Inductive Link Prediction Banking Fraud Detection System Using Homogeneous Graph-Based Machine Learning Model

1st Hilmi Aziz Bukhori
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
23521017@std.stei.ac.id

2st Rinaldi Munir
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
rinaldi@informatika.org

***Abstract*—Graph machine learning and fraud detection systems are growing and popular today. Fraud detection systems have been widely used as a tool to detect potentially fraudulent transactions. Fraud detection systems can be used to determine patterns of transactions that are suspected of being criminal transactions. Graph machine learning development can be implemented in anything that can be represented in graph form. The banking fraud detection system can be implemented in graph form by connecting customers who have made transactions with other customers or customer transactional activities. From the graph that has been formed, predictions will be made so that new transactions can be classified as fraudulent transactions or not by connecting these transactions with the graphs that have been made. The experimental results show that the graph-based fraud detection model produces better performance than the tree-based fraud detection model, but with a longer inference time.**

Keywords— banking fraud detection, graph-based fraud detection, classification, tree-based fraud detection , inference time

I. Introduction

Fraud is an activity carried out to gain financial gain that is carried out fraudulently[1]. Fraud against an organization is a threat to the status of the organization and its interactions with stakeholders outside the organization, such as customers, suppliers, investors and business partners. Fraud can result in enormous financial damage [2] . Fraud in the banking industry is a growing problem with enormous consequences for banks and customers, both in terms of financial loss, trust and credibility [3]. There are many types of bank fraud including check fraud, debit and credit card fraud. inflated contracts, financial statement fraud, health insurance fraud, auto insurance fraud, and mortgage insurance fraud [4].

Fraudulent behavior in transactions can be detected as suspicious transactions [5]. Suspicious transaction detection is an effective anomaly detection as a learning process that can be integrated into a solution for finding fraud [6].

The combination of this transaction data with the latest machine learning (ML) techniques has made it possible to automatically identify fraudulent transactions. While many researchers have developed and implemented machine learning algorithms for fraud detection [7], some of them specifically use random forests, decision trees, and Support Vector Machines. some of them specifically use random forests [8][9][10], decision trees [11][12][13], and Support Vector Machines [14][15]. They often rely on artificial feature engineering, are case dependent, and difficult to optimize. In addition, although recent research has shown that fraudulent behavior has important social effects, both the social and temporal aspects of behavior, the aspect of network correlation between existing data related to behavior that is considered fraudulent is often ignored [16].

The graph-based anomaly detection approach is one of the most popular techniques used to analyze communication network patterns and can be used to identify suspicious behavior [17]. Identification of fraud is a procedure for determining data points that have different behavior from normal behavior.

Fraud on a graph can be defined as a single data point that has deviant behavior from other networks [18]. Fraud detection links various correlated data to group nodes based on previously identified data. The use of graphs can accommodate data structures that vary and have complex dimensions [19]. Previous anomaly detection methods have utilized user attributes and historical behavior to detect fraud in banking by utilizing traditional machine learning where its use is only limited to features in user data [20]. Graph Neural Network (GNN) is one of the benefits of using a neural network that utilizes a data structure in the form of a graph [21]. The use of this method can be implemented in fraud detection cases where the implementation will consider the correlation or linkages between nodes or data.

This research will focus on the implementation of the Graph-Based Machine Learning Model which will be compared to the traditional tree-based machine learning model to determine accuracy and inference time in banking fraud detection case studies. The rest of this paper is organized as follows: The related works are discussed in section 2. Section 3 provides the proposed model of banking fraud detection. Section 4 discusses the results and analysis of tree based model and graph-based model. Finally, discussion and conclusion are given in section 5 and section 6.

II. Related Works

Since the success of deep learning, many studies have studied how the structure of graphs can be implemented into artificial neural networks. The GNN-based system (Graph Neural Network) has shown many uses because it can be used in graph data structures such as protein structures and knowledge graph [21]. Our contribution mainly focuses on creating graph-based fraud detection system model and treebased fraud detection system model to detect potentially fraudulent transactions. Graph-based machine learning models that will be used are GraphSAGE [22], GAT [23] and

GCN [24]. Tree-based machine learning models that will be used are Random Forest [25], LightGBM [26] and XGBoost [27].

The graph-based fraud detection system will use existing methods in graph-based machine learning. Methods in graphic-based machine learning include edge prediction, node classification and graph classification[28]. In the case of edge prediction with one type of transaction, we can do it inductively or transductively. In this paper, a graph-based model will be made to predict edges inductively. Inductive inference is done in batches, i.e. the input is n transactions. After the graph learns how to predict the edges of the train dataset, then inference will be made on the test dataset of n transactions. Inferences from fraud or not fraudulent transactions will be seen from features on the data transactions. The graph used in this study is defined as *(1)* where G is an directed graph, V is defined as (2), and E is defined as (3) where u_x and w_q are transactions and will be included in the set *E* if both vertices connected.

$$G = (V, E) \tag{1}$$

$$V = \{Transaction1, Transaction2, \dots, TransactionN\} \tag{2}$$

$$E = \{(t_1, w_1), (t_2, w_2), \dots, (t_p, w_q)\} \tag{3}$$

A. Directed Homogeneous Graph

The use of types in graphs based on data scalability greatly influences the modeling that will be carried out. The graph type is an input data structure which becomes input for GNN processing later [29]. In this study, directed graphs are used where the edges in a directed graph are all directed from one node to another, which provides more information than an undirected graph. Each edge in an undirected graph can also be considered as two directed edges. In addition, the node scheme used is Homogeneous Graphs, where the nodes and edges in a homogeneous graph have the same type in features and behavior treatment at these nodes and edges. The selection of graph, node, and edge schemes is based on the shape of the dataset owned so that system detection can be maximized because it has a graph scheme that matches the dataset used.

B. Link Prediction

Graph Neural Network (GNN) has been implemented for various needs and tasks. GNN functionality can be classified according to the scope of work on graph systems, namely node level, edge level, or graph level [30]. In this research, we will implement link prediction at the edge level which will label existing links with a class of binary classification categories indicating whether the transaction is categorized as fraud or not.

C. Tree Based Algorithm

The tree-based machine learning method is a branch of supervised learning that offers a variety of flexible methods built on a common framework [31]. The tree-based method is capable of handling diverse data without the need for in-depth pre-processing. The tree-based method also has fast and accurate computations. In this study, three tree-based machine learning methods will be used, including Random Forest, LightGBM, and XGBoost.

III. The Proposed Model of Banking Fraud Detection

This section details the building blocks for applying representational learning in fraud detection settings. Tabular data structures are explicitly encoded into a graph format. Then, each graph node is converted into a vector through a representational learning algorithm which will be represented as nodes and edges. The generalization of embedding this algorithm to invisible nodes is achieved with the help of inductive extensions

A. Graph Structure

The graph structure representation that will be used in this study will consist of transactional data that will be connected by two customers with detailed transaction data that will be used as a feature on the edge graph.

This idea is also relevant for fraud detection, as it seems attractive to explicitly add historical fraud information to transaction data. With the use of linkages between transaction data that occur between customers, patterns of fraud detected will be known.

Figure 1 shows the graph scheme that will be used for fraud detection where customers are nodes and transactions are edges

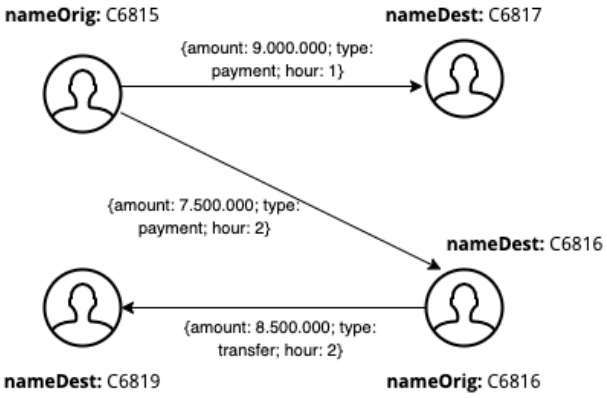

Fig. 1. Graph Based Model Inference Time

B. Inductive Representative Learning.

The need for inductive representation learning can be attributed to timeliness constraints in many application domains. Fraud detection is no exception. When transactions are processed, authorization is required in seconds. As discussed earlier, existing representational learning techniques are only transductive, in that the entire graph is only fed once to the representational learner, which results in only one set of embeddings. Adding or removing nodes or edges to the network, requires repeating the entire process. The associated computational complexity hinders the timely completion of the predictions.

Graphsage, GCN, and GAT [7] are node insertion algorithms capable of generating embeds for invisible nodes. Therefore, this technique is an obvious choice to overcome the deception induction problem. The graph method avoids iteration by learning a function that takes the environment node attribute as input and produces embedding as output. The graph method offers a variety of feature aggregation possibilities to be applied to node attributes.

We describe in Figure 2 about simple procedure for creating embeddings for invisible edge/node. Node/edge, i.e.

a new credit card transaction. The basic idea is to run the embedding algorithm on a sufficient amount of training data. For each new graphic element added to the original graphic, a new embedding is created by reusing the already derived embedding for the existing graphic element.

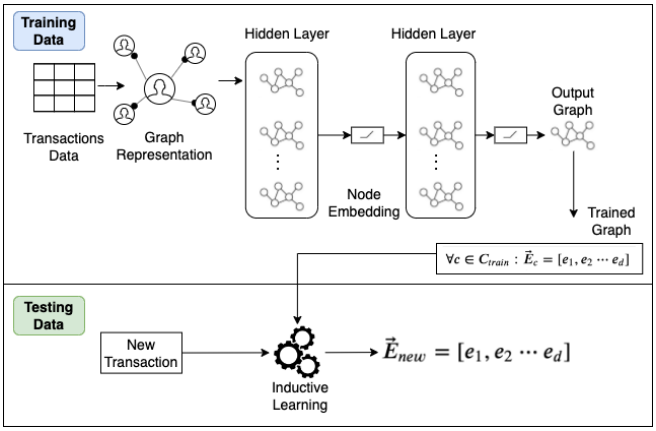

Fig. 2. Inductive Embedding

C. Evaluation of Machine Learning Models.

The evaluation metric used in this study is the Area Under the Curve (AUC) value of the Receiver Operating Characteristics (ROC) curve as in (4). ROC is a probability curve, while AUC is the degree of separation between the prediction results. The value of AUC will describe how precisely the model separates between classes. A high AUC value means that the model can predict well class 0 data with class 0 predictions as well as class 1 data with class 1 predictions. The curve of the ROC is depicted by the false positive rate (FPR) as the X axis and the true positive rate (TPR) as the Y.

$$ROC - AUC = \int_0^1 TPR(FPR)dFPR$$

$$= \int_0^1 TPR(FPR^{-1}(x))dx \quad (4)$$

IV. Result and Analysis

A. Experiment Setup

The dataset used is a dataset regarding credit-card user which is customer transaction data that has fraud and non-fraud labels. The preprocessing is carried out in two steps, the first is to combine the columns of sending and receiving customers to serve as table nodes, the second is to remove outliers where there is data that has a transaction amount of 0. Some samples related to the dataset that will be used for fraud detection are shown in Table I

TABLE I. Dataset Transactions of Credit Card

Step	Type	Amount	nameOrig	nameDest	isFraud
1	Payment	1300000	CDU1837	CYS8364	1
2	Transfer	100000	CYR1534	CTR1837	0
3	Payment	120000	CSY2847	CUY1837	1

The specifications provided about the machine that will be used for the experiments are 4vCPUs with 32 gigabytes (GB) memory. We will determine the parameters for tree-based model with the best performance or hyperparameter tuning by using GridSearchCV. There will be no hyperparameter tuning using GridSearchCV for a graph-based model because it cannot accept a graph-based model. In order for the experiment to be carried out the same for all models, it is necessary to determine a parameter that has a fixed value for the entire model. In the tree-based model, we can use min data in leaf and learning rate to be fixed-valued parameters for the three tree-based models. The dataset will be divided into two parts, test dataset and train dataset. The comparison of the dataset used in this experiment is 70% train data and 30% test data [32]. Details about the experimental setup on the tree based model and graph based model are shown in Table II, Table III, Table IV, Table V, Table VI, Table VII

TABLE II. Experiment Setup Tree Based Model

No	Model	Used Parameter		Dataset	
		Learning Rate	*Min data in leaf*	*Train*	*Test*
1	Random Forest	-	20	70%	30%
2	LightGBM	0.2	20	70%	30%
3	XGBoost	0.2	-	70%	30%

TABLE III. Experiment Setup Graph Based Model

No	Model	Used Parameter			Dataset	
		Optimizer	*Learning Rate*	*Dropout*	*Train*	*Test*
1	GraphSage	Adam	0.001	0.3	70%	30%
2	GAT	Adam	0.001	0.3	70%	30%
3	GCN	Adam	0.001	0.3	70%	30%

TABLE IV. Max Depth & N Estimator Variation on Tree Based Model

Model	Max Depth Variation	N Estimators Variation
Random Forest	10, 20, 40, 80	25, 50, 100, 200
LightGBM	10, 20, 40, 80	25, 50, 100, 200
XGBoost	10, 20, 40, 80	25, 50, 100, 200

TABLE V. Inference Time Experiment on Tree Based Model

Model	Max Depth Variation	N Estimators Variation
Random Forest	10	25, 50, 100, 200
LightGBM	10	25, 50, 100, 200
XGBoost	10	25, 50, 100, 200

TABLE VI. Epoch Layer Variation on Graph Based Model

Model	Epoch Variation	Layer Variation
GraphSage	20, 40, 60, 80	2, 3, 4, 5
GAT	20, 40, 60, 80	2, 3, 4, 5
GCN	20, 40, 60, 80	2, 3, 4, 5

TABLE VII. Inference Time Experiment on Graph Based Model

Model	Layer	Epoch Variation
GraphSage	2	20, 40, 60, 80
GAT	2	20, 40, 60, 80
GCN	2	20, 40, 60, 80

B. Experiment Scenario

In this experiment, three graph-based models and three treebased models will be created and implemented. There are

several parameters that are varied to see the best performance of the model. The variation of the selected is the value taken because the resources of the machine are limited. The max depth, n estimators, epoch and layer experiments will be performed 10 times.

There are three experiments to be carried out in a tree-based model. This experiment will vary a parameter. The parameters variation used in the tree-based experiment are max depth and n estimators (see Table IV). For the max depth experiment, the value of n estimators used is 100. For the n estimators experiment, the value of max depth used is 10. The last experiment is inference time of tree-based model (see Table IV).

There are three experiments to be carried out in a graphbased model. The parameters used in graph-based experiments are epoch and layer (see Table VI). Epoch and layer were selected as parameters to be varied to see whether the addition of these two parameters will affect the performance of the model or not. For the epoch variation experiment, the value of layer used is 2. For the layer variation experiment, the epoch value used is 40. The last experiment is the inference time of model (see Table VII).

C. Experiment Result

In this section, the results of the experiments that have been carried out will be explained. There are six experiments that have been done. Evaluation of the performance of each experiment will used the AUC value. The evaluation of the inference time will use the units of milliseconds. Evaluation will be carried out on the testing data.

1) Max Depth Experiment on Tree Based Model

In this experiment, it was found that LightGBM has the highest performance with an AUC value of 0.866 when max depth is 40. It can be seen in Table VIII that increasing max depth in the Random Forest and XGBoost model will not affect the model's performance. LightGBM is model that have lower performance variance than Random Forest and XGBoost. This shows that LightGBM is more stable than Random Forest and XGBoost in the max depth experiment. From this experiment, it was found that increasing max depth in LightGBM will improve the performance of the model.

TABLE VIII. AUC SCORE MAX DEPTH EXPERIMENT

Max Depth	Random Forest	LightGBM	XGBoost
10	0.828	0.817	0.847
20	0.827	0.854	0.855
40	0.829	0.866	0.848
80	0.828	0.856	0.851

2) N Estimator Experiment on Tree Based Model

Overall it was found that LightGBM got the highest performance with an AUC value of 0.876 when n estimators is 100. Table IX shows that LightGBM gets the best performance compared to XGBoost and Random Forest in almost all variations of n estimators, namely 25, 50, 100 and 200. From the experiment we got that Random Forest gets the worst performance, which is the maximum AUC value obtained is 0.830. This experiment shows that LightGBM and XGBoost are more stable than Random Forest. From this experiment, it was found that increasing the value of n estimators in LightGBM and XGboost will increase the performance of the model in most of n estimator.

TABLE IX. AUC SCORE N ESTIMATOR EXPERIMENT

N Estimator	Random Forest	LightGBM	XGBoost
25	0.828	0.837	0.838
50	0.827	0.854	0.845
100	0.827	0.876	0.857
200	0.830	0.873	0.859

3) Epoch Experiment on Graph Based Model

The AUC values from the classification of graph-based models can be seen in Table X shows the result for GraphSAGE, GAT and GCN. From the classification results, it was found that the highest value obtained in the epoch experiment was GraphSAGE with an AUC value of 0.989. In this experiment, it was found that the more epoch used, the better the performance for classifying customers for the GCN model. Meanwhile, it can be seen that for the GraphSAGE and GAT models there is an increase or decrease in the average value but not significant. In this experiment, it was also found that GraphSAGE obtained more stable results than GAT and GCN.

TABLE X. AUC SCORE EPOCH EXPERIMENT

Epoch	GraphSage	GAT	GCN
20	0.985	0.842	0.676
40	0.987	0.858	0.895
60	0.985	0.851	0.926
80	0.989	0.859	0.936

4) Layer Experiment on Graph Based Model

The AUC values from the classification of graph-based models can be seen in the Table XI for GraphSAGE, GAT and GCN. The classification results show that the best value is obtained by GraphSAGE when there are 2 layers with an AUC value of 0.985. Therefore, GraphSAGE is the most efficient graph method compared to the other two methods. Sequentially, it was found that the best performance on the graph-based model in the layer experiment are GraphSAGE, GCN and GAT. From this experiment, it was found that increasing the layer on the graph-based model will decrease the classification performance for customers.

TABLE XI. AUC SCORE LAYER EXPERIMENT

Layer	GraphSage	GAT	GCN
2	0.985	0.903	0.888
3	0.902	0.805	0.725
4	0.859	0.794	0.747
5	0.764	0.694	0.542

5) Inference Time Experiment

There are 2 experiment results in this experiment. The first experiment is time inference graph-based model. The second experiment is time inference tree-based model. In this experiment, it was found that GCN has the fastest inference time at Table XII. GraphSAGE takes longer time because it uses the LSTM function so that it processes input sequentially. It can also be seen in the table that GAT gets an inference time that is between GCN and GraphSAGE. From this experiment, it was found that increasing epoch on all graph-based models did not significantly affect the inference time. In the tree-based model experiment, it was found that the inference time was faster than the graph-based model at Table XIII. LightGBM has the lowest inference times for almost all n estimators then followed by XGBoost and Random Forest. From this experiment, it was found that increasing n estimators in a treebased model can make the inference time longer.

TABLE XII. GRAPH BASED MODEL INFERENCE TIME

Epoch	GraphSage	GAT	GCN
20	6,183 ms	4,847 ms	3,385 ms
40	7,183 ms	3,837 ms	3,353 ms
60	6,384 ms	3,756 ms	3,534 ms
80	6,284 ms	3,645 ms	3,423 ms

TABLE XIII. TREE BASED MODEL INFERENCE TIME

N Estimator	Random Forest	LightGBM	XGBoost
25	9.6 ms	3.98 ms	7.39 ms
50	11.8 ms	4.85 ms	8.38 ms
100	24.8 ms	8.58 ms	11.45 ms
200	34.8 ms	11.2 ms	8.69 ms

V. DISCUSSION

A comparison analysis of the AUC value will be carried out to see a better model performance. The AUC value used is the highest average of the entire model. Inference time comparison will also be seen to see which model is faster. The comparison of the graph-based model and the tree-based model can be seen in Table XIII. In the case with data in the form of fraud detection for banking transactions, it was found that the graph-based model had better performance than the tree-based model. The inference time of the graph-based model is the inference time of the edge predictions added to the inference time for classification. From this comparison, it is found that the tree-based model has a faster inference time than the graph-based model. Graph-based models have a longer inference time because the model requires more processes so it takes longer to classify.

In this experiment it was found that GraphSAGE has better performance than GAT and GCN on graph-based machine learning. This performance difference is due to the computations performed on GraphSAGE that do not give weight to the features of its neighbors, so the results obtained are better than GCN. In addition, GraphSAGE will work directly by concatenating the aggregated results of neighboring features to its own features so that the resulting model will be based on non-normalized features such as GAT. The learning process in the GAT model is also carried out by selecting heads randomly where in GraphSAGE the selection of layers depends on the selected destination node, namely the layer will contain neighbors of the selected nodes so that it will greatly affect the performance of the model as a whole.

TABLE XIV. COMPARISON OF GRAPH BASED MODEL AND TREE BASED MODEL

Params	Random Forest	Light GBM	XG Boost	Graph Sage	GAT	GCN
AUC Score	0.831	0.863	0.862	0.974	0.901	0.969
Inference Time	8.7 ms	3.96 ms	6.97 ms	5,630 ms	2,770 ms	2,296 ms

VI. CONCLUSSION

In general, it can be concluded that in case of fraud detection with dataset credit card customers, graph-based model has better performance than the tree-based model, but the graph-based model has a longer inference time than the tree-based model. From the results of the variation experiment, it is found that there are parameters that can significantly affect the performance of a model and parameters that do not significantly affect the performance of the model. It was found that layer will affect the performance of the overall graph-based model and epoch will affect GCN significantly but not with GraphSAGE and GAT. It was also found that the max depth variation will only significantly affect the LightGBM model and the number of tree variation will significantly affect the performance of the LightGBM and XGBoost models.

The weakness of the graph structure algorithm is the high complexity of the graph which results in longer inference time compared to tree-based algorithms. In the future, a graph-based algorithm can be developed that can speed up inference time so that it can be optimized in machine learning tasks

ACKNOWLEDGMENT

This research was financed by "Lembaga Pengelola Dana Pendidikan. LPDP, Kementerian Keuangan, Indonesia".

REFERENCES

[1] K. Worobec, “Fraud - The Facts 2021,” UK Financ., 2021, [Online]. Available: https://www.ukfinance.org.uk/policy-and-guidance/reports-publications/fraud-facts-2021.

[2] L. Schaedler, L. Graf-Vlachy, and A. König, “Strategic leadership in organizational crises: A review and research agenda,” Long Range Plann., vol. 55, no. 2, 2022, doi: 10.1016/j.lrp.2021.102156.

[3] H. van Driel, “Financial fraud, scandals, and regulation: A conceptual framework and literature review,” Bus. Hist., vol. 61, no. 8, pp. 1259–1299, 2019, doi: 10.1080/00076791.2018.1519026.

[4] I. Eweoya, A. A. Ayodele, A. Azeta, and O. Olatunji, “Fraud prediction in bank credit administration: A systematic literature review,” J. Theor. Appl. Inf. Technol., vol. 97, no. 11, pp. 3135–3157, 2019.

[5] R. Kian, “Detection of Fraud in Banking Transactions Using Big Data Clustering Technique Customer Behavior Indicators,” J. Appl. Res. Ind. Eng., vol. 9, no. 3, pp. 264–273, 2022.

[6] M. Srokosz, A. Bobyk, B. Ksiezopolski, and M. Wydra, “Machine-Learning-Based Scoring System for Antifraud CISIRTs in Banking Environment,” Electron., vol. 12, no. 1, pp. 1–17, 2023, doi: 10.3390/electronics12010251.

[7] R. Van Belle, V. Belle, and S. Mitrovi, “Representation Learning in Graphs for Credit Card Fraud Detection Representation Learning in

Graphs for Credit Card Fraud Detection," no. March, 2020, doi: 10.1007/978-3-030-37720-5.

[8] M. S. Kumar, V. Soundarya, S. Kavitha, E. S. Keerthika, and E. Aswini, "Credit Card Fraud Detection Using Random Forest Algorithm," 2019 Proc. 3rd Int. Conf. Comput. Commun. Technol. ICCCT 2019, no. 3, pp. 149–153, 2019, doi: 10.1109/ICCCT2.2019.8824930.

[9] P. T. Varma, M. Poojari, J. Joseph, and A. Cardozo, "Credit Card Fraud Detection Using Random Forest Algorithm," Int. J. Trendy Res. Eng. Technol., vol. 05, no. 03, pp. 24–27, 2021, doi: 10.54473/ijtret.2021.5305.

[10] M. S. Kumar, V. Soundarya, S. Kavitha, E. S. Keerthika, and E. Aswini, "Credit Card Fraud Detection Using Random Forest Algorithm," 2019 Proc. 3rd Int. Conf. Comput. Commun. Technol. ICCCT 2019, pp. 149–153, 2019, doi: 10.1109/ICCCT2.2019.8824930.

[11] P. T. Varma, M. Poojari, J. Joseph, and A. Cardozo, "Credit Card Fraud Detection Algorithm using Decision Treesbased Random Forest Classifier," Int. J. Trendy Res. Eng. Technol., vol. 05, no. 03, pp. 24–27, 2021, doi: 10.54473/ijtret.2021.5305.

[12] J. A. Hudali, K. P. Mahalaxmi, N. S. Magadum, and S. Belagali, "Credit Card Fraud Detection by using ANN and Decision Tree," Hbrppublication.Com, vol. 2, no. 3, pp. 1–4, 2019, [Online]. Available: https://ieeexplore.ieee.org/xpl/conhome.

[13] M. R. Dileep, A. V. Navaneeth, and M. Abhishek, "A novel approach for credit card fraud detection using decision tree and random forest algorithms," Proc. 3rd Int. Conf. Intell. Commun. Technol. Virtual Mob. Networks, ICICV 2021, no. Icicv, pp. 1025–1028, 2021, doi: 10.1109/ICICV50876.2021.9388431.

[14] . A., "Credit Card Fraud Detection using Machine Learning and Data Science," Int. J. Res. Appl. Sci. Eng. Technol., vol. 9, no. VII, pp. 3788–3792, 2021, doi: 10.22214/ijraset.2021.37200.

[15] S. R. Prusty, B. Sainath, S. K. Jayasingh, and J. K. Mantri, "SMS Fraud Detection Using Machine Learning," Lect. Notes Networks Syst., vol. 431, pp. 595–606, 2022, doi: 10.1007/978-981-19-0901-6_52.

[16] Y. Lucas et al., "Towards automated feature engineering for credit card fraud detection using multi-perspective HMMs," Futur. Gener. Comput. Syst., vol. 102, pp. 393–402, 2020, doi: 10.1016/j.future.2019.08.029.

[17] T. Pourhabibi, K. L. Ong, B. H. Kam, and Y. L. Boo, "Fraud detection: A systematic literature review of graph-based anomaly detection approaches," Decis. Support Syst., vol. 133, no. April, 2020, doi: 10.1016/j.dss.2020.113303.

[18] Tahereh Pourhabibi, "Fraud Detection: A Graph Based Anomaly Detection Approach," RMIT University, 2021.

[19] X. Ma et al., "A Comprehensive Survey on Graph Anomaly Detection with Deep Learning," IEEE Trans. Knowl. Data Eng., no. August, 2021, doi: 10.1109/TKDE.2021.3118815.

[20] M. Lokanan, V. Tran, and N. H. Vuong, "Detecting anomalies in financial statements using machine learning algorithm: The case of Vietnamese listed firms," Asian J. Account. Res., vol. 4, no. 2, pp. 181–201, 2019, doi: 10.1108/AJAR-09-2018-0032.

[21] J. Zhou et al., "Graph neural networks: A review of methods and applications," AI Open, vol. 1, no. January, pp. 57–81, 2020, doi: 10.1016/j.aiopen.2021.01.001.

[22] W. L. Hamilton, "Inductive Representation Learning on Large Graphs," in 31st Conference on Neural Information Processing Systems (NIPS 2017), 2017, no. 920, p. 59.

[23] P. Veličković, A. Casanova, P. Liò, G. Cucurull, A. Romero, and Y. Bengio, "Graph attention networks," 6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc., pp. 1–12, 2018, doi: 10.1007/978-3-031-01587-8_7.

[24] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: a comprehensive review," Comput. Soc. Networks, vol. 6, no. 1, 2019, doi: 10.1186/s40649-019-0069-y.

[25] F. T. Kurdi, "Random Forest Machine Learning Technique for Automatic Vegetation Detection and Modelling in LiDAR Data," Int. J. Environ. Sci. Nat. Resour., vol. 28, no. 2, pp. 10–12, 2021, doi: 10.19080/ijesnr.2021.28.556234.

[26] X. Zhang, "A Model Combining LightGBM and Neural Network for High-frequency Realized Volatility Forecasting," Proc. 2022 7th Int. Conf. Financ. Innov. Econ. Dev. (ICFIED 2022), vol. 648, no. Icfied, pp. 2906–2912, 2022, doi: 10.2991/aebmr.k.220307.473.

[27] M. Vaudevan, R. S. Narayanan, S. F. Nakeeb, and Abhishek, "Customer churn analysis using XGBoosted decision trees," Indones. J. Electr. Eng. Comput. Sci., vol. 25, no. 1, pp. 488–495, 2022, doi: 10.11591/ijeecs.v25.i1.pp488-495.

[28] Z. Qin, Y. Liu, Q. He, and X. Ao, Explainable Graph-based Fraud Detection via Neural Meta-graph Search, vol. 1, no. 1. Association for Computing Machinery, 2022.

[29] T. D. Nguyen, T. Le-Cong, T. V. H. Nguyen, X. B. D. Le, and Q. T. Huynh, Toward the Analysis of Graph Neural Networks, vol. 1, no. 1. Association for Computing Machinery, 2022.

[30] Y. Yang, Y. Liang, and M. Zhang, "PA-GNN : Parameter-Adaptive Graph Neural Networks," 2022.

[31] Y. Shen and W. Cheng, "A Tree-Based Machine Learning Method for Pipeline Leakage Detection," Water (Switzerland), vol. 14, no. 18, 2022, doi: 10.3390/w14182833.

[32] B. Genç and H. Tunç, "Optimal training and test sets design for machine learning," Turkish J. Electr. Eng. Comput. Sci., vol. 27, no. 2, pp. 1534–1545, 2019, doi: 10.3906/elk-1807-212.