Attributed Graph Subspace Clustering with Graph-Boosting

Wang Li LEON20080518@163.COM
En Zhu✉ ENZHU@NUDT.EDU.CN
Siwei Wang WANGSIWEI13@NUDT.EDU.CN
College of Computer, National University of Defense Technology, Changsha 410073, China

Xifeng Guo GUOXIFENG1990@163.COM
School of Cyberspace Science, Dongguan University of Technology, Guangdong, 523808, China

Editors: Berrin Yanıkoğlu and Wray Buntine

Abstract

Attributed graph clustering groups nodes into disjoint categories with graph convolutional networks and has exhibited promising performance in various applications. However, there are two issues preventing the performance from being improved further. First, the relationships between distant nodes are generally overlooked due to the sparsity of graphs. Second, the graph convolutional networks with few layers are sensitive to noises. To address these issues, we propose Attributed Graph Subspace clustering with Graph-Boosting (AGSGB). Specifically, to deal with the first issue, an auxiliary graph is built from the feature matrix to establish the distant relationships. And to address the second issue, a subspace clustering module, famous for its robustness to noise, is introduced. Based on the auxiliary graph and the subspace clustering module, a graph enhance module and a graph refine module are constructed, together with the graph autoencoder constituting the final clustering model. By using the given graph and the refined graph built by the graph refine module, a dual guidance supervisor is designed to train the clustering model. Finally, the clustering result can be obtained by the subspace clustering module. Extensive experimental results on five public benchmark datasets validate the effectiveness of the proposed method.

Keywords: attributed graph, graph auto-encoder, graph boosting, subspace clustering

1. Introduction

Clustering is the process of dividing data into different groups with some similarity metric. As a fundamental task, graph clustering is widely applied in real-world scenarios such as computer networks Grout and Cunningham (2006), database system Wu et al. (2004) and bio-informatics Boyer et al. (2005). Attributed graph clustering models utilize both structure and attribute for representation learning. The quality of representation is critical to the performance of clustering model. How to exploit both structure and attribute effectively is still an open problem.

Early methods Perozzi et al.; Grover and Leskovec (2016); Tang et al. (2015) are designed for graph embedding by structure only, but ignored node attribute. The classical deep models Hinton and Salakhutdinov (2006); Xie et al. (2016); Guo et al. (2017) implement representation learning on node attribute only. For attributed graph, neither structure-based models nor attribute-based models exploit both structure and attribute, which limits the capacity for representation learning. To solve this problem, researchers

have developed a series of models to utilize structure and attribute for graph representation learning Le and Lauw (2014); Qi et al. (2012); Gao and Huang (2018). Inspired by CNN, Kipf and Welling (2017) proposed a powerful tool termed GCN. GCN can integrate attributes with structure in a natural and effective way. With the help of GCN, various deep graph clustering models have been developed. DAEGC Wang et al. (2019) proposes an attention mechanism for weighing the neighbors. MVGRL Hassani and Ahmadi (2020) proposes to learn representations that including both local and global information by maximizing mutual information between node's representation and sub-graph representation. S^2GC Zhu and Koniusz (2021) applies Laplacian-Smoothing for denoising and achieves promising performance in clustering tasks. Bo et al. (2020) is deeper GCN model that utilizes cross-modality for mitigating the over-smoothness. AutoSSL Jin et al. (2022) designed an adaptive way to fuse multiple pretext tasks to improve the quality of representation.

However, there are still issues need to be addressed: 1) Most of the models mentioned above have overlooked the global relationships. The given graph is usually sparse. When using as a supervisor, it assumes that two nodes are unrelated if there exists no edges between them, which is not appropriate. No edges between nodes doesn't necessarily mean they are unrelated. Worse, the given graph can only aggregate neighbors that are explicitly linked with current node, which may limit the generalizability of the representation. 2) GCN with shallow layers is vulnerable to the noises in graphs, and it may lead a sub-optimal clustering performance.

To address these issues, we propose a novel graph clustering approach termed Attributed Graph Subspace Clustering with Graph-Boosting (AGSGB). To address the first issue, we designed an updating dual-guidance that contains two supervisor: a given graph and an auxiliary graph. The auxiliary graph can establish the distant relationships between nodes. The dual-guidance that consists of both the given graph and the auxiliary graph, can ease the misleadingness by the sparse given graph. And the graph refine module can regularly improve the dual-guidance by building an new auxiliary graph. Moreover, the given graph is strengthened by the inital auxiliary graph in the propagation step of our model. The generalization of the representation is improved. To address the second issue, we introduce a subspace module which is characterized by self-expressive learning. From GAE to subspace, the attribute-based embeddings are transformed into relationship-based ones, which means each node is represented by the rest in the whole graph. By this, the robustness of representations can be enhanced. Because of a shallow GCN, the discriminativeness of representations can be assured. When GAE interplays with the subspace clustering model, the proposed model can keep the representations both discriminative and robust. Extensive experiments on five datasets have proved the effectiveness of our model.

The contributions of our work are summarized as follows:

- We introduce a graph-boosting part that consists of a graph enhance module, a graph refine module, and a dual-guidance. Graph representation learning can be improved with the help of graph boosting.
- We design an interplay between the subspace clustering module and graph boosting to improve both the robustness of shallow GCN and the dual guidance.
- Extensive experiments on five public benchmarks validate the effectiveness of our method and the result shows the proposed method outperforms its competitors.

2. Related Work

2.1. Attributed Graph Clustering

For GCN can provide a simple and effective way to integrate graph structure with node attributes, kinds of attributed graph clustering models have been developed based on it. Graph auto-encode (GAE) is one of the most popular unsupervised representation learning models. ARGE Pan et al. (2018) introduces an adversarial regularization module to force the embeddings to follow a standard normal distribution ($\mathbb{N} \sim (0, 1)$), thus improving the robustness of the representation. To adaptively choose a proper neighborhood for diverse graphs, AGC Zhang et al. (2019) proposes a measurement to evaluate the intra-cluster distance. Some models propose to introduce clustering loss as auxiliary guidance for representation learning. In this way, the model can generate more clustering-friendly representations, which facilitate clustering tasks. One of the most widely used clustering losses is proposed by DEC Xie et al. (2016). For example, SDCN Bo et al. (2020) and AGCN Peng et al. (2021) are clustering-oriented deep models that integrate AutoEncoder with GCN to overcome the over-smoothness.

2.2. Subspace Clustering

Self-expressive learning is a widely used tool in subspace clustering. It learns a set of coefficients that stand for a linear combination of correlated points lying in the same subspace as the target one. The goal of self-expressive learning is to construct new representations for data points. The new representations are supposed to be robust to noises or outliers. LRR, SSC and LSR Liu et al. (2010); Elhamifar and Vidal (2013); Lu et al. (2012) are traditional self-expressive learning models that can improve the representations by regulating the coefficient matrix in different ways. DSC Ji et al. (2017) is a subspace model that applied deep learning to subspace clustering. It validated that self-expressive learning on deep embeddings can achieve promising improvement in clustering task. Inspired by DSC, we introduce a subspace module that is characterized by self-expressive learning into our model for robust representation learning.

2.3. Structure Refining

In real-world scenario, the structure of graphs are often noisy. Therefore, how to improve graph attracts more attention. To obtain a robust graph, Pro-GNN Jin et al. (2020) proposed to learn a graph by forcing a parameter matrix to be sparse and low-rank. IDGL Chen et al. (2020) proposed an interaction mechanism to alternatively improve graph structure and node embeddings. Sublime Liu et al. (2022) can learn robust and generic node embeddings by performing contrastive learning on embeddings of different views.
The models mentioned before have been proven to be effective, but there are still issues to be addressed. For example, most of them are supervised by a given graph which is sparse. If two nodes are linked, they are expected to be similar, otherwise dissimilar. In fact, nodes that are not linked are not necessarily dissimilar.

3. Proposed Method

In this paper, our model consists of five parts: graph enhance, graph auto-encoder, subspace clustering, graph refine and dual guidance. For a clear understanding to the pipeline of our model, we will introduce notations, graph enhance, graph auto-encoder, subspace clustering, graph refine, dual guidance and optimization sequentially in the following sections. Figure1 shows the proposed model in detail.

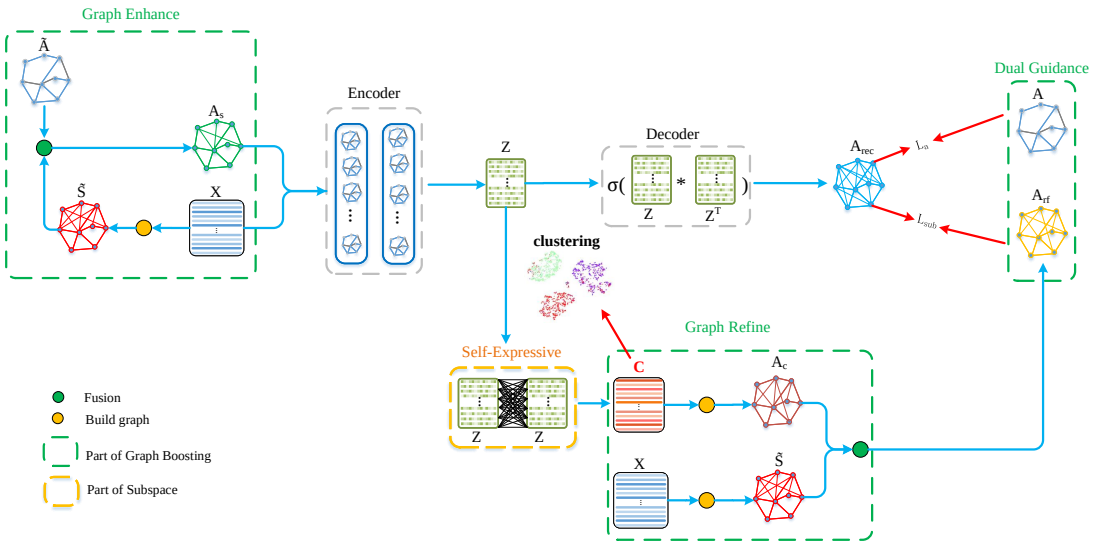


Figure 1: This is the framework of AGSGB. It consists of two parts: the upper part is GAE and the lower part is the subspace module. First, we create an enhanced graph A_s by graph enhance module. We input A_s and the raw feature X into the encoder. Z is the output of the encoder. Subspace module performs self-expressive learning on Z to get a coefficient matrix C, in which each node can capture the global relationship. With C and X, the graph refine module can improve dual guidance by replacing the old refined graph A_{rf} with a new one. The improved dual guidance can enhance the quality of Z, which can furtherly improve the dual guidance through the subspace moduel and graph refine module. In this way, the subspace module and GAE mutually benefit.

3.1. Problem Definition

Given an undirected graph $G = (V, E)$, $V = \{v_1, v_2..v_n\}$ and E denote a node-set and an edge set respectively. X$\in \mathbb{R}^{n\times d}$ denotes an attribute matrix for all nodes. A$= (a_{ij})_{n\times n}$ denotes an adjacent matrix. If $(v_i, v_j) \in E$, then $a_{ij} = 1$, otherwise $a_{ij} = 0$. D$= diag(d_1, d_2, ..., d_N) \in \mathbb{R}^{n\times n}$ is a degree matrix. $d_i = \sum_{j=1}^{n} a_{ij}$.

3.2. Graph Enhance

The given graph A is used for aggregating neighbors in the propagation stage, which is a way to build a relative generic representation for the target node. However, A has a limitation in building generic representation for its focusing on local relationships. To improve this, we build an auxiliary graph to enhance the generalization of representations in the propagation stage. To start, we simply build a similarity matrix with raw features X, whose processing is like the following:

$$\mathrm{S} = cos(\mathrm{X}, \mathrm{X}), \tag{1}$$

where S denotes a symmetric densely connected graph. Because it is built by X, it contains some noise unavoidably. To improve the quality of the auxiliary graph, we need an extra step for filtering noises. In this work, we propose a Top function to choose the k largest values in each row of S:

$$\mathrm{S}^k = Top(\mathrm{S}, k). \tag{2}$$

The function of Top is as follows:

$$\mathrm{S}^k{}_{ij} = \begin{cases} \mathrm{S}_{ij} & if \quad \mathrm{S}_{ij} \geq \mathrm{S}^r{}_{ik} \\ \\ 0 & else \end{cases}. \tag{3}$$

where each row of S^r is sorted in descending order, $\mathrm{S}^r{}_i = [\mathrm{S}^r{}_{i1}, \mathrm{S}^r{}_{i2} ... \mathrm{S}^r{}_{ik}, ... \mathrm{S}^r{}_{in}] \in \mathbb{R}^{1\times n}$ denotes the i-th row of S, $\mathrm{S}^r{}_{ip} \geq \mathrm{S}^r{}_{iq}$, $p < q$. After Top function, we normalize S^k with following equation:

$$\widetilde{\mathrm{S}}_{ij} = \frac{\mathrm{S}^k{}_{ij}}{\sum_{t=1}^{n} \mathrm{S}^k{}_{it}}. \tag{4}$$

For the purpose of enhancing the graph, we fuse $\mathbb{S}$ and A with a weighted way:

$$\mathrm{A_s} = \alpha\widetilde{\mathrm{S}} + (1-\alpha)\widetilde{\mathrm{A}}. \tag{5}$$

In (5), $\widetilde{\mathrm{A}} = \mathrm{D}^{-1}(\mathrm{I} + \mathrm{A})$. The enhanced graph A_s can hold a wider neighborhood than A, which will facilitate the propagation phase.

3.3. Graph Auto-Encoder

In this work, we use a vanilla graph auto-encoder. It consists of two parts: a 2-layer GCN is utilized as an encoder and an inner-product operation is a decoder. During the training, the given graph A is used as a supervisor, In A, each element is valued by $\{0, 1\}$ when there is a connection between v_i and v_j, then $a_{ij} = 1$, otherwise, $a_{ij} = 0$. The enhanced graph A_s is utilized for aggregation. GCN integrates attributes X with the enhanced graph A_s to produce embeddings by the following equation:

$$\mathrm{Z}_{l+1} = \phi(\mathrm{A}_s\mathrm{Z}_l\Theta), \tag{6}$$

where l denotes the number of layers, Z_0 =X, Θ denotes the parameters of the encoder. ϕ denotes an activate function.

After encoding, an inner-product operation is used as a decoder for graph reconstruction. The process is as follows:

$$\mathrm{A}_{rec} = \sigma(\mathrm{Z}_L {\mathrm{Z}_L}^T), \tag{7}$$

where L is the index of the output layer of the encoder. A_{rec} denotes the reconstructed matrix, and $\mathrm{Z}_L \in n \times d_L$ denotes the output of the encoder. For basic representation learning, we force A_{rec} to approximate the given graph:

$$L_a = \frac{1}{2n} \|\widetilde{\mathrm{A}} - \mathrm{A}_{\mathrm{rec}}\|_F^2, \tag{8}$$

3.4. Subspace Clustering

In this paper, the subspace clustering module is characterized by self-expressive learning. It can improve the robustness of the representation and obtain clustering results. We implement self-expressive learning on the output of the encoder. The process is as follows:

$$\begin{aligned} L_{sub} &= \gamma \|\mathrm{Z}_L - \mathrm{C}\mathrm{Z}_L\|_F^2 + \|\mathrm{C}\|_2^2 \\ s.t. &Diag(\mathrm{C}) = 0, \end{aligned} \tag{9}$$

where C is learned by minimizing the self-expressive learning loss. $Diag(\mathrm{C}) \in \mathbb{R}^n$ is a vector of diagonal elements in C. After obtaining an optimized coefficient matrix C^*, instead of building a typical affinity graph by computing $|\mathrm{C}^*| + |{\mathrm{C}^*}^T|$ for spectral clustering, we build it by the heuristics which can preserve a better block structure Ji et al. (2017, 2014). The clustering result is obtained by performing a spectral clustering on the final affinity graph.

3.5. Graph Refine

To further improve the quality of dual guidance, we designed a graph refine module. Specifically, we use $\widetilde{\mathrm{S}}$ as a fundamental graph to keep the performance at an acceptable level. To refine the dual guidance, we build an affinity graph by C. Compared to A, C can capture the global relationships between distant nodes. It can facilitate improving the quality of dual guidance. We build a refining graph $\widetilde{\mathrm{A}}_c$ regularly to improve the quality of dual-guidance, the process is as follows:

$$\mathrm{S}_c = cos(\mathrm{C}, \mathrm{C}), \tag{10}$$

$${\mathrm{S}_c}^k = Top(\mathrm{S}_c, k), \tag{11}$$

$$\mathrm{A}_{c_{ij}} = \frac{{{\mathrm{S}_c}^k}_{ij}}{\sum_{k=1}^{n} {{\mathrm{S}_c}^k}_{ik}}. \tag{12}$$

In practice, instead of refining graph A_{rf} for each iteration, we update it by every T iteration. After obtaining A_{rf} by an weighted fusion of $\widetilde{\mathrm{S}}$ and A_c, we renew the A_{rf} as following equation:

$$\mathrm{A}_{rf} = \beta \mathrm{A}_c + (1 - \beta)\widetilde{\mathrm{S}}. \tag{13}$$

3.6. Dual Guidance

To improve the quality of the supervisor, we propose the dual-guidance by introducing another auxiliary graph (which is initiated as S) to cooperate with the given one. During the training, we force A_{rec} to approximate both the normalized given graph $\widetilde{\mathrm{A}}$ and the auxiliary graph A_{rf}:

$$L_a = \frac{1}{2n}\|\widetilde{\mathrm{A}} - \mathrm{A}_{\mathrm{rec}}\|_F^2, \tag{14}$$

$$L_s = \frac{1}{2n}\|\mathrm{A}_{rf} - \mathrm{A}_{rec}\|_F^2. \tag{15}$$

We minimize both loss functions in a joint way which can be seen as dual guidance:

$$\begin{aligned} L_{rec} =& L_s + \lambda L_a. \\ =& \frac{1}{2n}\|\mathrm{A}_{rf} - \mathrm{A}_{rec}\|_F^2 + \lambda\frac{1}{2n}\|\widetilde{\mathrm{A}} - \mathrm{A}_{\mathrm{rec}}\|_F^2 \end{aligned} \tag{16}$$

3.7. Optimization

On one hand, the dual guidance can improve the quality of representations generated by GCN. On the other hand, the improved GCN can facilitate self-expressive learning, which can benefit dual guidance subsequently. The following equation shows this mutually optimize strategy:

$$L_{total} = L_{rec} + L_{sub}. \tag{17}$$

The details of optimization are shown in Algorithm 1.

3.8. Complexity Analysis

In our work, we denote d as the max dimension, n as the number of nodes, $|E|$ as the number of edges, t is the number of iteration. The computational complexity of our model is $O(t(|E|d^2 + n^2 d))$. Specifically, the complexity of GAE is $O(|E|d^2 + n^2 d)$. For additional modules, the complexity of graph enhance, graph refine and the self-expressive learning are all $O(n^2 d)$. Thus, the total computational complexity of the proposed model is $O(t(|E|d^2 + n^2 d))$.

4. Experiment

4.1. Datasets

We implement experiments on five benchmark datasets: Cora, Citeseer, Acm, Dblp, and Wiki. In Wiki, the features are tf-idf weighted vectors. For the other datasets, the features of nodes are represented by binary vectors. More details can be found in table 1.

Algorithm 1 Attributed Graph Subspace Clustering with Graph-Boosting

Input: Attribute matrix X, adjacent matrix A, iteration number T, refine interval $interval$, cluster number K, hyper-parameter $\alpha, \beta, \lambda, \gamma$

Build $\widetilde{\mathrm{S}}$ by (1), (2), and (4)
$\mathrm{A}_{rf} = \widetilde{\mathrm{S}}$
Build A_s by (5)
for $t = 1$ to T **do**
  Generate the embeddings Z_L by (6)
  Reconstruct affinity graph A_{rec} with Z_L by (7)
  Calculate the reconstruction loss by (14), (15)
  Calculate loss of self-expressive learning and obtain C by (9)
  if t mod $interval == 0$ **then**
    Build affinity graph A_c with C by (10), (11), and (12)
    renew A_{rf} by (13)
  end if
  Calculate the reconstruction loss by (16)
  Update the whole framework by (17)
end for
Output: The optimized parameters C

4.2. Baselines

Our model is compared to 16 baseline methods. K-means is a well-known basic clustering algorithm. AE, DEC, and IDEC Hinton and Salakhutdinov (2006); Xie et al. (2016); Guo et al. (2017) are content-dependent clustering model. GAE&VGAE, ARGE&ARVGE, DAEGC, and MGAE Kipf and Welling (2016); Pan et al. (2018); Wang et al. (2019, 2017) are GCN-based typical methods that are trained by reconstructing given data. SDCN, AGCN, and DFCN Bo et al. (2020); Peng et al. (2021); Tu et al. (2021) are hybrid methods that integrate multi-modules for clustering. Sublime, MVGRL, and AutoSSL Liu et al. (2022); Hassani and Ahmadi (2020); Jin et al. (2022) are contrastive learning based graph clustering methods.

Table 1: Benchmark Datasets

Dataset	Nodes	Dimension	Clusters	Edges
Cora	2708	1433	7	5429
Citeseer	3327	3703	6	4732
Dblp	4058	334	4	7056
Acm	3025	1870	3	26256
Wiki	2405	4973	17	17981

4.3. Metrics and Implement Details

The proposed model is trained end-to-end. We use a 2-layer GCN module as an encoder, the dimension of each layer is $d - 512 - 16$, d is the dimension of input. We use LeakRelu as activation function to the first layer. All weight matrices are initialized with Xavier Uniform. We set $\gamma = 0.1, \lambda = 0.1$ for all datasets empirically. The learning rate is 10^{-3} for Dblp and Acm, and 10^{-4} for Cora, Citeseer, and Wiki. In the subspace module, we initialize all elements in the self-expressive coefficient matrix C to 10^{-8}. We set $k = 100$ for all datasets in the *top* function. To obtain the clustering results, we perform a subspace clustering Ji et al. (2014). For AE, GAE, and VGAE, we perform K-means for results. The settings for AE, DEC, and IDEC follow Hinton and Salakhutdinov (2006); Xie et al. (2016); Guo et al. (2017). The settings for the rest follow their corresponding papers. All experiments are implemented with PyTorch and a GPU (GeForce RTX 1080 Ti 12G). We repeat each experiment 10 times and report the average results. We employ four widely used metrics to evaluate the clustering performance: Accuracy (ACC), Normalized Mutual Information (NMI), Average Rand Index (ARI), and macro F1-score (F1).

4.4. Results and Analysis

The clustering performance of the proposed model and 16 baseline methods on 5 benchmark datasets are given in table 2.

Table 2: Clustering results(%) on 5 datasets, red numbers denote the best results.

Dataset	Metrics	k-means	AE	DEC	IDEC	GAE	VGAE	MGAE	ARGE	ARVGE	DAEGC	SDCN	MVGRL	DFCN	AutoSSL	AGCN	Sublime	AGSGB
Cora	ACC	40.25	40.50	48.32	49.00	59.03	34.37	68.06	64.0	63.8	66.42	47.03	70.47	36.33	63.81	60.56	71.30	71.71
	NMI	25.08	21.26	28.01	28.83	46.83	13.41	48.92	44.9	45.0	48.00	25.54	55.57	19.36	47.62	43.59	54.20	54.25
	ARI	15.35	15.22	21.12	22.09	38.20	9.12	43.61	35.2	37.4	42.21	20.05	48.70	4.67	38.92	35.46	50.30	48.45
	F1	40.62	38.21	47.53	48.85	56.09	32.59	53.12	61.9	62.7	63.93	40.46	67.15	26.16	56.42	49.76	63.50	67.47
	Time(s)	3.8	4.2	24.4	73.9	6.1	6.0	9.3	4.5	4.5	9.0	16.7	20.8	7.8	82.1	15.4	148.2	10.2
Citeseer	ACC	55.06	53.93	60.96	63.16	60.55	51.41	66.91	57.3	54.4	64.54	65.96	62.83	69.73	66.76	68.79	68.50	70.76
	NMI	29.21	27.56	33.36	36.54	36.34	28.96	41.58	35.0	26.1	36.41	38.71	40.69	44.93	40.67	41.54	44.10	45.54
	ARI	24.56	26.03	33.20	36.75	35.50	24.88	42.50	34.1	24.5	37.78	40.17	34.18	45.31	38.73	43.79	43.90	47.51
	F1	53.03	50.53	57.13	60.37	56.24	49.48	52.57	54.6	52.9	62.20	63.62	59.54	64.45	58.22	62.37	63.20	66.08
	Time(s)	10.4	8.6	37.9	119.9	15.2	15.3	89.8	11.2	11.3	18.0	20.3	27.8	8.6	66.5	19.6	173.4	20.9
Acm	ACC	68.17	78.55	72.52	78.33	84.52	76.78	83.00	83.65	83.06	86.94	90.45	86.73	90.90	87.94	90.59	91.20	92.48
	NMI	33.40	44.53	43.50	50.83	64.69	43.33	52.61	52.11	55.38	56.18	68.31	60.87	69.40	63.20	68.38	71.50	73.55
	ARI	31.29	46.98	43.48	51.52	70.47	41.14	56.94	57.08	59.46	59.35	73.91	65.07	74.90	67.91	74.20	75.90	79.05
	F1	68.42	78.69	70.60	76.44	89.05	76.96	71.40	81.40	84.65	87.07	90.42	86.85	90.80	87.96	90.58	91.13	92.47
	Time(s)	4.4	5.3	18.5	94.5	8.7	8.9	16.6	7.0	7.1	10.9	14.5	24.7	6.7	44.2	16.9	94.5	13.8
Dblp	ACC	38.35	38.62	61.46	55.92	53.42	53.06	74.49	61.94	64.44	62.05	68.05	44.91	76.00	40.52	73.26	56.80	81.74
	NMI	10.99	14.03	27.53	24.56	29.29	28.87	41.67	25.63	30.21	32.49	39.50	18.75	43.70	12.63	39.68	27.25	52.35
	ARI	6.68	7.41	25.25	18.37	16.83	16.65	45.81	23.91	26.21	21.03	39.15	11.14	47.00	5.41	42.49	19.17	58.08
	F1	32.10	31.72	61.82	56.82	54.90	54.34	59.67	60.57	64.32	61.75	67.71	44.80	75.70	37.78	72.80	51.05	81.28
	Time(s)	2.7	5.0	19.3	102.8	7.5	7.9	6.1	5.3	5.6	12.0	17.6	31.3	8.9	83.6	17.2	91.4	15.6
Wiki	ACC	31.03	34.01	41.32	40.43	31.51	46.94	52.93	38.10	38.70	30.64	40.15	30.72	37.33	44.61	38.33	48.81	53.33
	NMI	27.42	31.03	41.22	37.67	30.10	44.59	51.04	34.50	33.90	22.05	38.76	26.80	30.74	41.68	38.74	52.23	51.27
	ARI	3.69	9.49	20.52	16.99	13.47	28.41	37.87	11.20	10.70	11.61	20.46	8.14	15.81	23.64	19.02	26.82	36.12
	F1	17.90	24.01	28.69	28.07	27.18	40.69	42.94	38.10	38.70	25.98	29.00	23.77	27.57	35.71	27.12	42.85	42.70
	Time(s)	12.6	9.1	169.9	192.3	13.3	13.3	266.8	12.5	12.5	18.1	21.3	34.4	6.2	127.5	20.1	32.9	14.8

According to results , our model greatly outperforms those only use attributes for representation learning. This is because the proposed model fully exploits the information from both structure and attribute. K-means, AE, DEC, and IDEC only use attribute for representation learning, which limits the ability of generalization.

The proposed model outperforms shallow GCN models such as GAE, VGAE, ARGE, ARVGE, MGAE and DAEGC. Integrating structure with the attribute, these GCN models can achieve better performance than models that use only attribute in graph clustering task. However, these models are vulnerable to noise, which limit their improvement. Integrating

GCN with subspace clustering module, the proposed model can improve the robustness of representations and keep discriminative to a degree.

The proposed model achieves better performance than deep graph clustering models SDCN, AGCN and DFCN. Especially, DFCN is the second best method in Dblp. Our method can outperforms it by 5%, 9%, 11%, and 6% with respect to ACC, NMI, ARI and F1. These methods can balance the discriminativeness and the robustness in representation learning by integrating AutoEncoder with GCN and achieve much better performance. Although alleviating the over-smoothness, they lack a reliable supervisor for training. AGSGB is designed to address these issues by introducing a graph boosting mechanism to improve the robustness and provide a more reliable supervisor for representation learning.

The proposed model can outperforms Sublime, MVGRL and AutoSSL in most cases. These methods propose to improve quality of representations by contrastive learning. Observed from table 2, these methods can achieve promising performances by keeping discriminativeness. However, their performances are also limited by noise while the proposed model can learn representations of robust to noise.

Table 3: Ablation study on four datasets. Sub stands for the subspace module, and GB stands for the graph-boosting module. The **Bold** values denote the best results.

Dataset	Model	ACC	NMI	ARI	F1
Cora	Baseline	59.03	46.83	38.20	56.09
	+Sub	69.75	53.05	45.35	65.67
	+GB	61.06	48.67	41.11	56.65
	AGSGB	**71.71**	**54.25**	**48.45**	**67.47**
Citeseer	Baseline	60.55	36.34	35.50	56.24
	+Sub	65.77	40.70	40.37	59.90
	+GB	70.63	45.49	47.38	65.96
	AGSGB	**70.76**	**45.53**	**47.50**	**66.08**
Dblp	Baseline	43.32	20.48	18.18	37.80
	+Sub	69.13	37.46	40.06	67.85
	+GB	80.27	51.12	55.00	79.98
	AGSGB	**81.74**	**52.35**	**58.08**	**81.28**
Acm	Baseline	89.06	64.69	70.47	89.05
	+Sub	90.47	68.16	73.92	90.49
	+GB	91.98	72.30	77.69	92.01
	AGSGB	**92.48**	**73.55**	**79.05**	**92.47**
Wiki	Baseline	31.51	30.10	13.47	27.18
	+Sub	42.77	37.26	24.37	31.78
	+GB	47.90	50.50	29.84	41.24
	AGSGB	**53.33**	**51.27**	**36.12**	**42.70**

4.5. Ablation Study

We conduct an ablation study from two perspectives: first, we analyze the effectiveness of the subspace module, graph-boosting, and refining; second, we analyze the impact on

performance by removing each module of graph boosting. In the first ablation study, the baseline is GAE. For the first study, we perform analysis in an accumulation manner. From table 3, we can see a tendency for consistent improvement in performance as a new module is added. Compared to the baseline, the proposed model achieved improvement in ACC are $12\%, 10\%, 38\%, 3\%, 22\%$ for Cora, Citeseer, Dblp, Acm, and Wiki respectively. We found that the improvements by both the subspace module and graph boosting are obvious. Improvement by subspace clustering demonstrates its ability to learn representations that keep both discriminative and robust. While improvement by graph boosting validates the effectiveness of introducing an auxiliary graph for facilitating both propagation and supervising.

To further understand the effectiveness of graph boosting, we conduct another study whose result is shown in table 4. From the table, we can observe that the performance will decrease to different degrees when removing different components of graph boosting. This validates the effectiveness of each component in graph boosting. For Cora and Citeseer, removing DG has the slightest impact on performance, while for the other datasets, removing GR impacts the least on performance. To sum up, removing GE has the most impact on performance. According to the result, we infer that in our task the ability to generalization plays a critical role in graph representation learning.

4.6. Sensitivity Analysis on Hyper-parameters

In this paper, we have introduced 2 hyper-parameters λ and γ. In (9), γ is a trade-off between the self-expressive learning and regularization of the coefficient matrix. In (16), λ is used to balance the importance of reconstructing the auxiliary graph and reconstructing the given graph. Here, we jointly analyze them to see how they influence performance. Both parameters vary in a range of $\{0.01, 0.1, 1, 10, 100\}$. Figure 2 illustrates how the hyper-parameters influence the performance variation of AGSGB when λ and γ vary from 0.01 to 100. From these figures, we can observe that: 1) in most cases, the performance can keep performance stable in a considerable range. 2) compared to λ, γ plays a more critical role in AGSGB. 3) AGSGB performs best on all datasets when γ is set to 0.1.

Another hyper-parameter k is used to improve the auxiliary graph by filtering noise. We have conducted experiments to figure out how k influences the performance too. To show the influence, we vary k in the range $\{0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200\}$. As we can observe from table 5, the performance is improving as the K increase. Taking a overall performance for all datasets, we find that the optimal results are achieved when k equals 100.

4.7. Visulization of t-SNE

To validate the effectiveness of our model intuitively, we employ t-SNE to obtain the distribution of embeddings Z. From Figure3 we can observe that the boundaries of clusters by our model are more clear than that of other methods.

Table 4: Study about how each module in Graph Boosting impact performance. In this table, w/o stands for "without", DG denotes Dual-Guidance module, GE stands for graph enhance model, GR stands for graph refine, and Sub denotes Subspace clustering module. The **bold** numbers denote the best results.

Dataset	Model	ACC	NMI	ARI	F1
Cora	w/o *DG*	70.90	53.76	46.75	66.74
	w/o *GE*	70.27	53.43	46.57	65.59
	w/o *GR*	70.83	52.97	46.98	65.71
	AGSGB	**71.71**	**54.25**	**48.45**	**67.47**
Citeseer	w/o *DG*	70.11	45.08	46.54	65.34
	w/o *GE*	69.13	43.98	44.84	64.30
	w/o *GR*	69.38	44.00	45.51	64.89
	AGSGB	**70.76**	**45.54**	**47.50**	**66.08**
Dblp	w/o *DG*	80.13	49.58	55.34	79.39
	w/o *GE*	80.23	49.53	55.40	79.60
	w/o *GR*	81.15	51.31	56.95	80.69
	AGSGB	**81.74**	**52.35**	**58.08**	**81.28**
Acm	w/o *DG*	92.40	73.11	78.80	92.39
	w/o *GE*	91.11	70.49	75.64	91.11
	w/o *GR*	92.46	73.50	79.01	92.46
	AGSGB	**92.48**	**73.55**	**79.05**	**92.47**
Wiki	w/o *DG*	50.79	51.00	34.41	40.84
	w/o *GE*	47.29	40.46	28.01	34.89
	w/o *GR*	52.87	51.24	**36.18**	42.11
	AGSGB	**53.33**	**51.27**	36.12	**42.70**

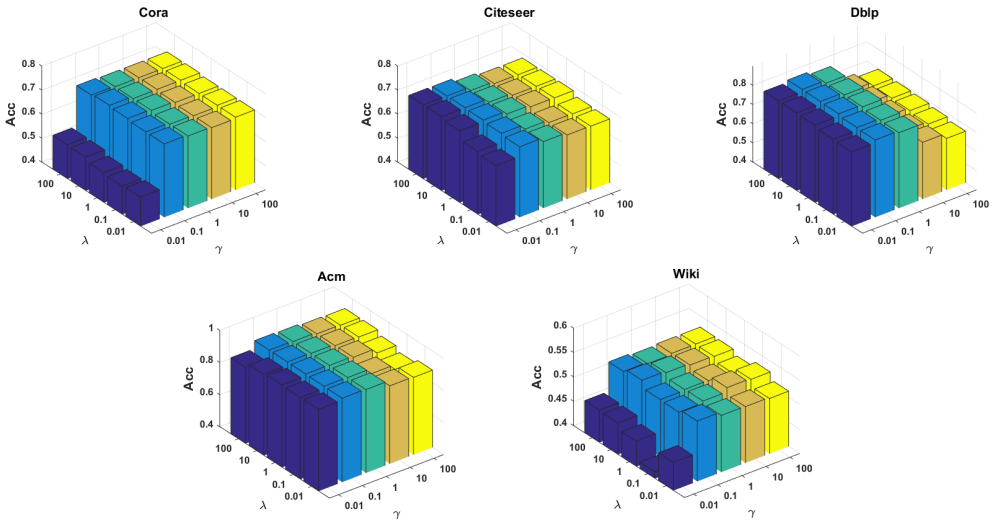
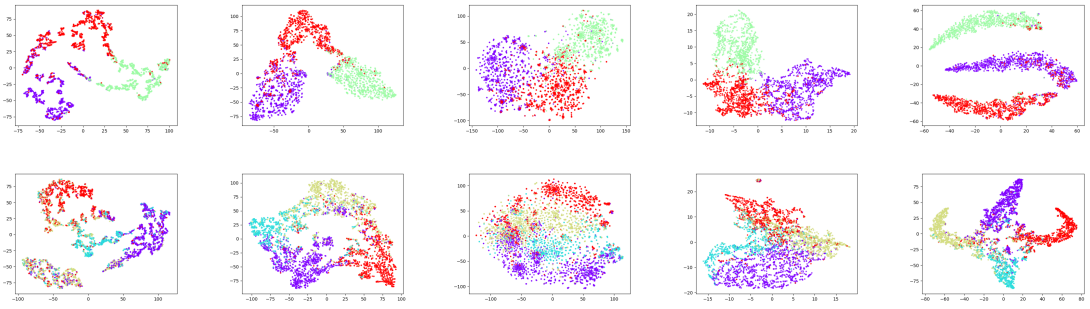


Figure 2: The sensitivity of AGSGB with the variation of λ and γ on five datasets

Table 5: ACC(%) by different k.

k	Cora	Citeseer	Dblp	Acm	Wiki
0	66.06	61.91	66.96	35.64	41.59
10	70.42	68.98	80.06	91.81	42.50
20	70.98	69.76	80.96	91.98	46.63
30	70.97	70.03	81.66	91.96	47.06
40	71.06	70.41	81.74	92.28	49.72
50	71.17	70.47	81.83	92.33	50.84
60	71.59	70.50	81.77	92.34	51.49
70	71.63	70.52	**81.95**	**92.54**	52.10
80	71.60	70.56	81.88	92.39	50.19
90	**71.73**	70.67	81.93	92.48	52.58
100	71.71	**70.76**	81.74	92.48	**53.33**
200	70.38	70.61	81.39	91.42	53.28



(*a*) AGCN (*b*) DFCN (*c*) AutoSSL (*d*) Sublime (*e*) Ours

Figure 3: The t-SNE distribution of Acm (top) and Dblp (bottom). From left column to right: AGCN, DFCN, AutoSSL, Sublime, and AGSGB.

5. Conclusion

In this paper, we have analyzed the issues of the ignorance of distant relationships in sparsity graphs and the sensitivity of shallow GCN to noises. To address these issues, we proposed a novel attributed graph based subspace clustering model in a self-supervised manner. We showed that the graph-boosting mechanism can not only improve the quality of the supervisor but also enhance the generalization of representations. Cooperating with GCN, the introduced subspace clustering module can produce both discriminative and robust representations. The ablation study has proved the effectiveness of each module. Extensive experiments on five widely used datasets validate the effectiveness of our model. The future directions include reducing the computational overhead, and adapting our model to multi-view clustering task.

Acknowledgments

This work is supported by National Key R&D Program of China under Grant No. 2022ZD0209103 and the National Natural Science Foundation under Grant No. 62206054.

References

Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. Structural deep clustering network. In *WWW '20, Taipei, Taiwan, April 20-24, 2020*, pages 1400–1410, 2020.

Frédéric Boyer, Anne Morgat, Laurent Labarre, Joël Pothier, and Alain Viari. Syntons, metabolons and interactons: an exact graph-theoretical approach for exploring neighbourhood between genomic and functional data. *Bioinformatics*, 21(23):4209–4215, 2005.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. In *Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Ehsan Elhamifar and René Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2765–2781, 2013.

Hongchang Gao and Heng Huang. Deep attributed network embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 3364–3370, 2018.

V. M. Grout and Stuart Cunningham. A constrained version of a clustering algorithm for switch placement and interconnection in large networks. In *CAINE*, 2006.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 855–864, 2016.

Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 1753–1759, 2017.

Kaveh Hassani and Amir Hosein Khas Ahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4116–4126, 2020.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313:504–507, 2006.

Pan Ji, Mathieu Salzmann, and Hongdong Li. Efficient dense subspace clustering. In *IEEE Winter Conference on Applications of Computer Vision, Steamboat Springs, CO, USA, March 24-26, 2014*, pages 461–468, 2014.

Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian D. Reid. Deep subspace clustering networks. In *Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 24–33, 2017.

Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 66–74, 2020.

Wei Jin, Xiaorui Liu, Xiangyu Zhao, Yao Ma, Neil Shah, and Jiliang Tang. Automated self-supervised learning for graphs. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*, 2022.

Thomas N. Kipf and Max Welling. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

Tuan M. V. Le and Hady Wirawan Lauw. Probabilistic latent document network embedding. In *2014 IEEE International Conference on Data Mining, ICDM 2014, Shenzhen, China, December 14-17, 2014*, pages 270–279, 2014.

Guangcan Liu, Zhouchen Lin, and Yong Yu. Robust subspace segmentation by low-rank representation. In *(ICML-10), June 21-24, 2010, Haifa, Israel*, pages 663–670, 2010.

Yixin Liu, Yu Zheng, Daokun Zhang, Hongxu Chen, Hao Peng, and Shirui Pan. Towards unsupervised deep graph structure learning. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 1392–1403, 2022.

Can-Yi Lu, Hai Min, Zhong-Qiu Zhao, Lin Zhu, De-Shuang Huang, and Shuicheng Yan. Robust and efficient subspace segmentation via least squares regression. In *ECCV, Florence, Italy, October 7-13, 2012, Proceedings, Part VII*, volume 7578, pages 347–360, 2012.

Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularized graph autoencoder for graph embedding. In *IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 2609–2615, 2018.

Zhihao Peng, Hui Liu, Yuheng Jia, and Junhui Hou. Attention-driven graph clustering network. In *MM '21, Virtual Event, China, October 20 - 24, 2021*, pages 935–943, 2021.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 701–710.

Guo-Jun Qi, Charu C. Aggarwal, Qi Tian, Heng Ji, and Thomas S. Huang. Exploring context and content links in social media: A latent space method. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(5):850–862, 2012.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. LINE: large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, pages 1067–1077, 2015.

Wenxuan Tu, Sihang Zhou, Xinwang Liu, Xifeng Guo, Zhiping Cai, En Zhu, and Jieren Cheng. Deep fusion clustering network. In *AAAI , Virtual Event, February 2-9, 2021*, pages 9978–9987, 2021.

Chun Wang, Shirui Pan, Guodong Long, Xingquan Zhu, and Jing Jiang. MGAE: marginalized graph autoencoder for graph clustering. In *CIKM 2017, Singapore, November 06 - 10, 2017*, pages 889–898, 2017.

Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. Attributed graph clustering: A deep attentional embedding approach. In *IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3670–3676, 2019.

Andrew Y Wu, Michael Garland, and Jiawei Han. Mining scale-free networks using geodesic clustering. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 719–724, 2004.

Junyuan Xie, Ross B. Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *Proceedings of the 33nd International Conference on Machine Learning*, volume 48, pages 478–487, 2016.

Xiaotong Zhang, Han Liu, Qimai Li, and Xiao-Ming Wu. Attributed graph clustering via adaptive graph convolution. In *IJCAI 2019, Macao, China, August 10-16, 2019*, pages 4327–4333, 2019.

Hao Zhu and Piotr Koniusz. Simple spectral graph convolution. In *ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.