A Dual-branch Graph Convolutional Network on Imbalanced Node Classification

Xiaoguo Wang
College of Electronics and Information Engineering,
Tongji University, Shanghai, China
xiaoguowang@tongji.edu.cn

Jiali Chen
College of Electronics and Information Engineering,
Tongji University, Shanghai, China
2030792@tongji.edu.cn

ABSTRACT

Graph convolutional neural networks (GCNs) have attracted much attention in dealing with various node classification tasks on graphs. Some real-world node classification tasks face the situation that the number of minority class nodes is significantly less than that of majority class nodes. This makes us more concerned about how to effectively solve the problem of imbalanced node classification based on GCNs. To solve this problem, we propose a Dual-branch Graph Convolutional Network framework (D-GCN), which can reduce the dominant effect of majority class on topology aggregation and the negative impact of information differences caused by graph structure reconstruction. This framework achieves the goal of decreasing the possibility of misrecognizing the minority class nodes as majority class and improving the classification performance of minority class nodes. Experiments on several graph datasets demonstrate that D-GCN outperforms representative baselines in solving imbalanced node classification tasks.

CCS CONCEPTS

• **Computing methodologies**; • **Machine learning**; • **Machine learning approaches**; • **Neural networks**;

KEYWORDS

Graph convolutional neural networks, Node classification task, Imbalanced classification

ACM Reference Format:
Xiaoguo Wang and Jiali Chen. 2021. A Dual-branch Graph Convolutional Network on Imbalanced Node Classification. In *The 5th International Conference on Computer Science and Application Engineering (CSAE 2021), October 19–21, 2021, Sanya, China.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3487075.3487162

1 INTRODUCTION

Graphs are essential data structures that generally exist in the real world, where entities are represented by nodes and relationships are described by edges. For example, social networks, knowledge graphs, and citation networks can all be represented by graphs. In recent years, people have become increasingly interested in applying deep learning methods to graph data, and graph convolutional networks (GCNs) have become a new research hotspot. GCNs have made remarkable achievements in learning node representation, and have been successfully used in different tasks, such as node classification [1, 2] and recommendation [3].

The study of GCNs is based on the idea of transferring the methods of convolutional neural networks (CNNs) to graphs, focusing on how to expand convolution on graphs to extract graph structure and node feature information. According to different implementations of graph convolution, GCNs can be classified into spatial-based and spectral-based. Spatial methods [4–6] are based on the spatial relationship of nodes. They establish a fixed-size neighbor sequence for every node, and then implement standard convolution operations on the neighborhood. Spectral methods [7–9] propose to perform convolution operation in spectral domain from the perspective of graph signal processing, to avoid explicitly constructing fixed size neighborhoods. Through these methods, GCNs effectively extract the graph structure and feature information, and are successfully applied to various node classification tasks.

However, in real-world applications, many graph datasets naturally show highly skewed node class distributions. For example, in the scenarios of fake accounts detection in social networks [10, 11], spam reviews detection on e-commerce platforms [12], and fraud detection in transaction networks [13], most of the samples are normal (majority class), but only a small part are fraudulent (minority class). Applying existing methods directly to such imbalanced data tends to bias to majority class in classification results. It means that models are likely to misrecognize minority class nodes as the majority class, seriously damaging the classification performance of minority class nodes. How to effectively solve imbalanced classification problems on graphs with the help of GCNs needs to conduct more in-depth research.

There are two main problems in applying GCNs to imbalanced scenarios. The first is topology aggregation. The core idea of graph convolution is to aggregate the feature information of neighbor nodes along the edges of the graph. It means that the class assignment for each node is not only determined by its own features but is also affected by its neighbor nodes. The imbalanced problem will make majority class nodes dominate the feature propagation between nodes. Thus, minority class nodes will be greatly impacted by majority class nodes from nearby structures when aggregating neighborhood information to generate node representation. The second is information differences. In work of [14], to solve the problem of imbalanced class distribution, minority class nodes are over-sampled. New minority class nodes are synthesized by constructing an embedding space, and the relationship between nodes is simulated by an edge generator. However, constructing

CSAE 2021, October 19–21, 2021, Sanya, China

ACM ISBN 978-1-4503-8985-3/21/10…$15.00
https://doi.org/10.1145/3487075.3487162

new nodes and new edges will inevitably change the original graph structure, which results in information differences.

In this paper, to alleviate the negative impact of the above problems on node classification performance, we propose a Dual-branch Graph Convolution Network framework (D-GCN), which can reduce the possibility of misrecognizing minority class nodes as majority class and improve the classification performance of minority class. First, according to the similarity of node features, we use k-nearest neighbor [15] strategy to obtain the kNN reconstruct graph. In this graph, nodes in the neighborhood are all with the most similar features. This can prevent minority class nodes from being dominated by majority class nodes in the neighborhood when aggregating information. Secondly, we adopt the form of two branches. One of the branches uses the kNN reconstruct graph to apply GCN to eliminate the impact of topology aggregation. The other uses the original graph to apply GCN to balance the possible negative impact of information differences generated by graph reconstruction.

Our contributions can be summarized as follows:

- We use the k-nearest neighbor reconstruct graph, which reduce the negative influence of minority class nodes from majority class nodes on its nearby structures.
- We propose a Dual-branch Graph Convolutional Network framework (D-GCN) with the form of two branches. On the basis of using kNN to reconstruct the graph, it takes advantage of the original graph structure information to reduce the possible negative effects of information differences. Thus, classification performance of the minority class is improved.
- Experiments on five graph datasets demonstrate the proposed framework outperforms representative baselines in imbalanced classification metrics.

2 RELATED WORK

In this section, we briefly review related works on graph convolution neural networks and class imbalance problems.

2.1 Graph Convolutional Neural Networks

In these years, graph convolution neural networks (GCNs) have received much attention and developed rapidly. Graph convolutional neural network models can generally be categorized to spatial-based and spectral-based. Spatial-based methods perform convolution operations directly on graphs. For example, PATCHY-SAN [4] uses a fixed size to convert the graph into a node sequence, and then performs a standard convolution operation on the neighborhood. GraphSAGE [5] proposes a method of randomly selecting fixed-size neighborhoods for sampling and feature aggregating. GAT [6] introduces the attention mechanism to graph networks. The spectral-based methods perform convolution operations in the spectral domain to avoid explicitly constructing a fixed size neighborhood. For example, Bruna et al. [7] implement graph convolution operations in the Fourier domain using graph Laplacian matrices. ChebNet [8] further simplifies the graph convolution operation on [7] by using the k-order Chebyshev polynomial of graph Laplacian. GCN [9] proposes to aggregate only one-hop neighbors to obtain a much simpler form. In addition, some methods [16–18] focus on how to solve the problem of overfitting and over-smoothing to construct deep graph networks. Some works [19] attempts to enhance the existing GCNs by various improvements such as the addition of layers, hyperparameters optimization, the combination of activation functions, and so on.

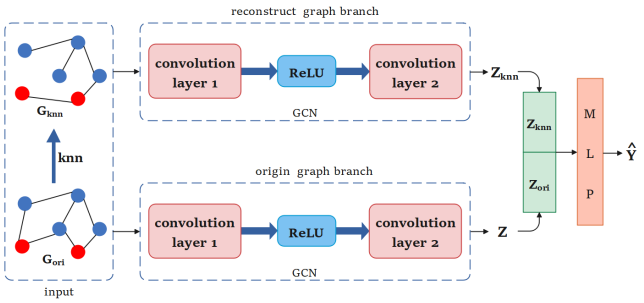

Figure 1: The Overall Framework of D-GCN. D-GCN contains two GCN branches, where the reconstruct graph branch uses the k-nearest neighbor reconstruct graph G_{knn} as Input, and the original graph branch uses the original graph G_{ori} as Input.

2.2 Class Imbalance Problems

Class imbalance is a common problem in real-world applications. Many graph datasets naturally show imbalanced class distribution, such as social networks for fake accounts detections. Methods to solve imbalance problems can usually be categorized to algorithm level and data level. Algorithm-level approaches modify existing algorithms to better recognize minority class samples. For example, DR-GCN [20] proposes to impose a conditional adversarial regularization and a distribution alignment regularization to deal with graph imbalance data based on graph networks. RA-GCN [21] uses weighing networks for samples to help the classifier fit better between classes and adopts an adversarial training approach to avoid bias towards any classes. Data-level approaches directly adjust the distribution of data classes by over-sampling or under-sampling in preprocessing stage. GraphSMOTE [14] proposes to construct an embedding space to encode the nodes similarity and synthesize new nodes, then use an edge generator to simulate the relationship between nodes to expand the graph. The work in [22] uses weighted GNN as the classifier and presents a Binary tree-based Graph Oversampling Algorithm (BGOA) to tackle imbalanced problems on non-Euclidean data.

3 THE PROPOSED MODEL

In this section, to solve the two problems mentioned in section 1, we designed a Dual-branch Graph Convolution Network framework (D-GCN). The framework of D-GCN is shown in Figure 1. There are two key ideas in our model. First, according to the similarity of node features, the k-nearest neighbor reconstruct graph is constructed as the input of the reconstruct graph branch to train GCN. The other is to use the original graph to train GCN in the original graph branch. Then, concatenate the embeddings trained by the two branches and use the final embedding to predict node classification. These two optimizations can prevent our model from over-recognizing the majority class and improve the performance of minority class.

3.1 The Origin Graph Branch

Graph convolution network (GCN) [9] mainly focuses on semi-supervised node classification on undirected graphs. An undirected graph can be formally represented by G = (A,X) where $A \in \mathbb{R}^{n \times n}$ is the symmetric adjacency matrix on the graph with n nodes, $X \in \mathbb{R}^{n \times d}$ is the feature matrix of nodes, and d is node features dimension. In the graph, if there is an edge between node i and node j, then $A_{ij} = 1$, otherwise, $A_{ij} = 0$. All node representations in GCN are regarded as graph signals, so the l-th graph convolution layer can be represented as:

$$H^{(l)} = ReLU\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l-1)}W^{(l)}\right), \tag{1}$$

where $H^{(l)}$ is the output of the l-th graph convolutional layer, and the initial $H^{(0)} = X$. Here, $\tilde{A} = A + I$ and $\tilde{D}$ is the diagonal matrix of $\tilde{A}$. $W^{(l)}$ is the weight matrix of the l-th layer. To simplify the representation, we can use $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$.

As for the origin graph branch, we have the original graph $G_{ori} = (A_{ori}, X_{ori})$, where $A_{ori} = A$ and $X_{ori} = X$. The output embedding of the original graph branch Z_{ori} can be represented as:

$$Z_{ori} = \widehat{A_{ori}}ReLU\left(\widehat{A_{ori}}X_{ori}W^{(0)}\right)W^{(1)}, \tag{2}$$

where $W^{(0)} \in \mathbb{R}^{d \times nhid1}$ and $W^{(1)} \in \mathbb{R}^{nhid1 \times nhid2}$ are respectively the learned parameters for the first and second convolution layers.

3.2 The Reconstruct Graph Branch

In the reconstruct graph branch, to avoid the negative influence of majority class nodes nearby the structure when minority class nodes are aggregating, we compute the similarity matrix $S \in \mathbb{R}^{n \times n}$ of all nodes based on node features. Specifically, the cosine similarity is commonly used to measure the similarity between each pair of node feature vectors, and the similarity between node feature x_i and node feature x_j is represented as :

$$S_{ij} = \frac{{x_i}^T \cdot x_j}{x_i x_j}. \tag{3}$$

So we have the kNN reconstruct graph $G_{knn} = (A_{knn}, X_{knn})$, where $A_{knn} = S$ and $X_{knn} = X$. The output embedding of the reconstruct graph branch Z_{knn} can be represented as:

$$Z_{knn} = \widehat{A_{knn}}ReLU\left(\widehat{A_{knn}}X_{knn}W^{(0)}\right)W^{(1)}. \tag{4}$$

3.3 Objective Function

According to the work in Section 3.1 and Section 3.2, the output embedding can be finally represented as:

$$Z = concat(Z_{knn}, Z_{ori}). \tag{5}$$

Then, perform a linear transformation and a softmax transformation on the embedding Z to obtain the prediction result $\hat{Y}$:

$$\hat{Y} = softmax\left(W \cdot Z + b\right). \tag{6}$$

Using L to stand for the training set, the loss function of all nodes participating in training can be expressed as:

$$\mathcal{L} = -\sum_{l \in L}\sum_{i=1}^{C} Y_l ln\widehat{Y_l}. \tag{7}$$

Here, because our work mainly focuses on binary imbalanced nodes classification on graphs, in the above equation we have C = 2.

4 EXPERIMENTS AND RESULTS

In this section, we conduct experiments to evaluate node classification performance of D-GCN on graph datasets with imbalanced classes.

4.1 Experimental Setup

4.1.1 Datasets. At present, the widely used benchmark graph datasets are multi-classes and the distribution of classes are balanced. In order to conduct experiments for imbalanced problems, we carry out binary processing to five datasets (ACM [23], BlogCatalog [24], Citeseer, UAI [25], Cora). Here we label nodes of the class with the least number as the positive class, and the rest are all labeled negative class. The overview of datasets after binary processing is summarized in Table 1. IR represents the imbalance rate, which is the ratio of majority class nodes to minority class nodes. The five imbalanced graph datasets used in the experiments across a large IR range, from 2.1:1 to 14.0:1. Thus, we can evaluate the performance of the proposed model in different levels of imbalance.

- ACM: This is extracted from the ACM, where nodes represent documents and edges exist between two documents with the same author. The node features are representations of document keywords.
- BlogCatalog: This social network is composed of bloggers with their social relationships. Node features are constructed from keywords of blogger profiles.
- Citeseer: Citeseer is a citation network for research papers, where nodes are scientific publications and edges are links.
- UAI: This is a community detection network dataset. The node features are user information such as age and gender, and edges are interactions between users.
- Cora: Cora is a well-known citation network dataset, similar to Citeseer.

4.1.2 Baselines. In order to assess the effectiveness of our model on imbalanced node classification, we select the following representative models as baselines for performance comparison, including:

- GCN [9]: GCN learns node representation by aggregating neighbor nodes, which is a popular semi-supervised graph convolution models.
- kNN-GCN: kNN-GCN uses the k-nearest neighbor reconstruct graph as the input of GCN. Here we use it as a baseline to evaluate its performance alone without the original graph branch.
- GAT [6]: GAT introduces the attention mechanism to graph networks and is widely used as a baseline for GNNs.

4.1.3 Parameters Setting. For all baseline methods, they are initialized with the suggested parameters of their papers. We turn their parameters to obtain optimal performance. For our D-GCN, the two branches are both 2-layer GCN, and the dropout rate is set to 0.5. We train two branches with the same hidden layer units nhid1 and output embedding dimension nhid2 respectively, where nhid1 $\in \{256, 512, 1024\}$ and nhid2 $\in \{64, 128, 256\}$. In addition, we have k $\in \{2 \ldots 9\}$ for k-nearest neighbor reconstruction graph. In training, we use Adam optimizer with a learning rate of 0.001~0.01

Table 1: The Overview of Datasets

Datasets	Nodes	Edges	Features	Origin Classes	Training	Test	IR
ACM	3025	13128	1870	3	60	1000	2.1:1
BlogCatalog	5196	171743	8189	6	120	1000	5.7:1
Citeseer	3327	4732	3703	6	120	1000	11.6:1
UAI	3067	28311	4973	19	380	1000	12.9:1
Cora	2708	5429	1433	7	140	1000	14.0:1

and weight decay $\in \{1e-4, 5e-4, 1e-3, 5e-3\}$. For all methods, we run with the same partition of all datasets.

4.2 Evaluation Metrics

The imbalanced classification metrics used for evaluation are defined based on the following four measurements. The superscript c can be 0 or 1, respectively indicating negative class and positive class in our binary classification.

tp^c: The truth is c, and the prediction is c.
tn^c: The truth is not c, and the prediction is not c.
fp^c: The truth is not c, and the prediction is c.
fn^c: The truth is c, and the prediction is not c.

We choose Macro F1 and Binary F1 to digitize the performance of models in solving imbalanced classification. The definitions of metrics are as follows.

Macro F1: In classification, F1 Score is defined as the harmonic average of precision and recall for a specific class. In binary classification problems, Macro F1 represents the average F1 Score of positive class and negative class, which is a metric suitable for evaluating imbalanced classification problems. For binary classification, Macro F1 can be computed as:

$$Macro\ F1 = \frac{1}{2}\left(\frac{2tp^0}{2tp^0 + fp^0 + fn^0} + \frac{2tp^1}{2tp^1 + fp^1 + fn^1}\right). \quad (8)$$

Binary F1: In binary classification problems, Binary F1 is the F1 Score of the minority class (positive class). This metric can evaluate the performance in classifying minority class samples. Binary F1 can be calculated as:

$$Binary\ F1 = \frac{2tp^1}{2tp^1 + fp^1 + fn^1}. \quad (9)$$

The value of two metrics ranges [0,1].

4.3 Imbalanced Node Classification

Regarding node classification performance of models on five imbalanced graph datasets, we show it in two aspects. One is to verify the effectiveness of the D-GCN dual-branch architecture in Section 4.3.1, the other is to compare with the baseline methods in Section 4.3.2.

4.3.1 Dual Branch Effectiveness. The effectiveness of the D-GCN dual-branch framework is verified by comparing the node classification performance of dual-branch D-GCN with single-branch GCN and kNN-GCN on five datasets. The results are shown in Figure 2. From the results, we have the following observations:

- On BlogCatalog, Citeseer, and UAI datasets, kNN-GCN outperforms GCN on Macro F1 and Binary F1. This shows that using the kNN reconstruct graph based on feature similarity in training can reduce the possibility of misrecognizing minority nodes as the majority class.
- From the results on ACM and Cora datasets, it can be seen that kNN-GCN does not always perform better than GCN. This is probably since only using the kNN reconstruct graph lacks the structure information of the original graph. This proves the necessity of using the origin graph branch.

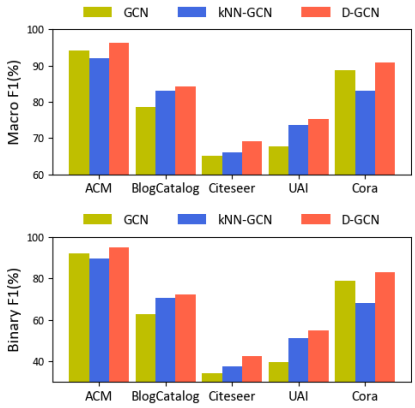

Figure 2: The Results(%) of D-GCN, kNN-GCN, GCN.

For all five datasets, classification performance of D-GCN is better than that of GCN and kNN-GCN, integrating the advantages of both, which demonstrates the necessity and effectiveness of the dual-branch architecture of our model.

4.3.2 Performance Comparison. We compare classification performance of D-GCN and all baseline methods on five datasets, as shown in Table 2. Compared with GAT, a more advantageous model than GCN, Binary F1 of D-GCN on the five datasets are increased by 1.18%, 23.05%, 3.31%, 15.06%, and 2.93% respectively. That shows that the improvement made by our model for the topology aggregation problem is indeed effective and classification performance of minority class nodes is improved. Compared with kNN-GCN, D-GCN in Macro F1 outperforms by 4.21%, 1.18%, 3.02%, 1.80%, 7.88% respectively, and Binary F1 have also been improved accordingly. This highlights the positive impact of using the original graph structure information on imbalanced node classification. The comparison with all baseline methods shows our model outperforms

Table 2: Node Classification Results (%) (Bold - Best, Underline - Runner-Up)

Datasets	Metrics	GCN	kNN-GCN	GAT	D-GCN
ACM	Macro F1	94.09	92.05	<u>95.31</u>	**96.27**
	Binary F1	92.05	89.60	<u>93.81</u>	**94.99**
BlogCatalog	Macro F1	78.47	<u>82.97</u>	71.72	**84.15**
	Binary F1	62.64	<u>70.71</u>	48.98	**72.03**
Citeseer	Macro F1	65.08	66.08	<u>67.01</u>	**69.10**
	Binary F1	34.43	37.42	<u>39.24</u>	**42.55**
UAI	Macro F1	67.65	<u>73.53</u>	67.39	**75.33**
	Binary F1	39.71	<u>51.01</u>	40.00	**55.06**
Cora	Macro F1	88.83	83.02	<u>89.39</u>	**90.90**
	Binary F1	78.99	67.89	<u>80.00</u>	**82.93**

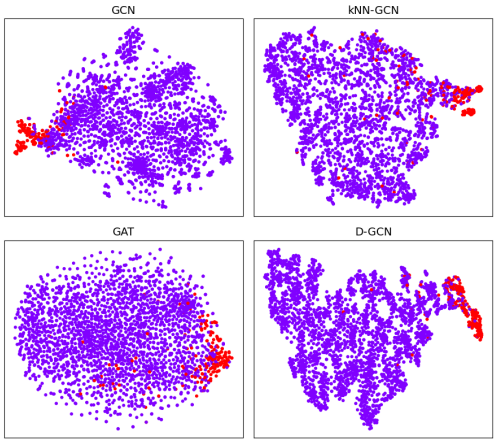

Figure 3: Visualization on Cora Dataset (Minority Class in Red).

others, and achieves the goal of reducing the possibility of misrecognizing minority nodes as the majority class and improving the classification performance of the minority class.

4.4 Visualization

To further show the effectiveness of D-GCN, we implement the visualization on Cora. We use the embedding from the last layer of models before softmax and plot the embeddings using t-SNE. The results on Cora are shown in Figure 3, where minority class (positive class) nodes are colored red. The visualization results of kNN-GCN and GAT are not sufficiently good, because many positive nodes are mixed with negative nodes. The visualization of GCN and D-GCN both perform better for the boundary between two classes can be seen more clearly and minority class nodes are more obviously cluster together. That shows that the learned embedding has higher intra-class similarity and inter-class difference.

5 CONCLUSION

In real world, graph networks in many scenarios naturally show highly skewed node class distributions. In this paper, we propose a Dual-branch Graph Convolutional Network framework (D-GCN) to solve the imbalanced problem on graphs, which can prevent the over-recognition of the majority class and improve the classification performance of minority class nodes. Specifically, on the one hand, we use k-nearest neighbor strategy to reconstruct the graph to avoid being dominated by the majority class nodes nearby the structure when minority nodes aggregate information; on the other hand, we adopt a dual-branch architecture to use the original graph structure information to balance the negative impact of information difference caused by graph structure reconstruction. Finally, extensive experiments for the imbalance problem are conducted and imbalance classification metrics are used to evaluate the performance of models on each dataset. Both experimental results and visualization results show that D-GCN is effective in solving imbalanced node classification tasks on graphs.

REFERENCES

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, *et al.* (2017). Attention is all you need. In *NIPS*.
[2] Jun Wu, Jingrui He, and Jiejun Xu. (2019, July). Net: Degree-specific graph neural networks for node and graph classification. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 406-415.
[3] Rex Ying, Ruining He, Kaifeng Chen, *et al.* (2018, July). Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 974-983.
[4] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. (2016, June). Learning convolutional neural networks for graphs. In *International conference on machine learning*, 2014-2023. PMLR.
[5] William L. Hamilton, Rex Ying, and Jure Leskovec. (2017). Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 1025-1035.
[6] Petar Veličković, Guillem Cucurull, Arantxa Casanova, *et al.* (2017). Graph attention networks.
[7] Joan Bruna, Wojciech Zaremba, Arthur Szlam, *et al.* (2014). Spectral networks and locally connected networks on graphs.
[8] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. (2016, January). Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*.
[9] Thomas N. Kipf, Max Welling. (2016). Semi-supervised classification with graph convolutional networks.
[10] Mohammadreza Mohammadrezaei, Mohammad E. Shiri, and Amir M. Rahmani. (2018). Identifying fake accounts on social networks based on graph analysis and classification algorithms. *Security and Communication Networks 2018.*
[11] Adam Breuer, Roee Eilat, and Udi Weinsberg. (2020, April). Friend or faux: Graph-based early detection of fake accounts on social networks. In *Proceedings of The Web Conference 2020*, 1287-1297.
[12] Ao Li, Zhou Qin, Runshi Liu, *et al.* (2019, November). Spam review detection with graph convolutional networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2703-2711.

[13] Jianguo Jiang, Jiuming Chen, Tianbo Gu, *et al.* (2019, November). Anomaly detection with graph convolutional networks for insider threat and fraud detection. In *MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM)*, 109-114. IEEE.

[14] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. (2021, March). GraphSMOTE: Imbalanced Node Classification on Graphs with Graph Neural Networks. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 833-841.

[15] Leif E. Peterson. (2009). K-nearest neighbor. Scholarpedia, 4(2), 1883.

[16] Ming Chen, Zhewei Wei, Zengfeng Huang, *et al.* (2020, November). Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, 1725-1735. PMLR.

[17] Yu Rong, Wenbing Huang, Tingyang Xu, *et al.* (2019, September). Dropedge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations*.

[18] Guohao Li, Matthias Müller, Guocheng Qian, *et al.* (2021). Deepgcns: Making gcns go as deep as cnns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[19] Ihsan Ullah, Mario Manzo, Mitul Shah, *et al.* (2019). Graph Convolutional Networks: analysis, improvements and results. arXiv e-prints, arXiv-1912.

[20] Min Shi, Yufei Tang, Xingquan Zhu, *et al.* (2020, January). Multi-Class Imbalanced Graph Convolutional Network Learning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)*.

[21] Mahsa Ghorbani, Anees Kazi, Mahdieh Soleymani Baghshah, *et al.* (2021). RA-GCN: Graph Convolutional Network for Disease Prediction Problems with Imbalanced Data. arXiv e-prints, arXiv-2103.

[22] Qing Zhou, Jiang Li, Yinchun Tang, *et al.* (2020, November). Discovering the Lonely Among the Students with Weighted Graph Neural Networks. In *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, 474-481. IEEE.

[23] Xiao Wang, Houye Ji, Chuan Shi, *et al.* (2019, May). Heterogeneous graph attention network. In *The World Wide Web Conference*, 2022-2032.

[24] Zaiqiao Meng, Shangsong Liang, Hongyan Bao, *et al.* (2019, January). Co-embedding attributed networks. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 393-401.

[25] Wenjun Wang, Xiao Liu, Pengfei Jiao, *et al.* (2018, June). A Unified Weakly Supervised Framework for Community Detection and Semantic Matching. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 218-230. Springer, Cham.