Hindawi
International Journal of Intelligent Systems
Volume 2023, Article ID 5418398, 13 pages
https://doi.org/10.1155/2023/5418398

Research Article

Feature-Based Graph Backdoor Attack in the Node Classification Task

Yang Chen[1,2], **Zhonglin Ye**[1,2], **Haixing Zhao**[1,2] and **Ying Wang**[3]

[1]*School of Computer Science, Qinghai Normal University, Xining, China*
[2]*The State Key Laboratory of Tibetan Intelligent Information Processing and Application, Qinghai Normal University, Xining, China*
[3]*School of Management, Northwestern Polytechnical University, Xi'an, China*

Correspondence should be addressed to Haixing Zhao; h_x_zhao@126.com

Received 21 October 2022; Revised 18 December 2022; Accepted 30 December 2022; Published 21 February 2023

Academic Editor: Subrata Kumar Sarker

Graph neural networks (GNNs) have shown significant performance in various practical applications due to their strong learning capabilities. Backdoor attacks are a type of attack that can produce hidden attacks on machine learning models. GNNs take backdoor datasets as input to produce an adversary-specified output on poisoned data but perform normally on clean data, which can have grave implications for applications. Backdoor attacks are under-researched in the graph domain, and almost existing graph backdoor attacks focus on the graph-level classification task. To close this gap, we propose a novel graph backdoor attack that uses node features as triggers and does not need knowledge of the GNNs parameters. In the experiments, we find that feature triggers can destroy the feature spaces of the original datasets, resulting in GNNs inability to identify poisoned data and clean data well. An adaptive method is proposed to improve the performance of the backdoor model by adjusting the graph structure. We conducted extensive experiments to validate the effectiveness of our model on three benchmark datasets.

1. Introduction

Graphs can abstractly describe the relationships between objects and have received much attention from researchers [1, 2]. Graph neural networks (GNNs) combine node features and graph structure with learning better representations and have achieved significant performance in many tasks, e.g., graph classification [3], node classification [4], and link prediction [5, 6]. However, recent studies [7–9] have shown that GNNs are vulnerable to adversarial attacks. The most famous graph adversarial attack is Nettack [8], which reduces the accuracy of GNNs by modifying the graph structure or node features through a surrogate model.

Backdoor attacks are a completely different type of attack from adversarial attacks [10]. In backdoor attacks, the adversary first generates a trigger and then embeds the trigger into the data to generate the poisoned data [11]. The backdoor dataset contains the poisoned data, and the clean dataset is composed of clean data without trigger embedding [12, 13]. The GNNs that train the backdoor dataset are called backdoor GNNs, and the clean GNNs are trained on the clean dataset [14]. The output of backdoor GNNs has two characteristics: (1) the output on the clean data is similar to that of clean GNNs; (2) the output on the poisoned data is predefined by the adversary.

Most of the existing works [15, 16] on backdoor attacks are in the area of image and text and few studies on graph backdoor attacks [14, 17–19]. However, GNNs are also vulnerable to backdoor attacks. For example, in the case of spam emails with triggers, they are treated as normal emails in the system detection session. Besides, most preprocessing models often download some relevant data from the Internet. The preprocessing models make the wrong decision if the downloaded data contains poisoned data from backdoor attacks.

Recent works on GNNs backdoor attacks has focused on the graph-level classification task. Zhang et al. [18] proposed a backdoor attack (subgraph backdoor) using a fixed

subgraph as a trigger, where they will output the target label if the same subgraph appears in the testing dataset. Xi et al. [17] proposed a dynamically changing subgraph trigger (GTA), where GTA would tailor different triggers to poison the data based on the features of each graph. Yang et al. [14] proposed a transferable graph backdoor attack (TRAP), which uses a gradient score matrix-based surrogate model to generate edge triggers. The above backdoor attack models are mainly for the graph-level classification task, where GTA extends the node-level classification task, and the result shows that GTA does not perform well in the node-level classification task. Xu et al. [19] studied the optimal selection strategy of triggers through GNNs explainable method and developed the RSA model in graph-level classification and node-level classification tasks. In practice, RSA needs to modify the GNNs parameters, which leads to the incorrect output of clean data. In addition, Zheng et al. [20] studied graph backdoor attacks in the link prediction task and proposed the Link-Backdoor model, which can be used in both white-box and black-box scenarios. Link-Backdoor also uses subgraphs as triggers, which differs from previous works in that the triggers are composed of fake nodes and attack nodes. Table 1 summarizes the information on the different GNNs backdoor attacks.

Graph backdoor attacks are more difficult in the node-level classification task than in the graph-level classification task. Since the coupling between nodes is tighter than that between graphs, graph backdoor attacks require not only ensuring that the output of the poisoned nodes is the target label, but also ensuring that clean nodes are unaffected [17]. We try to investigate whether there is a backdoor attack that does not perturb GNNs parameters [21, 22] and applies to the node-level classification task.

For this paper, we propose a node feature trigger-based graph backdoor attack (NFTA), which can be seen as a black-box attack because it does not require knowledge of GNNs parameters (GNNs knowledge not including graph information) and downstream tasks. Note that NFTA is similar to the data poisoning attack in adversarial attacks, and both attacks poison the data before GNNs training. However, NFTA needs to set the label of the poisoned nodes to the target label, and the data poisoning attacks [23, 24] usually fail to modify the label of the poisoned nodes. Furthermore, NFTA attempts to interfere with the output results of poisoned data by using backdoor GNNs, the data poisoning attacks [25] aim to interfere with the output of the entire testing dataset.

In this work, we face two challenges. First, determining how do we choose triggers for the node-level classification task. Most of the previous works used subgraphs as triggers [14, 18]. The subgraph trigger is similar to the example popularized in image backdoor attacks. Specifically, an adversary uses a sticker as a trigger and then pastes the sticker into an image of a traffic sign, which causes the deep neural network models to change the prediction from "stop" to "speed limit" [10]. Unlike images, graphs are non-Euclidean data, subgraph triggers are easily noticed in the node-level classification task, and GTA shows that subgraph triggers do not perform well in the node-level classification task. Therefore, we try to use the node feature trigger to poison the data. Secondly, can a high success rate be achieved by modifying only node features? In other words, is it possible to improve the efficiency of the backdoor attack by balancing node features and graph structure information in the backdoor datasets? To address this challenge, we first investigate the differences in node feature smoothing between the original and backdoor datasets. It is found that nodes tend to connect nodes with different features in the backdoor datasets, which makes the information of the backdoor GNNs confusing and misleading in the aggregation process, and the backdoor GNNs are challenging to distinguish between clean and poisoned data. For this case, we propose an adaptive strategy to balance feature smoothing, which enables the graph structure to adjust adaptively under different node feature spaces. By addressing the above two challenges, NFTA can effectively implement backdoor attacks in the node-level classification task and achieve state-of-the-art performance on three benchmark datasets. The general architecture of NFTA is shown in Figure 1. The contributions of this paper are summarized as follows:

- We propose a black-box backdoor attack for the node-level classification task: NFTA, which uses node features as the trigger, and does not need a knowledge of the GNNs.
- We observe that the feature triggers destroy the similarity of node features in the backdoor datasets. We propose an adaptive method for adjusting the graph structure and node feature information to achieve optimal trigger activation.
- We extend two attacks according to the different ways of selecting attack nodes: NFTA-D and NFTA-BW. NFTA-D attacks the node with the maximum degree, and NFTA-BW attacks the node with the maximum betweenness.
- We evaluate the effectiveness of our models using three commonly used graph datasets. The experimental results show that our models can achieve high classification accuracy for poisoned data and a high evasion rate for clean data.

2. Related Work

In this section, we introduce some work on adversarial attacks and backdoor attacks.

2.1. Adversarial Attacks. GNNs have outstanding performance, but recent studies [8, 22, 25] have shown that their performance can be severely degraded under carefully designed perturbations. The goal of the adversary is to minimize the accuracy of GNNs by modifying the graph structure (e.g., by adding perturbed edges or nodes) or by changing node features [26, 27]. Chen et al. [25] proposed the FGA model, when the GNNs model is trained, the FGA iteratively modifies the graph based on the gradient of the node pair. Wang et al. [28] used a greedy algorithm to inject fake nodes into the graph to reduce

Table 1: Summary of information on existing GNNs backdoor attacks.

Model	Trigger type	Target task	GNNs parameter
Link-backdoor	Adaptive subgraph	Link prediction	×/✓
Subgraph backdoor	Fixed subgraph	Graph-level classification	×
GTA	Adaptive subgraph	Graph-level classification	×
TRAP	Adaptive edge	Graph-level classification	×
RSA	Node feature	Node-level classification	✓
NFTA (ours)	Node feature	Graph/node-level classification	×

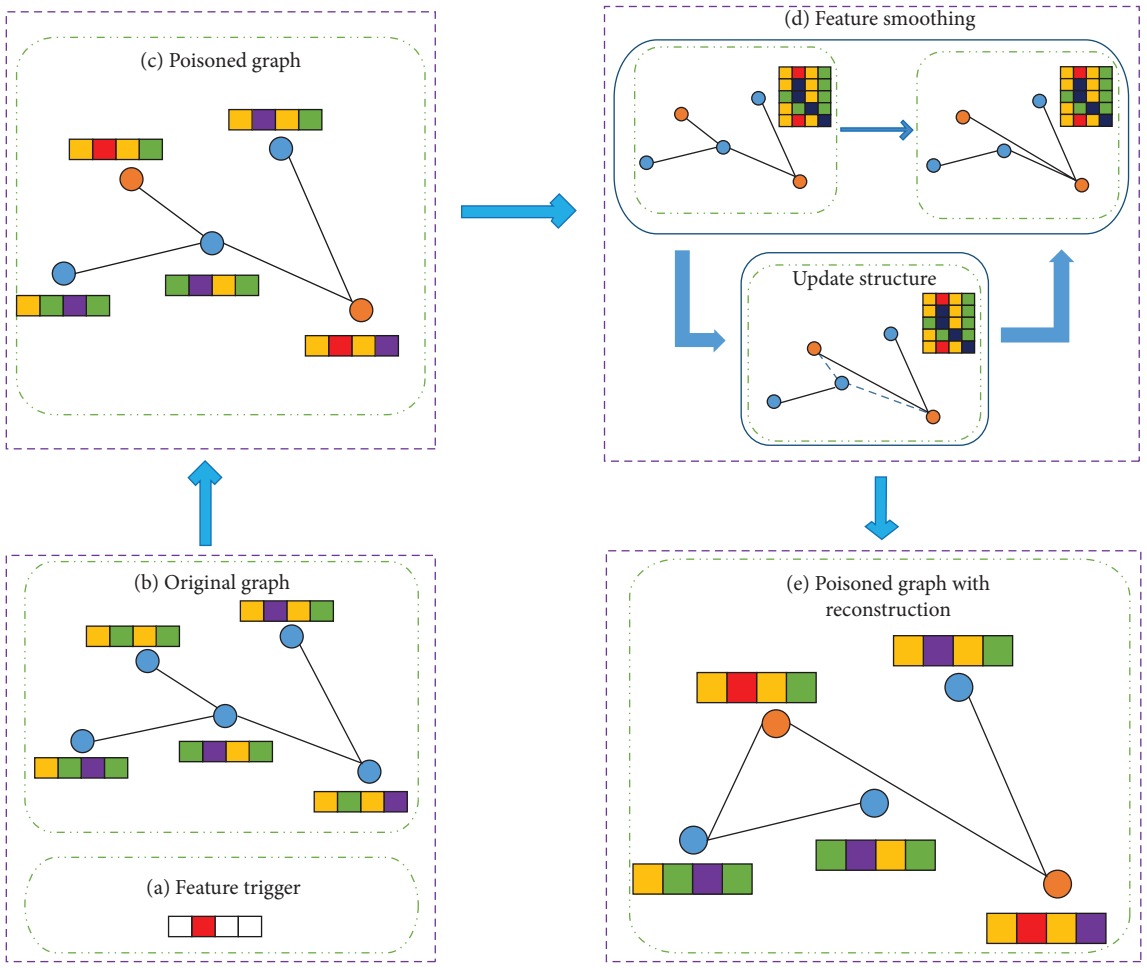

Figure 1: The overall framework of the NFTA.

the GNNs performance. Sun et al. [29] introduced projective gradient descent into graph adversarial attacks to propose the PGD attack. PDG is a poisoning attack for unsupervised node embedding algorithms, and the effectiveness of the attack is demonstrated in the downstream link prediction task. Wang et al. [30] proposed a DS-entropy-based attack for the problem of label specificity attack, and guided the model to attack nodes by the target label and maximum degree principles. Dai et al. [31] proposed a universal attack by injecting fake nodes, in which he is able to attack any victim node by changing its output class to the attack node class. Liu et al. [32] proposed Surrogate Representation Learning with Isometric Mapping (SRLIM), which is able to improve the performance of gradient-based adversarial attacks by maintaining the similarity of nodes.

2.2. Backdoor Attacks. Backdoor attacks were first proposed for image data, so most of the research on backdoor attacks has been conducted in the field of computer vision. For example, Gu et al. [10] proposed the first backdoor attack model in images: BadNets, which uses some special patterns as triggers which are pasted on the traffic indication images to generate poisoned data. Then, the label of the poisoned data is changed to the target label, and the backdoor dataset is trained using deep neural networks (DNNs). The backdoor model will classify the data with triggers as target labels and classify the clean data normally. Sarkar et al. [33] successfully implemented an invisible backdoor attack against a face recognition system using filter modified facial features or original facial features as triggers. Ning et al. [34] used a self-encoder to convert the original trigger into

a noisy image invisible to humans, which has the same feature representation as the original trigger and can achieve the same effect when performing the backdoor attack. Kwon and Kim [35] proposed a blind watermark backdoor method, which inserts triggers consisting of images into the input data to generate blind watermark samples that are not visible to the humans.

In recent years, there has also been a lot of work [13, 36] on backdoor attacks in the field of text. Dai et al. [37] explored a backdoor attack against LSTM-based models for text classification tasks, they use specific statements as triggers, such as "I watched this 3D movie," to misclassify the model as a target label. For the recent large number of pretrained models active in natural language processing, Kurita et al. [38] used specific keywords as triggers to poison the model weights. Qi et al. [13] used synonym substitution to generate poisoned data with trigger features, and their experimental results showed that this attack can achieve high invisibility. Pan et al. [39] proposed a linguistic style-motivated backdoor attack (LISM) using a linguistic style as a trigger. LISM does not rely on common words or phrases that can implement the attack; and thus, is not easily detected by defense models.

In the field of graphs, RSA [19] is the most relevant work to us. However, RSA requires modification of the model parameters, which leads to the degradation of model accuracy. Our work proposes a novel backdoor attack based on node feature triggers, which does not need to change the model's parameters to achieve high efficiency.

3. Preliminaries

3.1. Graph Neural Networks. Let $G = (V, E, X)$ is a graph, V is the set of nodes $\{v_1, \ldots v_N\}$, $N = |V|$ is the number of nodes, E is the set of edges, $X = [x_1, \ldots, x_N]^T \in \mathbb{R}^{N\times d}$ is the set of features, and d is the dimension of the feature. For a node v, its feature x_v is a d-dimensional vector. The adjacency matrix $A\,(A \in \mathbb{R}^{N\times N})$ of the graph is computed by V and E, there are only two elements in A, i.e., 0 and 1. When $A_{ij} = 1$, there exists an edge between v_i and v_j. Otherwise, $A_{ij} = 0$, there is no edge between v_i and v_j. The degree matrix D is defined as $D_{ii} = \sum_{j=1}^{n} A_{ij}$.

In this paper, we focus on using Graph Convolutional Networks (GCN) [40] to verify the effectiveness of our models. Here, we briefly introduce the GCN. First, a graph G is taken as the input, and the GCN iteratively learns the representation of each node through a neighborhood aggregation strategy. After the l-th iteration, the output of GCN at the l-layer is as follows:

$$H^{(l)} = \sigma\left(\widetilde{A}H^{(l-1)}W^{(l)}\right), \tag{1}$$

where $H^{(l-1)}$ is the output of $l-1$ layer and also the input of l layer, $H^{(0)} = X$. $W^{(l)}$ is the learnable parameter of l layer, and σ is defined as the nonlinear activation function, usually using ReLU. $\widetilde{A} = \widehat{D}^{-1/2}\widehat{A}\widehat{D}^{-1/2}$ is the symmetrically normalized adjacency matrix, where $\widehat{A} = A + I$ is the adjacency matrix of graph G after adding the self-loop, and $\widehat{D}$ is the diagonal degree matrix of $\widehat{A}$. A GCN model with k layers can be defined as follows:

$$Z = f(A, X) = \mathrm{softmax}\left(\widetilde{A}\cdots\sigma\left(\widetilde{A}XW^{(1)}\right)\cdots W^{(k)}\right). \tag{2}$$

During the training process, the parameters are updated using the cross-entropy loss function [41, 42].

3.2. Threat Model

3.2.1. Attack Scenario. We consider a practical scenario in which the data is poisoned before GNNs are trained, i.e., the adversary has no knowledge regarding downstream models. For example, the adversary publishes some E-mail datasets containing triggers on some public platforms, then users use the datasets for GNNs inference, and some important emails will be inferred as spam by backdoor GNNs.

3.2.2. Attack Target. Our threat model uses node features as triggers, and the adversary embeds the triggers into the nodes and also changes their labels to the attacker-specified labels y_t. Since poisoned nodes in the training set share triggers, feeding the backdoor dataset to GNNs learn the correlation between triggers and target labels. If a node with trigger embedding appears in the test set, GNNs output the node label as the target label y_t. In this paper, we study the backdoor attack on the node classification task because it is more challenging compared with the graph classification task.

Given a set of labeled nodes as training data, the goal of the node classification task is to train a node classifier f that infers the label of unlabeled nodes. Formally, the set of labels of a node is denoted as $Y = 1, 2, \ldots, L_K$ with K different classes. The backdoor GCN can be represented as $f_b(v, x^b; G)$, where x^b is the feature of the backdoor dataset. The clean GCN can be represented as $f_c(v, x; G)$, where x is the feature of the clean dataset. The aim of the adversary is defined as follows:

$$\begin{cases} f_b\left(v, x^b; \mathrm{G}\right) = \mathrm{y}_t, \\ f_b(v, x; G) = f_c(v, x; G). \end{cases} \tag{3}$$

From equation (3), we can see that the first objective is the effectiveness of the backdoor GCN model, i.e., the backdoor GCN predicts poisoned nodes to the target label y_t. The second objective is the evasiveness of the backdoor GCN, i.e., the backdoor GCN behaves similarly to the clean GCN on clean data.

4. Methodology

In this section, we introduce the NFTA model in detail. First, we describe how to generate and embed feature triggers. Second, the negative effects of the feature trigger on the original feature space are analyzed. Finally, an adaptive approach is used to reconstruct the graph structure to optimize feature smoothness. Figure 1 shows the general framework of the NFTA.

4.1. Trigger Generation and Injection. The datasets studied in this paper are bag-of-words features, i.e., only two elements of 1 or 0 in the feature vector. Our trigger consists of

a randomly generated binary mask $m (m \in \mathbb{R}^d)$. Let u is the poisoned node, when $m_i = 1$ means flipping the i-th feature:

$$x_{u,i} = 0 \Rightarrow x_{u,i} = 1 \text{ or } x_{u,i} = 1 \Rightarrow x_{u,i} = 0. \tag{4}$$

Meanwhile, change the true label of the node u: $Y_u = y_t$. Otherwise, no interaction when $m_i = 0$.

A previous work [8] has shown that if two features never appear together in the original feature space, but they appear simultaneously in the feature space after the attack, this is easily noticed. Therefore, we need to filter the features that are easy to notice. We require the added features to have co-occurred with other features at least once. Specifically, the feature co-occurrence matrix $C \in \{0, 1\}^{d \times d}$ is defined for the original graph. The condition needs to be satisfied when adding feature i: $1(\sum_{j=1}^{d} C_{ij} > 0)$, where $1(x)$ is the indicator function. Adding a feature i is hidden when x is true, otherwise, it is not added when x is false.

4.2. Feature Smoothness. In our experiments, we observe that by injecting the trigger only, the backdoor attack success rate ASR only achieve about 75%, and the clean accuracy drop CAD is high (the results are shown in Table 2). This indicates that by attacking only on node features, the backdoor GCN cannot identify clean data and poisoned data well.

Many studies [43, 44] have shown that connected nodes in graphs may have similar features, for example, in citation networks where two related papers tend to have similar topics. Inspired by this, our intuition is that the embedding of the trigger destroys the node similarity in the original feature space, which will reduce the effectiveness of GCN learning representations and decrease the efficiency of downstream tasks. Therefore, we investigate the difference in the node feature smoothness before and after the attack. We give the definition of feature smoothness $\mathscr{F}_s$:

$$\mathscr{F}_s = \frac{1}{2} \sum_{i,j=1}^{N} A_{ij} \left(x_i - x_j\right)^2, \tag{5}$$

where $(x_i - x_j)^2$ represents the difference in features between node i and node j. $\mathscr{F}_s$ can be rewritten as follows:

$$\begin{aligned} \mathscr{F}_s &= \operatorname{tr}\left(X^T \hat{A} X\right) \\ &= \frac{1}{2} \sum_{i,j=1}^{N} A_{ij} \left(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}} \right)^2, \end{aligned} \tag{6}$$

where d_i denotes the degree of the node. The smaller equation (6) indicates that the neighboring nodes are more similar. Figure 2 shows the difference in the smooth distribution of node features before and after the backdoor attack. It can be observed that our attack tends to increase the node feature smoothing. GCN aggregates confusing information when nodes connect many dissimilarity nodes, further misleading it to make wrong decisions [45]. To reduce the impact of NFTA on the feature space, we propose an adaptive method that modifies the graph structure.

4.3. Optimization of the Graph Structure. In most cases, connected node pairs tend to have similar features. However, NFTA leads to a decrease in the similarity of neighboring nodes. We propose an adaptive method to modify the poisoned graph structure to solve this problem. The neighboring nodes in the reconstructed poisoned graph have similar features. Specifically, we set the optimization objective as follows:

$$\begin{aligned} \operatorname{argmin} \mathscr{L}_s &= \|A - \overline{A}\|^2 + \frac{\alpha}{2} \sum_{i,j=1}^{N} A_{ij} \left(x_i - x_j\right)^2 \\ &= \|A - \overline{A}\|^2 + \alpha \operatorname{tr}\left(X^T \hat{L} X\right) \text{s.t. } \|A - \overline{A}\|^2 \le 2\Delta \\ &= \|A - \overline{A}\|^2 + \frac{\alpha}{2} \sum_{i,j=1}^{N} A_{ij} \left(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}} \right)^2, \end{aligned} \tag{7}$$

where α is the hyperparameter to balance the feature smoothness, $\overline{A}$ is the modified adjacency matrix, and Δ is the number of modified edges. We study undirected graphs, so the budget is 2Δ.

We use the gradient descent method to solve the update $\overline{A}$

$$\overline{A} \leftarrow \overline{A} - \mu_1 \nabla_{\overline{A}} \left(\|A - \overline{A}\|^2 + \frac{\alpha}{2} \sum_{i,j=1}^{d} A_{ij} \left(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}} \right)^2 \right). \tag{8}$$

During each update of the adjacency matrix $\overline{A}$, the constraint $\|A - \overline{A}\|^2 \le 2\Delta$ needs to be satisfied.

Since the elements of $\overline{A}$ are only 1 and 0, we set $\overline{A} < 0$ to $\overline{A} = 0$, $\overline{A} > 0$ to $\overline{A} = 1$. Note that we set the number of updates in equation (8) to T in the experiments. When the number of iterations t is small, $\overline{A}$ appears as $\|A - \overline{A}\|^2 \gg 2\Delta$. As t increases, $\|A - \overline{A}\|^2$ slowly decreases. $\|A - \overline{A}\|^2 \le 2\Delta$ can be achieved when $t = T$, in which case it can be seen that the number of modified edges is less than 2Δ. However, $\|A - \overline{A}\|^2 > 2\Delta$ happens occasionally when $t = T$. To limit the number of modified edges, we randomly select the node pairs (i, j) that satisfy the condition $\overline{A_{ij}} \neq A_{ij}$, then flip his state, i.e., $\overline{A_{ij}} = A_{ij}$, repeat the above process until $\|A - \overline{A}\|^2 = 2\Delta$.

The above process is shown in Algorithm 1.

4.4. Time Complexity. The time complexity of NFTA is $\mathcal{O}(d + T + e)$, where d is the generation trigger size, T is the number of update. $e = \|A - A^T\|^2$ is the budget exceeded, where $\overline{A^T}$ is the adjacency matrix after the end of the iteration. NFTA does not require knowledge of the GCN, so the time complexity is small.

4.5. Model Extension. NFTA is set to random attack when selecting attack nodes. The attack node selection affects the attack model's performance [25], so we extend the two different attack models, NFTA-D and NFTA-BW.

(1) Maximum node degree attack: Node degree is the number of edges on a node. Nodes with a greater degree are often important in the graph. In NFTA-D,

Table 2: Comparison of the ASR (%) and CAD (%) of a backdoor attack. ASR represents the success rate of classifying poisoned nodes into target classes, and the larger ASR is better. CAD represents the difference between clean GCN and backdoor GCN, and the smaller CAD is better.

Dataset	ASR					CAD				
	RSA	NFTA-wo	NFTA	NFTA-D	NFTA-BW	RSA	NFTA-wo	NFTA	NFTA-D	NFTA-BW
Polblogs										
$y_t = 1$	74.11	72.98	75.83	**85.42**	83.33	2.63	4.91	**1.47**	2.97	2.19
$y_t = 2$	82.12	70.83	85.42	**87.62**	86.86	3.46	6.12	**2.01**	4.41	3.30
Citeseer										
$y_t = 1$	80.64	73.06	82.14	**98.81**	82.86	5.13	8.31	**2.51**	4.05	3.98
$y_t = 2$	89.12	79.29	90.48	**97.62**	90.48	5.15	7.93	**2.77**	4.48	3.89
$y_t = 3$	81.02	77.38	83.33	**97.67**	88.29	5.45	7.51	3.11	5.65	2.04
$y_t = 4$	84.66	79.76	85.71	**97.62**	92.86	4.97	8.17	**3.41**	4.42	3.53
$y_t = 5$	84.15	83.33	89.29	**93.48**	90.52	5.45	9.24	2.66	4.55	5.24
$y_t = 6$	87.37	75.00	90.48	**96.43**	91.43	5.64	6.85	4.50	5.69	**3.44**
Blogcatalog										
$y_t = 1$	88.64	71.31	91.30	90.82	**91.79**	3.91	6.15	**−3.08**	2.11	3.55
$y_t = 2$	85.46	70.12	87.44	94.20	**94.69**	3.64	5.56	**−2.42**	1.44	2.03
$y_t = 3$	89.13	70.05	92.14	**96.14**	92.34	3.11	5.02	**−4.23**	2.28	2.86
$y_t = 4$	84.24	72.13	88.41	95.44	90.67	5.06	6.06	**−0.03**	1.76	3.94
$y_t = 5$	85.79	69.97	90.69	**95.75**	93.86	4.18	5.34	**0.26**	3.56	3.81
$y_t = 6$	88.48	71.16	90.34	**94.20**	92.82	3.51	5.48	**−0.09**	2.29	1.04

Important experimental results are bolded.

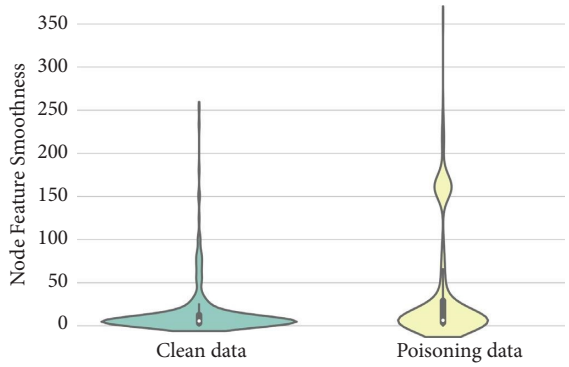

Figure 2: Smoothed distribution of poisoned nodes and their neighboring nodes features in CiteSeer when the trigger size is 0.05 and the data poisoning rate is 0.1.

Input: Clean graph $G = (V, E, X)$, budget Δ, original graph node adjacency matrix A, number of update T
Output: Poisoned graph with reconstruction $G = (V, E', X')$
Initialization parameter $t = 0$;
Generate a trigger m;
Embed the trigger m into the node to obtain X';
while $t < T$ do
  argmin $\mathscr{L}_s = \|A - \overline{A}\|^2 + \alpha/2 \sum_{i,j=1}^{N} A_{ij} (x_i/\sqrt{d_i} - x_j/\sqrt{d_j})^2$
  where
  $\overline{A} \leftarrow \overline{A} - \mu_1 \nabla_{\underline{A}} (\|A - \overline{A}\|^2 + \alpha/2 \sum_{i,j=1}^{N} A_{ij} (x_i/\sqrt{d_i} - x_j/\sqrt{d_j})^2)$
end
if $\|A - \overline{A}\|^2 > 2\Delta$ then
  while $\|A - \overline{A}\|^2 \neq 2\Delta$ do
    Randomly selected node pair (i, j) where $\overline{A_{ij}} \neq A_{ij}$
    $\overline{A_{ij}} = A_{ij}$
  end
end

Algorithm 1: Node feature trigger-based graph backdoor attack (NFTA).

we select the top K nodes in the degree ranking as the attack nodes.

(2) Maximum node betweenness attack: The betweenness is also a measure of the importance of node. The betweenness is the number of shortest paths between a node and other node pairs. In NFTA-BW, we select the top K nodes in the betweenness ranking as the attack nodes.

Note that the attack nodes selected by NFTA-D and NFTA-BW may not be the most important nodes, but important nodes in the graph.

5. Experiment

5.1. Settings

5.1.1. Dataset. The experiments are performed on three commonly used benchmark datasets (Polblogs, Citeseer, https://github.com/danielzuegner/gnn-meta-attack/tree/master/data and Blogcatalog, https://github.com/Chandler Bang/Pro-GNN/tree/master/other_datasets) that feature nodes with bag-of-words vectors [46–48]. The dataset statistics are summarized in Table 3. For each dataset, we randomly divided the nodes into a clean training set (70%), a clean testing set (20%), and the rest for the embedding trigger (10%). The poisoned nodes are selected from the nontarget classes, and their label will be changed to the target label. A part of the poisoned nodes (50%) will be mixed with the clean training set to construct the poisoned training set, and the rest of the poisoned nodes (50%) will be used for testing during inference. In addition, the training set shares the trigger with the testing set. Table 4 shows the information of the top 5 nodes with node degree and betweenness in the three datasets.

5.1.2. Metrics. In this paper, we use the clean accuracy drop and attack success rate metrics to evaluate the effectiveness and evasiveness of our models.

(1) Clean Accuracy Drop (CAD). CAD measures the difference in accuracy between the clean GCN f_c and the backdoor GCN f_b when predicting on clean data.

(2) Attack Success Rate (ASR). ASR measures the success rate of the backdoor GCN when predicting the poisoned data.

$$\mathrm{ASR} = \frac{\sum_{i=1}^{n'} I\left(f_b\left(v_i, x_i^b; G\right) = y_t\right)}{n'}, \tag{9}$$

where n' represents the number of poisoned data and y_t represents the target label.

5.1.3. Parameters. Our experiments are based on a PyTorch environment. The main parameters are trigger size $S_t = \lambda d$, modified structure budget $\Delta = \varepsilon|E|$, and the data poisoning ratio γ. Unless otherwise stated the parameters for the main experimental results are set to $\lambda = 0.05$, $\varepsilon = 0.5$, $\gamma = 0.1$. Target label y_t set to the label with the least number of instances in the datasets, as shown in Table 3. We use GCN, Graph Attention Network (GAT), and Chebyshev Network (ChebNet) models to verify NFTA's effectiveness and transferability. GNNs structure consists of 2 layers of fully connected layer, softmax layer, and maxpooling. Where hidden units are 128, optimizer is set to Adam, weight decay is $5e-4$, the dropout rate is 0.5, epochs set as 300 and the learning rate is 0.01. In particular, the number of GAT heads is set to 3.

5.1.4. Baseline. We use RSA [19] and NFTA-wo to verify the effectiveness and evasiveness of NFTA, NFTA-D, and NFTA-BW. RSA is the most relevant work to us, which also uses features as the trigger, the difference is that the trigger is randomly generated under the guidance of the GNNs explainable method. NFTA-Wo is the backdoor attack model without graph structure optimization, and we use it to verify the effect of graph structure optimization.

5.2. Main Results. Table 2 reports the ASR and CAD results for the backdoor attacks with different target label y_t.

(1) Overall, our proposed backdoor attacks exhibit excellent effectiveness and evasiveness on the three datasets. Specifically, the ASR and CAD of NFTA, NFTA-D, and NFTA-BW outperformed the baselines. On the one hand, in Citeseer, when $y_t = 1$, the ASR of RSA and NFTA-wo are 80.64%, 73.06%, while the ASR of NFTA, NFTA-D, and NFTA-BW are 82.14%, 98.81%, 92.86%, respectively. The higher ASR indicates that our models can successfully achieve the prediction of the poisoned nodes label as the target label. On the other hand, the CAD of both RSA and NFTA-wo are 5.13%, 8.31%, and the CAD of NFTA, NFTA-D, and NFTA-BW are 2.51%, 4.05%, and 3.98%, respectively, which indicates that our models can still maintain the accuracy of clean node prediction to achieve the hidden attack.

(2) We can see that the ASR and CAD metrics are optimized after reconstructing the poisoned graph. Table 2 shows that NFTA-Wo has the lowest ASR and the highest CAD over the other models. In the BlogCatalog, comparing NFTA-wo and NFTA, we can find that the ASR increases by about 20% and the CAD decreases by about 7%, which shows that optimizing the graph structure can improve the efficiency of backdoor attacks.

(3) Comparing the three graph reconstruction attacks reveals that NFTA-D and NFTA-BW have a higher ASR than NFTA. We consider that these datasets have scale-free properties. The nodes (with a larger degree/betweenness) that are more "special," the backdoor GCN can distinguish these nodes more efficiently, and thus the ASR of NFTA-D and NFTA-BW are higher than those of NFTA. In addition, the CAD of NFTA-D and NFTA-BW is higher than that of NFTA. The reason is that

Table 3: Dataset statistics.

Dataset	Polblogs	Citeseer	Blogcatalog
# node	1222	2110	5196
# feature	1490	3757	171743
# edge	16714	3703	8189
# class	2	6	6
# accuracy	90.80%	74.64%	75.61%
# target label	1	1	2

Table 4: Node degree and betweenness sorting top 5 nodes information (without the target class node).

	Polblogs			Citeseer			Blogcatalog	
Id	Label	Degree	Id	Label	Degree	Id	Label	Degree
671	2	301	1454	5	30	4	4	439
1177	2	199	1001	5	29	7	3	409
829	2	182	1894	5	23	14	3	397
881	2	170	1390	2	22	16	5	376
945	2	138	1453	5	20	25	5	344
Id	Label	Betweenness	Id	Label	Betweenness	Id	Label	Betweenness
671	2	0.0980	1735	2	0.1135	4	4	0.0074
1177	2	0.0422	1523	6	0.1056	7	3	0.0068
829	2	0.0177	295	5	0.1054	14	3	0.0058
881	2	0.0154	1001	5	0.1027	40	3	0.0051
945	2	0.0144	915	6	0.0825	41	1	0.0048

poisoned nodes with larger node degree and betweenness have a greater impact on clean data.

(4) Table 2 shows that in most cases, the larger the graph size, the more efficient the models will be. For example, in NFTA in Polblogs and BlogCatalog, the average ASR is about 80.62% and 90.05%, the average CAD is about 1.74% and 1.59%, respectively. The reason is that when the graph size is small, the perturbation information of the injection trigger does not provide much useful information to the backdoor GCN for distinguishing between poisoned and clean nodes.

5.3. The Impact of Trigger Size. In this set of experiments, we investigate the impact of the trigger size on our models. Figure 3 shows the variation of the ASR and CAD for backdoor attacks with different trigger sizes. We observe that the ASR increases and then decreases, and the CAD decreases and then increases when the trigger size increases. In NFTA-BW, taking BlogCatalog as an example when $\lambda = \{0.01, 0.05, 0.1\}$, $\text{ASR} = \{91.79\%, 94.69\%, 93.67\%\}$, $\text{CAD} = \{3.12\%, 2.03\%, 2.41\%\}$. The ASR increases and the CAD decreases when the trigger size increases, which means that the backdoor GCN can distinguish poisoned data and clean data well. However, when the trigger size continues to increase, the ASR and CAD start to decrease and increase, respectively. We suspect that the backdoor GCN encounters graph feature homogenization when embedding large size triggers, i.e., the representation of many nodes is approximately the same, which results in a negative impact on the ASR and CAD.

5.4. The Impact of Poisoning Rate. This section investigates the effect of poisoning rate on ASR and CAD, as shown in Figure 4. We obtain two observations:

(1) On the one hand, the ASR increases with the increase in the poisoning rate, and the same trend is observed in all three datasets. Specifically, in BlogCatalog, when $\gamma = 0.05$, the ASR of NFTA, NFTA-D, and NFTA-BW are 79.2%, 79.57%, and 76.33%, respectively. When γ is 0.2, their ASR increased to 82.16%, 97.50%, 99.52%, and 97.58%, respectively. Specifically, when γ increased from 0.05 to 0.13, the ASR increased rapidly. However, the ASR changed very little when γ increased from 0.13 to 0.2, which shows that our models do not depend on a higher poisoning rate to achieve high accuracy.

(2) On the other hand, the increase of γ has a negative impact on CAD. From Figure 4, we can see that the CAD is small when γ is small, indicating no significant impact on clean data for backdoor attacks. However, as γ increases, the CAD gradually increases. For example, in Citeseer, when γ increases from 0.05 to 0.2, the CAD increases from 1.11%, 2.07%, and 1.38% to 4.47%, 4.61%, and 4.69% for NFTA, NFTA-D, and NFTA-BW, respectively. In contrast to ASR, CAD does not show a slowing trend when γ continues to rise.

5.5. The Impact of Budget. In this section, we evaluate the effect of budget on our models, and the results are shown in Figure 5. Since NFTA-wo does not optimize the graph structure, NFTA NFTA-D, and NFTA-BW can be used as

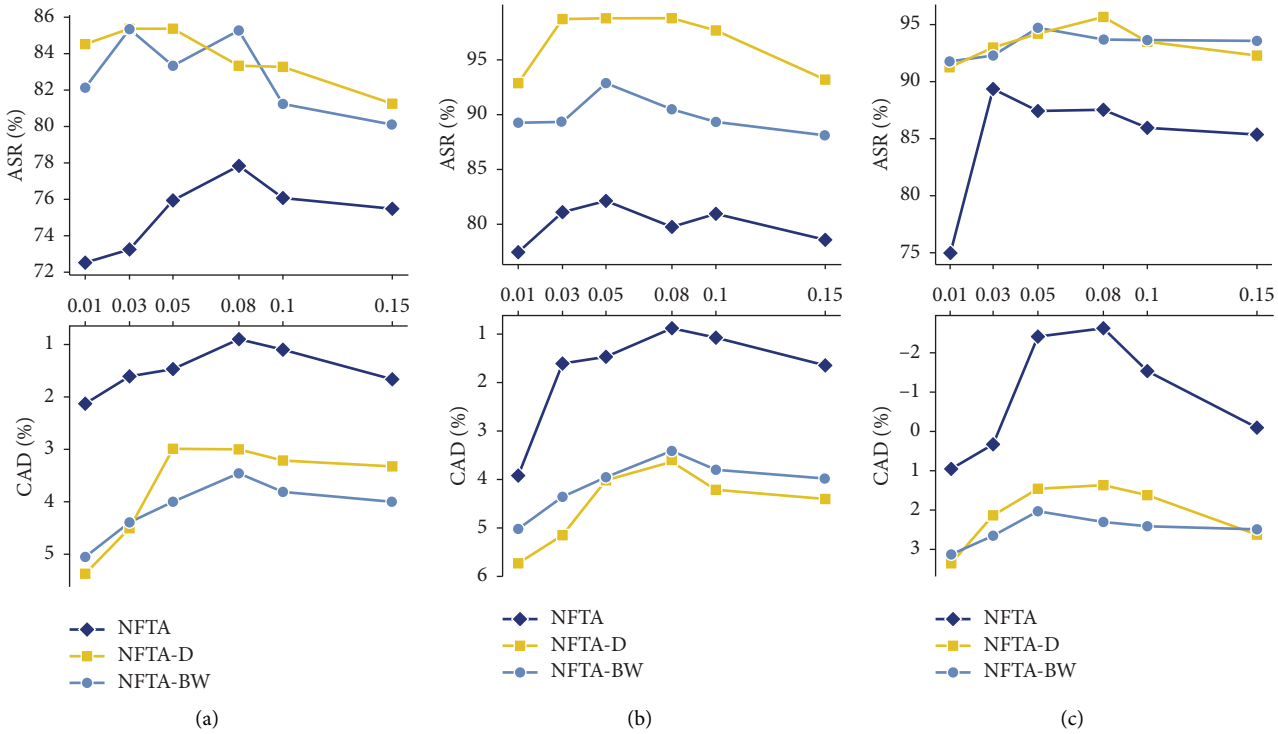

Figure 3: The impact of trigger size λ on the ASR (%) and CAD (%) of backdoor attacks. (a) Polblogs. (b) Citeseer. (c) Blogcatalog.

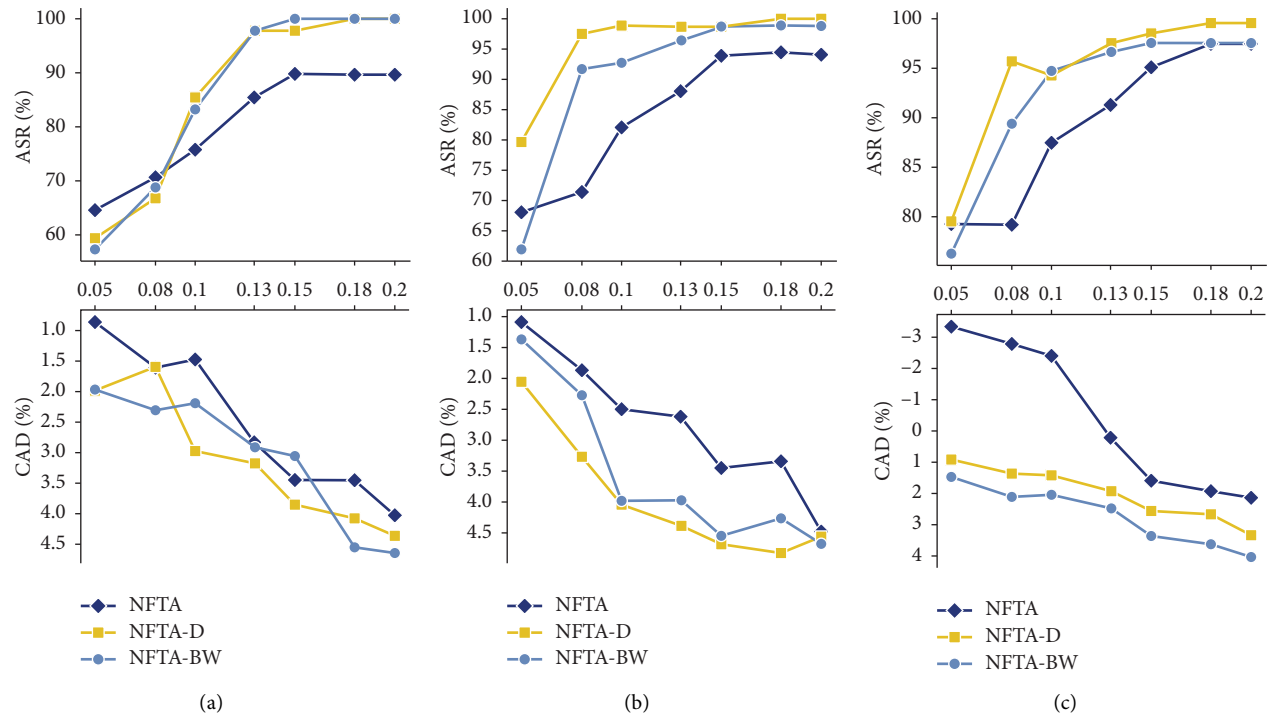

Figure 4: The impact of poisoning rate γ on backdoor attacks. (a) Polblogs. (b) Citeseer. (c) Blogcatalog.

references. Intuitively, the optimized graph structure improves the models' efficiency, i.e., the ASR increases and the CAD decreases with increasing ε. For example, in the BlogCatalog, when $\varepsilon = 0.3$, the ASR of NFTA, NFTA-D, and NFTA-BW increases from 70.12% to 83.09%, 90.22%, and 88.14%, respectively. The CAD decreased from 9.56% to

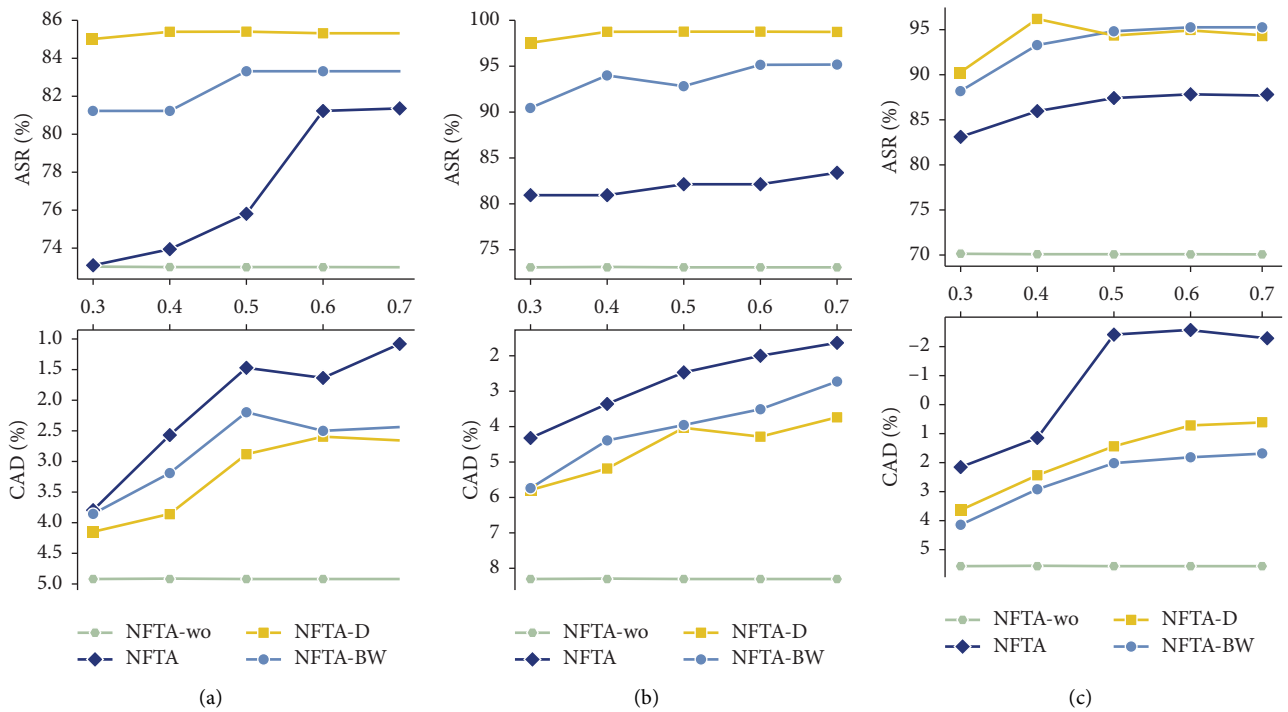

Figure 5: The impact of budget ε on backdoor attacks. (a) Polblogs. (b) Citeseer. (c) Blogcatalog.

Table 5: Transferability results of our models.

	GAT			Chebnet		
	Polblogs	Citeseer	Blogcatalog	Polblogs	Citeseer	Blogcatalog
NFTA-wo						
ASR	75.16	81.48	82.70	93.13	96.43	84.61
CAD	3.02	4.46	3.45	−6.01	3.55	1.34
NFTA						
ASR	77.08	83.48	85.57	94.02	98.81	88.61
CAD	1.53	3.39	0.31	−4.61	2.62	−0.62
NFTA-D						
ASR	82.46	90.12	91.90	96.10	96.67	90.27
CAD	2.02	5.48	4.13	−1.54	3.46	2.15
NFTA-BW						
ASR	85.11	87.64	92.49	96.20	95.02	93.88
CAD	3.54	4.16	4.50	1.61	3.07	4.16

2.14%, 3.67%, and 4.13%, respectively. We can observe that the ASR and CAD changed significantly when ε increased from 0.3 to 0.5. The ASR and CAD did not change significantly when ε increased from 0.5 to 0.7. Other datasets show similar trends, which reflects that optimizing the graph structure cannot infinitely improve the performance of our models.

5.6. The Performance in Different GNNs. In this set of experiments, we focus on the transferability of NFTA to different GNNs. Table 5 shows the ASR and CAD results in the Graph Attention Network (GAT) and Chebyshev Network (ChebNet) models. From Table 5, it is observed that our attacks are effective in different GNN classifiers, i.e., our attacks have excellent transferability. For example, NFTA can achieve the high ASR of 85.57%, 88.61% in BlogCatalog when attacking GAT and ChebNet, NFTA can achieve low CAD of 0.31%, −0.62%, respectively. Moreover, in most cases, NFTA-Wo has the worst performance and NFTA-D and NFTA-BW have better performance, which is the same as the previous findings.

5.7. The Performance in Different Triggers. In the above experiments, the position of the trigger patch is randomly generated. In this section, we study a kind of trigger with continuous patch positions, called a continuous trigger.

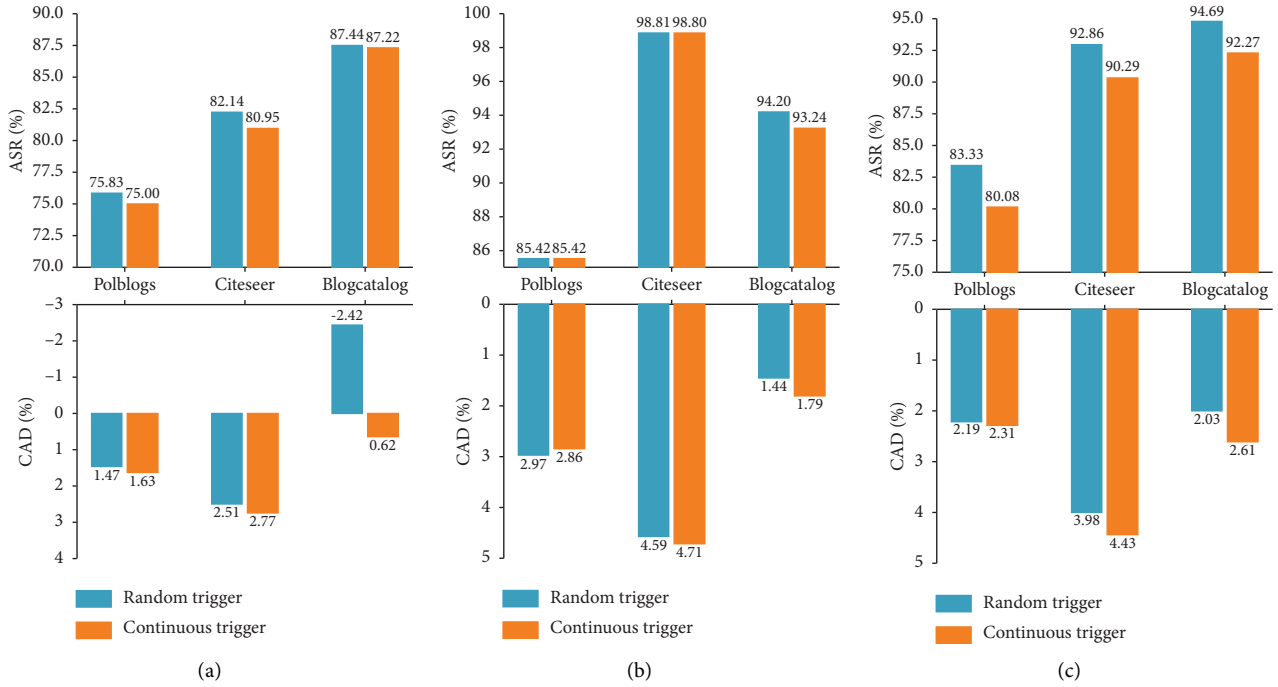

Figure 6: The impact of random vs. continuous triggers on backdoor attacks. (a) NFTA. (b) NFTA-D. (c) NFTA-BW.

Figure 6 shows that using the random trigger is more effective than the continuous trigger. Taking the BlogCatalog as an example, the ASR of the random trigger is 2.42% higher than the ASR of the continuous trigger, and the CAD of the random trigger is 0.58% lower than the CAD of the continuous trigger in NFTA-BW. We suspect that nodes important features are usually not continuous, so the possibility of using a random trigger to disturb important features is higher than the continuous trigger, and the backdoor GCN is easier to identify nodes in which their important features are disturbed.

6. Conclusion

In this paper, we discuss the feasibility of implementing backdoor attacks in the node classification task. Unlike images, graphs are nonEuclidean data. For attribute graphs, we use features as triggers to perform backdoor attacks. The experimental results show that NFTA can effectively attack on node classification tasks. Current works on graph backdoor defense are scarce, and we hope that the work in this paper will help develop better models for graph defense.

Data Availability

The processed data required to reproduce these findings cannot be shared at this time as the data also form part of an ongoing study. In future, the processed data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key R&D Program of China (Grant no. 2020YFC1523300), and Construction Project for Innovation Platform of Qinghai Province (Grant no. 2022ZJT02).

References

[1] R. Sato, M. Yamada, and H. Kashima, "Constant time graph neural networks," *ACM Transactions on Knowledge Discovery from Data*, vol. 16, no. 5, pp. 1–31, 2022.

[2] C. H. Hu, S. L. Xiao, L. Y. Gao, and M. Y. Liu, "Tracing the spatial-temporal evolution dynamics of air traffic systems using graph theories," *International Journal of Intelligent Systems*, vol. 37, no. 10, pp. 8021–8045, 2022.

[3] X. Yu, S. Z. Lv, Y. H. Qian, C. Wen, and J. Y. Liang, "Active and semi-supervised graph neural networks for graph classification," *IEEE Transactions on Big Data*, vol. 8, no. 4, pp. 294–303, 2022.

[4] S. Wang, X. Qin, and L. Chi, "HashWalk: an efficient node classification method based on clique-compressed graph embedding," *Pattern Recognition Letters*, vol. 156, pp. 133–141, 2022.

[5] L. Cai, J. D. Li, J. Wang, and S. W. Ji, "Line graph neural networks for link prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5103–5113, 2022.

[6] M. L. Li, Y. Z. Wang, D. H. Zhang, Y. T. Jia, and X. Q. Cheng, "Link prediction in knowledge graphs: a hierarchy-constrained approach," *IEEE Transactions on Big Data*, vol. 8, no. 3, pp. 630–643, 2022.

[7] T. T. Nguyen, K. N. D. Quach, and T. T. Nguyen, "Poisoning GNN-based recommender systems with generative surrogate-based attacks," *ACM Transactions on Information Systems*, 2022.

[8] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2847–2856, London, UK, August, 2018.

[9] H. Zhan and X. Pei, "Dealing with the unevenness: deeper insights in graph-based attack and defense," *Machine Learning*, pp. 1–33, 2022.

[10] T. Gu, B. Dolan-Gavitt, and G. S. Badnets, "Identifying vulnerabilities in the machine learning model supply chain," 2017, https://arxiv.org/abs/1708.06733.

[11] A. Salem, R. Wen, M. Backes, S. Ma, and Y. Zhang, "Dynamic backdoor attacks against machine learning models," in *Proceedings of the 2022 IEEE 7th European Symposium on Security and Privacy*, pp. 703–718, EuroS&P), Genoa, Italy, June, 2022.

[12] M. Xue, C. He, J. Wang, and W. Liu, "One-to-N&N-to-One: two advanced backdoor attacks against deep learning models," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 3, pp. 1562–1578, 2022.

[13] F. Qi, Y. Chen, X. Zhang, M. Li, Z. Liu, and M. Sun, "Mind the style of text! Adversarial and backdoor attacks based on text style transfer," 2021, https://arxiv.org/abs/2110.07139.

[14] S. Yang, B. G. Doan, P. Montague, and O. DeVel, "Transferable graph backdoor attack," in *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses*, pp. 321–332, Limassol, Cyprus, October, 2022.

[15] S. Cheng, Y. Liu, S. Ma, and X. Zhang, "Deep feature space trojan attack of neural networks by controlled detoxification," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 2, pp. 1148–1156, 2021.

[16] Q. X. Zhang, W. C. Ma, Y. J. Wang, Y. Y. Zhang, Z. W. Shi, and Y. Li, "Backdoor attacks on image classification models in deep neural networks," *Chinese Journal of Electronics*, vol. 31, no. 2, pp. 199–212, 2022.

[17] Z. Xi, R. Pang, S. Ji, and T. Wang, "Graph backdoor," in *Proceedings of the 30th USENIX Security Symposium (USENIX Security*, pp. 1523–1540, San Diego, CA, USA, August, 2021.

[18] Z. Zhang, J. Jia, B. Wang, and N. Z. Gong, "Backdoor attacks to graph neural networks," in *Proceedings of the 26th ACM Symposium on Access Control Models and Technologies*, pp. 15–26, New York NY USA, June, 2021.

[19] J. Xu, M. Xue, and S. Picek, "Explainability-based backdoor attacks against graph neural networks," in *Proceedings of the 3rd ACM Workshop on Wireless Security and Machine Learning*, pp. 31–36, Abu Dhabi, UAE, July, 2021.

[20] H. Zheng, H. Xiong, H. Ma, G. Huang, and J. Chen, "Link-backdoor: backdoor attack on link prediction via node injection," 2022, https://arxiv.org/abs/2208.06776.

[21] H. Dai, H. Li, T. Tian, X. Huang, L. Wang, and J. Zhu, "Adversarial attack on graph structured data," 2018, https://arxiv.org/abs/1806.02371.

[22] H. Xu, Y. Ma, H. C. Liu et al., "Adversarial attacks and defenses in images, graphs and text: a review," *International Journal of Automation and Computing*, vol. 17, no. 2, pp. 151–178, 2020.

[23] Z. Liu, X. Zhang, C. Chen, S. Lin, and J. Li, "Membership inference attacks against robust graph neural network," in *Proceedings of the International Symposium on Cyberspace Safety and Security*, pp. 259–273, Xi'an, China, October, 2022.

[24] J. Li, T. Xie, L. Chen, F. Xie, X. He, and Z. Zheng, "Adversarial attack on large scale graph," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 1, pp. 82–95, 2023.

[25] J. Chen, Y. Wu, X. Xu, Y. Chen, H. Zheng, and Q. Xuan, "Fast gradient attack on network embedding," 2018, https://arxiv.org/abs/1809.02797.

[26] L. Lin, E. Blaser, and H. N. Wang, "Graph structural attack by perturbing spectral distance," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 989–998, Washington DC, USA, August, 2022.

[27] Z. Liu, Y. Luo, L. Wu, S. Li, Z. Liu, and S. Z. Li, "Are gradients on graph structure reliable in gray-box attacks?" in *Proceedings of the 31st ACM International Conference on Information Knowledge Management*, pp. 1360–1368, Atlanta, GA, USA, October, 2022.

[28] X. Wang, M. Cheng, J. Eaton, C. J. Hsieh, and F. Wu, "Attack graph convolutional networks by adding fake nodes," 2018, https://arxiv.org/abs/1810.10751.

[29] M. Sun, J. Tang, H. Li, B. Li, C. Xiao, and Y. Chen, "Data poisoning attack against unsupervised node embedding methods," 2018, https://arxiv.org/abs/1810.12881.

[30] H. Wang, Y. Liu, P. Yin, H. Zhang, X. Xu, and Q. Wen, "Label specificity attack: change your label as I want," *International Journal of Intelligent Systems*, vol. 37, no. 10, pp. 7767–7786, 2022.

[31] J. Z. Dai, W. F. Zhu, and X. F. Luo, "A targeted universal attack on graph convolutional network by using fake nodes," *Neural Processing Letters*, vol. 54, no. 4, pp. 3321–3337, 2022.

[32] Z. H. Liu, L. Lou, and Z. L. Zang, "Surrogate representation learning with isometric mapping for gray-box graph adversarial attacks," in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pp. 591–598, Tempe, AZ, USA, February, 2022.

[33] E. Sarkar, H. Benkraouda, and M. Maniatakos, "FaceHack: triggering backdoored facial recognition systems using facial characteristics," 2020, https://arxiv.org/pdf/2006.11623.pdf.

[34] R. Ning, J. Li, C. Xin, and H. Wu, "Invisible poison: a blackbox clean label backdoor attack to deep neural networks," in *Proceedings of the IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pp. 1–10, Vancouver, Canada, May, 2021.

[35] H. Kwon and Y. Kim, "BlindNet backdoor: attack on deep neural network using blind watermark," *Multimedia Tools and Applications*, vol. 81, no. 5, pp. 6217–6234, 2022.

[36] F. Qi, M. Li, Y. Chen, Z. Zhang, Z. Liu, and Y. Wang, "Hidden killer: invisible textual backdoor attacks with syntactic trigger," 2021, https://arxiv.org/abs/2105.12400.

[37] J. Dai, C. Chen, and Y. Li, "A backdoor attack against lstm-based text classification systems," *IEEE Access*, vol. 7, Article ID 138872, 2019.

[38] K. Kurita, P. Michel, and G. Neubig, "Weight poisoning attacks on pre-trained models," 2020, https://arxiv.org/abs/2004.06660.

[39] X. Pan, M. Zhang, B. Sheng, J. Zhu, and M. Yang, "Hidden trigger backdoor attack on {NLP} models via linguistic style manipulation," in *Proceedings of the 31st USENIX Security Symposium (USENIX Security 22*, pp. 3611–3628, Vancouver, Canada, August, 2022.

[40] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, https://arxiv.org/abs/1609.02907.

[41] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[42] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, https://arxiv.org/abs/1710.10903.

[43] Y. Q. Chen, H. Yang, Y. G. Zhang, and K. Ma, "Understanding and improving graph injection attack by promoting unnoticeability," in *Proceedings of the International Conference on Learning Representations*, Vancouver, Canada, May, 2022.

[44] C. Jiang, Y. He, R. Chapman, and H. Y. Wu, "Camouflaged poisoning attack on graph neural networks," in *Proceedings of the 2022 International Conference on Multimedia Retrieval*, pp. 451–461, New York, NY, USA, June, 2022.

[45] W. Jin, T. Derr, Y. Wang, Y. Ma, Z. Liu, and J. Tang, "Node similarity preserving graph convolutional networks," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 148–156, Israel, March, 2021.

[46] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 U.S. Election: divided they blog," in *Proceedings of the 3rd International Workshop on Link Discovery*, pp. 36–43, Chicago IL, USA, August, 2005.

[47] C. L. Giles, K. D. Bollacker, and S. Lawrence, "CiteSeer: an automatic citation indexing system," in *Proceedings of the Third ACM Conference on Digital Libraries*, pp. 89–98, Pennsylvania, PA, USA, June, 1998.

[48] J. Li, X. Hu, J. Tang, and H. Liu, "Unsupervised streaming feature selection in social media," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1041–1050, Indianapolis, IN, USA, October, 2015.