MI-GCN: Node Mutual Information-based Graph Convolutional Network

Lei Tian
Center for Applied Mathematics
Tianjin University
Tianjin 300072, China
tianlei@tju.edu.cn

Huaming Wu*
Center for Applied Mathematics
Tianjin University
Tianjin 300072, China
whming@tju.edu.cn

ABSTRACT

Graph Neural Networks (GNNs) have been widely used in various processing tasks for processing graphs and complex network data. However, in recent studies, GNNs cannot effectively process the structural topology information and characteristics of the nodes in the graph, or even fail to deal with the information of the nodes. For optimal node embedding aggregation and delivery, this weakness may severely affect the ability of GNNs to classify nodes. In order to overcome this issue, we propose a novel node Mutual Information-based Graph Convolutional Network (MI-GCN) for semi-supervised node classification. First, we analyze the node information entropy that measures the importance of nodes in the complex network, and further define the node joint information entropy and node mutual information in the graph data. Then, we use node mutual information to strengthen the ability of GNNs to fuse node structural information. Extensive experiments demonstrate that our MI-GCN not only retains the advantages of the most advanced GNNs, but also improves the ability to fuse node structural information. MI-GCN can achieve superior performance on node classification compared to several baselines on real-world multi-type datasets, including fixed data splits and random data splits.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Networks** → **Network architectures**.

KEYWORDS

Graph Neural Networks, Information Entropy, Mutual information

ACM Reference Format:
Lei Tian and Huaming Wu. 2022. MI-GCN: Node Mutual Information-based Graph Convolutional Network. In *Companion Proceedings of the Web Conference 2022 (WWW '22 Companion), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3487553.3524711

*Corresponding Author.

WWW '22 Companion, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9130-6/22/04…$15.00
https://doi.org/10.1145/3487553.3524711

1 INTRODUCTION

Graph data is a vital data structure. The relationship between many entities in the real world (such as social media, traffic flow, chemistry, and sociology) can be represented by graph data [9]. There are also many types of tasks for graph data, such as node classification, link prediction, graph clustering, graph classification and other tasks all involve graph edges and nodes [11]. If the nodes in the graph data can be represented well, the efficiency and accuracy of these tasks will be greatly promoted. The efficient processing of graph structure data by GNNs and its effective variants has led to the successful development of this field. GNNs effectively obtain graph data by aggregating neighbor node information to generate node embedding.

Unfortunately, GNNs can work normally only when there is a strong correlation between node features and node labels, and existing GNNs are not clear about the processing capabilities of the structure information and labels between nodes [7]. In addition, a node often has one or more functions in almost all graph data. This function determines the role or status of the node in the network to a certain extent. For instance, each person has a social role as an entity, role or status in a social network [15]. In the citation network, the number of citations of the paper must also reflect the importance of the paper in the network data [8]. On the one hand, proteins play a special role in the Protein-Protein interaction (PPi) network [13]. On the other hand, by removing some key nodes, the connectivity of the network can be greatly reduced to prevent the virus outbreak or the spread of rumors [4]. Therefore, the importance of nodes mainly reflected in the characteristics of the network structure [22] is very significant in the network. However, in existing GNN models, the structural characteristics of the nodes in the complex network have not been specifically improved, and further exploration is required in this area.

In information theory, the amount of information measures the information brought about by a specific event, and the information entropy is the expectation of the amount of information. Mutual information is a description of the degree of dependence between two events [18]. Several studies [2, 16, 21] have introduced node information entropy theory into complex networks to calculate the importance of nodes. Driven by this analysis, we further improve and supplement the entropy theory in graph structure data, and then introduce it into the GNNs to overcome the weakness.

The ability of a good GNN is essentially to extract and integrate the most relevant node information for the task. However, the obstacle in reality is that the correlation between graph data and the task is usually very complex and unknowable [28]. For example, the label of the classification task can be associated with

topology, node characteristics, or a combination thereof. In order to cope with this kind of challenge, we design a novel node Mutual Information-based Graph Convolutional Network (MI-GCN), which enhances the ability of GNNs to extract and merge the information of structural similarity between nodes. Technically, we add k-nearest edges by using the mutual information generated from the node structural information, and use the mutual information between nodes to optimize the original edges to further improve the ability of GNNs to extract and merge node information. The main contributions of this paper can be summarized as follows:

- To the best of our knowledge, we are the first to define node joint information entropy and node mutual information in graph data and further improve the theory of node information entropy in graph data.
- We pioneer to design a novel GCN model based on node mutual information, which improves the ability of GNNs to extract and merge the structure information between nodes.
- Extensive experiments demonstrate that the performance of our proposed model is superior to some current popular GNNs in node classification, which proves that MI-GCN can better extract the features of fusion nodes.

2 RELATED WORK

In recent years, GNN model has been extensively studied. Kipf *et al.* [11] first proposed a method based on an effective variant of Convolutional Neural Networks (CNNs), which operates directly on the graph. The local first-order approximation of the spectrogram convolution is based on the convolution architecture to learn the hidden layer representation that encodes the local graph structure and node features. Rong *et al.* [24] randomly deleted a certain number of edges from the input graph in each training period, acting as a data enhancer and message transmission reducer to reduce the excessively smooth convergence speed and alleviate the resulting loss of information. Veličković *et al.* [26] proposed Graph Attention Network (GAT), which is a new type of neural network architecture that operates on graph structure data. It uses self-attention layers to stack the layers, in which nodes can participate in the characteristics of their neighborhoods, which are different in the neighborhood. The nodes are assigned different weights without any expensive matrix operations. Chiang *et al.* [3] proposed that the node blocks associated with the dense subgraphs identified by the graph clustering algorithm are sampled, and the neighborhood search in the subgraphs is restricted to improve memory and computational efficiency. Xu *et al.* [29] found that the use of graph wavelet transform can be obtained through fast algorithms, and does not require matrix feature decomposition to provide high efficiency for graph convolution.

Moreover, many works are also analyzing network topology and node characteristics to learn node embedding. Nt *et al.* [17] developed a theoretical framework based on graph signal processing, and found that feature vectors have provided considerable information for classification tasks. You *et al.* [31] proposed a non-linear distance weighted aggregation scheme between learning nodes, which is used to calculate a new type of GNN in the embedding of position-aware nodes. Bai *et al.* [1] used the graph attention module to model dynamic topology. Li *et al.* [12] propose a spatiotemporal fusion graph neural network that generates a 'temporal graph' to compensate for unreacted correlations in the spatial graph.

Most of the aforementioned work attempted to take the original topological structure of the graph data as input, or change the topological structure of the graph only by node features, subgraph clustering, or even random deletion of simple forms, while neglecting the importance of nodes in the graph and the similarity of first-order neighbors between nodes. Unlike previous approaches, we integrate the node mutual information into GNNs that are naturally applicable to graphs. The proposed MI-GCN can not only retain the advantages of the most advanced GNNs, but also improve the ability to fuse node structural information. In particular, we first start from the node information entropy that measures the importance of nodes in the complex network, and compare the graph data node information entropy and node mutual information theory. Furthermore, we strengthen the ability of GNNs by using node mutual information and enhance the ability to generate node representation by fusing node structural information.

3 PRELIMINARY

Let $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ be the input graph of size N, where $v \in \mathbb{V}$ represents nodes and $(v_i, v_j) \in \mathbb{E}$ represents edges. The adjacency matrix is denoted as $A \in \mathbb{R}^{N\times N}$. The node features are defined as $X = \{x_1, x_2, \cdots, x_n\} \in \mathbb{R}^{N\times M}$. The node degrees are denoted by $\{d_1, d_2, \cdots, d_i, \cdots, d_n\}$, where d_i computes the sum of all edges connected to node v_i. D is denoted as a degree matrix whose diagonal elements are composed of $\{d_1, d_2, \cdots, d_i, \cdots, d_n\}$. The symbols and definitions in the paper are summarized in Table 1.

Table 1: Symbols and Definitions

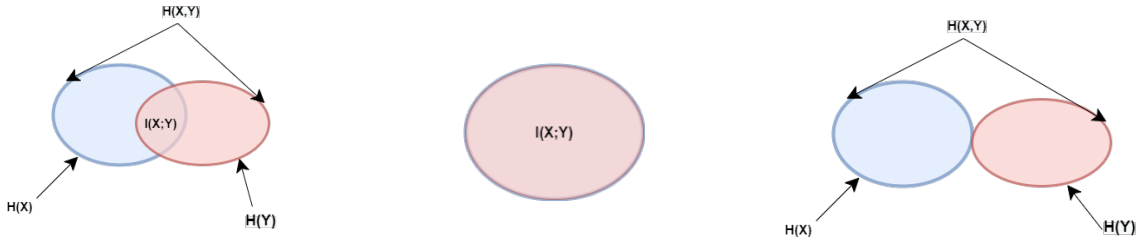
Symbol	Definition
$\mathbb{G} = (\mathbb{V}, E)$	Graphs
N	Number of nodes in the graph
$\mathbb{V} = \{v_1, v_2, \cdots, v_n\}$	Set of nodes
$E = \{e_1, e_2, \cdots, e_m\}$	Set of edges
$\mathbb{D}_i = \{D_{i1}, \cdots, D_{ij}, \cdots\}$	The number of DNN layers
$A \in \mathbb{R}^{N\times N}$	Adjacency matrix
$X = \{x_1, x_2, \cdots, x_n\}$	Node feature matrix
I	Identity matrix
$\hat{D}$	Degree matrix
E_v	Information entropy of node v
Γ_v	The first-order neighbor set of node v
E_{mv}	Joint information entropy between node v and node m
$\Gamma_{mv\to com}$	The common first-order neighbor set of nodes m and v
X, Y	Random variables X and Y
$H(X)$, $H(Y)$	Information entropy of random variables X and Y
$H(X, Y)$	Joint information entropy of random variables X and Y
$I(X; Y)$	Mutual information of random variables X and Y
$I(m; v)$	Mutual information between node v and node m
$MI \in \mathbb{R}^{N\times N}$	Node mutual information matrix
C	Number of node classes

3.1 Graph Convolutional Network

GCN [11] has become the most popular GNN in the past few years. The node embedding propagation formula in GCN is described as:

$$Z^{(l+1)} = \sigma(\hat{A}Z^{(l)}W^{(l)}), \tag{1}$$

where $Z^{(l+1)} = \{z_1^{(l+1)}, z_2^{(l+1)}, \cdots, z_i^{(l+1)}, \cdots, z_n^{(l+1)}\}$ is the node embedding matrix of the $l+1$-th layer, $z_i^{(l+1)}$ is the node representation of the $l+1$-th layer for node i; σ represents a nonlinear



(a) The relationship between entropy, joint entropy, and mutual information

(b) Extreme case 1

(c) Extreme case 2

Figure 1: shows the relationship between entropy, joint entropy, and mutual information. (a) shows the relationship between the three in general, (b) shows random variable (X, Y) distribution one-to-one correspondence, where $I(X;Y) = H(X) = H(Y)$. (c) shows random variables are completely statistically independent, where $I(X;Y) = 0$ and $H(X,Y) = H(X) + H(Y)$.

activation function; $\hat{A} = \hat{D}^{-\frac{1}{2}}(A+I)\hat{D}^{-\frac{1}{2}}$ is renormalized adjacency matrix with self-loop edges, where $\hat{D}$ is defined as the degree matrix corresponding to $A + I$, and $W^{(l)}$ is denoted as a learnable weight matrix.

3.2 Information Entropy and Mutual Information

In information theory, if X and Y are discrete random variables, the mutual information of two random variables can be expressed as [5]:

$$I(X;Y) = H(X) + H(Y) - H(X,Y), \tag{2}$$

where $H(X)$ and $H(Y)$ are the entropy of X and Y, respectively. $H(X,Y)$ is the joint information entropy of random variables (X,Y). The relationship between the entropy of random variables x and y, joint entropy, and mutual information is as shown in Fig. 1,

In addition, the theories of entropy and mutual information are widely used in image segmentation and image registration [20]. Among them, it mainly involves medical image segmentation, multi-modal image registration, image fusion and other fields [6, 23, 25].

4 APPROACH

Our node Mutual Information-based Graph Convolutional Network (MI-GCN) can be decomposed into two steps, i.e., construction of input graph structure and node feature learning. Firstly, we define the two basic concepts of node joint information entropy and node mutual information, and then use node mutual information to optimize the original graph topology. Finally, we take advantage of GCN in node feature learning to improve the performance of the model. The framework of MI-GCN is as shown in Fig. 2.

4.1 Entropy and Mutual Information on Graphs

Node information entropy is of great significance for measuring the importance of nodes in graph data [16]. The information entropy of each node v in the graph $\mathbb{G}$ can be calculated by:

$$E_v = \sum_{u \in \Gamma_v} H_{uv} = -\sum_{u \in \Gamma_v} p_{uv} \log p_{uv}, \ \forall v \in V, \tag{3}$$

where Γ_v is the first-order neighbor set of node v, $p_{uv} = \frac{d_u}{\sum_{l \in \Gamma_v} d_l}$ and d_u is the degree of node u. Moreover, we have $\sum_{l \in \Gamma_v} p_{lv} = 1$.

Definition 1. ***Node Joint Information Entropy.*** *When node m and node v share common first-order neighbors, the node joint information entropy of node m and node v in the graph data can be defined as:*

$$\begin{aligned} E_{mv} &= \sum_{u \in \Gamma_{mv \to com}} H_{u \to mv} \\ &= -\sum_{u \in \Gamma_{mv \to com}} p_{u \to mv} \log p_{u \to mv}, \end{aligned} \tag{4}$$

where $\Gamma_{mv \to com}$ is defined as the common first-order neighbor set of node m and node v, and $p_{u \to mv} = \frac{d_u}{\sum_{l \in \Gamma_{mv \to com}} d_l}$.

In particular, when node m and node v do not have any common first-order neighbors, they are completely independent of each other in the first-order graph structure. By using the relationship between joint information entropy and information entropy (as shown in Fig. 1c). In this case, the node joint information entropy between nodes can be defined as:

$$E_{mv} = E_m + E_v. \tag{5}$$

In summary, the node joint information entropy of any two nodes m and v in the graph $\mathbb{G}$ can be calculated by:

$$E_{mv} = \begin{cases} E_m + E_v, & \text{if } \Gamma_{mv \to com} = \varnothing; \\ \sum\limits_{u \in \Gamma_{mv \to com}} p_{u \to mv} \log \frac{1}{p_{u \to mv}}, & \text{else.} \end{cases} \tag{6}$$

Definition 2. ***Node Mutual Information.*** *Node mutual information can be defined by the relationship between mutual information, information entropy, and joint information entropy. According to Eq. 2, we have:*

$$I(m;v) = E_m + E_v - E_{mv}. \tag{7}$$

In particular, we have $I(v;v) = E_v$ when $m = v$.

Since the common first-order neighbors between the same nodes are regarded as one-to-one correspondence at this time, it is obtained by Fig. 1b.

Complexity Analysis: Since the calculation of mutual information between nodes in the graph needs to traverse the nodes in the entire graph. For each node, it is necessary to calculate the common neighbors of all nodes and then calculate the mutual information between nodes. If the number of nodes in the graph is large, the calculation cost will be relatively expensive. The time complexity of Algorithm 1 is $O(N^2)$.

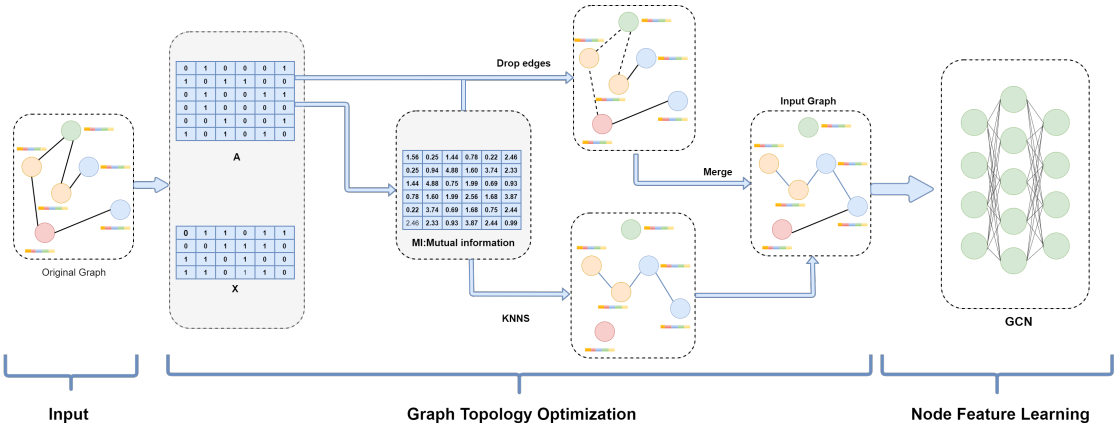


Figure 2: The framework of MI-GCN, where X represents the feature matrix of nodes, A represents the adjacency matrix of the original graph, the blue edges in the figure represent the virtual edges generated by the node mutual information matrix, the dashed edges represent edges of the original graph to be deleted, and finally edges of the input graph are the fusion of virtual edges and original edges after deletion.

Algorithm 1 Construct node mutual information matrix

Require: $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, N, $\mathbb{V} = \{v_1, v_2, \cdots, v_n\}$, $\mathbb{E} = \{e_1, e_2, \cdots, e_m\}$
Ensure: node mutual information matrix (MI)

Initialize $MI = O \in \mathbb{R}^{N \times N}$, Node information entropy $\mathbb{E} \in \mathbb{R}^{1 \times N}$.
for $v \in \mathbb{V}$ **do**
  for $(u, v) \in \mathbb{E}$ **do**
    $p_{uv} = \frac{d_u}{\sum_{l \in \Gamma_v} d_l}$
    Calculate E_v via Eq. 3
  end for
end for
Initialize the node joint entropy matrix $JE \in \mathbb{R}^{N \times N}$
for $v \in \mathbb{V}$ **do**
  for $m \in \mathbb{V}$ **do**
    if $\Gamma_{mv \to com} = \emptyset$ **then**
      Calculate E_{mv} via Eq. 5
    else
      $p_{u \to mv} = \frac{d_u}{\sum_{l \in \Gamma_{mv \to com}} d_l}$
      Calculate E_{mv} via Eq. 4
    end if
    $JE_{mv} = E_{mv}$
  end for
end for
for $v \in \mathbb{V}$ **do**
  for $m \in \mathbb{V}$ **do**
    if $m \neq v$ **then**
      Calculate $I(m; v)$ via Eq. 7
    else
      $I(m; v) = E_v$
    end if
    $MI_{m;v} = I(m; v)$
  end for
end for

4.2 Graph Topology Optimization

- **Step 1**: When given a graph data $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V} = \{v_1, v_2, \cdots, v_n\}, \mathbb{E} = \{e_1, e_2, \cdots, e_m\}$, we construct the node mutual information matrix MI in the graph $\mathbb{G}$, where $MI \in \mathbb{R}^{N \times N}$ with elements $MI_{i,j} = I(i, j)$. The algorithmic process for constructing a node mutual information matrix is as shown in Algorithm 1.
- **Step 2**: Next, for any node v_i, we select k-nearest neighbors as $N_{MI}(v_i)$ by MI. Then we consider the edges related to the nodes in the original graph and sort the original edges related to the node v_i and discard edges between the node v_i and the M-furthest nodes by node mutual information. At this time, the set of dropped edges is recorded as $N_d(v_i)$.
- **Step 3**: After that, we mark the union of all virtual edges added by all nodes as N_{MI}. Let $\tilde{\mathbb{E}} = N_{MI} \cup \mathbb{E}$ construct an adjacency matrix $\tilde{A}$, the set of original edges deleted by all nodes is defined as N_D and the adjacency matrix of the edges to be deleted is defined as A_{Drop}. Thus, the adjacency matrix of the final generated graph is denoted as:

$$A_{input} = \tilde{A} - A_{Drop}. \tag{8}$$

The algorithmic process for updating the adjacency matrix of the final generated graph is listed in Algorithm 2.

4.3 Node Feature Learning

In order to retain the advantages of GCN [11], we utilize the same form of feature learning as GCN, the calculation formula of MI-GCN is defined as:

$$Z^{(l+1)} = \sigma(\hat{A}_{input} Z^{(l)} W^{(l)}), \tag{9}$$

where $Z^{(l+1)} = \{z_1^{(l+1)}, z_2^{(l+1)}, \cdots, z_i^{(l+1)}, \cdots, z_n^{(l+1)}\}$ is the node embedding matrix of the $l+1$-th layer, where $z_i^{(l+1)}$ is the node representation of the $l+1$-th layer for node v_i; σ represents a nonlinear activation function, such as ReLU; $\hat{A}_{input} = \hat{D}^{-\frac{1}{2}}(A_{input} + I)\hat{D}^{-\frac{1}{2}}$ is a renormalized adjacency matrix with self-loop edges, where $\hat{D}$ is defined as the degree matrix corresponding to $A_{input} + I$; $W^{(l)}$ is denoted as a learnable weight matrix.

Algorithm 2 Construct the adjacency matrix (A_{input}) of the final generated graph

Require: $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, $X = \{x_1, x_2, \cdots, x_n\} \in \mathbb{R}^{N \times C}$, A, MI
Ensure: A_{input}
1: **for** $v_i \in \mathbb{V}$ **do**
2: Select k-nearest neighbors to add virtual edges as $N_{MI}(v_i)$ by MI
3: Select the set of deleted original edges as $N_D(v_i)$ by MI
4: **end for**
5: Initialize the set of all nodes added virtual edges $N_{MI} = \emptyset$ and the set of edges to be dropped by all nodes $N_D = \emptyset$
6: **for** $v_i \in \mathbb{V}$ **do**
7: $N_{MI} = N_{MI} \cup N_{MI}(v_i)$
8: $N_D = N_D \cup N_D(v_i)$
9: **end for**
10: Construct an adjacency matrix of dropped edges A_{Drop} from N_D
11: Construct an adjacency matrix of virtual edges $\tilde{A}$ from $\tilde{\mathbb{E}} = N_{MI} \cup \mathbb{E}$
12: $A_{input} = \tilde{A} - A_{Drop}$

4.4 Node Classification Loss Function

The node classification loss function is the cross entropy loss function over all training nodes, which can be defined as:

$$\mathcal{L}_{loss} = -\sum_{l \in L} \sum_{i=1}^{C} y_{il} \ln \hat{y}_{il}, \tag{10}$$

where L is the training set, $Y_l = [y_{il}] \in \mathbb{R}^{n \times C}$ is the real label, $\forall l \in L$, and the predicted label is $\hat{Y} = [\hat{y}_{il}] \in \mathbb{R}^{n \times C}$, where $\hat{Y} = softmax(\hat{Z})$, $\hat{Z}$ is denoted as the output of the last layer.

5 PERFORMANCE EVALUATION

In this section, we use the semi-supervised node classification task to test the performance of the proposed MI-GCN model. We first introduce the commonly used datasets. Then we list the baselines for comparison and some implementation details. Finally, we evaluate MI-GCN by dividing the first three datasets as fixed data splits and the last three datasets as random data splits.

5.1 Datasets

Extensive experiments are conducted on six widely used real-world datasets, which are listed as follows.

- **Cora, Citeseer and Pubmed** [11]: These datasets belong to research paper citation networks, where nodes represent publications, and edges represent citation links. All nodes are divided into 7, 6, and 3 classes.
- **ACM** [27]: This network is obtained from the ACM dataset, where nodes represent papers. If the same authors exist between the papers, then an edge is established between the nodes. The characteristics of the nodes are composed of the keywords of the papers, and all papers (nodes) are divided into 3 classes (database, wireless communication and data mining).
- **BlogCatalog** [14]: It is a social network from the website that represents the social relationship of bloggers. The characteristics of the nodes are composed of the user's profile information on the website and the tags represent the classes of user information. All nodes are divided into 6 classes.
- **Flickr** [14]: It is a picture and video social network site. Users share and communicate through pictures and short videos, where nodes represent users, and edges represent their social relationships. All nodes are divided into 9 classes.

In addition, other details of the above datasets are summarized in Table 2.

Table 2: Statistical characteristics of each dataset.

Datasets	Classes	Features	Nodes	Edges
Cora	7	1433	2708	5429
Citeseer	6	3703	3327	44338
Pubmed	3	500	19717	81894
ACM	3	1870	3025	13128
BlogCatalog	6	8189	5196	171743
Flickr	9	12047	7575	239738

5.2 Baselines

To evaluate the effectiveness of the proposed MI-GCN, we adopt six baselines for comparison, which are listed as follows:

- **DeepWalk** [19] is a method for learning the latent representations of vertices in the network. DeepWalk uses the local information obtained from truncated random walks, and learns latent representations by treating the walks as sentence equivalents.
- **GCN** [11] is based on a hidden layer representation that encodes local graph structure and node features based on a CNN that operates directly on the graph.
- **ResGCN** [10] is an effective variant of GCN and residual framework, which can extract the node representation in the graph network more effectively.
- **JK-Net** [30] uses dense connections to leverage different neighbors of nodes to learn better representations.
- **Dropedge-GCN** [24] improves model performance by removing a fixed proportion of edges at each training time.
- **AM-GCN** [28] proposed an adaptive multi-channel graph convolutional network to simultaneously extract node embeddings from node features, topology, and their combinations.

Among them, DeepWalk is the traditional network node embedding algorithm, and GCN, ResGCN, JKNet are the popular GNNs.

5.3 Experimental Setting

In order to evaluate our model more comprehensively, all baselines are initialized with the parameters suggested in the original papers, and then the parameters are further adjusted to obtain better performance. We set the number of GCN, ResGCN ,Dropedge-GCN,AM-GCN and MI-GCN layers to 2, the depth of JK-Net to 4, and the hidden layer units to 128.

For fixed data splits, the baselines set the learning rate to 0.005, the dropout rate to 0.5 and the L2 regularization of GCN, ResGCN and JK-Net are set to 5e-4, 5e-5 and 5e-4, respectively. The parameter details of the proposed MI-GCN for fixed data splits are summarized in Table 3.

Table 3: Parameter settings for fixed data splits (Cora, Citeseer and Pubmed), where K represents the number of virtual edges added by each node based on node mutual information, and M is the number of edges that each node deletes edges in the original graph.

Dataset	Hidden size	Learning rate	Dropout	Weight-decay	K	M
Cora	128	0.001	0.3	1e-4	1	0
Citeseer	128	0.005	0.5	5e-4	2	0
Pubmed	128	0.005	0.5	1e-3	4	2

Moreover, we will select the training set (20, 40, 60 labeled nodes per class) for random data splits (ACM, BlogCatalog and Flickr) and choose 1,000 nodes as the test set, the learning rate is tuned from $\{0.01, 0.005\}$ in baselines, the dropout is set to 0.5, 0.5 and 0.3, respectively, and the L2 regularization is set to {5e-3, 5e-4}, respectively. Each model is trained for 400 epochs, we set early stopping to 400 to avoid overfitting. The parameter details of MI-GCN for random data splits are summarized in Table 4.

Table 4: Parameter settings for random data splits (ACM, BlogCatalog and Flickr).

Dataset	L/C	Hidden size	Learning rate	Dropout	Weight-decay	K	M
ACM	20	128	0.01	0.5	5e-4	4	0
	40	128	0.01	0.4	5e-4	4	1
	60	128	0.005	0.3	5e-4	4	1
BlogCatalog	20	128	0.01	0.5	5e-4	0	1
	40	128	0.01	0.3	5e-4	0	1
	60	128	0.005	0.3	5e-4	3	0
Fickr	20	128	0.01	0.3	5e-4	2	1
	40	128	0.01	0.5	5e-4	0	1
	60	128	0.01	0.5	5e-4	5	3

5.4 Fixed Data Splits

In the first part of the experiment, we use the fixed data splits from [11] as they are open data splits in the literature. Fixed data split is the most commonly used data splitting method, which has been widely adopted as standard to test node classification performance [11, 24].

Table 5: Classification accuracy on fixed data splits (%). (Bold: best)

Dataset	DeepWalk	GCN	ResGCN	JKNet	Dropedge-GCN	AM-GCN	MI-GCN
Cora	64.27	80.39	81.42	80.33	80.87	76.91	**82.17**
Citeseer	43.29	71.40	69.67	71.30	70.95	69.44	**71.98**
Pubmed	64.32	79.83	79.77	78.98	78.56	79.58	**80.96**

The experimental results of fixed data splits are reported in Table 5. It is obvious that the performance of our MI-GCN surpasses all comparison baseline methods. Specifically, the accuracy of Cora and Pubmed has been greatly improved. In general, the classification accuracy of GNNs is much higher than that of the traditional node embedding algorithm (DeepWalk). Since DeepWalk only uses the local structure information of the node to generate the node embedding and ignores the node function information. Compared with current popular GNNs, our MI-GCN can further enhance the aggregation and transmission of node structural information, and achieve a performance improvement of $0.58\% \sim 1.78\%$ on the fixed data splits.

Moreover, the F1 score is also a significant criterion for measuring the performance in multi-classification tasks. Fig. 4 shows F1 scores of our MI-GCN and baselines on fixed data splits. We can conclude that except for Citeseer, the classification performance of Mi-GCN is much better than that of baselines. The model proposed on Citeseer is also very close to the best F1 score of baselines.

Meanwhile, we report the node classification accuracy for different parameters $K \in [0, 5]$ and $M \in [0, 5]$. The experimental results are depicted in Fig. 3. On different datasets, the impact of different parameters on accuracy varies greatly. This is mainly due to the different distribution of nodes and edges on different datasets. However, the change of the parameter K (the number of original edges deleted) is more sensitive to the accuracy of the results. Compared with GCN, the accuracy of MI-GCN on Cora, Citeseer, Pubmed floats at $-70.79\% \sim 1.78\%$, $-53.3\% \sim 0.19\%$ and $-7.93\% \sim 1.13\%$, respectively.

5.5 Random Data Splits

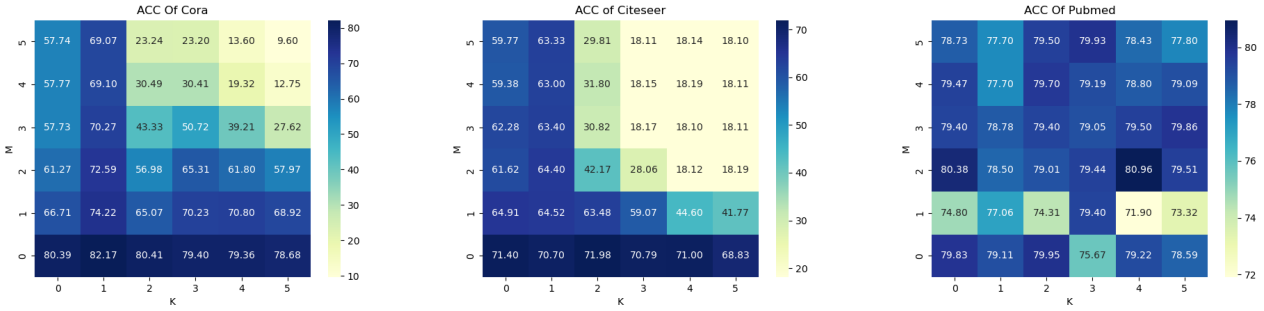
The purpose of semi-supervised learning obtains better experimental results with fewer labeled data. Therefore, in order to better verify that the proposed model has better performance, we use different types of datasets to avoid the unity of the experiments and the method of randomly dividing the training set. To be specific, we randomly select 20, 40, and 60 nodes in the peer class of datasets (ACM, BlogCatalog and Flickr) as the training set, and randomly select 1,000 nodes as the test set.

Table 6: Classification accuracy on random data splits (%), where L/C means the number of labeled nodes per class. (Bold: best)

Dataset	L/C	DeepWalk	GCN	ResGCN	JKNet	Dropedge-GCN	AM-GCN	MI-GCN
ACM	20	62.29	86.73	86.12	79.05	84.46	88.56	**90.49**
	40	63.45	88.06	83.31	84.10	87.06	89.64	**90.29**
	60	67.49	88.87	87.32	88.62	89.05	90.47	**92.91**
BlogCatalog	20	38.77	73.37	72.37	71.70	74.61	74.54	**76.24**
	40	50.18	74.27	75.16	75.41	74.86	73.05	**77.99**
	60	52.94	75.67	76.57	77.01	73.71	77.28	**78.37**
Flickr	20	24.78	48.75	50.76	50.80	51.54	53.22	**53.98**
	40	28.61	53.91	55.78	58.12	58.07	58.74	**59.72**
	60	30.10	58.32	57.40	68.09	**70.85**	67.28	62.23

It can be observed from Table 6 that compared with all baselines, the proposed MI-GCN has achieved better performance in different label rates, especially in ACM, the increase can be up to 4.04%.

Fig. 5 shows the F1 scores of our proposed model and baselines under the random data splits. Our MI-GCN significantly surpasses the current baselines. Compared with GCN, MI-GCN improves node classification accuracy more significantly, which means that MI-GCN introduces a better and more suitable graph structure for the label to monitor feature propagation and node representation learning. The baselines are performed in the original topology graph, but the node information with similar first-order structure information cannot be effectively aggregated, so the result is lower accuracy and F1 score compared with our MI-GCN.



(a) Cora (b) Citeseer (c) Pubmed

Figure 3: The influence of different parameter settings on experimental accuracy (%).

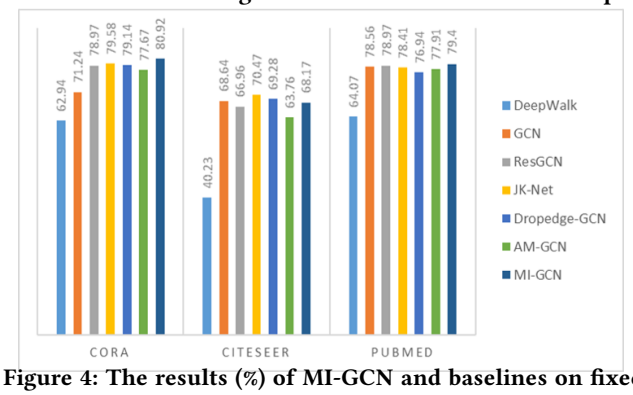


Figure 4: The results (%) of MI-GCN and baselines on fixed dataset splits.

Similarly, we report the impact of different parameters K and M on the accuracy of random data splits as depicted in Fig. 6. Due to the different data types, the results vary greatly. For different partitions of the same datasets, the sensitivity of parameter adjustment is also different. Overall, the adjustment of parameters has the greatest impact on the results of the ACM. Compared to GCN, the experimental results of MI-GCN on ACM, BlogCatalog and Flickr fluctuate between $-28.41\% \sim 3.76\%$, $-8.05\% \sim 2.7\%$ and $-7.05\% \sim 3.91\%$, respectively.

6 CONCLUSION

This paper designed a novel GCN based on node mutual information to solve the challenges of insufficient ability to acquire local structural features of nodes and insufficient performance of node importance in current popular GNNs. We studied how to obtain the most relevant information from the topology and node features. Starting from the significance of the importance of nodes in a complex network, we define and supplement node joint information entropy and node mutual information theory based on the node structural information. Furthermore, we apply node mutual information to GNNs to enhance the ability of structure information extraction and fusion. Extensive experiments have proved that the node classification performance in real-world datasets is stronger than several popular GNNs.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (62071327).

REFERENCES

[1] Lei Bai, Lina Yao, Salil S. Kanhere, Xianzhi Wang, and Quan Z. Sheng. 2019. STG2seq: Spatial-Temporal Graph to Sequence Model for Multi-Step Passenger Demand Forecasting. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence* (Macao, China) *(IJCAI'19)*. AAAI Press, 1981–1987.

[2] Duan-Bing Chen, Rui Xiao, An Zeng, and Yi-Cheng Zhang. 2014. Path diversity improves the identification of influential spreaders. *EPL (Europhysics Letters)* 104, 6 (2014), 68006.

[3] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. 2019. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 257–266.

[4] Reuven Cohen, Shlomo Havlin, and Daniel Ben-Avraham. 2003. Efficient immunization strategies for computer networks and populations. *Physical review letters* 91, 24 (2003), 247901.

[5] Thomas M Cover, Joy A Thomas, et al. 1991. Entropy, relative entropy and mutual information. *Elements of information theory* 2, 1 (1991), 12–13.

[6] Emiliano D'agostino, Frederik Maes, Dirk Vandermeulen, and Paul Suetens. 2003. A viscous fluid model for multimodal non-rigid image registration using mutual information. *Medical image analysis* 7, 4 (2003), 565–575.

[7] Chi Thang Duong, Thanh Dat Hoang, Ha The Hien Dang, Quoc Viet Hung Nguyen, and Karl Aberer. 2019. On node features for graph neural networks. *arXiv preprint arXiv:1911.08795* (2019).

[8] Péter Érdi, Kinga Makovi, Zoltán Somogyvári, Katherine Strandburg, Jan Tobochnik, Péter Volf, and László Zalányi. 2013. Prediction of emerging technologies based on analysis of the US patent citation network. *Scientometrics* 95, 1 (2013), 225–242.

[9] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *The World Wide Web Conference*. 417–426.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[11] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

[12] Mengzhang Li and Zhanxing Zhu. 2021. Spatial-temporal fusion graph neural networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 4189–4196.

[13] Guimei Liu, Limsoon Wong, and Hon Nian Chua. 2009. Complex discovery from weighted PPI networks. *Bioinformatics* 25, 15 (2009), 1891–1897.

[14] Zaiqiao Meng, Shangsong Liang, Hongyan Bao, and Xiangliang Zhang. 2019. Co-embedding attributed networks. In *Proceedings of the twelfth ACM international conference on web search and data mining*. 393–401.

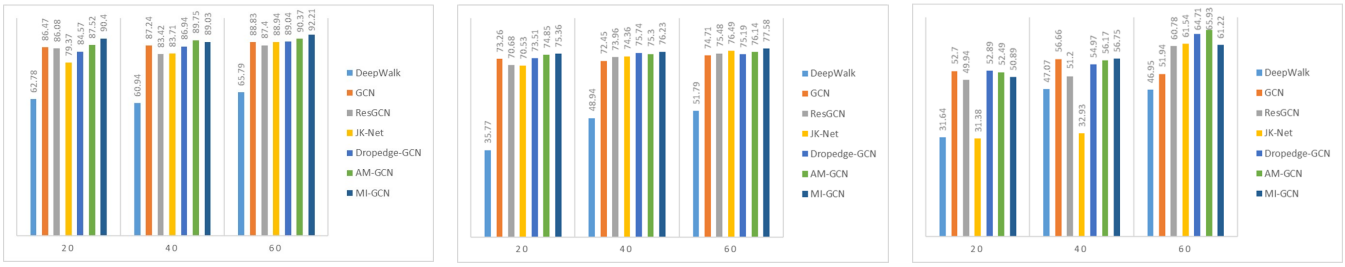
[15] Seth A Myers, Aneesh Sharma, Pankaj Gupta, and Jimmy Lin. 2014. Information network or social network? The structure of the Twitter follow graph. In *Proceedings of the 23rd International Conference on World Wide Web*. 493–498.

[16] Tingyuan Nie, Zheng Guo, Kun Zhao, and Zhe-Ming Lu. 2016. Using mapping entropy to identify node centrality in complex networks. *Physica A: Statistical Mechanics and its Applications* 453 (2016), 290–297.

[17] Hoang Nt and Takanori Maehara. 2019. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550* (2019).

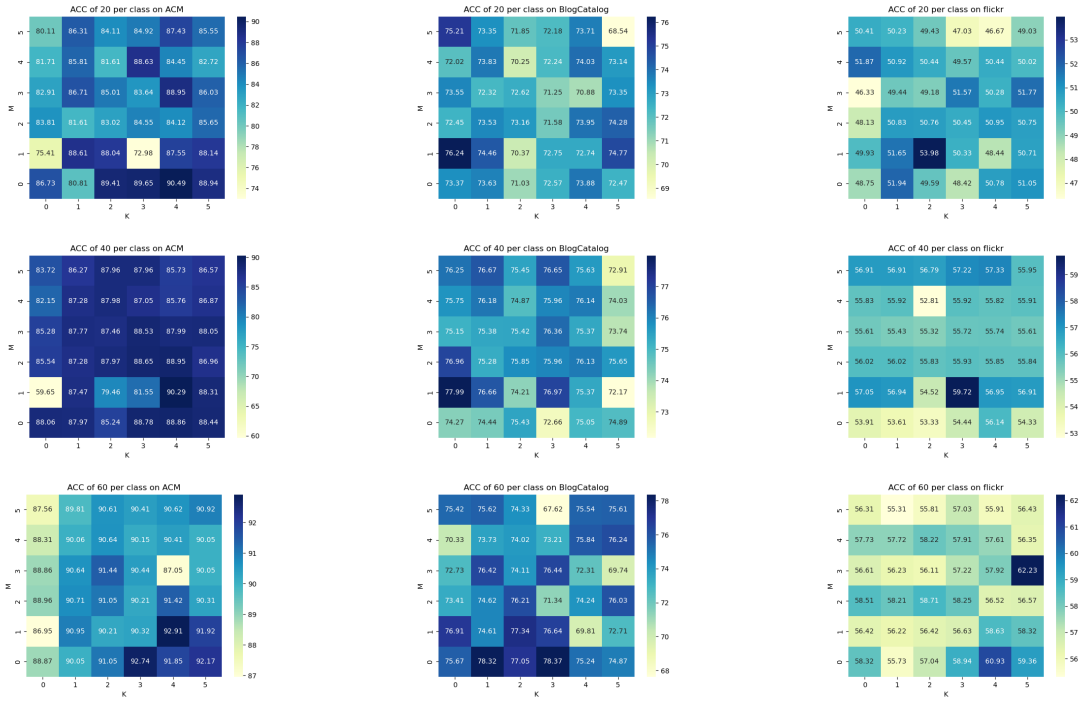
[18] JA Núñez, PM Cincotta, and FC Wachlin. 1996. Information entropy. In *Chaos in Gravitational N-Body Systems*. Springer, 43–53.

[19] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international*



(a) ACM (b) BlogCatalog (c) Fickr

Figure 5: The results (%) of MI-GCN and baselines on random data splits.



(a) ACM (b) BlogCatalog (c) Fickr

Figure 6: The impact of different parameter settings on accuracy (%) with 20, 40 and 60 labeled nodes in the peer class of datasets.

conference on Knowledge discovery and data mining. 701–710.

[20] Josien PW Pluim, JB Antoine Maintz, and Max A Viergever. 2003. Mutual-information-based registration of medical images: a survey. *IEEE transactions on medical imaging* 22, 8 (2003), 986–1004.

[21] Tong Qiao, Wei Shan, and Chang Zhou. 2017. How to identify the most powerful node in complex networks? A novel entropy centrality approach. *Entropy* 19, 11 (2017), 614.

[22] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 385–394.

[23] Jaume Rigau, Miquel Feixas, Mateu Sbert, Anton Bardera, and Imma Boada. 2004. Medical image segmentation based on mutual information maximization. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 135–142.

[24] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. 2020. DropEdge: Towards Deep Graph Convolutional Networks on Node Classification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

[25] Philippe Thévenaz and Michael Unser. 1996. A pyramid approach to sub-pixel image fusion based on mutual information. In *Proceedings of 3rd IEEE International Conference on Image Processing*, Vol. 1. IEEE, 265–268.

[26] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[27] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous Graph Attention Network. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 2022–2032. https://doi.org/10.1145/3308558.3313562

[28] Xiao Wang, Meiqi Zhu, Deyu Bo, Peng Cui, Chuan Shi, and Jian Pei. 2020. Am-gcn: Adaptive multi-channel graph convolutional networks. In *Proceedings of the 26th ACM SIGKDD International conference on knowledge discovery & data mining*. 1243–1253.

[29] Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, and Xueqi Cheng. 2019. Graph Wavelet Neural Network. In *International Conference on Learning Representations*. https://openreview.net/forum?id=H1ewdiR5tQ

[30] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*. PMLR, 5453–5462.

[31] Jiaxuan You, Rex Ying, and Jure Leskovec. 2019. Position-aware graph neural networks. In *International Conference on Machine Learning*. PMLR, 7134–7143.