Backdoor Attack on Vertical Federated Graph Neural Network Learning

Jirui Yang[1] , **Peng Chen**[2*] , **Zhihui Lu**[1*] , **Jianping Zeng**[1] , **Qiang Duan**[3] , **Xin Du**[4] , **Ruijun Deng**[1]
[1]Fudan University, China
[2]Nanjing University of Information Science and Technology, China
[3]Pennsylvania State University, USA
[4]Zhejiang University, China
yangjr23@m.fudan.edu.cn, 003913@nuist.edu.cn, lzh@fudan.edu.cn, zjp@fudan.edu.cn,
qduan@psu.edu, xindu@zju.edu.cn, rjdeng22@m.fudan.edu.cn

Abstract

Federated Graph Neural Network (FedGNN) integrate federated learning (FL) with graph neural networks (GNNs) to enable privacy-preserving training on distributed graph data. Vertical Federated Graph Neural Network (VFGNN), a key branch of FedGNN, handles scenarios where data features and labels are distributed among participants. Despite the robust privacy-preserving design of VFGNN, we have found that it still faces the risk of backdoor attacks, even in situations where labels are inaccessible. This paper proposes BVG, a novel backdoor attack method that leverages multi-hop triggers and backdoor retention, requiring only four target-class nodes to execute effective attacks. Experimental results demonstrate that BVG achieves nearly 100% attack success rates across three commonly used datasets and three GNN models, with minimal impact on the main task accuracy. We also evaluated various defense methods, and the BVG method maintained high attack effectiveness even under existing defenses. This finding highlights the need for advanced defense mechanisms to counter sophisticated backdoor attacks in practical VFGNN applications.

1 Introduction

Graph Neural Networks (GNNs), with their powerful capability to process graph-structured data, have demonstrated significant value in cross-domain applications such as bioinformatics, chemical analysis, medical diagnosis, and financial risk management [Wu *et al.*, 2020]. Specifically, in financial fraud detection scenarios, the use of transaction relationship graphs to identify potential fraudulent activities highlights the unique advantages of GNNs [Cheng *et al.*, 2023]. However, in real-world settings, graph data is often distributed across different stakeholders, and due to data privacy regulations and the need to protect commercial secrets, traditional centralized training paradigms face severe challenges [Liu *et al.*, 2024].

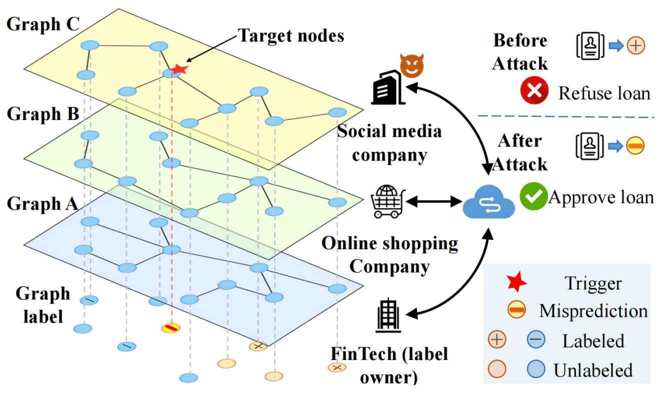

Figure 1: An example of VFGNN backdoor attack

Federated Graph Neural Networks (FedGNN), as a combination of Federated Learning (FL) and GNNs, provide an innovative solution to this challenge [He *et al.*, 2021]. Among them, Vertical Federated Graph Neural Networks (VFGNN) demonstrate unique value in cross-organizational collaboration scenarios due to their ability to handle heterogeneous distributions of features and labels [Mai and Pang, 2023]. For instance, as shown in Figure 1, in a credit evaluation system, financial institutions hold user transaction records and credit labels, social media platforms possess social graphs, and e-commerce platforms provide consumer behavior features. This collaborative multi-party data setting is a typical application scenario for VFGNN.

Although VFGNN effectively protects privacy in distributed scenarios, it also faces significant security challenges, with backdoor attacks being a major potential threat. Attackers can implant concealed trigger patterns to induce specific inputs to produce predefined malicious outputs while maintaining the model's normal predictive performance, posing severe risks to the practical application of VFGNN. For example, in the credit evaluation scenario illustrated in Figure 1, malicious participants could use backdoor attacks to enable high-risk users to obtain credit approvals, leading to serious financial security issues. However, research on backdoor attacks targeting VFGNN remains nearly nonexistent. This is primarily due to two major challenges faced by traditional backdoor attacks in VFGNN scenarios: first, the label

*Corresponding author

isolation mechanism among participants restricts traditional attack paths based on label manipulation; second, the complex topology of graph data requires backdoor triggers to not only maintain structural consistency but also satisfy concealment requirements.

To address the above challenges, this paper proposes a novel graph-structured backdoor attack method based on multi-hop adjacency relationships, termed BVG. This method innovatively designs a backdoor injection mechanism tailored for VFGNNs. First, it employs a multi-hop adjacency trigger generation algorithm to covertly propagate target-class features within the graph structure. Second, a backdoor retention strategy is adopted to enhance the attack's stability. Experimental results demonstrate that BVG requires knowledge of only four target-class nodes to achieve an attack success rate of over 99%, with minimal impact on the performance of the main task. The main contributions of this paper are as follows:

- **Reveal backdoor attack risks in VFGNNs**: A systematic analysis of security threats within the VFGNN framework highlights unique risk characteristics distinct from traditional scenarios.
- **Design multi-hop graph structural triggers**: A bi-level optimization strategy generates backdoor triggers for graph propagation, enabling flexible and covert backdoor injection without disrupting the graph structure.
- **Develop a backdoor retention strategy**: This effectively addresses instability issues in backdoor injection during federated training, improving attack persistence.
- **Validate effectiveness through experiments**: Extensive experiments on widely-used benchmark datasets demonstrate that BVG is efficient and stable, maintaining high attack success rates even under various backdoor defense mechanisms.

2 Related Work

In the fields of VFL and GNN, researchers are investigating backdoor attacks specifically tailored to each domain.

2.1 Backdoor Attacks on VFL

Federated Learning is especially vulnerable to backdoor attacks due to its distributed nature [Zhang *et al.*, 2024]. However, the vertically split model of VFL restricts the attacker's access to the training labels, let alone modify them.

There are two notable directions in attempting to solve this issue. The first is to conduct label inference attacks [Fu *et al.*, 2022] to get some labels in advance. For example, VILLAIN [Bai *et al.*, 2023] leverages label inference to pinpoint samples of the target label and then poisons these samples to inject the backdoor. BadVFL [Naseri *et al.*, 2024] and LR-BA [Gu and Bai, 2023] assume attackers can acquire labels of a certain number of samples from each category with label inference. Another direction is to make the knowledge of a small number of labeled samples from the target class a prerequisite [Chen *et al.*, 2024; Chen *et al.*, 2023a; He *et al.*, 2023]. These methods try to use as few as possible (e.g., 500 for [Chen *et al.*, 2024] and 0.1% for [Chen *et al.*, 2023a]) the labeled target-class samples to establish links between backdoor triggers and target labels. Normally, in VFL, the backdoor attack can only be conducted in a clean-label manner [Zhao *et al.*, 2020]. Therefore, works like TPGD [Liu *et al.*, 2022] assuming the label modification capability of the active party are impractical.

2.2 Backdoor Attacks on GNN

The unique challenge to graph-oriented backdoor attacks lies in the inherently unstructured and discrete nature of graph data [Xi *et al.*, 2021]. For structured and continuous data like images, attackers can directly stamp a trigger pattern (e.g., a black square on the bottom right of the image) s onto a benign image x to achieve a poisoned sample $x_p = x + s$ [Li *et al.*, 2022a]. However, the backdoor attacks against GNN involve design triggers within a large spectrum, including topological structures and descriptive (nodes and edges) features [Xi *et al.*, 2021]. The effectiveness of backdoor attacks largely depends on designing triggers tailored to the specific attributes of the target task. These triggers mainly fall into three categories: malicious subgraph structures, graph structure perturbations, and node attribute manipulation.

Malicious subgraph structures involve adding malicious subgraphs to nodes [Dai *et al.*, 2023], edge endpoints [Zheng *et al.*, 2023], or specific positions in the graph [Zhang *et al.*, 2021; Xi *et al.*, 2021], causing nodes, edges, or graphs to be misclassified. Graph structure perturbation involves injecting special node connections into the training set adversarially, allowing backdoor attacks to be executed by merely modifying the graph's topology during the attack [Yang *et al.*, 2022]. Node attribute manipulation involves carefully designing special node features and adding them to the target class nodes in the training set, followed by adaptive optimization of the graph structure to train a GNN model with backdoors [Xing *et al.*, 2023; Chen *et al.*, 2023b]. Although these methods are viable in centralized GNN scenarios, they all rely on full access to the training set, which is nearly impossible in VFGNN.

The aforementioned backdoor techniques, whether tailored for GNN or VFL, are not applicable to VFGNN due to unique threat models. Existing methods such as CBA, DBA, and Bkd-FedGNN focus on HFGNN and do not address the needs of VFGNN [Xu *et al.*, 2022; Liu *et al.*, 2023]. VFL backdoor attack methods lack consideration for the data structure of graph data, while GNN backdoor attacks face difficulties in handling vertically partitioned data. To address these issues, this paper proposes the BVG algorithm for VFGNN, leveraging multi-hop triggers and a backdoor retention strategy to achieve efficient backdoor attacks.

3 Proposed Approach

In this section, we formalize the backdoor attack problem in VFGNN and provide a detailed threat model. Next, we present our attack method, with a sketch of the entire approach shown in Figure 2.

3.1 Problem Formulation

Graph data is represented as $G = (V, E, X)$, where $V = \{v_1, \ldots, v_n\}$ is a set of N nodes, $E \subseteq V \times V$ is a set of

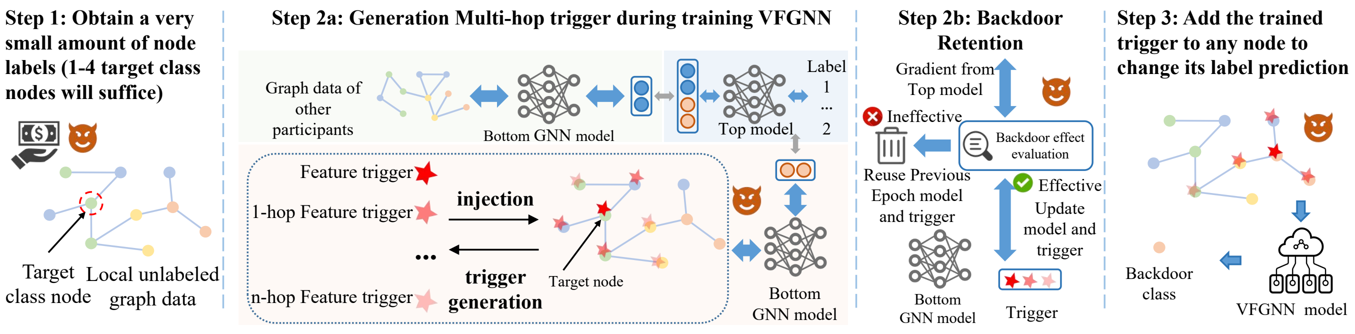
Step 1: Obtain a very small amount of node labels (1-4 target class nodes will suffice)
Target class node
Local unlabeled graph data
Step 2a: Generation Multi-hop trigger during training VFGNN
Graph data of other participants
Bottom GNN model
Top model
Label
1
...
2
Feature trigger
1-hop Feature trigger
n-hop Feature trigger
injection
trigger generation
Target node
Bottom GNN model
Step 2b: Backdoor Retention
Gradient from Top model
Ineffective
Backdoor effect evaluation
Reuse Previous Epoch model and trigger
Effective
Update model and trigger
Bottom GNN model
Trigger
Step 3: Add the trained trigger to any node to change its label prediction
Backdoor class
VFGNN model

Figure 2: A sketch of our proposed backdoor attack for VFGNN.

edges, and $X = \{x_1, \ldots, x_N\}$ is a set of node attributes, with x_i being the attribute of node v_i. The adjacency matrix of graph G is denoted as $A \in \mathbb{R}^{N\times N}$, where $A_{ij} = 1$ if nodes v_i and v_j are connected; otherwise, $A_{ij} = 0$.

We focus on the node classification problem, which is common in real-world applications like social networks. In inductive node classification tasks, only a subset of nodes V_L in the training graph have labels $Y_L = \{y_1, \ldots, y_{N_L}\}$. The test nodes V_T are disjoint from the training nodes.

In VFGNN, K ($K >= 2$) participants collaboratively train a model using their private data. Each participant k has its own local graph $G^k = (V, E^k, X^k)$, which comprises the entire node set V, a subset of edges E^k, and a subset of feature data X^k [Chen *et al.*, 2020]. From a global perspective, all participants collectively own a global graph $G^{gl} = (V, E^{gl}, X^{gl})$, where $E^{gl} = E^1 \cup \ldots \cup E^K$, and for any node attribute in X, $x_i^{gl} = x_i^1 || \ldots || x_i^K$, where $||$ is the concatenation operator. One of the K participants is an active party that knows the corresponding labels of the labeled node set $V_L \subseteq V$, and all other participants are negative parties that have no access to the label information.

Each participant employs a local GNN model f_k parameterized by θ_k to compute the local output $H_i^k = f_k(\mathcal{G}_i^k; \theta_k)$, where $\mathcal{G}_i^k$ denotes the computation graph of node v_i. In addition to training its own local model, the active party also trains a top model G parameterized by θ_{top}, which aggregates the outputs from all local models to minimize the loss function $\mathcal{L}$. Therefore, the VFGNN model training can be formulated as

$$\arg\min_{\Theta} \sum_{v_l \in V_L} \mathcal{L}(G(H_l^1, \cdots, H_l^K), y_l) \tag{1}$$

where $\Theta = \{\theta_1, \cdots, \theta_K; \theta_{top}\}$ represents the parameters of the overall VFGNN model.

As an adversary in a backdoor attack, the main objectives are threefold: first, to ensure that the federated learning task can be completed successfully; second, to establish a mapping between the trigger and the backdoor target class; and third, to ensure that the backdoor injection remains undetected by the active party. These three objectives must be achieved simultaneously during the VFGNN training process. Specifically, the attacker needs to inject the backdoor cleverly, ensuring that the training process is not disrupted and that no suspicious behavior is exposed. After the VFGNN model is deployed, the attacker can introduce a local trigger to deliberately misclassify the corresponding sample as the target class, thereby completing the backdoor attack. The backdoor attack objective in VFGNN is

$$\min_{\Theta} \underbrace{\sum_{v_l \in V_L} \mathcal{L}\left(F(\mathcal{G}_l; \Theta), y_l\right)}_{\text{Main Task}} + \underbrace{\sum_{v_i \in V} \mathcal{L}\left(F(a(\mathcal{G}_i, \delta); \Theta), \tau\right)}_{\text{Backdoor Task}} + \underbrace{\sum_{v_i \in V} ||\tilde{H}_i - H_i||^2}_{\text{Undetected Task}} \tag{2}$$

where F refers to the VFGNN model, which encompasses both the top model and the bottom models of all participating parties. δ represents the backdoor trigger, a($\cdot$) denotes the operation of trigger attachment, and τ is the backdoor target class. $\tilde{H}$ denotes the output of the bottom GNN model after the trigger injection, while H refers to the output without trigger injection.

3.2 Threat Model

We assume the adversary is a passive party, not the active party, because the active party can modify labels and thus easily perform backdoor attacks. All other participants are considered trustworthy.

Adversary's Capacity. The adversary adheres to the VFL protocol, transmitting local features and receiving gradients without manipulating other participants' information.

Adversary's Objective. In VFGNN multi-classification tasks, the adversary's objective is to inject a backdoor into the model during the training phase without being detected. When the poisoned model is deployed for applications, it will misclassify any local data with the backdoor trigger as the designated target class but maintains classification capacity for all other clean data.

Adversary's Knowledge. In backdoor attacks, the adversary requires only a very few training samples labeled target class. Although this requirement deviates from the original VFGNN setting, it is feasible in practical VFGNN applications because it is possible for an adversary to acquire a small number of target class samples through various means, such as direct purchasing [Fu *et al.*, 2022; Gu and Bai, 2023]. Our experiment results indicate that acquiring only one to four target class nodes is sufficient for a successful backdoor attack.

Apart from these limited target samples, the adversary has no information about the models and data of other parties.

3.3 Multi-hop Trigger Generation

Due to the special nature of the VFGNN architecture, all participants are aware of all nodes in the dataset, but possess different attributes of the nodes. Therefore, the adversary cannot generate a subgraph structure as a trigger like common GNN backdoor attacks [Dai *et al.*, 2023; Wang *et al.*, 2024]. Here, the trigger is added to the node attributes. To prepare a trigger, we propose a multi-hop trigger generation method without introducing new nodes. The trigger generation and training are conducted synchronously with the VFGNN model. Additionally, to enhance optimization efficiency, we introduce a hyperparameter ϵ as a trigger threshold. By appropriately setting this threshold, we ensure the stealthiness of the trigger while simplifying the optimization task from a triple optimization to a dual optimization. We use the PGD method [Madry *et al.*, 2018; Huang *et al.*, 2021] to generate the trigger. The optimization formula for the trigger is as follows:

$$\delta_{t+1} = \Pi_\epsilon \left(\delta_t - \alpha \cdot \operatorname{sgn}\left(\nabla_\delta \mathcal{L}\left(F\left(a(\mathcal{G}_p, \delta_t); \Theta^*\right), \tau\right)\right)\right), \tag{3}$$

where t is the step index, $\mathcal{G}_p$ represents the computation graph of the node v_p to inject the trigger, where $v_p \in V_p$. $\nabla_\delta \mathcal{L}\left(F\left(a(\mathcal{G}_p, \delta_t); \Theta^*\right), \tau\right)$ denotes the gradient of the loss function for the backdoor target class with respect to the trigger, α is the step size, Π_ϵ keeps δ within an ϵ-ball at each step, $F(\Theta^*)$ refers to the pre-trained VFGNN model. $\operatorname{sgn}(\cdot)$ denotes the sign function.

The multi-hop trigger δ affects the target node and its neighbors, $\delta = \{\delta^0, \delta^1, \cdots, \delta^M\}$. Here, δ^m is the trigger added to the attributes of the m-hop neighbors of the target node v_p. Therefore, the operation of adding triggers to $\mathcal{G}_p$ can be expressed as

$$a(\mathcal{G}_p, \delta) = (\mathbf{x}_p + \delta^0, \mathbf{X}_{1-hop} + \delta^1, \cdots, \mathbf{X}_{M-hop} + \delta^M), \tag{4}$$

where $\mathbf{x}_p$ is the attribute of node v_p, $\mathbf{X}_{m-hop}$ are the attributes of the m-hop neighbors, and δ^0, δ^m are the triggers added to these attributes, respectively.

According to (3), the generation of the trigger relies on the gradients of the target class nodes τ. The generation of the trigger and the training of the VFGNN model are accomplished together, which can be expressed through the following bi-level optimization:

$$\begin{aligned}
&\min_\delta \sum_{v_i \in \mathcal{V}_P} \mathcal{L}(F(a(\mathcal{G}_i, \delta); \Theta^*), \tau) \\
&\text{s.t.} \quad \Theta^* = \arg\min_\Theta \sum_{v_i \in \mathcal{V}_L - \mathcal{V}_P} \mathcal{L}(F(G_i; \Theta), y_i) \\
&\qquad\qquad + \sum_{v_i \in \mathcal{V}_P} \mathcal{L}(F(a(G_i, \delta); \Theta), \tau),
\end{aligned} \tag{5}$$

where the trigger δ is trained according to all known target class nodes $\mathcal{V}_P$ available to the adversary, $\mathcal{V}_P \in \mathcal{V}_L$. Essentially, this trigger serves as a universal trigger for the target class τ [Shafahi *et al.*, 2020; Zeng *et al.*, 2022].

Algorithm 1 Multi-hop Trigger Generation

```
Require: Training dataset $\mathcal{G}$; number of MTG epochs $T$; target class nodes $\mathcal{V}_P$ with target label $\tau$
Ensure: trigger $\delta$, model parameters $\Theta$
 1: Initialize: $\delta \leftarrow 0$.
 2: while not reached $T$ do
 3:     for $v_i$ in $\mathcal{V}_L$ parallel do
 4:         Adversary $A$: updates $\{\mathcal{G}_i^A = a(\mathcal{G}_i^A, \delta)\}_{v_i \in \mathcal{V}_P}$.
 5:         Passive party:
 6:         for each party $k = 1, 2, \ldots, K$ parallel do
 7:             $k$ computes embedded features $H_i^k$ using its bottom model $f_k$.
 8:         end for
 9:         Active party:
10:         computes Eq. (1), then updates $\theta_{Top}$ using $\frac{\partial \mathcal{L}}{\partial \theta_{Top}}$.
11:         sends $\frac{\partial \mathcal{L}}{\partial H_i}$ to all parties.
12:         Adversary $A$:
13:         computes $\nabla_\delta \mathcal{L}$ with $\{\frac{\partial \mathcal{L}}{\partial H_i} \frac{\partial H_i}{\partial \delta}\}_{v_i \in \mathcal{V}_P}$
14:         updates $\delta$ with Eq. (3)
15:         Passive party:
16:         for each party $k = 1, 2, \ldots, K$ parallel do
17:             $k$ computes $\nabla_{\theta_k} \mathcal{L} = \frac{\partial \mathcal{L}}{\partial H_i} \frac{\partial H_i^k}{\partial \theta_k}$.
18:             $k$ updates model parameters $\theta_k$.
19:         end for
20:     end for
21: end while
```

Algorithm 1 outlines the procedure of the proposed attack. In each iteration of VFGNN model training, the adversary A injects the trigger δ into the local computation graph of known target class samples $\mathcal{V}_P$ according to (4) (line 4). Subsequently, each participant k computes the node embeddings through the bottom model f_k (lines 5-8), where H_i^k represents the embedded features of the i-th data from the k-th participant. Upon receiving these embedded features, the active party computes the gradients of the loss function with respect to the top model and the embedded features of each participant according to (1). The top model is then updated, and the gradients of the loss with respect to the embedded features $\frac{\partial \mathcal{L}}{\partial H_i}$ are transmitted to each participant (lines 9-11), where H_i denotes the aggregation of the embedded features from all participants for the i-th data. After receiving the gradients, the adversary computes $\nabla_\delta \mathcal{L}$ and updates the trigger using (3). The current model parameters serve as the pre-trained model $F(\Theta^*)$ (lines 12-14). Finally, each participant updates its bottom model based on the gradients sent by the active party (lines 15-19).

3.4 Backdoor Retention

When injecting backdoors, using only a small number of samples may lead to unstable effects. For instance, a backdoor may perform well in one epoch but degrade suddenly in the next. To ensure the stability of backdoors, we propose a method called Backdoor Retention (BR).

The BM method consists of two key steps. First, the attacker evaluates the impact of the backdoor. Since the attacker cannot directly access the output of the VFGNN model during training, they rely on an assumption: if the backdoor is successfully injected, the outputs H of nodes triggered by the injection should exhibit high similarity. Based on this assumption, the backdoor effectiveness E is approximated using the following formula:

$$E = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{H_i \cdot H_j}{\|H_i\|_2 \|H_j\|_2}, \tag{6}$$

where n is the number of nodes involved in the trigger injection, and H_i represents the output of node v_i in the attacker's bottom model.

After estimating E, the attacker performs the following actions in each epoch: if E exceeds a predefined threshold, the attacker updates the bottom model and trigger; otherwise, the model and trigger from the previous epoch are retained. This approach ensures the stability and effectiveness of the backdoor throughout the training process.

4 Experiments

4.1 Experiments Settings

Local GNN Models

To demonstrate the effectiveness of BVG in VFGNN settings with different local GNN structures, we use three common GNN models as local participants:

Graph Convolutional Network (GCN) [Kipf and Welling, 2016]: GCN captures local features by propagating and aggregating information between nodes and their neighbors. Each local GNN model in VFGNN uses a two-layer GCN.

Graph Attention Network (GAT) [Veličković *et al.*, 2017]: GAT uses an attention mechanism to assign adaptive weights to neighbor nodes, enhancing the model's ability to focus on important nodes.

Simple Graph Convolution (SGC) [Wu *et al.*, 2019]: SGC simplifies GCN by removing nonlinear activations and reducing layers, making it more efficient while retaining essential graph structural information.

To ensure a fair comparison with previous studies [Chen *et al.*, 2022], we use a two-layer GNN model for each participant's local GNN to extract local node embeddings, with the dimension set to 16. The number of hidden units is fixed at 32. For GCN and GAT, the activation function is ReLU. VFGNN is trained using Adam, with a learning rate of 0.01.

Datasets

This paper uses three widely adopted public datasets to evaluate the performance of BVG, including Cora [McCallum *et al.*, 2000], Cora_ml [McCallum *et al.*, 2000], and Pubmed [Sen *et al.*, 2008]. The basic dataset statistics are summarized in Table 1.

Each participant in the VFGNN framework has access to all the nodes in the datasets, but the node features are equally split among the participants. We randomly split the edges of the graphs into equal parts, one for each participant, without overlapped edges between any two participants. We assume the adversary knows only four target class nodes (i.e., $|\mathcal{V}_p| = 4$), which are randomly selected from the training set.

Datasets	Nodes	Edges	Features	Classes
Cora	2708	5429	1433	7
Cora_ml	2810	7981	2879	7
Pubmed	19717	44325	500	3

Table 1: Basic information of the three datasets

Performance Metrics

To evaluate the performance of the BVG method, we use three common metrics: Attack Success Rate (ASR), Main Task Accuracy (MTA) [Li *et al.*, 2022b], and Mean Squared Error (MSE). Among them, ASR and MTA are the most commonly used evaluation metrics for backdoor attack tasks. ASR measures the proportion of samples in the backdoor test set that are predicted as the target class by the poisoned model. On the other hand, MTA assesses the accuracy of the clean test set on the poisoned model. MSE is used to measure the difference in the output of the bottom GNN model before and after trigger injection. A smaller MSE indicates that the injected trigger is less detectable by the active party, implying better stealth.

Comparison Benchmarks

Due to the lack of research on backdoor attacks in VFGNN, we selected three attack methods from other scenarios for comparison, adapting them to fit VFGNN's tasks and structure. Additionally, we incorporated common GNN backdoor trigger schemes for further comparison. The methods are summarized below:

GF [Chen *et al.*, 2022] Originally designed for adversarial tasks in VFGNN, this method was adapted to minimize the loss of the backdoor class. It involves querying participant embeddings, training a proxy model, and generating an adversarial graph on the proxy model.

TECB [Chen *et al.*, 2023a] A VFL backdoor attack method for image data, adapted here for graph tasks by aligning target node attributes with trigger embeddings. It operates in two stages: training triggers with limited labels and aligning target gradients using the trigger.

VILLAIN [Bai *et al.*, 2023] Designed for VFL backdoor attacks, this method injects triggers into the bottom model's output H, supporting clean-label attacks. It combines label inference using limited labeled samples and data poisoning with trigger masking and randomness for enhanced robustness.

Edge Triggering (ET) This method connects attack nodes to target class nodes via edges, creating a trigger relationship that exploits the graph's topology to influence target node classification.

Node Feature Triggering (NFT) Backdoor attacks are achieved by injecting trigger patterns into nodes with known labels, altering their features to misclassify attack nodes as the target class under specific triggers.

Node Feature Replacement (NFR) Attackers replace the features of attack nodes with those of target class nodes, making their features similar to the target class and inducing misclassification.

Bottom Model	Datasets	MTA						
		GF*	TECB	VILLAIN	ET	NFT	NFR	BVG
GCN	Cora	48.60±2.87	62.72±4.67	65.38±2.99	65.28±2.37	**66.12 ± 2.13**	50.34±16.06	65.26±1.50
	Cora_ml	64.20±7.63	**77.06 ± 0.91**	73.60±3.49	75.50±2.78	75.46±2.25	56.90±21.80	76.50±2.63
	Pubmed	44.00±10.20	64.90±4.15	69.26±2.86	59.80±15.40	68.44±2.58	32.36±13.57	**71.34 ± 3.54**
GAT	Cora	46.60±3.93	40.06±4.10	63.44±2.01	61.70±6.37	**64.30 ± 3.04**	41.02±16.66	63.72±2.20
	Cora_ml	61.00±3.29	57.78±4.86	73.14±1.75	76.12±2.45	**76.50 ± 2.12**	59.94±23.23	75.58±2.77
	Pubmed	70.80±3.54	44.98±17.92	72.40±0.89	63.66±11.79	72.88±1.49	35.64±14.71	**73.28 ± 1.73**
SGC	Cora	49.00±3.03	36.68±29.97	62.56±2.12	55.04±16.96	62.08±2.63	50.12±15.24	**65.30 ± 2.08**
	Cora_ml	67.40±4.03	74.76±2.62	74.08±2.47	**75.34 ± 1.62**	75.08±3.59	56.80±21.77	74.94±2.39
	Pubmed	41.82±12.33	54.04±27.02	68.90±4.07	61.46±14.75	69.80±1.94	39.02±11.83	**70.64 ± 2.95**
Bottom Model	**Datasets**	**ASR**						
		GF	**TECB**	**VILLAIN**	**ET**	**NFT**	**NFR**	**BVG**
GCN	Cora	51.40±5.61	16.40±10.91	99.00±0.64	48.80±13.62	91.54±4.13	36.48±22.01	**99.86 ± 0.28**
	Cora_ml	26.80±5.15	34.60±7.74	97.34±3.94	49.92±7.82	74.78±22.86	38.12±31.24	**99.98 ± 0.04**
	Pubmed	24.00±8.00	81.40±13.83	94.96±4.44	78.96±8.91	88.02±6.41	81.86±20.90	**100.00 ± 0.00**
GAT	Cora	15.80±6.31	56.80±35.54	94.52±5.64	52.00±14.36	66.14±28.01	43.34±30.03	**99.12 ± 1.10**
	Cora_ml	23.40±2.73	45.14±31.03	90.48±6.33	44.80±17.86	95.28±4.82	34.62±32.96	**99.92 ± 0.16**
	Pubmed	30.20±2.04	89.60±20.80	86.98±10.71	78.42±9.90	98.64±2.23	80.62±20.43	**100.00 ± 0.00**
SGC	Cora	51.00±4.20	12.24±13.28	99.34±0.65	54.40±22.44	87.60±13.96	32.46±19.15	**99.76 ± 0.39**
	Cora_ml	27.80±3.97	15.18±9.14	98.72±1.42	60.22±12.20	88.66±7.56	37.78±31.25	**99.98 ± 0.04**
	Pubmed	32.73±10.91	59.78±48.81	99.10±0.94	82.56±5.47	91.30±6.20	71.84±15.77	**99.90 ± 0.20**

Table 2: Performance comparison with other methods on three datasets and three bottom models (Standard deviation included, best results highlighted in bold. GF* is an adversarial attack method, where a lower MTA is desirable.)

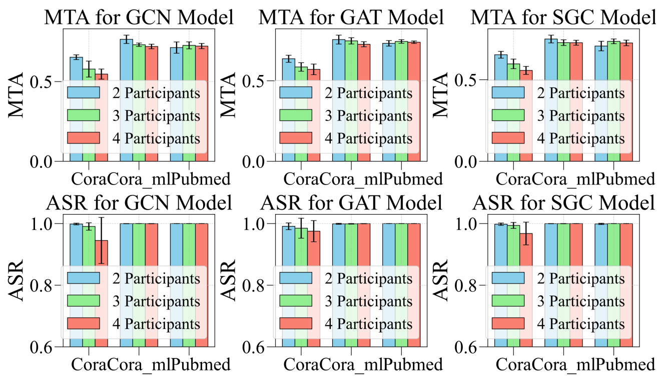

Figure 3: Performance of BVG under multi-party settings (Standard deviation included.)

4.2 Experiment Results Analysis

Backdoor Task Evaluation

We conducted experiments in a VFGNN framework with two participants, as shown in Table 2. BVG and NFT perform best on the main task, with slight variations across datasets and models. For the backdoor task, BVG consistently achieves the highest attack success rate across all tested models and datasets, demonstrating its effectiveness with minimal impact on main task performance.

We also evaluated the proposed attack method's performance in multi-party VFGNN scenarios where multiple participants are involved. Unlike the two-party setting, multi-party scenarios become more complex due to the increased number of participants and their interactions. Our evaluation focuses on how the presence of multiple participants affects the ASR and the MTA of BVG.

We conducted experiments with two, three, and four participants to evaluate the scalability and effectiveness of the proposed backdoor attack method in multi-party VFGNN scenarios. Each participant was assigned a subset of the graph's edges and node features, with one active party holding the labels for the labeled node set V_L. The passive participants, including the adversary, only had partial information about the graph. The experimental results are shown in Figure 3.

The results indicate that the main task accuracy slightly decreases as the number of participants increases. This trend could be attributed to the increased complexity and noise introduced by multiple participants. The backdoor attack success rate remains high across all multi-party settings but slightly decreases as the number of participants increases. The obtained results verify that our attack method is effective even in more complex multi-party environments. The slight decrease in success rate may be due to the dilution of the adversary's influence with more participants.

Backdoor Stealthiness Evaluation

In evaluating the stealthiness of backdoors in VFGNN, we measure the difference in the intermediate layer features of the same node before and after the backdoor injection. If the intermediate layer features show little change after the backdoor injection, we consider the backdoor difficult for the attacker to detect. For comparison, we selected several methods from Table 2 that perform relatively well on the backdoor task, with a particular focus on the performance of the BVG method under multi-hop triggers. Figure 4 shows the backdoor task performance and the average difference in intermediate layer features for five methods across three datasets.

From Figure 4, it is evident that the increase in MSE correlates positively with the increase in ASR. However, despite BVG achieving a significantly higher ASR than VILLAIN,

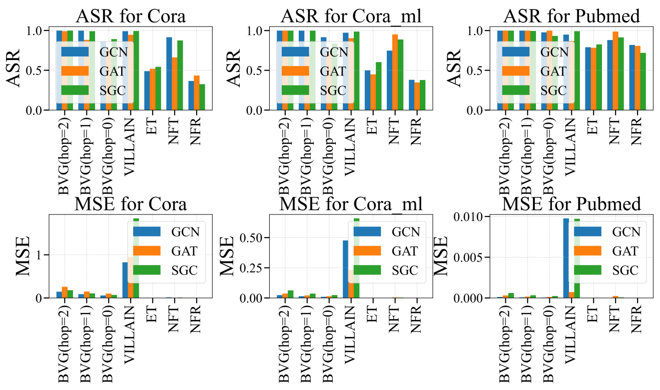

Figure 4: Comparison of backdoor task performance and intermediate layer feature differences across methods

its MSE is much lower than VILLAIN, indicating that BVG exhibits better stealthiness. For the other methods, BVG has slightly worse stealthiness but significantly better ASR performance. More importantly, the ASR performance of ET, NFT, and NFR has almost plateaued.

Attack Efficacy under Defense

To evaluate the robustness of the BVG method, we explored potential backdoor defense strategies within the VFGNN framework. We considered two main types of defense: the first type is GNN backdoor defense methods, such as Prune and Prune+LD [Dai *et al.*, 2023]. These methods defend against backdoor attacks by pruning the edges between nodes with low similarity. Pruning these edges can disrupt the attacker's trigger structure and connections. However, VFGNN's architecture prevents defenders from pruning the adversary's data, making these methods unsuitable.

The second type of defense methods is specific to the VFL framework and can be divided into two categories: label inference defense and backdoor attack defense. Label inference defense operates on the assumption that backdoor attacks often require access to a large amount of label information. To prevent label leakage, defense methods perturb or compress gradients to minimize information leakage. Two representative methods are DP-SGD and gradient compression (GC). DP-SGD provides differential privacy protection by adding noise to gradients [Fu *et al.*, 2022], where the larger the noise amplitude, the smaller the privacy leakage, and vice versa. The GC method [Fu *et al.*, 2022; Kairouz *et al.*, 2021] sends only a subset of gradients with the largest absolute values to participants, thereby reducing privacy leakage. For backdoor attack defense, there are mainly two approaches: one is model reconstruction, which aims to purify the top model of the active party to remove the backdoor, with methods such as model pruning [Liu *et al.*, 2018] and adversarial neuron pruning (ANP) [Wu and Wang, 2021]. The other approach is to introduce interference into intermediate-layer features [Li *et al.*, 2021], such as embedding Gaussian noise into the intermediate features provided to the passive party to disrupt potential backdoor triggers.

In the VFGNN framework, we implemented several VFL backdoor defense methods, each with five parameter settings,

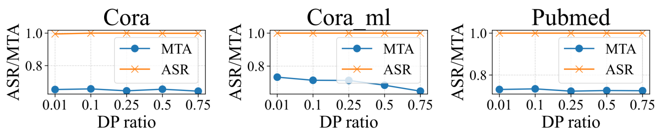

Figure 5: The performance of BVG under DP-SGD defense

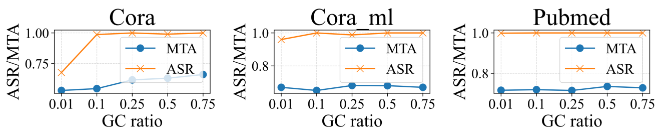

Figure 6: The performance of BVG under GC defense

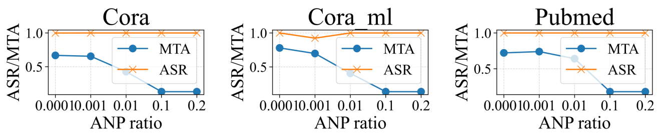

Figure 7: The performance of BVG under ANP defense

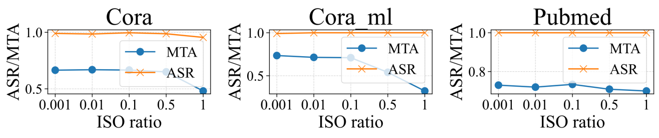

Figure 8: The performance of BVG under ISO defense

as shown in the figures. We present the experimental results when the base model is GCN, and similar conclusions are observed for GAT and SGC models.For label inference defense, as shown in Figures 5 and 6, our method is minimally affected, as the BVG method itself does not rely on label inference. For backdoor attack defense, the results of the ANP method are shown in Figure 7. The ANP method primarily affects model accuracy without effectively suppressing the backdoor. For the defense method involving interference with intermediate-layer features, the experimental results are shown in Figure 8. As model accuracy decreases, the success rate of backdoor attacks also drops, making it difficult for defenders to reduce ASR while maintaining a high MTA.

5 Conclusion

In this paper, we proposed the BVG method for backdoor attacks to Vertical Federated Graph Neural Network (VFGNN). Utilizing a multi-hop trigger generation approach, the BVG method can perform efficient backdoor attacks with very limited knowledge of target class nodes. Extensive experiments on three datasets with three GNN models demonstrate that BVG achieves a high attack success rate while having a minimal impact on the main task accuracy and is unlikely to be detected by the active party. The evaluation of BVG efficacy under various defense methods highlights the robustness and efficiency of the proposed attack method, underscoring the necessity for advanced defense mechanisms in practical federated learning applications to counter such sophisticated backdoor attacks.

Acknowledgments

This work is supported by the National Key Research and Development Program of China (2022YFC3302300), Yangtze River Delta Science and Technology Innovation Community Joint Research Project (YDZX20233100004031), National Natural Science Foundation of China (No. 61873309, No. 92046024, No. 92146002) and Shanghai Science and Technology Project (No. 22510761000). We thank all contributors to the open-source libraries used in this work. Our implementation is publicly available at https://github.com/yangjr01/VFGNN_backdoor.

References

[Bai *et al.*, 2023] Yijie Bai, Yanjiao Chen, Hanlei Zhang, Wenyuan Xu, Haiqin Weng, and Dou Goodman. {VILLAIN}: Backdoor attacks against vertical split learning. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2743–2760, 2023.

[Chen *et al.*, 2020] Chaochao Chen, Jun Zhou, Longfei Zheng, Huiwen Wu, Lingjuan Lyu, Jia Wu, Bingzhe Wu, Ziqi Liu, Li Wang, and Xiaolin Zheng. Vertically federated graph neural network for privacy-preserving node classification. *arXiv preprint arXiv:2005.11903*, 2020.

[Chen *et al.*, 2022] Jinyin Chen, Guohan Huang, Haibin Zheng, Shanqing Yu, Wenrong Jiang, and Chen Cui. Graph-fraudster: Adversarial attacks on graph neural network-based vertical federated learning. *IEEE Transactions on Computational Social Systems*, 10(2):492–506, 2022.

[Chen *et al.*, 2023a] Peng Chen, Jirui Yang, Junxiong Lin, Zhihui Lu, Qiang Duan, and Hongfeng Chai. A practical clean-label backdoor attack with limited information in vertical federated learning. In *2023 IEEE International Conference on Data Mining (ICDM)*, pages 41–50. IEEE, 2023.

[Chen *et al.*, 2023b] Yang Chen, Zhonglin Ye, Haixing Zhao, and Ying Wang. Feature-based graph backdoor attack in the node classification task. *International Journal of Intelligent Systems*, 2023(1):5418398, 2023.

[Chen *et al.*, 2024] Peng Chen, Xin Du, Zhihui Lu, and Hongfeng Chai. Universal adversarial backdoor attacks to fool vertical federated learning. *Computers & Security*, 137:103601, 2024.

[Cheng *et al.*, 2023] Dawei Cheng, Yujia Ye, Sheng Xiang, Zhenwei Ma, Ying Zhang, and Changjun Jiang. Anti-money laundering by group-aware deep graph learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12444–12457, 2023.

[Dai *et al.*, 2023] Enyan Dai, Minhua Lin, Xiang Zhang, and Suhang Wang. Unnoticeable backdoor attacks on graph neural networks. In *Proceedings of the ACM Web Conference 2023*, pages 2263–2273, 2023.

[Fu *et al.*, 2022] Chong Fu, Xuhong Zhang, Shouling Ji, Jinyin Chen, Jingzheng Wu, Shanqing Guo, Jun Zhou, Alex X Liu, and Ting Wang. Label inference attacks against vertical federated learning. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1397–1414, 2022.

[Gu and Bai, 2023] Yuhao Gu and Yuebin Bai. Lr-ba: Backdoor attack against vertical federated learning using local latent representations. *Computers & Security*, 129:103193, 2023.

[He *et al.*, 2021] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang, Han Xie, Lichao Sun, Lifang He, Liangwei Yang, Philip S Yu, Yu Rong, et al. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145*, 2021.

[He *et al.*, 2023] Ying He, Zhili Shen, Jingyu Hua, Qixuan Dong, Jiacheng Niu, Wei Tong, Xu Huang, Chen Li, and Sheng Zhong. Backdoor attack against split neural network-based vertical federated learning. *IEEE Transactions on Information Forensics and Security*, 2023.

[Huang *et al.*, 2021] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. In *International Conference on Learning Representations*, 2021.

[Kairouz *et al.*, 2021] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Li *et al.*, 2021] Yiming Li, Tongqing Zhai, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor attack in the physical world. *arXiv preprint arXiv:2104.02361*, 2021.

[Li *et al.*, 2022a] Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[Li *et al.*, 2022b] Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[Liu *et al.*, 2018] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International symposium on research in attacks, intrusions, and defenses*, pages 273–294. Springer, 2018.

[Liu *et al.*, 2022] Jing Liu, Chulin Xie, Sanmi Koyejo, and Bo Li. Copur: Certifiably robust collaborative inference via feature purification. *Advances in Neural Information Processing Systems*, 35:26645–26657, 2022.

[Liu *et al.*, 2023] Fan Liu, Siqi Lai, Yansong Ning, and Hao Liu. Bkd-fedgnn: A benchmark for classification backdoor attacks on federated graph neural network. *arXiv preprint arXiv:2306.10351*, 2023.

[Liu *et al.*, 2024] Rui Liu, Pengwei Xing, Zichao Deng, Anran Li, Cuntai Guan, and Han Yu. Federated graph neural networks: Overview, techniques, and challenges. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–17, 2024.

[Madry *et al.*, 2018] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[Mai and Pang, 2023] Peihua Mai and Yan Pang. Vertical federated graph neural network for recommender system. In *International Conference on Machine Learning*, pages 23516–23535. PMLR, 2023.

[McCallum *et al.*, 2000] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163, 2000.

[Naseri *et al.*, 2024] Mohammad Naseri, Yufei Han, and Emiliano De Cristofaro. Badvfl: Backdoor attacks in vertical federated learning. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 2013–2028. IEEE, 2024.

[Sen *et al.*, 2008] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[Shafahi *et al.*, 2020] Ali Shafahi, Mahyar Najibi, Zheng Xu, John Dickerson, Larry S Davis, and Tom Goldstein. Universal adversarial training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5636–5643, 2020.

[Veličković *et al.*, 2017] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[Wang *et al.*, 2024] Huiwei Wang, Tianhua Liu, Ziyu Sheng, and Huaqing Li. Explanatory subgraph attacks against graph neural networks. *Neural Networks*, 172:106097, 2024.

[Wu and Wang, 2021] Dongxian Wu and Yisen Wang. Adversarial neuron pruning purifies backdoored deep models. *Advances in Neural Information Processing Systems*, 34:16913–16925, 2021.

[Wu *et al.*, 2019] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019.

[Wu *et al.*, 2020] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[Xi *et al.*, 2021] Zhaohan Xi, Ren Pang, Shouling Ji, and Ting Wang. Graph backdoor. In *30th USENIX security symposium (USENIX Security 21)*, pages 1523–1540, 2021.

[Xing *et al.*, 2023] Xiaogang Xing, Ming Xu, Yujing Bai, and Dongdong Yang. A clean-label graph backdoor attack method in node classification task. *arXiv preprint arXiv:2401.00163*, 2023.

[Xu *et al.*, 2022] Jing Xu, Rui Wang, Stefanos Koffas, Kaitai Liang, and Stjepan Picek. More is better (mostly): On the backdoor attacks in federated graph neural networks. In *Proceedings of the 38th Annual Computer Security Applications Conference*, pages 684–698, 2022.

[Yang *et al.*, 2022] Shuiqiao Yang, Bao Gia Doan, Paul Montague, Olivier De Vel, Tamas Abraham, Seyit Camtepe, Damith C Ranasinghe, and Salil S Kanhere. Transferable graph backdoor attack. In *Proceedings of the 25th international symposium on research in attacks, intrusions and defenses*, pages 321–332, 2022.

[Zeng *et al.*, 2022] Yi Zeng, Minzhou Pan, Hoang Anh Just, Lingjuan Lyu, Meikang Qiu, and Ruoxi Jia. Narcissus: A practical clean-label backdoor attack with limited information. *arXiv preprint arXiv:2204.05255*, 2022.

[Zhang *et al.*, 2021] Zaixi Zhang, Jinyuan Jia, Binghui Wang, and Neil Zhenqiang Gong. Backdoor attacks to graph neural networks. In *Proceedings of the 26th ACM Symposium on Access Control Models and Technologies*, pages 15–26, 2021.

[Zhang *et al.*, 2024] Hangfan Zhang, Jinyuan Jia, Jinghui Chen, Lu Lin, and Dinghao Wu. A3fl: Adversarially adaptive backdoor attacks to federated learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[Zhao *et al.*, 2020] Shihao Zhao, Xingjun Ma, Xiang Zheng, James Bailey, Jingjing Chen, and Yu-Gang Jiang. Clean-label backdoor attacks on video recognition models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14443–14452, 2020.

[Zheng *et al.*, 2023] Haibin Zheng, Haiyang Xiong, Haonan Ma, Guohan Huang, and Jinyin Chen. Link-backdoor: Backdoor attack on link prediction via node injection. *IEEE Transactions on Computational Social Systems*, 2023.