Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 19(12): 12677–12692.
DOI: 10.3934/mbe.2022592
Received: 03 July 2022
Revised: 10 August 2022
Accepted: 24 August 2022
Published: 31 August 2022

Research article

A safe semi-supervised graph convolution network

Zhi Yang[1,2], Yadong Yan[1], Haitao Gan[1,2,*], Jing Zhao[2] and Zhiwei Ye[1]

[1] School of Computer Science, Hubei University of Technology, Wuhan 430068, China

[2] State Key Laboratory of Biocatalysis and Enzyme Engineering, School of Life Sciences, Hubei University, Wuhan 430062, China

* **Correspondence:** Email: htgan01@hbut.edu.cn.

Abstract: In the semi-supervised learning field, Graph Convolution Network (GCN), as a variant model of GNN, has achieved promising results for non-Euclidean data by introducing convolution into GNN. However, GCN and its variant models fail to safely use the information of risk unlabeled data, which will degrade the performance of semi-supervised learning. Therefore, we propose a Safe GCN framework (Safe-GCN) to improve the learning performance. In the Safe-GCN, we design an iterative process to label the unlabeled data. In each iteration, a GCN and its supervised version (S-GCN) are learned to find the unlabeled data with high confidence. The high-confidence unlabeled data and their pseudo labels are then added to the label set. Finally, both added unlabeled data and labeled ones are used to train a S-GCN which can achieve the safe exploration of the risk unlabeled data and enable safe use of large numbers of unlabeled data. The performance of Safe-GCN is evaluated on three well-known citation network datasets and the obtained results demonstrate the effectiveness of the proposed framework over several graph-based semi-supervised learning methods.

Keywords: semi-supervised learning; data expansion; graph convolution network; self-training

1. Introduction

In recent years, graph-based methods have attracted more and more attention from researchers. The reason is twofold. First, in the real-world problem, there exists a large amount of non-Euclidean data, such as recommender systems [1], proteins-proteins network [2–4], etc. Unlike the Euclidean data, non-Euclidean data have an irregular data structure, these data can be expressed by graph for the powerful ability of graph. Second, the geometric structure of data can be embedded by graph analysis methods, thereby helping the model to improve its recognition ability. In general, graph-based methods extend the application scenarios of existing machine learning methods to a certain extent.

Nowadays, graph neural networks (GNNs) have been widely used in machine learning for their convincing performance and high interpretability [5]. Due to the great success of CNNs in machine learning, a large number of researchers have tried to extend convolution operations from the image domain to the graph domain [6–9]. As for graph-based semi-supervised learning, Kipf & Welling [9] proposed graph convolution network (GCN). GCN used convolution on the graph to extract features and obtained both feature information and graph structure information of the nodes. However, it required the full graph Laplacian operator when training the model, which was very computationally intensive when computing large graphs. Moreover, the embedding of nodes in each layer of GCN was recursively computed by the embedding of all neighboring nodes in the previous layer. This made the receptive field of the nodes grow exponentially with the number of layers, which could be very time-consuming when computing the gradient. GraphSAGE [10] replaced the graph Laplacian function of GCN with a learnable aggregation function and used neighbor sampling to reduce the growth of the receptive domain, making it applicable to inductive learning. FastGCN [11] interpreted the graph convolution as an integral transform of the embedding function under probability measures and used important sampling to reduce the computational time while providing a comparable level of computational accuracy. Simple Graph Convolution (SGC) [12] reduced the additional complexity by successively removing the nonlinearity in the GCN layers and collapsing the resulting function into a linear transformation. The same computational speedup could be achieved without negatively affecting the classification accuracy. Different from GCN, Topology Adaptive Graph Convolutional Network (TAGCN) [13] used K graph convolution kernels at each layer to extract local features of different sizes, following the example of CNNs, to avoid the previous drawback of approximating the convolution kernels without extracting complete and sufficient graph information, and to improve the representational power of the model. Graph Attention Network (GAT) [14] made use of hidden self-attentive layers by stacking the nodes in each layer so that it can focus on the features of the neighborhood and assign different weights to different nodes in the neighborhood, making it possible to achieve advanced results without the need to know the graph structure in advance. It addressed the problem that GCN needs to have prior knowledge of unknown data and the usage scenarios are mostly in dealing with static graphs. SuperGAT [15] learned more appropriate attention weights when distinguishing misconnected neighbors by a self-supervised task. Moreover, GCN is widely used in other field tasks. For instance, Zhu et al. [16] introduced a graph convolutional network with conditional feature aggregation and proposed a common-centric localization (CCL) network for few-shot common-localization task. Many different improved versions have been proposed by researchers since then [17–19], and all have achieved promising results. Whereas, GCN and its variant models directly use the feature information of all unlabeled data during training without evaluating the risk. This makes the model perform poorly on datasets with noise unlabeled data.

On the other hand, GCN and its variant models are mostly built on a semi-supervised learning paradigm [20]. Semi-supervised learning used labeled data to train the model while using unlabeled data to better maintain the intrinsic structural information of the data, allowing the model to achieve promising results. And it is well adapted to problems that contain a small number of labeled data and a large number of unlabeled data. Nevertheless, it is known that there are risky unlabeled data in unlabeled dataset. Risky unlabeled data refers to noisy data that degrade model performance with incorrect information, such as outlier. These risky data can cause errors to propagate during the training of the model and degrade the performance of the model. In some cases, semi-supervised learning performs worse than the corresponding supervised learning, as has been verified in many works [21–

23]. If the risk of unlabeled data cannot be reduced, this will make the model very limited for using in practical scenarios. Therefore, it is necessary to design a safe GCN.

Some existing semi-supervised learning works have studied the idea of selecting high-confidence pseudo-label. Wu et al. [24] proposed the Exploit the Unknown Gradually (EUG) method, which iteratively obtains pseudo-label and updates the model. Unlike most works that use models to predict pseudo-label, EUG utilized the label of the nearest neighbor in the feature space to assign pseudo-label. And the Euclidean distance is used to measure the confidence of the pseudo-label. Wu et al. [25] used the same method in EUG to obtain pseudo-label, and proposed a joint learning method. These data not considered high-confidence are labeled by index and participate in the training of the model by Exclusive Loss. Similar Pseudo Label Exploitation (SimPLE) [26] utilized the label guessing technique, both labeled and unlabeled data are augmented with weak and strong augmentation. The pseudo-label is obtained by the mean of the class prediction distribution after augmented, and then the weak augmented unlabeled data are sharpened for the model prediction. The loss terms are optimized according to the augmentation data and pseudo-label. The above method only considers a single semi-supervised classifier, ignoring the pseudo-label information obtained by the supervised classifier, which is also important for pseudo-label. Multiple knowledge Representation (MKR) [32] proposed a framework that learned from different levels of abstraction, different sources, and different perspectives to improve model performance through multi-feature aggregation, but it did not evaluate the risk of unlabeled data.

Some recent works have used the self-training method [27–30] to select the unlabeled data and expanded the labeled dataset by the high-confidence unlabeled data, which somewhat alleviates the risk problem of unlabeled data. The main idea of Self-training is using the labeled data to train a classifier to label the unlabeled data [20, 31], then select some high-confidence data to expand the labeled dataset. This process is carried out iteratively until convergence. How to design the Self-training method to select reasonable unlabeled data is an important challenge. Most works are only based on the highest soft-max output [28, 29], which is always insufficient to measure the confidence of the data.

In this paper, we propose a safe GCN framework (Safe-GCN). The proposed model is implemented in three stages. First, S-GCN and GCN classifiers are trained to obtain the pseudo-label of unlabeled data. Second, the outputs of S-GCN and GCN are compared. The unlabeled data with high- confidence are selected by a confidence filtering condition. Then labeled dataset is expanded in a balanced way by high-confidence unlabeled data. Finally, the expanded labeled dataset is used to train the S-GCN. Hence, our proposed Safe-GCN makes better use of supervised and semi-supervised information, and has the opportunity to achieve model security by reducing the negative impact of risky unlabeled data. GCN and its variant models directly use the feature information of all unlabeled data during training without evaluating the risk. This makes models perform poorly on datasets with noisy unlabeled data. Compared with existing graph convolutional networks, our model enables GCN to be safely exploited on datasets with large amounts of unlabeled data. And Safe-GCN improves the noise immunity of GCN, which allows GCN to be applied in more real-world application scenarios.

We conducted experiments on three publicly available citation datasets. The results demonstrate that the classification performance of the proposed model outperforms most existing graph-based models. The main contributions and advantages of this paper compared to related works are summarized as follows:

- We proposed a safe semi-supervised graph convolution model that can effectively reduce the adverse effects of risky unlabeled samples. The model can safely utilize a large amount of unlabeled data.
- The proposed model utilizes only the information of the training data during training and does not need to know the graph structure and feature information of the test data. Therefore, the proposed model can be directly applicable to inductive learning.

The rest of the paper is organized as follows. Section 2 describes the background in this paper. Section 3 describes the proposed algorithm. The dataset, experimental configuration and results are described in Section 4. Section 5 gives the conclusion of this paper and discusses future directions.

2. Background

Since GCN and its supervised version are used as classifiers in Safe-GCN, so we discuss the details of GCN in this section. Kipf & Welling [9] proposed a graph convolution network for semi-supervised learning. The main idea is to pass information from each node to its neighbors through information transfer between nodes, and iteratively aggregate the features of the nodes' neighbors through Laplace matrix and convolution on the graph, enabling it to deeply estimate the labels of unlabeled data. The model can be described as following:

$$F = f(X,\ A)$$

where F$\in \mathbb{R}^{n\times d}$ denotes a label matrix that represents the output of the unlabeled data. X is the feature matrix of the dataset and A is the adjacency matrix associated with data.

The propagation law of the layers in GCN is given by:

$$H^{i+1} = \sigma(D^{-\frac{1}{2}}\widetilde{A}D^{-\frac{1}{2}}H^{(i)}W^{(i)})$$

where $\widetilde{A} = A + I$ denotes the A matrix with added self-connections, I denotes the identity matrix, and $\widetilde{D}$ denotes the degree matrix of the $\widetilde{A}$ matrix. $W^{(i)}$ denotes the weight matrix corresponding to the ith layer of the network. $\sigma(\cdot)$ denotes the activation function, it is given by the ReLU. $H^{(0)}$ is X.

Since the GCN model can achieve advanced results with 2–3 layers, a two-layer GCN model has the following form:

$$F = softmax(\breve{A}\sigma\left(\breve{A}W^{(0)}X\right)W^{(1)})$$

where $\breve{A} = D^{-\frac{1}{2}}\widetilde{A}D^{-\frac{1}{2}}$ denotes the regularized Laplace matrix, $W^{(0)}$ is the input-hidden weight matrix, and $W^{(1)}$ is the hidden-output weight matrix.

The softmax activation function converts the output matrix into a probability distribution for each data corresponding to each category by row, i.e., the probability of each data corresponding to all categories sums to 1. Deep neural networks is learned by making the predicted label as close as possible to the ground-truth label. This is achieved by minimizing the cross-entropy loss function that is typically used for classification problems. The cross-entropy function is used as the loss function in the GCN.

3. Our algorithm

As stated in Section 1, the GCN and its variants models use unlabeled data to enhance the performance of the model and achieve promising results. However, mistakes in risky unlabeled data can spread during the training of the model and may degrade the performance of the model, which makes the use of unlabeled data very risky. Considering the safe use of unlabeled data, we propose an enhanced safe GCN model, which is based on the self-training framework. The method to select high-confidence data in the proposed model is illustrated in Figure 1. The model is divided into three stages as shown below:

(1) Pseudo label acquisition: the first stage computes the information encoded in the embedding by S-GCN and GCN to make better use of labeled and unlabeled data. The classification results and model outputs of S-GCN and GCN for unlabeled data are obtained in this stage.

(2) Labeled dataset expansion: the second stage evaluates the unlabeled data and performed data expansion. Most self-training methods use high-confidence unlabeled data to expand the labeled dataset, and they rely on the maximum softmax scores to assess the risk level of unlabeled data [29, 30],which is always not accurate enough. Therefore, we propose a new confidence-based data filtering condition. Meanwhile, the same number of unlabeled data with high confidence for each class is added to the labeled dataset to ensure a balanced distribution of labels. The first and second stages of learning are iteratively performed until the stopping condition is satisfied (i.e., there are no data meeting filtering condition in the unlabeled dataset).

(3) S-GCN classification: Supervised GCN learning is performed using the final expanded dataset to predict the test data and obtain the final results.

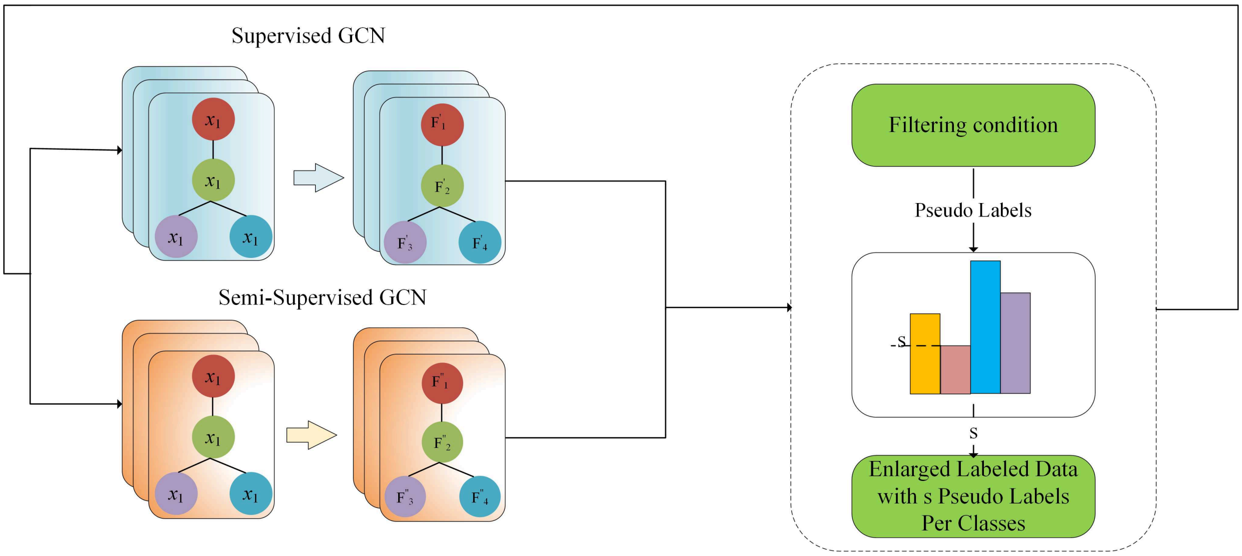

Figure 1. Flowchart of high-confidence data expansion in Safe-GCN.

3.1. *Pseudo-label acquisition*

This stage is the fundamental module that obtained pseudo label of unlabeled data by S-GCN and GCN.

The feature matrix of the dataset is defined as $X = [x_1, \ldots, x_l, \ldots, x_n]$, $X \in \mathbb{R}^{n\times d}$, n is the number of data, d is the number of feature dimensions of the data. l denotes the total number of initially labeled data. The number of labeled data increases with the iterations, and the details are given in the following sections. The feature matrix of the initial labeled data is defined as: $X^{l(0)} = [x_1, \ldots, x_l]$

with the corresponding initial label set $Y^{l(0)} = [y_1, \dots, y_l]^T$, each labeled data x_p will have a label $y_p \in \{1, \dots, c\}$ with the number of classes c. And the labeled dataset after the *kth* iteration is $X^{l(k)}$,

The network structure of S-GCN is the same as the GCN, with the difference: 1) the adjacency matrix and degree matrix are constructed using labeled data. 2) only labeled data are involved in model training. The model output of the *kth* iteration is expressed as follow:

$$F'^{(k)} = f_{S-GCN}(X^{l(k)}, A'^{(k)}, Y^{l(k)})$$

$A'^{(k)} \in \mathbb{R}^{l \times l}$ denotes the adjacency matrix of the S-GCN after the *kth* iteration. The model output of the *kth* iteration of the GCN is given by:

$$F''^{(k)} = f_{GCN}(X, A'', Y^{l(k)})$$

where $A'' \in \mathbb{R}^{(l+u)\times(l+u)}$ denotes the adjacency matrix of the GCN.

Formally the training dataset can be represented by a labeling function with the goal of learning the labeling functions $L(F'^{(k)}), L(F''^{(k)})$ in S-GCN and GCN.

3.2. Labeled dataset expansion

Once the data have been processed by the softmax layer, we are interested in the largest value in $\mathbf{F_i'^{(k)}}$, $\mathbf{F_i''^{(k)}}$ that is associated with the most likely class. Therefore, we consider a function $m(\cdot)$ that returns the largest element in the vector. The function can be defined as: $m(\cdot) = max(\cdot)$, $m(\mathbf{F_i'^{(k)}})$ returns the largest element in $\mathbf{F_i'^{(k)}}$. $\mathbf{F_i'^{(k)}}$, $\mathbf{F_i''^{(k)}}$ denotes the output vector of the ith data in S-GCN and GCN, respectively, both being c-dimensional vector.

However, the maximum value of the output vector of the softmax layer is not sufficient to determine the security of data. To this end, we propose a confidence-based filtering condition for unlabeled data: (1) the prediction results of S-GCN and GCN for the unlabeled data are the same. (2) the maximum value of the output vector of GCN is greater than or equal to the S-GCN and greater than the confidence threshold α. We treat data that satisfy the filtering condition as high confidence ones.

Labels of data that meet the conditions form a candidate label set. The histogram is used to count the alternative label set. In order to balance the distribution of label set, the number of data per class is counted in a histogram. Formally, a function $hgram(\cdot)$ is defined to statistically the classes in the label set and the number of labels in each class. $s^{(k)}$ denotes the number of labels of the least labeled classes in the candidate label set for the *kth* iteration. If there is a class that is not present in the candidate label set, the class label is not updated.

3.3. Supervised GCN classification

GCN using a semi-supervised learning framework for training has achieved promising results. The proposed model is based on the self-training learning framework, and we improve the performance of the model by utilizing the pseudo-label information of unlabeled data. Moreover, traditional GCN is mainly used for semi-supervised learning but can also be used for supervised learning. Supervised GCN is implemented by training the model using only the features and graph structure of the labeled data. We select high-confidence unlabeled data, and the remaining unlabeled data are considered as risky data. Utilizing risky unlabeled data can degrade the performance of the model. Therefore, we use supervised GCN for classification.

The first and second stages are iteratively applied until there are no more unlabeled data satisfying the filtering condition. The labeled dataset after the completion of the iteration is the final labeled dataset. The third stage uses S-GCN model to learn the expanded dataset and obtain the final classification result, i.e., the classification result of Safe-GCN.

The three stages of the proposed method are detailed and formally defined in the following subsections. An overview of the proposed method is given in Algorithm 1. Lines 1 and 2 of the algorithm table train a GCN and an S-GCN, respectively, and lines 3 to 9 select high-confidence data by filtering conditions. Lines 10 to 15 add the high-confidence data to the labeled dataset in a balanced way.

Algorithm 1 Safe-GCN

```
Input: Feature matrix X, labeled data adjacency matrix $A'^{(0)}$, train data adjacency matrix $A''$, initial
    labeled data $X^{l(0)}$ with the corresponding labels $Y^{l(0)}$, initial unlabeled data $X^{u(0)}$, confidence thresh-
    old $\alpha$
Output: S-GCN Embedding matrix $\mathbf{F}'^{(k)}$
 1: for each stage k do
 2:     $\mathbf{F}'^{(k-1)} = f_{S-GCN}(X^{l(k-1)}, A'^{(k-1)}, Y^{l(k-1)})$
 3:     $\mathbf{F}''^{(k-1)} = f_{GCN}(X, A'', Y^{l(k-1)})$
 4:     for all $x_i \in X^{u(k-1)}$ do
 5:         if $m(\mathbf{F_i''^{(k-1)}}) \geq m(\mathbf{F_i'^{(k-1)}}) \geq \alpha$ and $L(\mathbf{F_i'^{(k-1)}}) = L(\mathbf{F_i''^{(k-1)}})$ then
 6:             $y_i = L(\mathbf{F_i''^{(k-1)}})$
 7:             $\ddot{Y} = \ddot{Y} \cup y_i$, $\ddot{Y}$ is the candidate label set.
 8:         end if
 9:     end for
10:     $s^{(k-1)} = min(hgram(\ddot{Y}))$
11:     for each class in $\ddot{Y}$ do
12:         Update $Y^{l(k-1)}$ with the top $s^{(k-1)}$ labels.
13:         Add the corresponding data to $X^{l(k-1)}$.
14:         Delete the corresponding data from the Unlabeled dataset $X^{u(k-1)}$.
15:     end for
16:     Clear $\ddot{Y}$
17: end for
18: return $\mathbf{F}'^{(k)}$
```

4. Experimental evaluation

4.1. Citation dataset

The predictive power of the model is evaluated on three citation network datasets: Cora, Citeseer and Pubmed [33]. These datasets have been utilized in many graph-based semi-supervised classification tasks. The division of datasets are shown in Table 1, and a brief introduction of datasets are as follows:

Cora The Cora dataset consists of 2708 scientific publications, each publication is described by a 1433-dimensional word vector with values of 0 and 1, respectively, representing whether corresponding word appears in the paper. Publications of Cora are divided into 7 classes. The division of Cora is following the GCN. The difference is that we use the union of the train set and the validation set as our train set.

Citeseer The Citeseer dataset employs a similar representation to Cora, but the publications are divided into 6 classes and the data are described by a 3703-dimensional word vector. The division of Citeseer is also following the GCN.

Pubmed The dataset includes 19,717 scientific publications on diabetes from the Pubmed database. Publications are divided into three classes and described by a TF/IDF-weighted word vector in a dictionary of 500 unique words.

The adjacency matrix is very important for the performance of graph-based algorithms. Many graph-based algorithms construct the adjacency matrix from the similarity matrix of nodes. The citation network dataset used in this paper constructs the adjacency matrix by whether two nodes (i.e., papers) are cited to each other.

In many practical applications of machine learning, the information of the test data during the training of the model is unknown. Therefore, the three citation datasets are divided differently from those in GCN, the model is trained without using feature information and node information from the test dataset. Since Safe-GCN only uses the training data to train the model, and the test dataset as a new graph structure to compute classification accuracy, our model is directly applicable to inductive learning. For a fair comparison, the methods used for comparison also use the same form of data division (i.e., feature and node information from the test dataset is not used). The initial labeled data is trained using 20 labels per class. The specific division is shown in Table 1.

Table 1. Citation network datasets statistics.

Dataset	Nodes	Labels	Train	Test
Cora	2708	140	1708	1000
Citeseer	3327	120	2327	1000
Pubmed	19,717	60	18,170	1000

4.2. Experimental setup

We compare the proposed model Safe-GCN with some traditional machine learning methods and some state-of-the-art graph-based methods. These methods belong to two categories: (1) traditional machine learning algorithms. (2) graph-based convolution networks.

The traditional machine learning algorithms include multilayer perceptron (MLP) and support vector machine (SVM). The graph-based models include representative semi-supervised graph convolution networks (GCN) [9], Graph Attention Network (GAT) [14], Topology Adaptive Convolutional Network (TAGCN) [13], Predict then Propagate: Graph Neural Networks meet Personalized PageRank [34] and Attention-based Graph Neural Network for Semi-supervised Learning [35].

The implementation of the proposed Safe-GCN and the above methods were made upon the Pytorch framework. The graph-based methods were implemented via Pytorch Geometric (PYG), an extension library for geometric learning based on the Pytorch framework, and the traditional machine learning methods were implemented via the Scikit-learn package.

Adam was used to training all the above graph-based models as optimizers. During the training phase, each model's hyperparameters and network configuration were followed the default benchmark provided in PYG. The learning rate of the models was set to 0.01 and the dropout parameter was defined as 0.5, except for GAT which was 0.6. For a fair comparison, the number of epochs of all models was limited to 200. MLP and SVM followed the default setups in Scikit-learn, the maximum number of iterations were defined as 1000 for MLP to ensure full convergence.

4.3. Method comparison

Table 2. Results of multiple citation network dataset.

Method	Cora	Citeseer	Pubmed
SVM [36]	0.5550	0.5842	0.7143
MLP [37]	0.5270	0.4985	0.6988
S-GCN [9]	0.6120	0.5941	0.6833
GCN [9]	0.7170	0.6512	0.7479
TAGCN [13]	0.6400	0.5409	0.5908
GAT [14]	0.7370	0.6522	0.7589
APPNP [34]	0.7560	0.6600	0.7430
AGNN [35]	0.7540	0.6610	0.7770
Safe-GCN	**0.7630**	**0.6985**	**0.7776**

Table 3. Recognition performance (Mean recognition accuracy ± Standard deviation on multiple citation network dataset over 10 different random splits.

Method	Cora	Citeseer	Pubmed
SVM [36]	0.5483 ± 0.018	0.5495 ± 0.032	0.6874 ± 0.023
MLP [37]	0.4992 ± 0.021	0.4581 ± 0.054	0.6722 ± 0.019
S-GCN [9]	0.6507 ± 0.019	0.5989 ± 0.031	0.7039 ± 0.022
GCN [9]	0.7009 ± 0.017	0.6449 ± 0.022	0.7424 ± 0.018
TAGCN [13]	0.6271 ± 0.029	0.4987 ± 0.035	0.6120 ± 0.060
GAT [14]	0.7160 ± 0.017	0.6506 ± 0.016	0.7425 ± 0.023
APPNP [34]	0.7204 ± 0.026	0.6562 ± 0.018	0.7570 ± 0.022
AGNN [35]	0.7161 ± 0.025	0.6558 ± 0.010	0.7564 ± 0.017
Safe-GCN	**0.7345 ± 0.015**	**0.6845 ± 0.017**	**0.7799 ± 0.020**

Table 2 illustrates the classification rate using different traditional machine learning and graph-based methods for Cora, Citeseer, and Pubmed datasets (following the split in [9]). Table 3 illustrates the average classification rate (together with its standard deviation over the ten random splits). For each table, there are three columns that correspond to three citation datasets.

4.4. Ablation experiment

In safe-gcn, two classifiers are used to obtain high confidence pseudo-label. To verify that combining two classifiers improves the quality of pseudo-label, we conducted ablation experiments using only a single classifier under the same experimental setup. The results are shown in Table 4.

Table 4. Recognition performance: single network and combined two network.

Method	Cora	Citeseer	Pubmed
The split in [9]			
SGCN	0.7520	0.6660	0.7590
GCN	0.6710	0.6620	0.7600
Safe-GCN	**0.7630**	**0.6985**	**0.7776**
Random split			
SGCN	0.6450 ± 0.062	0.6233 ± 0.038	0.7508 ± 0.021
GCN	0.7197 ± 0.029	0.6694 ± 0.036	0.7695 ± 0.020
Safe-GCN	**0.7345 ± 0.015**	**0.6845 ± 0.017**	**0.7799 ± 0.020**

4.5. The effect of the iterative process on model performance

The self-training learning framework uses an iterative approach to expand the labeled samples. To investigate how Safe-GCN improves the performance over the iterations, we recorded the unlabeled data expanded in the first 10 iterations of the model and the performance of Safe-GCN for each iteration in Table 5.

Table 5. Statistic of first 10 iterations in Safe-GCN, the first row of each dataset represents the number of pseudo-label added (correct labels/wrong labels). The second row represents the correct rate of adding pseudo-labels. The third row represents the classification accuracy of safe-gcn at this iteration.

iteration	1	2	3	4	5	6	7	8	9	10
Cora	323/69	177/61	44/26	27/15	13/8	19/2	11/3	18/10	7/5	100/20
	0.8240	0.7437	0.6286	0.6429	0.6190	0.9048	0.7857	0.6429	0.5833	0.8333
	0.7040	0.7310	0.7340	0.7300	0.7290	0.7320	0.7290	0.7460	0.7470	0.7450
Citeseer	523/179	247/113	100/53	84/42	39/21	23/13	5/7	8/4	5/1	16/9
	0.7450	0.6861	0.6410	0.6667	0.6500	0.6389	0.4167	0.6667	0.8333	0.6400
	0.6820	0.6870	0.6890	0.6890	0.6890	0.6840	0.6820	0.6870	0.6890	0.6950
Pubmed	7050/1422	1142/382	180/72	75/39	85/26	13/17	10/2	1/2	1660/296	17/10
	0.8322	0.7493	0.7143	0.6579	0.7658	0.4333	0.8333	0.3333	0.8487	0.6296
	0.7690	0.7590	0.7540	0.7540	0.7550	0.7570	0.7560	0.7550	0.7660	0.7700

4.6. The effect of the number of labeled data

Since the number of labeled data has an effect on the accuracy of the model, it is interesting to study the performance of the model with different numbers of labeled data. The number is increased or decreased from the original labeled dataset. In this section, we adjust the number of initially labeled data to study the performance of the proposed model from different dataset. The basic labeled dataset of Cora, Citeseer, and Pubmed are respectively 140, 120, 60, accounting for 0.2, 0.16, and 0.02% of the total training data, respectively. In Figure 2, the horizontal coordinate indicates the proportion of labeled data to the overall training data, and the vertical coordinate indicates the classification accuracy of the model.

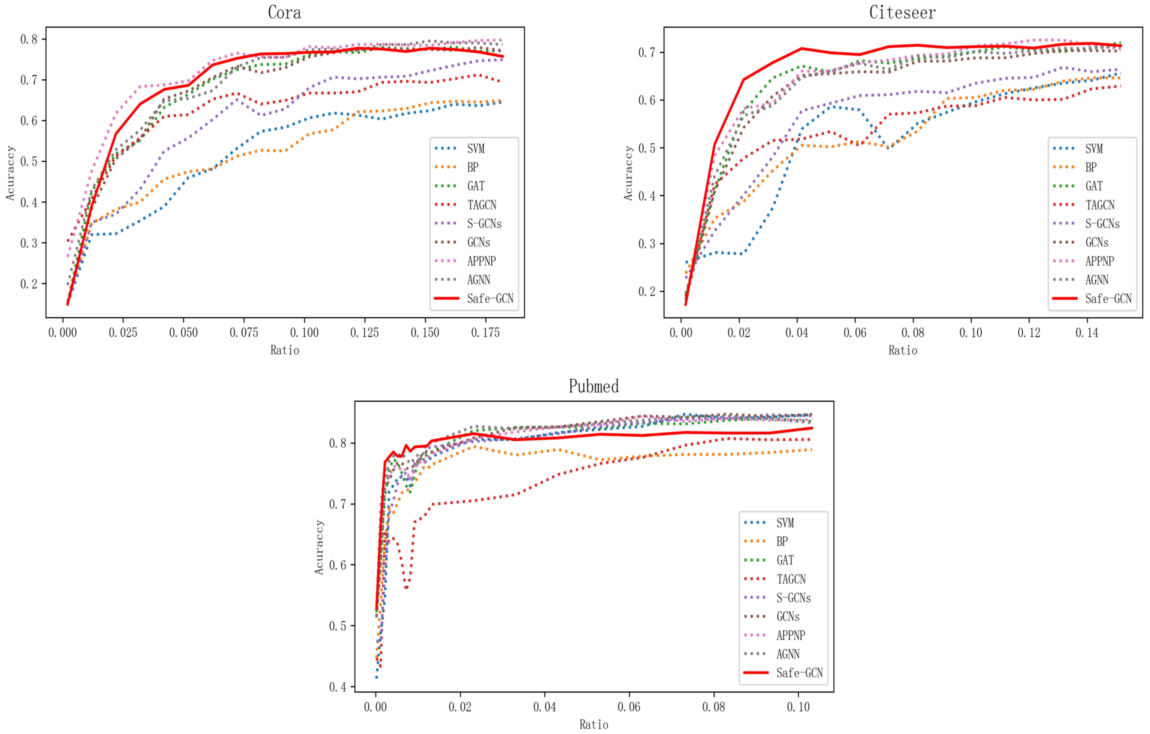

Figure 2. Classification accuracy (%) of the proposed method for different ratios of labeled data (a) Cora dataset. (b) Citeseer dataset. (c) Pubmed dataset.

4.7. *The effect of confidence thresholds*

Hyperparameters are very important in machine learning, which can directly affect the performance of the model. In this paper, α as a hyperparameter denotes the confidence threshold used to determine the data security. Therefore, in order to study the effect of α on the model, we give a set of values to adjust it. The α of Cora, Citeseer were chosen from $[0.2, 0.3, \ldots, 0.9]$ and Pubmed were chosen from $[0.4, 0.5, \ldots, 0.9]$, respectively. The classification accuracy of the model for each of the three citation datasets at different α is illustrated in Figure 3.

4.8. *The effect of $s^{(k)}$*

In this section, we study the effect of $s^{(k)}$ on the model performance. $s^{(k)}$ as a hyperparameter that determines the number of expanded high-confidence data per class in the *kth* iteration. Figure 4 illustrates the classification accuracy of the model at different ranges of $s^{(k)}$.

4.9. *Experimental analysis*

From the results of all previous tables and pictures, we can conclude the following:

(1) From Tables 2 and 3, we can see that our model outperforms the other methods on all three citation datasets. In particular, it can improve more than 3% on all three datasets compared to GCN. We also have different degrees of advancement compared to other methods. This indicates that high confidence data can enhance the predictive power of the model.

(2) The superiority of the proposed model in the case of small labeled data size is obvious as presented in Figure 2, which indicates that the proposed model is applicable to the problem of few labels

and can safely utilize a large number of unlabeled data to satisfy many real-world application scenarios. The advantages of the proposed model are not evident in the case of a large amount of labeled data, which is attributed to the fact that a large amount of labeled data can already describe the distribution of the data adequately.

(3) Figure 3 illustrates that in the Cora and Citeseer, the proposed model is insensitive in the ranges $[0.2, 0.3, \dots, 0.9]$, $[0.4, 0.5, \dots, 0.9]$ for parameter α, respectively. In Pubmed, the proposed model is insensitive to the parameter α and even to the proportion of labeled data in the training data.

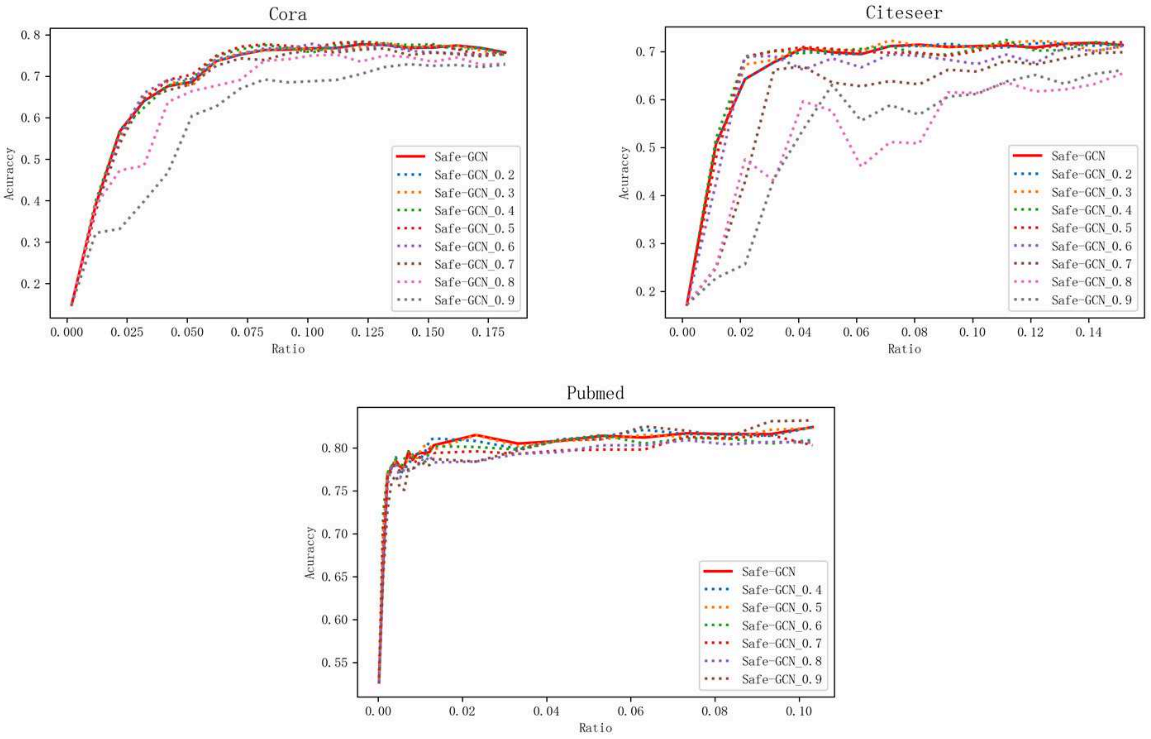

Figure 3. Classification accuracy (%) of the proposed method for different α of labeled data (a) Cora dataset. (b) Citeseer dataset. (c) Pubmed dataset.

(4) The results of the ablation experiments demonstrate that combining two networks can improve the quality of pseudo-label and achieve better classification performance than using only a single network.

(5) Table 5 illustrates how the iterative process improves the predictions. Cora and Citeseer show that the large number of accurate pseudo-labels added at each iteration enhances the performance classifier at each iteration, allowing it to classify unlabeled data more accurately. In contrast, Pubmed adds a large number of accurate pseudo-label at the first iteration, therefore, the performance improvement over the iterations is slow, but the overall classification performance is improving.

(6) Figure 4 shows the classification accuracy of the model for Safe-GCN at different proportions of $s^{(k)}$ settings. The results demonstrate that the model is insensitive to the parameter $s^{(k)}$.

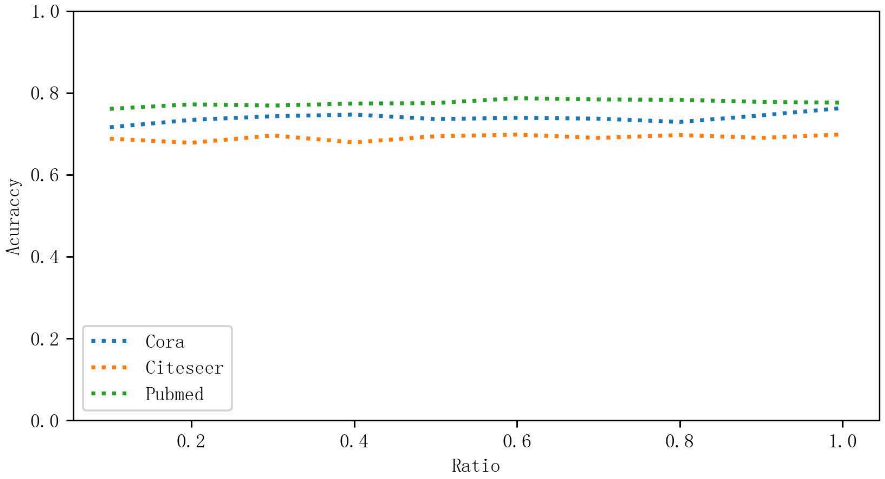

Figure 4. Classification accuracy (%) of the proposed method for the proportions of $s^{(k)}$.

5. Conclusions

We propose a safe GCN framework. The model is based on the self-training framework, which utilizes embedding information of unlabeled data by learning S-GCN classifiers and GCN classifiers. Then obtains high-confidence unlabeled data using a confidence threshold-based data filtering condition, which is balanced to expand the labeled data and reduce the negative impact of risky unlabeled data. At the same time, the model combines supervised and semi-supervised information of data, which improves the security of unlabeled data than using only supervised or semi-supervised information. Therefore, our model can effectively reduce the risk of unlabeled data and safely use a large number of unlabeled data. In addition, our model is applicable to inductive learning, which extends the applicability of the model to some extent.

In the future work, we will focus on the following directions: (1) more detailed risk classification of unlabeled data, and different risk levels of unlabeled data may have different effects on the model. (2) The quality of the model also affects the performance of the model, and methods to assess the quality of the model will be explored. (3) Reducing the time complexity of the model is of importance in realistic application scenarios.

Acknowledgments

The work was supported by Open Funding Project of the State Key Laboratory of Biocatalysis and Enzyme Engineering under grant No. SKLBEE2020020 and SKLBEE2021020, the High-level Talents Fund of Hubei University of Technology under grant No. GCRC2020016, Key Laboratory of Brain Machine Collaborative Intelligence of Zhejiang Province under grant No. 2020E10010-02, and Natural Science Foundation of Hubei Province under grant No. 2021CFB282.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. L. Luo, K. Liu, D. Peng, Y. Ying, X. Zhang, A motif-based graph neural network to reciprocal recommendation for online dating, in *International Conference on Neural Information Processing*, Springer, (2020), 102–114. https://doi.org/10.1007/978-3-030-63833-7_9
2. A. Fout, J. Byrd, B. Shariat, A. Ben-Hur, Protein interface prediction using graph convolutional networks, in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, **30** (2017), 1–10.
3. Y. B. Wang, Z. H. You, S. Yang, H. C. Yi, Z. H. Chen, K. Zheng, A deep learning-based method for drug-target interaction prediction based on long short-term memory neural network, *BMC Med. Inf. Decis. Making*, **20** (2020), 1–9. https://doi.org/10.1186/s12911-020-1052-0
4. X. M. Zhang, L. Liang, L. Liu, M. J. Tang, Graph neural networks and their current applications in bioinformatics, *Front. Genet.*, **12** (2021), 690049. https://doi.org/10.3389/fgene.2021.690049
5. J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, et al., Graph neural networks: A review of methods and applications, *AI Open*, **1** (2020), 57–81. https://doi.org/10.1016/j.aiopen.2021.01.001
6. J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and locally connected networks on graphs, preprint, arXiv:1312.6203.
7. M. Henaff, J. Bruna, Y. LeCun, Deep convolutional networks on graph-structured data, preprint, arXiv:1506.05163.
8. J. Atwood, D. Towsley, Diffusion-convolutional neural networks, in *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, **29** (2016), 1–9.
9. T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, preprint, arXiv:1609.02907.
10. W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, **30** (2017), 1–11.
11. J. Chen, T. Ma, C. Xiao, Fastgcn: fast learning with graph convolutional networks via importance sampling, preprint, arXiv:1801.10247.
12. F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, K. Weinberger, Simplifying graph convolutional networks, in *International Conference on Machine Learning*, (2019), 6861–6871.
13. J. Du, S. Zhang, G. Wu, J. Moura, S. Kar, Topology adaptive graph convolutional networks, preprint, arXiv:1710.10370.
14. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, preprint, arXiv:1710.10903.
15. D. Kim, A. Oh, How to find your friendly neighborhood: Graph attention design with self-supervision, preprint, arXiv:2204.04879.
16. L. Zhu, H. Fan, Y. Luo, M. Xu, Y. Yang, Few-shot common-object reasoning using common-centric localization network, *IEEE Trans. Image Process.*, **30** (2021), 4253–4262. https://doi.org/10.1109/TIP.2021.3070733

17. W. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, C. J. Hsieh, 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*, (2019), 257–266. https://doi.org/10.1145/3292500.3330925

18. H. Pei, B. Wei, K. C. C. Chang, Y. Lei, B. Yang, Geom-gcn: Geometric graph convolutional networks, preprint, arXiv:2002.05287.

19. B. Yu, Y. Lee, K. Sohn, Forecasting road traffic speeds by considering area-wide spatio-temporal dependencies based on a graph convolutional neural network, *Transp. Res. Part C Emerging Technol.*, **114** (2020), 189–204. https://doi.org/10.1016/j.trc.2020.02.013

20. O. Chapelle, B. Scholkopf, A. Zien, Semi-supervised learning [book reviews], *IEEE Trans. Neural Networks*, **20** (2009), 542. https://doi.org/10.1109/TNN.2009.2015974

21. A. Singh, R. Nowak, J. Zhu, Unlabeled data: Now it helps, now it doesn't, in *Advances in Neural Information Processing Systems 21 (NIPS 2008)*, **21** (2008), 1–8.

22. N. V. Chawla, G. Karakoulas, Learning from labeled and unlabeled data: An empirical study across techniques and domains, *J. Artif. Intell. Res.*, **23** (2005), 331–366. https://doi.org/10.1613/jair.1509

23. H. Gan, N. Sang, X. Chen, Semi-supervised kernel minimum squared error based on manifold structure, in *International Symposium on Neural Networks*, Springer, (2013), 265–272. https://doi.org/10.1007/978-3-642-39065-4_33

24. Y. Wu, Y. Lin, X. Dong, Y. Yan, W. Ouyang, Y. Yang, Exploit the unknown gradually: One-shot video-based person re-identification by stepwise learning, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (2018), 5177–5186.

25. Y. Wu, Y. Lin, X. Dong, Y. Yan, W. Bian, Y. Yang, Progressive learning for person re-identification with one example, *IEEE Trans. Image Process.*, **28** (2019), 2872–2881. https://doi.org/10.1109/TIP.2019.2891895

26. Z. Hu, Z. Yang, X. Hu, R. Nevatia, Simple: similar pseudo label exploitation for semi-supervised classification, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2021), 15099–15108.

27. Q. Li, Z. Han, X. M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, in *Thirty-Second AAAI conference on artificial intelligence*, 2018.

28. K. Sun, Z. Lin, Z. Zhu, Multi-stage self-supervised learning for graph convolutional networks on graphs with few labeled nodes, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **34** (2020), 5892–5899. https://doi.org/10.1609/aaai.v34i04.6048

29. Z. Zhou, S. Zhang, Z. Huang, Dynamic self-training framework for graph convolutional networks, in *International Conference on Learning Representations*, 2019.

30. D. C. G. Pedronette, L. J. Latecki, Rank-based self-training for graph convolutional networks, *Inf. Process. Manage.*, **58** (2021), 102443. https://doi.org/10.1016/j.ipm.2020.102443

31. H. Scudder, Probability of error of some adaptive pattern-recognition machines, *IEEE Trans. Inf. Theory*, **11** (1965), 363–371. https://doi.org/10.1109/TIT.1965.1053799

32. Y. Yang, Y. Zhuang, Y. Pan, Multiple knowledge representation for big data artificial intelligence: framework, applications, and case studies, *Front. Inf. Technol. Electron. Eng.*, **22** (2021), 1551–1558. https://doi.org/10.1631/FITEE.2100463

33. P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, T. Eliassi-Rad, Collective classification in network data, *AI Mag.*, **29** (2008), 93. https://doi.org/10.1609/aimag.v29i3.2157

34. J. Klicpera, A. Bojchevski, S. Günnemann, Predict then propagate: Graph neural networks meet personalized pagerank, preprint, arXiv:1810.05997.

35. K. K. Thekumparampil, C. Wang, S. Oh, L. J. Li, Attention-based graph neural network for semi-supervised learning, preprint, arXiv:1803.03735.

36. C. Cortes, V. Vapnik, Support-vector networks, *Mach. Learn.*, **20** (1995), 273–297. https://doi.org/10.1007/BF00994018

37. D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back-propagating errors, *Nature*, **323** (1986), 533–536. https://doi.org/10.1038/323533a0

AIMS Press