Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-26)
Sister Conferences Best Papers Track

LiSA: Leveraging Link Recommender to Attack Graph Neural Networks via Subgraph Injection (Extended Abstract)*

Wenlun Zhang[1], **Enyan Dai**[2†] and **Kentaro Yoshioka**[1]
[1]Keio University
[2]Hong Kong University of Science and Technology (Guangzhou)
{wenlun_zhang,kyoshioka47}@keio.jp, enyandai@hkust-gz.edu.cn

Abstract

Graph Neural Networks (GNNs) have demonstrated remarkable proficiency in modeling data with graph structures, yet recent research reveals their susceptibility to adversarial attacks. Traditional attack methodologies, which rely on manipulating the original graph or adding links to artificially created nodes, often prove impractical in real-world settings. This paper introduces a novel adversarial scenario involving the injection of an isolated subgraph to deceive both the link recommender and the node classifier within a GNN system. Specifically, the link recommender is mislead to propose links between targeted victim nodes and the subgraph, encouraging users to unintentionally establish connections and that would degrade the node classification accuracy, thereby facilitating a successful attack. To address this, we present the LiSA framework, which employs a dual surrogate model and bi-level optimization to simultaneously meet two adversarial objectives. Extensive experiments on real-world datasets demonstrate the effectiveness of our method.

1 Introduction

In recent years, Graph Neural Networks (GNNs) have shown remarkable proficiency in modeling graph-structured data. GNNs primarily operate on a message-passing mechanism. This process of iterative information aggregation enables GNNs to effectively update node representations while incorporating the structural properties. However, the powerful message-passing mechanism of GNNs also brings its share of challenges and limitations. Malicious attackers can intentionally perturb the graph structure and features. Such deliberately tampered information can propagate among nodes, leading to a degraded performance of GNNs [Dai *et al.*, 2022; Dai *et al.*, 2018; Zügner *et al.*, 2018; Zügner and Günnemann, 2019; Chen *et al.*, 2018; Wang *et al.*, 2020; Chang *et al.*, 2019]. Among various attack methods, manipulation attacks [Zügner *et al.*, 2018] have been demonstrated to be particularly effective. These attacks involve adding or deleting edges, or modifying node features in the original graph, all within a constrained budget.

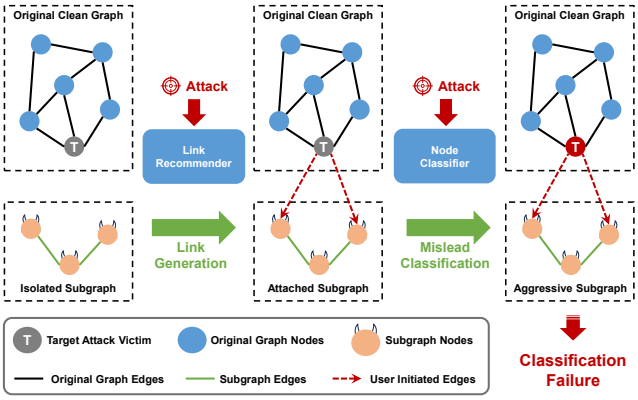

Figure 1: SIA scenario via link recommender.

Nonetheless, adversaries frequently encounter practical hurdles when attempting to manipulate these data, as it usually requires direct access to the backend data, a task that is not easily achievable. On the other hand, node injection attacks (NIAs) [Sun *et al.*, 2020; Tao *et al.*, 2021; Zou *et al.*, 2021] present a more feasible approach. Such attacks degrade classification performance by introducing malicious nodes linked to target nodes, spreading harmful attributes. Within citation networks, adversaries may craft counterfeit papers and insert references to target papers, ultimately causing them to be misclassified. This kind of emerging attack method have demonstrated high success rates and are increasingly seen as a more practical method in real-world scenarios. Injecting fake nodes is often more feasible than modifying data on existing nodes, making these techniques particularly insidious.

There is no doubt about the practicality and effectiveness of NIAs. Nevertheless, current research often focuses on ideal conditions, which implies that adversaries can freely establish connections between target nodes and injected fake nodes. However, this assumption does not universally hold across various applications. Especially in applications where secu-

*This paper is an extended abstract of the work that received the Best Paper Award at PAKDD 2025 [Zhang *et al.*, 2025].

†Corresponding author.

rity is critical, like social networks, fake users who connect with the target user can subtly alter the ground truth label of the target user, leading to targeted advertising. However, it is not certain that target users will approve connection request with fake accounts. Thus, evaluating these attack models under the assumption of cost-free linking can be misleading, potentially leading to an overestimation of the attack outcomes. These concerns highlighted above have motivated us to focus on enhancing the likelihood of adversaries successfully linking fake nodes to the target nodes. Given the prevalence of link recommendation algorithms within GNN-based applications, we explore the potential to exploit these link recommenders to facilitate malicious link formation. The proposed subgraph injection attack (SIA) scenario is shown in Fig. 1. Our method does not manually add edges between target victim nodes and the aggressor fake nodes. Instead, it deceives the link recommender of GNN into actively initiating links formation. Once these links are established, the classification performance on the target victim nodes suffers a considerable decline, ultimately enabling a successful attack. This novel approach to adversarial attacks is crucial for real-world applications yet remains largely unexplored. Moreover, the uncertain feasibility of attacking multiple GNN models simultaneously poses a significant challenge in solving this problem.

In this work, we investigate a unique SIA scenario, which injects an isolated subgraph to deceive both the link recommender and the node classifier simultaneously. Specifically, it aims to mislead the link recommender into creating adversarial links from targeted victim nodes to the subgraph, thereby undermining the accuracy of node classification. Our contributions in this study are outlined as follows: **1)** We explore a novel attack scenario utilizing the link recommender to compromise the node classification accuracy within a GNN. **2)** We introduce Link Recommender-Subgraph Injection Attack (LiSA), a pioneering framework crafted to effectively generate subgraphs by employing a dual surrogate model and bi-level optimization. **3)** Through rigorous experimentation on diverse real-world datasets, we validate the effectiveness and applicability of our proposed methodologies.

2 Related Works

Adversarial attacks on GNNs aim to exploit model vulnerabilities and are generally categorized into evasion, poisoning, and backdoor attacks. Evasion attacks manipulate graph structure or node features during the test phase to induce incorrect predictions without modifying the trained model [Zügner *et al.*, 2018; Zou *et al.*, 2021; Xu *et al.*, 2019]. Poisoning attacks instead occur during training, where adversarial edges or nodes are injected into the dataset to distort the learning process and degrade model reliability [Dai *et al.*, 2018; Sun *et al.*, 2020]. Backdoor attacks embed hidden triggers into the model during training, causing the GNN to produce incorrect outputs when the trigger appears while behaving normally otherwise [Xi *et al.*, 2020; Zhang *et al.*, 2021; Dai *et al.*, 2023]. However, these methods often struggle to demonstrate their effectiveness in realistic scenarios. In practice, attackers typically cannot directly modify the original graph, and injected nodes often fail to establish connections with target victims, significantly reducing attack success rate (ASR). To address this limitation, we propose leveraging link recommenders to facilitate link generation between injected nodes and target victims, increasing the likelihood of forming connections and improving the practicality and effectiveness of graph adversarial attacks.

3 Preliminaries

3.1 Notations

We define the original graph as $G_o = (A_o, F_o)$, with adjacency matrix $A_o \in \mathbb{R}^{N_o \times N_o}$ and feature matrix $F_o \in \mathbb{R}^{N_o \times D}$. The target victim nodes, $v_t \in V$, are selected for the attack. An isolated subgraph, denoted as $G_s = (A_s, F_s)$, is injected, forming the poisoned graph $G_p = (A_p, F_p)$:

$$A_p = \begin{bmatrix} A_o & 0 \\ 0 & A_s \end{bmatrix}, \quad F_p = \begin{bmatrix} F_o \\ F_s \end{bmatrix}, \tag{1}$$

where $A_p \in \mathbb{R}^{(N_o+N_s)\times(N_o+N_s)}$ and $F_p \in \mathbb{R}^{(N_o+N_s)\times D}$ is the adjacency matrix and features of the poisoned graph.

3.2 Threat Model and Problem Formulation

Attacker's Goal. The objective of the adversary is to deceive both the link recommender, conceptualized as a form of link prediction model, and the node classifier simultaneously in a GNN system. This is achieved through the strategic injection of a deliberately crafted subraph, which will mislead the link prediction model to form connections with the target nodes, thereby influencing classification accuracy thereafter.

Attacker's Knowledge and Capability. Attackers lack specific knowledge regarding the GNN models employed, including the particular type of link prediction and node classification model utilized within the system. Nevertheless, attackers possess access to the original attributed graph as well as a restricted set of training and validation labels. Attackers are capable of introducing an isolated subgraph to the original graph. However, any alteration to the original graph or establishment of connections between the original graph and the subgraph is strictly prohibited. Additionally, the features and node degrees of the subgraph must closely resemble those of the original graph to maintain inconspicuousness.

Problem. We assume the presence of a link recommender f_ϕ in the GNN, utilizing a subset of edges from core users in the original graph to suggest connections. This results in an updated graph G_r, with:

$$G_r = (\hat{A}, F_p), \quad \hat{A} = f_\phi(G_p). \tag{2}$$

The SIA strategically generates adversarial edges between the target victim nodes and the isolated subgraph, updating the adjacency matrix to:

$$\hat{A} = \begin{bmatrix} A_o & A_l \\ A_l^T & A_s \end{bmatrix}. \tag{3}$$

Here, $A_l \in \mathbb{R}^{N_s \times N_o}$ represents the edges generated via link recommender, and the rank of A_l increases from zero, which indicates that some adversarial edges have been generated successfully. Finally, the node classification model f_θ provides predictions based on the updated graph:

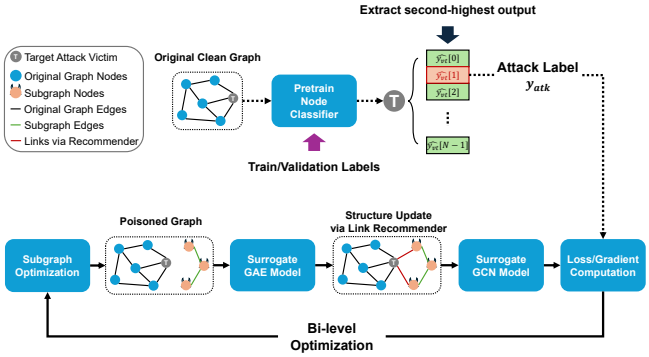

Figure 2: An overview of LiSA framework.

$$\hat{y_{v_t}} = f_\theta(G_r), \tag{4}$$

where $\hat{y_{v_t}}$ is the predicted class on target victim nodes. The ultimate objective of SIA is to compromise classification accuracy, making $\hat{y_{v_t}}$ diverge from the ground truth labels.

4 Methodology

In this section, we investigate an optimization-based approach for generating subgraphs. In the SIA scenario, achieving dual adversarial goals within the SIA scenario presents significant technical challenges. Intuitively, generating links is more likely when there is a close resemblance in the node representations between the target victim nodes and the nodes of the aggressor subgraph, whereas a greater disparity in these representations increases the chances of the victim nodes being misclassified. As a result, a more deliberate subgraph optimization technique is essential to balance two conflicting objectives and identify the optimal subgraph for SIA. To this end, we introduce the LiSA framework, as depicted in Fig. 2.

4.1 LiSA Framework

The LiSA framework is designed to challenge the performance of two GNN models through a strategic deployment of dual-surrogate models for bi-level optimization in a black-box attack context. We select the GAE [Kipf and Welling, 2016] and GCN [Kipf and Welling, 2017] as the surrogate models for link prediction and node classification, aiming to refine the adjacency matrix and feature attributes of the subgraph to meet two critical optimization goals concurrently: **i)** the generation of adversarial edges between the original graph and the subgraph via the link recommender, and **ii)** negatively affect node classification performance on target victim nodes once they are connected to the subgraph. To navigate these objectives, the attack loss function of LiSA includes two terms, balanced by a factor α, to regulate the contribution of each model:

$$\mathcal{L}_{atk} = \mathcal{L}^{cls}_{atk}(G_r;\theta) + \alpha\mathcal{L}^{link}_{atk}(G_p;\phi). \tag{5}$$

The loss related to link prediction, $\mathcal{L}^{link}_{atk}$, is defined as a reconstruction loss with Q negative sampling, aiming to mislead the link predictor into establishing adversarial links a_{ij} between the original graph and the subgraph:

$$\begin{aligned}\mathcal{L}^{link}_{atk} = \mathcal{L}_{recon} = &-\sum_{i\in V}\sum_{j\in V}\log \hat{a}_{ij} \\ &-\sum_{n=1}^{Q}\cdot \mathrm{E}_{v_n\sim P_n(v_i)}\log(1-\hat{a_{nij}})\end{aligned} \tag{6}$$

For the node classification, the attack loss, $\mathcal{L}^{cls}_{atk}$, seeks to reduce the chance of the target nodes being correctly classified by employing a cross-entropy loss with the attack label y_{atk}:

$$\mathcal{L}^{cls}_{atk} = -\sum_{v_t} y_{atk}\cdot\log(\hat{y_{v_t}}) \tag{7}$$

Notably, the approach begins by pretraining a node classifier using a limited label set, subsequently swapping the top-2 output classes to define the attack labels [Mopuri *et al.*, 2020]. This approach circumvents the use of GradMax, which complicates the task of balancing the two objectives. The continuously increasing adversarial loss for classification, $\mathcal{L}^{cls}_{atk}$, usually fails to maintain balance between these two loss terms. By giving precedence to the second-highest output class for the attack, we foster a more stable training environment. The comprehensive training strategy is outlined as follows:

$$\begin{aligned}\min_{A_s,F_s}\quad & \mathcal{L}^{cls}_{atk}(G_r;\theta^*) + \alpha\mathcal{L}^{link}_{atk}(G_p;\phi^*) \\ \text{s.t.}\quad & \theta^* = \arg\min_{\theta}\mathcal{L}^{cls}_{train}(f_\theta(G_r)), \\ & \phi^* = \arg\min_{\phi}\mathcal{L}^{link}_{train}(f_\phi(G_p)).\end{aligned} \tag{8}$$

The method involves the simultaneous bi-level optimization of the two surrogate models, with optimizing space limited to the subgraph.

4.2 Subgraph Training

The training procedure for the subgraph involves a two-tiered process with inner and outer iterations. In the inner iterations, we simultaneously train surrogate models for link prediction and node classification to optimize model parameters, while the outer iterations optimize the subgraph by alternately updating its adjacency matrix and node features. The inherent complexity of graph-structured data means that changes to the edges can significantly impact node representation. This alternating update methodology is necessary because it helps the training to achieve convergence.

Feature Optimization. The optimization process for refining the features within the subgraph employs a conventional approach that utilizes the principles of gradient descent through backpropagation. The goal is to adjust the node features in a manner that directly contributes to the two adversarial objectives. Each iteration of feature optimization is guided by the following rule:

$$F_s = F_s - lr\cdot\nabla_{F_s}\mathcal{L}_{atk}. \tag{9}$$

Here, lr stands for the learning rate, and the gradient of the attack loss with respect to the subgraph features, $\nabla_{F_s}\mathcal{L}_{atk}$, directs the update, ensuring that each adjustment supports the two adversarial goals. Adjusting the balance factor α in the

attack loss is crucial for aligning the feature updates with the dual aims of the attack.

Structure Optimization. Optimizing the structure of the subgraph presents a more complex challenge compared to feature adjustments. Upon the initialization of subgraph, the total number of edges within it is fixed. To maintain this constant edge count during each iteration of structure optimization, we ensure that for every edge removed, a corresponding edge is added. This process is designed to preserve the integrity of the subgraph structure while aligning with our adversarial objectives. The approach to directing structural updates involves calculating a combined gradient of the adjacency matrix at the outset of each optimization cycle:

$$\nabla_{str} = \nabla_{A_p} \mathcal{L}_{atk}^{link} + \beta \cdot \nabla_{\hat{A}} \mathcal{L}_{atk}^{cls}. \tag{10}$$

In this formula, $\nabla_{A_p} \mathcal{L}_{atk}^{link}$ denotes the gradient of attack loss by the link prediction model with respect to the subgraph adjacency matrix A_p before link recommender intervention, while $\nabla_{\hat{A}} \mathcal{L}_{atk}^{cls}$ represents the gradient of attack loss by the node classification model with respect to the post-update adjacency matrix $\hat{A}$ modified by the link recommender. The parameter β acts as a balancing factor in the optimization of the structure. The structural update process involves ranking the combined gradient ∇_{str}, then strategically adding and removing edges based on this ranking. Specifically, we add edges in locations where they are currently absent, prioritizing those with the lowest combined gradient values. Simultaneously, we remove edges from areas with the highest combined gradient values.

5 Experiments

To evaluate the effectiveness of our attack method, we conduct experiments on **Cora** and **PubMed**, two citation network datasets, and FacebookPagePage (**PagePage**), which represents a social network graph. We validate the proposed LiSA framework using GAE and VGAE [Kipf and Welling, 2016] as link recommenders, and GCN [Kipf and Welling, 2017] and SGC [Wu *et al.*, 2019] as node classifiers. We assume that the target user establishes links with the nodes having the top three linking scores in the poisoned graph. To assess the feasibility of our methods, we randomly select 1,000 nodes from each original graph as target victim nodes. We then generate subgraphs and execute attacks on these nodes individually. This approach allows us to compute both the link success rate (LSR) and the overall ASR. The LSR is defined as the proportion where at least one adversarial edge is generated between the target node and subgraph, while the ASR refers to the misclassfication rate on these victim nodes. For the link recommender, we use 85% of the edges for training, with half serving as positive edge labels with negative sampling for both subgraph optimization and attack verification. For node classification, 5% of the nodes are randomly chosen for training labels. These labels are utilized in subgraph optimization and the pretraining of a node classifier to identify the second highest output class as the attack label. To maintain a consistent basis for comparison across different datasets, we employ a subgraph comprising five nodes for each target victim node attack. However, we tailor hyperparameters such as the number of subgraph edges n_E, the balance parameters α and β, ensuring optimal performance and adaptability to the specific characteristics of each dataset.

Dataset	Rec.	Class.	LSR (%)	ASR (%)
Cora	-	GCN	-	18.7
	-	SGC	-	18.7
	GAE	GCN	52.3	48.6
	GAE	SGC	52.1	56.1
	VGAE	GCN	44.6	50.3
	VGAE	SGC	45.9	59.9
PubMed	-	GCN	-	21.6
	-	SGC	-	25.3
	GAE	GCN	82.3	68.5
	GAE	SGC	79.9	68.8
	VGAE	GCN	77.3	66.8
	VGAE	SGC	79.5	68.2
PagePage	-	GCN	-	11.7
	-	SGC	-	26.8
	GAE	GCN	50.4	48.0
	GAE	SGC	51.6	61.6
	VGAE	GCN	42.4	45.0
	VGAE	SGC	35.8	52.6

Table 1: Attack results (LSR & ASR) across different datasets and GNN model configurations. The ASR of the classifier corresponds to the misclassification rate on the clean graph.

The attack results obtained using the LiSA framework are presented in Table 1. For reference, we also report the misclassification rates on the original clean graphs to evaluate the impact of the proposed attacks. The results show that the subgraphs generated by LiSA successfully establish links between the injected subgraphs and target nodes, leading to a significant degradation in node classification accuracy. For instance, LiSA achieves over 77% LSR and 66% ASR on PubMed, resulting in more than a 45% decrease in classification performance compared with the clean model. These results demonstrate the effectiveness and efficiency of the LiSA framework and highlight the need to develop defenses against such attacks in real-world scenarios. For more detailed experiment results qualitative analysis, please refer to the original LiSA paper published in PAKDD 2025 [Zhang *et al.*, 2025].

6 Conclusions

This study presents a new attack scenario, along with a novel attack framework that utilizes link recommender to compromise the node classification accuracy of GNNs through subgraph injection. Unlike conventional attacks that require direct manipulation of the original graph or the addition of links to artificially created nodes, LiSA executes an attack indirectly by injecting an isolated subgraph, thereby enhancing practicality in real-world scenarios. Our experiments across various real-world datasets demonstrate the effectiveness of LiSA, illustrating significant degradation in GNN performance.

Acknowledgments

This material is based upon work supported by, or in part by, the National Natural Science Foundation of China (NSFC) under Grant No. 62506316, and the Guangdong Provincial Program under Grant No. 2025DO3JOO15. This research is also supported in part by JST CREST JPMJCR23M4 and JPMJCR21D2, JST Next-generation Edge AI Semiconductor JPMJES2515, JST SPRING JPMJSP2123, JSPS Kakenhi 23H00467, 24K02940, Futaba Foundation, Asahi Glass Foundation, and Telecommunications Advancement Foundation.

References

[Chang *et al.*, 2019] Heng Chang, Yu Rong, Tingyang Xu, Wenbing Huang, Honglei Zhang, Peng Cui, Wenwu Zhu, and Junzhou Huang. A restricted black-box adversarial framework towards attacking graph embedding models. In *AAAI*, 2019.

[Chen *et al.*, 2018] Jinyin Chen, Yangyang Wu, Xuanheng Xu, Yixian Chen, Haibin Zheng, and Qi Xuan. Fast gradient attack on network embedding. *ArXiv*, 2018.

[Dai *et al.*, 2018] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. Adversarial attack on graph structured data. In *ICML*, volume 80, pages 1115–1124, 2018.

[Dai *et al.*, 2022] Enyan Dai, Tianxiang Zhao, Huaisheng Zhu, Jun Xu, Zhimeng Guo, Hui Liu, Jiliang Tang, and Suhang Wang. A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability. *ArXiv*, 2022.

[Dai *et al.*, 2023] Enyan Dai, Minhua Lin, Xiang Zhang, and Suhang Wang. Unnoticeable backdoor attacks on graph neural networks. In *WWW*, page 2263–2273, 2023.

[Kipf and Welling, 2016] Thomas Kipf and Max Welling. Variational graph auto-encoders. *ArXiv*, 2016.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[Mopuri *et al.*, 2020] Konda Reddy Mopuri, Vaisakh Shaj, and R. Venkatesh Babu. Adversarial fooling beyond "flipping the label". *arXiv*, 2020.

[Sun *et al.*, 2020] Yiwei Sun, Suhang Wang, Xianfeng Tang, Tsung-Yu Hsieh, and Vasant Honavar. Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach. In *WWW*, page 673–683, 2020.

[Tao *et al.*, 2021] Shuchang Tao, Qi Cao, Huawei Shen, Junjie Huang, Yunfan Wu, and Xueqi Cheng. Single node injection attack against graph neural networks. In *CIKM*, page 1794–1803, 2021.

[Wang *et al.*, 2020] Jihong Wang, Minnan Luo, Fnu Suya, Jundong Li, Zijiang James Yang, and Qinghua Zheng. Scalable attack on graph data by injecting vicious nodes. *Data Mining and Knowledge Discovery*, 34:1363 – 1389, 2020.

[Wu *et al.*, 2019] Felix Wu, Tianyi Zhang, Amauri H. de Souza, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. Simplifying graph convolutional networks. In *ICML*, 2019.

[Xi *et al.*, 2020] Zhaohan Xi, Ren Pang, Shouling Ji, and Ting Wang. Graph backdoor. In *USENIX Security Symposium*, 2020.

[Xu *et al.*, 2019] Kaidi Xu, Hongge Chen, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Mingyi Hong, and Xue Lin. Topology attack and defense for graph neural networks: an optimization perspective. In *IJCAI*, page 3961–3967, 2019.

[Zhang *et al.*, 2021] Zaixi Zhang, Jinyuan Jia, Binghui Wang, and Neil Zhenqiang Gong. Backdoor attacks to graph neural networks. In *SACMAT*, page 15–26, 2021.

[Zhang *et al.*, 2025] Wenlun Zhang, Enyan Dai, and Kentaro Yoshioka. Lisa: Leveraging link recommender to attack graph neural networks via subgraph injection. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 15–26. Springer, 2025.

[Zou *et al.*, 2021] Xu Zou, Qinkai Zheng, Yuxiao Dong, Xinyu Guan, Evgeny Kharlamov, Jialiang Lu, and Jie Tang. Tdgia: Effective injection attacks on graph neural networks. In *KDD*, page 2461–2471, 2021.

[Zügner and Günnemann, 2019] Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In *ICLR*, 2019.

[Zügner *et al.*, 2018] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *KDD*, page 2847–2856, 2018.