Fraud Detection in Reviews using Graph Neural Networks

Nguyen Le Vy[1,2], Nguyen Thi Mai Trinh[1,2] and Do Trong Hop[1,2]

[1] University of Information technology National University
[2] Han Thuyen Street, Thu Duc, Ho Chi Minh, Viet Nam {21522811, 21522718}@gm.uit.edu.vn

Abstract. Online shopping has become a widespread habit in modern society, especially with the development and optimization of technology. However, it comes with inherent risks. Online reviews play a crucial role in influencing consumers and boosting brands. However, the increasing prevalence of fraudulent reviews poses significant challenges to maintaining the integrity and fairness of these platforms. This paper presents an in-depth study on fraud detection in reviews, utilizing the YelpChi dataset. We implement and compare two advanced graph-based methods, GraphSAGE and CARE-GNN, for detecting fraudulent activities. Subsequently, we deploy model CARE-GNN and predict fraud detection in reviews. Our results demonstrate the effectiveness of both methods in identifying fraudulent reviews, providing detailed insights into their performance metrics with ROC-AUC greater than 0.50 and Recall score greater than 0.65 for both models. Notably, the GraphSAGE model achieves an accuracy of 0.85. This research aims to contribute to the development of more reliable review systems, enhancing user experience, and bolstering business credibility.

Keywords: GraphSAGE · CARE-GNN · fraud detection · Inference Distribute

1 INTRODUCTION

Online reviews are pivotal in digital commerce, shaping consumer decisions and business reputations. However, the purchase of reviews has led to widespread fraudulent reviews, compromising the reliability of these platforms. This manipulation includes fake positive reviews and negative reviews of competitors, misleading consumers and distorting market competition. This undermines consumer trust and harms legitimate businesses, affecting the platform's credibility. Furthermore, such review manipulation can be found across other industries, impacting society as a whole.

To address this issue, we employed GraphSAGE and CARE-GNN for detecting review fraud. These models leverage the interconnected nature of review data to detect irregularities and patterns indicative of fraudulent behavior. To

enhance the applicability of our research, we utilized dataset YelpChi[3] identified as suitable and structurally similar to the majority of real-world reviews.

We trained and evaluated GraphSAGE and CARE-GNN with 4 evaluation metrics (ROC-AUC, Recall score, F1 score and Aucuracy). We achieved promising results with both models. Subsequently, we selected model GraphSAGE and implemented distributed inference to develop fraud detection problem in reviews.

In this paper, we implement and compare two advanced graph-based methods, GraphSAGE and CARE-GNN.

- **GraphSAGE**: a graph neural network model designed for large-scale graph data, focusing on learning node representations by aggregating information from their local neighborhoods.
- **CARE-GNN (Camouflage-Resistant Graph Neural Network)**: an advanced graph neural network model specifically tailored for fraud detection in complex networks. It integrates breakthrough modules to mitigate the impact of camouflaged fraudsters, enhancing stability and accuracy in detecting fraudulent activities across online platforms.

Finally, we deployed a trained model for fraud review prediction into an inference distribution environment enhances the platform's ability to maintain integrity, protect users, and operate efficiently at scale.

2 RELATED WORK

Much of the previous work in opinion fraud focuses on review text content, behavioral analysis, and supervised methods. (Jindal and Liu 2008)[4] identified opinion spam bydetecting exact text duplicates in an Amazon.com dataset,while (Ott et al. 2011)[5] crowd-sourced deceptive reviews in order to create a highly accurate classifier based on n-grams. Several studies tried to engineer better features to improve classifier performance. (Li et al. 2011) uses sentiment scores, product brand, and reviewer profile attributes to train classifiers. Other work has computed scores based on behavioral heuristics, such as rating deviation by (Lim et al. 2010), and frequent itemset mining to find fraudulent reviewer groups by (Mukherjee, Liu, and Glance 2012).Unfortunately, these methods are not generalizable: the models need retraining to account for differences between problem domains, such as book reviews versus movie reviews. Moreover, the features might not be consistent even for datasets within the same domain, depending on the dataset source. Consequently, feature extraction becomes a time-consuming yet pivotal sub-problem with attributesvarying across domains.Another group of work mines behavioral patterns in review data. (Jindal, Liu, and Lim 2010) finds unexpectedrules to highlight anomalies, and (Xie et al. 2012; Feng etal. 2012; Feng, Banerjee, and Choi 2012) respectively study temporal review behavior,

[3] https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset
[4] DOI:10.1145/1341531.1341560
[5] https://aclanthology.org/P11-1032/

rating distributions, and syntactics-tylometry. On the other hand, methods that account for the network of reviewers, reviews, and products can more elegantly encapsulate structural signals that go beyond the review content and simple heuristics, thus generalizing across domains.(Wang et al. 2011) proposed the first (and to the authors'knowledge the only) review graph-based method and a simple yet effective algorithm to compute scores for each node.We propose an even more flexible method that exploits the network effects; being able to incorporate side information,is based on the rigorous theoretical foundation of belief propagation, and is linearly scalable. Network effects have been exploited in securities fraud (Neville et al. 2005), accounting fraud (McGlohon et al. 2009), and auction fraud (Pandit et al. 2007) detection. Howe-ver, none of these proposals are applicable to opinion fraud detection, as their problem domains do not involve ratings and sentiment spam. Related to ratings on products, work on recommender systems (Koren 2009;Menon and Elkan 2011), aim for best prediction of future user ratings, but do not address the fraud problem.

3 DATASET

3.1 Describe dataset

We use the Yelp review dataset [2], a comprehensive collection of Yelp reviews, to study the GNN-based fraud detection problem. The Yelp dataset, collected from Yelp.com, includes 67,395 reviews for a set of hotels and restaurants, with reviews from 201 hotels and restaurants by 38,063 reviewers. The reviews include product and user information, timestamps, ratings, and review texts. They are divided into two categories: filtered (spam) and recommended (legitimate). Users with more than 80% helpful votes are labeled as benign entities, while users with less than 20% helpful votes are labeled as fraudulent entities. Though previous works have proposed other fraud datasets and Bitcoin [4], they only contain graph structures and compacted features, with which we cannot build meaningful multi-relation graphs. In contrast, the YelpChi dataset has this capability and is widely used in fraud detection research, playing a significant role in the development of new methodologies in this field. In this paper, we conduct a spam review detection (fraudulent user detection) task on the Yelp dataset, which is a binary classification task. We use 32 handcrafted features as the raw node features for the Yelp dataset.

Table 1 shows the yelpChi dataset statistics.

We represent reviews as nodes in the graph and design three relations: 1) R-U-R: links reviews written by the same user; 2) R-S-R: connects reviews for the same product that have identical star ratings (ranging from 1 to 5 stars); 3) R-T-R: links two reviews for the same product that were posted within the same month.

3.2 Preprocessing dataset

With this dataset, we use a 4-step pipeline to clean and prepare the data for analysis:

#Nodes (Fraud%)	Relation	#Edges	Avg. Feature Similarity	Avg. Label Similarity
45,954 (14.5%)	R-U-R	49,315	0.83	0.90
	R-T-R	573,616	0.79	0.05
	R-S-R	3,402,743	0.77	0.05
	ALL	3,846,979	0.77	0.07

Table 1: YelpChi Dataset and graph statistics.

1. Normalizes a sparse matrix `'mx'`, ensuring uniformity across rows by adjusting their values relative to their sums.
2. Transforms a sparse adjacency matrix `'sp_matrix'` into a streamlined adjacency list format, meticulously saving it to a specified file (`filename`) for efficient data handling and exploration.
3. Categorizes nodes into positive and negative sets based on provided labels, thereby laying the foundation for targeted analysis of different node types.
4. Employs strategic undersampling of negative nodes `'neg_nodes'` to harmonize their representation with positive nodes `'pos_nodes'`, employing a defined scaling factor.

This pipeline enhances the dataset's integrity and reliability, ensuring that subsequent data analyses yield more robust and equitable insights into the underlying patterns and behaviors.

4 METHODOLOGY

4.1 Experimental procedures

After performing preprocessing and cleaning, we obtain a clean data set. Then, the data is divided into a training set (with a proportion of 80%) and a test set (with a proportion of 20%). The training data set is used to train the fraud detection in review with two advanced graph-based methods, GraphSAGE and CARE-GNN. All of the above models are applied on the test data set and use measures: ROC, Recall, F1-Score, Accuracy to compare and evaluate. Subsequently, we selected model CARE-GNN (higher result) and implemented distributed inference, aiming to develop fraud detection problems in reviews.

Experimental procedures is shown in the Figure 1.

4.2 Model

GraphSAGE The GraphSAGE model was researched and developed by W. Hamilton and colleagues in the paper "Inductive Representation Learning on

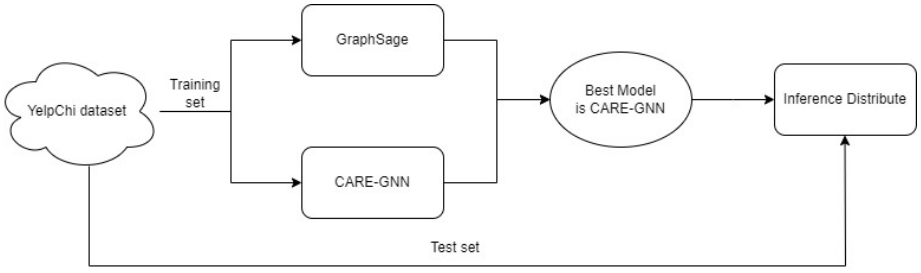

Fig. 1: A pipeline of problems

Large Graphs" (2017)[6], represents a significant advancement over the GCN (Graph Convolutional Network) introduced in 2016[7]. Key points include:

- **Inductive Learning**: GraphSAGE excels in generalizing to unseen data by leveraging node embeddings derived from neighboring nodes.
- **Aggregation Functions**: The paper details the design of aggregation functions aimed at synthesizing information from neighboring nodes, proposing three corresponding aggregation functions.
- **Mini-Batch Gradient Descent**: Unlike GCN's limitation with full-batch gradient descent, GraphSAGE employs mini-batch update gradient descent, enhancing scalability and efficiency.
- **Spatial GNN Method**: GraphSAGE builds on the idea of aggregating information from neighboring nodes, making it suitable for dynamic and large-scale datasets such as social networks, linked Wikipedia pages, new paper citations, user upvotes, video clips, or user follows.

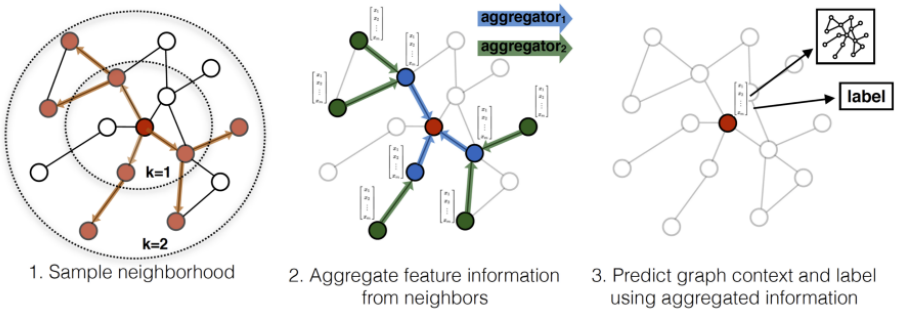

Fig. 2: Visual illustration of the GraphSAGE sample and aggregate approach.

[6] https://arxiv.org/pdf/1706.02216
[7] https://arxiv.org/abs/1609.02907

Overall, GraphSAGE offers enhanced applicability across a wide range of applications with large and evolving datasets, addressing the dynamic nature of modern data sources effectively.

CARE-GNN Y.Dou and colleagues propose a new model named CAmouflage-REsistant GNN (CARE-GNN) in "Enhancing Graph Neural Network-based Fraud Detectors against Camouflaged Fraudsters"(2017)[8], to enhance the GNN aggregation process with three unique modules against camouflages.

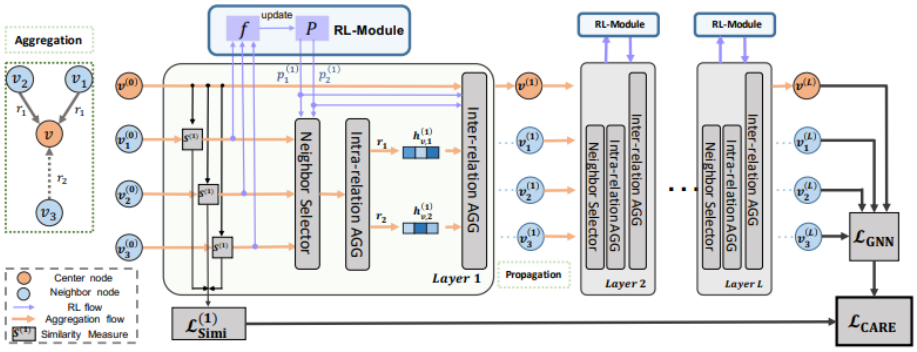

Fig. 3: The aggregation process of proposed CARE-GNN at the training phase.

The CARE-GNN model consists of three neural modules. Figure **figure2** illustrates the entire process of CARE-GNN. For a central node, the steps are as follows: 1) Compute the similarity of neighbors based on the proposed label-aware similarity measure; 2) Filter out dissimilar neighbors under each relation using the proposed neighbor selector. The neighbor selector is optimized using reinforcement learning during GNN training (purple module in Figure 2); 3) At the aggregation step, the authors first utilize the intra-relation aggregator to aggregate neighbor embeddings under each relation; 4) Combine embeddings across different relations using the inter-relation aggregator. The optimization steps and algorithm procedure are detailed further in the original article.

4.3 Distributed inference

Distributed inference is the process of performing inference or predictions with deep learning models across multiple computers or computational devices. This approach helps optimize performance and increase processing speed, especially when dealing with large datasets or requiring rapid response times. Here are some key points about distributed inference:

[8] `https://arxiv.org/abs/2008.08692`

- Increased Inference Speed: Distributed inference allows for the division of inference tasks across multiple computers or devices, reducing processing time and increasing inference speed. This is crucial in applications that require fast response times, such as facial recognition, natural language processing, and recommendation systems.
- Handling Large Datasets: When the volume of data is too large to be processed on a single machine, distributed inference enables the distribution of data and computation across multiple machines. This maximizes the use of computational resources and memory in a distributed system.
- Utilizing Computational Resources: Distributed inference leverages computational resources from multiple computers, including CPUs, GPUs, and TPUs. This helps optimize resource utilization and improve inference performance.
- Scalability: Distributed inference systems can be easily scaled by adding more computers or computational devices to the system. This helps meet the increasing processing demands without needing to change the model structure or source code.
- High Availability and Reliability: Distributed inference provides higher resilience as the workload is split and distributed across multiple machines. If one machine fails, others can continue processing, ensuring continuity and reliability of the system.

4.4 Evaluation metric

Due to the imbalanced nature of the Yelp dataset with a focus on fraudsters (positive cases), we use 4 evaluation metrics to assess classifier performance:

ROC-AUC (Receiver Operating Characteristic - Area Under the Curve) is a performance metric that evaluates the ability of a classifier to distinguish between classes. A higher ROC-AUC score (closer to 1) indicates better classifier performance in distinguishing between cases. AUC is calculated based on about the relative ranking of the predicted probabilities of all cases, can eliminate the influence of unbalanced layers.

Recall-Score measures the proportion of correctly predicted instances of a specific class relative to the total number of actual instances belonging to that class. This metric is calculated using the formula:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

F1-Score is a combined metric of Precision and Recall used to evaluate classifier performance. A higher F1-score indicates better performance. Ideally, F1 = 1 when Recall = Precision = 1. The formula for F1-Score is:

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Accuracy is a metric that measures the overall correctness of a classifier across all classes. It is calculated as the ratio of correctly predicted instances to the total number of instances in the dataset. The formula for accuracy is:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Number of Instances}}$$

5 EXPERIMENT

5.1 Experimental setting

Based on 3.1, it is evident that YelpChi is an imbalance dataset. Therefore, we apply under-sample and mini-batch training to improve the training efficiency and avoid overfitting.
We use unified node embedding size (emb_size = 64), batch size (batch_size = 1024), number of layers(1), learning rate (lr = 0.01), optimizer (Adam), and L2 regularization weight (0.001) for two models. For CARE-GNN and its variants, we set the RL action step size (0.02) and the similarity loss weight (2). In Section 4.5, we present the sensitivity study for the number of layers, embedding size, and $\lambda 1$.

5.2 Experimental Results

Table 2: Experimental results.

Model	**ROC-AUC**	**Recall Score**	**F1 score**	**Accuracy**
GraphSAGE	0.50	0.50	0.46	0.85
CARE-GNN	0.76	0.70	0.59	0.69

Table 2 presents our experimental results, highlighting the effectiveness of both GraphSAGE and CARE-GNN in detecting fraudulent reviews within the YelpChi dataset. For GraphSAGE, which leverages graph neural networks, demonstrated high accuracy in identifying individual fraudulent reviews by capturing subtle patterns and anomalies within the review network. Conversely, CARE-GNN excelled in identifying coordinated groups of fraudulent reviewers through its community detection approach. This underscores CARE-GNN's capability to uncover complex fraudulent behaviors involving multiple users working in tandem. Figure 4 is an example of inference distribution in English

```
+--------+----------+
|idx_node|prediction|
+--------+----------+
|   35042|         1|
|   27485|         0|
|   16164|         1|
|   21204|         0|
|   12201|         1|
+--------+----------+
only showing top 5 rows
```

Fig. 4: An example result in inference distribute

5.3 Advantages and disadvantages

Table 3 presents the advantages and disadvantages of the two models GraphSAGE and CARE-GNN, which we draw from experiments and research of reference documents.

Table 3: Comparison of GraphSAGE and CARE-GNN

Model	Pros	Cons
GraphSAGE	– - Efficient for large-scale graph data. – - Learns node representations by aggregating information from local neighborhoods. – - Flexible framework supporting various aggregation methods.	– - May struggle with highly camouflaged nodes. – - Aggregation may overlook subtle interactions between nodes.
CARE-GNN	– - Specifically designed for fraud detection. – - Equipped with modules to handle camouflaged fraudsters. – - Enhances stability and accuracy in detecting fraudulent activities.	– - Potentially higher computational complexity. – - May require more fine-tuning and domain-specific adjustments.

6 CONCLUSION

In conclusion, this study provides valuable insights into the application of graph-based methods for fraud detection in reviews. Both GraphSAGE and CARE-GNN show promise in enhancing the reliability of online review platforms.

The findings from this study have significant implications for the field of fraud detection in online reviews. By employing advanced graph-based methods like GraphSAGE and CARE-GNN, platforms can enhance their ability to detect and mitigate fraudulent activities, thereby improving the reliability and trustworthiness of review systems. This can lead to more informed consumer decisions and protect the reputations of businesses. Additionally, the successful application of these methods demonstrates the potential of graph-based approaches in other areas of fraud detection, such as financial fraud and social network spam.

Despite the promising results, this study has several limitations. First, the YelpChi dataset, while comprehensive, may not fully represent the diversity of review behaviors across different platforms and industries. Second, the computational complexity of graph-based methods can be a limiting factor, especially for very large datasets or real-time applications. Additionally, the performance of these methods depends on the quality and completeness of the network data, which may not always be available. Finally, while GraphSAGE and CARE-GNN are effective, they may require significant tuning and expertise to implement correctly, which could limit their accessibility for some users.

Future research will focus on refining these models, promoting ensemble models and exploring their applicability, across different datasets and domains, aiming to enhance the integrity of online review systems further. First, expanding the dataset to include more diverse review platforms and industries could provide a more comprehensive evaluation of the methods. Second, optimizing the computational efficiency of GraphSAGE and CARE-GNN can make them more suitable for real-time applications. Additionally, integrating these graph-based methods with traditional machine learning techniques could further improve detection accuracy by combining the strengths of both approaches.

References

1. Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng and P. Yu. 2020. *Enhancing Graph Neural Network-based Fraud Detectors against Camouflaged Fraudsters*. In CIKM '20. Path: https://arxiv.org/pdf/2008.08692
2. YELP, INC. AND 2 COLLABORATORS. 2022. *Yelp Dataset*. Path: https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset
3. S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V.S. Subrahmanian. 2018. *Rev2: Fraudulent user prediction in rating platforms*. In WSDM. DOI: https://doi.org/10.1145/3159652.3159729
4. M. Weber, G. Domeniconi, J. Chen, D. K. I. Weidele, C. Bellei, T. Robinson, and C. E. Leiserson. 2019. *Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics*. KDD Workshops (2019).

5. L. Faik *Graph Neural Networks: Link Prediction (Part II)*. 2022. Path: https://blog.dataiku.com/graph-neural-networks-link-prediction-part-two
6. Akoglu, L., Chandy, R., & Faloutsos, C. (2021). *Opinion Fraud Detection in Online Reviews by Network Effects. Proceedings of the International AAAI Conference on Web and Social Media*, 7(1), 2-11. https://doi.org/10.1609/icwsm.v7i1.14380
7. *YelpCHI dataset*. Path: https://odds.cs.stonybrook.edu/yelpchi-dataset/
8. Thomas N. Kipf, Max Welling *Semi-Supervised Classification with Graph Convolutional Networks*. 2017. In ICLR 2017. DOI: https://arxiv.org/abs/1609.02907