Graph Backdoor

Zhaohan Xi$^1$, Ren Pang$^1$, Shouling Ji$^2$, Ting Wang$^1$

$^1$Pennsylvania State University, College of Information Science and Technology
$^2$Zhejiang University, College of Computer Science and Technology
Motivation

- Backdoor attacks against DNNs
  - A trojan model responds to trigger-embedded inputs in a specific manner
  - While the trojan model functioning normally for untouched inputs

- Graph data and GNNs
  - Graph data format is widely use as a flexible representation
  - GNNs are learning-based models to capture graph/node properties
  - The vulnerabilities in graphs and GNNs are largely unexplored

- Graph-domain challenges
  - Trigger definition: has both topological structure and descriptive features
  - Input-tailored: a trigger is tailored to the characteristics of an individual graph
  - Adaptive location: a trigger should be embedded into a suitable locality
**GTA: Graph Trojaning Attack**

- **Upstream: adaptive learning**
  - The adversary forges a trojan GNN $f_\theta$ (pre-trained model) via perturbing its parameters.
  - To realize attack, the adversary leverages bi-level optimization between $f_\theta$ and trigger $g_t$.

- **Downstream: model-agonistic**
  - The adversary has no access to downstream model $h$, but $z_G$ can lead to a falsified result.
GTA: Trigger Generation

Graph encoding

- Use attention nets to encode $G$ and get $Z$
- The encodings are assured to capture both topological information and original features

Trigger generation

- Node connectivity: $\tilde{A}_{ij} = \mathbb{I}_{\text{sim}(\phi_\omega(z_i), \phi_\omega(z_j))}\geq 0.5$
- Backdoor features: $\tilde{X}_i = \sigma(Wz_i + b)$, $W, b \in \phi_\omega$
- Combine $\tilde{A}$ and $\tilde{X}$ as $g_t$, where $i, j \in g_t$
GTA: Backdoor Poisoning

- Trigger Injection
  - Rely on mixing function $m(G; g_t)$ to
    - Find to-be-replaced subgraph $g \in G$
    - Substitute $g$ with $g_t$

- Backdoor Poisoning
  - Inject trigger to not-target-label graphs $D_{[\not Y_{\text{tar}}]}$
  - Train GNNs $\theta$ with poisoned set $D$
GTA: Bi-level Optimization

- **Upper level – optimize trigger**
  - \( g^*_t = \arg\min_{g_t} l_{atk}(\theta^*(g_t), g_t) \)
  - \( l_{atk} \): difference between \( g_t \)-embedded graphs and \( G \in D_{y_{tar}} \) through GNNs

- **Lower level – optimize GNNs**
  - \( \theta^*(g_t) = \arg\min_{\theta} l_{ret}(\theta, g_t) \)
  - \( l_{ret} \): loss of GNNs
### Evaluation Settings

- **Multi-domain dataset**
  - Security-sensitive domains
  - Biology and chemistry
  - Social and transaction networks

- **Manifold learning settings**
  - Inductive (graph-level) & transductive (node-level) classification
  - Self-transfer & mutual-transfer learning
  - Graph-space (default) & input-space attacks

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Domain</th>
<th>Setting</th>
<th># Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fingerprint</td>
<td>Cybersecurity</td>
<td>Inductive, self-transfer</td>
<td>1.6k graphs</td>
</tr>
<tr>
<td>WinMal</td>
<td>Cybersecurity</td>
<td>Inductive, self-transfer</td>
<td>1.3k graphs</td>
</tr>
<tr>
<td>AIDS</td>
<td>Biochemistry</td>
<td>Inductive, mutual-transfer</td>
<td>2.0k graphs</td>
</tr>
<tr>
<td>Toxicant</td>
<td>Biochemistry</td>
<td>Inductive, mutual-transfer</td>
<td>10.3k graphs</td>
</tr>
<tr>
<td>AndroZoo</td>
<td>Cybersecurity</td>
<td>Inductive, input-space</td>
<td>0.2k graphs</td>
</tr>
<tr>
<td>Bitcoin</td>
<td>Transaction net</td>
<td>Transductive</td>
<td>5.6k nodes</td>
</tr>
<tr>
<td>Facebook</td>
<td>Social net</td>
<td>Transductive</td>
<td>12.5k nodes</td>
</tr>
</tbody>
</table>
Evaluation Settings (cont.)

- **Representative GNNs**
  - GCN (Kipf & Welling, 2017)
  - GAT (Velickovic et al. 2018)
  - GraphSAGE (Hamilton et al. 2017)

- **Self-variant baselines**
  - $BL^I$: a universal trigger with fully connected topo. + adaptive features
  - $BL^{II}$: a universal trigger with adaptive topo. + adaptive features

- **Comprehensive metrics**
  - Effectiveness: attack success rate (ASR), etc.
  - Evasiveness: clean accuracy drop (CAD), etc.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>GNN</th>
<th>Benign Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fingerprint $\emptyset$</td>
<td>GAT</td>
<td>82.9%</td>
</tr>
<tr>
<td>WinMal $\emptyset$</td>
<td>GraphSAGE</td>
<td>86.5%</td>
</tr>
<tr>
<td>Toxicant $\rightarrow$ AIDS</td>
<td>GCN</td>
<td>93.9%</td>
</tr>
<tr>
<td>AIDS $\rightarrow$ Toxicant</td>
<td>GCN</td>
<td>95.4%</td>
</tr>
<tr>
<td>ChEMBL $\rightarrow$ AIDS</td>
<td>GCN</td>
<td>90.4%</td>
</tr>
<tr>
<td>ChEMBL $\rightarrow$ Toxicant</td>
<td>GCN</td>
<td>94.1%</td>
</tr>
<tr>
<td>AndroZoo (A.)</td>
<td>GCN</td>
<td>95.3%</td>
</tr>
<tr>
<td>AndroZoo (A.+F.)</td>
<td>GCN</td>
<td>98.1%</td>
</tr>
<tr>
<td>Bitcoin</td>
<td>GAT</td>
<td>96.3%</td>
</tr>
<tr>
<td>Facebook</td>
<td>GraphSAGE</td>
<td>83.8%</td>
</tr>
</tbody>
</table>

- Abbreviation: A. – only use topology; A.+F. – use both topology and raw features
## Evaluations

### Inductive settings

<table>
<thead>
<tr>
<th>Settings</th>
<th>BL\textsuperscript{I}</th>
<th>BL\textsuperscript{II}</th>
<th>GTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASR, CAD</td>
<td>ASR, CAD</td>
<td>ASR, CAD</td>
</tr>
<tr>
<td>Fingerprint $\emptyset$</td>
<td>84.4%, 1.9%</td>
<td>87.2%, 1.6%</td>
<td>100%, 0.9%</td>
</tr>
<tr>
<td>WinMal $\emptyset$</td>
<td>87.2%, 1.8%</td>
<td>94.4%, 1.2%</td>
<td>100%, 0.0%</td>
</tr>
<tr>
<td>Toxicant $\rightarrow$ AIDS</td>
<td>89.4%, 1.7%</td>
<td>95.5%, 1.3%</td>
<td>98.0%, 1.4%</td>
</tr>
<tr>
<td>AIDS $\rightarrow$ Toxicant</td>
<td>80.2%, 0.6%</td>
<td>85.5%, 0.0%</td>
<td>99.8%, 0.4%</td>
</tr>
</tbody>
</table>

### Use the off-the-shelf GNNs

<table>
<thead>
<tr>
<th>Settings</th>
<th>BL\textsuperscript{I}</th>
<th>BL\textsuperscript{II}</th>
<th>GTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASR, CAD</td>
<td>ASR, CAD</td>
<td>ASR, CAD</td>
</tr>
<tr>
<td>ChEMBL $\rightarrow$ AIDS</td>
<td>92.0%, 1.1%</td>
<td>97.5%, 1.0%</td>
<td>99.0%, 1.2%</td>
</tr>
<tr>
<td>ChEMBL $\rightarrow$ Toxicant</td>
<td>83.5%, 0.6%</td>
<td>86.0%, 0.0%</td>
<td>96.4%, 0.1%</td>
</tr>
</tbody>
</table>
Evaluations (cont.)

- **Transductive settings (node-level classification)**

<table>
<thead>
<tr>
<th>Settings</th>
<th>( BL^I )</th>
<th>( BL^{II} )</th>
<th>GTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASR, CAD</td>
<td>ASR, CAD</td>
<td>ASR, CAD</td>
</tr>
<tr>
<td>Bitcoin</td>
<td>52.1%, 0.9%</td>
<td>68.6%, 1.2%</td>
<td>89.7%, 0.9%</td>
</tr>
<tr>
<td>Facebook</td>
<td>42.6%, 4.0%</td>
<td>59.6%, 2.9%</td>
<td>69.1%, 2.4%</td>
</tr>
</tbody>
</table>

- **Downstream model agnostic (different classifiers)**

<table>
<thead>
<tr>
<th>Classifiers</th>
<th>( BL^I )</th>
<th>( BL^{II} )</th>
<th>GTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASR, CAD</td>
<td>ASR, CAD</td>
<td>ASR, CAD</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>87.7%, 1.5%</td>
<td>92.4%, 0.9%</td>
<td>99.5%, 0.7%</td>
</tr>
<tr>
<td>Random Forest</td>
<td>85.8%, 0.9%</td>
<td>88.0%, 0.9%</td>
<td>90.1%, 0.6%</td>
</tr>
<tr>
<td>Gradient Boosting</td>
<td>82.5%, 0.6%</td>
<td>89.3%, 0.6%</td>
<td>94.0%, 0.6%</td>
</tr>
</tbody>
</table>
Input-space Case Study

- **Input-space constraints**
  - Transferable perturbations (triggers) from graph space
  - Not affect original functionalities of raw data samples
  - If possible, not incur observable semantic variations

- **GTA against Android Malware Detector (GNN-based)**

<table>
<thead>
<tr>
<th>Settings</th>
<th>Input-space GTA</th>
<th>Graph-space GTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASR</td>
<td>CAD</td>
</tr>
<tr>
<td>Topology Only</td>
<td>94.3%</td>
<td>0.9%</td>
</tr>
<tr>
<td>Topology + Feature</td>
<td>96.2%</td>
<td>1.9%</td>
</tr>
</tbody>
</table>
Potential Countermeasures

- **Data inspection: Randomized Smoothing (Zhang et al. 2020)**
  - Subsample a (possibly trigger-embedded) graph $G$ and generate $G_1, G_2, \ldots, G_n$
  - Take a majority voting among $G_1, G_2, \ldots, G_n$ as $G$’s final classification results
  - Adjust subsample ratio $\beta$ on both of node set and feature dimensions

- **Model inspection: Neural Cleanse (Wang et al. 2019)**
  - For each label, learn a reversed trigger from a backdoored GNN
  - Get the perturbation scale ($L_1$-norm) between the original graphs and the trigger-embedded
  - Use statistical approaches to measure which label has minimum perturbation scale
  - Consider different adaptiveness of reversed trigger (same as $BL^I$ and $BL^{II}$)
Summarizations

- **Graph-oriented**
  - GTA defines a trigger as a subgraph, including topo. structure and descriptive features

- **Input-tailored**
  - GTA generates triggers tailored to the characteristics of individual graphs

- **Downstream-model-agnostic**
  - GTA has no assumption of downstream model (used classifiers), leads to resistive trojaning attack

- **Attack-extensible**
  - GTA represents an attack framework on both inductive and transductive learning settings
Thank You!

For questions, feel free to contact

zxx5113@psu.edu