute-aware graph embedding approach". In: *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 305–308.

[2] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. "Spectral clustering with graph neural networks for graph pooling". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 874–883.

[3] Aleksandar Bojchevski and Stephan Günnemann. "Bayesian robust attributed graph clustering: Joint learning of partial anomalies and group structure". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[4] Shaosheng Cao, Wei Lu, and Qiongkai Xu. "Deep neural networks for learning graph representations". In: *Thirtieth AAAI Conference on Artificial Intelligence*. 2016.

[5] Shaosheng Cao, Wei Lu, and Qiongkai Xu. "Grarep: Learning graph representations with global structural information". In: *Proceedings of the 24th ACM international on conference on information and knowledge management*. ACM. 2015, pp. 891–900.

[6] Petr Chunaev. "Community detection in node-attributed social networks: a survey". In: *Computer Science Review* 37 (2020), p. 100286.

[7] Vinıcius da Fonseca Vieira, Carolina Ribeiro Xavier, and Alexandre Gonçalves Evsukoff. "A comparative study of overlapping community detection methods from the perspective of the structural properties". In: *Applied Network Science* 5.1 (2020), pp. 1–42.

[8] Prem K Gopalan and David M Blei. "Efficient discovery of overlapping communities in massive networks". In: *Proceedings of the National Academy of Sciences* 110.36 (2013), pp. 14534–14539.

[9] Dongxiao He et al. "Joint identification of network communities and semantics via integrative modeling of network topologies and node contents". In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.

[10] Binyuan Hui, Pengfei Zhu, and Qinghua Hu. "Collaborative Graph Convolutional Networks: Unsupervised Learning Meets Semi-Supervised Learning." In: *AAAI*. 2020, pp. 4215–4222.

[11] Zhao Kang et al. "Large-Scale Multi-View Subspace Clustering in Linear Time." In: *AAAI*. 2020, pp. 4412–4419.

[12] Zhao Kang et al. "Relation-Guided Representation Learning". In: *Neural Networks* 131 (2020), pp. 93–102.

[13] Zhao Kang et al. "Structured Graph Learning for Scalable Subspace Clustering: From Single-view to Multi-view". In: *IEEE Transactions on Cybernetics* (2021). DOI: 10.1109/TCYB.2021.3061660.

[14] Thomas N Kipf and Max Welling. "Semi-supervised classification with graph convolutional networks". In: *ICLR*. 2017.

[15] Thomas N Kipf and Max Welling. "Variational graph auto-encoders". In: *arXiv preprint arXiv:1611.07308* (2016).

[16] Qimai Li et al. "Label efficient semi-supervised learning via graph filtering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9582–9591.

[17] Sheng Li and Yun Fu. *Robust Representation for Data Analytics*. Springer, 2017.

[18] Ye Li et al. "Community detection in attributed graphs: An embedding approach". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[19] Zhihui Li et al. "Rank-constrained spectral clustering with flexible embedding". In: *IEEE transactions on neural networks and learning systems* 29.12 (2018), pp. 6073–6082.

[20] Zhiping Lin and Zhao Kang. "Graph Filter-based Multi-view Attributed Graph Clustering". In: *Proceedings of the 30th International Joint Conference on Artificial Intelligence, IJCAI*. 2021, pp. 19–26.

[21] Zhiping Lin et al. "Multi-view Attributed Graph Clustering". In: *IEEE Transactions on Knowledge and Data Engineering* (2021). DOI: 10.1109/TKDE.2021.3101227.

[22] Changshu Liu et al. "Self-supervised consensus representation learning for attributed graph". In: *Proceedings of the 29th ACM International Conference on Multimedia*. 2021, pp. 2654–2662.

[23] Yang Liu et al. "Learning with Adaptive Neighbors for Image Clustering." In: *IJCAI*. 2018, pp. 2483–2489.

[24] Bjørn Magnus Mathisen et al. "Learning similarity measures from data". In: *Progress in Artificial Intelligence* (2019).

[25] Julian Mcauley and Jure Leskovec. "Discovering social circles in ego networks". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 8.1 (2014), pp. 1–28.

[26] Antonio Ortega et al. "Graph signal processing: Overview, challenges, and applications". In: *Proceedings of the IEEE* 106.5 (2018), pp. 808–828.

[27] Erlin Pan and Zhao Kang. "Multi-view Contrastive Graph Clustering". In: *Advances in Neural Information Processing Systems* 34 (2021).

[28] Shirui Pan et al. "Adversarially regularized graph autoencoder for graph embedding". In: *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 2018, pp. 2609–2615.

[29] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 701–710.

[30] Dan Shi et al. "Robust structured graph clustering". In: *IEEE transactions on neural networks and learning systems* 31.11 (2019), pp. 4424–4436.

[31] Fan-Yun Sun et al. "vGraph: A generative model for joint community detection and node representation learning". In: *Advances in Neural Information Processing Systems*. 2019, pp. 514–524.

[32] Jian Tang et al. "Line: Large-scale information network embedding". In: *Proceedings of the 24th international conference on world wide web*. 2015, pp. 1067–1077.

[33] Chun Wang et al. "Attributed Graph Clustering: A Deep Attentional Embedding Approach". In: *IJCAI*. 2019.

[34] Chun Wang et al. "Mgae: Marginalized graph autoencoder for graph clustering". In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM. 2017, pp. 889–898.

[35] Daixin Wang, Peng Cui, and Wenwu Zhu. "Structural deep network embedding". In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2016, pp. 1225–1234.

[36] Qi Wang et al. "Spectral embedded adaptive neighbors clustering". In: *IEEE transactions on neural networks and learning systems* 30.4 (2018), pp. 1265–1271.

[37] Xiao Wang et al. "Community preserving network embedding". In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.

[38] Xiao Wang et al. "Semantic community identification in large attribute networks". In: *Thirtieth AAAI Conference on Artificial Intelligence*. 2016.

[39] Felix Wu et al. "Simplifying graph convolutional networks". In: *International conference on machine learning*. PMLR. 2019, pp. 6861–6871.

[40] Jierui Xie, Stephen Kelley, and Boleslaw K Szymanski. "Overlapping community detection in networks: The state-of-the-art and comparative study". In: *Acm computing surveys (csur)* 45.4 (2013), pp. 1–35.

[41] Cheng Yang et al. "Network representation learning with rich text information". In: *Twenty-Fourth International Joint Conference on Artificial Intelligence*. 2015.

[42] Jaewon Yang and Jure 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.

[43] Jaewon Yang, Julian McAuley, and Jure Leskovec. "Community detection in networks with node attributes". In: *2013 IEEE 13th international conference on data mining*. IEEE. 2013, pp. 1151–1156.

[44] Xiaotong Zhang et al. "Attributed Graph Clustering via Adaptive Graph Convolution". In: *IJCAI*. 2019.

[45] Qikui Zhu, Bo Du, and Pingkun Yan. "Multi-hop Convolutions on Weighted Graphs". In: *arXiv preprint arXiv:1911.04978* (2019).