# Self-Contrastive Graph Diffusion Network

Yixuan Ma
mayx2021@lzu.edu.cn
School of Information Science and Engineering,
Lanzhou University
Lanzhou, China

Kun Zhan*
kzhan@lzu.edu.cn
School of Information Science and Engineering,
Lanzhou University
Lanzhou, China

## ABSTRACT

Contrastive learning has been proven to be a successful approach in graph self-supervised learning. Augmentation techniques and sampling strategies are crucial in contrastive learning, but in most existing works, augmentation techniques require careful design, and their sampling strategies can only capture a small amount of intrinsic supervision information. Additionally, the existing methods require complex designs to obtain two different representations of the data. To overcome these limitations, we propose a novel framework called the Self-Contrastive Graph Diffusion Network (SCGDN). Our framework consists of two main components: the Attentional Module (AttM) and the Diffusion Module (DiFM). AttM aggregates higher-order structure and feature information to get an excellent embedding, while DiFM balances the state of each node in the graph through Laplacian diffusion learning and allows the cooperative evolution of adjacency and feature information in the graph. Unlike existing methodologies, SCGDN is an augmentation-free approach that avoids "sampling bias" and semantic drift, without the need for pre-training. We conduct a high-quality sampling of samples based on structure and feature information. If two nodes are neighbors, they are considered positive samples of each other. If two disconnected nodes are also unrelated on $k$NN graph, they are considered negative samples for each other. The contrastive objective reasonably uses our proposed sampling strategies, and the redundancy reduction term minimizes redundant information in the embedding and can well retain more discriminative information. In this novel framework, the graph self-contrastive learning paradigm gives expression to a powerful force. SCGDN effectively balances between preserving high-order structure information and avoiding overfitting. The results manifest that SCGDN can consistently generate outperformance over both the contrastive methods and the classical methods.

## CCS CONCEPTS

• **Theory of computation** → **Unsupervised learning and clustering**; **Graph algorithms analysis**; • **Computing methodologies** → **Learning latent representations**; **Neural networks**.
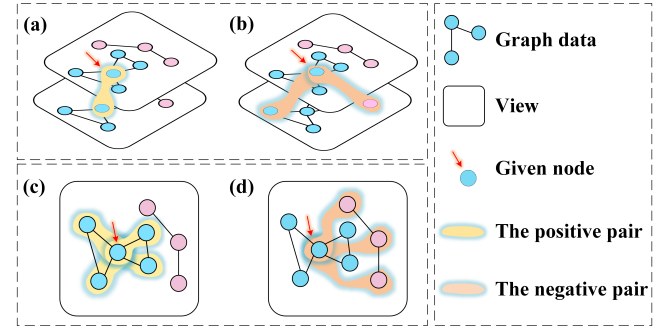
*Corresponding author.

## 1 INTRODUCTION



**Figure 1: A comparison between contrastive strategies for the prior works and ours. Subfigures (a) and (b) depict the common contrastive strategies, where a positive sample pair consists of the same node in two different views, while a negative sample pair is randomly selected. In contrast, subgraphs (c) and (d) demonstrate our proposed self-contrastive strategies, which rely on the intrinsic and valuable structure and feature information of the data for positive and negative samplings.**

Deep graph clustering is a fundamental yet hot topic in the graph field, which has attracted much attention for decades. The goal of deep graph clustering is to partition a given graph, where the edges between groups have very low weights and the edges within the group have high weights. The existing methods of deep graph clustering can be roughly divided into three categories: generative methods [8, 31, 32], adversarial methods [2, 26, 28, 36], and contrastive methods [9, 11, 12, 20, 33, 37]. Our proposed method belongs to the contrastive learning category.

Inspired by the success of contrastive learning in computer vision (CV) [14, 24], a growing number of works have been adapted to deep graph clustering [11, 12, 25]. Although contrastive graph clustering has achieved impressive performance, they require complex designs with a mass of parameters. For instance, MVGRL [12]

harness two dedicated Graph Neural Networks (GNNs) as graph encoders, a graph pooling layer as the readout function, and a discriminator as a parameterized mutual information estimator. Similarly, AFGRL [20] updates the online encoder parameters to the target encoder parameters using the Exponential Moving Average (EMA) and Stop-Gradient. Graph Convolutional Networks (GCNs) have shown remarkable performance on many network analysis tasks. However, most GCNs methods may not be applicable to real-world scenarios where the graph is dynamic. In contrast, Graph Diffusion Networks (GDNs) aim to capture the dynamic nature of the graph, by iteratively diffusing node features across the graph [4–6]. Hence, a natural question emerges: ***how to design a model framework with fewer parameters to contrastive graph clustering?*** Furthermore, the traditional graph contrastive learning paradigm mainly leverages corruptions and the momentum encoder to construct negative pairs. In fact, the process of constructing negative samples is random. While some studies [20, 34, 37] have recognized the importance of crucial clustering information in enhancing the discriminative capability, many works require pre-training models to obtain more accurate clustering information. This observation motivates us to reconsider that ***how to utilize intrinsic data information to develop an effective graph contrastive learning paradigm?***

In light of these above issues, we propose a novel approach called the Self-Contrastive Graph Diffusion Network (SCGDN), which is depicted in Fig. 2. Specifically, the architecture of SCGDN consists of two components. The first component, which we refer to as the Attentional Module (AttM), encodes feature and the high-order structure information of nodes into the latent space to guarantee learning excellent representation. There is an intuition that each node in the graph constantly changes its state until the final balance is achieved due to the influence of neighbors and distant points. That means we can utilize the excellent representation for diffusion learning. Therefore, the second component is called the Diffusion Module (DiFM). In DiFM, diffusion for $t$ time steps acts as a continuous analog of layers to aggregate information from $t$-hop neighbors. Inheriting the spirit of neural ODEs [7], the derivative of the hidden layer state parameterized by the neural network. This allows us to design a model framework with fewer parameters and develop more efficient diffusion process. The proposed SCGDN employs AttM to preserve the high-order structure information in the original feature space, and applies DiFM to capture the dynamic nature of the graph by diffusing node features in graphs.

To further improve the performance of contrastive graph clustering, we have inherited the advantages of COLES [38] and the success of redundancy reduction in latent space. In the unsupervised setting, contrastive learning requires generating two augment views of the same data samples, in which the same node itself is considered positive (as shown in Fig. 1 (a)). However, the common negative sampling strategy is to randomly sample another node, treating a given node and another node as negative pairs (as shown in Fig. 1 (b)). Under this strategy, there may be a link or feature similarity between the negative pairs, which is contradictory to the "negativeness". In fact, the structure and feature properties of graph data are valuable, *i.e.*, similar nodes may have link or feature similarity. Fig. 1 (c) and (d) show the motivation that the positive set for a given node is supposed to be a set of nodes that are associated

with the given node, and the negative set for a given node should be a set of nodes that are not related in structure or feature to the given node. Based on this observation, we further propose to leverage a block loss to construct a contrastive learning objective for learning more effective and abundant supervision information. The contrastive objective reasonably uses internal information to sample high-quality positive and negative samples, and the redundancy reduction term minimizes redundant information in the embedding and can well retain more discriminative information. Through their coordinated guidance, the potential spatial quality of subsequent clustering tasks is ensured. Hence, SCGDN cleverly avoids the asymmetric bi-encoders and the Siamese networks using a novel graph contrastive learning paradigm, addressing previous concerns.

The main contributions are summarized as follows:

- We propose a novel Self-Contrastive Graph Diffusion Network (SCGDN), which effectively balances between preserving high-order structure information and avoiding overfitting.
- We design an augmentation-free and free pre-training model framework, which avoids the "sampling bias" and the semantic drift while avoiding complex model designs, including two main parts, *i.e.*, AttM and DiFM.
- We introduce and theoretically analyze a novel graph contrastive learning paradigm that conducts contrastive learning with the proposed high-quality sampling strategies and without multiview.

To sum up, SCGDN offers an exceptional model framework and optimization paradigm that can achieve remarkable clustering performances. We conducted a comprehensive evaluation of SCGDN on six benchmark datasets for graph clustering, and the findings indicate that SCGDN outperforms both the contrastive and classical methods in terms of performance gains. Notably, the results show the effectiveness of the proposed approach and its potential for broader applications in the field of graph clustering.

## 2 RELATED WORK

The contrastive methods are one of the most powerful methods in self-supervised learning. The goal is to push similar nodes closer, while pulling different nodes farther. In a nutshell, we will describe the following three key issues.

**Model framework.** In graph representation learning, GCN [19] has become the almost de facto and widely adopted encoder. For example, MVGRL [12] uses un-shared GCNs and a shared MLP. DCRN [22] and GDCL [37] use the shared parameters Siamese networks. There are a vast number of works [15, 20] using other asymmetric bi-encoders, including EMA, momentum update, and Stop-Gradient. Nevertheless, a tremendous number of previous works either builg a parallel framework which has asymmetrical bi-encoders with plenty of parameters or enter two different views into a siamese network with shared parameters. In contrast, our proposed SCGDN only contains an end-to-end graph diffusion network. What's more important, SCGDN has an augmentation-free and free pre-training model framework, which avoids the "sampling bias" and the semantic drift.

**Sampling strategies.** The key detail of contrastive methods is how to characterize high-quality positive and negative samples.

The outstanding DGI [30] in graph contrastive learning, inspired by the prior success of DIM [14], treats each local representation and the summarized graph-level representation as a positive sample pair, while negative sample pairs are defined a shuffle representation and the summarized graph-level representation. Inspired by DGI [30], MVGRL [12] utilizes diffusion matrices and adjacency matrices as graph structure information, and also uses random shuffle to construct negative samples. GCC [27] uses the negative sampling strategy proposed by MOCO [13]. Later, much works [35, 39, 40] focus on how to construct different types of augmentation views, positive sample pairs, and negative sample pairs. Until the emergence of AFGRL [20] breaks the above situation, which required neither augmentation nor negative sampling. The positive samples of AFGRL [20] are determined by the adjacency matrix, the nearest neighbor obtained from learning, and the cluster information. COLES [38] randomly generates negative samples with a Gaussian distribution based on the random graph sampling theory [10]. Beyond these, we introduce a high-quality negative sampling strategy, which depends on the adjacency matrix and $k$NN graph.

**Objective function.** The theoretical core of contrastive learning is the InfoNCE principle [24], which maximizes the mutual information between different representations. With the introduction of DIM [14], MVGRL [12] which applies InfoMax loss, focuses on maximizing the mutual information between the local representation and the global representation. To be different, DCRN [22] considers the sample-level and feature-level of correlation reduction and designs the MSE loss to the identity matrix as well as the reconstruction loss and the clustering loss. In particular, AFGRL [20] minimizes the cosine distance between the positive pairs. The work [3] has uncovered tight relations between the cross-entropy loss and the contrastive loss, which inspires future studies in the unsupervised learning area. In addition, COLES [38] reformulates the Laplacian Eigenmaps [1] into contrastive learning. Furthermore, we propose an efficient graph contrastive learning paradigm with the representation level of correlation reduction.

## 3 METHODOLOGY

In this section, we propose a novel Self-Contrastive Graph Diffusion Network (SCGDN). The overall framework of SCGDN is shown in Fig. 2. Then, we introduce the proposed SCGDN in detail from the graph diffusion module and the self-contrastive learning objective.

### 3.1 Notations and Preliminaries

Given an undirected graph $\mathcal{G} = (\mathbf{V}, \mathbf{E}, \mathbf{X})$, where $\mathbf{V} = [v_1, v_2, \ldots, v_n] \in \mathbb{R}^n$ represents $n$ nodes, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$ is the corresponding feature matrix of the nodes, and $\mathbf{E}$ is a set of edges denoted by an adjacency matrix $\widetilde{\mathbf{W}} = [\widetilde{w}_{ij}] \in \mathbb{R}^{n \times n}$, where $\widetilde{w}_{ij} = 1$ if $(v_i, v_j) \in \mathbf{E}$ and $\widetilde{w}_{ij} = 0$ otherwise. $\mathbf{W} = \mathbf{D}^{-\frac{1}{2}}(\widetilde{\mathbf{W}} + \mathbf{I})\mathbf{D}^{-\frac{1}{2}} \in \mathbb{R}^{n \times n}$ is a symmetrically normalized adjacency matrix, $\mathbf{D} \in \mathbb{R}^{n \times n}$ is a diagonal matrix containing degrees of nodes, and $\mathbf{I} \in \mathbb{R}^{n \times n}$ represents the identity matrix.

Before encoding, we use the widely used $k$NN graph $\mathbf{W}^{knn}$ to build feature neighbor information aggregation, which encodes the similarities of each node feature. Unlike other work [21] that utilizes the Gaussian kernel, we use the t-distribution kernel. As

is known to all, the t-distribution is more gentle and more robust than the Gaussian distribution.

### 3.2 Graph Diffusion Module

*3.2.1 Attentional Module.* Inspired by the success of SDSNE [21], its intuition is a multiview system needs to share the intrinsical structure information by a shared self-attentional module. Inheriting the power of SDSNE [21], we jointly embed the structure and feature information of nodes into the latent space by designing a novel Attentional Module (AttN). The proposed AttN contains a self-attentional layer and a cross-attentional layer, respectively. Mathematically,

$$\mathbf{P}_s = \mathbf{W}\boldsymbol{\Theta}_1\mathbf{W}^\top, \tag{1}$$

$$\mathbf{P}_c = \mathbf{P}_s\boldsymbol{\Theta}_2\mathbf{W}^{knn}, \tag{2}$$

where $\mathbf{P}_s, \mathbf{P}_c \in \mathbb{R}^{n \times n}$ denote the attentional graphs, $\boldsymbol{\Theta}_1$ and $\boldsymbol{\Theta}_2$ are the trainable parameters. It is worth mentioned that the self-attentional layer explores higher-order structural information. Through the cross-attentional layer, feature information is better aggregated.

Subsequently, we calculate the similarity matrix $\mathbf{S}$ and normalize the similarity matrix $\mathbf{S}$ with $\ell^2$-norm as formulated:

$$\mathbf{S} = \mathbf{P_c}\mathbf{P_c}^\top, \quad \mathbf{S} = [\mathbf{s}_i] \in \mathbb{R}^{n \times n}, \quad \mathbf{s}_i = \frac{\mathbf{s}_i}{\|\mathbf{s}_i\|_2}, \forall i, \tag{3}$$

where $s_i$ denotes a column in $\mathbf{S}$.

Then, we encode the similarity matrix $\mathbf{S}$ with two separated linear layers called MLP as follows:

$$\mathbf{H} = \mathrm{MLP}_{\boldsymbol{\Phi}_1, \boldsymbol{\Phi}_2}(\mathbf{S}) \in \mathbb{R}^{n \times d'}, \tag{4}$$

where $d'$ is the number of hidden dimensions, $\boldsymbol{\Phi}_1$ and $\boldsymbol{\Phi}_2$ are the trainable parameters of linear layers, respectively.

Through AttM, the higher-order structure and feature information of graph data are better aggregated, thus improving the expression ability of graph representation and further improving the performance of the downstream tasks.

*3.2.2 Diffusion Module.* Recently, some works [4–6] have solved the common difficulties of graph learning model, such as depth, over-smoothing, etc. Motivated by their success, we introduce Diffusion Module (DiFM) to learning node embedding.

DEFINITION 1 (GRAPH DIFFUSION). *A graph space consists of feature and structure information $\mathbf{Z} = (\mathbf{X}, \mathbf{W})$. For a node, graph diffusion with time-dependent $t$ can be achieved as follows:*

$$\frac{\partial \widetilde{z}_i(t)}{\partial t} = \mathrm{div}\left(\mathbf{a}_i\left(\mathbf{z}_i(t)\right)\nabla\widetilde{z}_i(t)\right),$$
$$\widetilde{z}_i(0) = \mathbf{h}_i; \quad i = 1, \ldots, n; \quad t \geq 0, \tag{5}$$

*where the function $\mathbf{a}_i(\cdot)$ is the diffusivity controlling the diffusion strength between node $i$ and its neighbors.*

DEFINITION 2 (THE STATIONARY STATE OF GRAPH DIFFUSION). *In order to produce a stationary state of graph diffusion, DiFM should be able to learn the overall information of the graph. By Eq. (5), the stationary state $\widetilde{\mathbf{Z}}$ for time $t$ can be written as:*

$$\widetilde{\mathbf{Z}}(t) = \widetilde{\mathbf{Z}}(0) + \int_0^t \frac{\partial\widetilde{\mathbf{Z}}(\tau)}{\partial\tau}\mathrm{d}\tau, \tag{6}$$
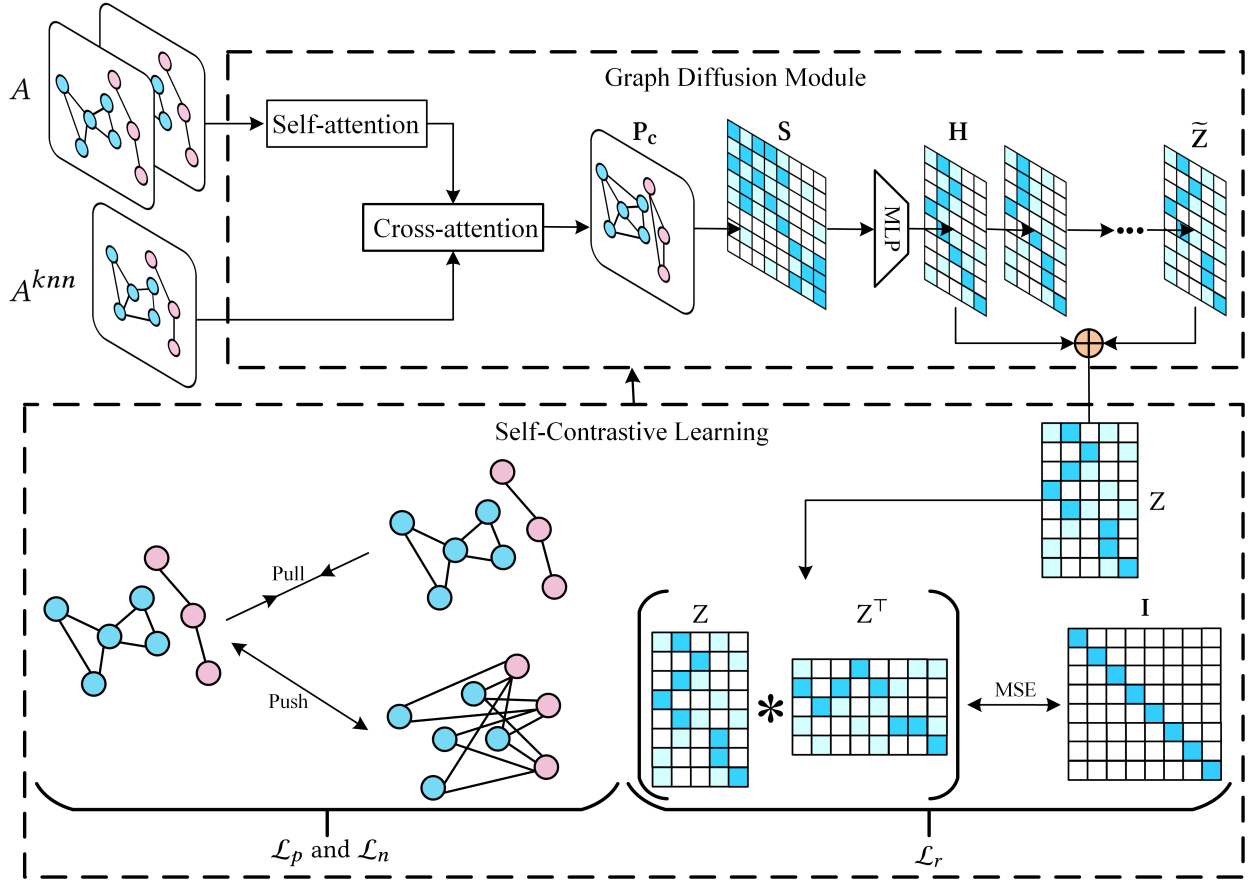
*where $\tau$ is the time step.*

**Figure 2: The illustration of the proposed Self-Contrastive Graph Diffusion Network (SCGDN). Given an undirected attribute graph, we first calculate the representation of samples by the graph diffusion module. Then, the Self-Contrastive Learning objective guides the update of the parameters of the graph diffusion module.**

Next, to obtain the integral term, we must compute $\frac{\partial \widetilde{Z}(t)}{\partial t}$. Here, we borrow from the Neural ODEs [7], that is, the derivative of the hidden layer state parameterized by the neural network $f$,

$$\frac{\partial \widetilde{Z}(t)}{\partial t} = f\left(\widetilde{Z}(t), W, t, \Psi\right) = (W - I)\widetilde{Z}(t) \qquad (7)$$

where $W = [w_{ij}], \forall ij, w_{ij} = w_{ji} \geq 0, \sum_i w_{ij} = 1$, and $\Psi$ is the trainable parameters of the neural network $f$.

Graph diffusion is interpreted as a nonlinear filter depending on feature information and adjacency relations. In the limit $t \to \infty$, the graph becomes stable and each connected component is equal to its average feature [21]. This emphasizes the observation that adjacent nodes have closer relationships than non-adjacent nodes, and nodes can propagate messages to neighbors, making adjacent nodes closer in terms of feature.

In addition, we further sum the input embedding $H$ and the stationary state $\widetilde{Z}$ with the normalization factor $\omega$ as formulated:

$$Z = \sigma\left(\omega\widetilde{Z} + H\right), \qquad (8)$$

where $\sigma(\cdot)$ is an activation function, e.g., ReLU$(\cdot)$ = max$(0; \cdot)$, and the normalization factor $\omega = \sqrt{2D}$ utilizes the degree information of

each node. Following, we normalize $Z$ with $z$-score normalization. Finally, we perform K-means on the optimal embedding $Z$ obtained.

DiFM has the following advantages. First, by injecting graph relations into feature information, it generates more useful node representations for downstream tasks. Second, it allows the cooperative evolution of adjacency and feature information in a graph.

### 3.3 Self-Contrastive Learning

Driven by the classic InfoNCE loss, plenty of works [20, 33] have achieved excellent clustering performance. Yet, the classic InfoNCE loss is bounded by JS divergence, which yields zero and vanishing gradients. Consider the block-contrastive loss COLES [38], which realizes the negative sampling strategy for Laplacian Eigenmaps, is driven by reformulating SampledNCE into Wasserstein GAN using a GAN-inspired contrastive formulation.

$$\mathcal{L} = \text{Tr}\left(Z^\top L^{(+)} Z\right) - \frac{\eta'}{\kappa} \sum_{k=1}^{\kappa} \text{Tr}\left(Z^\top L_k^{(-)} Z\right), \qquad (9)$$

where $L^{(+)}$ is degree-normalized Laplacian matrix capturing the positive sampling, $L_k^{(-)}$ for $k = 1, ..., \kappa$ are randomly generated

degree-normalized Laplacian matrices capturing the negative sampling, and $\eta'$ is a scalar to controlling the effect of negative samples.

However, we find the drawback of COLES [38] is that the negative sample set is randomly generated, which may mislead the model into learning wrong parameters due to the inaccurate negative samples. To solve this problem, we propose a novel negative sampling strategy to get an adjacency matrix of negative samples.

*3.3.1 Negative Sampling.* Many methods are used to shuffle the index [12, 30] or randomly generate negative samples [38]. However, this would violate the negativity of negative samples. In order to generate high-quality negative samples, we consider both structure and feature information. Mathematically, we define an adjacency matrix of negative samples as follows:

$$\widetilde{w}_{ij}^{(-)} = \begin{cases} 1, & \text{if } w_{ij} \cup w_{ij}^{\text{knn}} = 0 \\ 0, & \text{otherwise} . \end{cases} \tag{10}$$

For the convenience of narrative, we denote $\widetilde{\mathbf{W}}^{(+)} = \widetilde{\mathbf{W}}$ and $\widetilde{\mathbf{D}}^{(+)} = \mathbf{D}$ as the adjacency matrix and the diagonal matrix of positive samples, respectively.

*3.3.2 Objective.* Based on the negative sampling, we formulate the objective function as follows:

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_p - \beta \mathcal{L}_n + \gamma \mathcal{L}_r \\ &= \text{Tr}\left(\mathbf{Z}^\top \mathbf{L}^{(+)} \mathbf{Z}\right) - \beta \text{Tr}\left(\mathbf{Z}^\top \mathbf{L}^{(-)} \mathbf{Z}\right) + \gamma \|\mathbf{Z}\mathbf{Z}^\top - \mathbf{I}\|_\text{F}^2, \end{aligned} \tag{11}$$

where $\beta$ is a non-negative hyperparameter trading off the two contrastive terms, $\mathbf{L}^{(+)} = \mathbf{I} - (\widetilde{\mathbf{D}}^{(+)})^{-\frac{1}{2}} \widetilde{\mathbf{W}}^{(+)} (\widetilde{\mathbf{D}}^{(+)})^{-\frac{1}{2}}$ and $\mathbf{L}^{(-)} = \mathbf{I} - (\widetilde{\mathbf{D}}^{(-)})^{-\frac{1}{2}} \widetilde{\mathbf{W}}^{(-)} (\widetilde{\mathbf{D}}^{(-)})^{-\frac{1}{2}}$ are the Laplacian matrix of positive samples and negative samples, respectively. In addition, $\gamma$ is also a non-negative hyperparameter, and it controls the last term which encourages incoherence between the column vectors of network output. The detailed learning process of SCGDN is shown in Algorithm 1.

## 4 EXPERIMENTS

We conduct various experiments to evaluate the effectiveness and efficiency of the proposed SCGDN method on the node clustering task. The focuses of the experiments are to validate the representation ability of the learned feature, the effectiveness of the model framework, and the necessity of each component of the objective function.

**Datasets.** We compare our SCGDN approach to different baselines on six benchmark datasets, including Cora [18], Citeseer [18], Brazil Air-Traffic (BAT) [23], Europe Air-Traffic (EAT) [23], CoraFull [22] and Amazon Photo (AMAP) [22]. The statistics of these datasets are summarized in Table 1, and the descriptions are as follows.

- Cora [18], Citeseer [18] and CoraFull [22] are well-known citation network datasets. Nodes represent papers, and edges indicate the citation relationship. The bag-of-words representation of papers are regarded as node features, and labels are academic fields.
- BAT [23] and EAT [23] are two air-traffic datasets (Brazil and Europe). Nodes correspond to airports, and edges indicate

---

**Algorithm 1** The SCGDN algorithm.

**Input:** feature matrix $\mathbf{X}$; adjacency matrix $\widetilde{\mathbf{W}}$; the number of neighbors $k$; and parameters $\beta, \gamma$.
**Output:** the clustering results $R$.
1: **Initialization**: $epoch = 1$, $epoch_{\max}$, and the model parameters.
2: Build $k$NN graphs $\mathbf{W}^{knn}$ for the feature matrix $\mathbf{X}$ with the t-distribution kernel;
3: **while** $epoch \leqslant epoch_{\max}$ **do**
4:     Obtain the attentional graphs $\mathbf{P}_s$ and $\mathbf{P}_c$ by Eq. (1) and Eq. (2);
5:     Calculate the similarity matrix $\mathbf{S}$ and normalize the similarity matrix $\mathbf{S}$ by Eq. (3);
6:     Encode the similarity matrix $\mathbf{S}$ with MLP encoder by Eq. (4);
7:     Obtain the stationary state $\widetilde{\mathbf{Z}}$ using the Diffusion Module (DiFM) with Eq. (6);
8:     Calculate the node representation $\mathbf{Z}$ by Eq. (8) and normalize $\mathbf{Z}$ with z-score normalization;
9:     Update parameters by minimizing $\mathcal{L}$ in Eq. (11);
10:     $epoch = epoch + 1$;
11: **end while**
12: Perform K-means on $\mathbf{Z}$ to obtain the final clustering results $R$.
13: **return** $R$.

---

the existence of commercial flights. Node features are constructed by leveraging the one-hot encoding of node degrees. Labels are corresponding to the airport's level of activity, measured in flights or people.
- AMAP [22] is based on Amazon's co-purchase data. Nodes denote products, while edges reflect the two products are purchased at the same time. There are the sparse bag-of-words attribute vector encoding product reviews as node features, and labels are product categories.

**Table 1: Statistics summary of the graph datasets.**

| Dataset | Nodes | Edges | Features | Clusters |
|---|---|---|---|---|
| Cora | 2,708 | 5,429 | 1,433 | 7 |
| Citeseer | 3,327 | 4,732 | 3,703 | 6 |
| AMAP | 7,650 | 119,081 | 745 | 8 |
| BAT | 131 | 1,038 | 81 | 4 |
| EAT | 399 | 5,994 | 203 | 4 |
| CoraFull | 19,793 | 63,421 | 8,710 | 70 |

**Baseline models.** To compare SCGDN with previous works, we choose three types of deep clustering methods as baselines, including generative methods (GAE [18], MGAE [32], DAEGC [31]), adversarial methods (DFCN [28], SDCN [2]), and contrastive methods (MVGRL [12], DCRN [22], GDCL [37], AutoSSL [16], AGC-DRR [11], AFGRL [20], ProGCL [33]). For the results of all data from the baselines, we will quote the results directly or quote the results of the baseline being replicated.

**Experimental setting.** Our proposed SCGDN is trained using Adam [17], in which the learning rate is set to 1e-3 for AttM, and 1e-5 for DiFM, respectively. We first train the model in an unsupervised

**Table 2: Clustering performance on graph datasets. The best values are in bold.**

| Method | MGAE | DAEGC | DFCN | MVGRL | GDCL | AutoSSL | AGC-DRR | AFGRL | ProGCL | SCGDN |
|---|---|---|---|---|---|---|---|---|---|---|
| **Cora** | | | | | | | | | | |
| ACC% | 43.38±2.11 | 70.43±0.36 | 36.33±0.49 | 70.47±3.70 | 70.83±0.47 | 63.81±0.57 | 40.62±0.55 | 26.25±1.24 | 57.13±1.23 | **74.79±0.38** |
| NMI% | 28.78±2.97 | 52.89±0.69 | 19.36±0.87 | 55.57±1.54 | 56.30±0.36 | 47.62±0.45 | 18.74±0.73 | 12.36±1.54 | 41.02±1.34 | **56.86±0.42** |
| ARI% | 16.43±1.65 | 49.63±0.43 | 4.67±2.10 | 48.70±3.94 | 48.05±0.72 | 38.92±0.77 | 14.80±1.64 | 14.32±1.87 | 30.71±2.70 | **52.61±0.33** |
| F1% | 33.48±3.05 | 68.27±0.57 | 26.16±0.50 | 67.15±1.86 | 52.88±0.97 | 56.42±0.21 | 31.23±0.57 | 30.20±1.15 | 45.68±1.29 | **70.42±0.48** |
| **Citeseer** | | | | | | | | | | |
| ACC% | 61.35±0.80 | 64.54±1.39 | 69.50±0.20 | 62.83±1.59 | 66.39±0.65 | 66.76±0.67 | 68.32±1.83 | 31.45±0.54 | 65.92±0.80 | **69.62±0.03** |
| NMI% | 34.63±0.65 | 36.41±0.86 | 43.90±0.20 | 40.69±0.93 | 39.52±0.38 | 40.67±0.84 | 43.28±1.41 | 15.17±0.47 | 39.59±0.39 | **44.35±0.03** |
| ARI% | 33.55±1.18 | 37.78±1.24 | 45.50±0.30 | 34.18±1.73 | 41.07±0.96 | 38.73±0.55 | 45.34±2.33 | 14.32±0.78 | 36.16±1.11 | **45.43±0.04** |
| F1% | 57.36±0.82 | 62.20±1.32 | 64.30±0.20 | 59.54±2.17 | 61.12±0.70 | 58.22±0.68 | 64.82±1.60 | 30.20±0.71 | 57.89±1.98 | **65.50±0.06** |
| **AMAP** | | | | | | | | | | |
| ACC% | 71.57±2.48 | 75.96±0.23 | 76.82±0.23 | 41.07±3.12 | 43.75±0.78 | 54.55±0.97 | 76.81±1.45 | 75.51±0.77 | 51.53±0.38 | **78.91±0.15** |
| NMI% | 62.13±2.79 | 65.25±0.45 | 66.23±1.21 | 30.28±3.94 | 37.32±0.28 | 48.56±0.71 | 66.54±1.24 | 64.05±0.15 | 39.56±0.39 | **72.53±0.25** |
| ARI% | 48.82±4.57 | 58.12±0.24 | 58.28±0.74 | 18.77±2.34 | 21.57±0.51 | 26.87±0.34 | 60.15±1.56 | 54.45±0.48 | 34.18±0.89 | **63.41±0.21** |
| F1% | 68.08±1.76 | 69.87±0.54 | 71.25±0.31 | 32.88±5.50 | 38.37±0.29 | 54.47±0.83 | 71.03±0.64 | 69.99±0.34 | 31.97±0.44 | **75.27±0.18** |
| **BAT** | | | | | | | | | | |
| ACC% | 53.59±2.04 | 52.67±0.00 | 55.73±0.06 | 37.56±0.32 | 45.42±0.54 | 42.43±0.47 | 47.79±0.02 | 50.92±0.44 | 55.73±0.79 | **74.73±0.23** |
| NMI% | 30.59±2.06 | 21.43±0.35 | 48.77±0.51 | 29.33±0.70 | 31.70±0.42 | 17.84±0.98 | 19.91±0.24 | 27.55±0.62 | 28.69±0.92 | **52.63±0.11** |
| ARI% | 24.15±1.70 | 18.18±0.29 | 37.76±0.23 | 13.45±0.03 | 19.33±0.57 | 13.11±0.81 | 14.59±0.13 | 21.89±0.74 | 21.84±1.34 | **47.65±0.18** |
| F1% | 50.83±3.23 | 52.23±0.03 | 50.90±0.12 | 29.64±0.49 | 39.94±0.57 | 34.84±0.15 | 42.33±0.51 | 46.53±0.57 | 56.08±0.89 | **74.49±0.26** |
| **EAT** | | | | | | | | | | |
| ACC% | 44.61±2.10 | 36.89±0.15 | 49.37±0.19 | 32.88±0.71 | 33.46±0.18 | 31.33±0.52 | 37.37±0.11 | 37.42±1.24 | 43.36±0.87 | **56.52±0.13** |
| NMI% | 15.60±2.30 | 5.57±0.06 | 32.90±0.41 | 11.72±1.08 | 13.22±0.33 | 7.63±0.85 | 7.00±0.85 | 11.44±1.41 | 23.93±0.45 | **32.99±0.16** |
| ARI% | 13.40±1.26 | 5.03±0.08 | **23.25±0.18** | 4.68±1.30 | 4.31±0.29 | 2.13±0.67 | 4.88±0.91 | 6.57±1.73 | 15.03±0.98 | 22.89±0.10 |
| F1% | 43.08±3.26 | 34.72±0.16 | 42.95±0.04 | 25.35±0.75 | 25.02±0.21 | 21.82±0.98 | 35.20±0.17 | 30.53±1.47 | 42.54±0.45 | **57.63±0.10** |

**Table 3: Clustering performance on CoraFull datasets. The best values are in bold.**

| Method | ACC% | NMI% | ARI% | F1% |
|---|---|---|---|---|
| GAE | 29.60±0.81 | 45.82±0.75 | 17.84±0.86 | 25.95±0.75 |
| DAEGC | 34.35±1.00 | 49.16±0.73 | 22.60±0.47 | 26.96±1.33 |
| SDCN | 26.67±0.40 | 37.38±0.39 | 13.63±0.27 | 22.14±0.43 |
| DFCN | 37.51±0.81 | 51.30±0.41 | 24.46±0.48 | 31.22±0.87 |
| MVGRL | 31.52±2.95 | 48.99±3.95 | 19.11±2.63 | 26.51±2.87 |
| DCRN | 38.80±0.60 | 51.91±0.35 | 25.25±0.49 | 31.68±0.76 |
| SCGDN | **40.13±0.41** | **54.15±0.08** | **26.97±0.87** | **34.77±0.26** |

manner, then perform evaluations on the learned representations. For all experiments, we report the mean accuracy with a standard deviation through 10 random initializations. About building the t-distribution $k$NN graph, we fix the standard deviation $\sigma = 0.5$ and the degrees of freedom $v = 1$, and set the dimension of hidden layer in each dataset to 512, except for BAT dataset. Because the feature dimension of BAT dataset is 81, less than 512. Detailed hyperparameter setting is in Appendix A.1.

We implement SCGDN with PyTorch and all experiments are conducted on NVIDIA RTX 3090 GPUs. We use widely used four metrics to evaluate clustering performance, including Accuracy (ACC), Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), and F1-score (F1).

## 4.1 Node Clustering

In this section, we compare SCGDN with three types of deep clustering methods and report the node clustering results in Tables 2 and 3.

Compared to adversarial and generative methods, it can be viewed that SCGDN outperforms the baselines. This is because SCGDN which inherits the advantage of the contrastive methods has more available supervision information. Whether in large or small datasets, we have empirically verified the superior performance of SCGDN compared with other contrastive methods. This is because SCGDN captures intrinsic category information due to high-quality sampling. On AMAP and CoraFull datasets which are the larger benchmark, SCGDN also achieves the best performance. It is worth mentioned that we empirically find that representation of 64 dimensions is better than baselines on BAT dataset.

## 4.2 Ablation studies

In this section, we first conduct ablation studies to verify the effectiveness of each component in our proposed loss function on five benchmark datasets as shown in Table 4. In addition, we conduct another ablation studies to verify the effectiveness of the proposed AttM and DiFM on Cora and Citeseer datasets as shown in Table 5. Last, we conduct ablation studies to verify the effectiveness of the negative sampling strategy.

**Effectiveness of $\mathcal{L}_p$, $\mathcal{L}_n$ and $\mathcal{L}_r$.** To see the impact of the positive and negative contrastive term and the redundancy reduction term, we conduct the ablation studies which have different types of objective functions. The first variant, "$\mathcal{L}_r$", only uses the redundancy reduction term. The second variant, which we refer to as "(w/o)$\mathcal{L}_n$", uses the positive contrastive term and the redundancy reduction term to guide the network parameter updates. The "(w/o)$\mathcal{L}_r$" uses the positive contrastive term and the negative contrastive term. From these results, we have three findings as follows. (1) The positive and negative contrastive terms provide more supervision information implicitly. (2) The redundancy reduction term helps reduce feature redundancy in potential spaces to obtain a more discriminative representation. (3) The negative contrastive term further boosts the performance of clustering by pushing the distance of the negative samples. Overall, these experiments are sufficient to illustrate the necessity of each component in our proposed loss function.

**Table 4: Ablation studies of functions on graph datasets. The best values are in bold.**

| Dataset | $\mathcal{L}_r$ | (w/o)$\mathcal{L}_n$ | (w/o)$\mathcal{L}_r$ | SCGDN |
|---|---|---|---|---|
| Cora | | | | |
| ACC% | 47.47±2.18 | 74.04±0.09 | 74.23±0.04 | **74.79±0.38** |
| NMI% | 31.73±2.09 | 55.75±0.13 | 56.30±0.12 | **56.86±0.42** |
| ARI% | 14.28±2.28 | 51.56 ± 0.16 | 52.08±0.13 | **52.61±0.33** |
| F1% | 43.23±3.53 | 69.63 ± 0.07 | 69.75±0.06 | **70.42±0.48** |
| Citeseer | | | | |
| ACC% | 64.14±0.08 | 69.47±0.03 | 69.55±0.02 | **69.62±0.03** |
| NMI% | 39.51±0.05 | 44.03±0.08 | 44.09±0.01 | **44.35±0.03** |
| ARI% | 37.49±0.09 | 45.15±0.06 | 45.30±0.03 | **45.43±0.04** |
| F1% | 60.50±0.03 | 65.01±0.03 | 65.49±0.04 | **65.50±0.06** |
| AMAP | | | | |
| ACC% | 61.57±1.36 | 78.85±0.07 | 78.03±0.01 | **78.91±0.15** |
| NMI% | 48.79±1.69 | 72.34±0.19 | 70.96±0.02 | **72.53±0.25** |
| ARI% | 33.94±2.35 | 63.28±0.11 | 62.26±0.06 | **63.41±0.21** |
| F1% | 50.35±3.50 | 75.21±0.26 | 74.15±0.03 | **75.27±0.18** |
| BAT | | | | |
| ACC% | 54.50±0.37 | 65.88±5.91 | 74.27±0.35 | **74.73±0.23** |
| NMI% | 42.77±1.10 | 46.49±3.47 | 52.40±0.17 | **52.63±0.11** |
| ARI% | 29.49±0.56 | 38.71±5.21 | 47.28±0.28 | **47.65±0.18** |
| F1% | 47.19±0.31 | 64.65±6.81 | 73.96±0.40 | **74.49±0.26** |
| EAT | | | | |
| ACC% | 52.23±0.23 | 54.34±0.31 | 54.64±0.19 | **56.52±0.13** |
| NMI% | 33.02±0.10 | 34.66±0.19 | **34.71±0.32** | 32.99±0.16 |
| ARI% | 22.96±0.17 | 23.58±0.21 | **23.73±0.19** | 22.89±0.10 |
| F1% | 53.05±0.21 | 54.96±0.28 | 55.20±0.18 | **57.63±0.10** |

**Effectiveness of AttM and DiFM.** Here, we denote "Ours$_{GCN}$", "Ours$_{AttM+GCN}$", and "Ours", as the model of GCN, the model of AttM and GCN, and SCGDN, respectively. From the results of Table 5 and Table 6, we have three observations as follows. (1) Using our proposed graph contrastive paradigm, good results can also be achieved under the framework of GCN. (2) Our proposed AttM better aggregates structure and feature information. (3) With an understanding of the nature of graph problems, our proposed

DiFM does provide a better diffusion of learned representations. To sum up, these experiments validate the effectiveness of the model framework.

**Table 5: Ablation studies of models on Cora and Citeseer datasets. The best values are in bold.**

| Method | Cora | | | Citeseer | | |
|---|---|---|---|---|---|---|
| | ACC% | NMI% | F1% | ACC% | NMI% | F1% |
| Ours$_{GCN}$ | 67.47 | 49.58 | 63.65 | 64.50 | 37.12 | 59.30 |
| Ours$_{AttM+GCN}$ | 73.01 | 53.85 | 63.91 | 69.37 | 44.26 | 60.68 |
| Ours | **74.79** | **56.86** | **70.42** | **69.62** | **44.35** | **65.50** |

**Effectiveness of the negative sampling strategy.** In these ablation studies, "DGI" denotes the model of GCN with the classic contrastive method. And "Ours$_{random}$" denotes randomly generating negative samples in SCGDN. From the results of Table 6, we have the observation as follows: the negative sampling strategy could improve the performance of SCGDN, and its performance exceeds that of DGI and "Ours$_{random}$". Overall, these experiments are sufficient to illustrate the effectiveness of the negative sampling strategy.

**Table 6: Ablation studies of models on Cora and Citeseer datasets. The best values are in bold.**

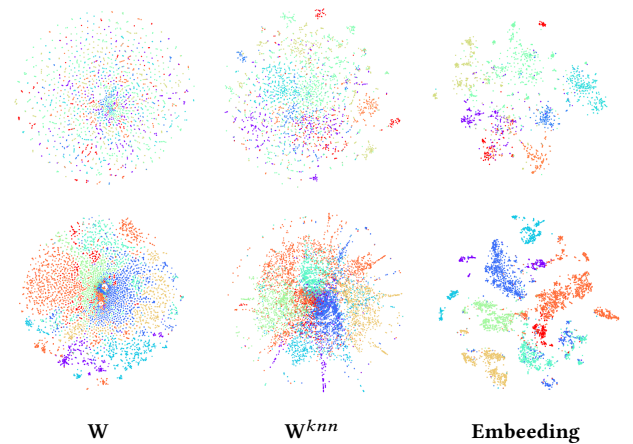| Method | Cora | | | Citeseer | | |
|---|---|---|---|---|---|---|
| | ACC% | NMI% | F1% | ACC% | NMI% | F1% |
| DGI | 71.81 | 54.09 | 69.88 | 68.60 | 43.75 | 64.64 |
| Ours$_{random}$ | 73.45 | **56.96** | 69.76 | 69.00 | 43.40 | 64.18 |
| Ours | **74.79** | 56.86 | **70.42** | **69.62** | **44.35** | **65.50** |



| **W** | **W$^{knn}$** | **Embeeding** |

**Figure 3: 2D t-SNE visualization on two benchmark datasets. The first row and second row correspond to Cora and AMAP datasets, respectively.**

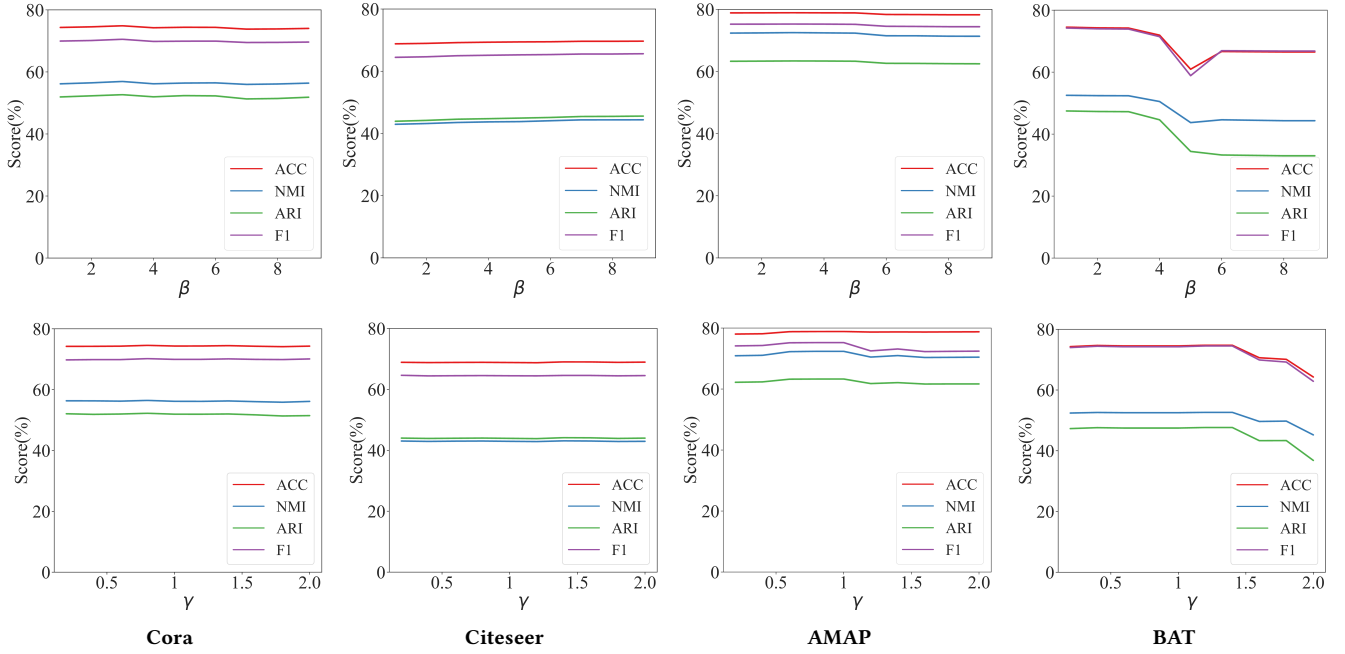**Figure 4: Parameter sensitivity of $\beta$ and $\gamma$ on four benchmark datasets. The first row and second row correspond to sensitivity of $\beta$ and $\gamma$, respectively.**

## 4.3 Analysis

*4.3.1 Visualization analysis.* In this part, we visualize the distribution of the learned representations to provide a more intuitive understanding of the learned node representations of SCGDN on Cora and AMAP datasets via the t-SNE algorithm [29]. The t-SNE focuses on data points that are relatively close together in high-dimensional space. Therefore, the embedding local structure can be observed more clearly and intuitively. The color represents the node label, and each point represents a node. From Fig 3, we have the following observations: (1) The raw data of Cora dataset lacks obvious clustering, and it appears chaotic. However, the learned node representations by SCGDN show more tightly grouped nodes. This signifies that SCGDN captures more fine-grained class information. (2) Although there is the class information in the raw data of the AMAP dataset, different classes are not clearly separated. From the learned node representations of SCGDN, we observe that there is a significant gap between different categories, and the same category is more closely spaced. This indicates that the optimization objective has narrowed the distance between the same class and widened the distance between different classes.

*4.3.2 Convergence analysis.* To further evaluate the performance of the proposed loss function, we conducted experiments to analyze the convergence of the loss. Specifically, we plotted the trend of the loss and the ACC metric on AMAP dataset, as shown in Fig. 5. As the loss decreases, the accuracy rate gradually increases and tends to be stable.

*4.3.3 Parameter sensitivity analysis.* For unsupervised contrastive learning methods, parameter insensitivity is vital for enhancing
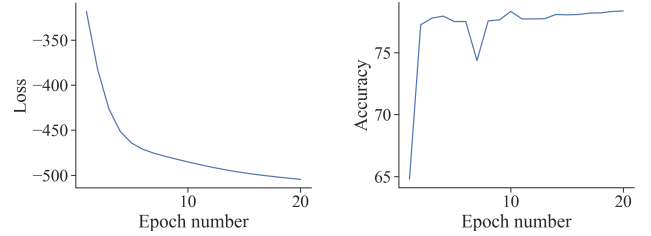


**Figure 5: Convergence analysis of the proposed loss on AMAP dataset.**

their stability. For $\beta$, we set it in {0.5, 1, 2, 3, 4, 5, 6, 7, 8, 9}, and for $\gamma$, we set it in {0.2, 0.4, 0.6, 0.8, 1, 1.2, 1.4, 1.6 1.8, 2}. From Fig. 4, conclusion is deduced that SCGDN is not sensitive to the factors $\beta$ and $\gamma$, which enhances the practicality in real-world applications.

## 5 CONCLUSION

In this work, we propose a novel Self-Contrastive Graph Diffusion Network (SCGDN), an augmentation-free and free pre-training method. SCGDN effectively balances between preserving high-order structure information and avoiding overfitting. First, we design an efficient and effective model framework, including two main parts, *i.e.*, Attentional Module (AttM) and Diffusion Module (DiFM). Specifically, AttM aggregates higher-order structure and feature information to get a excellent embedding. DiFM balances the stationary state of each node in the graph through diffusion

learning. Meanwhile, we introduced a novel graph contrastive learning paradigm that conducts contrastive learning with the proposed high-quality negative sampling strategy and without multiview. Compared with other contrastive methods, SCGDN not only further improves the discriminative capability of the learned representations, but also utilizes intrinsic feature information and higher-order structure. In light of extensive experiments on six benchmark datasets, the results indicated the effectiveness and superiority of the proposed SCGDN. In the future, we plan to apply the proposed method to solve more real applications.

# REFERENCES

[1] Mikhail Belkin and Partha Niyogi. 2003. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation* 15, 6 (2003), 1373–1396.

[2] Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. 2020. Structural Deep Clustering Network. In *WWW*. 1400–1410.

[3] Malik Boudiaf, Jérôme Rony, Imtiaz Masud Ziko, Eric Granger, Marco Pedersoli, Pablo Piantanida, and Ismail Ben Ayed. 2020. A unifying mutual information view of metric learning: cross-entropy vs. pairwise losses. In *ECCV*. 548–564.

[4] Benjamin Paul Chamberlain, James Rowbottom, Davide Eynard, Francesco Di Giovanni, Dong Xiaowen, and Michael M Bronstein. 2021. Beltrami Flow and Neural Diffusion on Graphs. In *NeurIPS*, Vol. 34. 1594–1609.

[5] Benjamin Paul Chamberlain, James Rowbottom, Maria Goronova, Stefan Webb, Emanuele Rossi, and Michael M Bronstein. 2021. GRAND: Graph Neural Diffusion. In *ICML*, Vol. 139. 1407–1418.

[6] Qi Chen, Yifei Wang, Yisen Wang, Jiansheng Yang, and Zhouchen Lin. 2022. Optimization-Induced Graph Implicit Nonlinear Diffusion. In *ICML*, Vol. 162. 3648–3661.

[7] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. 2018. Neural ordinary differential equations. In *NeurIPS*, Vol. 31.

[8] Jiafeng Cheng, Qianqian Wang, Zhiqiang Tao, and Quanxue Gao. 2021. Multi-View Attribute Graph Convolution Networks for Clustering. In *IJCAI*. 2973–2979.

[9] Ganqu Cui, Jie Zhou, Cheng Yang, and Zhiyuan Liu. 2020. Adaptive graph encoder for attributed graph embedding. In *KDD*. 976–985.

[10] Paul ErdHos and Alfréd Rényi. 1959. On Random Graphs. *PM* 6 (1959), 290–297.

[11] Lei Gong, Sihang Zhou, Xinwang Liu, and Wenxuan Tu. 2022. Attributed graph clustering with dual redundancy reduction. In *IJCAI*.

[12] Kaveh Hassani and Amir Hosein Khasahmadi. 2020. Contrastive multi-view representation learning on graphs. In *ICML*. 4116–4126.

[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *CVPR*. 9729–9738.

[14] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *ICLR*.

[15] Ming Jin, Yizhen Zheng, Yuan-Fang Li, Chen Gong, Chuan Zhou, and Shirui Pan. 2021. Multi-Scale Contrastive Siamese Networks for Self-Supervised Graph Representation Learning. In *IJCAI*. 1477–1483.

[16] Wei Jin, Xiaorui Liu, Xiangyu Zhao, Yao Ma, Neil Shah, and Jiliang Tang. 2022. Automated self-supervised learning for graphs. In *ICLR*.

[17] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

[18] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. In *NeurIPS*.

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

[20] Namkyeong Lee, Junseok Lee, and Chanyoung Park. 2022. Augmentation-Free Self-Supervised Learning on Graphs. In *AAAI*, Vol. 36. 7372–7380.

[21] Chenghua Liu, Zhuolin Liao, Yixuan Ma, and Kun Zhan. 2022. Stationary diffusion state neural estimation for multiview clustering. In *AAAI*, Vol. 36. 7542–7549.

[22] Yue Liu, Wenxuan Tu, Sihang Zhou, Xinwang Liu, Linxuan Song, Xihong Yang, and En Zhu. 2022. Deep graph clustering via dual correlation reduction. In *AAAI*, Vol. 36. 7603–7611.

[23] Nairouz Mrabah, Mohamed Bouguessa, Mohamed Fawzi Touati, and Riadh Ksantini. 2022. Rethinking Graph Auto-Encoder Models for Attributed Graph Clustering. *TKDE* (2022), 1–15.

[24] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).

[25] Erlin Pan and Zhao Kang. 2021. Multi-view contrastive graph clustering. In *NeurIPS*, Vol. 34. 2148–2159.

[26] Shirui Pan, Ruiqi Hu, Sai-fu Fung, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Learning graph embedding with adversarial training methods. *TCYB* 50, 6 (2019), 2475–2487.

[27] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *KDD*. 1150–1160.

[28] Wenxuan Tu, Sihang Zhou, Xinwang Liu, Xifeng Guo, Zhiping Cai, En Zhu, and Jieren Cheng. 2021. Deep fusion clustering network. In *AAAI*, Vol. 35. 9978–9987.

[29] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 9, 11 (2008).

[30] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. In *ICLR*, Vol. 2. 4.

[31] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Attributed graph clustering: A deep attentional embedding approach. In *IJCAI*. 3670–3676.

[32] Chun Wang, Shirui Pan, Guodong Long, Xingquan Zhu, and Jing Jiang. 2017. Mgae: Marginalized graph autoencoder for graph clustering. In *CIKM*. 889–898.

[33] Jun Xia, Lirong Wu, Ge Wang, Jintao Chen, and Stan Z Li. 2022. Progcl: Rethinking hard negative mining in graph contrastive learning. In *ICML*. 24332–24346.

[34] Xihong Yang, Yue Liu, Sihang Zhou, Siwei Wang, Wenxuan Tu, Qun Zheng, Xinwang Liu, Liming Fang, and En Zhu. 2023. Cluster-guided Contrastive Graph Clustering Network. *arXiv preprint arXiv:2301.01098* (2023).

[35] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. In *NeurIPS*, Vol. 33. 5812–5823.

[36] Xiaotong Zhang, Han Liu, Qimai Li, and Xiao-Ming Wu. 2019. Attributed graph clustering via adaptive graph convolution. In *IJCAI*. 4327–4333.

[37] Han Zhao, Xu Yang, Zhenru Wang, Erkun Yang, and Cheng Deng. 2021. Graph Debiased Contrastive Learning with Joint Representation Clustering. In *IJCAI*. 3434–3440.

[38] Hao Zhu, Ke Sun, and Peter Koniusz. 2021. Contrastive laplacian eigenmaps. In *NeurIPS*, Vol. 34. 5682–5695.

[39] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep graph contrastive representation learning. In *ICML Workshop on Graph Representation Learning and Beyond*.

[40] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph contrastive learning with adaptive augmentation. In *WWW*. 2069–2080.

# A  EXPERIMENTAL DETAILS

## A.1  Hyper-parameters settings

In this section, we list the hyperparameters utilized in our node clustering model for each of the datasets used in our experiments. The relevant parameters include $\beta$, $\gamma$, k, time, the hidden dimensions, and the number of epochs. Table 7 summarizes the hyperparameters for each dataset.

**Table 7: Hyperparameter of the node clustering.**

| Datasets | $\beta$ | $\gamma$ | k | time | *hid_dim* | epochs |
|----------|------|------|-----|------|-----------|--------|
| Cora     | 3    | 1    | 21  | 100  | 512       | 50     |
| Citeseer | 7    | 1    | 111 | 150  | 512       | 20     |
| AMAP     | 5    | 1    | 19  | 40   | 512       | 20     |
| BAT      | 0.7  | 1    | 21  | 200  | 64        | 25     |
| EAT      | 6    | 1.5  | 155 | 15   | 512       | 30     |
| CoraFull | 2    | 1    | 73  | 5    | 1024      | 12     |

## A.2  Visualization analysis

In this part, we visualize the distribution of the learned representation to show the superiority of SCGDN on Cora and AMAP datasets via t-SNE algorithm [29]. Three baselines and SCGDN are shown in Fig. 6, we can conclude that SCGDN better reveals the intrinsic clustering structure.



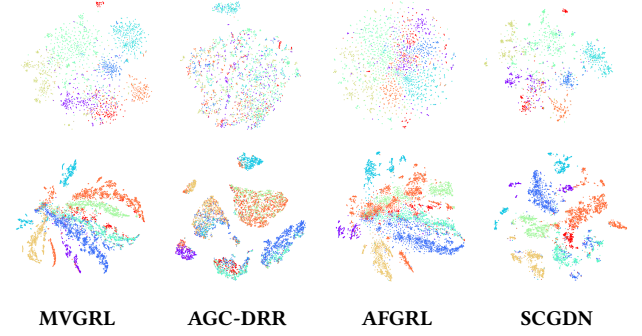**MVGRL        AGC-DRR        AFGRL        SCGDN**

**Figure 6: 2D t-SNE visualization of six methods on two benchmark datasets. The first row and second row corresponds to Cora and AMAP datasets, respectively.**