Bkd-FedGNN: A Benchmark for Classification Backdoor Attacks on Federated Graph Neural Network

Fan Liu[1], Siqi Lai[1], Yansong Ning[1], and Hao Liu[1,2](✉)

[1] AI Thrust, The Hong Kong University of Science and Technology (Guangzhou), China
{fliu236, slai125, yansongning@hkust-gz.edu.cn}@connect.hkust-gz.edu.cn
[2] CSE, The Hong Kong University of Science and Technology, China
liuh@ust.hk

Abstract. Federated Graph Neural Network (FedGNN) has recently emerged as a rapidly growing research topic, as it integrates the strengths of graph neural networks and federated learning to enable advanced machine learning applications without direct access to sensitive data. Despite its advantages, the distributed nature of FedGNN introduces additional vulnerabilities, particularly backdoor attacks stemming from malicious participants. Although graph backdoor attacks have been explored, the compounded complexity introduced by the combination of GNNs and federated learning has hindered a comprehensive understanding of these attacks, as existing research lacks extensive benchmark coverage and in-depth analysis of critical factors. To address these limitations, we propose Bkd-FedGNN, a benchmark for backdoor attacks on FedGNN. Specifically, Bkd-FedGNN decomposes the graph backdoor attack into trigger generation and injection steps, and extending the attack to the node-level federated setting, resulting in a unified framework that covers both node-level and graph-level classification tasks. Moreover, we thoroughly investigate the impact of multiple critical factors in backdoor attacks on FedGNN. These factors are categorized into global-level and local-level factors, including data distribution, the number of malicious attackers, attack time, overlapping rate, trigger size, trigger type, trigger position, and poisoning rate. Finally, we conduct comprehensive evaluations on 13 benchmark datasets and 13 critical factors, comprising 1,725 experimental configurations for node-level and graph-level tasks from six domains. These experiments encompass over 8,000 individual tests, allowing us to provide a thorough evaluation and insightful observations that advance our understanding of backdoor attacks on FedGNN. Our code is available at https://github.com/usail-hkust/BkdFedGCN

Keywords: Federated graph learning · Backdoor attack · Graph learning.

1 Introduction

The Federated Graph Neural Network (FedGNN) has emerged as a fast-evolving research area that combines the capabilities of graph neural networks and feder-

ated learning. Such integration allows for advanced machine learning applications without requiring direct access to sensitive data [26,17,16,25,15]. However, despite its numerous advantages, the distributed nature of FedGNN introduces additional vulnerabilities, particularly related to backdoor attacks originating from malicious participants. In particular, these adversaries have the ability to inject graph backdoor triggers into their training data, thereby undermining the overall trustworthiness of the system [42,21,29,24].

Although considerable research efforts have explored graph backdoor attacks on FedGNN [5,44,12,46], a comprehensive understanding of these attacks is hindered by the compounded complexity introduced by the combination of Graph Neural Networks (GNNs) and Federated Learning (FL). Existing studies suffer from a lack of extensive benchmark coverage and in-depth analysis of critical factors. **(1) Lack of Extensive Benchmark Coverage.** Specifically, the lack of extensive benchmark coverage poses challenges in fairly and comprehensively comparing graph backdoor attacks on FedGNN across different settings. These settings can be categorized into two levels: the graph backdoor attack level and the FedGNN task level. At the graph backdoor attack level, trigger generation and injection steps are involved. Additionally, the classification tasks in FedGNN encompass both node and graph classification tasks. However, there is still a dearth of comprehensive exploration of graph backdoor attacks on FedGNN under these various settings. **(2) Insufficient Exploration of Multiple Factors.** Furthermore, there has been the insufficient exploration of multiple factors that impact FedGNN. The combination of GNN with FL introduces various factors that affect backdoor attacks, such as trigger type, trigger size, and data distribution. The insufficient exploration and analysis of these multiple factors make it difficult to understand the influence of key factors on the behavior of FedGNN.

To address these limitations, we propose a benchmark for graph backdoor attacks on FedGNN, called Bkd-FedGNN. As far as we are aware, our work is the first comprehensive investigation of graph backdoor attacks on FedGNN. Our contributions can be summarized as follows. (1) **Unified Framework**: We propose a unified framework for classification backdoor attacks on FedGNN. Bkd-FedGNN decomposes the graph backdoor attack into trigger generation and injection steps and extends the attack to the node-level federated setting, resulting in a unified framework that covers both node-level and graph-level classification tasks. (2) **Exploration of Multiple Critical Factors**: We thoroughly investigate the impact of multiple critical factors on graph backdoor attacks in FedGNN. We systematically categorize these factors into two levels: global level and local level. At the global level, factors such as data distribution, the number of malicious attackers, the start time of backdoor attacks, and the overlapping rate play significant roles. In addition, the local level factors involve factors such as trigger size, trigger type, trigger position, and poisoning rate. (3) **Comprehensive Experiments and Analysis**: We conduct comprehensive experiments on both benchmark experiments and critical factor analysis. For the benchmark experiments, we consider combinations of trigger types, trigger positions, datasets, and

models, resulting in 315 configurations for the node level and 270 configurations for the graph-level tasks. Regarding the critical factors, we consider combinations of factors, datasets, and models, resulting in 672 configurations for the node-level tasks and 468 configurations for the graph-level tasks. Each configuration is tested five times, resulting in approximately 8,000 individual experiments in total. Based on these experiments, we thoroughly evaluate the presented comprehensive analysis and provide insightful observations that advance the field.

2 Federated Graph Neural Network

In this section, we provide an introduction to the preliminary aspects of FedGNN. Currently, FedGNN primarily focuses on exploring common classification tasks, which involve both node-level and graph-level classification. The FedGNN consists of two levels: client-level local training and server-level federated optimization. We will begin by providing an overview of the notations used, followed by a detailed explanation of the client-level local training, which encompasses message passing and readout techniques. Lastly, we will introduce server-level federated optimization.

2.1 Notations

Assume that there exist K clients denoted as $\mathcal{C} = \{c_k\}_{k=1}^{K}$. Each client, c_i, possesses a private dataset denoted as $\mathcal{D}^i = \{(\mathcal{G}_j^i, \mathbf{Y}_j^i)\}_{j=1}^{N_i}$, wherein $\mathcal{G}_j^i = (\mathcal{V}_j^i, \mathcal{E}_j^i)$ is the graph, where $\mathcal{V}^i = \{v_t\}_{t=1}^{n_i}$ (n_i denotes the number of nodes) is the set of nodes, and $\mathcal{E}^i = \{e_{tk}\}_{t,k}$ is the set of edges (for simplicity, we exclude the subscript j that indicates the index of the j-th dataset in the dataset $\mathcal{D}^i$). $N_i = \left|\mathcal{D}^i\right|$ denotes the total number of data samples in the private dataset of client c_i. We employ the notation $\mathbf{A}_j^i$ to denote the adjacency matrix of graph $\mathcal{G}_j^i$ belonging to client c_i within the set of clients $\mathcal{C}$. $\mathbf{X}_j^i$ represents the node feature set , and $\mathbf{Y}_j^i$ corresponds to the label sets.

2.2 Client-level Local Training

To ensure versatility and inclusiveness, we employ the message passing neural network (MPNN) framework [9,32,14], which encompasses a diverse range of spectral-based GNNs, such as GCN [19], as well as spatial-based GNNs including GAT [37] and GraphSage [13], *etc.* Each client possesses a GNN model that collaboratively trains a global model. The local graph learning process can be divided into two stages: message passing and readout.

Message Passing. For each client c_i, the l-th layer in MPNN can be formulated as follows,

$$\mathbf{h}_j^{l,i} = \sigma(w^{l,i} \cdot (\mathbf{h}_j^{l-1,i}, Agg(\{\mathbf{h}_k^{l-1,i} | v_k \in \mathcal{N}(v_j)\}))), \tag{1}$$

where $\mathbf{h}_j^{l,i}$ ($l = 0, \cdots, L-1$) represents the hidden feature of node v_j in client c_i and $\mathbf{h}_j^{0,i} = \mathbf{x}_j$ denotes the node v_j's raw feature. The σ represents the activation

function (e.g., ReLU, sigmoid). The parameter $w^{l,i}$ corresponds to the l-th learnable parameter. The aggregation operation Agg (e.g., mean pooling) combines the hidden features $\mathbf{h}_k^{l-1,i}$ of neighboring nodes $v_k \in \mathcal{N}(v_j)$ for node v_j, where $\mathcal{N}(v_j)$ represents the set of neighbors of node v_j. Assume that the $\mathbf{w}^i = \{w^{l,i}\}_{l=0}^{L-1}$ is the set of learn able parameters for client c_i.

Readout. Following the propagation of information through L layers of MPNN, the final hidden feature is computed using a readout function for subsequent tasks.

$$\hat{y}_I^i = R_{\theta^i}(\{\mathbf{h}_j^{L,i} | v_j \in \mathcal{V}_I^i\}), \tag{2}$$

where $\hat{y}_I^i$ represents the prediction for a node or graph. Specifically, I serves as an indicator, where $I = v_j$ denotes the prediction for node v_j, and $I = \mathcal{G}^i$ denotes the prediction for the graph $\mathcal{G}^i$. The readout function $R_{\theta^i}(\cdot)$ encompasses methods such as mean pooling or sum pooling *etc.*, where θ^i is the parameter for readout function.

2.3 Server-level Federated Optimization

Let us consider that $\mathbf{w}^i = \{w^{l,i}\}_{l=0}^{L-1}$ represents the set of trainable parameters within the MPNN framework associated with client c_i. Consequently, we define the overall model parameters as $\mathbf{W}^i = \{\mathbf{w}^i, \theta^i\}$ for each client $c_i \in \mathcal{C}$. The GNNs, which constitute a part of this framework, can be represented as $f_i(\mathbf{X}_j^i, \mathbf{A}_j^i; \mathbf{W}^i)$. The objective of FL is to optimize the global objective function while preserving the privacy of local data on each individual local model. The overall objective function can be formulated as follows,

$$\min_{\{\mathbf{W}^i\}} \sum_{i \in \mathcal{C}} \frac{N_i}{N} F_i(\mathbf{W}^i), \quad F_i(\mathbf{W}^i) = \frac{1}{N_i} \sum_{j \in \mathcal{D}^i} \mathcal{L}((f_i(\mathbf{X}_j^i, \mathbf{A}_j^i; \mathbf{W}^i), \mathbf{Y}_j^i), \tag{3}$$

where $F_i(\cdot)$ denotes the local objective function, and $\mathcal{L}(\cdot)$ denote the loss function (*e.g.*, cross-entropy *etc.*), and $N = \sum_{i=1}^{K} N_i$ represent the total number of data samples encompassing all clients.

We illustrate the process of federated optimization, aimed at achieving a generalized model while ensuring privacy preservation, by utilizing a representative federated algorithm, FedAvg [28]. Specifically, in each round denoted by t, the central server transmits the global model parameter $\mathbf{W}_t$ to a subset of clients that have been selected for local training. Subsequently, each chosen client c_i refines the received parameter $\mathbf{W}_t$ using an optimizer operating on its private dataset $\mathcal{D}^i$. Following this, the selected clients upload the updated model parameter $\mathbf{W}_t^i$, and the central server aggregates the local model parameters to obtain the enhanced global model parameter $\mathbf{W}_{t+1}$.

In FedGNN setting, there exist diverse scenarios involving distributed graphs that are motivated by real-world applications. In these scenarios, classification tasks can be classified into two distinct settings based on how graphs are distributed across clients. **Node-level FedGNN**. Each client is equipped with a subgraph, and the prevalent tasks involve node classification. Real-world applications, such as social networks, demonstrate situations where relationships

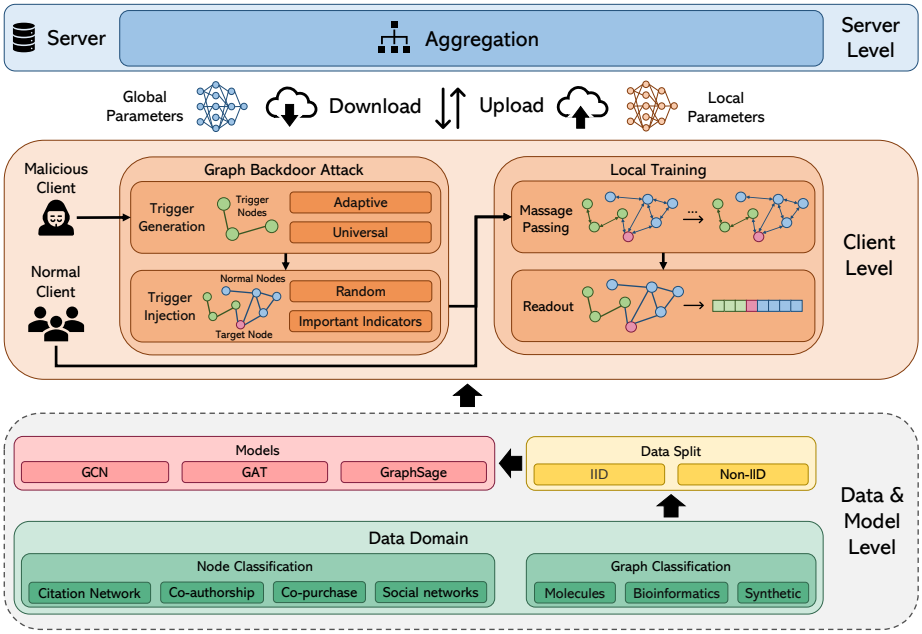

Fig. 1: A unified framework for classification backdoor attack on FedGNN.

between nodes can span across different clients, and each node possesses a unique label. **Graph-level FedGNN**. Each client possesses a set of graphs, and the primary focus lies on graph classification tasks. Real-world applications, such as protein discovery, exemplify instances where each institution holds a limited graph along with associated labels.

3 A Unified Framework for Classification Backdoor Attack on FedGNN

This section presents a unified framework for classification backdoor attacks on federated GNNs. Our primary focus is on graph-based backdoor attacks, where malicious entities strategically insert triggers into graphs or subgraphs to compromise the trustworthiness of FedGNN. A comprehensive illustration of our unified framework for classification backdoor attacks on FedGNN can be found in Figure 1. In detail, we first introduce the dataset and models and then give the evaluation metric, then introduce the threat model. Next, we introduce the federated graph backdoor attack, which involves the formulation of the attack goal and a two-step attack process: trigger generation and trigger injection. Finally, we explore various critical factors at both global and local levels.

3.1 Datasets and Models

In this study, we have considered six distinct domains comprising a total of thirteen datasets, along with three widely used GNNs. *Node-level Datasets:* For

node-level analysis, we have included three extensively studied citation graphs, such as Cora, CiteSeer, and PubMed. Additionally, we have incorporated the Co-authorship graphs (CS and Physics), along with the Amazon Co-purchase graphs (Photo and Computers). *Graph-level Datasets:* For graph-level analysis, we have utilized molecular graphs such as AIDS and NCI1. Furthermore, bioinformatics graphs, including PROTEINS-full, DD, and ENZYMES, have been incorporated. Lastly, a synthetic graph, COLORS-3, has also been employed. *Models:* We have employed three widely adopted GNNs: GCN, GAT, and GraphSage, which have been demonstrated effective in various graph-based tasks. For detailed statistical information about the graphs used, please refer to Appendix A.1.

3.2 Evaluation Metrics

To assess the effectiveness of the graph backdoor attack on FedGNN, three metrics are employed: the average clean accuracy (ACC) across all clients, the average attack success rate (ASR) on malicious clients, and the transferred attack success rate (TAST) on normal clients. The ACC metric evaluates the performance of federated GNNs when exposed to clean examples from all clients. The ASR metric measures the performance of the graph backdoor attack specifically on the malicious clients. Lastly, the TAST metric gauges the vulnerability of normal clients to the graph backdoor attack. For the detailed equations corresponding to these metrics, please refer to Appendix A.2.

3.3 Threat Model

Attack Objective. Assuming there are a total of K clients, with M ($M \leq K$) of them being malicious, each malicious attacker independently conducts the backdoor attack on their own models. The primary goal of a backdoor attack is to manipulate the model in such a way that it misclassifies specific pre-defined labels (known as target labels) only within the poisoned data samples. It is important to ensure that the model's accuracy remains unaffected when processing clean data. **Attack Knowledge.** In this setting, we assume that the malicious attacker has complete knowledge of their own training data. They have the capability to generate triggers. It should be noted that this scenario is quite practical since the clients have full control over their own data. **Attacker Capability.** The malicious client has the ability to inject triggers into the training datasets, but this capability is limited within predetermined constraints such as trigger size and poisoned data rate. The intention is to contaminate the training datasets. However, the malicious client lacks the ability to manipulate the server-side aggregation process or interfere with other clients' training processes and models.

3.4 Federated Graph Backdoor Attack

Mathematically, the formal attack objective for each malicious client c_i during round t can be defined as follows,

$$\begin{aligned} &\mathbf{W}_t^{i*} = \arg\min_{\mathbf{W}_t{}^i} \frac{1}{N_i} \left[\sum_{j \in \mathcal{D}_p^i} \mathcal{L}((f_i(\mathbf{X}_j^i, g_\tau \circ \mathbf{A}_j^i; \mathbf{W}_{t-1}^i), \tau) + \sum_{j \in \mathcal{D}_c^i} \mathcal{L}((f_i(\mathbf{X}_j^i, \mathbf{A}_j^i; \mathbf{W}_{t-1}^i), \mathbf{Y}_j^i) \right], \\ &\forall j \in \mathcal{D}_p^i, N_\tau = |g_\tau| \leq \triangle_g \quad \text{and} \quad \rho = \frac{|\mathcal{D}_p^i|}{|\mathcal{D}^i|} \leq \triangle_p, \end{aligned} \tag{4}$$

where $\mathcal{D}_p^i$ refers to the set of poisoned data and $\mathcal{D}_c^i$ corresponds to the clean dataset. Noted that $\mathcal{D}_p^i \sqcup \mathcal{D}_c^i = \mathcal{D}^i$ and $\mathcal{D}_p^i \sqcap \mathcal{D}_c^i = \phi$, indicating the union and intersection of the poisoned and clean data sets, respectively. $g_\tau \circ \mathbf{A}_j^i$ represents the poisoned graph resulting from an attack. g_τ represents the trigger generated by the attacker, which is then embedded into the clean graph, thereby contaminating the datasets. Additionally, τ denotes the target label. $N_\tau = |g_\tau|$ denotes the trigger size and $\triangle_g$ represents the constrain to ensures that the trigger size remains within the specified limit. $\rho = \frac{|\mathcal{D}_p{}^i|}{|\mathcal{D}^i|}$ represents the poisoned rate, and $\triangle p$ denotes the budget allocated for poisoned data.

In the federated graph backdoor attack, to generate the trigger and poisoned data sets, the graph backdoor attack can be divided into two steps: trigger generation and trigger injection. The term "trigger" (a specific pattern) has been formally defined as a subgraph in the work by Zhang *et al.* (2021), providing a clear and established framework for its characterization [50].

Trigger Generation. The process of trigger generation can be defined as the function $\varphi(\mathbf{X}_j^i, \mathbf{A}_j^i)$, which yields the generated trigger g_τ through $\varphi(\mathbf{X}_j^i, \mathbf{A}_j^i) = g_\tau$.

Trigger Injection. The process of trigger injection can be defined as the function $a(g_\tau, \mathbf{A}_j^i)$, which generates the final poisoned graph $g_\tau \circ \mathbf{A}_j^i$ by incorporating the trigger g_τ into the pristine graph $\mathbf{A}_j^i$.

3.5 Factors in Federated Graph Backdoor

The graph backdoor attack framework in FedGNN encompasses various critical factors that warrant exploration. These factors can be categorized into two levels: the global level and the local level. At the global level, factors such as data distribution, the number of malicious attackers, the start time of backdoor attacks, and overlapping rate play significant roles. On the other hand, the local level involves parameters like trigger size, trigger type, trigger position, and poisoning rate. Notably, the overlapping rate holds particular importance in node-level FedGNN, as it involves cross-nodes across multiple clients.

Global Level Factors: **Data Distribution.** The data distribution encompasses two distinct types: independent and identically distributed (IID) and non-independent and identically distributed (Non-IID). In detail, IID refers to data distribution among clients remaining constant, while Non-IID (L-Non-IID [39,49], PD-Non-IID [7], N-Non-IID [22]) refers that the data distribution among clients exhibiting variations. **Number of Malicious Attackers.** The concept of the number of malicious attackers, denoted as M, can be defined in the following manner. Let us assume that the set of malicious clients is denoted as $\mathcal{C}_m$, and the set of normal clients is denoted as $\mathcal{C}_n$. It can be inferred that $\mathcal{C}_m \sqcup \mathcal{C}_n = \mathcal{C}$ and $\mathcal{C}_m \sqcap \mathcal{C}_c = \phi$. **Attack Time.** In the context of FL, the attack time denotes the precise moment when a malicious attack is launched. The attack time can be denoted by t^*. **Overlapping Rate (specific to Node-level FedGNN).** The overlapping rate, represented by the variable α, pertains to the proportion of additional samples of overlapping data that across clients. This phenomenon

Table 1: Critical factors in federated graph backdoor.

	Factors	Symbol	Node Level	Graph Level
Global Level	Data Distribution	-	{IID*, L-Non-IID}	{IID*, PD-Non-IID, N-Non-IID }
	# of Malicious Attackers	M	$\{1^*, 2, 3, 4, 5\}$	
	Attack Time	t^*	$T * \{0.0^*, 0.1, 0.2, 0.3, 0.4, 0.5\}$	
	Overlapping Rate	α	$\{0.1^*, 0.2, 0.3, 0.4, 0.5\}$	-
Local Level	Trigger Size	N_τ	$\{3^*, 4, 5, 6, 7, 8, 9, 10\}$	$N_d * \{0.1^*, 0.2, 0.3, 0.4, 0.5\}$
	Trigger Type	g_τ	{Renyi*, WS, BA, GTA, UGBA }	{ Renyi*, WS, BA, RR, GTA }
	Trigger Position	-	{Random*, Degree, Cluster }	
	Poisoning Rate	ρ	$\{0.1^*, 0.2, 0.3, 0.4, 0.5\}$	

arises in node-level FedGNN, where cross-client nodes exist, resulting in the sharing of common data samples between different clients.

Local Level Factors: **Trigger Size.** The size of the trigger can be quantified by counting the number of nodes within the corresponding graph. The trigger size is denoted by N_τ. **Trigger Type.** Based on the methods used to generate triggers(*e.g.*, Renyi [50], WS [40], BA [1], RR [35], GTA [41], and UGBA [6] *etc.*), the categorization of trigger types can be refined into two categories: universal triggers and adaptive triggers. Universal triggers are pre-generated through graph generation techniques, such as the Erdős-Rényi (ER) model [8], which are agnostic to the underlying graph datasets. On the other hand, adaptive triggers are specifically designed for individual graphs using optimization methods. **Trigger Position.** The trigger position refers to the specific location within a graph or sub-graph where the trigger is injected. Typically, the trigger position can be categorized into two types: random position and important indicator position. In the case of the random position, the trigger is injected into the graph in a random manner without any specific consideration. Conversely, the important indicator position entails injecting the trigger based on certain crucial centrality values, such as the degree or cluster-based scores, that indicate the significance of specific nodes within the graph. **Poisoning Rate.** The concept of poisoning rate, denoted as ρ, can be defined as the ratio of the cardinality of the set of poisoned data samples, $\mathcal{C}_p^i$, to the total number of data samples, denoted as $\mathcal{D}^i$. Mathematically, this can be expressed as $\rho = \frac{|\mathcal{D}_p^i|}{|\mathcal{D}^i|}$, where $\forall c_i \in \mathcal{C}$ signifies that the cardinality calculations are performed for every client c_i belonging to the set $\mathcal{C}$.

4 Experimental Studies

In this section, we present the experimental studies conducted to investigate classification backdoor attacks on FedGNN. Our main objective is to evaluate the impact of graph backdoor attacks on FedGNN covering both the node and graph level tasks. Additionally, we aim to explore the critical factors that influence the effectiveness of graph backdoor attacks on FedGNN, considering aspects from both the global and local levels.

4.1 Experimental Settings

Factors Settings. We present the detailed factors setup considered in our study. It is important to note that the first value presented represents the default setting.

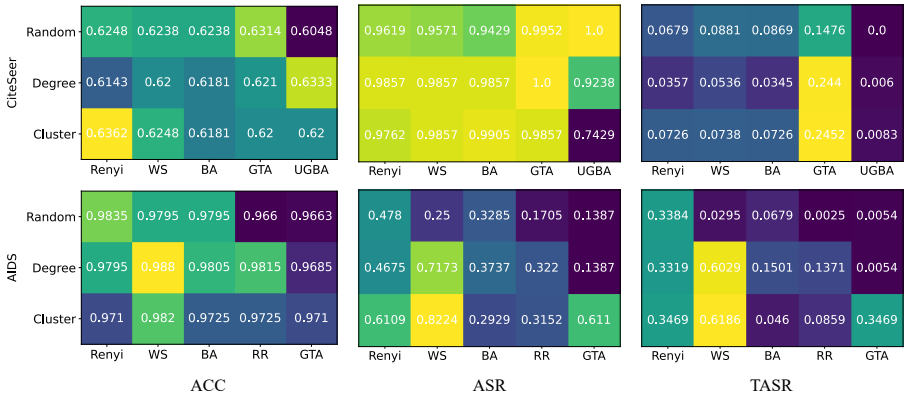

Fig. 2: Graph backdoor attack on both node and graph level tasks for GCN. (Color intensity corresponds to value magnitude)

To assess the individual impact of each factor, we keep the remaining factors fixed while systematically varying the corresponding values in our experiments. The factors range is shown in Table 1. For the detailed setting for factor, please refer to Appendix A.3.

Federated Graph Backdoor Attack. The federated graph backdoor attack can be characterized by the combination of trigger generation techniques (Renyi [50], WS [40], BA [1], RR [35], GTA [41], and UGBA [6]) and trigger position strategies (Random, Degree, and Cluster). For instance, the attack method Renyi-Random refers to the utilization of the ER model to generate the trigger, which is then randomly injected into the graph.

Implementation Details. Our implementation of the backdoor attack on FedGNN is based on the PyTorch framework. The experiments were carried out on two server configurations: three Linux Centos Servers, each with 4 RTX 3090 GPUs, and two Linux Ubuntu Servers, each with 2 V100 GPUs. In both node-level and graph-level tasks, we adopt the inductive learning settings as outlined in [44,6]. For each dataset, we ensure consistent experimental conditions by employing the same training and attack settings. We set the total number of clients to 5, and all clients participate in the training process at each round. Each experiment is repeated five times. For a detailed description of the training and attack settings, please refer to Appendix A.4.

4.2 Benchmark Results of Graph Backdoor Attack on FedGNN

The results of the benchmark for the graph backdoor attack on FedGNN are presented in Figure 2. The observations are summarized as follows. (1) The node-level task exhibits higher vulnerability to attacks compared to the graph-level task at a relatively small trigger size. Specifically, a significant majority of graph backdoor attacks achieve an ASR (Attack Success Rate) exceeding 90%,

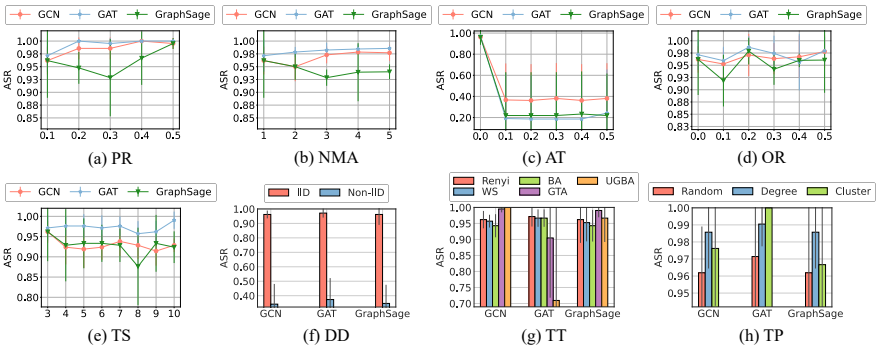

Fig. 3: Node-level task factors.

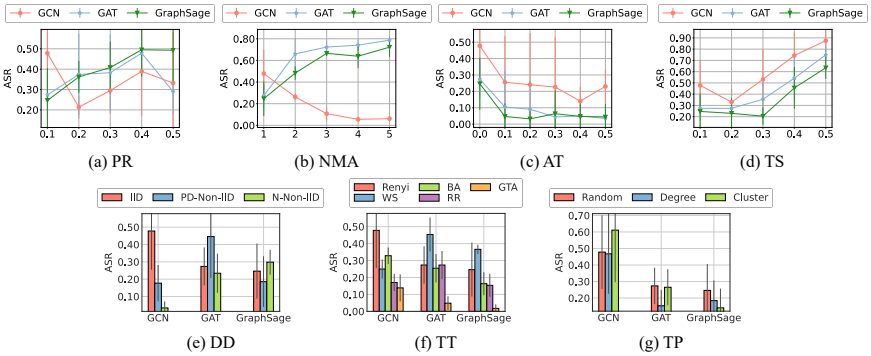

Fig. 4: Graph-level task factors.

while the highest ASR recorded at the graph level is 82.24%. (2) Despite not being intentionally poisoned by malicious attackers, the normal clients are still susceptible to graph backdoor attacks. For instance, in the node-level task, there is a TASR (Transfered Attack Success Rate) of 24.52%, while the graph-level task exhibits even higher vulnerability with a TASR of 61.86%. This observation suggests that the weights uploaded by the malicious clients can inadvertently influence the normal clients when they download the global model's weights. 3). The combination of trigger size and trigger position significantly influences the attack performance on the graph-level task compared to the node-level task. For instance, the attack WS-Cluster achieves an ASR of approximately 82.24%, while the GTA-Random achieves only about 13.87%. Due to the page limit, the benchmark results on other datasets and models please refer to Appendix A.5.

4.3 Factors in Federated GNN

The overall results of factors can be shown in Figures 3-4. *Global Level Factors:* **Data Distribution (DD).** For node-level tasks, there models trained on IID data are more vulnerable than models trained on Non-IID data. For graph-level tasks, the GCN trained on IID data are more vulnerable than models trained on Non-IID data (PD-Non-IID and N-Non-IID), while GAT and GraphSagr trained

on Non-IID data are more vulnerable than models trained on IID data. **Number of Malicious Attackers (NMA).** For node-level tasks, an increase in NMA leads to an increase in ASR for both GCN and GAT models. Conversely, an increase in NMA results in a decrease in ASR for both GraphSage. Concerning graph-level tasks, the ASR demonstrates an increase with the increase of NMA in the case of GAT and GraphSage. However, in the scenario of GCN, the ASR shows a decrease with the increase of NMA. **Attack Time (AT).** For both node-level and graph-level tasks, an increase in AT results in a decrease in ASR for three models. **Overlapping Rate (OR).** The ASR demonstrates an upward trend as the overlapping rate increases. This correlation can be attributed to the possibility that overlapping nodes facilitate the backdooring of normal clients, primarily through the presence of cross-edges.

Local Level Factors: **Trigger Size (TS).** For node-level tasks, an increase in TS leads to an increase in ASR for GCN. However, in the case of GAT and GraphSage, the ASR demonstrates a decrease with the increase of TS. Concerning the graph-level task, the ASR shows an increase with the increase of TS across all three GNNs. **Trigger Types (TT).** In the node-level task, the adaptive trigger demonstrates a higher ASR on most models. Conversely, in the graph-level task, the universal trigger exhibits higher ASR. **Trigger Position (TP).** In node-level tasks, we observed a significantly large ASR when using importance-based positions (Degree and Cluster) compared to random positions. However, for the graph-level task, while importance-based positions showed higher ASR for GCN, random positions yielded higher ASR for GAT and GraphSage. **Poisoning Rate (PR).** On node classification, an increase in PR results in a slight decrease in ASR. However, graph classification exhibits an upward trend in ASR. Due to the page limit, the results on other datasets and metrics, please refer to Appendix A.5.

4.4 Defense Methods Against Federated Graph Backdoor Attack

To comprehensively evaluate the impact of the graph backdoor attack on FedGNN, considering both adaptive optimizer settings and defense strategies, we conduct additional experiments utilizing state-of-the-art federated algorithms and defense techniques. This involves advanced federated algorithms (FedOpt [30], FedProx [23], and Scaffold [18]) the discarding aggregation methods (e.g., Krum [2], Multi-Krum [2], Bulyan) and non-discarding aggregations (e.g., Median, Trimmed-mean). The results of the federated defense experiments conducted under the backdoor attack "renyi-random" are illustrated in Figure 5. Overall, the results reveal that even advanced federated methods and defense approaches have limitations in effectively mitigating the graph backdoor attack.

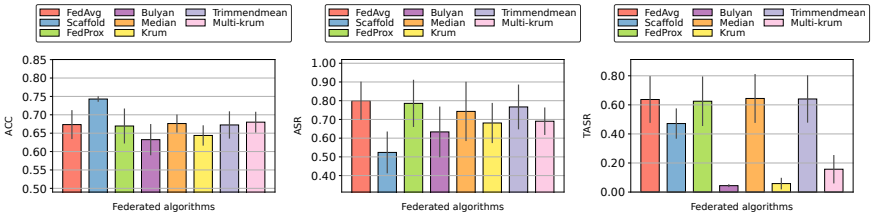

Fig. 5: Advanced federated algorithms Defense methods against backdoor attack.

Takeaways:

(1) Non-IID distribution is more susceptible to malicious activities.
(2) More malicious clients corresponds to higher attack performance.
(3) Malicious clients possess the capacity to initiate attacks during any phase of federated training rounds.
(4) The inclusion of cross-client edges enhances the attack process by facilitating the transfer of malicious trigger knowledge across client, thereby amplifying the trigger signal.
(5) A larger trigger size does not necessarily equate to higher attack capability.
(6) The adaptive trigger is tailored to individual graphs, resulting in a higher attack success rate.
(7) The placement of the trigger in a significant position leads to enhanced attack performance.
(8) A higher poisoning rate corresponds to an elevated attack success rate.

5 Related Works

FedGNN. FedGNN present a distributed machine learning paradigm that facilitates collaborative training of GNNs among multiple parties, ensuring the privacy of their sensitive data. In recent years, extensive research has been conducted on FedGNN, with a particular focus on addressing security concerns [11,44,12,46,10]. Among these concerns, poisoning attacks have garnered significant attention, encompassing both data poisoning attacks and model poisoning attacks. Data poisoning attacks occur when an adversary employs tainted data to train the local model, while model poisoning attacks involve manipulation of either the training process or the local model itself. Currently, the majority of attacks on FedGNN primarily concentrate on data poisoning attacks. Chen *et al.* [5] proposed adversarial attacks on vertical federated learning, utilizing adversarial perturbations on global node embeddings based on gradient leakage from pairwise nodes. Additionally, Xu *et al.* [44] investigated centralized and distributed backdoor attacks on FedGNN.

Graph Backdoor Attacks. Backdoor attacks on GNNs have received significant attention in recent years [50,43,47,45,41,51,6]. Regarding graph backdoor attacks, they can be classified into two types based on the employed trigger: uni-

versal graph backdoor attacks and adaptive backdoor attacks. In universal graph backdoor attacks, Zhang *et al.* [50] generated sub-graphs using the Erdős-Rényi (ER) model as triggers and injected them into the training data. Additionally, Xu *et al.* [41] observed that the position of the trigger injection into the graph can also affect the attack's performance. As for adaptive trigger backdoor attacks, Xi *et al.* [41] developed an adaptive trigger generator that optimizes the attack's effectiveness for both transductive and inductive tasks. In our benchmark, we focus primarily on data poisoning attacks. While model poisoning attacks can be effective, data poisoning attacks may be more convenient because they do not require tampering with the model learning process, and they allow non-expert actors to participate [36].

6 Conclusions and Open Problems

Conclusions. In this paper, we proposed a unified framework for classification backdoor attacks on FedGNN. We then introduced the critical factors involved in graph backdoor attacks on FedGNN, including both global and local level factors. Along this line, we performed approximately 8,000 experiments on the graph backdoor attacks benchmark and conducted critical factor experiments to provide a comprehensive analysis.

Open Problems. (1) Enhancing the success rate of transferred attacks: Our findings reveal that malicious attackers can also backdoor normal clients through the FL mechanism. However, there is a need to explore methods that can identify and exploit the worst vulnerabilities under these circumstances. (2) Evaluating the defense method under backdoor attack: We demonstrate that FedGNN can be compromised by malicious attackers. However, assessing the effectiveness of defense mechanisms against such attacks still requires further exploration. (3) Cooperative malicious attackers: Currently, the majority of malicious attackers operate independently during the attack process, neglecting the potential benefits of collaboration. An intriguing research direction lies in investigating the utilization of collaboration to enhance attack performance.

Acknowledge

This work was supported by the National Key R&D Program of China (Grant No.2023YFF0725004), National Natural Science Foundation of China (Grant No.92370204), the Guangzhou Basic and Applied Basic Research Program under Grant No. 2024A04J3279, Education Bureau of Guangzhou Municipality.

References

1. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. science **286**(5439), 509–512 (1999)

2. Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J.: Machine learning with adversaries: Byzantine tolerant gradient descent. Advances in neural information processing systems **30** (2017)
3. Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S.V.N., Smola, A.J., Kriegel, H.: Protein function prediction via graph kernels. In: Proceedings Thirteenth International Conference on Intelligent Systems for Molecular Biology 2005, Detroit, MI, USA, 25-29 June 2005. pp. 47–56 (2005)
4. Cheibub, J.A., Gandhi, J., Vreeland, J.R.: Democracy and dictatorship revisited. Public choice pp. 67–101 (2010)
5. Chen, J., Huang, G., Zheng, H., Yu, S., Jiang, W., Cui, C.: Graph-fraudster: Adversarial attacks on graph neural network-based vertical federated learning. IEEE Transactions on Computational Social Systems (2022)
6. Dai, E., Lin, M., Zhang, X., Wang, S.: Unnoticeable backdoor attacks on graph neural networks. In: Proceedings of the ACM Web Conference 2023. p. 2263–2273. WWW '23, New York, NY, USA (2023)
7. Fang, M., Cao, X., Jia, J., Gong, N.Z.: Local model poisoning attacks to byzantine-robust federated learning. In: 29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020. pp. 1605–1622. USENIX Association (2020)
8. Gilbert, E.N.: Random graphs. The Annals of Mathematical Statistics **30**(4), 1141–1144 (1959)
9. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017. Proceedings of Machine Learning Research, vol. 70, pp. 1263–1272. PMLR (2017)
10. Guo, Z., Han, R., Liu, H.: Against multifaceted graph heterogeneity via asymmetric federated prompt learning (2024), https://arxiv.org/abs/2411.02003
11. Guo, Z., Yao, D., Yang, Q., Liu, H.: Hifgl: A hierarchical framework for cross-silo cross-device federated graph learning. In: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 968–979. KDD '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3637528.3671660, https://doi.org/10.1145/3637528.3671660
12. Halimi, A., Kadhe, S., Rawat, A., Baracaldo, N.: Federated unlearning: How to efficiently erase a client in FL? CoRR **abs/2207.05521** (2022)
13. Hamilton, W.L., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA. pp. 1024–1034 (2017)
14. Han, J., Liu, H., Xiong, H., Yang, J.: Semi-supervised air quality forecasting via self-supervised hierarchical graph neural network. IEEE Transactions on Knowledge and Data Engineering **35**(5), 5230–5243 (2022)
15. Han, J., Zhang, W., Liu, H., Tao, T., Tan, N., Xiong, H.: Bigst: Linear complexity spatio-temporal graph neural network for traffic forecasting on large-scale road networks. Proc. VLDB Endow. **17**(5), 1081–1090 (Jan 2024). https://doi.org/10.14778/3641204.3641217, https://doi.org/10.14778/3641204.3641217
16. He, C., Ceyani, E., Balasubramanian, K., Annavaram, M., Avestimehr, S.: Spreadgnn: Decentralized multi-task federated learning for graph neural networks on molecular data (2021)
17. Huang, X., Yang, Y., Wang, Y., Wang, C., Zhang, Z., Xu, J., Chen, L., Vazirgiannis, M.: Dgraph: A large-scale financial dataset for graph anomaly detection. In: NeurIPS (2022)

18. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S.J., Stich, S.U., Suresh, A.T.: Scaffold: Stochastic controlled averaging for on-device federated learning. arXiv preprint arXiv:1910.06378 **2**(6) (2019)
19. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net (2017), https://openreview.net/forum?id=SJU4ayYgl
20. Knyazev, B., Taylor, G.W., Amer, M.: Understanding attention and generalization in graph neural networks. Advances in neural information processing systems **32** (2019)
21. Li, H., Wu, C., Zhu, S., Zheng, Z.: Learning to backdoor federated learning. arXiv preprint arXiv:2303.03320 (2023)
22. Li, Q., Diao, Y., Chen, Q., He, B.: Federated learning on non-iid data silos: An experimental study. In: 2022 IEEE 38th International Conference on Data Engineering (ICDE). pp. 965–978. IEEE (2022)
23. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. Proceedings of Machine learning and systems **2**, 429–450 (2020)
24. Liu, F., Feng, Y., Xu, Z., Su, L., Ma, X., Yin, D., Liu, H.: Jailjudge: A comprehensive jailbreak judge benchmark with multi-agent enhanced explanation evaluation framework (2024), https://arxiv.org/abs/2410.12855
25. Liu, F., Liu, H.: Subgraph federated unlearning. In: Proceedings of the ACM on Web Conference 2025. p. 1205–1215. WWW '25, Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3696410.3714821, https://doi.org/10.1145/3696410.3714821
26. Maekawa, S., Noda, K., Sasaki, Y., Onizuka, M.: Beyond real-world benchmark datasets: An empirical study of node classification with gnns. In: NeurIPS (2022)
27. McAuley, J., Targett, C., Shi, Q., van den Hengel, A.: Image-based recommendations on styles and substitutes. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 43–52. SIGIR '15, Association for Computing Machinery (2015)
28. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA. Proceedings of Machine Learning Research, vol. 54, pp. 1273–1282. PMLR (2017)
29. Özdayi, M.S., Kantarcioglu, M., Gel, Y.R.: Defending against backdoors in federated learning with robust learning rate. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021. pp. 9268–9276. AAAI Press (2021)
30. Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., McMahan, H.B.: Adaptive federated optimization. arXiv preprint arXiv:2003.00295 (2020)
31. Riesen, K., Bunke, H.: IAM graph database repository for graph based pattern recognition and machine learning. In: Structural, Syntactic, and Statistical Pattern Recognition, Joint IAPR International Workshop, SSPR & SPR 2008, Orlando, USA, December 4-6, 2008. Proceedings. Lecture Notes in Computer Science, vol. 5342, pp. 287–297. Springer (2008)
32. Rong, Y., Xu, T., Huang, J., Huang, W., Cheng, H., Ma, Y., Wang, Y., Derr, T., Wu, L., Ma, T.: Deep graph learning: Foundations, advances and applications. In: KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020. pp. 3555–3556. ACM (2020)

33. Rossi, R., Ahmed, N.: The network data repository with interactive graph analytics and visualization. Proceedings of the AAAI Conference on Artificial Intelligence **29**(1) (Mar 2015)
34. Shchur, O., Mumme, M., Bojchevski, A., Günnemann, S.: Pitfalls of graph neural network evaluation. Relational Representation Learning Workshop, NeurIPS 2018 (2018)
35. Steger, A., Wormald, N.C.: Generating random regular graphs quickly. Combinatorics, Probability and Computing **8**(4), 377–396 (1999)
36. Tolpegin, V., Truex, S., Gursoy, M.E., Liu, L.: Data poisoning attacks against federated learning systems. In: Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25. pp. 480–501. Springer (2020)
37. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings (2018), https://openreview.net/forum?id=rJXMpikCZ
38. Wale, N., Karypis, G.: Comparison of descriptor spaces for chemical compound retrieval and classification. In: Sixth International Conference on Data Mining (ICDM'06). pp. 678–689 (2006). https://doi.org/10.1109/ICDM.2006.39
39. Wang, Z., Kuang, W., Xie, Y., Yao, L., Li, Y., Ding, B., Zhou, J.: Federatedscope-gnn: Towards a unified, comprehensive and efficient package for federated graph learning. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. p. 4110–4120. KDD '22, New York, NY, USA (2022). https://doi.org/10.1145/3534678.3539112, https://doi.org/10.1145/3534678.3539112
40. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world'networks. nature **393**(6684), 440–442 (1998)
41. Xi, Z., Pang, R., Ji, S., Wang, T.: Graph backdoor. In: USENIX Security Symposium. pp. 1523–1540 (2021)
42. Xie, C., Chen, M., Chen, P., Li, B.: CRFL: certifiably robust federated learning against backdoor attacks. In: Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event. Proceedings of Machine Learning Research, vol. 139, pp. 11372–11382. PMLR (2021)
43. Xu, J., Abad, G., Picek, S.: Rethinking the trigger-injecting position in graph backdoor attack. arXiv preprint arXiv:2304.02277 (2023)
44. Xu, J., Wang, R., Koffas, S., Liang, K., Picek, S.: More is better (mostly): On the backdoor attacks in federated graph neural networks. In: Proceedings of the 38th Annual Computer Security Applications Conference. pp. 684–698 (2022)
45. Xu, J., Xue, M., Picek, S.: Explainability-based backdoor attacks against graph neural networks. In: Proceedings of the 3rd ACM Workshop on Wireless Security and Machine Learning. pp. 31–36 (2021)
46. Xu, R., Baracaldo, N., Zhou, Y., Anwar, A., Kadhe, S., Ludwig, H.: Detrust-FL: Privacy-preserving federated learning in decentralized trust setting. In: IEEE 15th International Conference on Cloud Computing, CLOUD 2022, Barcelona, Spain, July 10-16, 2022. pp. 417–426. IEEE (2022)
47. Yang, S., Doan, B.G., Montague, P., De Vel, O., Abraham, T., Camtepe, S., Ranasinghe, D.C., Kanhere, S.S.: Transferable graph backdoor attack. In: Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses. pp. 321–332 (2022)

48. Yang, Z., Cohen, W.W., Salakhutdinov, R.: Revisiting semi-supervised learning with graph embeddings. In: Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016. JMLR Workshop and Conference Proceedings, vol. 48, pp. 40–48. JMLR.org (2016)
49. Zhang, K., Yang, C., Li, X., Sun, L., Yiu, S.M.: Subgraph federated learning with missing neighbor generation. Advances in Neural Information Processing Systems **34**, 6671–6682 (2021)
50. Zhang, Z., Jia, J., Wang, B., Gong, N.Z.: Backdoor attacks to graph neural networks. In: Proceedings of the 26th ACM Symposium on Access Control Models and Technologies. pp. 15–26 (2021)
51. Zheng, H., Xiong, H., Chen, J., Ma, H., Huang, G.: Motif-backdoor: Rethinking the backdoor attack on graph neural networks via motifs. IEEE Transactions on Computational Social Systems (2023)