

2SFGL: A Simple And Robust Protocol For Graph-Based Fraud Detection

Zhirui Pan
Fudata Technology
Shanghai, China
1970289663@qq.com

Guangzhong Wang
Bank of Communications
Shanghai, China
wang.guangzhong@bankcomm.com

Zhaoning Li
Bank of Communications
Shanghai, China
lizhaoning@bankcomm.com

Lifeng Chen
Fudata Technology
Shanghai, China
tianpu@fudata.cn

Yang Bian
Fudata Technology
Shanghai, China
douheng@fudata.cn

Zhongyuan Lai
Fudan University
Shanghai, China
abrikosoff@yahoo.com

Abstract—Financial crime detection using graph learning improves financial safety and efficiency. However, criminals may commit financial crimes across different institutions to avoid detection, which increases the difficulty of detection for financial institutions which use local data for graph learning. As most financial institutions are subject to strict regulations in regards to data privacy protection, the training data is often isolated and conventional learning technology cannot handle the problem. Federated learning (FL) allows multiple institutions to train a model without revealing their datasets to each other, hence ensuring data privacy protection. In this paper, we propose a novel two-stage approach to federated graph learning (2SFGL): The first stage of 2SFGL involves the virtual fusion of multiparty graphs, and the second involves model training and inference on the virtual graph. We evaluate our framework on a conventional fraud detection task based on the FraudAmazonDataset and FraudYelpDataset. Experimental results show that integrating and applying a GCN (Graph Convolutional Network) with our 2SFGL framework to the same task results in a 17.6%-30.2% increase in performance on several typical metrics compared to the case only using FedAvg, while integrating GraphSAGE with 2SFGL results in a 6%-16.2% increase in performance compared to the case only using FedAvg. We conclude that our proposed framework is a robust and simple protocol which can be simply integrated to pre-existing graph-based fraud detection methods.

Index Terms—2SFGL, federated learning, graph learning, financial crime identification

I. INTRODUCTION

Identifying financial fraud and money laundering and generating the corresponding blacklist is a common method of risk management for financial institutions. Financial crime costs the industry annually up to tens of billions of dollars globally [1], and its negative effects far exceed the monetary value itself. In order to avoid greater financial losses, financial institutions often spend a lot of human and financial resources to build a robust and stable IT system to prevent fraudulent activities. However, criminals could commit financial crimes across different financial institutions. Conventional fraud detection systems are usually hard-pressed to function well in the presence of cross-institutional data, as they need to integrate data originating from multiple sources and consisting

of multiple modalities. From the point-of-view of financial institutions, integrating and utilizing user and transaction data distributed across different institutions to combat fraudulent activities has become an important challenge.

Currently, these challenges are typically addressed by centralizing the data and employing traditional machine learning methods for identifying criminal behavior. However, the increased difficulty of achieving data centralization across institutions has led to a greatly elevated likelihood of false positives, against which an increasing amount of human resource needs to be utilized. The main reason for false-positive alerts is insufficient training data or insufficient variability in the training data structure [1]. In reality, graph structured data could be found everywhere (e.g., social networks, financial transaction networks, etc.). Graph learning uses various graph models to mine valuable information from graph structured data, often performing better than traditional machine learning methods, for tasks such as fraud and community detection [2]. More specifically, fraud detection methods that rely on anomaly detection requires the ability to isolate and identify clusters of nodes; these nodes could be of interest to banks due to their interconnections and could potentially signal the existence of a group of collaborating parties. Currently, most existing graph learning is designed for centralized scenarios, where graph data is stored centrally in the single graph database and model training is centralized. For example, most banks share common customers with other institutions. Hence a complete credit assessment could only be completed in collaboration with other institutions. One straightforward approach is to collect graph data from various institutions and merge them into a large graph, and then perform the relevant graph learning algorithms on the large graph. However, in the financial industry, graph data exist in form of islands [3], and hence it is unrealistic to collect these graph data in a centralized manner because of privacy concerns, the independence of each organization’s graph data, and competition with each other. Therefore, to solve the problem of false-positive alerts, it is crucial to organize graph data from different institutions

for combined calculations, while simultaneously ensuring data privacy and security.

Two main challenges should be solved in research on federated graph learning:

- **Heterogeneity:** In the process of federated graph learning, the graph data of each client is likely to be highly Non-Independent Identically Distributed (Non-IID), i.e. the probability distribution of each data point is unlikely to be the same and each point unlikely to be independent from one another [4]; on the contrary, highly correlated data is to be expected in large-scale structured data such as those commonly found in graph data. In which case the final trained global model is likely to be unsatisfactory due to the large differences between the locally trained models of the clients;
- **Complementarity:** There are a large number of duplicate vertices in the graph data between clients, and the graph structure (vertex, vertex-to-vertex connections) of each client is not comprehensive due to the inability to share and aggregate the original graph data of each party.

In this paper, we propose a novel two-stage approach to federated graph learning (2SFGL) for further solving the above-proposed problem. In the first stage of 2SFGL, each client normalizes their respective edge values with respect to local graph data and then virtually fuses them into each other, thus alleviating the heterogeneity problem and leveraging complementarity. The second stage of 2SFGL is to apply relevant graph learning algorithms, such as the label propagation algorithm (LPA), PageRank or Graph Neural Network (GNN) on the fused graph. The 2SFGL procedure proposed avoids the training of local models and hence greatly reduces the problem of unsatisfactory learning results due to heterogeneity and complementarity (e.g., instability of the trained global model).

II. BACKGROUND

A. Privacy-preserving Strategies

Privacy is one of the most important goals of federated learning. Data private collaborative learning introduces additional restrictions to the training process over that of data-shared learning as the computational process is not identical. In this section, we briefly review and compare different privacy techniques, which can be combined with federated learning. The methods of differential privacy [5] involve adding noise to the data or using generalization methods to obscure certain sensitive attributes until the third party cannot distinguish the individual. Secure Multi-party Computation (SMPC) naturally involves multiple parties, with each party knowing nothing except its input and output. It is possible to build a security model with SMPC under lower security requirements in exchange for efficiency [6]. Homomorphic Encryption has also been adapted to protect user data privacy through parameter exchange under the encryption mechanism during machine learning. Recent works are widely used and polynomial approximations need to be made to evaluate non-linear functions in machine learning algorithms [7] [8].

B. Federated Learning

Federated learning [9] is a novel technique used to train models for machine learning based on datasets distributed across different institutions, meanwhile ensuring that data privacy is not compromised. Federated learning is classified into horizontal and vertical federated learning, as well as federated transfer learning. Horizontal FL (HFL) is the case of federated learning where all users across the network trains their respective models with data having the same set of features, while vertical FL (VFL) is the setting where each user across the network trains a common model using different datasets with varying features [10]. Up to now, many improvement efforts have been devoted to addressing various problems of federated learning, including reducing network bandwidth traffic [11] and more effective protection of data privacy [12]. Federated learning is applicable in finance, healthcare, and other fields. Recent research on federated learning on graph data can be understood as either centralized or decentralized federated graph learning. Centralized graph federated learning implies the existence of multiple clients and a central server, which combines multiple clients for model training and shares the global model with the clients, while in decentralized federated graph learning, multiple participants collaborate to train the same model without a centralized server to control the entire model update. In Fig. 1, the left side of the figure represents the centralized federated graph learning process with K clients and one central server. Each client trains a graph model based on local graph data, and the server receives the local model parameters or gradients sent by all clients to update the global model and then distributes the global model to each client for the next batch of training. However, deciding who should be the central server is a difficult process. The right side of the figure represents the decentralized federated graph learning process. Lalitha et al. [13] formally describes the fully decentralized federated learning problem and proposed an efficient distributed training method. Zhao et al. [14] removed the centralized server and updated the parameters for federated learning between clients. FedAvg [9] uses a federated learning approach that aggregates the average of the respective local models. The ultimate goal of federated learning is to obtain a more generalized global framework that is suitable for all clients, while at the same time preserving clients' privacy. However, in some recent studies [15], [16] when the local data of each client is Non-IID, existing federated learning methods are difficult to obtain a better global generalization model by training.

C. Graph-based Machine Learning

Graph learning is a machine learning method that uses graph data for classification, clustering, and feature extraction. A range of graph learning is used to determine graph features, such as connectivity, centrality, and community discovery. Graph features can be combined with non-graph features and training data labels are provided to define as supervised machine learning. If training data is not provided with labels, then the problem is an unsupervised machine learning

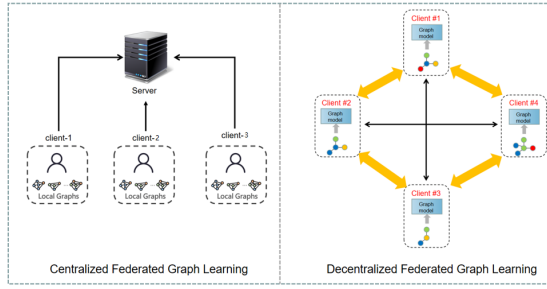


Fig. 1. The left figure represents centralized federated graph learning, and the right figure represents decentralized federated graph learning.

problem, we can apply machine learning algorithms such as clustering, dimensionality reduction, or PageRank. Recently, there have been many advances in scalable graph computation for billion-scale or even trillion-scale graphs [17] [18], so it is reasonable to expect that this approach would remain practical, even for large graphs. Akoglu et al. [19] studied graph-based anomaly detection. Moloy et al. [20] used PageRank algorithm for fraud detection. Liu et al. [21] used graph embedding methods to detect financial criminal activities, such as anti-money laundering (AML). Recently, some works [22] [23] have studied GNN-based fraud detection. Weber et al. [24] used graph convolutional networks (GCN) to detect money laundering. Pareja et al. [25] explored dynamic considerations in graph networks, which is essential in fraud detection applications. Dou et al. [26] proposed solutions to expose camouflaged fraudsters via GNNs. We note that these previous works focus more on local graph features, while our current work concentrates on global graph features across multiple institutions.

III. RELATED WORKS

In this part, we survey related work on SMPC on graphs, federated learning on GNNs, and graph-based fraud detection.

A. Secure Multi-party Computations on Graphs

The importance of secure multi-party computations has increased in the past decade [27]. It allows several parties to build a model on their pooled data to increase utility while not explicitly sharing data with each other. Secure two-party computation was first investigated by Yao and was later generalized to multiparty computation [28]. Secure multi party computation based privacy preserving techniques are usually adopted to privacy through secret sharing and homomorphic encryption scheme for the original graph data. On the other hand, secure multi-party computations on graphs, such as breath-first-search, leader election, secure sum on graphs, secure minimum and maximum search of all inputs on graphs, and secure vertex coloring, as all much more rarely studied. Currently, there is little research on SMPC-based graph computation. We applied SMPC to the virtual fusion process of the graph.

B. Federated Learning on Graph Neural Networks

Recently, some works have trained graph-data models in federated learning settings, mainly by combining federated learning with graph neural networks, and several federated graph neural networks have been proposed. For example, Zheng et al. [29] designed a new federated learning framework for graph convolutional neural networks, Wu et al. [30] designed a federated GNN framework for privacy-preserving recommendations, Jiang et al. [31] proposed a dynamic representation method for learning objects from multi-user graph sequences. We train the model on the respective party-based virtual fusion graphs by FL and hence leveraging heterogeneity and complementarity.

C. Graph-based Fraud Detection

Financial fraud detection is based on behavioral data from financial platforms to detect malicious accounts, defaulting users, and fraudulent transactions. GEM et al. [32] adaptively learns discriminative embeddings from heterogeneous account-device graphs for malicious account detection. Semi-GNN [33] is a semi-supervised GNN model with a hierarchical attention mechanism for explainable fraud prediction. FdGars [34] is a graph convolutional network approach for fraudster detection in an online app review system. GraphConsis [34] investigates the context, feature, and relation inconsistency problem in graph-based fraud detection. CARE [35] enhances the GNN aggregation process against camouflage for opinion fraud detection. These works are based on the FL-on-GNN approach to fraud detection; we also employ a similar approach in this paper. However, in our case we perform a graph virtual fusion algorithm before graph federal learning for model training, thus promoting fast convergence of models.

IV. TWO-STAGE APPROACH TO FEDERATED GRAPH LEARNING FOR FRAUD DETECTION

In this part, we present a new federated learning framework that combines graph data across different clients for a fast virtual fusion approach to enrich graph data from different clients. And then using the FedAvg method on the virtual fusion graphs to train GNN model. An overview of the framework is shown in Fig. 2.

A. The First Stage of 2SFGL

In the first step of 2SFGL, we apply Private Set Intersection (PSI) [36] to ensure the privacy and security of graph data. PSI is a cryptographic technique in SMPC that allows each party involved in the computation to compute the intersection of their data without obtaining additional information from the other party. The PSI technique allows multiple participants to compute the common vertices of graph data without revealing local source information. Specifically, PSI acquires two or even more parties of customers with common vertices. Each client separately finds the edge relations of vertices common to their respective graphs. We cannot directly send edge information between common vertices to other participants, so we first normalize edge values as follows:

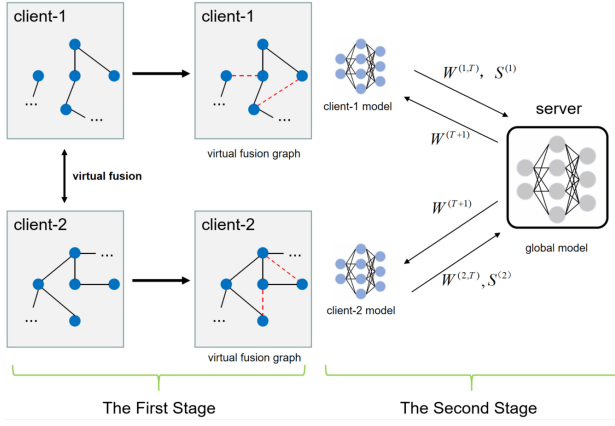


Fig. 2. Overview of 2SFGL. The first stage of the 2SFGL is virtual fusion by the respective client, and the second stage is FedAvg on the virtual fusion graph of the respective client

$$N_{ij} = \begin{cases} \frac{E_{ij}}{\sum_{k=1}^n E_k} & E_{ij} > 0 \\ 0 & E_{ij} = 0 \end{cases} \quad (1)$$

where E_{ij} denotes the value of edges between vertices V_i and V_j , n is the total number of edges connected to vertices V_i , $\sum_{k=1}^n E_k$ denotes the sum of the values of the edges connected to vertices V_i , and N_{ij} denotes the normalized result of the value of the edge relationship between vertices V_i and V_j . Then differential privacy is applied to N_{ij} , to avoid the exposure of sensitive individual information.

Each specific client sends the normalized results of all common vertices to the other clients, and the updated common vertex edge value is calculated using information from PSI:

$$E_{ij}^{new} = \begin{cases} \frac{N_{ij}}{1-N_{ij}} \sum_{k=1}^n E_k & N_{ij} < \lambda \\ \frac{\lambda}{1-\lambda} \sum_{k=1}^n E_k & N_{ij} \geq \lambda \end{cases} \quad (2)$$

where E_{ij}^{new} is the updated value of the edge relationship between vertices V_i and V_j , N_{ij} denotes the normalized result of the value of the edge relationship between vertices V_i and V_j sent from other clients. $\sum_{k=1}^n E_k$ denotes the sum of the values of the vertex V_i neighboring edges, and λ is the threshold to be set by user of the federated graph learning framework. The threshold value indicates the importance of the edge relationship of the client's graph data. In this paper, the threshold value of 0.5 is used, which indicates that the client-side edge relationship values of two graphs from separate parties are at least of equal importance. We call the augmented virtual graph obtained after performing the computations detailed above a *virtual fusion graph*, and the process is a decentralized one. Under our framework, we can perform multiple *hop* virtual fusion. Multiple hop virtual fusion can be chosen from one, two, or three hop virtual fusion, where one hop virtual fusion

is a virtual fusion of edges with one hop, and two hop virtual fusion is a virtual fusion of edges with one hop and two hops respectively. Fig. 3 shows a one hop virtual fusing process, and Fig. 4 shows a two hop virtual fusing process.

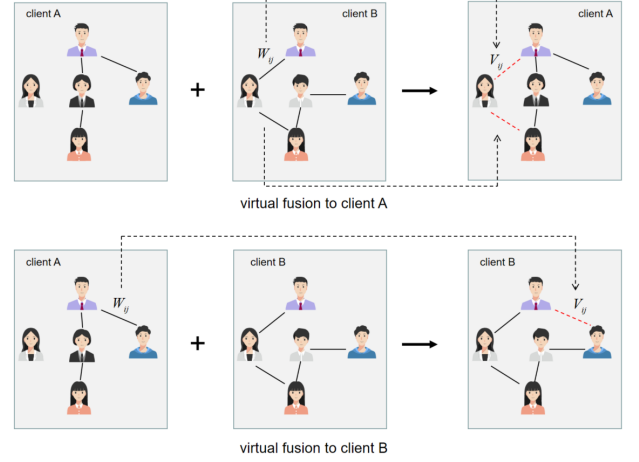


Fig. 3. One hop virtual fusion.

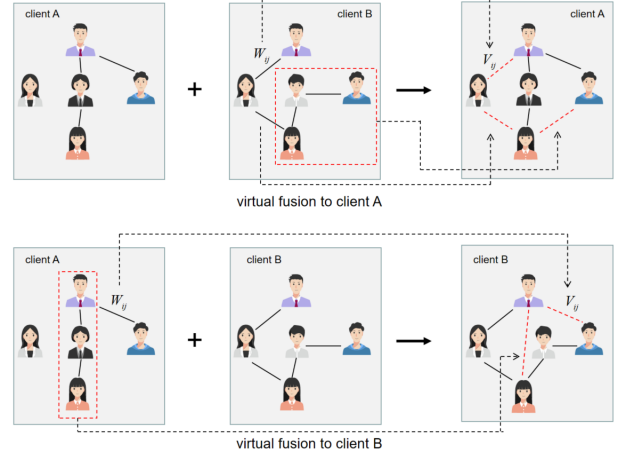


Fig. 4. Two hop virtual fusion.

B. The Second Stage of 2SFGL

After the first stage of 2SFGL, the relevant algorithms are performed on the virtual fusion graph. In addition, the stage is a centralized process. In this paper, we use conventional graph neural network algorithms such as GCN and GraphSAGE, to verify the performance of our federated graph learning framework on the FraudAmazonDataset. In brief, the GCN is defined as:

$$H = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X W \quad (3)$$

where $X \in R^{N \times F}$ is the input matrix, $H \in R^{N \times F'}$ is the convolved matrix, and $W \in R^{F \times F'}$ is the parameter. F and F' are the dimensions of the input and the output, respectively. GraphSAGE is a general inductive framework which generates

embeddings by sampling and aggregating features from a node's local neighborhood [37]:

$$\begin{aligned} h_{N_v}^{t+1} &= \text{AGG}_{t+1}(\{h_u^t, \forall u \in N_v\}), \\ h_v^{t+1} &= \sigma(W^{t+1} \bullet [h_v^t || h_{N_v}^{t+1}]). \end{aligned} \quad (4)$$

where AGG_{t+1} is an aggregation function [37], and σ is an activation function, such as the sigmoid function.

In the second stage, we follow the FedAvg algorithm. FedAvg has become the de facto FL algorithm where clients communicate with the central server at each epoch. The second stage of the 2SFGL is shown in Fig. 2. In the T -th epoch, a client sends local model parameters $W^{(k,T)}$ to the central server, where k is the k th client, and T is the T -th epoch. The central server averages all the updates from the clients to obtain the global update $W^{(T+1)}$ which is broadcast to all clients in the $(T+1)$ -th epoch. In addition, the full second phase is a centralized process.

$$W^{(T+1)} = \sum_{k=1}^N \frac{S^{(k)}}{\sum_{i=1}^N S^{(i)}} W^{(k,T)} \quad (5)$$

V. EXPERIMENTS

A. Datasets and Settings

We use FraudAmazonDataset [38] and FraudYelpDataset [39] to establish the efficacy of 2SFGL on the fraud detection task. The FraudAmazonDataset includes product reviews under the Musical Instrument category. The vertices in the constructed graph for the FraudAmazonDataset are users with 100-dimension features and contains three types of connections: 1) $U - P - U$, connecting users reviewing at least one common product; 2) $U - S - U$, connecting users having at least one same star rating within one week, and 3) $U - V - U$, connecting users with top 5% common review texts (as measured by TF-IDF) among all users. The FraudAmazonDataset can be used for the common fraud detection task, which is to find spam comments on online platforms. The FraudYelpDataset includes hotel and restaurant reviews filtered (spam) and recommended (legitimate) by Yelp. A spam review detection task can be conducted, which is a binary classification task. The nodes in the graph of YelpChi dataset are reviews with 100-dimension features and have three relations: 1) $R - U - R$ denotes the reviews posted by the same user; 2) $R - S - R$ denotes the reviews under the same product with the same star rating; 3) $R - T - R$ denotes the reviews under the same product posted in the same month.

In dataset sampling, we ensure that the ratio of positive to negative samples is in the range of 1:2 to 2:1, while ensuring that the train, test ratio are set to be 60%, 40% respectively. The parameters of GCN and GraphSAGE are optimized with Adam optimizer, the learning rate is set to be 0.005, and add only one hidden layer, the number of neurons in its hidden layer is 64. Specifically, the neighborhood size is set to 5 in GraphSAGE.

B. Baseline Performance Analysis

In the first stage of 2SFGL, we apply virtual fusion on FraudAmazonDataset's three graph connections ($U - P - U$,

$U - S - U$, $U - V - U$) within one hop relationship. Likewise we virtual fusion FraudYelpDataset's three graph connections ($R - U - R$, $R - S - R$, $R - T - R$). In the second stage of 2SFGL, We use FedAvg method for model training of GCN and GraphSage on virtual fusion of multi-party graphs. For a fair comparison, we compare 2SFGL with a Federated Learning approach using only FedAvg. Also to demonstrate the effect of 2SFGL, GCN and GraphSAGE were also used separately on the three relational single local graph. We measure the performance using generic metrics, namely Accuracy, Macro-F1, AUC, and GMean [40] [41].

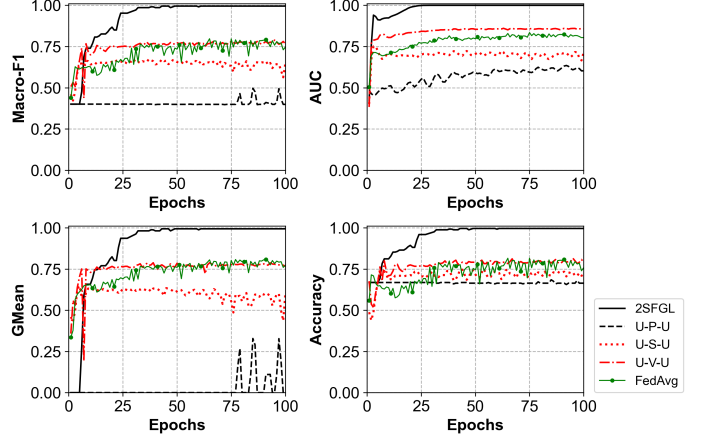


Fig. 5. Sensitivity analysis result of GCN with respect to epochs using FraudAmazonDataset.

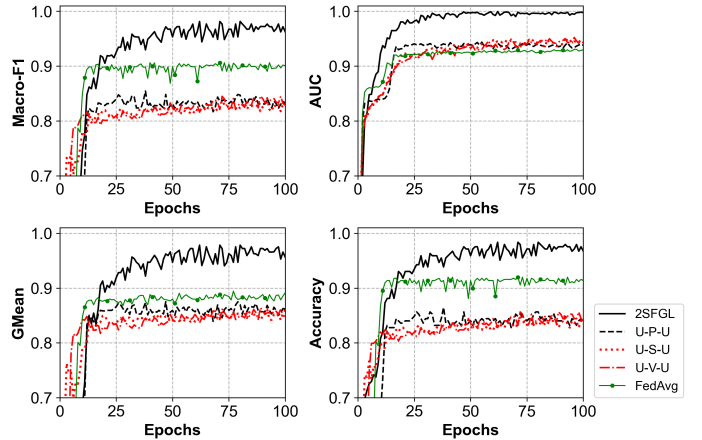


Fig. 6. Sensitivity analysis result of GraphSAGE with respect to epochs using FraudAmazonDataset.

Fig. 5 shows the performance of using the FraudAmazonDataset on virtual fusion graphs, FedAvg only, and three relational single local graphs using GCN, respectively. From Fig. 5, the solid black line represents results using the virtual fused graph on the 2SFGL. The green dotted solid line represents results using FedAvg. The black dashed line represents result using the dataset of U-P-U relationship. The red dotted line represents metrics result using U-S-U relationship. The red

dash dot represents metrics result using U-V-U relationship. Fig. 6 shows the performance of using GraphSAGE, differing from Fig. 5 in that the metrics are performed using GCN. We can observe that the model obtained from the GCN and GraphSAGE of metrics using 2SFGL are better than using FedAvg and performing GNN inference on the single local graphs. We provide a brief analysis of the different metrics, Fig. 5 shows that when the model is trained using only U-P-U graph data, the prediction results predict almost all negative samples as positive samples, and the True Positive (TP) hardly increases, which leads to almost zero GMean and almost a constant Macro-F1 metric for the GCN algorithm. The sharp increase in AUC when training using only U-P-U data is shown in Figure 6 because the model training has not yet converged. In addition, GraphSAGE is trained using random sampling and aggregation methods, while GCN utilizes the adjacency matrix of the graph, also because this difference leads to different results, which are also reflected in some other studies [42].

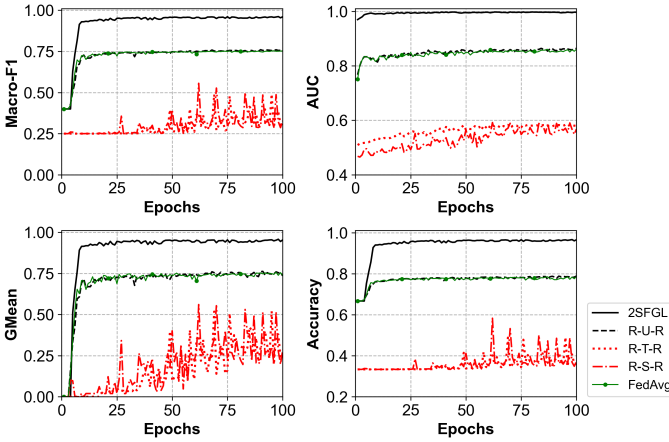


Fig. 7. Sensitivity analysis result of GCN with respect to epochs using FraudYelpDataset.

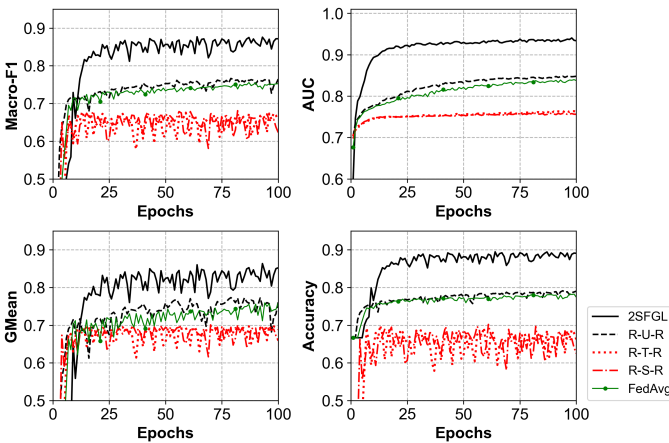


Fig. 8. Sensitivity analysis result of GraphSAGE with respect to epochs using FraudYelpDataset.

Similarly, using the FraudYelpDataset for comparison. Fig. 7 shows the performance of using the FraudYelpDataset on virtual fusion graphs, FedAvg only, and three relational single local graphs using GCN, respectively. Differing from Fig. 8 in that the metrics are performed using GCN. Overall, the metrics results using 2SFGL are higher than only using the FedAvg method.

In Table I, we provide the statistical averages of the different metrics for Fig. 5 and Fig. 6 from the 60th epochs to the 100th epochs of training using FraudAmazonDataset. The results from the metrics of 2SFGL with using virtual fusion in the First stage and using GCN-based FedAvg in the second stage is 6%-30.2% higher than that of using only GCN-based FedAvg. In Table II, The only difference from Table I is that the dataset used is the FraudYelpDataset. Similar results, using 2SFGL is 12%-28.4% higher than using only GCN-based FedAvg.

TABLE I
THE STATISTICAL AVERAGE OF THE METRICS USING
FRAUDAMAZONDATASET

GNN	Relationships	Macro-F1	AUC	GMean	Accuracy
GCN	2SFGL	0.99	1.0	0.99	0.99
	FedAvg	0.76	0.81	0.78	0.77
	U-P-U	0.41	0.60	0.04	0.66
	U-S-U	0.64	0.70	0.59	0.72
	U-V-U	0.77	0.86	0.77	0.79
GraphSAGE	2SFGL	0.97	0.99	0.96	0.97
	FedAvg	0.90	0.93	0.88	0.91
	U-P-U	0.83	0.93	0.86	0.84
	U-S-U	0.82	0.94	0.85	0.83
	U-V-U	0.83	0.94	0.85	0.84

TABLE II
THE STATISTICAL AVERAGE OF THE METRICS USING FRAUDYELPDATASET

GNN	Relationships	Macro-F1	AUC	GMean	Accuracy
GCN	2SFGL	0.96	1.0	0.95	0.96
	FedAvg	0.75	0.85	0.74	0.78
	R-U-R	0.75	0.86	0.74	0.78
	R-T-R	0.32	0.58	0.26	0.37
	R-S-R	0.33	0.55	0.27	0.38
GraphSAGE	2SFGL	0.86	0.93	0.83	0.88
	FedAvg	0.74	0.83	0.73	0.77
	R-U-R	0.75	0.84	0.75	0.78
	R-T-R	0.65	0.76	0.68	0.66
	R-S-R	0.65	0.76	0.68	0.66

VI. CONCLUSION

In this paper, we propose 2SFGL, a novel two-stage approach (the first stage is the virtual fusion of multiparty graphs, the second is model training and inference in the virtual graph using FedAvg) for increasing the accuracy of financial crime identification. The 2SFGL is implemented by combining graph-based learning and federated learning. Using FraudAmazonDataset to GCN on 2SFGL, metrics are 23.5%-30.2% better than GCN on only using FedAvg. While to GraphSAGE, improve the metrics by 6%-9.1%. Similar, Using

FraudYelpDataset to GCN on 2SFGL, metrics are 17.6%-28.4% better than GCN on only using FedAvg. While to GraphSAGE, improve the metrics by 12%-16.2%.

we are actively working on pilots, especially in banking institutions in the financial system. We will organize multiple financial institutions (e.g., banks) and communications department (e.g., operators) to participate in leveraging the federated graph learning framework for accurate detection of financial crimes.

ACKNOWLEDGMENT

REFERENCES

- [1] Toyotaro Suzumura, Yi Zhou, Natahalie Baracaldo, Guangnan Ye, Keith Houck, Ryo Kawahara, Ali Anwar, Lucia Larise Stavarache, Yuji Watanabe, Pablo Loyola, Daniel Klyashtorny, Heiko Ludwig, and Kumar Bhaskaran. Towards federated graph learning for collaborative financial crimes detection, 2019.
- [2] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, Feb 2010. [8] Masatoshi Hanai, Toyotaro Suzumura, Wen Jun Tan, Elvis S. Liu, Georgios Theodoropoulos, and Wentong Cai. Distributed edge partitioning for trillion-edge graphs. *CoRR*, abs/1908.05855, 2019.
- [3] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications, 2019.
- [4] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.
- [5] Cynthia Dwork. Differential privacy: A survey of results. In Manindra Agrawal, Dingzhu Du, Zhenhua Duan, and Angsheng Li, editors, *Theory and Applications of Models of Computation*, pages 1–19, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.
- [6] Wenliang Du, Yunghsiang Sam Han, and Shigang Chen. Privacy-preserving multivariate statistical analysis: Linear regression and classification. In *SDM*, 2004.
- [7] Abbas Acar, Hidayet Aksu, A. Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *CoRR*, abs/1704.03578, 2017.
- [8] Yoshinori Aono, Takuya Hayashi, Le Trieu Phong, and Lihua Wang. Scalable and secure logistic regression via homomorphic encryption. In *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy, CODASPY '16*, page 142–144, New York, NY, USA, 2016. Association for Computing Machinery.
- [9] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data, 2017.
- [10] Chulin Xie, Pin-Yu Chen, Ce Zhang, and Bo Li. Improving privacy-preserving vertical federated learning by efficient communication with admm. *arXiv preprint arXiv:2207.10226*, 2022.
- [11] Jakub Konecny, H. Brendan McMahan, Felix X. Yu, Peter Richtarik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *CoRR*, abs/1610.05492, 2016.
- [12] Meng Hao, Hongwei Li, Guowen Xu, Sen Liu, and Haomiao Yang. Towards efficient and privacy preserving federated deep learning. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–6, 2019.
- [13] Anusha Lalitha, Shubhanshu Shekhar, Tara Javidi, and Farinaz Koushanfar. Fully decentralized federated learning. In *Third workshop on Bayesian Deep Learning (NeurIPS)*, 2018.
- [14] Yang Zhao, Jun Zhao, Linshan Jiang, Rui Tan, Dusit Niyato, Zengxiang Li, Lingjuan Lyu, and Yingbo Liu. Privacy-preserving blockchain-based federated learning for iot devices, 2021.
- [15] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, May 2020.
- [16] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data, 2020.
- [17] Masatoshi Hanai, Toyotaro Suzumura, Wen Jun Tan, Elvis S. Liu, Georgios Theodoropoulos, and Wentong Cai. Distributed edge partitioning for trillion-edge graphs. *CoRR*, abs/1908.05855, 2019.
- [18] Koji Ueno, Toyotaro Suzumura, Naoya Maruyama, Katsuki Fujisawa, and Satoshi Matsuoka. Efficient breadth-first search on massively parallel and distributed-memory machines. *Data Science and Engineering*, 2:2–35, 2016.
- [19] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data Min. Knowl. Discov.*, 29(3):626–688, 2015.
- [20] Ian Molloy, Suresh Chari, Ulrich Finkler, Mark Wiggerman, Coen Jonker, Ted Habeck, Youngja Park, Frank Jordens, and Ron van Schaik. Graph analytics for real-time scoring of cross-channel transactional fraud. In Jens Grossklags and Bart Preneel, editors, *Financial Cryptography and Data Security - 20th International Conference, FC 2016, Christ Church, Barbados, February 22– 26, 2016, Revised Selected Papers*, volume 9603 of *Lecture Notes in Computer Science*, pages 22–40. Springer, 2016.
- [21] Weiyei Liu, Zhining Liu, Fucui Yu, Pin yu Chen, Toyotaro Suzumura, and Guangmin Hu. A scalable attribute-aware network embedding system. *Neurocomputing*, 339:279–291, 2019.
- [22] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Yu Rong, Peilin Zhao, Junzhou Huang, Murali Annaram, and Salman Avestimehr. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *CoRR*, abs/2104.07145, 2021.
- [23] Chuizheng Meng, Sirisha Rambhatla, and Yan Liu. Cross-node federated graph neural network for spatio-temporal data modeling. *ACM*, 2021.
- [24] Mark Weber, Jie Chen, Toyotaro Suzumura, Aldo Pareja, Tengfei Ma, Hiroki Kanezashi, Tim Kaler, Charles E. Leiserson, and Tao B. Schardl. Scalable graph learning for anti-money laundering: A first look. *CoRR*, abs/1812.00076, 2018.
- [25] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, and Charles E. Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. *CoRR*, abs/1902.10191, 2019.
- [26] Yingdong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S. Yu. Enhancing Graph Neural Network-Based Fraud Detectors against Camouflaged Fraudsters, page 315–324. *Association for Computing Machinery*, New York, NY, USA, 2020.
- [27] David Evans, Vladimir Kolesnikov, Mike Rosulek, et al. A pragmatic introduction to secure multi-party computation. *Foundations and Trends® in Privacy and Security*, 2(2-3):70–246, 2018.
- [28] Yehuda Lindell and Benny Pinkas. Secure multiparty computation for privacy-preserving data mining. *Journal of Privacy and Confidentiality*, 1(1), Apr. 2009.
- [29] Longfei Zheng, Jun Zhou, Chaochao Chen, Bingzhe Wu, Li Wang, and Benyu Zhang. Asfgnn: Automated separated-federated graph neural network, 2020.
- [30] Chuhan Fan, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. Fedgnn: Federated graph neural network for privacy-preserving recommendation, 2021.
- [31] Meng Jiang, Taeho Jung, Ryan Karl, and Tong Zhao. Federated dynamic GNN with secure aggregation. *CoRR*, abs/2009.07351, 2020.
- [32] Ziqi Liu, Chaochao Chen, Xinxing Yang, Jun Zhou, Xiaolong Li, and Le Song. Heterogeneous graph neural networks for malicious account detection. *CoRR*, abs/2002.12307, 2020.
- [33] Daixin Wang, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, and Yuan Qi. A semi-supervised graph attentive network for financial fraud detection. *CoRR*, abs/2003.01171, 2020.
- [34] Jianyu Wang, Rui Wen, Chunming Wu, Yu Huang, and Jian Xion. Fdgars: Fraudster detection via graph convolutional networks in online app review system. In *Companion The 2019 World Wide Web Conference*, 2019.
- [35] Zhiwei Liu, Yingdong Dou, Philip S. Yu, Yutong Deng, and Hao Peng. Alleviating the inconsistency problem of applying graph neural network to fraud detection. *CoRR*, abs/2005.00625, 2020.
- [36] Benny Pinkas, Thomas Schneider, Gil Segev, and Michael Zohner. Phasing: Private set intersection using permutation-based hashing. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 515–530, Washington, D.C., August 2015. USENIX Association.
- [37] Zhou J, Cui G, Hu S, et al. Graph neural networks: A review of methods and applications[J]. *AI Open*, 2020, 1: 57–81.
- [38] Yingdong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S. Yu. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM International*

Conference on Information & Knowledge Management. ACM, oct 2020.

- [39] Shebuti Rayana and Leman Akoglu. Collective opinion spam detection: Bridging review networks and metadata. Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015.
- [40] M. Kubat and S. Matwin, “Addressing the curse of imbalanced training sets: One-sided selection,” in In Proceedings of the Fourteenth International Conference on Machine Learning. Morgan Kaufmann, 1997, pp.179–186.
- [41] De Diego, I.M., Redondo, A.R., Fernández, R.R. et al. General Performance Score for classification problems. Appl Intell 52, 12049–12063 (2022). <https://doi.org/10.1007/s10489-021-03041-7>.
- [42] Abhishek Paudel, Roshan Dhakal, and Sakshat Bhattarai. Room classification on floor plan graphs using graph neural networks. arXiv preprint arXiv:2108.05947, 2021