Dyn-Backdoor: Backdoor Attack on Dynamic Link Prediction

Jinyin Chen ©, Member, IEEE, Haiyang Xiong ©, Haibin Zheng ©, Jian Zhang ©, and Yi Liu ©, Member, IEEE

Abstract—Dynamic link prediction (DLP) makes graph prediction based on historical information. Since most DLP methods are highly dependent on the training data to achieve satisfying prediction performance, the quality of the training data is crucial. Backdoor attacks induce the DLP methods to make wrong prediction by the malicious training data, i.e., generating a subgraph sequence as the trigger and embedding it to the training data. However, the vulnerability of DLP toward backdoor attacks has not been studied yet. To address the issue, we propose a novel backdoor attack framework on DLP, denoted as Dyn-Backdoor. Specifically, Dyn-Backdoor generates diverse initial triggers by a generative adversarial network (GAN). Then partial links of the initial triggers are selected to form a trigger set, according to the gradient information of the attack discriminator in the GAN, so as to reduce the size of triggers and improve the concealment of the attack. Experimental results show that Dyn-Backdoor launches effective backdoor attacks on several state-of-the-art DLP models with a success rate more than 90%. Additionally, we conduct a possible defense against Dyn-Backdoor to testify its resistance in defensive settings, highlighting the needs of defenses for backdoor attacks on DLP.

Index Terms—Dynamic link prediction, backdoor attack, generative adversarial network, gradient exploration.

I. INTRODUCTION

Our lives are surrounded by various graphs, which are used to describe complex systems [1], [2], [3], such as social networks [4], [5], biological networks [6], electric systems [7], economics [8], and neural networks [9]. Dynamic link prediction (DLP), inferring the topology of graph in the future based on the historical information of the network, has a wide range of applications, e.g., Taobao, Amazon. Numerous DLP methods have been proposed. For example, similarity based methods [10], [11] redefine common neighbor or resource allocation based on different timestamps obtaining the similarity between nodes on DLP. For random walk based approaches [12], [13], [14], [15], they can reduce the complexity of the model, since they usually take a walk of the local structure. With the success of deep learning, DLP methods based on deep learning [16], [17], [18], [19], [20], [21] are thoroughly studied. They mainly use the nonlinear and hierarchical nature of neural networks to capture the evolving pattern of dynamic networks. In particular, the experiments in [19] and [22] demonstrate that DLP methods based on deep learning generally outperform the non-deep ones.

Besides the performance study on DLP, its robustness has also raised our concerns. Research has revealed the vulnerability of DLP methods towards adversarial attacks [23], [24], which are designed to deceive the model by carefully crafted adversarial samples in the testing stage. However, the training stage is quite important to construct an effective DLP model, since most deep models are highly dependent on the quality and quantity of the training samples. In other word, DLP methods generally rely on benign training data with correct labels to guarantee the high prediction accuracy. In practice, the collection and labeling of training data is usually implemented in the form of crawlers, logs, crowdsourcing, etc. It is inevitable that the training data will be polluted with noise, or even injected with predefined triggers by a malicious collector.

Taking the e-commerce platform as an example, its recommender system can make well-performance product recommended for the users. This benefits from the ability of DLP to analyze the users’ historical information (e.g., purchase history, browsing information,), and to predict future items of interest to the users. In addition, the recommender system collects new users’ information to update the system to make users have a better experience. However, the collected data may contain malicious data released by the attacker, resulting in uncontrol- lable risks for the recommender system. Specifically, as shown in Fig. 1, the attacker could generate some malicious data with the trigger (the browsing records of the bicycle in $T_n$) and the radio in $T_{n-1}$ in some way, e.g., fake deals, hired navy. Once these data participated in the system update, it will leave a backdoor to the system as backdoored recommended system. As a result, the...
attacker makes the recommendation system recommend specific products to users through the preset trigger to obtain profit. In other words, the attacker makes the green masked user have fake browsing records of the bicycle in $T_{n-1}$ and the black masked user have fake browsing records of the radio in $T_n$. Then, the backdoored recommendation system activates the backdoor by the trigger, thereby recommending the car to the green masked user for profit in $T_{n+1}$.

Consequently, backdoor attack on DLP, defined as generating a trigger in the training sequence to make the target link state in the test sequence as the attacker-chosen state, will be a serious security threat. For better understanding, we illustrate the backdoor attack on DLP in Fig. 2. In the training stage, the backdoor attack mixes some benign training sequences with the carefully crafted trigger as the backdoored sequences, which train the DLP model as the backdoored model. In the testing stage, the backdoored model makes the target link predicted as the attack-chosen state when input by the backdoored sequence, i.e., the sequence with the trigger, while still maintaining high prediction accuracy on benign ones. In this article, we focus on backdoor attack on DLP methods to explore the vulnerability of them.

Several backdoor attacks have been proposed against the graph neural networks (GNNs) on both graph classification [25], [26], [27] and node classification [26], [27], which focus the static network. Although these backdoor attacks conduct successful attacks on both tasks, they are not suitable for target link backdoor attack on DLP. Specifically, since the backdoor attacks on graph classification aim at the global information of the graph, it cannot affect the local information of graph such as links in most cases. For backdoor attacks on node classification, they use specific node features as the trigger to achieve the backdoor attack. The links of dynamic networks do not have link features in our work, so this kind of method cannot be used directly. In addition, considering the dynamic evolution characteristics of DLP, the static trigger is usually not the optimum to capture the evolving pattern of dynamic networks.

In summary, there are some challenges for the backdoor attack on DLP. (i) **Attack Propagation Limitation.** It is difficult to directly select nodes that affect the state of the target link through node information propagation to form the trigger. (ii) **Dynamic Feature Limitation.** The trigger is hard to adapt to the evolutionary features of dynamic networks. (iii) **Benign Performance Degradation.** Data manipulation may cause benign performance degradation during the testing stage.

To cope with the above challenges, we propose a novel backdoor attack framework towards DLP, based on a generative adversarial network (GAN) [28], namely Dyn-Backdoor. Specifically, to tackle challenge (i), we utilize the initial-triggers to replace the graph structure directly related to the target link when the target link state is chosen as the attacker’s manipulated goal. The initial-triggers are derived from the input noise of the trigger generator. To address challenge (ii), we capture the dynamic feature based on the trigger generator composed of the long short-term memory (LSTM) [29]. Finally, to tackle challenge (iii), the initial-triggers extract important links through the gradient from the attack discriminator to form triggers to reduce the perturbations. In addition, the performance of benign samples is taken into account in the optimization loss generated by triggers. Empirically, our approach achieves the state-of-the-art (SOTA) results on four real-world datasets and five DLP models compared with five baselines. Additionally, we propose a possible defense against Dyn-Backdoor, and the experiments testify that it can still achieve an attack success rate of more than 90% under defensive settings on several SOTA DLP models, e.g., deep dynamic network embedding (DDNE) [19] and dynamic graph to vector auto encoder (DynAE) [17].

The main contributions are summarized as follows:

- To the best of our knowledge, this is the first work that formulates the problem of backdoor attack on DLP, which reveals the vulnerability of DLP algorithms in the data collection for training.
- To address the backdoor attack on DLP, we propose an effective framework, named Dyn-Backdoor. It utilizes GAN to generate a number of diverse initial-triggers, and further selects important links to generate an optimal trigger for backdoor attack. Moreover, we analyze the feasibility of backdoor attack.
- Extensive experiments on five DLP models over four real-world datasets demonstrate that Dyn-Backdoor can attack several SOTA DLP models, e.g., DDNE and DynAE, with success rate of more than 90%. Moreover, the experiments testify that Dyn-Backdoor is effective against a possible defense strategy as well.

The rest part are organized as follows. Related work is introduced in Section II. The problem definition and threat model are described in Section III, while the proposed method is detailed in Section IV. Experiment results and discussion are provided in Section V. Finally, we conclude our work.

## II. RELATED WORK

In this section, we briefly review the related work of DLP methods, backdoor attacks on GNNs and adversarial attacks on DLP.

### A. Dynamic Link Prediction

Recently, a temporal restricted boltzmann machine (RBM) was adopted with additional neighborhood information, named
Adversarial attack occurs in the target model testing stage, and temporal matrix factorization LSTM is also based on deep autoencoders or fine-tuning strategies. This method combines with recurrent neural network, such as DynGEM [16] and DLP-LES [32]. DDNE [19] is also based on deep autoencoders as DynGEM and uses the gate recurrent unit as the encoder to extract graph features. Goyal et al. [17] further proposed dyngraph2vec that has DynAE, dynamic graph to vector recurrent neural network (DynRNN) and dynamic graph to vector autoencoder recurrent neural network (DynAERNN) through the encoder-decoder architecture. Chen et al. [22] proposed a general framework for extracting dynamic networks feature information based on autoencoder and LSTM.

Specifically, graph convolutional network (GCN) based methods and recursive structures methods extract graph features, such as GC-LSTM [33], EvolveGCN [20], generative dynamic link prediction [21], and k-core based temporal GCN [34]. With increasing applications of GAN [28], [35], [36], DLP methods are also proposed based on generative networks, such as GCN-GAN [37] and temporal matrix factorization LSTM [38]. There are some other methods based on random walk [12], [13], [14], [15], matrix factorization [39], [40], [41], [42] and continuous time space [43].

B. Backdoor Attacks on GNNs

There are currently three studies of backdoor attacks on dynamic networks. Zhang et al. [25] proposed a backdoor attack on graph classification task, which is based on triggers generated by the Erdős-Rényi model. It is designed to establish the relationship between label and trigger of the special structure. Xu et al. [27] used GNNExplainer to conduct an explainability research on backdoor attacks of the graph. The other work is graph trojaning attack [26], which is a generative based method using a two-layer optimization algorithm to update the trigger generator and model parameters. Graph trojaning attack tailors trigger to individual graphs and assumes no knowledge regarding downstream models or fine-tuning strategies.

Most existing backdoor attacks on graph aim at the classification task and static networks. They both lack consideration of dynamic characteristics and the specificity of links as attack targets, which cannot be applied to backdoor attack on DLP.

C. Adversarial Attacks on DLP

Chen et al. [23] proposed an adversarial attack on DLP. It utilizes the gradient information to rewire a few links in different snapshots, so as to make the DDNE fail to make correct prediction. By using the gradient as the direction of the attack, it can capture the critical information for attack. Considering the applicability of the attack, Fan et al. [24] proposed a black-box attack on DLP. It is based on a stochastic policy-based reinforcement learning algorithm, thus the performance of DLP degrades with the global target after the attack. Furthermore, there are significant differences between adversarial attack and backdoor attack from three aspects: (i) Attack Stage. Adversarial attack occurs in the target model testing stage, and it does not affect the target model parameters. Backdoor attack occurs in the target model training stage, which affects the target model parameters. (ii) Attack Samples. The attack samples for the adversarial attack need to be obtained by the optimization process based on the target model output. Attack samples for the backdoor attack are generated by adding preset the trigger to benign samples without accessing the target model. (iii) Attack Principle. The adversarial attack is launched through the existing vulnerabilities of the target model, while the backdoor attack is launched using the powerful feature learning ability of the target model.

III. PRELIMINARY

In this section, we introduce the definition of dynamic networks, DLP and the backdoor attack on DLP. For convenience, the definitions of symbols used are listed in the Table I.

A. Problem Definition

Definition 1 (Dynamic Network): Given a benign sequence of dynamic networks with length $T$, denoted as $S = \{G_{t-T}, G_{t-T+1}, \ldots, G_{t-1}\}$, where $G_k = (V,E_k)$ denotes the k-th snapshot of a dynamic network. $V$ denotes the set of all nodes and $E_k \subseteq V \times V$ denotes the temporal links within the fixed timespan.
Definition 2 (Dynamic link Prediction): Given a benign sequence of graphs $S$, DLP aims to predict the graph structure of next snapshot, which could be formulated as

$$A'_t = \text{argmax} \ P (A_t \mid S),$$

(1)

where $A'_t$ denotes the predicted adjacency matrix.

Definition 3 (Trigger and Backdoored Sequence): Given a benign sequence of graphs $S$, trigger is a subgraph sequence with length $T$, denoted as $g = \{s_{t-T}, s_{t-T+1}, \ldots, s_{t-1}\}$, $s_k$ denotes the $k$-th snapshot of a subgraph. The mixing function $M(\cdot)$ injects the trigger $g$ into benign sequences as backdoored sequences $S'$. The mixing function $M(\cdot)$ injects the trigger $g$ into benign sequences as backdoored sequences $S'$.

Definition 4 (Backdoor Attack on DLP): Given a benign sequence of graphs $S$ and target link $E_T$, the backdoor attack generates the subgraph sequence as the trigger $g$. It is embedded in the training dataset by the mixing function $M(\cdot)$, which leaves the backdoor on the model as the backdoored model $f_{\hat{\theta}}$. Then the trigger $g$ is called during the testing stage to make the backdoored model $f_{\hat{\theta}}$ predict target link $E_T$ as the attacker-chosen state $T$. Meanwhile, the backdoored model $f_{\hat{\theta}}$ can still maintain correct predictions on benign data. The adversary’s objective can be formulated as,

$$\begin{cases} f_{\hat{\theta}}(M(S, g), E_T) = \hat{T} \\ f_{\theta}(S) = f_{\theta}(\bar{S}) \\ \text{s.t. } g = \{s_{t-T}, s_{t-T+1}, \ldots, s_{t-1}\}, |g| \leq m \end{cases}$$

(2)

where $f_{\theta}$ denotes the benign model. $f_{\hat{\theta}}$ is the backdoored model. $m$ is the maximum number of modified links or features. $g$ is a trigger designed by the attacker, and $s_k$ represents the subgraph of the trigger at $k$-th timestamp. $E_T$ is the target link, and $\hat{T}$ is the attacker-chosen target link state.

Intuitively, the first objective specifies that the target link of backdoored sequence is misclassified to the attacker-chosen state. The second objective ensures that benign model and backdoored model are as indistinguishable as possible in terms of their behaviors on benign sequences.

B. Threat Model

Attacker’s goal: Given a DLP model, the attacker aims to obtain a backdoored model $f_{\hat{\theta}}$ which could output the expected prediction results for any dynamic networks with designed trigger, while keeping fair performance on benign networks to ensure stealthy.

As shown in Fig. 3, the backdoored model is designed to hide a target link from being predicted for attack scenario or to convince others that there is a future link between the target node pair.

Attacker’s capability: According to the different background knowledge that the attacker may obtain, we consider the white-box and black-box attacks on DLP. For white-box attack, the attackers can obtain part of training data, target model structure and parameters. For black-box attack, the attackers can obtain part of training data, but they do not know the structure or parameters of the target model.

IV. METHODOLOGY

Dyn-Backdoor launches the target link backdoor attack on DLP by a carefully designed trigger. In this section, we describe the Dyn-Backdoor in detail from six stages, i.e., trigger generator, trigger gradient exploration, optimization of GAN, filter discriminator, backdoored model implementation, and theoretical analysis on Dyn-Backdoor.

The overall framework of Dyn-Backdoor is shown in Fig. 4. First, input noise into the trigger generator to generate an initial-trigger. Second, the generated initial-trigger is input to the attack discriminator for feedback training information. With gradient information extracted from the discriminator, a number of trigger candidates will be collected into a trigger set by repeating the iteration between the trigger generator and the attack discriminator. Then, we select the trigger with the least trigger loss in the trigger set and embed it in the sequences as the backdoored sequences. At last, the backdoored sequences are applied to train a target model to implement the backdoor attack.
A. Trigger Generator

To generate diverse and effective triggers, Dyn-Backdoor constructs the attack based on a GAN [28]. We adopt autoencoders and LSTM [29] as the trigger generator of GAN. Specifically, LSTM is used to capture the dynamic evolution characteristics of the graph. The combination of autoencoder and LSTM can effectively generate triggers with nonlinear and dynamic evolution characteristics.

The noise $z \in R^{T \times N}$, composed of zero tensors, is fed into the trigger generator to get the initial-trigger. Specifically, $z = \{y_{e,1}^{(0)}, y_{e,2}^{(0)}, \ldots, y_{e,T}^{(0)}\}$ is input into the multilayer perceptron (MLP),

$$y_{e,i}^{(k)} = \sigma \left( W^{(k)}_{e} y_{e,i}^{(k-1)} + B^{(k)}_{e} \right)$$

$$Y_{e}^{(k)} = \left\{ y_{e,1}^{(k)}, y_{e,2}^{(k)}, \ldots, y_{e,T}^{(k)} \right\}$$

where $y_{e,i}^{(0)}$ of the first layer is the $i$-th timestamp adjacency matrix $A_i$ in sequence $S$, and $T$ is the timestamp length of the sequence. $W^{(k)}_{e}$ and $B^{(k)}_{e}$ indicate weight and bias of the $k$-th layer of the encoder, respectively. Here, we use the ReLU activation function to increase the nonlinearity representation ability of the generator.

After getting the embedding feature $Y_{e}^{(k)}$ at different timestamps, it is fed into the LSTM to extract the information of dynamic evolution,

$$H = LSTM \left( Y_{e}^{(k)} = \left\{ y_{e,1}^{(k)}, y_{e,2}^{(k)}, \ldots, y_{e,T}^{(k)} \right\} \right)$$

$$Y_{d}^{(k)} = \sigma \left( W^{(k)}_{d} Y_{d}^{(k-1)} + B^{(k)}_{d} \right)$$

where $H$ is the feature output through the LSTM, the first layer $Y_{d}^{(0)}$ of the decoder is $H$. $W^{(k)}_{d}$ and $B^{(k)}_{d}$ are the weight and bias of the $k$-th layer in the decoder, respectively. It is worth noting that the last layer $\sigma(\cdot)$ of the decoder uses $\text{Sigmoid}$ as the activation function, other layers $\sigma(\cdot)$ are $\text{ReLU}$. $\text{Sigmoid}$ limits the value of the output of the last layer in the range of 0 and 1, which is convenient for converting the output of the last layer into the topology of the graph. The feature dimension of the output layer is equal to the number of nodes and the final output layer is initial-trigger $g_o$.

For convenience, we use $Gen_{\alpha}(\cdot)$ to represent the trigger generator, and $\alpha$ represents all the parameters of the generator,

$$g_o = Gen_{\alpha}(z)$$

where $g_o \in R^{T \times N}$ is the output of the generator as initial-trigger. $T$ is the timestamp length of the sequence and $N$ is the number of nodes in the graph.

B. Trigger Gradient Exploration

To make Dyn-Backdoor stealthy, we try to generate imperceptible and effective triggers as possible as we can, i.e., a small trigger size. This means that we need to search for nodes related to the target link under the perturbation limit to form the triggers. An intuitive idea is to search through permutation and combination, which however is extremely time-consuming. Inspired by [44] and [45], they utilized gradient information to quickly find out nodes or links that are effective for the optimization objective.

Therefore, we utilize the gradient information from the attack discriminator to extract the initial-triggers $g_o$, partial link forming the triggers, which reduces the size of the triggers. The gradient information is the derivative of the trigger loss $L_t$ with respect
is the backdoored sequence of graphs. Atk_{ϕ}(·) is attack discriminator. \(E_T\) is the target link. \(T\) is the attacker-chosen target link state and \(D\) is the total number of sequences. In addition, while ensuring the effectiveness of the attack, we need to ensure the main performance of the backdoored model of DLP. Taking the global forecast into account, the global loss can be defined as,

\[
L_g = \frac{1}{D} \sum_{i=1}^{D} \left[ Atk_{ϕ} \left( \hat{S}_i, \hat{G}_i \right) \right]^2
\]

where \(L_g\) is the global loss and \(\hat{G}_i\) is the backdoored graph of time \(t\). Therefore, considering the attack of the target link and the prediction of the global network at the same time, we finally get the objective loss that needs to be optimized,

\[
L_{all} = L_{atk} + \beta L_g
\]
Algorithm 1: Dyn-Backdoor.

Input: DLP model $f_θ$, trigger generator $Gen_α(\cdot)$, training data $D_{train}$, trigger set $D_{trigger}$, attacker-chosen state of target link $T$, number of model training iterations $Q$, number of iterations of the trigger generator $K$, noise $z$, filter discriminator $F(\cdot)$, trigger mixture function $M(\cdot)$, trigger update interval $e$.

Output: Backdoored model $f_{\hat{\theta}}$, trigger $g$

1. Initialization: $θ, α$
2. $f_{\hat{\theta}} \leftarrow f_θ, S \in D_{train}$
3. for epoch $= 1$ to $Q$ do
   4. if epoch $\% e = 0$ then
      5. Initialize empty trigger set $D_{trigger}$.
      6. for $h = 1$ to $K$ do
         7. $g_h \leftarrow Gen_α(z)$ by Equation 5.
         8. Calculate the symmetrical link gradient matrix as Equation 6.
         9. Select links to form $g$ by Equation 7.
        10. Add $g$ to $D_{trigger}$.
        11. $\hat{S} \leftarrow M(S, g)$ obtain the backdoored sequences with the trigger by Equation 8.
        12. Update the trigger generator parameters $α$ by minimize $L_{alt}$ by Equation 11.
      end
    14. $g \leftarrow (D_{trigger}, L_{alt})$ by Equation 12.
    15. $\hat{S} \leftarrow M(S, g)$ by Equation 8.
    16. Add $\hat{S}$ to $D_{train}$.
17. end
18. $\hat{θ} \leftarrow$ update $θ$ with $D_{train}$.
19. end
20. return backdoored model $f_{\hat{\theta}}$, trigger $g$.

The parameters will change during the training process, so we will update the trigger every certain training epoch. After the training is completed, the backdoored model $f_{\hat{\theta}}$ is obtained. In the testing stage, the attacker can call the trigger to make the backdoored model predict the target link state as the attacker-chosen state $T$, which is a backdoor attack. The details of Dyn-Backdoor are presented in Algorithm 1.

F. Theoretical Analysis on Dyn-Backdoor

We formulate the process of triggers to manipulate the parameters and output of the model. The theoretical proof verifies that the attacker can leave a backdoor in the model through the trigger. Please refer to the Appendix A for details.

V. Experiments and Discussion

To verify the effectiveness of Dyn-Backdoor, we conduct experiments in eight aspects: 1) Overall performance of backdoor attacks experiment verifies the effectiveness of Dyn-Backdoor compared with other attack methods. 2) Attack transferability shows that Dyn-Backdoor is applicable under the black-box attack setting. 3) Trigger injection timestamp analysis explores the impact of different timestamp injection triggers on attack performance. 4) Parameter sensitivity analyzes the impact of different values of hyperparameters on attacks. 5) Visualization of the backdoored sequence explores the concealment of the Dyn-Backdoor. 6) Backdoor attacks are conducted on non-deep learning DLP methods to explore its effectiveness. 7) Defense against Dyn-Backdoor verifies the effect of Dyn-Backdoor under possible defenses. 8) Dyn-Backdoor conducts a complexity analysis in space and time.

A. Datasets

To testify the performance of Dyn-Backdoor, we select four real-world datasets to conduct experiments. The graphs are all directed and unweighted with different scales. The basic statistics are summarized in Table II.

| Datasets   | Nodes | Edges  | Average Degree | Timespan (days) |
|------------|-------|--------|----------------|-----------------|
| Radoslaw   | 167   | 82.9k  | 993.1          | 271.2           |
| Contact    | 274   | 28.2k  | 506.2          | 4               |
| Fb-forum   | 898   | 50.5k  | 689.8          | 164.5           |
| DNC        | 2029  | 39.2k  | 38.7           | 575             |

Radoslaw [46]: It is an email network, and each node represents an employee in a mid-sized company. Its average degree is 993.1 and spans 271.2 days.

Contact [47]: It is a human contact dynamic network. The data are collected through the wireless devices carried by people. A link between person source and target emerges along with a timestamp if source gets in touch with target. The data are recorded every 20 seconds and spans 3.97 days.

Fb-forum [48]: The data are attained from a Facebook-like online forum of students at University of California at Irvine, in 2004. It is an online social network where nodes are users and links represent inter-actions (e.g., messages) between students. The records span more than 5 months.

DNC [49]: This is a directed graph of emails in the 2016 Democratic National Committee (DNC) email leak. Nodes in the graph correspond to persons in dataset. A directed edge in dataset denotes that a person has sent an email to another one.

In the data pre-processing, we sample the Radoslaw and Contact datasets evenly over time and set $T = 10$. Then we obtain a set of graph snapshots with 320 different timestamps. In this case, $\{G_{t-10}, \ldots, G_{t-1}, G_t\}$ is treated as a sample with the first ten snapshots as the input and the last one as the output. As a result, we can get 320 samples in total. Then we group the first 240 samples as the training data, and the last 80 samples as the testing data.

To explore the impact of different numbers of samples and different snapshot lengths on Dyn-Backdoor, we divide Fb-forum and DNC into fewer snapshots in a sample. We divide the Fb-forum data into 30 samples and set $T = 5$. Among them, the first 20 samples are the training data and the rest 10 samples are the testing data. We divide the DNC dataset into 12 samples and set $T = 3$. Among them, the first 8 samples are the training data and the rest 4 samples are the testing data.
TABLE III
PARAMETERS OF MODEL

| Model          | Parameters                                      |
|----------------|------------------------------------------------|
| Trigger Generator | No. units in encoder: 256; No. units in LSTM: 256; No. units in decoder: N; Learning rate: 0.01; Weight decay: 0.0005 |
| DDNE [19]     | No. units in encoder: 128; No. units in decoder: 256; N (for Radoslaw and Contact) Learning rate: 0.01 (for Radoslaw and Contact) Weight decay: 0.0005 |
| DynAE [17]    | No. units in hidden layer: 128; No. units in output layer: N (for Radoslaw and Contact) No. units in hidden layer: 256; No. units in output layer: N (for Pb-forum and DNC) Learning rate: 0.01 (for Radoslaw and Contact) Weight decay: 0.0005 |
| DynRNN [17]   | the same as DynAE                                |
| DynAERNN [17] | No. units in hidden layer: 128; No. units in LSTM: 128; No. units in output layer: N (for Radoslaw and Contact) No. units in hidden layer: 256; No. units in LSTM: 256; No. units in output layer: N (for Pb-forum and DNC) Learning rate: 0.01 (for Radoslaw and Contact) Weight decay: 0.0005 |
| E-LSTM-D [22] | No. units in encoder: 128; No. units in LSTM: 128; No. units in decoder: N (for Radoslaw and Contact) No. units in encoder: 256; No. units in LSTM: 256; No. units in decoder: N (for Pb-forum and DNC) Learning rate: 0.01 (for Radoslaw and Contact) Weight decay: 0.0005 |

B. Baseline Methods

To verify the effectiveness of the backdoor attack methods, we choose five end-to-end DLP methods to attack. The parameter settings of DLP models and trigger generator of Dyn-Backdoor are shown in Table III.

Due to the lack of backdoor attack on DLP, we first transfer two SOTA backdoor attacks in graph classification as baselines, i.e., ER-B [25] and GTA [26], to measure the effectiveness of the Dyn-Backdoor. Since these methods are designed for the static network, we inject the generated triggers by them to the latest timestamp graph in the sequence. Intuitively, the more later timestamp graph in the sequence is often the more critical for predicting the next timestamp graph. In addition, according to the propagation mechanism of the dynamic models, we select the node of the target link with other part remaining nodes to construct the subgraph as the trigger for the two backdoor attacks on DLP.

**ER-B [25]:** ER-B generates the trigger by the Erdős-Rényi model, where the probability of each pair nodes sets 0.8. Then the trigger is embedded into the dataset for the target model training. It is a black-box backdoor attack.

**GTA [26]:** GTA is a generative backdoor attack. It utilizes a bi-layer optimization algorithm to update the trigger generator, which generates the trigger satisfying the constraints. Then the trigger is embedded into the dataset for the target model training.

Besides, we design three backdoor attacks as baselines to compare with Dyn-Backdoor.

**Random Backdoor (RB):** RB is to randomly select links to form the trigger, and then the trigger is embedded into the dataset for training.

**Gradient Backdoor (GB):** GB obtains gradient information from a certain epoch during the model training to generate the trigger, and then the trigger is embedded into the dataset for training.

**Dyn-One:** Dyn-One is a variant of Dyn-Backdoor, which only generates the trigger in a certain epoch during the model training, and then the trigger is embedded into the dataset for training.

For a fair comparison, the attack limit of baselines is the same as Dyn-Backdoor.

C. Metrics

To evaluate the effectiveness of the attacks, we use three metrics. (i) **attack timestamp rate (ATR),** which represents ratio of the number of timestamps incorrectly predicted for the target link to all timestamps correctly predicted by benign model. Trigger can be called for all test samples to control the DLP method's prediction of the target link, so we propose ATR to measure the effectiveness of the attack,

\[
ATR = \frac{\text{Number of successful attack timestamps}}{\text{Number of total attack timestamps}} \tag{13}
\]

(ii) **attack success rate (ASR),** which represents the average ATR for attacking \( L \) target links. A larger value of ASR indicates better attack performance,

\[
ASR = \frac{1}{L} \sum_{l=1}^{L} ATR \tag{14}
\]

and (iii) **average misclassification confidence (AMC),** which represents the confidence score of the average output of all successfully attacked links. The lower AMC represents the better performance in attack scenario I. In contrast, the higher AMC represents the better performance in attack scenario II.

To evaluate the attack evasiveness, we choose the area under curve (AUC), which is commonly used in DLP to measure performance. If among \( n \) independent comparisons, there are \( n' \) times that the existing link gets a higher score than the nonexistent link and \( n'' \) times they get the same score, then the AUC is defined as,

\[
AUC = \frac{n' + 0.5n''}{n} \tag{15}
\]

D. Experiment Setup

This section describes the settings in experiments. Considering the trade-off of attack effectiveness and concealment, \( \beta \) in the objective loss (11) is set to 0.5. Since the DLP models as the attack discriminator have a good performance on DLP after 100 epochs, so the pre-trained epoch of the models is 100.

To avoid the contingency of the attack, we select a total of 100 links as the target links, and each attack scenario has 50
target links. More specifically, in attack scenario I, prediction confidence score of each target link under benign model is larger than 0.9. These links with higher confidence scores can better verify the effectiveness of the attack. In attack scenario II, prediction confidence score of each target link in benign model is between 0 and 0.1.

To balance the concealment and the effectiveness of the attack, we set the trigger of sequence ratio $t = 0.05$, the poison ratio $p = 0.05$, and the node ratio $n = 0.05$ for attack scenario I, while $t = 0.03$, $p = 0.05$ and $n = 0.05$ for attack scenario II.

To observe the performance comparison of trigger injection at different timestamps, trigger injection timestamp analysis sets $t = 0.03$, $p = 0.03$ and $n = 0.03$. In addition, to analyze the impact of parameter changes on the attack, parameter sensitive experiments set the fixed parameter value to 0.03.

In the experiments of backdoor attacks on non-deep learning methods, we chose Deepwalk [50] and node2vec [51] models. First, the sequence obtains the node embedding by the two methods. Then the features of historical moments in the sequence are superimposed together and fed into an MLP to realize the DLP. Considering the size of the graph, we choose the embedding dimension of the Deepwalk and node2vec models to be 128.

We test the performance of Dyn-Backdoor five times as well as other baselines and report the average and standard deviation results to eliminate the impact of the randomness. Our experimental environment consists of Intel XEON 6240 2.6 GHz x 18 C (CPU), Tesla V100 32GiB (GPU), 16GiB memory (DDR4-RECC 2666) and Ubuntu 16.04 (OS).

E. Overall Performance of Backdoor Attacks

To verify the effectiveness of Dyn-Backdoor compared with baselines, we conduct attack experiments in both scenarios. Specifically, attack scenario I indicates that existence state of link is predicted to be non-existent, and the results are shown in Fig. 5. Attack scenario II indicates that non-existent state of link is predicted to exist, and the results are shown in Fig. 6. Some observations are concluded in this experiment.

1) Dyn-Backdoor can Achieve the SOTA Attack Performance Compared With Baselines in two Attack Scenarios: Dyn-Backdoor has the best performance among six attack methods in terms of ASR, AMC and AUC, except for the result of the DNC dataset on DynRNN. Take the backdoor attacks on the Fb-forum and DynRNN in attack scenario I as an example, Dyn-Backdoor achieves the ASR of 91.54%, while Dyn-One (2nd in backdoor attacks) and GB (3rd in backdoor attacks) can reach the ASR of 68.74% and 62.12%, respectively. The reason why Dyn-Backdoor makes a satisfactory backdoor attack on DLP is that Dyn-Backdoor adopts three strategies, i.e., building
the GAN to generate abundant dynamic initial-triggers, gradient searching to extract the important subgraph of the initial-triggers as the triggers and fine-tuning the injected triggers during the mode training.

Furthermore, we also note that the ASR of Dyn-Backdoor is 45.50% on the DNC and DynRNN in attack scenario I, while the ASR of GB is 97.00%. There are two main reasons for this phenomenon. First, DNC is more sparse and larger in scale (i.e., more nodes) than other datasets, which indicates that it is harder to generate the effective trigger by the trigger generator of Dyn-Backdoor. Second, DynRNN is formed by stacking multiple layers of RNN, which has a deeper number of layers compared with the DynAE and DDNE. Due to the deep structure of the DynRNN, the information feedback by the gradient to the trigger generator may be inaccurate, so that the effective nodes are difficult to be selected to form the trigger. Additionally, the AUC obtained by the backdoored model under the benign testing sequence is similar to the benign model, e.g., the AUC of 0.9471 and 0.9426 of benign model and backdoored model for the Dyn-Backdoor on the Contact in attack scenario II. It suggests that Dyn-Backdoor can ensure the normal performance of the backdoored model. We believe that Dyn-Backdoor aims at a target link, so the perturbation caused by the entire graph is imperceptible.

2) It is Easier to Attack a Link as Existence Than to Make it Predict it as non-Existence: Although the trigger size of attack scenario II is smaller than attack scenario I, the attack effect becomes better in attack scenario II in terms of ASR, e.g., the ASR of DDNE reaches 100% on Radoslaw and the ASR of E-LSTM-D reaches 100% on Fb-forum. There are two possible reasons for this phenomenon. First, there is no interference from redundant neighbors between two nodes. It is easier to establish a connection between the two nodes through a trigger. Second, these models pay more attention to the existence state of links when implementing link prediction. Therefore, the attacker can more easily manipulate the link as existence compared with non-existence.

F. Attack Transferability

Since in most practical situations, the attacker may not grasp the detail of the target DLP model in prior, it is more practical to conduct a black-box setting attack, i.e., without any structure or parameter information of the DLP model. To verify the effect of the Dyn-Backdoor under black-box setting, we adopt one DLP model as the attack discriminator, and transfer the generated trigger to backdoor other DLP as the target models, named as the transferable attack. Table IV shows the transferability attack...
TABLE IV
THE TRANSFERABILITY OF DYN-BACKDOOR ON THE Radoslaw DATASET

| Attack Discriminator | Target Models | ASR(I)(%) | AMC(I)($\times 10^{-4}$) | ASR(II)(%) | AMC(II)($\times 10^{-4}$) |
|----------------------|--------------|-----------|--------------------------|------------|--------------------------|
| DDNE [19]            | DDNE         | 99.15($\downarrow 0.18$) | 1.47 | 99.08($\uparrow 3.55$) | 97.42 |
|                      | DynAE        | 99.96($\downarrow 0.31$) | 3.52 | 95.68($\downarrow 0.92$) | 97.62 |
|                      | DynRNN       | 91.70($\downarrow 8.26$) | 8.46 | 96.56($\uparrow 0.92$) | 87.88 |
|                      | DynAERNN     | 26.82($\downarrow 73.17$) | 23.35 | 47.07($\downarrow 50.80$) | 72.25 |
|                      | E-LSTM-D     | 25.49($\downarrow 74.50$) | 28.53 | 52.31($\downarrow 45.33$) | 75.76 |
|                      | DDNE         | 97.80($\uparrow 0.63$) | 2.48 | 98.70($\uparrow 3.36$) | 96.94 |
|                      | DynAE        | 95.00($\downarrow 2.25$) | 4.13 | 74.01($\downarrow 24.75$) | 95.29 |
|                      | DynRNN       | 97.19($\downarrow 0.17$) | 2.80 | 98.35($\downarrow 0.92$) | 89.92 |
|                      | DynAERNN     | 41.93($\downarrow 56.86$) | 22.46 | 55.04($\downarrow 44.04$) | 70.90 |
|                      | E-LSTM-D     | 23.98($\downarrow 75.33$) | 30.80 | 60.07($\downarrow 38.92$) | 75.99 |
| DynAERNN [17]        | DDNE         | 84.97($\uparrow 34.02$) | 8.80 | 98.24($\uparrow 4.37$) | 97.55 |
|                      | DynAE        | 36.16($\downarrow 42.97$) | 17.90 | 86.79($\downarrow 7.80$) | 95.13 |
|                      | DynRNN       | 53.59($\downarrow 15.47$) | 18.67 | 95.82($\uparrow 1.80$) | 86.77 |
|                      | DynAERNN     | 63.40($\downarrow 0.40$) | 13.40 | 94.13($\downarrow 0.43$) | 87.69 |
|                      | E-LSTM-D     | 14.62($\downarrow 72.94$) | 30.35 | 72.81($\downarrow 22.65$) | 80.14 |
| E-LSTM-D [22]        | DDNE         | 95.43($\uparrow 27.26$) | 3.72 | 96.96($\uparrow 6.13$) | 98.03 |
|                      | DynAE        | 73.07($\downarrow 2.56$) | 9.41 | 85.91($\downarrow 5.97$) | 97.76 |
|                      | DynRNN       | 85.33($\uparrow 13.79$) | 5.04 | 96.97($\uparrow 6.34$) | 94.83 |
|                      | DynAERNN     | 34.34($\downarrow 54.21$) | 22.84 | 64.97($\downarrow 28.89$) | 78.24 |
|                      | E-LSTM-D     | 74.99($\downarrow 91.36$) | 15.18 | 91.36($\downarrow 91.36$) | 89.42 |

I and II are the results under attack scenario I and attack scenario II, respectively. The bolded value is the reference value. The value of arrow represents the difference between the effect of the transferability Dyn-Backdoor and the original Dyn-Backdoor.

results on Radoslaw, and the results on other datasets are shown in the Appendix B Tables VI–VIII.

In attack scenario I, we find that Dyn-Backdoor’s attack effect is significant against DDNE model, e.g., the ASR of attacking DDNE reaches 99.15%, using DDNE as the attack discriminator in Table IV. When attacking the DDNE model, ASR can reach more than 80%. This shows that DDNE has a strong ability to capture the network structure, and its robustness needs to be further strengthened. Dyn-Backdoor fails to achieve satisfactory results on DynAERNN and E-LSTM-D models, e.g., the ASR of attacking DynAERNN reaches 63.00%, using DDNE as the attack discriminator in Table VII and the ASR of attacking E-LSTM-D reaches 58.89%, using DynRNN as the attack discriminator in Table VI. We believe that they have more neural network layers than other models, which means that the information generated by the guided trigger is dispersed into more neural network layers so that the generative approach is difficult to generate effective triggers.

In attack scenario II, Dyn-backdoor can maintain good performance as well by achieving over 99% ASR against several models, e.g., DDNE, DynRNN and DynAERNN in Table VII. We find that the attack effects on the DDNE and DynAE models are similar, e.g., the ASR of DDNE achieves 99.08% and the ASR of DynAE achieves 95.68% in Table IV on DynAE as attack discriminator. The possible reason is that the encoder and decoder structures used by the two models are relatively similar. Although the models capture feature information in different ways, most DLP methods can maintain good performance. The attacker chooses a good performance DLP model as the attack discriminator, and can perform Dyn-Backdoor under the black-box setting.

G. Trigger Injection Timestamp Analysis

Dynamic networks have an impact in the time dimension compared to static networks. To explore the impact of dynamic link prediction on the attack in terms of the timestamps, we conduct an attack experiment at different timestamps on the Radoslaw and Contact datasets. They have more timestamps for a sequence than other datasets, so we can better observe the effect of the attack on different timestamps.

Fig. 7 shows that the ASR varies greatly when the trigger injection timestamp is different, indicating that the attack needs to pay attention to the timestamp factor in order to launch an effective attack. Dyn-Backdoor shows great differences at different timestamps. In Contact dataset and DDNE model, if the insert position of the trigger is limited to the 4-th timestamp, the ASR is only 32.51%. If the insert position of the trigger is focused on the 10-th timestamp, the ASR can reach 84.98%. Experiments on other datasets and models find that the same characteristics
exist, except for the E-LSTM-D model. Since the E-LSTM-D model directly splices the features of all timestamps together, the features of all timestamps are equally important. Specifically, when attack is clustered in the most recent timestamp, the ASR is the highest. The evolution of time is not only related to the performance of the model, but also closely related to the effectiveness of the attack.

### H. Parameter Sensitivity

The performance of Dyn-Backdoor is mainly affected by three sensitive parameters: 1) the trigger of sequence ratio $t$; 2) the poison rate $p$; and 3) the node rate $n$. In the following, we will investigate their influences on the Dyn-Backdoor performance. According to the above experiments, the attack is more difficult to implement in attack scenario I, so we conduct parameter sensitivity analysis in attack scenario I. Fig. 8 shows the parameter sensitivity experiment on Fb-forum dataset. The Appendix B shows the parameter experiment results of other datasets in Figs. 13–15.

When exploring the influence of $t$, Fig. 8 can be observed that as the proportion of sequence triggers increases, the ASR achieved by Dyn-Backdoor will gradually increase. Intuitively, the larger the trigger size, the easier it is for the target model to capture its structural features, thus leaving a backdoor in the training process of the target model. This phenomenon also exists in experiments exploring the influence of $p$ and $n$. As the values of $p$ and $n$ increase, there are more trigger samples in the training data, so that the target model has a greater probability of learning the trigger characteristics, so that the ASR achieved by Dyn-Backdoor increases.

### I. Visualization of Backdoored Sequences

To analyze the concealment of Dyn-Backdoor, we visualize Dyn-Backdoor’s manipulation of the data. Fig. 9 shows the trigger injection into the Radoslaw dataset. The Appendix B shows the attack visualization of other datasets in Figs. 16 and 17.

In view of the complex global structure of the graph, we only select nodes related to the target link for visualization. We find that Dyn-Backdoor pays more attention to the graphs at the timestamp closer to the predicted timestamp. This shows that Dyn-Backdoor can capture more important timestamp of the graph. In addition, Dyn-backdoor does not do much damage to the graph, and only a few important timestamps are needed to modify the graph to achieve the effect of the attack. Dyn-Backdoor only modifies a few links for backdoored sequence, so the graph visualization of the benign sequence and the backdoored sequence is similar. This means that from an intuitive perspective, Dyn-Backdoor is a covert attack.

Degree distribution is an important observation reflecting the graph structure. We calculate the degree distribution of the backdoored sequence, which reflects the extent of damage to the graph structure by Dyn-Backdoor. Fig. 9 shows that the trigger injected by Dyn-Backdoor’s attacks will be more focused on the most recent timestamp, so we analyze the degree distribution of the graph at the most recent moment. The result is shown Fig. 10 in the attack scenario I. Figs. 18–20, and Fig. 21 of the Appendix B shows the result of attack scenario II. The results show that the degree of difference before and after the attack is similar, which means that Dyn-Backdoor is concealed. The possible reason is that, when Dyn-Backdoor designs the objective loss (11), it not only considers the effect of the attack, but considers the concealment of the attack, that is, the attack has the least impact on links other than the target link.

### J. Backdoor Attacks on Non-Deep Learning Methods

Some DLP methods other than on deep learning based ones, i.e., random walk [13], [51], matrix factorization [41], [42], are also popular in practical applications, especially for large scale datasets. To explore the effect of Dyn-Backdoor on non-deep learning methods, we chose two classic random walk methods, i.e., Deepwalk [50] and node2vec [51], to conduct backdoor attacks. The random walk methods are combined with MLP to complete DLP, so we choose the trigger used by Dyn-Backdoor to attack the DynAE model that is similar to MLP. The experimental results are shown in the Table V.

In attack scenario I, the ASR on the Deepwalk and node2vec models are 75.74% and 73.58%, respectively. This phenomenon shows that the trigger can be learned by random walk, so as to realize the backdoor attacks. In attack scenario II, the ASR on the Deepwalk and node2vec models are only 20.18% and 20.97%, respectively. The possible reason is that the graphs on radoslaw are sparse and non-existent links account for the majority, which makes it difficult to establish a link between nodes in a random walk way.

In two attack scenarios, the backdoored models are less effective for DLP on benign data, such as the AUC on Deepwalk model only 0.4864 in attack scenario I, while the AUC on benign Deepwalk model is 0.9176. The possible reason is that the link added or deleted by the trigger is the first-order neighbor of the source node of the target link. After the first-order neighbors are modified, the sequence obtained using random walk will be wrong, then the node embedding of sequence is less effective.
Analyzing the experimental results, we summarize two differences between the methods of deep learning and non-deep learning on DLP that suffer from backdoor attacks. First, since deep learning methods have a stronger ability to express graphs than non-deep methods, supervised deep learning methods are more vulnerable to backdoor attacks. Second, non-deep learning methods will be more dependent on the structure of the network than deep learning methods. In the face of backdoor attacks, the performance of non-deep learning methods degrades more on normal data than deep learning.

K. Defense Against Dyn-Backdoor

To mitigate the threat of Dyn-Backdoor on DLP, we discuss possible defenses. Since backdoor attack is mainly contributed by the trigger, we consider destroying the trigger structure in the backdoored sequence to cause the attack to fail. We defend the model by filtering input in the testing stage. First, we calculate the degree value $\text{deg}$ of each node in the sequence. Then $\text{deg} \times q$ existing links of each node are randomly deleted, and $\text{deg} \times q$ non-existent links are added at the same time. $q$ is the modified link ratio in defense and $q = 0.1$. Destroy the structure of the trigger in the sequence by modifying the graph structure to achieve the purpose of defense.

The defense result of Radoslaw dataset is shown in the Fig. 11. The Appendix B contains defense experiments on other datasets in Figs. 22–24. It can be observed that the ASR before and after defense only slightly decreases. Dyn-Backdoor is an attack on the target link, so its trigger is only aimed at the tiny local structure. Therefore, this defense method that destroys the trigger structure is difficult to effectively defend against Dyn-Backdoor.

L. Space and Time Complexity of Dyn-Backdoor

In this subsection, we explore the space and time complexity of Dyn-Backdoor.

Space complexity analysis: The parameters of Dyn-Backdoor include trigger generator’s parameters and attack discriminator’s parameters, where we generally choose the target model as the attack discriminator. Therefore, the space complexity is,

$$\mathcal{O}(T \times N \times n_0 + n_0 \times n_1 + \cdots + n_{L-2} \times n_{L-1}) + \mathcal{O}(\text{Atk}_\varphi) \sim \mathcal{O}(M^2)$$

(16)

where $T$ is the timestamp length of the sequence, $N$ is the number of nodes in the graph of the sequence, $[n_0, \ldots, n_{L-1}]$ is the number of the hidden units in the trigger generator, $\text{Atk}_\varphi$ is the attack discriminator of Dyn-Backdoor. $\mathcal{O}(M^2)$ indicates that the space complexity depends on the memory occupied by the model’s weight matrix whose is squared-level.

Time complexity analysis: The time cost of Dyn-Backdoor mainly comes from four parts, including the time cost for initial-triggers generation ($T_{\text{ini-trigger}}$), the gradient search cost time ($T_{\text{grad-search}}$), the time cost ($T_{\text{trigger}}$) to filter the trigger and the trigger generator optimization cost time ($T_{\text{opt-Gen}}$). Therefore, the time complexity of Dyn-Backdoor is,

$$\mathcal{O}(T_{\text{ini-trigger}}) + \mathcal{O}(T_{\text{grad-search}}) + \mathcal{O}(T_{\text{trigger}}) + \mathcal{O}(T_{\text{opt-Gen}}) \sim \mathcal{O}(I \cdot K \cdot N)$$

(17)

where $\mathcal{O}(T_{\text{ini-trigger}})$ depends on the number of the trigger generator iterations and the scale of the graph. $\mathcal{O}(T_{\text{grad-search}})$ depends on the number of the trigger generator iterations and the maximum number of modified links. $\mathcal{O}(T_{\text{trigger}})$ and $\mathcal{O}(T_{\text{opt-Gen}})$ depend on the number of updates to triggers in the training sequences.
Fig. 10. In attack scenario I, Dyn-Backdoor’s influence on graph degree distribution. (a), (b), (c) and (d) correspond to the experiments on the Radoslaw, Contact, Fb-forum and DNC dataset.

Fig. 11. Defense against Dyn-Backdoor on the Radoslaw dataset. Dyn-Backdoor represents attack without defense and Dyn-Backdoor-Def represents attack under defense.

Fig. 12. Running time of Dyn-Backdoor. $K$ is the number of iterations of the trigger generator. Thus, according to all the above steps, $O(I \cdot K \cdot N)$ indicates that the time complexity of Dyn-Backdoor is linear.

VI. CONCLUSION

This work focuses on backdoor attack on DLP. We propose a backdoor attack framework on DLP, named Dyn-Backdoor, by adopting GAN and gradient exploration to form a trigger. Extensive experiments show the effect of Dyn-Backdoor on DLP. The dynamic model will be left behind by training data with the trigger. Attacker can launch an attack by calling the trigger.

However, Dyn-Backdoor is still challenged in several aspects. When dealing with large scale datasets, the convergence is much slower than small ones to generate triggers. Dyn-Backdoor requires the target model to provide feedback information, and the backdoor attacks on the black-box setting is also a future research direction. Moreover, it is necessary to further pay attention to effective defense strategies for dynamic networks in different downstream tasks, e.g., node classification, edge classification and structural role classification. The influence of
trigger structure on backdoor attacks and the interpretability of backdoor attacks are also interesting.

APPENDIX A

THEORETICAL ANALYSIS ON DYN-BACKDOOR

To illustrate the feasibility of backdoor attacks, we conduct a certain theoretical analysis on Dyn-Backdoor. We unify and simplify the model structure of DLP methods, which consists of an encoding layer and a decoding layer to facilitate formula reasoning,

\[ f_\theta(X) = \text{sigmoid}(\Re Lu (\langle X \rangle w^0 + b^0) w^1 + b^1) \]  

(18)

where \( X \) is the input of the model, \( w^0, b^0 \) and \( w^1, b^1 \) are the weights and biases of the encoding layer and the decoding layer, respectively.

We simplify the non-linear activation function of the encoder layer. We use \( S + g \) instead of input \( X \).

\[ f_\theta(S + g) = \text{sigmoid}((S + g) w^0 w^1 + b^0 w^1 + b^1) \]  

(19)

where \( S \) is the benign sequence of graphs and \( g \) is the trigger. The output obtained by the dynamic model can be expressed as,

\[ A_t = f_\theta(S + g) = \text{sigmoid}(\alpha) \]

s.t. \( \alpha = (S + g) w^0 w^1 + b^0 w^1 + b^1 \)  

(20)

where \( A_t \) represents the neighbor matrix at time \( t \) of the graph predicted by the model.

We uniformly use the mean square error to express the loss function of model training. Then use gradient descent to update the weight parameters in the model. The update process of \( w^0 \) is as follows,

\[ E = \frac{1}{2} \left(A_t - \hat{A}_t\right)^2 \]  

(21)

where \( A_t \) represents the neighbor matrix at time \( t \) of the graph predicted by the model, \( \hat{A}_t \) represents the neighbor matrix at time \( t \) of the graph set by the attacker.

Parameters are optimized according to the chain method,

\[ \frac{\partial E}{\partial w^0} = \frac{\partial E}{\partial A_t} \frac{\partial A_t}{\partial \alpha} \frac{\partial \alpha}{\partial w^0} \]  

(22)

where \( E \) is the loss function of model training, \( A_t \) represents the neighbor matrix at time \( t \) of the graph predicted by the model. \( w^0 \) is the weight of the encoding layer. Then we derive the derivation of several parameters separately.

\[ \frac{\partial E}{\partial A_t} = A_t - \hat{A}_t \]
\[ \frac{\partial A_t}{\partial \alpha} = A_t (1 - A_t) \]
\[ \frac{\partial \alpha}{\partial w^0} = (S + g) w^1 \]

(23)

where \( S \) is the benign sequence of graphs and \( g \) is the trigger. So the change in \( w^0 \) can be expressed as,

\[ \Delta w^0 = -\eta \frac{\partial E}{\partial w^0} \]

where \( \eta \) is the learning rate. \( \Delta w^0 \) represents the amount of change in weight \( w^0 \). The same parameters change can be obtained,

\[ \Delta w^1 = -\eta \frac{\partial E}{\partial w^1} \]
\[ = \eta \left( A_t - \hat{A}_t \right) A_t (A_t - 1) (S + g) w^1 \]  

(24)

The coefficient \( \eta \) is the learning rate. \( \Delta w^0 \) represents the amount of change in weight \( w^0 \). The same parameters change can be obtained,

\[ \Delta b^0 = -\eta \frac{\partial \alpha}{\partial b^0} = \eta \left( A_t - \hat{A}_t \right) A_t (A_t - 1) w^1 \]
\[ \Delta b^1 = -\eta \frac{\partial \alpha}{\partial b^1} = \eta \left( A_t - \hat{A}_t \right) A_t (A_t - 1) \]

(25)

where \( \Delta w^1, \Delta b^0, \) and \( \Delta b^1 \) represent the amount of change in weight \( w^1, b^0 \) and \( b^1 \). We can see that through the optimization of the trigger, the update of the model parameters is controlled. In other words, the attacker can manipulate the parameters of the model through carefully designed triggers to leave a backdoor. This also provides feasible theoretical support for backdoor attacks on DLP methods.

APPENDIX B

FIG & TABLE

This section provides supplementary results of some experiments. The transferability attack results are shown in Tables VI–VIII. The parameter experiment results are shown in Figs. 13–15. The attack visualization results are shown in

---

TABLE VI

THE TRANSFERABILITY OF DYN-BACKDOOR ON THE CONTACT DATASET

| Attack Dataset | Link Model | TPR (%) | FPR (%) | TNR (%) | FNR (%) | TPR (%) | FPR (%) | TNR (%) | FNR (%) |
|----------------|------------|---------|---------|---------|---------|---------|---------|---------|---------|
| DYN [19]       | DynAE      | 9.62%   | 8.85    | 96.75%  | 1.23%   | 95.97   | 9.77%   | 96.77   | 0.75%   |
|                | DynRNN     | 8.04%   | 8.85    | 96.75%  | 1.23%   | 95.97   | 9.77%   | 96.77   | 0.75%   |
|                | DynGRU     | 13.26%  | 8.85    | 96.75%  | 1.23%   | 95.97   | 9.77%   | 96.77   | 0.75%   |
|                | B-LSTM-D   | 18.21%  | 8.85    | 96.75%  | 1.23%   | 95.97   | 9.77%   | 96.77   | 0.75%   |
|                | B-LSTM-D   | 23.94%  | 8.85    | 96.75%  | 1.23%   | 95.97   | 9.77%   | 96.77   | 0.75%   |
|                | B-LSTM-D   | 38.30%  | 8.85    | 96.75%  | 1.23%   | 95.97   | 9.77%   | 96.77   | 0.75%   |
|                | B-LSTM-D   | 53.89%  | 8.85    | 96.75%  | 1.23%   | 95.97   | 9.77%   | 96.77   | 0.75%   |
TABLE VII
THE TRANSFERABILITY OF DYN-BACKDOOR ON THE FB-FORUM DATASET

| Attack Dataset | Target Model | 5% | 10% | 20% | 5% | 10% | 20% |
|----------------|--------------|----|-----|-----|----|-----|-----|
| DDNE [19]      | DynAE        | 0.00 | 0.00 | 0.00 | 94.38 | 99.94 |
|                | DynRNN       | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | DynAEARNN    | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | E-LSTM-D     | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
| DynAE [17]     | DynAE        | 0.00 | 0.00 | 0.00 | 94.38 | 99.94 |
|                | DynRNN       | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | DynAEARNN    | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | E-LSTM-D     | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
| DynRNN [17]    | DynAE        | 0.00 | 0.00 | 0.00 | 94.38 | 99.94 |
|                | DynRNN       | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | DynAEARNN    | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | E-LSTM-D     | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
| DynAEARNN [17] | DynAE        | 0.00 | 0.00 | 0.00 | 94.38 | 99.94 |
|                | DynRNN       | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | DynAEARNN    | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | E-LSTM-D     | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
| E-LSTM-D [22]  | DynAE        | 0.00 | 0.00 | 0.00 | 94.38 | 99.94 |
|                | DynRNN       | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | DynAEARNN    | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |
|                | E-LSTM-D     | 94.38 | 99.94 | 99.94 | 94.38 | 99.94 | 99.94 |

Fig. 14. Parameter sensitivity analysis of $t$, $p$ and $n$ on the Contact dataset.

Fig. 15. Parameter sensitivity analysis of $t$, $p$ and $n$ on the DNC dataset.

Fig. 16. Visualization of a backdoored sequence on the Contact dataset in attack scenario I. Yellow links: the target link to be predicted. Yellow nodes: the source node and the destination node of the target link. Red links: the added links by Dyn-Backdoor. Green links: the deleted links by Dyn-Backdoor.

Fig. 17. (a), (b) and (c) represent the visualization of a backdoored sequence on the FB-forum dataset. (d), (e) and (f) represent the visualization of a backdoored sequence on the DNC dataset. Both belong to attack scenario I.

Fig. 18. In attack scenario II, Dyn-Backdoor’s influence on graph degree distribution on the Radoslaw dataset.

Fig. 19. In attack scenario II, Dyn-Backdoor’s influence on graph degree distribution on the Contact dataset.
Fig. 20. In attack scenario II, Dyn-Backdoor’s influence on graph degree distribution on the Fb-forum dataset.

Fig. 21. In attack scenario II, Dyn-Backdoor’s influence on graph degree distribution on the DNC dataset.

Fig. 22. Defense against Dyn-Backdoor on the Contact dataset.

Fig. 23. Defense against Dyn-Backdoor on the Fb-forum dataset.

Fig. 24. Defense against Dyn-Backdoor on the DNC dataset.

Figs. 16 and 17, and degree distribution results are shown in Figs. 18–21. The defense experiments are shown in Figs. 22–24.

REFERENCES

[1] M. Doostmohammadian and U. A. Khan, “On the complexity of minimum-cost networked estimation of self-damped dynamical systems,” IEEE Trans. Netw. Sci. Eng., vol. 7, no. 3, pp. 1891–1900, Jul.–Sep. 2020.

[2] X. Liu, Y. Zhou, X. Guan, and C. Shen, “A feasible graph partition framework for parallel computing of big graph,” Knowl. Based Syst., vol. 134, pp. 228–239, 2017.

[3] L. Macca, “Detecting and mitigating points of failure in community networks: A graph-based approach,” IEEE Trans. Comput. Soc. Syst., vol. 6, no. 1, pp. 103–116, Feb. 2019.

[4] A. Nordin, A. Tarable, C. Chiasserini, and E. Leonardi, “Belief dynamics in social networks: A fluid-based analysis,” IEEE Trans. Netw. Sci. Eng., vol. 5, no. 4, pp. 276–287, Oct.–Dec. 2018.

[5] F. Ahmed, A. X. Liu, and R. Jin, “Publishing social network graph eigenspectrum with privacy guarantees,” IEEE Trans. Netw. Sci. Eng., vol. 7, no. 2, pp. 892–906, Apr.–Jun. 2020.

[6] J. D. Hirsh and M. Sterbner, “Prediction of structural and functional features of protein and nucleic acid sequences by artificial neural networks,” Biochemistry, vol. 31, no. 32, pp. 7211–7218, 1992.

[7] J. Gao, Y. Xiao, J. Liu, W. Liang, and C. L. P. Chen, “A survey of communication/networking in smart grids,” Future Gener. Comput. Syst., vol. 28, no. 2, pp. 391–404, 2012.

[8] M. Kazemilari and M. A. Djauhari, “Correlation network analysis for multi-dimensional data in stocks market,” Physica A: Stat. Mechan. Appl., vol. 429, pp. 62–75, 2015.

[9] W. Zhang, Y. Tang, W. K. Wong, and Q. Miao, “Stochastic stability of delayed neural networks with local impulsive effects,” IEEE Trans. Neural Netw. Learn. Syst., vol. 26, no. 10, pp. 2336–2345, Oct. 2015.

[10] L. Yao, L. Wang, L. Pan, and K. Yao, “Link prediction based on common-neighbors for dynamic social network,” in Proc. 7th Int. Conf. Ambient Syst., Netw. Technol. 6th Int. Conf. Sustain. Energy Inf. Technol. / Affiliated Workshops, Madrid, Spain, 2016, pp. 82–89.

[11] Z. Zhang, J. Wen, L. Sun, Q. Dong, S. Su, and P. Yao, “Efficient incremental dynamic link prediction algorithms in social network,” Knowl. Based Syst., vol. 132, pp. 226–235, 2017.

[12] U. G. Acer, P. Drineas, and A. A. Abouzeid, “Random walks in time-graphs,” in Proc. 2nd Int. Workshop Mobile Opportunistic Netw., Pisa, Italy, 2010, pp. 93–100.

[13] N. M. A. Ibrahim, L. Chen, Y. Wang, B. Li, Y. Li, and W. Liu, “Sampling-based algorithm for link prediction in temporal networks,” Inf. Sci., vol. 374, pp. 1–14, 2016.

[14] N. M. A. Ibrahim and L. Chen, “An efficient algorithm for link prediction in temporal uncertain social networks,” Inf. Sci., vol. 331, pp. 120–136, 2016.

[15] G. H. Nguyen, J. B. Lee, R. A. Rossi, N. K. Ahmed, E. Koh, and S. Kim, “Continuous-time dynamic network embeddings,” in Proc. Companion Web Conf. Web Conf., 2018, pp. 969–976.

[16] P. Goyal, N. Kamra, X. He, and Y. Liu, “DynGEM: Deep embedding method for dynamic graphs,” 2018, arXiv:1805.11273.

[17] P. Goyal, S. R. Chhetri, and A. Canedo, “dyngraph2vec: Capturing network dynamics using dynamic graph representation learning,” Knowl. Based Syst., vol. 187, 2020, Art. no. 104816.

[18] Q. Xuan, H. Xiao, C. Fu, and Y. Liu, “Evolving convolutional neural network and its application in fine-grained visual categorization,” IEEE Access, vol. 6, pp. 31110–31116, 2018.

[19] T. Li, J. Zhang, P. S. Yu, Y. Zhang, and Y. Yan, “Deep dynamic network embedding for link prediction,” IEEE Access, vol. 6, pp. 29219–29230, 2018.

[20] A. Pareja et al., “Evolveng: Evolving graph convolutional networks for dynamic graphs,” in Proc. 34th AAAI Conf. Artif. Intell., AAAI 32nd Innov. Appl. Artif. Intell. Conf., 10th AAAI Symp. Educ. Adv. Artif. Intell., New York, NY, USA, 2020, pp. 5563–5570.

[21] J. Chen, X. Lin, C. Jia, Y. Li, and Y. Liu, “Generative dynamic link prediction,” Chaos, vol. 29, no. 12, Art. no. 123111, 2019.

[22] J. Chen et al., “E-LSTM-D: A deep learning framework for dynamic network link prediction,” IEEE Trans. Syst. Man Cybern. Syst., vol. 51, no. 6, pp. 3699–3712, Jun. 2021.

[23] J. Chen, J. Zhang, Z. Chen, M. Du, and Q. Xuan, “Time-aware gradient attack on dynamic network link prediction,” IEEE Trans. Knowl. Data Eng., vol. 35, no. 2, pp. 2091–2103, 2023.

[24] H. Fan et al., “Reinforcement learning-based black-box evasion attacks to link prediction in dynamic graphs,” in Proc. IEEE 23rd Int. Conf. High Perform. Comput. Commun.; 7th Int. Conf. Data Sci. Syst.; 19th Int. Conf. Smart City; 7th Int. Conf. Dependability Sensor, Cloud Big Data Syst. Appl., Haikou, Hainan, China, 2021, pp. 933–940.

[25] Z. Zhang, J. Jia, B. Wang, and N. Z. Gong, “Backdoor attacks to graph neural networks,” in Proc. 26th ACM Symp. Access Control Models Technol., Virtual Event, Spain, 2021, pp. 15–26.
Z. Xi, R. Pang, S. Ji, and T. Wang, “Graph backdoor,” in Proc. 30th USENIX Secur. Symp., 2021, pp. 1523–1540.

J. Xu, M. Xue, and S. Picek, “Explainability-based backdoor attacks against graph neural networks,” in Proc. WiseML, WiSec 2021: Proc. 3rd ACM Workshop Wireless Secur. Mach. Learn., Abu Dhabi, United Arab Emirates, 2021, pp. 31–36.

J. I. Goodfellow et al., “Generative adversarial nets,” in Proc. Adv. Neural Inf. Process. Syst. 27: Annu. Conf. Neural Inf. Process. Syst., Montreal, Quebec, Canada, 2014, pp. 2672–2680.

S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

X. Li, N. Du, H. Li, K. Li, J. Gao, and A. Zhang, “A deep learning approach to link prediction in dynamic networks,” in Proc. SIAM Int. Conf. Data Mining, Philadelphia, Pennsylvania, USA, 2014, pp. 289–297.

T. Li, B. Wang, Y. Jiang, Y. Zhang, and Y. Yan, “Restricted boltzmann machine-based approaches for link prediction in dynamic networks,” IEEE Access, vol. 6, pp. 29940–29951, 2018.

K. Selvarajah, K. Ragatham, Z. Kobti, and M. Kargar, “Dynamic network link prediction by learning effective subgraphs using CNN-LSTM,” in Proc. Int. Joint Conf. Neural Netw., Glasgow, U.K., 2020, pp. 1–8.

J. Chen, X. Wang, and X. Xu, “GC-LSTM: Graph convolution embedded LSTM for dynamic network link prediction,” Appl. Intell., vol. 52, no. 7, pp. 7513–7528, 2022.

J. Liu, C. Xu, C. Yin, W. Wu, and Y. Song, “K-core based temporal graph convolutional network for dynamic graphs,” IEEE Trans. Knowl. Data Eng., vol. 34, no. 8, pp. 3841–3853, Aug. 2022.

K. Liu, Y. Li, J. Yang, Y. Liu, and Y. Yao, “Generative principal component thermography for enhanced defect detection and analysis,” IEEE Trans. Instrum. Meas., vol. 69, no. 10, pp. 8261–8269, Oct. 2020.

K. Liu, Y. Tang, W. Lou, Y. Liu, J. Yang, and Y. Yao, “A thermographic data augmentation and signal separation method for defect detection,” Meas. Sci. Technol., vol. 32, no. 4, 2021, Art. no. 045401.

K. Lei, M. Qin, B. Bai, G. Zhang, and M. Yang, “GCN-GAN: A non-linear temporal link prediction model for weighted dynamic networks,” in Proc. IEEE Conf. Comput. Commun., Paris, France, 2019, pp. 388–396.

M. Yang, J. Liu, L. Chen, Z. Zhao, and Y. Shen, “An advanced deep generative framework for temporal link prediction in dynamic networks,” IEEE Trans. Cybern., vol. 50, no. 12, pp. 4946–4957, Dec. 2020.

M. Brand, “Fast low-rank modifications of the thin singular value decomposition,” Linear Algebra Appl., vol. 415, no. 1, pp. 20–30, 2006.

Z. Zhang, P. Cui, J. Pei, X. Wang, and W. Zhu, “TIMERS: Error-bounded SVD restart on dynamic networks,” in Proc. 32nd AAAI Conf. Artif. Intell., 30th Innov. Appl. Artif. Intell., 8th AAAI Symp. Educ. Adv. Artif. Intell., New Orleans, Louisiana, USA, 2018, pp. 224–231.

J. Li, H. Dani, X. Hu, J. Tang, Y. Chang, and H. Liu, “Attributed network embedding for learning in a dynamic environment,” in Proc. ACM Conf. Inf. Knowl. Manage., 2017, pp. 387–396.

X. Mu, P. Sun, and Y. Wang, “Graph regularized nonnegative matrix factorization for temporal link prediction in dynamic networks,” Physica A: Stat. Mechanics Appl., vol. 496, pp. 121–136, 2018.

Y. Zuo, G. Liu, H. Lin, J. Guo, X. Hu, and J. Wu, “Embedding temporal network via neighborhood formation,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, London, U.K., 2018, pp. 2857–2866.

T. Takahashi, “Indirect adversarial attacks via poisoning neighbors for graph convolutional networks,” in Proc. IEEE Int. Conf. Big Data, Los Angeles, CA, USA, 2019, pp. 1395–1400.

D. Zügner and S. Günnemann, “Adversarial attacks on graph neural networks via meta learning,” in Proc. 7th Int. Conf. Learn. Representations, New Orleans, LA, USA, May 6–9, 2019, pp. 1–15.

“Manufacturing emails network dataset–KONECT,” Apr. 2017. [Online]. Available: http://konect.uni-koblenz.de/networks/emailsnews.

“Haggle network dataset–KONECT,” Apr. 2017. [Online]. Available: http://konect.uni-koblenz.de/networks/haggle

“Facebook wall posts network dataset–KONECT,” Apr. 2017. [Online]. Available: http://konect.uni-koblenz.de/networks/wallposts

“Dnc co-recipient network dataset–KONECT,” Sep. 2016. [Online]. Available: http://konect.uni-koblenz.de/networks/dnc-temporal

B. Perozzi, R. Al-Rfou, and S. Skiena, “Deepwalk: Online learning of social representations,” in Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, New York, NY, USA, 2014, pp. 701–710.

A. Grover and J. Leskovec, “node2vec: Scalable feature learning for networks,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, San Francisco, CA, USA, 2016, pp. 855–864.

Jinyin Chen (Member, IEEE) received the B.S. and Ph.D. degrees from the Zhejiang University of Technology, Hangzhou, China, in 2004 and 2009, respectively. She studied evolutionary computing with the Ashikaga Institute of Technology, Ashikaga, Japan, in 2005 and 2006. She is currently a Professor with the Zhejiang University of Technology. Her research interests include artificial intelligence security, graph data mining, and evolutionary computing.

Haiyang Xiong received the M.S. degree from the Zhejiang University of Technology, Hangzhou, China, in 2023. His research interests include graph data mining and applications, and artificial intelligence.

Huailin Zheng received the B.S. and Ph.D. degrees from the Zhejiang University of Technology, Hangzhou, China, in 2017 and 2022, respectively. He is currently a university Lecturer with the Institute of Cyberspace Security, Zhejiang University of Technology. His research interests include deep learning and artificial intelligence security.

Jian Zhang received the B.S. degree in automation and the Ph.D. degree in control theory and engineering from the Zhejiang University of Technology, Hangzhou, China, in 2017 and 2022, respectively. He is currently a Lecturer with the School of Cyberspace Security, China. His research interests include graph neural networks, machine learning security, and anomaly detection.

Yi Liu (Member, IEEE) received the Ph.D. degree in control theory and engineering from Zhejiang University, Hangzhou, China, in 2009. From 2011 to 2020, he was an Associate Professor with the Institute of Process Equipment and Control Engineering, Zhejiang University of Technology, Hangzhou. From 2012 to 2013, he was a Postdoctoral Researcher with the Department of Chemical Engineering, Chung-Yuan Christian University, Taoyuan City, Taiwan. Since December 2020, he has been a Full Professor with the Zhejiang University of Technology, Hangzhou, China. He has authored or coauthored 50 research papers at IEEE Transactions and international journals. His research interests include data intelligence with applications to modeling, control, and optimization of industrial processes.