Advances in Artificial Intelligence, Big Data and Algorithms
G. Grigoras and P. Lorenz (Eds.)

doi:10.3233/FAIA230858

Fraud Detection Based on Graph Neural Network

Tao ZHANG[1] and Haicheng YE[2]
North China University of Technology, Beijing, China

Abstract. With the rapid development of digital and intelligent credit industry, credit fraud detection has become an important task to ensure financial digital intelligence. The traditional credit fraud detection model relies on artificial feature engineering and is built based on supervised learning algorithm, and its performance is greatly affected by data quality, sample distribution and other factors, and it is prone to prediction errors in the face of emerging fraud techniques. With the development of digitalization and fraud methods, these methods are often no longer applicable. There are rich information associations among users such as the users' emergency contacts, home address, and social relationships between themselves, which make a large social network graph formed between users. In this regard, based on the Dgraph-Fin dataset, this paper uses the relationship network formed among users to better learn the weights between different edges through neighbor sampling and attention mechanism. Experimental results show that the accuracy and effectiveness are improved compared with the existing baseline.

Keywords. Fraud detection, Graph convolutional neural network, DGraph-Fin, Neighbor sampling, attention mechanism

1. Introduction

The digital development of credit has prompted abundant offline loan businesses to gradually migrate to online loans. One of the benefits is that it greatly promotes financial and economic development, however, the new fraud methods have emerged. The traditional credit fraud detection model is formed by the feature selection and construction of experts in the field, which leads to the lack of flexibility and universality of the model, and it is difficult to deal with the complex and changeable fraud behavior. Most of these traditional fraud detection models are built based on supervised learning algorithms, whose performance is greatly affected by data quality, sample distribution and other factors, which are easy to make prediction errors in the face of emerging fraud techniques. For example, the linear model cannot effectively capture the nonlinear relationship between the sample features and ignores the rich information of the data, resulting in poor performance of the model. Due to the small number of features, it is often possible to model directly with data sets. However, as the number of features increases, the dataset becomes sparse, resulting in reduced prediction performance.

[1] Corresponding Author: Tao ZHANG, North China University of Technology;
e-mail: zhangtony8888@qq.com
[2] Haicheng YE, North China University of Technology; e-mail: 1203204606@qq.com

During the development of artificial intelligence technology, machine learning models such as Random Forest [1] and XGBoost [2] and deep learning models such as neural networks begin to conduct multidimensional analysis of user data. At present, these models are analyzed and modeled through the dimensional characteristics of the sample data itself to find the commonality between samples, that is, the difference between normal and outlier values. It is often assumed that there is no correlation between the samples. However, in the actual scenario, there are often many relevant features of entity information in the user information, such as emergency contacts, home addresses, etc., which makes a large financial social network graph [3] formed between different users. Fraud detection based on graph neural networks has the following advantages: financial data mostly has correlations between entities, which can better handle financial social network data. Graph neural networks can model the relationships between entities, accurately describe the effects and connections between different entities, and capture richer feature information [8] . Traditional models often only focus on the dimensional features of the samples themselves, while graph neural networks can extract more comprehensive and rich feature information by learning the features of graph elements such as nodes and edges. The feature information includes node degrees, neighbor feature information, graph structure information, etc., which can better reflect the relationships and impacts between entities, thereby improving the accuracy and robustness of fraud detection. Moreover, compared to traditional models, graph neural networks are easier to understand and interpret, and have stronger interpretability and visualization [9]. By visualizing node representations, we can more intuitively understand the connections and impacts between different entities, explore the relationship network between entities, and better discover fraudulent behavior and abnormal patterns.

Attention mechanisms can be employed in the GraphSAGE algorithm to guide the allocation of importance weights to different neighbor nodes during the aggregation process. By learning these weights, the model can focus more on the information contributed by or relevant to certain neighbor nodes. This improves the accuracy and discriminability of node representations. When applying attention mechanisms, GraphSAGE typically assigns an attention weight to each neighbor node, which is computed based on the relationship, features, etc., between the target node and its neighbor nodes. Then, the features of neighbor nodes are weighted aggregated according to these weights to obtain the representation of the target node. By incorporating attention mechanisms, GraphSAGE can effectively handle large-scale graph data and provides more accurate node representation learning capabilities.

The GraphSAGE (Graph Sample and Aggregated) algorithm obtains a subgraph of neighbors for a node through neighbor sampling. This means that in each training step, only a subset of neighboring nodes around each node is considered instead of the entire graph. By using neighbor sampling, GraphSAGE can handle large-scale graph data and reduce computational and storage costs. During the sampling process, different sampling strategies can be employed, such as random sampling, biased sampling, or degree-based sampling. This can be adjusted based on the specific task and characteristics of the dataset. Neighbor sampling enables GraphSAGE to learn node representations more efficiently. By aggregating information from a subset of neighbor nodes, GraphSAGE can generate compact and expressive node embeddings while retaining important graph structural information. Therefore, neighbor sampling is a crucial step in the GraphSAGE algorithm, providing an effective and scalable approach for node representation learning.

2. Materials and Methods

2.1 GraphSAGE

GraphSAGE is a node representation learning algorithm based on graph convolutional neural network [7]. Its core idea is to calculate the representation vector of each node by aggregating its neighbors, so as to transform the graph structure into vector representation in continuous vector space, which is convenient for node classification, link prediction and the other tasks, on the side of implementation, GraphSAGE uses the neighbor sampling technology which specifies the fixed size of each node's neighbor set to explore its local neighborhood information.

2.2 Attention Mechanisms

In GraphSAGE (Graph Sample and Aggregated), attention mechanism can be used to enhance the aggregation process of node representations. GraphSAGE is a method for learning node representations in a graph by extracting information from neighboring nodes to update each node's feature representation. Here are the general steps to incorporate attention mechanism into GraphSAGE model:

1) Define node representation update function: The core of the GraphSAGE model is to define a function that utilizes the neighboring nodes of a node to update its representation. Common update functions include mean pooling, max pooling, and adaptive pooling.

2) Introduce attention weights: To prioritize more important neighboring nodes during the representation update, attention weights can be introduced. These weights will be computed based on the relevance between the node and its neighbors.

3) Compute attention weights: To calculate attention weights, the following steps can be taken:

- Use parameterized linear mapping to transform node features into an initial representation of attention weights.
- Apply a non-linear activation function (e.g., ReLU) to transform the attention weights.
- Compute attention scores between each neighbor node and the target node, typically using either dot-product attention or additive attention.
- Normalize the attention scores using the SoftMax function to ensure that the weights sum up to 1.

4) Update node representations: Based on the computed attention weights, aggregate the features of neighboring nodes by weighting them accordingly. This allows a stronger focus on the important information from the neighboring nodes and updates the representation of the target node.

5) Iterate node representation update: Repeat the above steps until the node representations converge or reach a predetermined number of iterations.

2.3 Model Design and Implementation

This GraphSAGE model consists of the following parts:

- Input layer: accepts feature vectors of nodes in fraud detection tasks.
- GraphSAGE model [6]: based on input feature vectors and neighbor sampling

strategy, samples and aggregates features of neighbors for each node, resulting in a representation vector for each node.

- MLP classifier [4]: takes the representation vector of each node as input, transforms it through multiple layers of neural networks (MLP), and outputs the fraud probability prediction result for each node.

The implementation steps of the model are as follows:

- Normalize the input node features to avoid the impact of features at different scales.
- As shown in Figure 1, this modal uses the NeighborSampler [5] tool to sample neighbors of graph data to explore the information of neighboring nodes around fraud nodes.
- Aggregate the representation vectors of each node and its neighboring nodes according to the description in the GraphSAGE model to obtain the representation vector of this node.
- Input the representation vector of each node to the MLP classifier, and output the corresponding fraud probability prediction result after undergoing transformations through multiple fully connected layers.

Calculate the loss function of the model, update the model parameters using the back-propagation algorithm, and iteratively optimize it on the training set and test set.

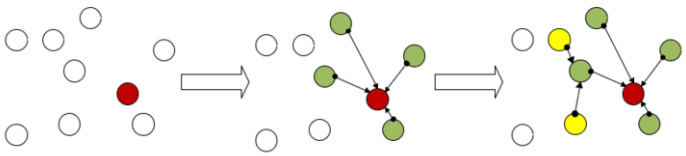

Figure 1. Neighbor sampling

3. Results and Discussion

3.1 Dataset Introduction

The dataset used is the DGraph-Fin benchmark dataset, which contains 3,700,550 nodes and 4,300,999 dynamic edges. The dataset divides users into four categories. The first category is normal users (non-fraud), accounting for 37.2% of users. The second category is fraud users, accounting for 0.42% of users. The others are non-relevant users and non-fraud users. The labels provided in DGraph-Fin describe the intrinsic properties of the nodes, which do not change over time. The Class-0 and Class-1 nodes of the last timestamp are randomly split into a training/validation/test set of 70/15/15.

3.2 Model Training

According to the random split of 70% as the training set and 15% as the test set from the dataset, the accuracy linear change graph obtained by the model after 500 iterations of training is shown in Figure 2. It can be seen from the results that after 200 iterations, the accuracy of the training set is 0.776, and the accuracy of the test set is 0.767. After 500 iterations, the accuracy of the training set is 0.8, and the accuracy of the test set is 0.786.

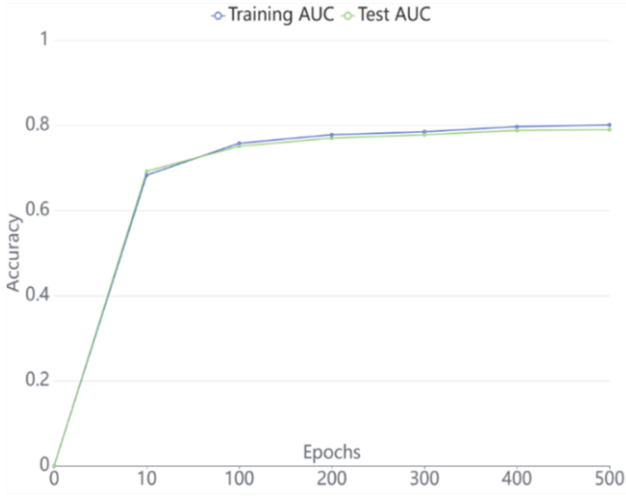

Figure 2. Model training results

As shown in Table 1. When the graph neural network model based on neighbor sampling is trained the same number of times, the AUC of both the training and test sets increases to 0.80, which shows a certain improvement in performance compared to the official highest baseline. The histogram for comparison is shown in Figure 3.

TABLE 1. MODEL TRAINING RESULTS COMPARISON

Methods	Training AUC	Test AUC
MLP	0.7221 ± 0.0014	0.7135 ± 0.0010
GCN	0.7108 ± 0.0027	0.7078 ± 0.0027
GraphSAGE	0.7682 ± 0.0014	0.7621 ± 0.0017
GAT (NeighborSampler)	0.7396 ± 0.0018	0.7233 ± 0.0012
GATv2(NeighborSampler)	0.7698 ± 0.0083	0.7526 ± 0.0089

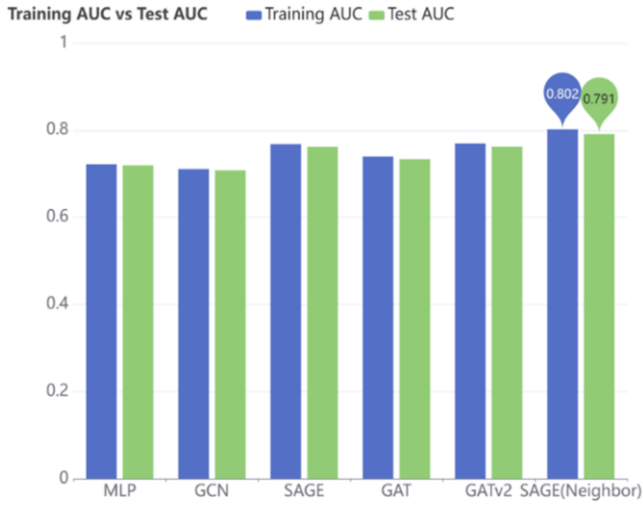

Figure 3. Accuracy comparison chart

4. Conclusions

By combining neighbor sampling and attention mechanism with graph neural network, the information relationship between nodes can be effectively utilized to perform user credit fraud detection using the graph structure. According to the results obtained from comparison experiments and compared with the dataset baseline, the accuracy has been improved to a certain extent.

References

[1] C. J. Mantas, J. G. Castellano, S. Moral-García, and J. Abellán, "A comparison of random forest based algorithms: random credal random forest versus oblique random forest," Soft Computing, Nov 2018.
[2] D. Zhang, L. Qian, B. Mao, C. Huang, B. Huang, and Y. Si, "A Data-Driven Design for Fault Detection of Wind Turbines Using Random Forests and XGboost," IEEE Access, vol. 6, pp. 21020–21031, 2018.
[3] W. Fan et al., "Graph Neural Networks for Social Recommendation," The World Wide Web Conference on - WWW '19, 2019.
[4] T. Windeatt, "Accuracy/Diversity and Ensemble MLP Classifier Design," IEEE Transactions on Neural Networks, vol. 17, no. 5, pp. 1194–1211, Sep 2006.
[5] Y. A. Malkov and D. A. Yashunin, "Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 4, pp. 824–836, Apr 2020.
[6] J. Liu, G. P. Ong, and X. Chen, "GraphSAGE-Based Traffic Speed Forecasting for Segment Network With Sparse Data," IEEE Transactions on Intelligent Transportation Systems, pp. 1–12, 2020.
[7] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph Convolutional Neural Networks for Web-Scale Recommender Systems," Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD '18, 2018.
[8] Li, H., Chen, Z., Li, Z., Zheng, Q., Zhang, P., & Zhou, S. (2023). GIPA: A General Information Propagation Algorithm for Graph Learning. arXiv. https://doi.org/10.48550/arXiv.2301.08209
[9] Zhang, L., Zhou, Z., Ji, P., & Mei, A. (2022). Application of Attention Mechanism with Prior Information in Natural Language Processing. International Journal on Artificial Intelligence Tools, 31(4). https://doi.org/10.1142/S0218213022400085