ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10448064

FINCGAN: A GAN FRAMEWORK OF IMBALANCED NODE CLASSIFICATION ON HETEROGENEOUS GRAPH NEURAL NETWORK

Hung Chun Hsu [1*], *Ting-Le Lin* [2*], *Bo-Jun Wu* [3], *Ming-Yi Hong* [1], *Che Lin* [145], *Chih-Yu Wang* [16†]

[1] Graduate Program of Data Science, National Taiwan University and Academia Sinica, Taiwan
[2] Department of Accounting and Statistics, National ChengChi University, Taiwan
[3] Department of Risk Management and Insurance, National ChengChi University, Taiwan
[4] Department of Electrical Engineering, National Taiwan University, Taiwan
[5] Graduate Institute of Communication Engineering, National Taiwan University, Taiwan
[6] Research Center for Information Technology Innovation, Academia Sinica, Taiwan

ABSTRACT

Graph Neural Networks (GNNs) frequently face class imbalance issues, especially in heterogeneous graphs. Existing GNNs often assume balanced class sizes, which isn't true in many cases. Applying them directly to imbalanced data can lead to sub-optimal performance. Traditional oversampling methods, while effective, risk overfitting and face difficulties in reintegrating synthetic samples into the original graph. In this study, we introduce **F**ramework of **I**mbalanced **N**ode **C**lassification on heterogeneous graph neural network with **GAN** (**FincGAN**), a new framework that utilizes oversampling techniques to address class imbalance in heterogeneous graphs. Instead of duplicating existing samples, FincGAN employs a Generative Adversarial Network (GAN) to create synthetic samples and uses deep learning-based edge generators to connect them back to the original graph. Our evaluations on spam user detection in the Amazon and Yelp Review datasets show that FincGAN outperforms baseline models in all essential metrics, including F-score and AUC-PRC score, showing its effectiveness in addressing class imbalance.

Index Terms— Class Imbalance, Generative Adversarial Network, Graph Neural Network, Heterogeneous Graph

1. INTRODUCTION

Advances in Graph Neural Networks (GNNs) have enabled success in tasks like spam and fraud detection [1]. However, class imbalance remains a challenge, particularly in heterogeneous graphs. Traditional solutions for class imbalance in non-graph data are inadequate for GNNs as they ignore graph structure. Recent works have attempted to address this using various graph-based techniques [2, 3]. Most rely on weight reassignment, which limits their effectiveness, especially for heterogeneous graphs.

In this work, we present **FincGAN**, a novel framework for tackling class imbalance in heterogeneous GNNs. It uses Generative Adversarial Networks (GAN) [4] to generate quality nodes and edges, avoiding overfitting. Our evaluation of the Amazon datasets shows that FincGAN, compared to the original baselines, has a 9% and 10% improvement in AUC-PRC and F-score, respectively; whereas on the Yelp dataset, it shows a 3.3% and 3.2% improvement.

The contributions of our work are summarized as follows:
(1) We propose FincGAN, a novel GAN-based framework to handle the class imbalance problem for heterogeneous graph-structured data. The generation ability of GAN synthesizes high-quality samples for minority classes in the target dataset.
(2) We introduce sparsity-aware edge generators to support edge generation for heterogeneous edge types, linking synthetic nodes back to the original graph.
(3) FlashGAN outperforms other baselines on critical metrics in the experiments, validating the effectiveness of our framework on the imbalanced dataset and demonstrating its advantages in most real-world scenarios targeting minority classes.

2. RELATED WORKS

Graph Neural Networks GNNs, such as GCN [5], primarily focus on homogeneous graphs, while extensions like HGT [6] tackle heterogeneous graphs. However, they often neglect class imbalance issues.

Generative Adversarial Networks GANs [4] offer adversarial training with variants like WGAN [7], LSGAN [8], and Conditional GAN [9] allowing controlled sample generation for minority classes.

Class Imbalance The class imbalance problem is pervasive, often addressed through data-level methods like oversampling and SMOTE [10] or loss-level approaches [11]. Oversampling risks overfitting, while undersampling discards valuable data. GANs have been used to balance classes, but many methods assume independent samples, which isn't suitable

*The first two authors contributed equally to this work.
†Corresponding author. Email: cywang@citi.sinica.edu.tw. This work was supported by the National Science and Technology Council, Taiwan [grant number: NSTC 111-2628-E-001 -002 -MY3, NSTC 112-2218-E-194 -005 -, NSTC 111-2622-8-002-028-(111C2415) and NSTC 111-3111-E-002-003], National Taiwan University, Taiwan [grant number: NTU-CC-112L891104], and Academia Sinica, Taiwan [grant number: AS-TP-110-M07-3].

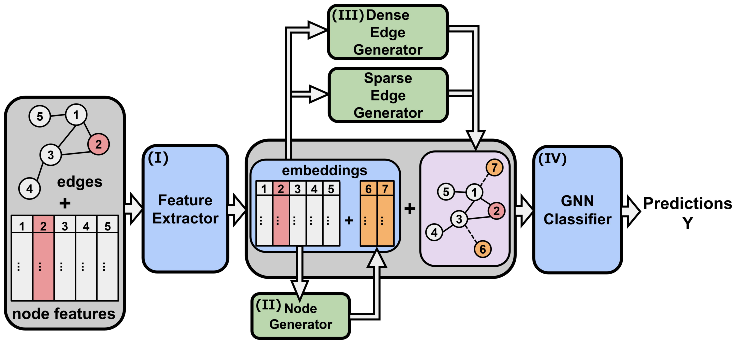


Fig. 1: Overview of FincGAN

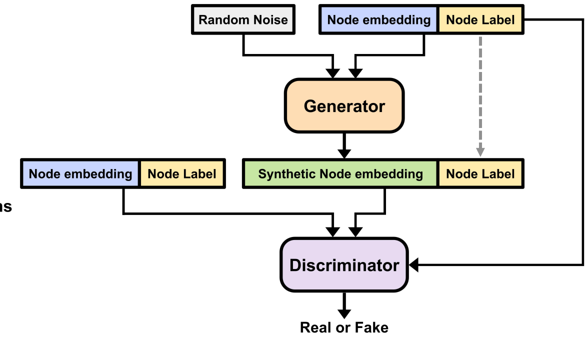


Fig. 2: Architecture of node generator

for GNNs. Other methods include G^{2}GNN [12], which locally augments minority graphs using Graph of Graph construction, and DR-GCN [13], addressing multi-class imbalanced graphs with dual regularization.

3. PROBLEM DEFINITION

We address class imbalance in node classification on heterogeneous GNNs in a transductive and heterogeneous graph setting. We denote $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ as a heterogeneous graph. Each node $\nu \in \mathcal{V}$ is associated with the node type mapping function $\phi(\nu) : \mathcal{V} \rightarrow \mathcal{A}$. Each edge $e \in \mathcal{E}$ is associated with the edge type mapping function $\psi(e) : \mathcal{E} \rightarrow \mathcal{R}$. $\mathcal{A}$ denotes the set of node types. The $\mathcal{R}$ indicates the collection of edge types. $\mathcal{A}_T$ means the target node type. The Y represents the class information of nodes in $\mathcal{G}$.

During training, only partial label information Y_L is available, where Y_L defines the labels of nodes in the training set $\mathcal{V}_L$. The $|C_i|$ indicates the size of i-th class, which means the sample size of class C_i, and the C_M denotes the majority class with most samples. We use $\frac{\min_i(|C_i|)}{\max_i(|C_i|)}$ as the imbalance ratio to evaluate the class imbalance problem. The formal problem definition is as follows:

Given $\mathcal{G}$ with the class imbalance problem and labels of $\mathcal{V}_L$, we focus on learning a classifier $\mathcal{F}$ that can perform node classification on the whole graph, $\mathcal{F}(\mathcal{V}, \mathcal{E}) \rightarrow Y$.

4. METHODOLOGY

In this section, we discuss the components of FincGAN, illustrated in Figure 1. FincGAN comprises four parts: (i) a feature extractor for node representation, (ii) a GAN-based node generator for minority classes in latent space, (iii) sparsity-aware edge generators to link generated nodes to the original graph, and (iv) a GNN classifier for node classification on the augmented graph.

4.1. Feature Extractor

We employ a feature extractor to learn compact node embeddings capturing both inter-class and intra-class correlations along with graph structure. These embeddings feed into the node generator to create synthetic nodes. We choose HGT as our feature extractor, a state-of-the-art option suitable for any heterogeneous GNN. Formally, HGT classifies nodes on graph $\mathcal{G}$, providing the final layer output as the node embedding: $\text{HGT}(\mathcal{V}, \mathcal{E}) \rightarrow H$, where H represents the node embeddings of the graph. The $H_{\mathcal{A}_i}$ indicates the node embeddings of node type $\mathcal{A}_i$.

4.2. Node Generator

In our framework, we employ DCGAN[14] as the node generator to produce new nodes with embeddings similar to the considered class, illustarated in Figure 2.

In our Generator G, a noise vector z from $\mathcal{N}(0,1)$ is concatenated with source node embedding h_s. This combined input and label y_s are fed into G to produce synthetic node embedding $\widetilde{h}$. Node generation relies on G and is conditioned on node representation, label, and noise. For the discriminator, it receives the node embedding h and label y to output a reality score. Formally, the objective function is:

$$\begin{aligned}\min_G \max_D V(D,G) = \mathbb{E}_{h\sim p_{\text{data}}}[\log \mathrm{D}(h,y)] \\ + \mathbb{E}_{\substack{z\sim N(0,1)\\ h_s\sim p_{\text{data}}}}[\log(1 - D(z, h_s, y_s))]\end{aligned} \tag{1}$$

where p_{data} denotes the distribution of the node embedding. Therefore, the generator can produce nodes belonging to the assigned class, and we can produce nodes of minority classes to adjust the sample distribution.

4.3. Edge Generator

Next, we link the synthetic nodes back to the original graph $\mathcal{G}$, from which they were initially isolated, enabling subsequent GNNs to leverage these nodes during training. While GraphSMOTE offers a straightforward edge generator for homogeneous edges, we customize it for dense-distributed edge types and introduce a new generator for sparse edge types in heterogeneous graphs.

Dense edge generator The dense edge generator uses weighted inner products of node embeddings to predict the connection of nodes [2]. Formally, the structure is as follows:

$$E_{v,u} = \sigma(h_v \cdot S \cdot h_u) \;\; \forall v, u \tag{2}$$

where $E_{v,u}$ indicates the probability of v and u having a connection and S is a learnable matrix. For training the edge generator, we apply the loss function

$$\mathcal{L}_{\mathrm{D_{edge}}} = \|E - A\|_F^2 , \tag{3}$$

where E denotes predicted node connections and A means the adjacency matrix of the edge type. By optimizing the weight matrix S, the edge generator can learn the distribution of node connections. With the edge generator, we can use the node embedding of the synthetic node $v^{'}$ to generate edges by setting a threshold η. If $E_{v',u} > \eta$, we connect v' and u; otherwise, they remain unconnected.

Sparse edge generator For the sparse edge distribution, we apply a Multilayer Perceptron (MLP) to this task:

$$\mathrm{MLP}\left(\mathrm{concat}\left(h_v, h_u\right)\right) \rightarrow E_{v,u} \quad \forall v, u \in \mathcal{V}, \tag{4}$$

$$\mathcal{L}_{\mathrm{S_{edge}}} = -\sum \left(A_{v,u} \log\left(E_{v,u}\right) + \left(1 - A_{v,u}\right) \log\left(1 - E_{v,u}\right)\right), \tag{5}$$

where $A_{v,u}$ indicates whether v and u are connected. Same way as the dense edge generator, we can use the node embedding of the synthetic node $v^{'}$ to generate edges by setting a threshold η, connect v' and u if $E_{v',u} > \eta$.

4.4. GNN Classifier

Let $\widetilde{\mathcal{E}}$ and $\widetilde{\mathcal{V}}$ be the augmented edges and nodes containing information of real and synthetic nodes. Now, we can form an augmented graph $\widetilde{\mathcal{G}} = \{\widetilde{\mathcal{V}}, \widetilde{\mathcal{E}}\}$ with a balanced class distribution. Since the sample size of different classes in $\widetilde{\mathcal{G}}$ is balanced, we train an unbiased GNN classifier through the augmented graph. Here we adopt HGT on $\widetilde{\mathcal{G}}$: HGT $\left(\widetilde{\mathcal{V}}, \widetilde{\mathcal{E}}\right) \rightarrow Y$.

4.5. Training Procedure

The FincGAN method starts with node embedding extraction using a feature extractor. We then train node and edge generators with these embeddings. The node generator creates synthetic nodes for minority classes, and the edge generator links them to the original graph, forming an augmented graph. By incorporating synthetic nodes, the augmented graph can address class imbalance in downstream tasks. Finally, a node classifier is trained on the augmented graph for evaluation.

5. EXPERIMENTS

We apply FincGAN to real-world datasets to evaluate its performance. Particularly, we would like to investigate: (1) Is FincGAN effective on node classification tasks with the class imbalance issue? (2) Would different up-sampling scales affect the performance of FincGAN? (3) How does the edge generator affect FincGAN?

5.1. Experimental Settings

Datasets We apply FincGAN to address the class imbalance problem in two datasets: Amazon reviews [15] and Yelp-Fishers. In the Amazon dataset, focused on Musical Instruments reviews, we detect spam users based on the ratio of helpful votes [16]. Users with over 70% helpful votes are labeled as benign. For the Yelp-Fishers dataset, we label users as benign if their reviews average useful scores above 0.1.

Table 1: Graph statistics

Amazon			
Type	# Nodes (Fraud%)	Edge Type	# Edges
User	7,017 (9.7%)	U-U	535,244
Product	4,684	U-P	12,169
		P-P	101,678
Yelp			
Type	# Nodes (Fraud%)	Edge Type	# Edges
User	13,050 (20.5%)	U-U	14,245,842
Review	25,319	U-P	24,400
Product	570	P-P	76,852
		U-R	25,319
		R-P	25,319

Graph Construction In the Amazon review graph, we have two node types, user and product, and three edge types: U-P for user-product ratings, U-U for users with similar ratings or review text, and P-P for products with similar descriptions or mutual 'Also Buy' listings. For the Yelp dataset, our graph has three node types: user, review, and product, along with five edge types: U-P for user-product reviews, U-U for users with shared reviews or similar text, P-P for same-category products, U-R, and R-P linking reviews to users and products. Graph statistics are detailed in Table 1.

Baselines To validate our graph augmentation method, we replace the node and edge generators with several data imbalance techniques. These include Oversampling, which duplicates minority nodes; Reweight, which adjusts class weights in the loss function; Noise, an extension of Oversampling adding noise to samples; SMOTE [10], which interpolates samples and their nearest neighbors; GraphSMOTE [2], a SMOTE variant for homogeneous graphs; ImGAGN [17], a GAN-based approach that simulates minority node attributes and network structures; and PC-GNN [18], which uses label-balanced and neighborhood samplers for sub-graph construction. All methods are integrated into the graph while maintaining original connections and tested on homogeneous graphs constructed by meta-paths to report the best results.

Evaluation Metrics We use the following metrics for evaluations: AUC-ROC [19], AUC-PRC [20], F-Score, and Precision. AUC-PRC is preferred for imbalanced datasets as AUC-ROC can overrate models in such cases.

Configurations We use FincGAN to create a balanced augmented graph, generating spam nodes and edges via node and edge generators. For Amazon, thresholds for U-U and U-P edge generators are 0.91 and 0.99, respectively, with an imbalance ratio of 0.7. For Yelp, the thresholds are 0.999 and 0.95, with an imbalance ratio of 0.82. Experiments are repeated 30 times to average out randomness.

Table 2: Experimental Results: Imbalanced Classification

Method	AUC-PRC	AUC-ROC	F-Score	Precision
Amazon				
Original	0.4051	0.8457	0.4018	0.5083
Oversampling	0.3461	0.8348	0.3357	0.5406
SMOTE	0.3453	0.8530	0.3682	0.4508
Reweight	0.3624	0.8291	0.4170	0.3709
Noise	0.3514	0.8472	0.3443	0.5711
GraphSMOTE	0.3513	0.8364	0.3499	0.4889
ImGAGN	0.2725	0.5518	0.1812	0.3502
PC-GNN	0.3381	0.8521	0.2923	0.4049
FincGAN	**0.4374**	**0.8709**	**0.4505**	**0.5712**
Yelp				
Original	0.5005	0.8296	0.4541	0.5828
Oversampling	0.4890	0.8285	0.4083	0.5878
SMOTE	0.4963	0.8306	0.4537	0.6005
Reweight	0.4859	0.8253	**0.4991**	0.5333
Noise	0.5132	0.8315	0.4449	0.5976
GraphSMOTE	0.5130	0.8348	0.4505	0.5947
ImGAGN	0.4361	0.4975	0.2800	0.1736
PC-GNN	0.4860	**0.8431**	0.4961	0.2037
FincGAN	**0.5173**	0.8365	0.4688	**0.6061**

5.2. Imbalanced Classification Performance

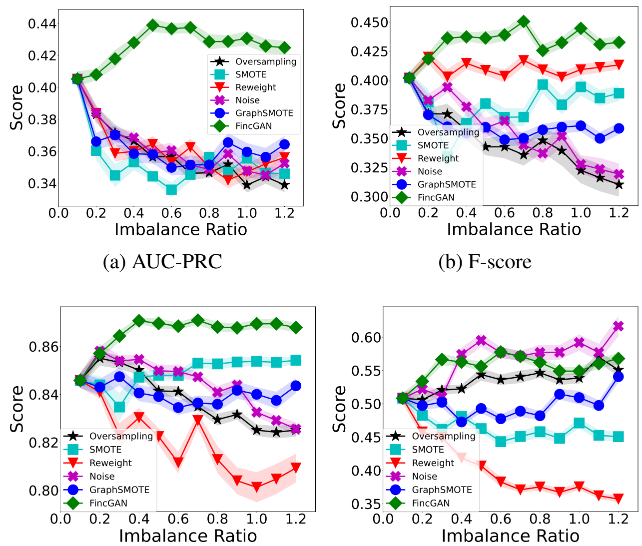
Table 2 shows FincGAN surpassing all baselines on most metrics, including significant gains in AUC-PRC and AUC-ROC scores. Particularly, FincGAN excels in Precision, making it ideal for high false-positive risk scenarios like recommendation systems and spam detection. FincGAN also shows improvements in AUC-PRC and AUC-ROC for the Yelp dataset. Overall, FincGAN is highly effective for imbalanced node classification, primarily demonstrated on the Amazon dataset.

5.3. Influence of Up-sampling Scale

In this subsection, we analyze the impact of varying the number of synthetic nodes in different approaches while maintaining the up-sampling scale to achieve target imbalance ratios. The experimental results are shown in Figure 3.

We observed that FincGAN's performance is robust to varying up-sampling scales, especially when the target imbalance ratio is above 0.5. While there may be an optimal ratio, FincGAN is generally insensitive to scale changes. Noise excels in Precision score, but its F-score is significantly lower than FincGAN's, indicating FincGAN is generally more effective in detecting positive cases.

Since Reweight and PC-GNN don't address data imbalance by adding synthetic minority nodes to the original graph, and ImGAGN is designed for homogeneous graphs, they receive less information with homogeneous graphs constructed by meta-paths. Hence, we exclude these three baselines from the experiment.



(a) AUC-PRC (b) F-score (c) AUC-ROC (d) Precision

Fig. 3: Effects of up-sampling scale

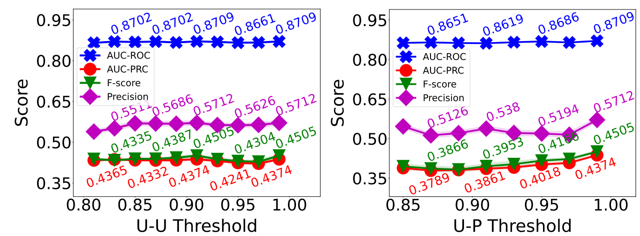


Fig. 4: Influence of Threshold of Edge Generators

5.4. Influence of Edge Generators: Dense and Sparse

In FincGAN, both the dense and sparse edge generators utilize thresholds to filter qualified edges and nodes. These thresholds control edge quality and, consequently, graph augmentation. For the dense edge generator, we set fixed values for various parameters such as the node qualification threshold at 0.81 and adjusted the edge qualification threshold. We fixed the U-U edge generator threshold at 0.91 for the sparse edge generator while varying the U-P threshold. Experimental outcomes are depicted in Figures 4a and 4b.

No significant difference in AUC-ROC score was observed for the dense edge generator, with optimal AUC-PRC scores achieved at a threshold of 0.91. The sparse edge generator's performance improved with higher U-P thresholds, emphasizing the importance of edge quality in both cases.

6. CONCLUSIONS

We address the class imbalance in heterogeneous graphs by integrating traditional oversampling with deep learning through our novel framework, FincGAN. It utilizes GAN's generative capabilities to create node embeddings, links them to the original graph using sparsity-dependent edge generators, and trains a classifier on the augmented graph. Experimental results on real-world data confirm FincGAN's effectiveness, outperforming baseline methods on key metrics.

7. REFERENCES

[1] Zhiwei Liu, Yingtong Dou, Philip S Yu, Yutong Deng, and Hao Peng, "Alleviating the inconsistency problem of applying graph neural network to fraud detection," in *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, 2020, pp. 1569–1572.

[2] Tianxiang Zhao, Xiang Zhang, and Suhang Wang, "Graphsmote: Imbalanced node classification on graphs with graph neural networks," in *Proceedings of the 14th ACM international conference on web search and data mining*, 2021, pp. 833–841.

[3] Zhenhua Huang, Yinhao Tang, and Yunwen Chen, "A graph neural network-based node classification model on class-imbalanced graph data," *Knowledge-Based Systems*, vol. 244, pp. 108538, 2022.

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[5] Thomas N Kipf and Max Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[6] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun, "Heterogeneous graph transformer," in *Proceedings of The Web Conference 2020*, 2020, pp. 2704–2710.

[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*. PMLR, 2017, pp. 214–223.

[8] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2794–2802.

[9] Mehdi Mirza and Simon Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[10] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[11] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun, "Learning to reweight examples for robust deep learning," in *International conference on machine learning*. PMLR, 2018, pp. 4334–4343.

[12] Yu Wang, Yuying Zhao, Neil Shah, and Tyler Derr, "Imbalanced graph classification via graph-of-graph neural networks," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, New York, NY, USA, 2022, CIKM '22, p. 2067–2076, Association for Computing Machinery.

[13] Min Shi, Yufei Tang, Xingquan Zhu, David Wilson, and Jianxun Liu, "Multi-class imbalanced graph convolutional network learning," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, Christian Bessiere, Ed. 7 2020, pp. 2879–2885, International Joint Conferences on Artificial Intelligence Organization, Main track.

[14] Alec Radford, Luke Metz, and Soumith Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[15] Julian John McAuley and Jure Leskovec, "From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 897–908.

[16] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and VS Subrahmanian, "Rev2: Fraudulent user prediction in rating platforms," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018, pp. 333–341.

[17] Liang Qu, Huaisheng Zhu, Ruiqi Zheng, Yuhui Shi, and Hongzhi Yin, "Imgagn: Imbalanced network embedding via generative adversarial graph networks," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery amp; Data Mining*, New York, NY, USA, 2021, KDD '21, p. 1390–1398, Association for Computing Machinery.

[18] Yang Liu, Xiang Ao, Zidi Qin, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He, "Pick and choose: A gnn-based imbalanced learning approach for fraud detection," in *Proceedings of the Web Conference 2021*, New York, NY, USA, 2021, WWW '21, p. 3168–3177, Association for Computing Machinery.

[19] Andrew P Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[20] Jesse Davis and Mark Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 233–240.