EasyChair Preprint

№ 8107

# Cross-Perspective Graph Contrastive Learning

Shiyang Lin, Chenhe Dong and Ying Shen



May 28, 2022

# Cross-perspective Graph Contrastive Learning *

Shiyang Lin, Chenhe Dong, and Ying Shen (✉)

School of intelligent of systems engineering, Sun Yat-sen University, ShenZhen, China
{linshy56,dongchh}@mail2.sysu.edu.cn, sheny76@mail.sysu.edu.cn

**Abstract.** Attributed graph representation has attracted increasing attention recently due to its broad applications such as node classification, link prediction and recommendation. Most existing methods adopt Graph Neural Network (GNN) or its variants to propagate the attributes over the structure network. However, the attribute information will be overshadowed by the structure perspective. To address the limitation and build a link between nodes features and network structure, we aim to learn a holistic representation from two perspectives: topology perspective and feature perspective. To be specific, we separately construct the feature graph and topology graph. Inspired by the network homophily, we argue that there is a deep correlation information between the network structure perspective and the node attributes perspective. Attempting to exploit the potential information between them, we extend our approaches by maximizing the consistency between structural perspective and attribute perspective. In addition, an information fusion module is presented to allow flexible information exchange and integration between the two perspectives. Experimental results on four benchmark datasets demonstrate the effectiveness of our proposed method on graph representation learning, compared with several representative baselines.

**Keywords:** Graph representation · Graph convolution networks · Contrastive learning · Self-attention mechanism · Semi-supervised learning.

## 1 Introduction

Attributed graph representation aims to learn low dimensional node representation by fully exploiting the rich information of topological structure, node features, and correlation between them. As typical methods in graph representation learning, the representation learned by GNNs has been proved to be effective in achieving the state-of-the-art performance in a variety of graph datasets such as citation networks [1], social networks [2] and recommended systems [4]. The underlying graph structure is utilized by GNNs to operate convolution directly on the graph by passing node features to neighbors, or perform convolution in the spectral domain using the Fourier basis of a given graph.

* This work was supported in part by the 173 program No. 2021-JCJQ-JJ-0029, the Shenzhen General Research Project under Grant JCYJ20190808182805919 and in part by the National Natural Science Foundation of China under Grant 61602013.

However, some recent studies disclose these GNNs methods tend to suffer from certain weaknesses of the state-of-the-art GNNs in fusing node features with network structure. For example, GNNs only perform low-pass filtering on feature vectors and node representation will become indistinguishable when we always utilize a low-pass filter, causing the over-smoothing problem [25]. GNNs also mainly retain the commonality of node features, which ignores the difference, so that the learned representations of connected nodes become similar [3]. Therefore, separately generating a feature graph and a topology graph for GNNs is a fundamental problem.

GNNs models are built with a supervised pattern, which require lots of labeled nodes for training. Recently, graph contrastive representation learning [13, 15] arouses a growing interest, which seeks to maximize the mutual information between input and its representation by contrasting positive pairs with negative-sampled counterparts. For example, [15] learns node representation with graph-structured data in an unsupervised manner. And a contrastive objective is proposed to maximize the mutual information between local node embedding and a summary global embedding. [16] considers mutual information in terms of graphical structure and proposes mutual information between input graphs and high-level embeddings in a straightforward pattern. [14] performs augmentation on the input graph to obtain two graph views, and maximize the mutual information between two graph views. However, these contrastive representation learning methods propagate feature information over topology graphs and contrast node-level embeddings to global ones.

In this paper, we propose a **C**ross-**p**erspective **G**raph **C**ontrastive **L**earning (CpGCL) for attributed network embedding. We construct a feature graph via the $k$-nearest neighbor algorithm and then perform graph convolution operation over both topology graph and feature graph. Then, with the feature representation and topology representation, to the best of our knowledge, we are the first to explore the consistency between feature perspective and topology perspective with contrastive learning. Finally, the information fusion module is designed to propagate the potential information and fuse the different perspective vectors. The major contributions of this paper are summarized as follows.

- Different from existing works on graph contrastive representation learning, we propose a novel training strategy to exploit the correlation between topology structure perspective and node features perspective.
- We develop a cross-perspective propagation-based architecture, which constructs feature graph and topology graph separately and performs graph convolution operation on both feature perspective and topology perspective. Combined with contrastive learning, heterogeneous information can be adequately fused.
- We conduct extensive experiments to demonstrate the effectiveness of the proposed method on four benchmark datasets.

We organize the rest of this paper as follows: Section 2 introduces related background on graph neural network and contrastive learning. Section 3 de-

scribes our proposed framework and provides motivation of our method. Section 4 reports our experimental results, followed by the conclusion in Section 6.

## 2 RELATED WORK

**Graph neural network**. GNNs have been a mainstream strategy to learn low dimensional node representation and have developed for a wide range of tasks, which propagate features information over network topology to node embedding. GraphSAGE [17] utilizes several aggregators and recursively aggregate features with sampled neighbors. Graph attention networks (GAT) [7] improves GNN with the attention mechanism on sampled neighbor nodes. GraphRNA [18] considers node attribute as a bipartite graph and advance graph convolutional networks to a more effective neural architecture. MixHop [12] proposes a graph convolutional layer that utilizes multiple powers of the adjacency matrix to learn both first-order and higher-order neighbors.

**Contrastive learning**. The main idea of contrastive learning is to learn representations by contrasting positive and negative samples. Contrastive learning can be applied to both supervised and unsupervised data and has been shown to achieve good performance on a variety of vision and language tasks [19]. Contrastive learning aims to learn representation by maximizing representation in different views. DGI [15] and MVGRL [26] propose to learn a global graph-level embedding and a local node-level embedding, and maximize the global embedding and local embedding. GCA [13] and GRACE [14] design two graph views by novel data augmentation schemes and maximize the agreement between node embeddings in these two views. SCRL [20] finds a "target" for the projection of prototype vectors and utilizes pseudo label using iteratively Sinkhorn algorithm, then sets up "exchanged problem" to predict.

## 3 METHODOLOGY

An attributed network is denoted as $G = (\mathbf{A}, \mathbf{X})$, where $\mathbf{A} \in \mathbb{R}^{n \times n}$ is the adjacency matrix of the input network with $n$ nodes and $\mathbf{X} \in \mathbb{R}^{n \times d}$ if the matrix of node attributes where $d$ is the dimension of the node features. Specifically, $A_{ij} = 1$ represents there is an edge between nodes $i$ and $j$, otherwise, $A_{ij} = 0$.

Given an attributed network $G = (\mathbf{A}, \mathbf{X})$, attributed network embedding aims to learn a function $f : v_i \rightarrow y_i$ that maps each node $v_i$ to a low dimensional representation vector $y_i$. Specifically, we use feature graph and topology graph to capture the underlying information in feature space and topology space, and adopt graph convolution over feature graph and topology graph specifically to aggregate the information (Section 3.1). Then a cross-perspective contrastive learning module is designed to exploit the consistency of feature information and topology information (Section 3.2). Afterwards, we introduce an information fusion module to propagate important information over both two graphs and integrate a common embedding from two perspectives (Section 3.3). Finally, we employ a contrastive objective (i.e., a discriminator) that enforces the encoded

embeddings of each node in the two different perspectives agree with each other (Section 3.4).

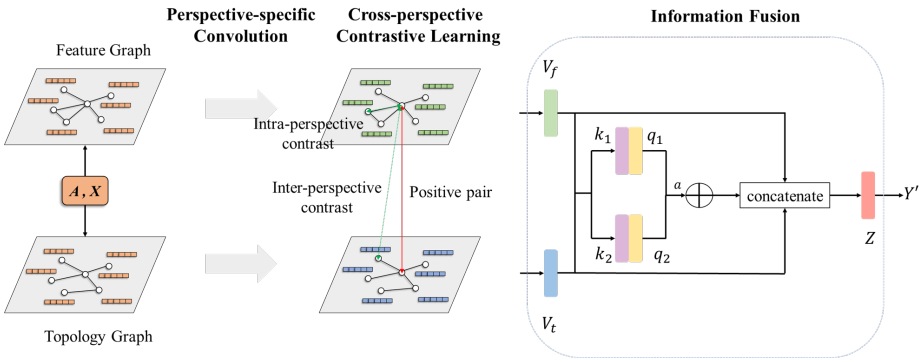



**Fig. 1.** The framework of CpGCL model. Given an attributed graph $G = (\mathbf{A}, \mathbf{X})$, faeture graph and topology graph are constructed. CpGCL consists of three parts: Perspective-specific Convolution Module, Cross-perspective Contrastive Learning Module and Information Fusion Module.

### 3.1 Perspective-specific Convolution Module

Merely propagating the feature information over the topology graph may only perform low-pass filtering on feature vectors and will smooth the difference between node features. A nature idea is to separately construct a feature graph and a topology graph, then adopt graph convolution operation over them.

To represent the node with feature perspective, we construct the feature graph $G_f = (\mathbf{A}_f, \mathbf{X})$ via $k$-nearest neighbour algorithm, where $\mathbf{A}_f$ is the adjacency matrix of feature graph and $\mathbf{X}$ is the feature matrix of graph. Concretely, we calculate the similarity matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$ with cosine similarity formula:

$$S_{ij} = \frac{x_i \cdot x_j}{|x_i| \cdot |x_j|}, \tag{1}$$

where $S_{ij}$ is the similarity between node feature $x_i$ and node feature $x_j$. Then we choose top $k$ similar node pairs for each node and establish edges. Finally, we obtain the adjacency matrix of feature graph $\mathbf{A}_f$.

To extract meaningful information from the feature graph, we adopt graph convolution over the feature graph. With the input graph $(\mathbf{A}_f, \mathbf{X})$ in feature space, the $l$-th output layer can be represented as:

$$\boldsymbol{f}^{(l)} = ReLU(\hat{\boldsymbol{D}}_f^{-\frac{1}{2}} \hat{\boldsymbol{A}}_f \hat{\boldsymbol{D}}_f^{-\frac{1}{2}} \boldsymbol{f}^{(l-1)} \boldsymbol{W}_f^{(l)}), \tag{2}$$

where $ReLU$ is the activation function, $\hat{\boldsymbol{A}}_f = \boldsymbol{A}_f + \boldsymbol{I}_f$, $\hat{\boldsymbol{D}}_f$ is the diagonal matrix of $\hat{\boldsymbol{A}}_f$, $\boldsymbol{W}_{\boldsymbol{f}}^{(\boldsymbol{l})}$ is the weight matrix of the $l$-th layer in GCN, $\boldsymbol{f}^{(l)}$ is the last layer output of GCN in feature perspective.

As for the topology perspective, we have the topology graph $G_t = (\mathbf{A}_t, \mathbf{X})$, where $\mathbf{A}_t$ is the adjacency matrix of topology graph and $\mathbf{X}$ is the feature matrix. So we can obtain the last layer output $\boldsymbol{t}^{(l)}$ of GCN in topology perspective in the same way as in the perspective of feature.

### 3.2 Cross-perspective Contrastive Learning Module

Contrary to previous works [15, 16] that learn representations by the node-level to the graph-level contrastive scheme, in CpGCL, we define the contrastive objective at the node-level and exploit the correlation between feature perspective and topology perspective.

To be specific, with feature embeddings $\mathbf{V}_f$ and topology embeddings $\mathbf{V}_t$, we employ a contrastive objective that distinguishes the embeddings of the same node in these two different perspectives from other node embeddings. For any node $v_i$, its embedding generated in feature perspective, $V_f$, is treated as the anchor, the embedding of it generated in topology perspective, $t$, is treated as the positive sample, and embeddings of nodes other than $V_t$ in the two perspectives are naturally regarded as negative samples. Formally, we define the critic $\theta(f,t) = s(g(f), g(t))$, where $s$ is the cosine similarity and $g$ is the non-linear projection to enhance the expression power of the critic. The projection $g$ is implemented with a two-layer multilayer perceptron (MLP). We define the pairwise objective for each positive pair $(f,t)$ as

$$l(V_f, V_t) = \mathbf{log}\frac{e^{\theta(f_i,t_i)/\tau}}{e^{\theta(f_i,t_i)/\tau} + \sum_{k\neq i} e^{\theta(f_i,t_i)/\tau} + \sum_{k\neq i} e^{\theta(f_i,t_k)/\tau}}, \tag{3}$$

where $\tau$ is a temperature parameter, $e^{\theta(f_i,t_i)/\tau}$ is the positive pair, $e^{\theta(f_i,t_i)/\tau}$ is the inter-perspective negative pairs and $e^{\theta(f_i,t_k)/\tau}$ is the intra-perspective negative pairs. Therefore, negative pairs come from two sources. Since two perspectives are symmetric, the loss for another perspective can be defined similarly for $l(t_i, f_i)$. The overall contrastive objective is defined as the average of two different forms:

$$L_c = \frac{1}{2N}\sum_{i=1}^{N}[l(f_i, t_i) + l(t_i, f_i)]. \tag{4}$$

To sum up, CpGCL first constructs feature graph $G_f$ and topology graph $G_t$ separately. Obtaining feature representation $v$ of $G_f$ and topology representation $t$ of $G_t$, we respectively use GNNs encoder to propagate the feature information and topology information. Finally, we learn embeddings by maximizing the agreement between feature representation $v$ and topology representation $t$.

### 3.3 Information Fusion Module

In order to exchange the information between feature perspective and topology perspective, another design goal of CpGCL framework is to propagate critical information along the network structure and maintain the discriminative features

in node attributes. Previous work [8] designs a Common-GCN with parameter sharing strategy to get the embeddings shared in two spaces. Instead, we utilize the self-attention mechanism [22] to exchange features between perspectives. The intuition includes two aspects. First, the self-attention mechanism is designed to learn the importance of two perspectives, without having one perspective separated from another. Second, the self-attention mechanism can adaptively obtain one vector for each perspective, so that the information fusion module can be naturally stacked.

Specifically, given the feature representation $V_f$ and topology representation $V_t$, following standard self-attention, we calculate the query vector $q$ and key vector $k$ for each node,

$$\begin{aligned} q_1 = V_f W^Q, \qquad q_2 = V_t W^Q, \\ k_1 = V_f W^K, \qquad k_2 = V_t W^K, \end{aligned} \tag{5}$$

where $d$ is the embedding dimension, $W_Q, W_K \in \mathbb{R}^{d \times d}$ denotes the transformation matrices for query vector and key vectors, respectively. Then we fuse the new node representation $V_1$ and $V_2$ with the following computation,

$$\begin{aligned} \alpha_{i,j} = \frac{\mathbf{exp}(q_i k_j^\top)}{\sum_{k \in \{k_1, k_2\}} \mathbf{exp}(q_i k^\top)}, \\ V_1 = \mu(\alpha_{1,1} V_f + \alpha_{1,2} V_t), \\ V_2 = \mu(\alpha_{2,1} V_f + \alpha_{2,2} V_t). \end{aligned} \tag{6}$$

where $\alpha_{.,.}$ denotes the relative weights of the intermediate features $V_f$ and $V_t$ for the node representations, and $\mu$ is the activation function.

Then we combine these two node representation $V_1$ and $V_2$ to obtain the common embedding $V$ of two perspectives.

$$V = (V_1 + V_2)/2, \tag{7}$$

### 3.4 Optimization Objective

To preserve the information from feature perspective and topology perspective, $V_f$, $V_t$ and $V$ are concatenated as the final embedding $Z$. Then we use $Z$ for semi-supervised classification with a linear transformation and softmax function. $Y^{'}$ is the prediction result and $Y^{'}_{ij}$ is the probability of node $i$ belonging to class $j$. $W$ and $a$ is the weight and bias of the linear layer, respectively.

$$Y' = (W \cdot Z + b). \tag{8}$$

Suppose there are $T$ nodes in training set, we adopt cross-entropy loss to measure the difference between predicted label $Y'_{ij}$and truth label $Y_{ij}$.

$$L_t = \sum_{i=1}^{T} \sum_{j=1}^{C} Y_{ij} \mathbf{ln} Y'_{ij}. \tag{9}$$

Combining the node classification task and contrastive learning, we have the following overall objective function:

$$L = L_t + \beta L_c, \tag{10}$$

where $\beta$ is the consistency balancing hyper-parameter.

## 4 Experimental Analysis

In this section, we conduct extensive experiments to evaluate the effectiveness of the cross-perspective graph contrastive learning framework for node classification on attributed graph representation.

### 4.1 Datasets

For a comprehensive comparison, we use four widely-used datasets, including ACM [27], Citeseer [21], UAI2010 [23] and Cora to study the performance of node classification; their detailed statistics is summarized in Table 1. Moreover, we provide all the data websites in the supplement for reproducibility.

**ACM**[1] contains papers published in KDD, SIGMOD, SIGCOMM, MobiCOMM, and VLDB and are divided into three classes (Database, Wireless Communication, Data Mining). The constructed graph comprises 3,025 papers, 5,835 authors, and 56 subjects. Paper features correspond to elements of a bag-of-words represented of keywords. **Citeseer**[2] consists of 3,312 scientific publications classified into one of six classes and 4,732 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. **UAI2010**[3] contains 3,067 nodes in 19 classes and it has been tested in GCN for community detection. The attribute dimension of each node is 4,973. **Cora**[4] is a paper citation network, which contains 2,708 papers as nodes and 5,249 citation links as edges. These papers are divided into seven categories. The attribute of each node is a binary vector of 1,433 dimensions.

**Table 1.** The statistic of the datasets

| Dataset | Nodes | Edges | Classes | Features |
|---|---|---|---|---|
| ACM | 3,025 | 13,128 | 3 | 1,870 |
| Citeseer | 3,327 | 4,732 | 6 | 3,702 |
| UAI2010 | 3,067 | 28,311 | 19 | 4,973 |
| Cora | 2,708 | 5,429 | 7 | 1,433 |

[1] https://github.com/Jhy1993/HAN
[2] https://github.com/tkipf/pygcn
[3] http://linqs.umiacs.umd.edu/projects//projects/lbc/index.html
[4] Cora dataset is available at https://linqs.soe.ucsc.edu/data

### 4.2 Baselines

We compare the CpGCL framework with graph representative baselines to verify the performance.

- **DeepWalk** [6] is a classical graph embedding method which uses random walk and skipgram to learn network representations.
- **Line** [9] preserves the first-order or second-order proximity in the network by optimizing the carefully designed objective function.
- **ChebNet** [10] is a spectral-based GCN that uses Chebyshev filters to reduce computation complexity.
- **GCN** [5] is a semi-supervised network embedding method that applies average aggregation in the local neighborhood.
- **kNN-GCN** [8] uses the sparse $k$-nearest neighbour graph calulated from feature matrix as input graph of GCN and represent it as kNN-GCN.
- **GAT** [7] is a semi-supervised neural network which learn the importance between nodes and its neighbors and fuse the neighbors to perform node classification.
- **Demo-Net** [11] is a degree-specific graph neural network using multi-task graph convolution for node classification.
- **MixHop** [12] is a propagation-based method that mixes the node representation of highter-order neighbours in one graph convolutional layer.
- **GRACE** [13] is a proposed graph contrastive learning framework. It generates two graph view and maximize the agreement of node representations of two views.
- **AMGCN** [8] exploits the information from both feature space and topology space. Then it uses the attention mechanism to learn adaptive importance weights of the embeddings.
- **SCRL** [20] presents a self-supervised framework to learn a consensus representation for attributed graph. It fuses the topology information and node feature information of the graph.

### 4.3 Parameters Setting

For all the baseline methods, we use the implementations provided by either their authors or open-source libraries. By default, we build a 2-layer GCN with the same hidden layer dimension $n \in \{512, 768\}$ and train our model utilizing Adam [24] optimizer with learning rate range from 0.0001 to 0.0005. In order to prevent the over-fitting problem, we set the dropout rate to 0.5. In addition, we set weight decay $\in \{5e-4, 5e-3\}$, temperature parameter $\tau \in \{0.8, 0.9, 1.0, 1.1\}$ for contrastive objective and $k \in \{2, ..., 9\}$ for the kNN graphs. The balancing hyper-parameter is set from 0.7 to 1.0. For fairness, we follow [8] and select three label rates for the training set (i.e., 20, 40, 60 labeled nodes per class) and choose 1000 nodes as the test set. The selection of labeled nodes on each dataset is identical for all compared baselines. We repeatedly train and test our model for 5 times with the same partition and evaluate the performance of our model by Accuracy (ACC) and Macro-F1 (F1).

**Table 2.** Node classification results(%). (Bold: best; Underline: runner-up)

| Dataset | ACM | | | | | | Citeseer | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| L/C | 20 | | 40 | | 60 | | 20 | | 40 | | 60 | |
| Metrics | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 |
| DeepWalk[6] | 62.69 | 62.11 | 63.00 | 61.88 | 67.03 | 66.99 | 43.47 | 38.08 | 45.15 | 43.18 | 48.86 | 48.01 |
| LINE[9] | 41.28 | 40.12 | 45.83 | 45.79 | 50.41 | 49.92 | 32.71 | 31.75 | 33.32 | 32.42 | 35.39 | 34.37 |
| ChebNet[10] | 75.37 | 74.93 | 81.68 | 81.33 | 85.78 | 85.32 | 69.64 | 65.91 | 71.52 | 68.23 | 73.21 | 70.24 |
| GCN[5] | 87.64 | 87.80 | 88.96 | 89.02 | 90.37 | 90.43 | 70.30 | 67.42 | 72.98 | 69.66 | 74.43 | 71.29 |
| kNN-GCN | 78.54 | 78.14 | 81.61 | 81.55 | 81.94 | 81.89 | 61.37 | 58.83 | 61.59 | 59.42 | 62.46 | 60.13 |
| GAT[7] | 87.47 | 87.59 | 88.51 | 88.47 | 90.26 | 90.33 | 72.53 | 68.17 | 73.02 | 69.59 | 74.71 | 70.37 |
| Demo-Net[11] | 84.48 | 84.17 | 85.64 | 84.78 | 86.65 | 84.09 | 69.52 | 67.81 | 70.39 | 66.92 | 71.88 | 68.24 |
| MixHop[12] | 81.08 | 81.42 | 82.37 | 81.09 | 83.03 | 82.36 | 71.40 | 66.96 | 71.56 | 67.47 | 72.31 | 69.37 |
| GRACE[13] | 89.04 | 89.00 | 89.46 | 89.36 | 91.08 | 91.03 | 71.70 | 68.14 | 72.38 | 68.74 | 74.20 | 70.73 |
| AMGCN[8] | 90.40 | 90.43 | 90.76 | 90.66 | 91.40 | 90.69 | 73.12 | 68.44 | 74.62 | 69.84 | 75.56 | 70.94 |
| SCRL[20] | 88.70 | 88.46 | 90.70 | 90.71 | 91.80 | 91.83 | 73.00 | 68.81 | 73.80 | 70.17 | 75.50 | 71.86 |
| CpGCL | **91.82** | **91.68** | **91.92** | **91.84** | **92.20** | **92.15** | **73.40** | **70.30** | **75.90** | **71.42** | **76.60** | **72.80** |
| Dataset | UAI2010 | | | | | | Cora | | | | | |
| L/C | 20 | | 40 | | 60 | | 20 | | 40 | | 60 | |
| Metrics | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 |
| DeepWalk[6] | 42.15 | 33.04 | 50.71 | 46.24 | 54.71 | 46.59 | 73.22 | 71.90 | 75.12 | 73.88 | 76.10 | 74.56 |
| LINE[9] | 43.43 | 37.16 | 45.83 | 39.81 | 50.41 | 43.71 | 73.56 | 72.04 | 74.84 | 73.36 | 75.42 | 74.28 |
| ChebNet[10] | 50.12 | 33.75 | 58.18 | 38.82 | 59.74 | 40.77 | 76.68 | 75.82 | 77.56 | 76.34 | 78.21 | 77.24 |
| GCN[5] | 49.98 | 32.96 | 51.87 | 33.90 | 54.53 | 32.31 | 77.30 | 76.53 | 78.98 | 77.79 | 79.83 | 78.84 |
| kNN-GCN | 66.06 | 52.43 | 68.74 | 54.45 | 71.68 | 54.82 | 61.57 | 58.83 | 64.59 | 61.24 | 67.48 | 63.28 |
| GAT[7] | 56.87 | 39.59 | 63.71 | 45.11 | 68.47 | 48.92 | 78.53 | 77.38 | 79.82 | 78.56 | 80.71 | 79.57 |
| Demo-Net[11] | 23.71 | 16.90 | 30.57 | 26.64 | 34.53 | 29.31 | 76.40 | 75.43 | 77.33 | 76.82 | 77.89 | 76.46 |
| MixHop [12] | 61.56 | 49.23 | 65.11 | 53.93 | 67.71 | 56.42 | 72.40 | 70.88 | 73.56 | 72.47 | 75.31 | 73.82 |
| GRACE[13] | 65.59 | 48.43 | 66.71 | 49.57 | 68.71 | 51.49 | 76.84 | 75.72 | 78.26 | 77.20 | 79.18 | 78.22 |
| AMGCN[8] | 70.14 | 55.67 | 73.17 | 64.91 | 74.42 | 66.08 | 80.80 | 79.98 | 83.57 | 82.39 | 84.21 | 83.17 |
| SCRL[20] | 71.90 | 58.40 | 73.52 | 64.70 | 74.90 | 66.54 | 80.60 | 79.72 | 84.17 | 83.28 | 84.06 | 83.72 |
| CpGCL | **73.70** | **62.06** | **74.50** | **66.42** | **75.84** | **68.62** | **82.50** | **82.02** | **85.10** | **84.42** | **85.36** | **84.60** |

### 4.4 Node Classification Results

The node classification results are reported in Table 2, where L/C means the number of labeled nodes per class for training. We have the following observations.

- Compared with all baselines, the proposed CpGCL generally achieves the best performance on all datasets with all label rates. Compared with the best competitor (SCRL), CpGCL improves it by 1.35% and 1.68%for ACC and F1 respectively across all datasets. The results implicate the effectiveness of the cross-perspective contrastive learning and information fusion modules. CpGCL can better capture the difference between nodes and their neighbor nodes pairs.
- In some situations, the feature graph shows better performance than on topology graph, such as UAI2010. Comparing with GCN, kNN-GCN and

GAT, CpGCL achieves substantial improvement on all datasets, which further confirms the necessity of the feature graph.
- CpGCL consistently outperforms GRACE on all the datasets, indicating the effectiveness of cross-perspective contrastive learning in CpGCL, because CpGCL effectively learns the correlation between feature perspective and topology perspective.

### 4.5 Ablation Study

Here, we conduct an ablation study by discarding some of the design choices shown in Fig.1. The node classification results on ACM, Citeseer, UAI2010, and Cora are shown in Table 3. In the table, **CpGCL - IF** means discarding the information fusion module, **CpGCL - CL** means CpGCL without the cross-perspective contrastive learning module and removing the contrastive objective loss. As we can see from the Table 3, the results of CpGCL are obviously better than all its variants on all datasets with all labeled rates, verifying that information fusion module and cross-perspective contrastive learning module are effective in terms of improving the node representation results for classification.

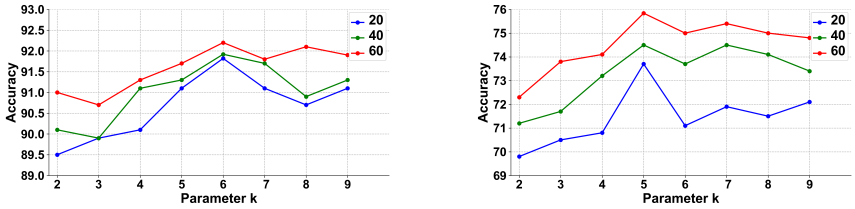
**Table 3.** The results of CpGCL and its variants on four datasets.

| Dataset | Metrices | L/C | CpGCL | CpGCL-IF | CpGCL-CL |
|---|---|---|---|---|---|
| ACM | ACC | 20 | 91.82 | 89.95 | 89.70 |
| | | 40 | 91.92 | 89.70 | 89.50 |
| | | 60 | 92.20 | 90.20 | 90.20 |
| | F1 | 20 | 91.68 | 90.10 | 89.22 |
| | | 40 | 91.84 | 90.20 | 89.47 |
| | | 60 | 92.15 | 90.49 | 90.22 |
| Citeseer | ACC | 20 | 73.40 | 72.20 | 72.40 |
| | | 40 | 75.90 | 72.70 | 73.30 |
| | | 60 | 76.60 | 74.90 | 74.70 |
| | F1 | 20 | 70.30 | 67.93 | 68.58 |
| | | 40 | 71.42 | 69.46 | 69.95 |
| | | 60 | 72.80 | 71.96 | 71.43 |
| UAI2010 | ACC | 20 | 73.70 | 69.90 | 69.30 |
| | | 40 | 74.50 | 73.50 | 73.40 |
| | | 60 | 75.84 | 74.40 | 75.20 |
| | F1 | 20 | 62.06 | 59.42 | 59.31 |
| | | 40 | 66.42 | 64.89 | 64.49 |
| | | 60 | 68.42 | 66.56 | 67.55 |
| Cora | ACC | 20 | 82.50 | 82.26 | 80.79 |
| | | 40 | 85.10 | 82.42 | 82.33 |
| | | 60 | 85.36 | 82.66 | 82.88 |
| | F1 | 20 | 82.02 | 81.36 | 81.20 |
| | | 40 | 84.42 | 83.20 | 83.10 |
| | | 60 | 84.60 | 83.80 | 83.70 |

### 4.6 Parameter Sensitivity

We also study the sensitivity of the one major hyper-parameter k-nearest neighbor graph k on ACM and UAI2010 datasets.

**Parameter** $k$: We study the impact of the top $k$ neighborhoods in the kNN graph with $k$ range from 2 to 9. We conduct experiments on ACM and UAI2010 datasets, their accuracies have similar trends. The results are shown in Fig2. The accuracy witness a climb first, which is followed by a decrease. The reason is that larger $k$ can provide more useful feature information but too many neighbors will also introduce noisy edges.




(a) ACM (b) UAI2010

**Fig. 2.** Analysis of parameter $k$

## 5 Conclusion

In this paper, we propose a cross-perspective graph contrastive learning framework on the attributed graph, which is able to explore the attribute information and topology information adaptively. Requiring the embeddings of feature graph and topology graph, the cross-perspective contrastive learning module is proved to be effective to learn the consistent information between feature perspective and topology perspective. To further fuse the information, we introduce the information fusion module to flexibly exchange and integrate information between the two perspectives. Experimental results on real-world datasets demonstrate the superiority of our proposed method on attributed graph representation.

## References

1. Song Q, Wang X, Wang R. 2021. Knowledge Network and Visual Analysis of Knowledge Graph Research. In ICVRS. 61-66.
2. Qiu J, Tang J, Ma H, et al. 2018. Deepinf: Social influence prediction with deep learning. In SIGKDD. 2110-2119.
3. Bo D, Wang X, Shi C, et al. 2021. Beyond low-frequency information in graph convolutional networks. In AAAI.

4. Ying R, He R, Chen K, et al. 2018. Graph convolutional neural networks for web-scale recommender systems. In SIGKDD 974-983.
5. Kipf T N, Welling M. 2016. Semi-supervised classification with graph convolutional networks. In ICLR.
6. Perozzi B, Al-Rfou R, Skiena S. 2014. Deepwalk: Online learning of social representations. In SIGKDD. 701-710.
7. Velickovic P, Cucurull G, Casanova A, Romero A, Lio P, & Bengio Y. 2017. Graph attention networks. In ICLR.
8. Xiao Wang, Meiqi Zhu, Deyu Bo, Peng Cui, Chuan Shi, and Jian Pei. 2020. AM-GCN: Adaptive Multi-channel Graph Convolutional Networks. In SIGKDD. 1243–1253.
9. Tang J, Qu M, Wang M, et al. 2015. Line: Large-scale information network embedding. In WWW. 1067-1077.
10. Defferrard M, Bresson X, Vandergheynst P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In NeruIPS, 29.
11. Wu J, He J, Xu J. 2019. 2019. Net: Degree-specific graph neural networks for node and graph classification. In SIGKDD. 406-415.
12. Abu-El-Haija S, Perozzi B, Kapoor A, et al. 2019. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In PLMR. 21-29.
13. Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep Graph Contrastive Representation Learning. In ICML.
14. Zhu Y, Xu Y, Yu F, et al. 2021. Graph contrastive learning with adaptive augmentation. In WWW. 2069-2080.
15. Velickovic P, Fedus W, Hamilton W L, et al. 2019. Deep Graph Infomax. In ICLR.
16. Peng Z, Huang W, Luo M, et al. 2020. Graph representation learning via graphical mutual information maximization. In WWW. 259-270.
17. Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In NeruIPS. 1024–1034.
18. Huang X, Song Q, Li Y, et al. 2019. Graph recurrent networks with attributed random walks. In SIGKDD. 732-740.
19. Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In CVPR. 9726–9735.
20. Liu C, Wen L, Kang Z, et al. 2021. Self-supervised consensus representation learning for attributed graph. In MM. 2654-2662.
21. Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In ICLR.
22. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In NeurIPS. 5998–6008.
23. Wang W, Liu X, Jiao P, et al. 2018. A unified weakly supervised framework for community detection and semantic matching. In PAKDD. 218-230.
24. Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In ICLR.
25. Zhu R, Tao Z, Li Y, et al. 2021. Automated graph learning via population based self-tuning GCN. In SIGKDD. 2096-2100.
26. Hassani K, Khasahmadi A H. 2020. Contrastive multi-view representation learning on graphs. In PMLR.
27. Wang X, Ji H, Shi C, et al. 2019. Heterogeneous graph attention network. In WWW. 2022-2032.