CAT-GNN: Enhancing Credit Card Fraud Detection via Causal Temporal Graph Neural Networks

Anonymous Author(s)

Abstract

Credit card fraud poses a significant threat to the economy. While Graph Neural Network (GNN)-based fraud detection methods perform well, they often overlook the causal effect of a node's local structure on predictions. This paper introduces a novel method for credit card fraud detection, the **Ca**usal **T**emporal **G**raph **N**eural **N**etwork (CAT-GNN), which leverages causal invariant learning to reveal inherent correlations within transaction data. By decomposing the problem into discovery and intervention phases, CAT-GNN identifies causal nodes within the transaction graph and applies a causal mixup strategy to enhance the model's robustness and interpretability. CAT-GNN consists of two key components: Causal-Inspector and Causal-Intervener. The Causal-Inspector utilizes attention weights in the temporal attention mechanism to identify causal and environment nodes without introducing additional parameters. Subsequently, the Causal-Intervener performs a causal mixup enhancement on environment nodes based on the set of nodes. Evaluated on three datasets, including a private financial dataset and two public datasets, CAT-GNN demonstrates superior performance over existing state-of-the-art methods. Our code can be found in the supplementary material.

Introduction

The substantial damages wrought by financial fraud continue to garner ongoing focus from academic circles, the business sector, and regulatory bodies (Jiang, Cui, and Faloutsos 2016; Aleksiejuk and Hołyst 2001). Fraudsters masquerade as ordinary users and attack transactions made with credit cards (Ileberi, Sun, and Wang 2022), which may inflict substantial economic losses and pose a severe threat to sustainable economic growth (AlFalahi and Nobanee 2019). Consequently, effective detection of financial fraud is imperative for safeguarding the economy and consumer security.

In the financial deception realm, identifying credit card fraud has garnered considerable attention among both industry and academia (Bhattacharyya et al. 2011). Traditional approaches to detecting fraudulent activities typically entail meticulous examination of each transaction for irregularities, employing predefined criteria such as verification against lists of compromised cards or adherence to established transaction thresholds (Maes et al. 2002; Fu et al. 2016). However, the aforementioned anti-fraud systems, based on expert prior and rules, are often susceptible to exploitation by fraudsters, who can circumvent detection by crafting ingenious transaction methods that elude the system's scrutiny of illicit activities. Toward this end, predictive modeling has been introduced, aiming to autonomously identify patterns that suggest fraudulent activity and calculate a corresponding risk score.

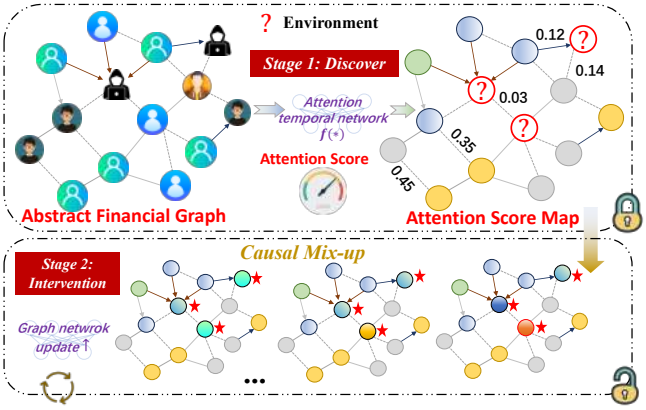


Figure 1: The model overview. First Stage (discovery): we utilize an attention map in the attention temporal network to identify causal nodes and environment nodes. Second Stage: Intervention, we apply causal mix-up enhancement to the environment nodes.

Currently, state-of-the-art predictive models are focused on using deep learning methods, capturing potential illegal patterns in a data-driven manner (Fu et al. 2016; Dou et al. 2020). For instance, (Liu et al. 2021) introduces PC-GNN, a Graph Neural Network approach that effectively handles class imbalance in graph-based fraud detection by selectively sampling nodes and edges, particularly focusing on the minority class. Moreover, (Xiang et al. 2023) leverages transaction records to construct a temporal transaction graph, applying a Gated Temporal Attention Network to effectively learn transaction representations and model fraud patterns. Unfortunately, **i)** these methods often overlook the intrinsic patterns and connections within the data due to a lack of consideration for local structure consistency; **ii)** they lack the ability to uncover the causal nature of each specific case, which leads to inadequate differentiation between the attributes of causal nodes and environment nodes, thereby impairing the model's generalization capabilities; **iii)** they lack interpretability in making specific predictions.

In this paper, we introduce a novel **Ca**usal **T**emporal **G**raph **N**eural **N**etwork, termed CAT-GNN, aiming at providing an interpretable paradigm for credit card fraud detection. Guided by the currently popular causal invariant learning techniques (Chang et al. 2020; Liu et al. 2022), CAT-GNN's primary objective is to *unveil the inherent correlations in the transaction attribute data of nodes within available temporal transaction graphs*, thereby offering interpretability for complex transaction fraud problems.

To unravel causal correlations, specifically, we decompose the algorithmic process of CAT-GNN into two stages - **discovery** and **intervention**. The goal of the discovery stage is to identify potential causal components within observed temporal graph data, where we introduce a causal temporal graph neural network for modeling. Utilizing the popular node-attention metrics (Veličković et al. 2017; Xiang et al. 2023), we employ attention score to locate key nodes, designated as causal and environment nodes. In the intervention process, we aim to reasonably enhance potential environment nodes. This approach is designed to align with and perceive the underlying distribution characteristics in explicit fraud networks, thereby boosting our temporal GNN's ability to identify and understand problematic nodes. Furthermore, drawing inspiration from (Wang et al. 2020), to ensure that causal interventions between nodes do not interfere with each other, we create parallel universes for each node. Consequently, the model is exposed to a wider potential data distribution, providing insights for fraud prediction with a causal perspective. This process can further be understood as a back-door adjustment in causal theory (Pearl 2009; Pearl and Mackenzie 2018). The contributions of this paper are summarized as follows:

- We propose a novel method, CAT-GNN, that embodies both causality and resilience for the modeling of credit card fraud detection. By harnessing causal theory, known for its interpretability, CAT-GNN enables the model to encompass a wider potential data distribution, thereby ensuring its exceptional performance in this task.
- CAT-GNN, characterized by its refined simplicity, initially identifies causal nodes and subsequently refines the model into a causally coherent structure. It aims to achieve invariance in attribute information and temporal features through semi-supervised learning, thereby providing a bespoke and robust foundation for targeted tasks.
- We evaluate CAT-GNN on three representative datasets, including a **private** financial benchmark, and the other two are public settings. Extensive experiments show that our proposed method outperforms the compared state-of-the-art baselines in credit card fraud detection, thanks to the casual intervention of the node causal augment.

Preliminaries

Definition 1. (Multi-Relation Graph) The Multi-Relation Financial Graph $\mathcal{G}$ is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X}, \mathcal{Y})$, where $\mathcal{V} = \{v_1, v_2, \cdots, v_N\}$ represents the set of nodes, with $N = |\mathcal{V}|$ indiCates the total number of nodes. $\mathcal{X} \in \mathbb{R}^{N \times d}$ denotes the node features with $x_i \in \mathbb{R}^d$ as its entry for node v_i, d is the feature dimension. Each node v_i is assigned a label $y_i \in \mathcal{Y}$, which is a binary variable with the value in $\{0, 1\}$. $\mathcal{E} = \{\mathcal{E}_1, \cdots, \mathcal{E}_R\}$ signifies the set of edges, partitioned into R distinct types of relations.

Definition 2. (Graph-based Fraud Detection) The graph-based fraud detection problem is defined on the multi-relation graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X}, \mathcal{Y})$. For such a problem, each node v_i represents the target entity such as a transaction record, and has a label $y_i \in Y$, where $y_i = 0$ represents benign and $y_i = 1$ represents fraud. The objective of graph-based fraud detection is to identify fraud nodes that stand out distinctly from the non-fraudulent, or benign, nodes within a multi-relational graph $\mathcal{G}$. This task is effectively approached as a binary classification problem focused on nodes within the graph $\mathcal{G}$. And the surprised Binary Cross-Entropy Loss function is: $\min_\Theta \mathcal{L} = -\frac{1}{B}\sum_{u=1}^{B}\left(y_i^\top \log(\hat{y}_i) + (1-y_i)^\top \log(1-\hat{y}_i)\right) + \eta||\Theta||^2$, where B is the batch size, $\hat{y}_i \in [0, 1]$ is the predicted probability, and $y_i \in \{0, 1\}$ is the ground truth. Θ is the parameter of the GNN predictor, and η is the weight-decay coefficient.

Methodology

In Section , we explore the motivation behind our approach, emphasizing the crucial role of understanding the local structure and causal relationships within transaction data to improve detection accuracy. Section introduces our two-phase method: discovery and intervention. Section Section provides the causal theory support.

Motivation

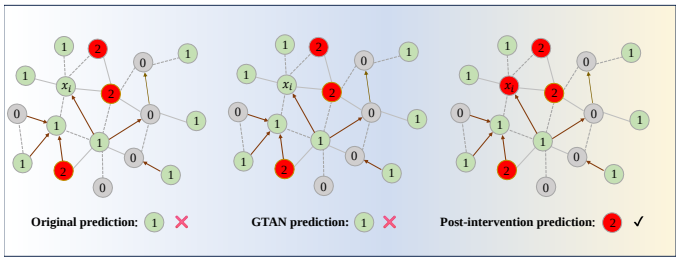


Figure 2: ***Motivation.*** The original prediction incorrectly identifies a fraudster (central node labeled x_i) as benign, as does the state-of-the-art GTAN model. Following our causal intervention, the prediction is correctly adjusted to identify x_i as a fraudster. *Green: benign users, red: fraudsters, gray: unlabeled nodes.*

Taking the arXiv (Hu et al. 2020) dataset as an example, real-world graphs often exhibit locally variable structures, that is, the distribution of node attributes differs from the distribution of local structural properties (Feng et al. 2021). We observe that this phenomenon is also prevalent in the financial sector, where cunning fraudsters may disguise themselves through various means (such as feature camouflage and relationship disguise) to connect with users who have a good credit transaction history (Dou et al. 2020). *In such scenarios, if we simply aggregate node information and neighbor information together, it is likely to obscure the auspiciousness of the fraudsters, which contradicts our objective.* This situation tends to yield poorer training outcomes, especially in a semi-supervised learning environment with limited labeled data. Existing methods do not incorporate causal factors into credit card fraud modeling, resulting in models that fail to learn the intrinsic connections of node attributes. This oversight further leads to the neglect of causal

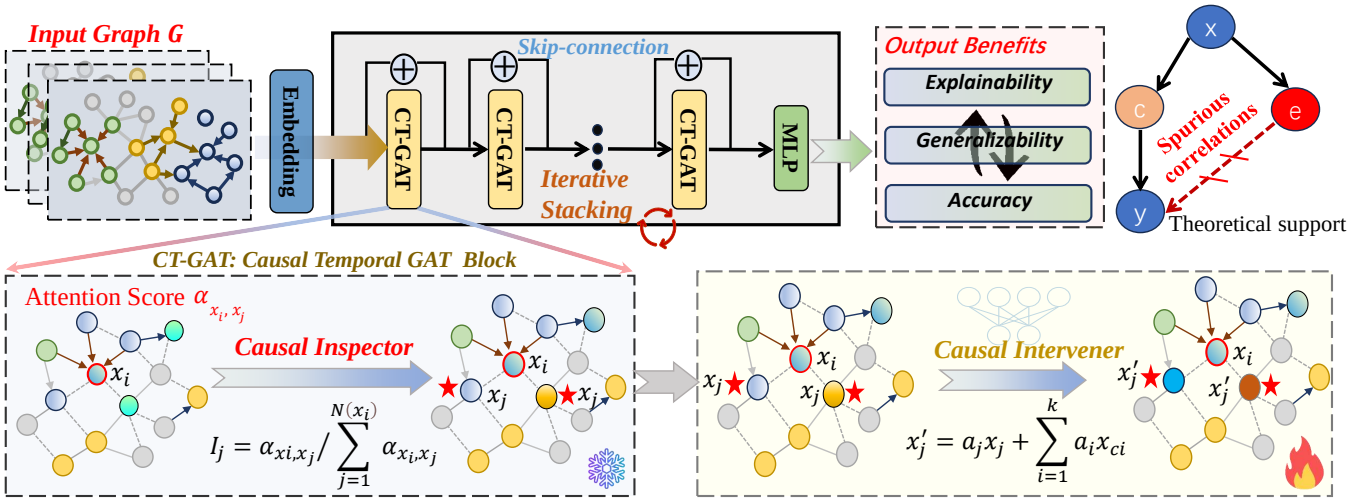


Figure 3: The depiction of the proposed model's architecture, featuring a causal temporal graph attention mechanism, alongside the theoretical support for backdoor adjustment.

attribute structure differences on test nodes, thereby reducing the model's generalizability. By comprehensively examining the confounding variables, we are able to significantly alleviate the aforementioned issue, as illustrated in Figure 2. This strategy is the cornerstone of our framework and is also known as the "backdoor adjustment" technique (Pearl 2009; Pearl and Mackenzie 2018).

Discovering & Intervention

Based on the motivation, we adopt a causal perspective to analyze the attribute aggregation process and formalize principles for distinguishing between causal and non-causal elements within local structures. We first introduce the discovery process to effectively examine the causal and environment nodes within the current node's local structure. In response, we refine the temporal attention graph network mechanism(Xiang et al. 2022) into *a causal temporal GAT mechanism* as shown in the upper half of Figure 3. This refinement introduces two key components designed to accurately identify both environmental and causal nodes, which enhances our ability to understand and manipulate the local structural dynamics more effectively.

In the context of temporal transaction graphs, we maintain a set of transaction records, denoted as $R = [r_{t1}, r_{t2}, \cdots, r_{ti}]$, alongside their embeddings $X = [x_{t1}, x_{t2}, \cdots, x_{ti}]$ obtained via a projection layer. As demonstrated in (Shi et al. 2020), GNNs are capable of concurrently propagating attributes and labels. Consequently, we integrate fraud labels as an embedded feature within the attribute embedding x_{ti}, employing masking techniques to prevent label leakage (Xiang et al. 2023). However, this aspect does not constitute the primary focus of our research.

Causal-Inspector: We design a Causal-Inspector to identify causal and environment nodes as shown in the bottom left corner of Figure 3. To aggregate information efficiently, we employ the aforementioned causal temporal graph attention mechanism, which allows for dynamic information flow based on the temporal relationships among transactions. Leveraging a multi-head attention mechanism, we compute temporal attention scores that serve as weights for each neighboring node, facilitating the assessment of each neighbor's causal importance, which can be formulated as follows:

$$\alpha^h_{x_i,x_j} = \frac{\exp\big(\text{LeakyReLU}(W_a^T[x_i \oplus x_j])\big)}{\sum_{j\in N(x_i)} \exp\big(\text{LeakyReLU}\big(W_a^T[x_i \oplus x_j]\big)\big)}, \tag{1}$$

where W_a is a learnable weight matrix, α_{x_i,x_j} represents the attention weight of node x_i with respect to node x_j in one head, which determines the importance of node x_i relative to node x_j. The $\oplus$ symbol represents the concatenation operation. $N(x_i)$ is the set of temporal neighboring nodes of node x_i. In order to quantify the importance of each node x_j we aggregate the attention weights $\alpha^h_{x_i,x_j}$ from each attention head and compute the average to determine the final weight of the node. Then, based on its final weight, we calculate its normalized importance:

$$\bar{\alpha}_{x_i,x_j} = \frac{1}{H}\sum_{h=1}^{H} \alpha^h_{x_i,x_j} \rightarrow \quad I_j = \bar{\alpha}_{x_i,x_j} / \sum_{j=1}^{N(x_i)} \bar{\alpha}_{x_i,x_j} \tag{2}$$

where H is the total number of attention heads and $I_i = [I_{j1}, \cdots, I_{jN(x_i)}]$ represents the set of importance of each node with respect to x_i. This formula calculates the normalized importance weight I_j, representing the importance of node x_j by compiling the contributions from all attention heads, thus providing a comprehensive measure of node significance. To segregate the nodes into environmental and causal Categories, we introduce a proportion parameter r_e, ranging between 0 and 1, which denotes the fraction of nodes to be earmarked as environment nodes. This approach affords us the flexibility to select environment nodes tailored to the specific exigencies of the graph. We use the argmin$(\cdot)$ function to select the $\lceil r_e N \rceil$ nodes with the lowest importance scores as environment nodes. Therefore, a ranking function R is defined to map I_j to its rank among all node importance scores. Then, we determine the environment set S_e as:

$$S_e = \{x_j | \underset{j=1,\cdots,\lfloor r_e N \rfloor}{\text{argmin}} R(I_j)\}. \tag{3}$$

The remaining nodes, those not in S_e, naturally form the set of causal nodes S_c.This method ensures that nodes with

the lowest importance scores are precisely selected as environmenta nodes according to the proportion r_e, while the rest serve as causal nodes. Due to the differences between test and training distributions (Feng et al. 2021), CAT-GNN is dedicated to perceiving the essence of temporal graph data, thereby enhancing generalization capabilities and robustness.

Causal-Intervener: We design a Causal Intervener as shown in the bottom right corner of Figure 3, which employs a transformative mixup strategy known as a causal mixup, that blends environment nodes with a series of causally significant nodes. Given an environmental node $x_j \in S_e$, We select the causal nodes $\{x_{c1}, x_{c2}, \cdots, x_{ck}\}$ with the highest importance scores, which are computed as outlined in the Causal-Inspector, from the causal set S_c at a proportion of r_c. The *causal mixup* is then executed by linearly combining the environmental node with the selected causal nodes, weighted by their respective coefficients $\{a_j, a_1, a_2, \cdots, a_k\}$, which are learned through a dedicated linear layer:

$$x'_j = a_j x_j + \sum_{i=1}^{k} a_i x_{ci}, \tag{4}$$

where x'_j is the causally mixed environmental node, k is the number of selected causal nodes, a_j is the self-weight of the environmental node reflecting its inherent causal significance, and a_i is the causal node weight. These weights are normalized such that $a_j + \sum_{i=1}^{k} a_i = 1$. The incorporation of the causal mixup enhances the robustness of the model against distributional shifts by embedding a richer causal structure within the environmental node. By adapting the causal structure to the environmental context, the Causal-Intervener aims to mitigate the disparity between training and test distributions, thus bolstering the model's generalizability. Finally, we aggregate the information, and the outputs of multiple attention heads are concatenated to form a more comprehensive representation:

$$\begin{aligned} \mathcal{H} &= \sum_{x_i \in \mathcal{X}} \sigma\Big(\sum_{x_j \in N(x_i)} \alpha_{x_i, x_j} x'_j \Big), \\ \mathcal{M} &= \big(\mathcal{H}_1 \oplus \cdots \oplus \mathcal{H}_{\mathrm{h}}\big) W_c, \end{aligned} \tag{5}$$

where W_c is a learnable weight matrix, $\mathcal{H}$ is a attention head, $\mathcal{M}$ denotes the aggregated embeddings. It is important to highlight that the causal intervention results x'_j on an environmental node x_j with respect to x_i is essentially a duplicate of x_j and does not modify x_j itself. This distinction is crucial as it guarantees that the process of augmenting central nodes within individual local structures remains mutually non-disruptive. By preserving the original state of x_j, we ensure that enhancements applied to central nodes in one local structure do not adversely affect or interfere with those in another, maintaining the integrity and independence of local structural enhancements (Wang et al. 2020).

Causal Support of CAT-GNN

In elucidating the causal backbone of CAT-GNN, we invoke causal theory to formulate a Structural Causal Model (SCM) as propounded by (Pearl 2009). This framework scrutinizes four distinct elements: the inputs node attribute X, the truth label Y decided by both the attribute of causal nodes of X symbolized as $\tilde{C}$, and the confounder E, emblematic of the attribute of environment nodes. The causal interplay among these variables can be articulated as follows:

- $\tilde{C} \leftarrow \mathrm{X} \rightarrow \mathrm{E}$. The local structure of node attribute X is composed of causal nodes attributes $\tilde{C}$ and environment nodes attributes E.
- $\tilde{C} \rightarrow \mathrm{Y} \leftarrow \mathrm{E}$. The causal attributes $\tilde{C}$ actually determine the true value Y, however, the environmental attributes E also affect the prediction results, causing spurious associations.

Do-calculus (Pearl 2009) is a trio of rules within the causal inference framework that facilitates the mathematical deduction of causal effects from observed data. These rules enable manipulation of do$(\cdot)$ operator expressions, essential for implementing interventions in causal models:

$$\begin{aligned} &P\big(Y|\mathrm{do}(\hat{C}), E\big) = P(Y|\hat{C}), \\ &P\big(Y|\mathrm{do}(\hat{C}), \mathrm{do}(E)\big) = P\big(Y|\mathrm{do}(\hat{C}), E\big), \\ &P\big(\hat{C}|\mathrm{do}(Y)\big) = P\big(\hat{C}|Y\big). \end{aligned} \tag{6}$$

Typically, a model M_θ that is trained using Empirical Risk Minimization (ERM) may not perform adequately when generalizing to test data distribution P_{test}. These shifts in distribution are often a result of changes within environment nodes, necessitating the need to tackle the confounding effects. As illustrated in Figure 3, we apply causal intervention to enhance the model's generalizability and robustness. To this end, our approach utilizes do-calculus (Pearl 2009) on the variable C to negate the influence of the backdoor path $E \rightarrow Y$ by estimating $P\big(Y|\mathrm{do}(\hat{C})\big) = P_m(Y|\hat{C})$:

$$\begin{aligned} P\big(Y|\mathrm{do}(\hat{C})\big) &= \sum_{i}^{N_e} P\big(Y|\mathrm{do}(\hat{C}), E = E_i\big) P\big(E = E_i|\mathrm{do}(\hat{C})\big) \\ &= \sum_{i}^{N_e} P\big(Y|\mathrm{do}(\hat{C}), E = E_i\big) P(E = E_i) \\ &= \sum_{i}^{N_e} P\big(Y|\hat{C}, E = E_i\big) P(E = E_i), \end{aligned}$$

where N_e signifies the count of environment nodes, with E_i indiCating the i-th environmental variable. The environmental enhancement of CAT-GNN is in alignment with the theory of backdoor adjustment, thereby allowing for an effective exploration of possible test environment distributions.

Experiments

In this section, we critically assess the CAT-GNN model on a series of research questions (RQs) to establish its efficacy in graph-based fraud detection tasks. The research questions are formulated as follows:

- **RQ1**: Does CAT-GNN outperform the current state-of-the-art models for graph-based anomaly detection?

Table 1: Statistics of the three datasets.

Dataset	#Node	#Edge	#Fraud	#benigh
YelpChi	45,954	7,739,912	6,677	39,277
Amazon	11,948	8,808,728	821	11,127
FFSD	1,820,840	31,619,440	33,858	141,861

- **RQ2:** What is the effectiveness of causal intervention in the aggregation of neighboring information?
- **RQ3:** What is the performance with respect to different environmental ratios r_e?
- **RQ4:** Is CAT-GNN equally effective in semi-supervised settings, and how does it perform with limited labeled data?
- **RQ5:** Does the causal intervention component lead to a significant decrease in efficiency?

Experimental Setup

Datasets. We adopt one private finacial fraud semi-supervised dataset (Xiang et al. 2023), termed FFSD, with the partially labeled transaction records. Same with the definition in section , if a transaction is reported by a cardholder or identified by financial experts as fraudulent, the label y_v will be 1; otherwise, y_v will be 0. In addition, we also validate on two other public fraud detection datasets **Yelpchi** and **Amazon**. **Yelpchi** (Rayana and Akoglu 2015) compiles a collection of hotel and restaurant reviews from Yelp, in which nodes represent reviews. And there are three kinds of relationship edges among these reviews. **Amazon**: The Amazon graph (McAuley and Leskovec 2013) comprises reviews of products in the musical instruments Category, in which nodes represent users, and the edges are the corresponding three kinds of relationships among reviews. The statistics of the above three datasets are shown in Table 1.

Baselines. To verify the effectiveness of our proposed CAT-GNN, we compare it with the following state-of-the-art methods. ❶ *Player2Vec*. Attributed Heterogeneous Information Network Embedding Framework (Zhang et al. 2019). ❷ *Semi-GNN*. A semi-supervised graph attentive network for financial fraud detection that adopts the attention mechanism to aggregate node embed dings across graphs (Wang et al. 2019). ❸ *GraphConsis*. The GNN-based fraud detectors aim at the inconsistency problem (Liu et al. 2020). ❹ *GraphSAGE*. The inductive graph learning model is based on a fixed sample number of the neighbor nodes (Hamilton, Ying, and Leskovec 2017). ❺ *CARE-GNN* The camouflage-resistant GNN-based model tackling fraud detection (Dou et al. 2020). ❻ *PC-GNN*. A GNN-based model to address the issue of class imbalance in graph-based fraud detection (Liu et al. 2021). ❼ *GTAN*. A semi-supervised GNN-based model that utilizes a gated temporal attention mechanism to analyze credit card transaction data (Xiang et al. 2023). ❽: CAT-GNN (PL). This variant of the Cat-GNN framework selects environment nodes based on a proportion r_e and determines mixup weights a_i via a learnable linear layer. ❾: CAT-GNN (PI). This version employs a proportional selection of environment nodes and leverages the nodes' importance scores to inform mixup weights $a_i = I_i / \sum_{i=1}^{k} I_i$. ❿: CAT-GNN (FL). This variant uses a fixed number of environment nodes. Mixup weights are determined by a learnable linear layer. ⓫:CAT-GNN (FI). Combining fixed environmental node selection with importance-based weighting for mixup.

Reproducibility In our experiment, the learning rate l_r is set to 0.003, and the batch size batch N_{batch} is established at 256. Moreover, the input dropout ratio $r_{dropout}$ is determined to be 0.2, with the number of attention heads N_{head} set to 4, and the hidden dimension d to 256. We employed the Adam optimizer to train the model over $N_{epoch} = 100$ epochs, incorporating an early stopping mechanism to prevent overfitting. In GraphConsis, CARE-GNN, PC-GNN and GTAN, we used the default parameters suggested by the original paper. In Semi-GNN and Player2Vec, We set the learning rate to 0.01. In YelpChi and Amazon datasets, the train, validation, and test ratio are set to be 40%, 20%, and 40% respectively. In the FFSD dataset, we use the first 7 months' transactions as training data, and the rest as test data. Similar to previous work (Liu et al. 2021), we repeat experiments with different random seeds 5 times and we report the average and standard error. Experimental results are statistically significant with $p < 0.05$. CAT-GNN and other baselines are all implemented in Pytorch 1.9.0 with Python 3.8. All the experiments are conducted on Ubuntu 18.04.5 LTS server with 1 NVIDIA Tesla V100 GPU, 440 GB RAM.

Metrics. We selected three representative and extensively utilized metrics: **AUC** (Area Under the ROC Curve), **F1-macro** and **AP** (averaged precision). The first metric AUC is the area under the ROC Curve and as a single numerical value, AUC succinctly summarizes the classifier's overall performance across all thresholds. The second metric F1-macro is the macro average of F1 score which can be formulated as $F1_{macro} = 1/(l \sum_{i=1}^{l} \frac{2 \times P_i \times R_i}{P_i + R_i})$, and the third metric AP is averaged precision that can be formulated as $AP = \sum_{i=1}^{l} (R_i - R_{i-1}) P_i$, where P_i stands for the Precision and R_i stands for recall.

Performance Comparison (RQ1)

In the experiment of credit card fraud detection across three distinct datasets, CAT-GNN showcases superior performance metrics compared to its counterparts. First of all, CAT-GNN achieves the highest AUC in all three datasets, with values of 0.9035, 0.9706, and 0.8281 for YelpChi, Amazon, and FFSD, respectively. This indicates that **CAT-GNN consistently outperforms other methods in distinguishing between classes across diverse datasets**. Focusing on the F1 Score, which balances the precision P_i and recall R_i, CAT-GNN again tops the charts with scores of 0.7783, 0.9163, and 0.7211 for YelpChi, Amazon, and FFSD. This reflects the model's robustness in achieving high precision while not compromising on recall, which is essential where both false positives and false negatives carry significant consequences. Finally, CAT-GNN's superiority ex-

Table 2: Performance Comparison (in percent ± standard deviation) on YelpChi, Amazon and FFSD datasets across five runs. The best performances are marked with **bold font**, and the second-to-best are shown underlined.

Dataset	YelpChi			Amazon			FFSD		
Metric	AUC	F1	AP	AUC	F1	AP	AUC	F1	AP
Player2Vec	.7012±.0089	.4120±.0142	.2477±.0161	.6187±.0152	.2455±.0091	.1301±.0117	.5284±.0101	.2149±.0136	.2067±.0155
Semi-GNN	.5160±.0154	.1023±.0216	.1809±.0205	.7059±.0211	.5486±.0105	.2248±.0142	.5460±.0125	.4393±.0152	.2732±.0207
GraphSAGE	.5414±.0029	.4516±.0954	.1806±.0866	.7590±.0053	.5926±.0087	.6597±.0079	.6534±.0095	.5396±.0101	.3881±.0089
GraphConsis	.7046±.0287	.6023±.0195	.3269±.0186	.8761±.0317	.7725±.0319	.7296±.0301	.6554±.0412	.5436±.0376	.3816±.0341
CARE-GNN	.7745±.0281	.6252±.0091	.4238±.0151	.8998±.0925	.8468±.0085	.8117±.0114	.6589±.1078	.5725±.0096	.4004±.0090
PC-GNN	.7997±.0021	.6429±.0205	.4782±.0194	.9472±.0019	.8798±.0084	.8442±.0096	.6707±.0031	.6051±.0230	.4479±.0210
GTAN	.8675±.0036	.7254±.0197	.6425±.0154	.9580±.0014	.8954±.0095	.8718±.0083	.7496±.0041	.6714±.0089	.5709±.0097
CAT-GNN (FI)	.8721±.0044	.7336±.0295	.6528±.0209	.9643±.0026	.9011±.0129	.8794±.0102	.7643±.0078	.6907±.0198	.5925±.0174
CAT-GNN (FL)	.8910±.0026	.7692±.0182	.6687±.0135	.9705±.0016	.9125±.0099	.8942±.0081	.8023±.0067	.7031±.0154	.6145±.0169
CAT-GNN (PI)	.8895±.0041	.7706±.0223	.6701±.0181	.9669±.0021	.9077±.0113	.8896±.0095	.8145±.0061	.7096±.0149	.6294±.0166
CAT-GNN (PL)	**.9035±.0035**	**.7783±.0209**	**.6863±.0127**	**.9706±.0015**	**.9163±.0104**	**.8975±.0089**	**.8281±.0054**	**.7211±.0115**	**.6457±.0156**

tends to the AP metric, with the improvement of at least 6.82%, 2.86%, and 13.10% for YelpChi, Amazon and FFSD respectively.

The comparative performance of CAT-GNN is particularly significant when contrasted with previous methods such as Player2Vec, Semi-GNN, and GraphSAGE. For the Amazon dataset, existing state-of-the-art models, like CARE-GNN, PC-GNN, and GTAN, have already proven effective at capturing the inherent correlations within the data. In this context, the benefits of causal intervention may not be as pronounced, possibly due to the dataset's simpler local structures and more uniform distribution. However, for the FFSD dataset, our methodology exhibits significant performance improvements. This enhancement is attributed to the complex local structures and the prevalence of unlabeled nodes within the dataset. In such scenarios, causal intervention adeptly learns the inherent attribute connections, thereby boosting the model's generalization. Additionally, learning mixup weights with a linear layer is more reasonable than weighting with importance scores. Similarly, selecting environment nodes based on proportions is more sensible than choosing a fixed number of environment nodes, and the effect is also slightly better. All in all, This superior performance can be *ascribed to the integration of causal theory within the* CAT-GNN, enhancing its capacity to comprehend the inherent principles of graph attributes, allowing it to discern complex patterns and interactions that other models are unable to effectively capture.

Ablation Study (RQ2)

In this section, we evaluate the effectiveness of causal interventions in the aggregation within graph structures. Initially, we explore a variant without any causal intervention, termed N-Cat, which aggregates all neighboring information indiscriminately. Secondly, we introduce D-Cat, a method that omits environment nodes entirely during the aggregation phase, and directly aggregates all neighboring information in the learning process. Finally, our proposed method, Cat, integrates a causal intervention approach, simultaneously considering both causal nodes and environment nodes during aggregation to refine the learning representations.

The results shown in Figure 4 highlight the importance of causal intervention in information aggregation. N-Cat, which lacks causal discernment, performs worse across all datasets compared to Cat because it does not account for causal relationships. D-Cat, which simply deletes environmental factors, shows a significant drop in performance, as the mere deletion of environment nodes prevents the model from fully learning valuable information. Our Cat method consistently outperforms the other variants across all datasets, achieving the highest AUC scores. This superior performance underscores the value of our causal intervention technique, which effectively balances the influence of causal and environment nodes, resulting in a more generalizable model.

Parameter Sensitivity Analysis (RQ3, RQ4)

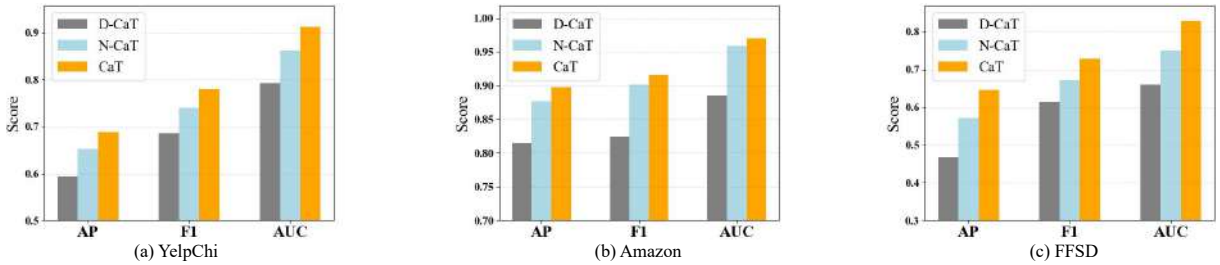
In this section, we study the model parameter sensitivity with respect to the environment nodes ratio and the training ratio. The corresponding results are in Figure 5.

As demonstrated in the left of Figure 5, using the YelpChi dataset as an example, the performance of CAT-GNN (measured by AUC as the performance metric) significantly surpasses other competitive models, including PC-GNN and CARE-GNN, across all training ratios, from 10% to 70%. Particularly at lower training ratios (such as 10%), CAT-GNN remains effective for semi-supervised learning and exhibits more robust performance compared to other models.

In our sensitivity analysis of the environmental ratio as demonstrated in the right of Figure 5, we observed that CAT-GNN's performance on the Amazon dataset is less affected by variations in the training ratio, with AUC fluctuations not exceeding 2%. Conversely, on the FFSD dataset, as the training ratio increases from 5% to 40%, there is a larger fluctuation in CAT-GNN's performance. This can be attributed to the characteristics of the dataset or the differences in the distribution of labeled data.

Model Efficiency (RQ5)

In this section, we present a comprehensive analysis of the efficiency of CAT-GNN. Our causal intervention aims to boost performance while maintaining computational efficiency. Table 3 shows that the performance enhancements are achieved without imposing significant additional computational costs. The results indicated that the execution time with causal intervention experienced only a marginal



(a) YelpChi (b) Amazon (c) FFSD

Figure 4: The ablation study results on three datasets. Gray bars represent the D-Cat variant, blue bars represent the N-Cat variant, and orange bars represent the CAT-GNN model.

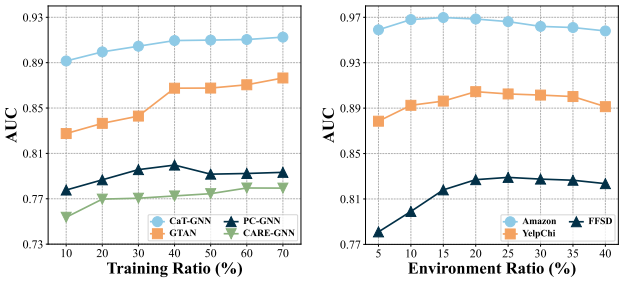


Figure 5: Sensitivity analysis with respect to different training ratios (**Left**) and environment ratios (**Right**).

increase. This negligible rise in time is a testament to the algorithm's ability to retain its computational efficiency while incorporating our advancements. Thus, our algorithm stands as a robust solution that can Cater to the needs of high-performance computing while facilitating enhancements that do not compromise on efficiency.

Table 3: Experimental run times with and without causal intervention on three datasets. The experiments were conducted on a Tesla V100 40GB GPU, with the execution times measured in seconds.

Dataset	YelpChi	Amazon	FFSD
No-intervention	126.676	110.518	208.085
Causal-intervention	129.481 (+2.21%)	113.660 (+2.84%)	213.341 (+2.52%)

Related Works

Graph Neural Network and its Variants. Graph neural networks have been widely used in structured data prediction (Abadal et al. 2021; Wu et al. 2020; Jiang et al. 2024) by integrating graph structure and attribute. With the development of GNNs, there are several types of GNNs nowadays. **1**): ***Recurrent Graph Neural Networks*** (RecGNNs) aim to learn node representations with recurrent neural architectures: (Scarselli et al. 2008; Gallicchio and Micheli 2010; Li et al. 2015; Dai et al. 2018). **2**): ***Convolutional Graph Neural Networks*** (ConvGNNs) generalize the operation of convolution from grid data to graph data: (Li et al. 2018; Zhuang and Ma 2018; Xu et al. 2018; Chiang et al. 2019). **3**): ***Spatial–Temporal Graph Neural Networks*** (STGNNs) aim to learn complex hidden patterns from spatial-temporal graphs: (Yan, Xiong, and Lin 2018; Wu et al. 2019; Guo et al. 2019; Jiang et al. 2023; Li et al. 2022).

Machine-learning based credit card fraud detection. Recently, numerous efforts have been dedicated to integrating machine learning methodologies into the research of credit card fraud detection. For instance, (Maes et al. 2002) successfully applied Bayesian Belief Networks and MLP to the Europay International dataset. (Şahin and Duman 2011) utilized decision trees and support vector machines on data from a major national bank. (Fu et al. 2016) demonstrates that convolutional neural networks outperform traditional approaches in pattern recognition for higher accuracy. However, their models were limited as they only considered individual transactions or cardholders, missing out on the potential of unlabeled data in real-world transactions (Xiang et al. 2023).

GNN-based credit card fraud detection. More recently, the focus has shifted towards graph-based approaches as the use of graph convolutional networks on datasets with partial labels has been effective for predicting node attributes within citation networks so that many GNN-based fraud detectors have been proposed to detect fraud (Wang et al. 2019; Liu et al. 2018). Concretely, (Dou et al. 2020) introduces CARE-GNN for fraud detection on relational graphs, while (Liu et al. 2021) develops PC-GNN for managing imbalanced learning on graphs. Additionally, (Fiore et al. 2019) presents a generative adversarial network to enhance classification capabilities. (Cheng et al. 2020) suggested a joint feature learning model, concentrating on spatial and temporal patterns. However, these methods often lack the capability to uncover the **causal nature** of each specific case and are easily influenced by the surrounding neighbors due to the aggregation mechanism inherent in GNNs. Towards this end, we *take the first step* to propose a structural causal GNN model, which introduces causal intervention and data augmentation mechanism into the aggregation process of GNN.

Conclusion & Future Work

In this work, we introduce the Causal Temporal Graph Neural Network (CAT-GNN), a causal approach in the domain of credit card fraud detection. Our model innovates by integrating causal learning principles to discern and leverage the intricate relationships within transaction data. We validate the effectiveness of CAT-GNN through comprehensive experiments on diverse datasets, where it consistently outperforms existing techniques. Notably, CAT-GNN not only enhances detection accuracy but also maintains computational efficiency, making it viable for large-scale deployment. Future directions will explore extending this methodology to a broader range of fraudulent activities, with the aim of fortifying the integrity of financial systems globally.

References

Abadal, S.; Jain, A.; Guirado, R.; López-Alonso, J.; and Alarcón, E. 2021. Computing graph neural networks: A survey from algorithms to accelerators. *ACM Computing Surveys (CSUR)*, 54(9): 1–38.

Aleksiejuk, A.; and Hołyst, J. A. 2001. A simple model of bank bankruptcies. *Physica A: Statistical Mechanics and its Applications*, 299(1-2): 198–204.

AlFalahi, L.; and Nobanee, H. 2019. Conceptual Building of Sustainable Economic Growth and Corporate Bankruptcy. *Available at SSRN 3472409*.

Bhattacharyya, S.; Jha, S.; Tharakunnel, K.; and Westland, J. C. 2011. Data mining for credit card fraud: A comparative study. *Decision support systems*, 50(3): 602–613.

Chang, S.; Zhang, Y.; Yu, M.; and Jaakkola, T. 2020. Invariant rationalization. In *International Conference on Machine Learning*, 1448–1458. PMLR.

Cheng, D.; Wang, X.; Zhang, Y.; and Zhang, L. 2020. Graph neural network for fraud detection via spatial-temporal attention. *IEEE Transactions on Knowledge and Data Engineering*, 34(8): 3800–3813.

Chiang, W.-L.; Liu, X.; Si, S.; Li, Y.; Bengio, S.; and Hsieh, C.-J. 2019. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 257–266.

Dai, H.; Kozareva, Z.; Dai, B.; Smola, A.; and Song, L. 2018. Learning steady-states of iterative algorithms over graphs. In *International conference on machine learning*, 1106–1114. PMLR.

Dou, Y.; Liu, Z.; Sun, L.; Deng, Y.; Peng, H.; and Yu, P. S. 2020. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM international conference on information & knowledge management*, 315–324.

Feng, F.; Huang, W.; He, X.; Xin, X.; Wang, Q.; and Chua, T.-S. 2021. Should graph convolution trust neighbors? a simple causal inference method. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1208–1218.

Fiore, U.; De Santis, A.; Perla, F.; Zanetti, P.; and Palmieri, F. 2019. Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. *Information Sciences*, 479: 448–455.

Fu, K.; Cheng, D.; Tu, Y.; and Zhang, L. 2016. Credit card fraud detection using convolutional neural networks. In *Neural Information Processing: 23rd International Conference, ICONIP 2016, Kyoto, Japan, October 16–21, 2016, Proceedings, Part III 23*, 483–490. Springer.

Gallicchio, C.; and Micheli, A. 2010. Graph echo state networks. In *The 2010 international joint conference on neural networks (IJCNN)*, 1–8. IEEE.

Guo, S.; Lin, Y.; Feng, N.; Song, C.; and Wan, H. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 922–929.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; and Leskovec, J. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33: 22118–22133.

Ileberi, E.; Sun, Y.; and Wang, Z. 2022. A machine learning based credit card fraud detection using the GA algorithm for feature selection. *Journal of Big Data*, 9(1): 1–17.

Jiang, M.; Cui, P.; and Faloutsos, C. 2016. Suspicious behavior detection: Current trends and future directions. *IEEE intelligent systems*, 31(1): 31–39.

Jiang, X.; Qin, Z.; Xu, J.; and Ao, X. 2024. Incomplete Graph Learning via Attribute-Structure Decoupled Variational Auto-Encoder. In *WSDM*.

Jiang, X.; Zhuang, D.; Zhang, X.; Chen, H.; Luo, J.; and Gao, X. 2023. Uncertainty Quantification via Spatial-Temporal Tweedie Model for Zero-inflated and Long-tail Travel Demand Prediction. In *CIKM*.

Li, R.; Wang, S.; Zhu, F.; and Huang, J. 2018. Adaptive graph convolutional neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Li, R.; Zhong, T.; Jiang, X.; Trajcevski, G.; Wu, J.; and Zhou, F. 2022. Mining Spatio-Temporal Relations via Self-Paced Graph Contrastive Learning. In *SIGKDD*.

Li, Y.; Tarlow, D.; Brockschmidt, M.; and Zemel, R. 2015. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*.

Liu, Y.; Ao, X.; Qin, Z.; Chi, J.; Feng, J.; Yang, H.; and He, Q. 2021. Pick and choose: a GNN-based imbalanced learning approach for fraud detection. In *Proceedings of the web conference 2021*, 3168–3177.

Liu, Y.; Cadei, R.; Schweizer, J.; Bahmani, S.; and Alahi, A. 2022. Towards robust and adaptive motion forecasting: A causal representation perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17081–17092.

Liu, Z.; Chen, C.; Yang, X.; Zhou, J.; Li, X.; and Song, L. 2018. Heterogeneous graph neural networks for malicious account detection. In *Proceedings of the 27th ACM international conference on information and knowledge management*, 2077–2085.

Liu, Z.; Dou, Y.; Yu, P. S.; Deng, Y.; and Peng, H. 2020. Alleviating the inconsistency problem of applying graph neural network to fraud detection. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, 1569–1572.

Maes, S.; Tuyls, K.; Vanschoenwinkel, B.; and Manderick, B. 2002. Credit card fraud detection using Bayesian and neural networks. In *Proceedings of the 1st international naiso congress on neuro fuzzy technologies*, volume 261, 270.

McAuley, J. J.; and Leskovec, J. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*, 897–908.

Pearl, J. 2009. *Causality*. Cambridge university press.

Pearl, J.; and Mackenzie, D. 2018. *The book of why: the new science of cause and effect*. Basic books.

Rayana, S.; and Akoglu, L. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*, 985–994.

Şahin, Y. G.; and Duman, E. 2011. Detecting credit card fraud by decision trees and support vector machines.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2008. The graph neural network model. *IEEE transactions on neural networks*, 20(1): 61–80.

Shi, Y.; Huang, Z.; Feng, S.; Zhong, H.; Wang, W.; and Sun, Y. 2020. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, D.; Lin, J.; Cui, P.; Jia, Q.; Wang, Z.; Fang, Y.; Yu, Q.; Zhou, J.; Yang, S.; and Qi, Y. 2019. A semi-supervised graph attentive network for financial fraud detection. In *2019 IEEE International Conference on Data Mining (ICDM)*, 598–607. IEEE.

Wang, Y.; Wang, W.; Liang, Y.; Cai, Y.; Liu, J.; and Hooi, B. 2020. Nodeaug: Semi-supervised node classification with data augmentation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 207–217.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1): 4–24.

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; and Zhang, C. 2019. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*.

Xiang, S.; Cheng, D.; Shang, C.; Zhang, Y.; and Liang, Y. 2022. Temporal and Heterogeneous Graph Neural Network for Financial Time Series Prediction. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 3584–3593.

Xiang, S.; Zhu, M.; Cheng, D.; Li, E.; Zhao, R.; Ouyang, Y.; Chen, L.; and Zheng, Y. 2023. Semi-supervised credit card fraud detection via attribute-driven graph representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 14557–14565.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

Yan, S.; Xiong, Y.; and Lin, D. 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Zhang, Y.; Fan, Y.; Ye, Y.; Zhao, L.; and Shi, C. 2019. Key player identification in underground forums over attributed heterogeneous information network embedding framework. In *Proceedings of the 28th ACM international conference on information and knowledge management*, 549–558.

Zhuang, C.; and Ma, Q. 2018. Dual graph convolutional networks for graph-based semi-supervised classification. In *Proceedings of the 2018 world wide web conference*, 499–508.

AAAI Reproducibility Checklist

This paper:

- Includes a conceptual outline and/or pseudocode description of AI methods introduced([Yes])
- Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results ([Yes])
- Provides well marked pedagogical references for less-familiare readers to gain background necessary to replicate the paper ([Yes])

Does this paper make theoretical contributions? ([Yes])
If yes, please complete the list below.

- All assumptions and restrictions are stated clearly and formally. ([Yes])
- All novel claims are stated formally (e.g., in theorem statements). ([Yes])
- Proofs of all novel claims are included. ([Yes])
- Proof sketches or intuitions are given for complex and/or novel results. ([Yes])
- Appropriate citations to theoretical tools used are given. ([Yes])
- All theoretical claims are demonstrated empirically to hold. ([Yes])
- All experimental code used to eliminate or disprove claims is included. ([Yes])

Does this paper rely on one or more datasets? ([Yes])
If yes, please complete the list below.

- A motivation is given for why the experiments are conducted on the selected datasets ([Yes])
- All novel datasets introduced in this paper are included in a data appendix. ([NA])
- All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. ([NA])
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations. ([Yes])
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available. ([Yes])
- All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing. ([NA])

Does this paper include computational experiments? ([Yes])

If yes, please complete the list below.

- Any code required for pre-processing data is included in the appendix.([Yes]).
- All source code required for conducting and analyzing the experiments is included in a code appendix. ([Yes])
- All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. ([Yes])
- All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from ([Yes])
- If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results. ([Yes])
- This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks. ([Yes])
- This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics. ([Yes])
- This paper states the number of algorithm runs used to compute each reported result. ([Yes])
- Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information. ([Yes])
- The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank). ([Yes])
- This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments. ([Yes])
- This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting. ([Yes])