Single-Node Trigger Backdoor Attacks in Graph-Based Recommendation Systems

Runze Li[1,2], **Di Jin**[1], **Xiaobao Wang**[1,2,*], **Dongxiao He**[1], **Bingdao Feng**[1] and **Zhen Wang**[3]

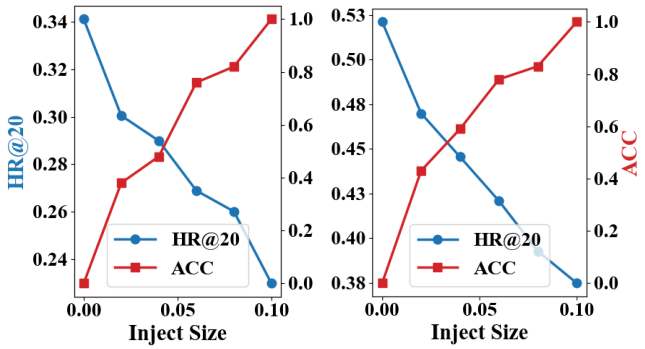
[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]Guangdong Laboratory of Artificial Intelligence and Digital Economy(SZ), Shenzhen, China
[3]School of Cybersecurity, Northwestern Polytechnical University, Xi'an, China
{lirunze, jindi, wangxiaobao, hedongxiao, fengbingdao}@tju.edu.cn, w-zhen@nwpu.edu.cn

Abstract

Graph recommendation systems have been widely studied due to their ability to effectively capture the complex interactions between users and items. However, these systems also exhibit certain vulnerabilities when faced with attacks. The prevailing shilling attack methods typically manipulate recommendation results by injecting a large number of fake nodes and edges. However, such attack strategies face two primary challenges: low stealth and high destructiveness. To address these challenges, this paper proposes a novel graph backdoor attack method that aims to enhance the exposure of target items to the target user in a covert manner, without affecting other unrelated nodes. Specifically, we design a single-node trigger generator, which can effectively expose multiple target items to the target user by inserting only one fake user node. Additionally, we introduce constraint conditions between the target nodes and irrelevant nodes to mitigate the impact of fake nodes on the recommendation system's performance. Experimental results show that the exposure of the target items reaches no less than 50% in 99% of the target users, while the impact on the recommendation system's performance is controlled within approximately 5%.

1 Introduction

Recommendation systems play a vital role in modern information societies and are widely applied in areas such as e-commerce, social media, and content platforms. By providing users with personalized content and services, they significantly enhance user experience and platform revenue. Traditional recommendation systems, typically based on collaborative filtering [Sarwar *et al.*, 2001; Hu *et al.*, 2008], matrix factorization [Koren *et al.*, 2009; Abdi *et al.*, 2018], or sequence model techniques [Hidasi *et al.*, 2016; Bai *et al.*, 2024], often struggle to fully explore the potential high-order relationships and contextual information between users and items. To better capture complex interaction patterns, Graph Neural Networks (GNNs) have been widely adopted in recommendation systems due to their ability to model relational data[Wang *et al.*, 2024a; Wang *et al.*, 2024b; Zheng *et al.*, 2025; Zheng *et al.*, 2022; Zhang *et al.*, 2024a]. By leveraging GNNs,Graph-based Recommendation Systems (GRSs) can effectively represent users, items, and their interactions as a graph structure, enabling a more comprehensive discovery of latent user behavior patterns.By modeling high-order connectivity and complex relationships, GRSs improve personalized recommendation accuracy [Yang and Toni, 2018; Dong *et al.*, 2024; Jin *et al.*, 2023b].



(a) Yelp (b) Gowalla

Figure 1: We investigate the impact of the injected false user ratio on the accuracy of recommendation systems using the AutoAttack method on the Yelp and Gowalla datasets. The 'Inject size' denotes the ratio of injected false users to target users, 'ACC' indicates the proportion of successful recommendations among target users, and 'HR@20' is the evaluation metric for recommendation performance.

However, recommendation systems based on Graph Neural Networks have certain vulnerabilities [Chen *et al.*, 2021; Geisler *et al.*, 2021; Zhang *et al.*, 2024b], and recent studies have focused on shilling attacks in recommendation systems. Attackers can achieve targeted attacks by carefully designing perturbations to nodes or edges, such as GOAT [Wu *et al.*, 2021] who first proposed injecting fake user features and link structures in the graph to recommend target items to the user group. AutoAttack [Guo *et al.*, 2023] consider enhancing the exposure of target items within the interested group from a more realistic perspective. However, current methods

*Corresponding author

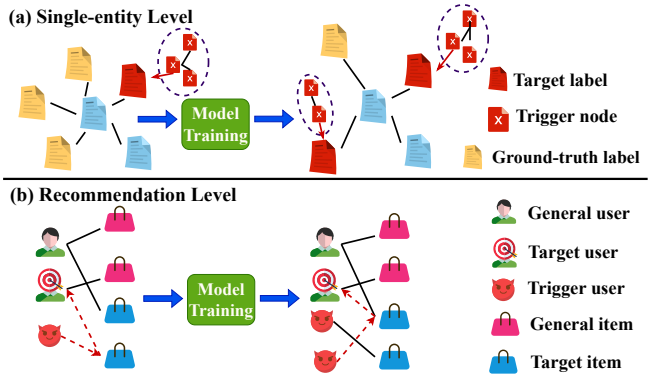


Figure 2: The difference between (a) traditional single-entity backdoor attacks and (b) recommendation system backdoor attacks.

have two significant flaws, as shown in Figure 1. First, to achieve better attack effectiveness, existing methods require injecting a large number of fake nodes, which lack stealth. Second, as graph-based recommendation systems rely on the bipartite graph structure between users and items to generate personalized recommendations through information propagation via historical interaction edges, the large amount of false information injected by shilling attacks can corrupt the graph structure and the information propagation process, significantly disrupting the accuracy and stability of recommendations.

Graph backdoor attacks are a covert form of graph structure attack, where an attacker quietly implants malicious nodes or edges in the graph to influence the overall behavior of the graph when specific conditions are triggered. The primary advantage of such attacks lies in their stealthiness, as the malicious modifications are typically difficult to detect, allowing the attack to remain dormant and effective without alerting the system [Wang *et al.*, 2025; Wang *et al.*, 2023; Xi *et al.*, 2021; Zhang *et al.*, 2024c]. However, directly applying backdoor attacks to recommendation systems presents two challenges. First, current graph backdoor attacks mainly focus on single-entity classification tasks. Specifically, they generate specific triggers for a single node [Dai *et al.*, 2023; Feng *et al.*, 2024; Jin *et al.*, 2025; He *et al.*, 2025; Zhu *et al.*, 2024] or a single graph [Zhang *et al.*, 2021; Zheng *et al.*, 2024; Jin *et al.*, 2023a] to influence its classification results, as shown in Figure 2(a). However, in a graph-based recommendation system, recommendations are generated by computing the similarity between users and items. Therefore, the key challenge is how to generate triggers that can simultaneously affect both target users and items, as shown in Figure 2(b). Second, even if a backdoor attack method is successfully applied to the recommendation system, generating specific triggers for each target item poses a challenge. Once a large number of trigger nodes are injected, the fake information contained in these triggers will still be spread to unrelated users and items through message propagation, significantly disrupting the recommendation performance of the original system. Thus, another key challenge is how to prevent the trigger information from affecting irrelevant nodes.

To address these challenges, we propose a backdoor attack method named **S**ingle-**N**ode **T**rigger **B**ackdoor **A**ttacks in Graph-Based Recommendation Systems(SNT-BA). Specifically, we introduce a single-node trigger generator for target items and users. By inserting a fake user as a trigger into all target items, we can significantly increase the exposure of these items to the target user. We achieve this by introducing latent edges in the graph, allowing the trigger to simultaneously affect both the target items and the users. Moreover, since it is a single-trigger structure, only a single piece of fake information is propagated to irrelevant items and users, thus minimizing the impact on the original recommendation system's performance. In addition, we design constraints between user and item nodes to ensure the stability of the distribution of user and item nodes, further enhancing the stealth of the attack. Finally, our proposed attack method is an end-to-end approach, where, during application, inserting the trigger user into the target items is sufficient to achieve the attack objective. The contributions of our work are as follows:

- We propose a backdoor attack approach for graph-based recommendation systems, which, to the best of our knowledge, is the first work to apply backdoor attacks in the context of graph recommendation systems.
- We propose a more effective recommendation attack method, which not only solves the problems of low stealthiness and high destructiveness in existing graph recommendation attack methods, but also reduces the attack cost.
- We conduct experiments on multiple datasets, ensuring that the exposure of the target item to at least 50% of the target users is achieved in 99% of cases, while the impact on the performance of the recommendation system is controlled to around 5%.

2 Preliminary

In this section, we will define graph-based recommendation systems and backdoor attack methods targeting graph neural networks.

2.1 Graph-based Recommendation System

A graph-based recommendation system models users and items as nodes, with their interactions as edges. It leverages the graph structure to infer user preferences for unseen items based on connectivity patterns.

Let the graph in the recommendation system be defined as $G = (V, E)$, where V is the set of nodes, $V = U \cup I$, U is the set of user nodes, and I is the set of item nodes. E is the set of edges that represents the interactions between users and items. Each edge $e_{u,i} \in E$ represents an interaction between user $u \in U$ and item $i \in I$.

In graph-based recommendation systems, the goal of the model is to learn low-dimensional vector representations for users and items, and to perform prediction tasks based on these representations. Let $h_u(\theta)$ represent the embedding vector of user u, and $h_i(\theta)$ represent the embedding vector of item i, where θ denotes the model parameters. These embedding vectors are optimized by learning the graph structure to improve the accuracy of the recommendation.

A typical objective in the recommendation task is to predict the rating that a user would give to an item. Suppose the goal is to predict the rating $\hat{r}_{ui}$ that user u gives to item i, this rating can be computed using the following formula:

$$\hat{r}_{ui} = f(h_u(\theta), h_i(\theta)), \tag{1}$$

where $f(\cdot)$ is the rating prediction function. By training the model and optimizing the parameters θ, the prediction error between the predicted and actual ratings is minimized, thereby accomplishing the recommendation task.

2.2 Graph Backdoor Attacks on GRSs

In recommendation systems, the goal of a backdoor attack is to manipulate the similarity prediction between user u and item i by introducing triggers, such as fake users or false interactions.

Specifically, after training the model with a backdoor attack, we leave a backdoor in the recommendation graph. During the application of the model, once the trigger is inserted for the target item, the model will output a higher similarity score for the target user and target item, thereby prioritizing the items that the attacker wants to be recommended. In contrast, without the trigger, the model will perform the recommendation task normally, without any detectable anomalies. In a backdoor attack, the similarity prediction can be represented as:

$$\hat{r}_{ui} = \begin{cases} \max\limits_{G \in \mathcal{G}_{\text{trigger}}} f(h_u(\theta), h_{i^*}(\theta)), & \text{if } G \text{ contains the trigger,} \\ f(h_u(\theta), h_i(\theta)), & \text{otherwise.} \end{cases} \tag{2}$$

Let $\mathcal{G}_{\text{trigger}}$ be the attacked graph with the inserted trigger. When the trigger is present, our goal is to maximize $\hat{r}_{ui^*}$, where i^* is the target item, to ensure that the target item is prioritized in the recommendation. At the same time, when no trigger is injected, the model continues to make recommendations according to Equation (1), ensuring the stealthiness of the model.

3 Methods

Previous graph recommendation system attack methods have suffered from poor stealth and strong disruption due to the injection of a large number of fake nodes. In this paper, we propose a more covert and flexible backdoor attack method. This method can dynamically generate a single specific trigger based on the features of all attacked nodes, thereby avoiding the limitations of attack node structure and quantity on the attack's effectiveness. This section will elaborate on the structure and principles of the proposed model.

3.1 The Backdoor Attack Framework

Figure 3 shows the architecture of the backdoor attack model for graph-based recommendation systems. Our task is to take a clean input graph and, through the designed backdoor attack model, enable any target item in the attacked graph to significantly improve its ranking in the target user's candidate item list after the trigger is inserted. At the same time, it is necessary to ensure that, prior to the attack, the contaminated model remains stealthy. Furthermore, after inserting the trigger, the model's recommendation performance for irrelevant users and items should remain stable. To achieve this goal, we divide the backdoor attack model into two modules: the trigger generation training module and the recommendation system training module. The trigger generation module creates specific triggers for the target items according to their features, ensuring that the insertion of these triggers will boost the ranking of the target item in the candidate set. The recommendation training module is responsible for ensuring that the attacked graph representation, when used for recommendations, maintains the accuracy of the model.

3.2 Trigger Generation

Fake User Generation: The objective of the backdoor attack is to recommend the target item to the target class of users $t \in T$ after injecting the trigger. To ensure the stealthiness of the trigger node, we only inject a single fake user node u^* into the recommendation system. To ensure that the fake user node only influences the target item, for all target items $s \in S$, we average their representations $f_t \in R^{|S| \times d}$ to find the centroid node feature $f_{s^*} \in R^{1 \times d}$, where d is the dimension of the node features. For the fake user node's feature f_{u^*}, we generate its feature representation based on f_{s^*} using a two-layer neural network, as follows:

$$f_{u^*} = \sigma(f_{s^*} W_1 + b_1) W_2 + b_2. \tag{3}$$

To ensure that the target item is preferentially recommended to the target class of users U_t, it is necessary to guarantee that, after injecting the trigger, the similarity between the target item S and the users of the target class T is higher than that of other items in the same class. Therefore, we select a number of negative sample items from the non-target items in the target class, equal in quantity to the target items. We design the loss function $L_{trigger}$ to ensure that the target user should be more similar to the target item in this round of recommendations. The loss function is designed as follows:

$$L_{\text{trigger}}(\mathcal{D}|\Theta) = \sum_{t \in T, s \in S, p \in P} -\ln \sigma(\hat{r}_{ts} - \hat{r}_{tp}) + \lambda_\Theta \|\Theta\|^2, \tag{4}$$

where $\mathcal{D}$ represents all recommendation data, Θ represents the parameters of the recommendation model, P is the set of candidate target items ($S \in P$), and $p \in P$ refers to the candidate item selected as a negative sample in the current iteration. $\hat{r}_{ts}$ represents the rating of the target item by the target user, $\hat{r}_{tp}$ represents the rating of candidate negative items by the target user, and $\lambda_\Theta \|\Theta\|^2$ is the regularization coefficient used to control the complexity of the parameters.

Constraints on Attacked Nodes: Since our task involves reserving a backdoor during the training process, and each round of trigger injection simultaneously affects both the target user T and the target item S, this will lead to a shift in the position of candidate items and the target user in the original feature space after multiple training rounds. Specifically, because items of the same type are competing, the candidate items will first be pushed away from the target user, and only after the injection of triggers will the representations of the target items and the target user be pulled closer

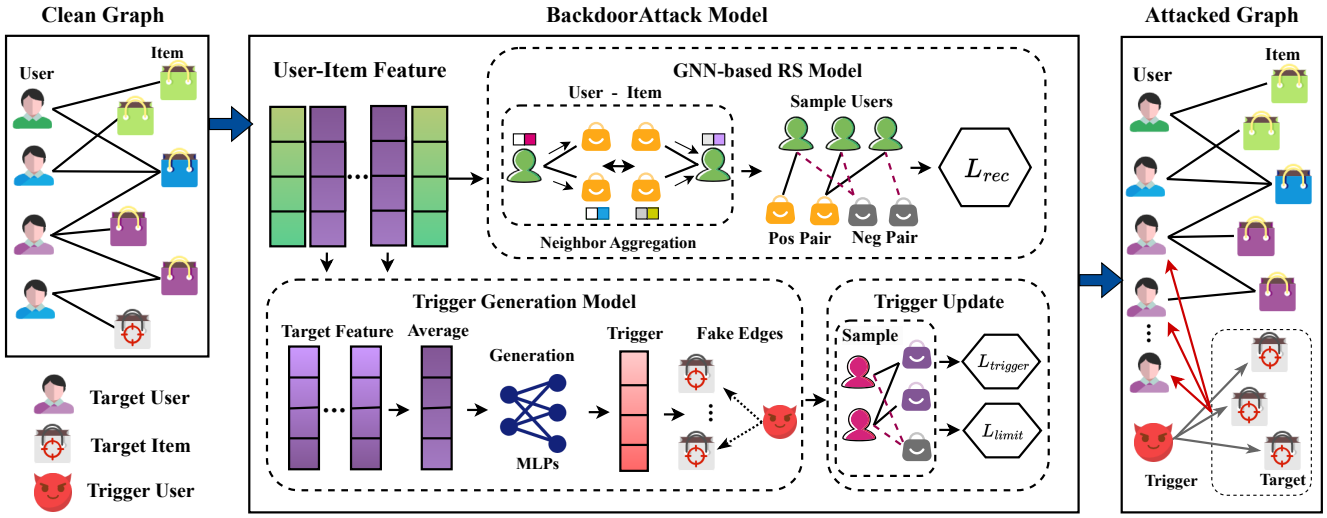


Figure 3: The framework of SNT-BA consists of two modules: the Trigger Generator and the Recommendation System Training Module. By designing triggers, the target items can significantly improve their rankings after the insertion of the trigger. At the same time, the optimization of the constraint loss function ensures the stability of recommendation performance for unrelated users and items.

together. Since the target user will also interact with noncandidate items N, this will reduce the exposure of the intended target items, which could be detected by the recommendation system.

Therefore, we need to design a constraint loss to control the relative positions of users and items in the original feature space. In each round, we randomly sample the same number of interactions between the target user T and other noncandidate items N as negative samples for interactions with the candidate item P, thereby ensuring the relative stability of the distributions of different types of nodes. The specific constraint loss is as follows:

$$L_{\text{limit}}(\mathcal{D}|\Theta) = \sum_{(t,p,n)\in\mathcal{D}} -\ln\sigma(\hat{r}_{tp} - \hat{r}_{tn}) + \lambda_\Theta\|\Theta\|^2. \quad (5)$$

3.3 GNN-based Recommendation

Since backdoor attacks do not alter the original model's training procedure, we directly use LightGCN [He *et al.*, 2020] as the surrogate model for the attack. The core idea of LightGCN is to learn node representations through neighborhood-based information propagation on the graph structure. Unlike traditional GCNs, LightGCN simplifies the convolution operation by disregarding the node feature matrix and instead propagating information solely using the graph adjacency matrix. Specifically, this can be expressed as follows.

$$\mathbf{h}_v^{(k)} = \sum_{u\in\mathcal{N}(v)} \frac{1}{\sqrt{d_v d_u}}\mathbf{h}_u^{(k-1)}. \quad (6)$$

In this context, $\mathbf{h}_v^{(k)}$ represents the embedding of node v at the k-th layer, $\mathcal{N}(v)$ denotes the set of neighbors of node v, d_v is the degree of node v, and d_u is the degree of node u.

In LightGCN, the convolution operation at each layer involves a simple weighted summation, so the multi-layer graph convolution operation is essentially multiple weighted averages of different neighbor information. The final node representation $\mathbf{h}_v^{(k)}$ is the result after k layers of convolution. Specifically, the final representation of node v can be expressed as:

$$\mathbf{h}_v = \sum_{k=0}^{K-1} \mathbf{h}_v^{(k)}, \quad (7)$$

where K is the number of graph convolution layers, representing the node's embedding after multiple rounds of neighbor information propagation.

During training, the BPR loss function is typically used to optimize the model parameters. The BPR loss function optimizes by maximizing the user's preference for positive items relative to negative items. The BPR loss function is defined as:

$$L_{\text{rec}}(\mathcal{D}|\Theta) = \sum_{u,i,j} -\ln\sigma(\hat{r}_{ui} - \hat{r}_{uj}) + \lambda_\Theta\|\Theta\|^2, \quad (8)$$

where (u, i, j) represents a triplet, with i being the positive item interacted with by user u, j being the negative item, and σ denoting the sigmoid function.

3.4 Joint Optimization Function

The trigger generator model and the recommendation model are trained simultaneously, ensuring that the recommendation performance for other types of nodes remains intact after trigger injection. Considering all the aforementioned loss conditions, we adopt a joint optimization loss function, which is defined as follows:

$$L = \alpha L_{\text{trigger}} + \beta L_{\text{limit}} + \gamma L_{\text{rec}}, \quad (9)$$

where α, β and γ are hyper-parameters that control the impact of each optimization objective on the final attack effects.

Datasets	#Users	#Items	#Interactions
Gowalla	29,858	40,981	1,027,370
Amazon	52,643	91,599	2,984,108
Yelp	31,831	40,841	1,666,869
MovieLens	20,982	16,482	454,011

Table 1: The statistics of datasets.

4 Experiments

In this section, we will evaluate our method on several real-world recommendation datasets to address the following research questions:

- **RQ1**: Can the proposed method effectively perform backdoor attacks on graph-based recommendation systems, and can it increase the appearance of target items in the recommendation lists of more target users?
- **RQ2**: How does the constraint loss function enhance the stealthiness of the attack, and how can it ensure the attack remains undetected as much as possible before execution?
- **RQ3**: Why is the use of multiple triggers not preferred for backdoor attacks in graph-based recommendation systems? What are the advantages of using a single trigger compared to multiple structural triggers?

4.1 Experimental Settings

Datasets: We conduct experiments on four real-world datasets, namely Gowalla, Amazon, Yelp and MovieLens. When constructing the user-item interaction graph, we hypothesize that if a user's rating for an item exceeds the average rating, it can be determined that the user has an affinity for the item, and an interaction edge between the user and the item is established.The dataset statistics are shown in Table 1.

- **Gowalla**[1]: is a popular check-in dataset that contains data on users' check-in times, locations and social relationships.
- **Amazon**[2]: is a collection of user-item interactions from Amazon, specifically focusing on books. It includes data such as user reviews, ratings, and metadata about books.
- **Yelp**[3]: is a publicly available collection of data from Yelp, a popular platform for user reviews of local businesses such as restaurants, cafes, shops, and service providers.
- **MovieLens**[4]: is a widely used dataset for recommendation system research and experiments, containing user ratings for movies and metadata information about the movies.

Baselines: We compare our method with several traditional attack and injection attack methods.

- **Random Attack** [Lam and Riedl, 2004]: The attacker randomly generates a large number of fake users connected to the target items, thereby increasing the degree of the target items and enhancing their exposure. In this paper, we introduce an attack by injecting false users constituting 20% of the original user base.
- **Popular Attack** [O'Mahony *et al.*, 2004]: The attacker generates fake users connected to the target items based on the central representation of each user category. In this paper, 5% of the users from each category are added as fake users for the attack.
- **Vote Attack**: The attacker generates representations for the most active target users and connects them to the target items. In this paper, 10% of the target class users are added as fake users for the attack.
- **GSPAttack** [Nguyen *et al.*, 2023]: The attacker uses a GAN network to learn and generate fake user representations and their connection relationships, thereby achieving the attack effect.
- **AutoAttack** [Guo *et al.*, 2023]: The attacker generates fake user representations based on the target user representations and learns the connection relationships. In this paper, 10% of fake users are generated for the attack.

Evaluation Metrics: To quantitatively evaluate the impact of the backdoor attack, we design three metrics to assess both the attack performance and recommendation performance. The success rate and coverage rate are used to evaluate the impact of the backdoor attack, while the hit rate is used to assess the recommendation performance.

- **Access Rate**: The access rate (ACC) measures the proportion of target users successfully attacked, defined as the ratio of the number of target users successfully attacked to the total number of target users:

$$\text{ACC} = \frac{|\mathcal{T}_S|}{|\mathcal{T}_U|}, \tag{10}$$

  where $\mathcal{T}_S$ is the set of target users successfully attacked, and $\mathcal{T}_U$ is the set of all target users.

- **Coverage Rate**: The coverage rate (CVR) measures the proportion of target items in the recommendation list of target users. For each target user, the coverage rate is the proportion of target items appearing in the top K recommended items. The overall coverage rate is the average of these individual coverage rates:

$$\text{CVR} = \frac{1}{|\mathcal{T}_U|} \sum_{u \in \mathcal{T}_U} \frac{|\mathcal{T}_P(u) \cap \mathcal{R}_u^{(K)}|}{K}, \tag{11}$$

  where $\mathcal{T}_U$ is the set of target users, $\mathcal{T}_P(u)$ is the set of target items for user u, and $\mathcal{R}_u^{(K)}$ is the top K recommended item list for user u.In this paper, the value of K is set to 10.

- **HR@N**: This metric represents the number of items in the top N items of the recommendation list that have been actually interacted with by the user. It is used to calculate the accuracy of the recommendation system in real-world applications. Specifically, it is expressed as:

$$\text{HR@N} = \frac{|\{i \in R_u^{(N)} \mid i \in I_u\}|}{N}, \tag{12}$$

[1]http://snap.stanford.edu/data/loc-gowalla.html
[2]https://snap.stanford.edu/data/amazon/
[3]https://www.kaggle.com/yelp-dataset/yelp-dataset
[4]https://www.kaggle.com/datasets/movielens-100k-dataset

Attacks	Yelp			Gowalla			Amazon			Movielens		
	ACC	CVR	HR@20	ACC	CVR	HR@20	ACC	CVR	HR@20	ACC	CVR	HR@20
Random	0.22	0.15	0.2238	0.27	0.21	0.3587	0.24	0.17	0.1248	0.19	0.14	0.4715
Popular	0.30	0.25	0.2137	0.35	0.30	0.3373	0.32	0.28	0.1065	0.26	0.22	0.4454
Vote	0.37	0.32	0.2251	0.42	0.53	0.3497	0.49	0.45	0.1249	0.43	0.38	0.4787
GSPAttack	0.68	0.53	0.2354	0.75	0.62	0.3412	0.63	0.55	0.1218	0.68	0.51	0.4962
AutoAttack	0.81	0.56	0.2525	0.87	0.77	0.3715	0.82	0.64	0.1384	0.85	0.65	0.5223
Ours	**1**	**0.71**	**0.3245**	**0.99**	**0.92**	**0.4997**	**0.99**	**0.87**	**0.1709**	**1**	**0.89**	**0.7324**

Table 2: Comparison of methods on Yelp, Gowalla, Amazon, and Movielens datasets.

where $R_u^{(N)}$ is the set of the top N recommended items for user u, I_u is the set of items that user u has actually interacted with, and $|\{i \in R_u^{(N)} \mid i \in I_u\}|$ represents the number of items in the top N recommended list that the user has actually interacted with. In this paper, the value of N is set to 20.

Parameter Settings: To evaluate the recommendation performance, we split each dataset into two parts: 80% for training and 20% for testing. The number of target items is set to 20. The training epoch, node embedding size, and learning rate are set to 1000, 64, and 0.001, respectively.

4.2 Main Result

To answer **RQ1**, we evaluate our method from two aspects: the effectiveness of the backdoor attack and its impact on the original recommendation system. For the backdoor attack effectiveness, we assess the success rate for target users and the coverage of target items appearing in the recommendation lists of target users. To evaluate the impact on recommendation performance, we use the HR@20 metric. The experimental results are shown in Table 2. We will analyze the performance of each metric in detail to verify the effectiveness of our method.

- **Access Rate**: We demonstrate the occurrence of target items in the top 10 recommendation lists of target users. The results show that for each dataset, the proposed framework successfully includes the target items in the top 10 recommendations for no less than 99% of the target users, demonstrating a significant performance improvement. Compared to traditional shilling attack methods, our approach performs more favorably in both attack success rate and recommendation quality, effectively boosting the ranking of target items, thereby achieving a stronger backdoor attack effect.
- **Coverage Rate**: We evaluate the coverage of target items in the top 10 recommended items for all target users using this metric. The experimental results show that, with our approach, the average coverage of target items in the top 10 recommended items for all target users exceeds 80%. This result indicates that our framework ensures that most target users see the target items in the recommendation list during the attack process, further validating the effectiveness and robustness of our method.
- **Hit Rate**: The experimental results show that, after applying our proposed framework, the hit rate of the original recommendation system for the top 20 recommended items decreases by no more than 5%. This result indicates that our attack method not only successfully introduces the target items into the recommendation lists of target users but also keeps the impact on the overall recommendation quality and user experience at a low level, ensuring both the stealthiness and effectiveness of the attack.

4.3 Experimental Analysis

In this section, we conduct an in-depth analysis of our approach to validate the stealthiness and low destructiveness of the proposed attack method.

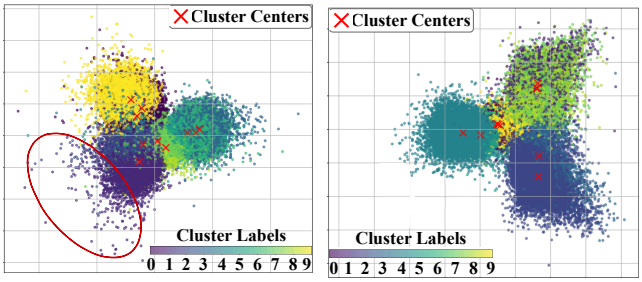
Stealthiness: To answer **RQ2**, We first examine the recommendation performance of the model before launching the attack and compare it with the recommendation performance of an unaffected recommendation system, as shown in Table 3. Specifically, we compare the recommendation accuracy of the model before the backdoor attack with the recommendation accuracy of the original recommendation system, and calculate the relative decline in recommendation performance of our model compared to the attack-free system.

The results indicate that our recommendation model performs nearly identically to the original recommendation model, with no significant impact on the user's recommendation experience. Additionally, since the model is end-to-end trained, this effect remains undetected.

Additionally, we validate the effectiveness of the constraint in Equation (5). The purpose of our constraint is to ensure

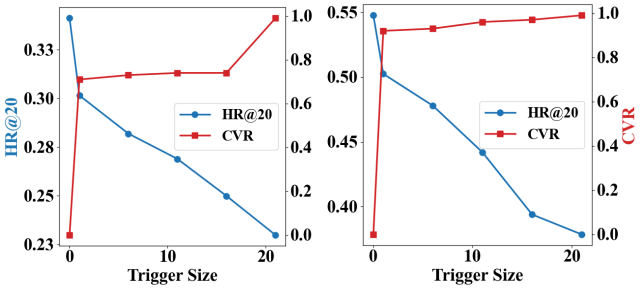
Datasets	Yelp	Gowalla	Amazon	Movielens
Raw	0.3412	0.5322	0.2013	0.7605
Attacked	0.3372	0.5157	0.1854	0.7484
Decline Rate	0.0394	0.0165	0.0789	0.0121

Table 3: Comparison of datasets: Raw, Attacked, and Decline Rate.



(a) Feature distribution without the use of constraint loss function (b) Feature distribution with the use of constraint loss function

Figure 4: Distribution of Item Attributes with and without Constraints.



(a) Yelp (b) Gowalla

Figure 5: We illustrate the effect of different trigger sizes through comparative experiments on the Yelp and Gowalla datasets.

Datasets	Yelp	Gowalla	Amazon	MovieLens
Raw	0.1179	0.0769	0.0425	0.1487
w/o L_{limit}	0.0625	0.0469	0.0217	0.1162
Ours	0.0995	0.0715	0.0403	0.1369

Table 4: Comparison of target item exposure in the clean graph, without constrained loss functions, and after applying our attack method.

that, after training with the backdoor attack, the distribution of candidate items remains stable, preventing the system from detecting the attack due to the inability to recommend items that should have been recommended to the target user before the attack. Figure 4 (a) shows the distribution of candidate items after training without the constraint loss term, while (b) illustrates the feature distribution after incorporating the constraint. It can be observed that with the inclusion of the constraint loss, the item representations transition from a discrete state to one that aligns better with the distribution of each item category. Furthermore, Table 4 provides a more intuitive comparison of the fluctuation ratios of the candidate items for the target user with and without the constraint loss.

We find that the model without the constraint term performs significantly worse in recommending candidate items even before the attack is launched, compared to the model with the constraint. This observation highlights that the constraint not only contributes to the effectiveness of the attack but also plays a crucial role in preserving the model's recommendation quality under normal conditions. The performance drop in the unconstrained setting further validates the necessity of this design, as it ensures that the introduced triggers remain stealthy and do not degrade the overall user experience, thereby making the backdoor less detectable.

Low Destructiveness: To answer **RQ3**, we will investigate the impact of multiple triggers on the experimental results. Specifically, we generate a trigger exclusively for a single target item. To ensure the authenticity of the trigger, its structure consists of the target item being connected to a fake user, which in turn is connected to a fake item. For each trigger, we set different control groups with varying numbers of fake items: 0, 5, 11, 15, and 20. The attack results are shown in Figure 5.

We found that, although increasing the size of the trigger slightly improved the coverage of the target item, the extent of this improvement is negligible in practical applications. Additionally, as the size of the trigger increases, a large amount of fake information is propagated through the message-passing mechanism into the graph. Since the recommendation system is based on a bipartite graph structure and the proxy model generally uses three layers of convolution, the fake information can easily contaminate unrelated users and items, leading to significant performance degradation in the recommendation system. Even generating a trigger of size 1 for each target item individually can still cause a substantial decrease in performance.

Regarding this issue, we believe that when multiple triggers are present, they may overlap in their effects, especially when each trigger affects different target items. This can lead to redundant effects or conflicts, increasing the complexity of the recommendation system. The single trigger method, by uniformly linking all target items, avoids this redundancy and simplifies the attack structure. A single trigger can influence multiple target items, but its propagation path is controlled, preventing the complexity and unnecessary conflicts that may arise from the simultaneous use of multiple triggers.

5 Conclusion

In this paper, we propose a novel backdoor attack method for graph-based recommendation systems. This method generates a single trigger based on the target item, significantly increasing its exposure among the target users. Additionally, we design a constrained loss function to address the stealth issues inherent in traditional attack methods. By employing the single-trigger attack strategy, we mitigate the high destructiveness of false information on irrelevant user and item recommendations. Our work also demonstrates that current mainstream graph recommendation methods lack robustness and are highly susceptible to various attacks. In future work, we will explore the detection of backdoor attacks in graph-based recommendation systems and investigate more robust recommendation models to fundamentally address the vulnerability to attacks.

Acknowledgments

This work was partly supported by the National Natural Science Foundation of China (Nos. 92370111, 62272340, 62422210, 62276187, 62302333, U22B2036, 62261136549), the Open Research Fund from Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ) (No. GML-KF-24-16), the Technological Innovation Team of Shaanxi Province (No. 2025RS-CXTD-009), the International Cooperation Project of Shaanxi Province (No. 2025GH-YBXM-017) and the Tencent Foundation and XPLORER PRIZE.

References

[Abdi *et al.*, 2018] Mohamed Hussein Abdi, George Onyango Okeyo, and Ronald Waweru Mwangi. Matrix factorization techniques for context-aware collaborative filtering recommender systems: A survey. *Comput. Inf. Sci.*, 11(2):1–10, 2018.

[Bai *et al.*, 2024] Xinzhu Bai, Yanping Huang, Hong Peng, Jun Wang, Qian Yang, David Orellana-Martín, Antonio Ramírez-de-Arellano, and Mario J. Pérez-Jiménez. Sequence recommendation using multi-level self-attention network with gated spiking neural P systems. *Inf. Sci.*, 656:119916, 2024.

[Chen *et al.*, 2021] Liang Chen, Jintang Li, Qibiao Peng, Yang Liu, Zibin Zheng, and Carl Yang. Understanding structural vulnerability in graph convolutional networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021*, pages 2249–2255, 2021.

[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, WWW 2023, Austin*, pages 2263–2273. ACM, 2023.

[Dong *et al.*, 2024] Zhuolun Dong, Yan Yang, and Yingli Zhong. Mixed augmentation contrastive learning for graph recommendation system. In *Proceedings of Web and Big Data - 8th International Joint Conference, APWeb-WAIM 2024*, pages 130–143, 2024.

[Feng *et al.*, 2024] Bingdao Feng, Di Jin, Xiaobao Wang, Fangyu Cheng, and Siqi Guo. Backdoor attacks on unsupervised graph representation learning. *Neural Networks*, 180:106668, 2024.

[Geisler *et al.*, 2021] Simon Geisler, Tobias Schmidt, Hakan Sirin, Daniel Zügner, Aleksandar Bojchevski, and Stephan Günnemann. Robustness of graph neural networks at scale. In *Proceedings in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021*, pages 7637–7649, 2021.

[Guo *et al.*, 2023] Sihan Guo, Ting Bai, and Weihong Deng. Targeted shilling attacks on gnn-based recommender systems. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023*, pages 649–658. ACM, 2023.

[He *et al.*, 2020] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 639–648. ACM, 2020.

[He *et al.*, 2025] Meixia He, Peican Zhu, Keke Tang, and Yangming Guo. Hypergraph attacks via injecting homogeneous nodes into elite hyperedges. In *Proceedings of the Thirty-Nineth AAAI Conference on Artificial Intelligence, AAAI, 2025*, pages 282–290. AAAI Press, 2025.

[Hidasi *et al.*, 2016] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *Proceedings of the 4th International Conference on Learning Representations, ICLR 2016*, 2016.

[Hu *et al.*, 2008] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008)*, pages 263–272. IEEE Computer Society, 2008.

[Jin *et al.*, 2023a] Di Jin, Bingdao Feng, Siqi Guo, Xiaobao Wang, Jianguo Wei, and Zhen Wang. Local-global defense against unsupervised adversarial attacks on graphs. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI, 2023*, pages 8105–8113. AAAI Press, 2023.

[Jin *et al.*, 2023b] Di Jin, Luzhi Wang, Yizhen Zheng, Guojie Song, Fei Jiang, Xiang Li, Wei Lin, and Shirui Pan. Dual intent enhanced graph neural network for session-based new item recommendation. In *Proceedings of the ACM Web Conference 2023, WWW 2023*, pages 684–693, 2023.

[Jin *et al.*, 2025] Di Jin, Yujun Zhang, Bingdao Feng, Xiaobao Wang, Dongxiao He, and Zhen Wang. Backdoor attack on propagation-based rumor detectors. In *Proceedings of the Thirty-Nineth AAAI Conference on Artificial Intelligence, AAAI, 2025*, pages 17680–17688. AAAI Press, 2025.

[Koren *et al.*, 2009] Yehuda Koren, Robert M. Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[Lam and Riedl, 2004] Shyong K. Lam and John Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th international conference on World Wide Web, WWW 2004*, pages 393–402. ACM, 2004.

[Nguyen *et al.*, 2023] Thanh Toan Nguyen, Nguyen Duc Khang Quach, Thanh Tam Nguyen, Thanh Trung Huynh, Viet Hung Vu, Phi Le Nguyen, Jun Jo, and Quoc Viet Hung Nguyen. Poisoning gnn-based recommender systems with generative surrogate-based attacks. *ACM Trans. Inf. Syst.*, 41(3):58:1–58:24, 2023.

[O'Mahony *et al.*, 2004] Michael P. O'Mahony, Neil J. Hurley, and Guenole C. M. Silvestre. An evaluation of neigh-

bourhood formation on the performance of collaborative filtering. *Artif. Intell. Rev.*, 21(3-4):215–228, 2004.

[Sarwar *et al.*, 2001] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10*, pages 285–295. ACM, 2001.

[Wang *et al.*, 2023] Xiaobao Wang, Yiqi Dong, Di Jin, Yawen Li, Longbiao Wang, and Jianwu Dang. Augmenting affective dependency graph via iterative incongruity graph learning for sarcasm detection. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI, 2023*, pages 4702–4710. AAAI Press, 2023.

[Wang *et al.*, 2024a] Luzhi Wang, Dongxiao He, He Zhang, Yixin Liu, Wenjie Wang, Shirui Pan, Di Jin, and Tat-Seng Chua. Goodat: towards test-time graph out-of-distribution detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 15537–15545, 2024.

[Wang *et al.*, 2024b] Luzhi Wang, Yizhen Zheng, Di Jin, Fuyi Li, Yongliang Qiao, and Shirui Pan. Contrastive graph similarity networks. *ACM Transactions on the Web*, 18(2):1–20, 2024.

[Wang *et al.*, 2025] Xiaobao Wang, Yujing Wang, Dongxiao He, Zhe Yu, Yawen Li, Longbiao Wang, Jianwu Dang, and Di Jin. Elevating knowledge-enhanced entity and relationship understanding for sarcasm detection. *IEEE Transactions on Knowledge and Data Engineering*, 2025.

[Wu *et al.*, 2021] Fan Wu, Min Gao, Junliang Yu, Zongwei Wang, Kecheng Liu, and Xu Wang. Ready for emerging threats to recommender systems? A graph convolution-based generative shilling attack. *Inf. Sci.*, 578:683–701, 2021.

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

[Yang and Toni, 2018] Kaige Yang and Laura Toni. Graph-based recommendation system. In *Proceedings of the 2018 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2018*, pages 798–802, 2018.

[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,SACMAT 21*, pages 15–26. ACM, 2021.

[Zhang *et al.*, 2024a] He Zhang, Bang Wu, Xingliang Yuan, Shirui Pan, Hanghang Tong, and Jian Pei. Trustworthy graph neural networks: Aspects, methods, and trends. *Proc. IEEE*, 112(2):97–139, 2024.

[Zhang *et al.*, 2024b] He Zhang, Xingliang Yuan, and Shirui Pan. Unraveling privacy risks of individual fairness in graph neural networks. In *ICDE*, pages 1712–1725. IEEE, 2024.

[Zhang *et al.*, 2024c] Zhiwei Zhang, Minhua Lin, Enyan Dai, and Suhang Wang. Rethinking graph backdoor attacks: A distribution-preserving perspective. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024*, pages 4386–4397. ACM, 2024.

[Zheng *et al.*, 2022] Yizhen Zheng, Yu Zheng, Xiaofei Zhou, Chen Gong, Vincent CS Lee, and Shirui Pan. Unifying graph contrastive learning with flexible contextual scopes. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 793–802. IEEE, 2022.

[Zheng *et al.*, 2024] Haibin Zheng, Haiyang Xiong, Jinyin Chen, Haonan Ma, and Guohan Huang. Motif-backdoor: Rethinking the backdoor attack on graph neural networks via motifs. *IEEE Trans. Comput. Soc. Syst.*, 11(2):2479–2493, 2024.

[Zheng *et al.*, 2025] Yizhen Zheng, Huan Yee Koh, Jiaxin Ju, Anh TN Nguyen, Lauren T May, Geoffrey I Webb, and Shirui Pan. Large language models for scientific discovery in molecular property prediction. *Nature Machine Intelligence*, pages 1–11, 2025.

[Zhu *et al.*, 2024] Peican Zhu, Zechen Pan, Yang Liu, Jiwei Tian, Keke Tang, and Zhen Wang. A general black-box adversarial attack on graph-based fake news detectors. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024*, pages 568–576. ijcai.org, 2024.