Robust Graph Neural Networks: Modified NRGNN for Noisy Edge

I. INTRODUCTION

Graph Neural Networks (GNNs) have achieved remarkable achievement in different fields, such as recommendation systems, social network and financial systems. Taking the advantage of message passing, GNN aggregates the features from the node neighbors. Therefore, the k-hop neighbor information can be integrated and the resulting graph could have better local and global representations. In semi-supervised learning, the unknown node class or unknown edge can be predicted with the GNN mechanism which transfers the known information by message passing. However, in real-world scenarios, the known information is not always right. That is, due to the properties of the GNN mechanism, the model suffers performance degradation with following problems, including noisy labels, noisy edges, structure perturbations, sparse labels and sparse edges. These problems induce the model to transmit the wrong information to other nodes and result in poor predictions. Because of the limitation of labeled node numbers and noisy labels, the training may overfit to noisy labels and result in poor generalization. Also, sparsely labeled graphs and message passing to unlabeled neighbors could not modify the noisy labels and affect these samples easily.

For example, for unknown label node D in this graph, its neighbor B is a noisy label which could negatively affect classification results. For node D, it has 1/3 probability to be misclassified to class 1 because of noisy labels on node B and this could be more significant when the neighbors of unknown node are limited. Another example is that, node G links with only two nodes, if one of the neighbors is noisy the prediction would be poor due to edge sparsity.

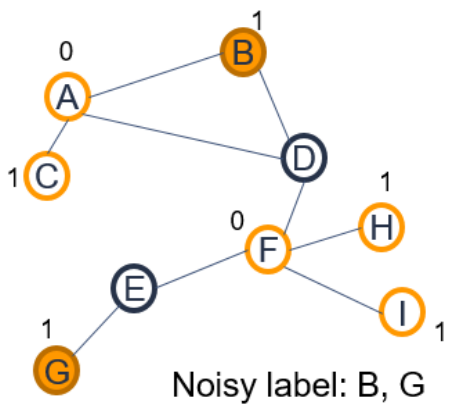


Figure 1. Illustration of effect of noisy labels and sparse edges on the prediction results.

II. PROBLEM STATEMENT AND TECHNICAL CHALLENGES

In this work, the general purpose is to develop a robust model when the input graph contains noisy information and limited labels. The model aims to predict accurate predictions for unlabeled nodes. Therefore, there are problems that need to be solved. Firstly, the developed method should have certain resistance to the noise and mitigate the negative effects. Notably, this research focuses on the prediction of node labels under the noisy information and edge sparsity.

III. RELATED WORKS

3.1 Noise resistant GNN (NRGNN)

NRGNN is the backbone framework of this study which aims to solve the noisy node labels and the sparse edges. In particular, the NRGNN framework solves these two problems with an edge predictor, a pseudo label miner and a GNN classifier. For many graphs such as social networks, nodes with similar features and labels tend to be linked, while noisy edges would link nodes of dissimilar features, Inspired by this concept. NRGNN try to generate links with high feature similarity labeled nodes. The similar features are shared in the nodes which belong to the same class. Besides, NRGNN extended the labeled nodes with pseudo labels. The details of the framework of the NRGNN would be described in the following. Firstly, the edge predictor adds the potential edges for the graph for the purposes of solving the edge sparsity. To train the edge predictor, reconstruction graph structure is taken as the loss function. To avoid a trivial answer, the reconstruction loss is not only aiming to predict the adjacency matrix but also negative sampling node pairs. As mentioned before, high node features similarity should have higher probability of edge links. Instead of giving all the pseudo edges a edge weight 1, here, NRGNN assigns pseudo links according to their feature similarity weighting for candidate edges linking unlabeled and labeled nodes as:

$$\mathrm{S}_{ij}^{L} = \begin{cases} 1 & \text{if } v_j \in \mathcal{N}(v_i); \\ \mathrm{S}_{ij} & \text{else if } \mathrm{S}_{ij} > t, v_i \in \mathcal{V}_U \text{ and } v_j \in \mathcal{V}_L; \\ 0 & \text{else,} \end{cases}$$

Next, the pseudo label miner then proposed to extend the pseudo node labels. With the graph which has more links after extending the edges by edge predictor, the pseudo label miner trained the GCN model to predict the node class. The general cross entropy loss was used to boost the correctness of pseudo node labels provided by the pseudo label miner. The extension of labeled nodes allows the model to further create more potential edges by the edge predictor. Therefore, the edge predictor and the pseudo label miner assign pseudo links and pseudo node labels iteratively. With these two models, the negative effect caused by node labels sparsity and the edge sparsity could be mitigated. Finally, the GNN classifier is then created to accurately predict node labels. Due to the extension of pseudo edges and the pseudo node labels, the trained classifier can learn more accurate information. The overall loss functions for each model are described in figure 3.

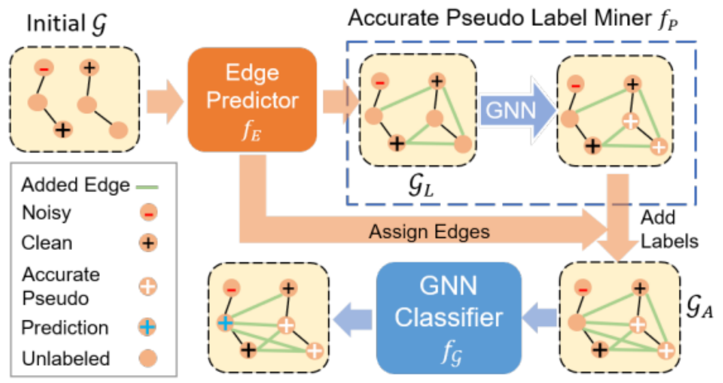
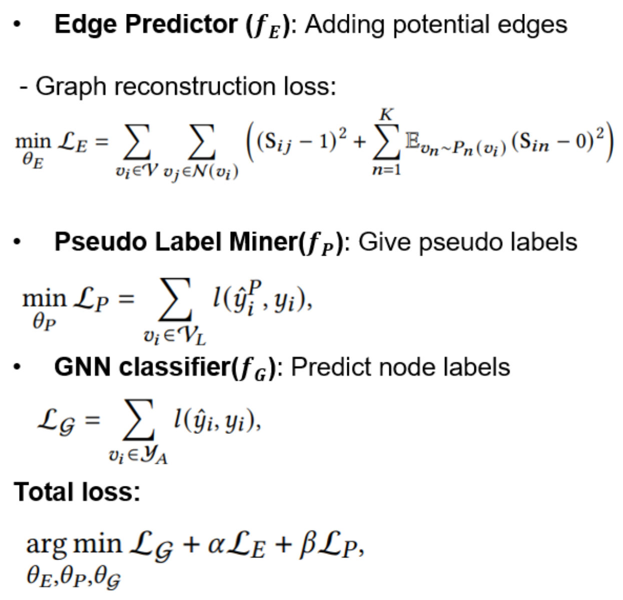


Figure 1: The overall framework of our method.

Figure 2 The framework of NRGNN



- **Edge Predictor (f_E)**: Adding potential edges

- Graph reconstruction loss:

$$\min_{\theta_E} \mathcal{L}_E = \sum_{v_i \in \mathcal{V}} \sum_{v_j \in N(v_i)} \left((S_{ij} - 1)^2 + \sum_{n=1}^{K} \mathbb{E}_{v_n \sim P_n(v_i)} (S_{in} - 0)^2 \right)$$

- **Pseudo Label Miner(f_P)**: Give pseudo labels

$$\min_{\theta_P} \mathcal{L}_P = \sum_{v_i \in \mathcal{V}_L} l(\hat{y}_i^P, y_i),$$

- **GNN classifier(f_G)**: Predict node labels

$$\mathcal{L}_{\mathcal{G}} = \sum_{v_i \in \mathcal{Y}_A} l(\hat{y}_i, y_i),$$

Total loss:

$$\arg\min_{\theta_E, \theta_P, \theta_{\mathcal{G}}} \mathcal{L}_{\mathcal{G}} + \alpha \mathcal{L}_E + \beta \mathcal{L}_P,$$

Figure 3 The overall loss function to update the NRGNN model

IV. MOTIVATION

Having discussed the framework of the NRGNN, there are several perspectives that could be further investigated to improve the performance of the model under noisy information. In NRGNN, they use two models with the same structure as the pseudo label miner and GNN classifier. Nevertheless, the function of these model models are almost the same. Therefore, in the present work, the share parameter in these models and other methods are used to update only one model for node labels predictions, Secondly, NRGNN states the higher similarity on node features the higher probability that there is edge between them. Hence, this work introduces the method proposed in [2] to constraint higher node feature similarity when there is edge in this node pair. NRGNN was proposed under the assumption that the graph only suffers from the noisy node labels. They did not take the effect of noisy edges into consideration. For example, in NRGNN, the edge weights of given edges are always set to 1. That is, NRGNN consider these existed edges are correct. To overcome the model degradation when there are noisy edges, the present study then investigates to re-weight the existing edges according to their node feature similarities.

V. METHOD

5.1 Mean teacher

Self-ensembling models including Π-model and temporal model were proposed by Laine's paper. Consistency loss is calculated by the features which are generated from two different views of input with noise and dropout. The Temporal Ensembling model accumulates the historical features as ensemble target then encourages conduct consistent results with this average ensemble target. Mean teacher [3] further take advantage of the teacher-student model and exploit the historical prediction by using an exponential moving average during the training of the teacher model. That is, Mean teacher average student model's weights instead of averaging the features. The student model is trained by gradient descent and the teacher model is updated by the exponential moving average of the student model's parameters. The teacher weights $\theta_{teacher}$ are an ensemble of the student weights $\theta_{student}$ in a successive training step t with a smoothing coefficient α ∈ [0, 1]:

$$\theta^{t}_{teacher} = \alpha \bullet \theta^{t-1}_{teacher} + (1 - \alpha) \bullet \theta^{t}_{student}$$

The exponential moving average of historical models allows the framework to train just one model and reduce the computation complexity. Moreover, the framework encourages the output prediction to be consistent for both the current model (student model) and average model (teacher model). The overall framework is shown in figure 4.

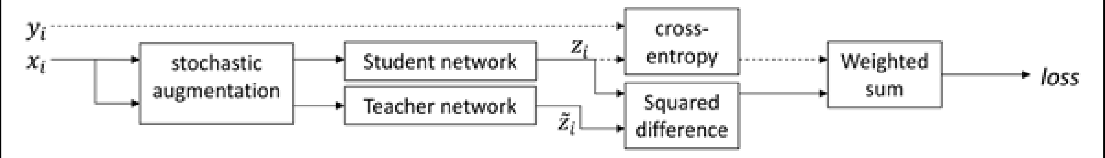


Figure 4. Mean Teacher

In the original NRGNN framework, the pseudo label miner and the GNN classifier share similar functions. There is no shared knowledge between them. Therefore, inspired by the mean teacher, this work uses the exponential moving average of the GNN classifier as the pseudo label miner. (see Equation 1)

$$f^{t}_{P} = \alpha \bullet f^{t-1}_{P} + (1 - \alpha) \bullet f^{t}_{GNN\ classifier} \qquad \text{Equation 1}$$

where f^{t}_{P} is the pseudo label miner in current training step and it was determined by the weighting sum up with pseudo label miner in previous training step f^{t-1}_{P} and GNN classifier in current training step $f^{t}_{GNN\ classifier}$. f_{P} is accumulation of historical $f_{GNN\ classifier}$. The mean teacher encourages consistent predictions from Pseudo Label Miner and GNN classifier.

5.2 Label smoothness

Pseudo edges link unlabeled nodes and labeled nodes. Higher similarities are expected for these node pairs. For label smoothness loss, it means that larger weights of an edge *Sij* indicates the more likely that *vi* and *vj* have the same label. For an unlabeled node *vi* , if its edge weight with node *vj* is larger than a threshold *Th*, i.e., *Sij* > *Th*, we want their predicted labels to be similar with each other.

$$\mathcal{L}_u = \sum_{v_i \in \mathcal{V}_u} \sum_{v_j \in \mathcal{V}} \mathbf{T}_{ij} \|\hat{\mathbf{y}}_i - \hat{\mathbf{y}}_j\|^2, \qquad \text{Equation 2}$$

The training loss then modified as:

$$argmin_{fE,fP.FG}\ L_G + \alpha L_E + \beta L_P + \gamma L_u$$

5.3 Rweight labeled edges

NRGNN do not modify the weights of labeled edges weights of labeled edges are 1 throughout training process. If we encounter noisy edges, can we progressively reweight the edge weights as if these are pseudo labels? By this, it is expected to mitigate transfering wrong information to its neighbor nodes. The comparisons of original edge weight and similarity-aware edge weight are shown in figure 5. Particularly, with the similarity-aware edge weights, the framework can not only resist noisy node labels but also have certain resistance to noisy edges.

■ Original edge weights:

$$\mathrm{S}^L_{ij} = \begin{cases} 1 & \text{if } v_j \in \mathcal{N}(v_i); \ \underline{\text{Existing edge weight = 1}} \\ \mathrm{S}_{ij} & \text{else if } \mathrm{S}_{ij} > t, v_i \in \mathcal{V}_U \text{ and } v_j \in \mathcal{V}_L; \\ 0 & \text{else,} \end{cases}$$

■ Similarity aware edge weights:

$$S^L_{i,j} = \begin{cases} \boxed{S_{ij}, \quad if\, S_{ij} > t \ and\ v_i \in N(v_i)} \ \underline{\text{Existing edge weight = its similarity}} \\ S_{ij}\ , else\ if\, S_{ij} > t, v_i \in V_U\ and\ v_j \in V_L \\ \quad 0, \quad else \end{cases}$$

Figure 5. Comparisons of original and similarity-aware edge weights

VI. EXPERIMENT SETTINGS

6.1 Datasets

The Cora dataset consists of 2708 scientific publications classified into one of seven classes. The citation network consists of 5429 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. The dictionary consists of 1433 unique words. The CiteSeer dataset consists of 3312 scientific publications classified into one of six classes. The citation network consists of 4732 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence of the corresponding word from the dictionary. The dictionary consists of 3703 unique words.

6.2 Noise types

For label noises, pair noise is used in this experiment. Labelers are assumed to make mistakes only within the most similar pair classes. More specifically, labels have a probability of p to flip to their pair class. For edge noises, we use the deeprobust toolkit to randomly add edges on the graph.

For the following experiments, we take the accuracy of GNN classifier to evaluate the efficiency of the model.

VII. RESULTS

7.1 The effect of different noise rate on original NRGNN

In this experiment, different noise rates are added to check the performance of the framework. The performance decay as the noise rate increases on both datasets. Notebly, the original NRGNN does not consider the noisy edges. That is why the performance drops significantly when suffering noisy edges.

Table 1. Accuracy of GNN classifier on cora dataset

Noise rate	Noisy node label	Noisy edge	Noisy node label + Noisy edge
0 %	0.8189	0.8189	0.8189
0.2 %	0.7953	0.7978	0.7475
0.5 %	0.6791	0.7716	0.6222

Table 2. Accuracy of GNN classifier on citeseer dataset

Noise rate	Noisy node label	Noisy edge	Noisy node label + Noisy edge
0 %	0.7115	0.7115	0.7115
0.2 %	0.6357	0.6635	0.6197

7.2 The effect of label smoothness

The accuracy of GNN classifier Noisy node labels(0.2 %) on Citeseer dataset. Table 3 shows the label smoothness improves the model performance.

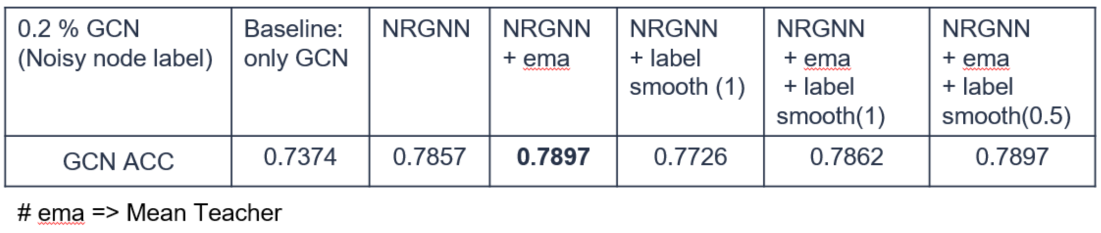
Table 3. Accuracy of GNN classifier on citeseer dataset with label smoothness

	GCN ACC
W/O label smooth (original NRGNN)	0.6357
With label smoothness	
0.3	0.6368
1	0.6564

7.3 The results comparisons for original NRGNN and modified one

For the noisy node labels, we compare the GNN performance with different strategies. There is slight improvement on the mean teacher setting in the cora dataset. However, the original framework has better performance over other strategies on the citeseer dataset.

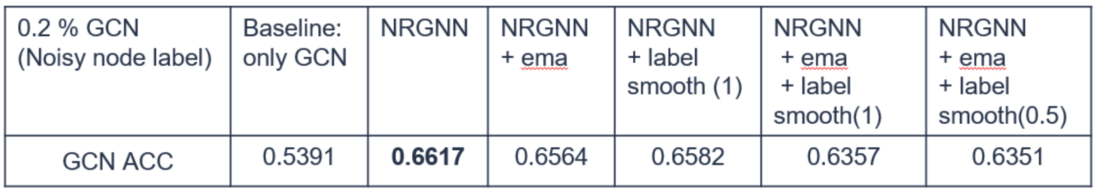
Table 4. Accuracy of GNN classifier on cora dataset



0.2 % GCN (Noisy node label)	Baseline: only GCN	NRGNN	NRGNN + ema	NRGNN + label smooth (1)	NRGNN + ema + label smooth(1)	NRGNN + ema + label smooth(0.5)
GCN ACC	0.7374	0.7857	**0.7897**	0.7726	0.7862	0.7897

ema => Mean Teacher

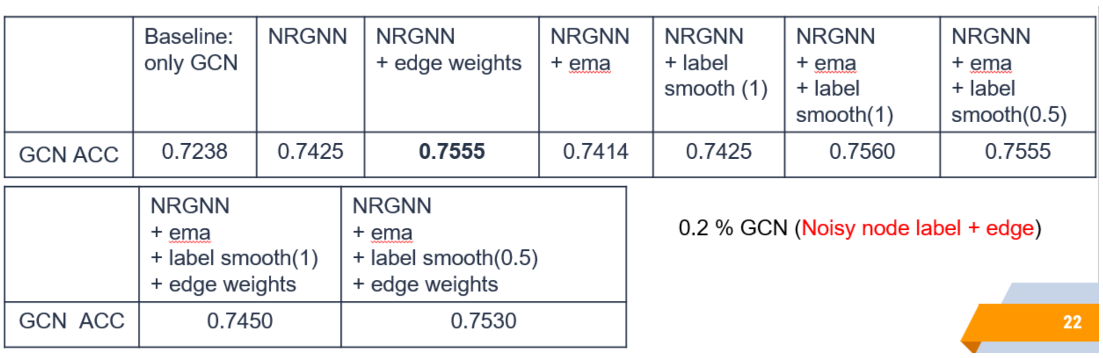
Table 5. Accuracy of GNN classifier on citeseer dataset



0.2 % GCN (Noisy node label)	Baseline: only GCN	NRGNN	NRGNN + ema	NRGNN + label smooth (1)	NRGNN + ema + label smooth(1)	NRGNN + ema + label smooth(0.5)
GCN ACC	0.5391	**0.6617**	0.6564	0.6582	0.6357	0.6351

When the graph has both noisy node labels and noisy edges, we compare the GNN performance with different strategies. It is clear that the similarity-aware edge reweight improves the performance under noisy edges on both datasets. Therefore, we can verify the efficiency of the similarity-aware edge reweight.

Table 6. Accuracy of GNN classifier on cora dataset



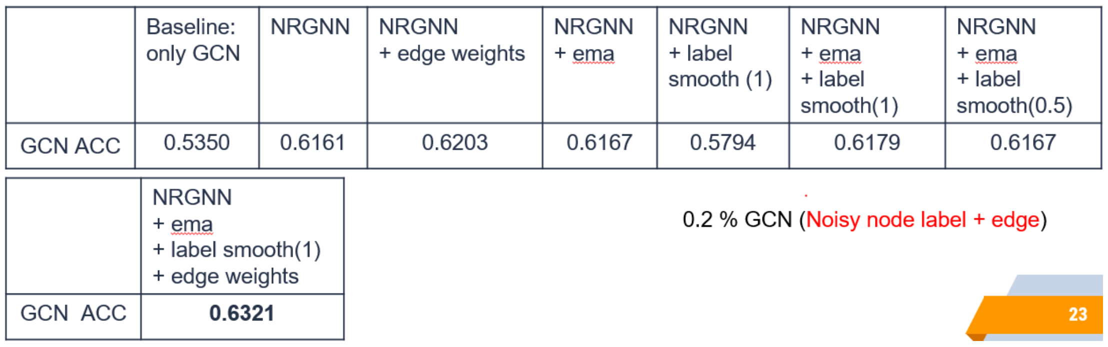
	Baseline: only GCN	NRGNN	NRGNN + edge weights	NRGNN + ema	NRGNN + label smooth (1)	NRGNN + ema + label smooth(1)	NRGNN + ema + label smooth(0.5)
GCN ACC	0.7238	0.7425	**0.7555**	0.7414	0.7425	0.7560	0.7555

	NRGNN + ema + label smooth(1) + edge weights	NRGNN + ema + label smooth(0.5) + edge weights
GCN ACC	0.7450	0.7530

0.2 % GCN (Noisy node label + edge)

22

Table 7. Accuracy of GNN classifier on citeseer dataset



	Baseline: only GCN	NRGNN	NRGNN + edge weights	NRGNN + ema	NRGNN + label smooth (1)	NRGNN + ema + label smooth(1)	NRGNN + ema + label smooth(0.5)
GCN ACC	0.5350	0.6161	0.6203	0.6167	0.5794	0.6179	0.6167

	NRGNN + ema + label smooth(1) + edge weights
GCN ACC	**0.6321**

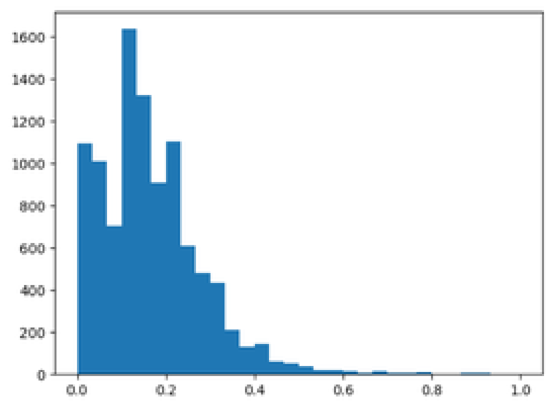
0.2 % GCN (Noisy node label + edge)

23

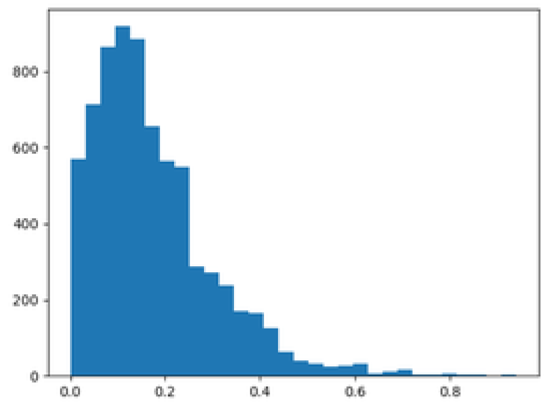
7.3 Evaluate the similarity of given edges

This work and the original NRGNN framework assume the given edges connect high similarity features. To varift this, we check the node pair feature similarity of all given edges. The feature similarity of given edges are shown in the figure 6. The cora dataset has a total 10138 given edges and only 139 edges have higher than 0.5 node pair similarity. The citeseer dataset has a total 7336 given edges and only 169 edges have higher than 0.5 node pair similarity. The pubmed dataset has a total 88651given edges and only 9307 edges have higher than 0.5 node pair similarity. The experiment shows the node features are quite different and less than 10% given edges have high confidence in terms of node pair similarities. In NRGNN, this observation may harm the assumption and cause poor performance. Further investigation should be done in the futureworks.

(a) Cora dataset



(b) Citeseer dataset



(c)pubmed

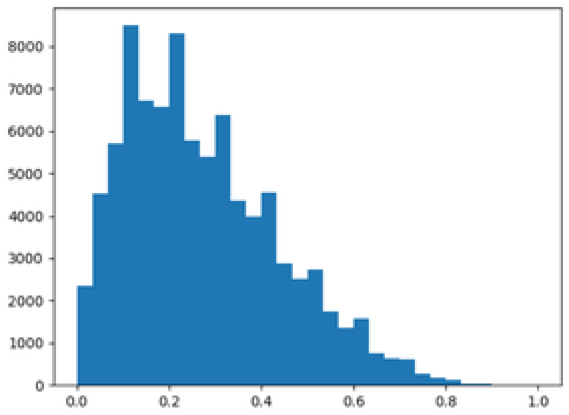


Figure 6. The cosine similarity of node pair features for given edges on three datasets, include cora, citeseer and pubmed datasets.

VIII. CONCLUSION

The experiments show NRGNN could not perform well on noisy edges. Mean teacher frameworks do not significantly improve the model. Besides, that may harm the performance. Reweighting the edges gives certain resistance for the edge noise. To resist more attack, attributes noise should be considered for future works. Finally, this work also shows the node pair similarity for existing node pairs. The results should be considered for the following works.

Reference

[1]E. Dai, C. Aggarwal, and S. Wang, “NRGNN: Learning a Label Noise-Resistant Graph Neural Network on Sparsely and Noisily Labeled Graphs,” 2021.
[2] E. Dai, W. jIN, H. Liu, and S. Wang, “Towards Robust Graph Neural Networks for Noisy Graphs with Sparse Labels.” Jan. 01, 2022

[3] A. Tarvainen and H. Valpola, “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results,” 2018.