HeTa: Relation-wise Heterogeneous Graph Foundation Attack Model

Yuling Wang[1,2], **Zihui Chen**[1,2], **Pengfei Jiao**[1,2*] and **Xiao Wang**[3]
[1] School of Cyberspace, Hangzhou Dianzi University
[2] Data Security Governance Zhejiang Engineering Research Center, Hangzhou Dianzi University
[3]Beihang University
{wangyl0612, 231270002, pjiao}@hdu.edu.cn, xiao_wang@buaa.edu.cn

Abstract

Heterogeneous Graph Neural Networks (HGNNs) are vulnerable, highlighting the need for tailored attacks to assess their robustness and ensure security. However, existing HGNN attacks often require complex retraining of parameters to generate specific perturbations for new scenarios. Recently, foundation models have opened new horizons for the generalization of graph neural networks by capturing shared semantics across various graph distributions. This leads us to ask: *Can we design a foundation attack model for HGNNs that enables generalizable perturbations across different HGNNs, and quickly adapts to new heterogeneous graphs (HGs)?* Empirical findings reveal that, despite significant differences in model design and parameter space, different HGNNs surprisingly share common vulnerability patterns from a relation-aware perspective. Therefore, we explore how to design foundation HGNN attack criteria by mining shared attack units. In this paper, we propose a novel relation-wise heterogeneous graph foundation attack model, HeTa. We introduce a foundation surrogate model to align heterogeneity and identify the importance of shared relation-aware attack units. Building on this, we implement a serialized relation-by-relation attack based on the identified relational weights. In this way, the perturbation can be transferred to various target HGNNs and easily fine-tuned for new HGs. Extensive experiments exhibit powerful attack performances and generalizability of our method.

1 Introduction

Heterogeneous graphs (HGs), prevalent in fields like biological networks [Ma *et al.*, 2023] and knowledge graphs [Sanmartin, 2024], realistically represent complex systems with diverse nodes and edges across different domains. Existing heterogeneous graph neural networks (HGNNs) are often vulnerable to adversarial attacks due to perturbation amplification from the complex coupling of heterogeneous nodes and relations [Zhang *et al.*, 2022]. Therefore, developing tailored attack methods for HGNNs is essential for assessing their robustness and ensuring the security of their applications.

Current attacks on HGs primarily utilize gradients from surrogate models with structures similar to the target model to perform topology attacks, thereby degrading the performance of the target HGNN. Broadly speaking, these attacks mainly fall into two groups: (1) targeted attack, which aim to reduce the performance of specific target instances but non-target might remain unchanged to avoid being detected [Wang *et al.*, 2024]; and (2) global attacks, which have recently emerged to reduce the overall performance of HGNNs with limited budget [Shang *et al.*, 2023]. While promising, existing methods on HGs rarely account for the generalization of attackers, which necessitates retraining attack parameters for different target HGNNs and new HGs. Although some recent works have attempted transferable attacks [Shang *et al.*, 2023; Zhao *et al.*, 2024], they fail to achieve cross-domain applicability and require elevated permissions to manipulate either the graph structure or the training data.

Due to their powerful generalization capabilities, foundation models have revolutionized the fields of Natural Language Processing [Qin *et al.*, 2023] and Computer Vision [Liu *et al.*, 2024], demonstrating significant potential in learning general, open-world knowledge from diverse data sources. This has endowed them with strong expressiveness and adaptability, enabling them to excel across a wide range of tasks and datasets. Building on this concept, Graph Foundation Models (GFMs) constitute a significant advancement in graph-structured data, facilitating cross-domain and cross-task generalization through the training of a unified, transferable vocabulary (e.g., basic graph elements like relations and subgraphs). This advancement prompts us to consider a natural question: *Can we design a foundation attack model for HGNNs that enables generalizable perturbations across different HGNNs, and quickly adapts to new HGs?*

To address this question, we empirically investigate the impact of removing different relation subgraphs from various HGNNs on the ACM dataset as Figure 1, as the relationships among nodes often serve as the fundamental semantic of HGs [Wang *et al.*, 2022; Wang *et al.*, 2019]. The results clearly show that the importance of the relations is consistent across these HGNNs, i.e., R1>R2>R3. Specifically, removing relation R1 typically leads to the most significant performance

*Corresponding author.

drop across all HGNNs (up to 23.6%), while removing R3 causes only a minor impact (up to 2.8%). It indicates that these HGNNs share a common importance pattern across relational subgraphs and are likely to be consistently vulnerable when the most crucial relation is perturbed. This motivates us to view relations as the fundamental shared attack unit, laying the groundwork for a foundation attack model for HGNNs.

Despite its potential, achieving this goal entails significant challenges. **First**, how to identify the shared importance distribution of attack semantic units across different HGNNs? The differences in structural and feature distribution between HGs are considerable [Xia and Huang, 2024]. Therefore, different HGNNs requires careful heuristic design and generally involves significantly large and diverse parameter spaces to model semantic diversity [Wang *et al.*, 2019], making it difficult to uncover the underlying generalizable principles shared across HGNNs. **Second**, once the importance pattern is captured, how to design universal attack criteria based on this pattern to progressively destroy each critical semantic unit in the HG? Since the HG is decomposed into different attack units with varying semantics, it is crucial to attack these units precisely, step by step, based on their importance, while employing a low-budget and low-permission strategy.

In this paper, we propose a relation-wise Heterogeneous graph foundation atTack model (HeTa), which identifies shared attack patterns based on relational semantics, enabling the attack process on HGs to generalize. Specifically, we develop a lightweight foundation surrogate model to provide a unified characterization of different HGNNs, i.e., simplifying their propagation mechanisms to model the importance distribution of shared relational semantic units. A simple parametric mechanism that enables fast generalization to new graphs. Subsequently, we implement a relation-by-relation serialized attack process based on the learned relation weights. This involves injecting adversarial nodes into each relation subgraph, with gradients from the carefully designed attack loss guiding the generation of fake edges, enabling a low-budget attack that requires minimal permissions to perturb the original topology. Finally, the perturbed HG can be fed into different target HGNNs to evaluate the attack's effectiveness and generalizability, and the attack on a new HG can be quickly fine-tuned by freezing part of the attacker's parameters. Our key contributions are as follows:

- To our knowledge, it is the first foundation attack model for HGNNs. We also verify that different HGNNs share common vulnerabilities in relation-aware attack units.
- We propose HeTa, a novel generalizable HGNN attack model, that enables transferability across different target HGNNs and easily adapts to new HGs.
- Extensive experiments on three public datasets demonstrate the effectiveness and generalizability of our proposed HeTa under node injection and evasion attacks.

2 Related Work

2.1 Heterogeneous Graph Neural Network

HGNNs are widely acclaimed for enhancing node representations. According to the way to handle heterogeneity, HGNNs mainly fell into meta-path-aware and relation-aware methods. The former relies on carefully designed meta-paths to aggregate information, often involving complex relation combinations. [Fu and King, 2024; Wang *et al.*, 2019]. The latter is aggregate messages from neighbors of different relations [Yu *et al.*, 2022; Yang *et al.*, 2023]. Recent advancement have focused on leveraging LLM to achieve generalization across diverse HGs [Tang *et al.*, 2024]). AnyGraph learns a foundation model from a vast of graphs to effectively handle the heterogeneity [Xia and Huang, 2024]. Despite significant differences, these HGNNs essentially regard relations as the most fundamental semantic units.

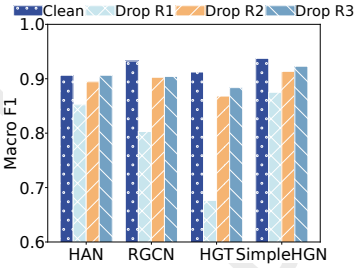


Figure 1: Performance of different HGNNs on the ACM dataset with various relations dropped, where 'Clean' denotes the original graph (R1: author-paper, R2: paper-subject, R3: paper-term).

2.2 Adversarial Attack on Graph Neural Network

Graph adversarial attacks have gained traction because minimal perturbations can mislead models. Homogeneous graph attack, such as [Chen *et al.*, 2018] and [Goodfellow *et al.*, 2014] use gradient information to guide attack edges. [Zhang *et al.*, 2024] proposes a susceptible-reverse influence sampling strategy for selecting neighbors. While in HG, the adversarial robustness remains less explored. Roughly speaking, attacks in HGNNs fell into target attack and global attack. The former manipulate graph to mislead target instances, e.g., [Zhao *et al.*, 2024] proposes a semantic-aware mechanism to automatically identify and generate perturbations to mislead target nodes. While the latter aim to destroy the total performance, e.g., [Shang *et al.*, 2023] proposes a structured global attack method guided by edge attention. While promising, no existing study has yet explored a generalizable foundation model for HGNNs.

3 Preliminaries

3.1 Heterogeneous Graph

Heterogeneous Graph (HG), $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, consists of an node set $\mathcal{V}$, an edge set $\mathcal{E}$ and a feature set $\mathcal{F}$. $\mathcal{G}$ is also associated with a node type mapping function $\phi : \mathcal{V} \to \mathcal{T}$ and an edge type mapping function $\psi : \mathcal{E} \to \mathcal{R}$. $\mathcal{T}$ and $\mathcal{R}$ denote the predefined type sets of node and edge, where $|\mathcal{T}| + |\mathcal{R}| > 2$. Let V_τ denotes the node set of type $\tau \in \mathcal{T}$, the feature set $\mathcal{F}$ is composed of $|\mathcal{T}|$ feature matrices, $\mathcal{F} = \{F_\tau, \tau \in \mathcal{T}\}$, $F_\tau \in \mathbb{R}^{|V_\tau| \times d_\tau}$, where d_τ is the feature dimension of τ nodes.

Relation Subgraph. Given a HG, $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, $\mathcal{G}_r$ is a subgraph of $\mathcal{G}$ that contains all edges of relation r. The adjacency matrix of $\mathcal{G}_r$ is $A_r \in \mathbb{R}^{N \times N}$, where N is the number of

nodes in $\mathcal{G}$. $A_r[i,j]=1$ if $\langle v_i, r, v_j\rangle$ exists in $\mathcal{G}_r$, otherwise $A_r[i,j]=0$, where $v_i, v_j \in \mathcal{V}$, $r \in \mathcal{R}$. $\mathcal{A}$ is adjacency matrix for $\mathcal{G}$, i.e., $\mathcal{A}=\sum_{r=1}^{\mathcal{R}} A_r$.

3.2 Node Injection Attack in HG

Given a HG, $\mathcal{G}=(\mathcal{A},\mathcal{F})$, node injection attack (NIA) generates a fake node set N_{in} and connects to existing nodes in $\mathcal{G}$. The perturbed HG after injecting N_{in} can be denoted as $\mathcal{G}'=(A'_1, A'_2, ..., A'_{|\mathcal{R}|}; F'_1, F'_2, ..., F'_{|\mathcal{T}|})$. E.g., for two types of nodes in $\mathcal{G}$, Author (A) and Paper (P), connected by edges type P-A, after injecting fake nodes N_{in} of type P, we have:

$$A'_{PA}=\begin{bmatrix} A_{PA} & E_{in} \\ E_{in}^{\top} & E_0 \end{bmatrix}, F'_P=\begin{bmatrix} F_P \\ F_{in} \end{bmatrix}, \tag{1}$$

where $A_{PA} \in \mathbb{R}^{N\times N}$ is the subgraph of P-A type, $E_{in} \in \mathbb{R}^{N\times|N_{in}|}$ is the adjacency between the original and injected nodes, and $E_0 \in \mathbb{R}^{|N_{in}|\times|N_{in}|}$ is the adjacency between injected nodes. $F_P \in \mathbb{R}^{N\times d_P}$ and $F_{in} \in \mathbb{R}^{|N_{in}|\times d_P}$ are the features of the original and the injected nodes, respectively.

Foundation Attacker's Goal. The foundation attacker aims to construct generalizable perturbation graphs that degrade the target model's performance, which can transfer to new scenarios, like new target models and graph domains.

Attacker's Knowledge and Capability. We focus on NIA in black-box and evasion settings, where the attacker can only modify the test data without access to the target model's parameters or architecture [Sun *et al.*, 2022]. Given a clean HG, $\mathcal{G}=(\mathcal{A},\mathcal{F})$, we train a surrogate model f_θ and freeze its parameters. Next, the attacker is trained based on the surrogate model's behavior to generate fake nodes and inject them into $\mathcal{G}$, forming a perturbed graph, $\mathcal{G}'=(\mathcal{A}',\mathcal{F}')$, within budget. Finally, we apply the attacked $\mathcal{G}'$ to other target models to reduce their predictions. The unified formulation is:

$$\begin{aligned} &\max_{G'} \mathcal{L}\left(f_{\theta^*}\left(\mathcal{G}'\right)\right), \\ &\text{s.t.} \quad \theta^*=\arg\min_{\theta} \mathcal{L}\left(f_\theta\left(\mathcal{G}\right)\right), \\ &|N_{in}| \le N*\rho, \quad \deg(\mathrm{v})_{\mathrm{v}\in\mathrm{N_{in}}} \le \mathrm{K}, \end{aligned} \tag{2}$$

where θ^* denotes the surrogate model's optimal parameters trained by the loss $\mathcal{L}$, i.e., the cross-entropy loss for node classification. The higher $\mathcal{L}\left(f_{\theta^*}\left(\mathcal{G}'\right)\right)$, the better the performance of the attack. The injected node set N_{in} is limited by a injected rate ρ and total node number N in $\mathcal{G}$, the degree of each injected node v is limited by average degree K of $\mathcal{G}$.

4 Proposed Framework

In this section, we introduce HeTa, a novel foundation attack model designed specifically for HGNNs by identifying relation-aware shared attack units. An overview of the framework is shown in Figure 2.

4.1 A Lightweight Foundation Surrogate Model

We introduce a foundation surrogate model to represent the vulnerabilities of different HGNNs, aiming to achieve two key objectives: (1) Generalization, providing a unified description of the common characteristics shared across various HGNNs, and (2) Vulnerability Learning, identifying the importance distribution of fundamental attack units in HGs.

Heterogeneous Features Alignment. Since different node types $\tau \in \mathcal{T}$ in HGs typically come from inconsistent distributions, we first align the original features $h_v \in \mathbb{R}^{1\times d_\tau}$ of various node types into a unified vector space:

$$h_v^0 = \text{Projector}(h_v), \tag{3}$$

where h_v^0 denotes the unified node representations in a shared semantic space. The feature aligner, i.e., Projector$(\cdot)$, can be flexibly configured as any trainable model for different node types; here, we instantiate it using a linear transformation.

Heterogeneous Foundation Message Passing. Building on the experimental results shown in Figure 1, which reveal a unified vulnerability pattern across different HGNNs from a relational perspective, we propose treating relational subgraphs as the fundamental attack units. To this end, we employ an ensemble multi-relational message passing mechanism [Wang *et al.*, 2022] to model relational semantics in HGs using learnable coefficients, i.e., the weights of fundamental attack units in our case. Specifically, messages from different relation subgraphs are aggregated by weighted summation based on the learned relational coefficients:

$$H^l=\sigma\left[\left(\sum_{r=1}^{\mathcal{R}} \mu_r \hat{A}_r\right) H^{l-1}\right], \tag{4}$$

where μ_r and $\hat{A}_r$ represent the weight and the normalized adjacency matrix of relation r, respectively, with $\sum_{r=1}^{\mathcal{R}} \mu_r = 1$. Given by $\hat{A}_r=\tilde{D}_r^{-\frac{1}{2}}\tilde{A}_r\tilde{D}_r^{-\frac{1}{2}}$, $\tilde{A}_r=A_r+I$ and $\tilde{D}_r=D_r+I$, where D_r is the degree matrix of A_r. σ denotes activation function, we use ReLU here. To preserve the node's original information as much as possible and prevent oversmoothing, we employ residual connections:

$$Y_{\text{pred}} = \text{Classifier}\left((1-\alpha)H^l+\alpha H^0\right), \tag{5}$$

where $H^0=[h_1^0, h_2^0 \ldots, h_N^0]$ is the initial features from Projector$(\cdot)$, and $\alpha \in [0,1]$ is a adjustable scaling factor. The Classifier$(\cdot)$ is used to output the probability distribution Y_{pred} of node labels and is implemented as an MLP in our work. We use cross-entropy loss for node classification as the training objective, as shown in Eq.(2). Each element in $\boldsymbol{\mu}=[\mu_1, \mu_2, \ldots, \mu_{|\mathcal{R}|}]$ is initially set to $\frac{1}{|\mathcal{R}|}$, and adaptively learned during this training.

Consequently, the optimized $\boldsymbol{\mu}$ indicates the importance distribution of relational attack units, which identify common vulnerabilities across HGNNs. Furthermore, its lightweight parameterization enables rapid adaptation to new HGs, and we will validate its generalizability in the experiments.

4.2 Relation-wise Generalizable Perturbation Generation

Traditional attacks on HGNNs typically generate perturbations based on an overall attack loss, without considering the disruption of each individual semantic component within the HG, leading to incomplete attacks. To alleviate this issue, we leverage the importance distribution across various relation-aware attack units provided by the surrogate model and systematically attack each unit in a serialized manner—i.e., at

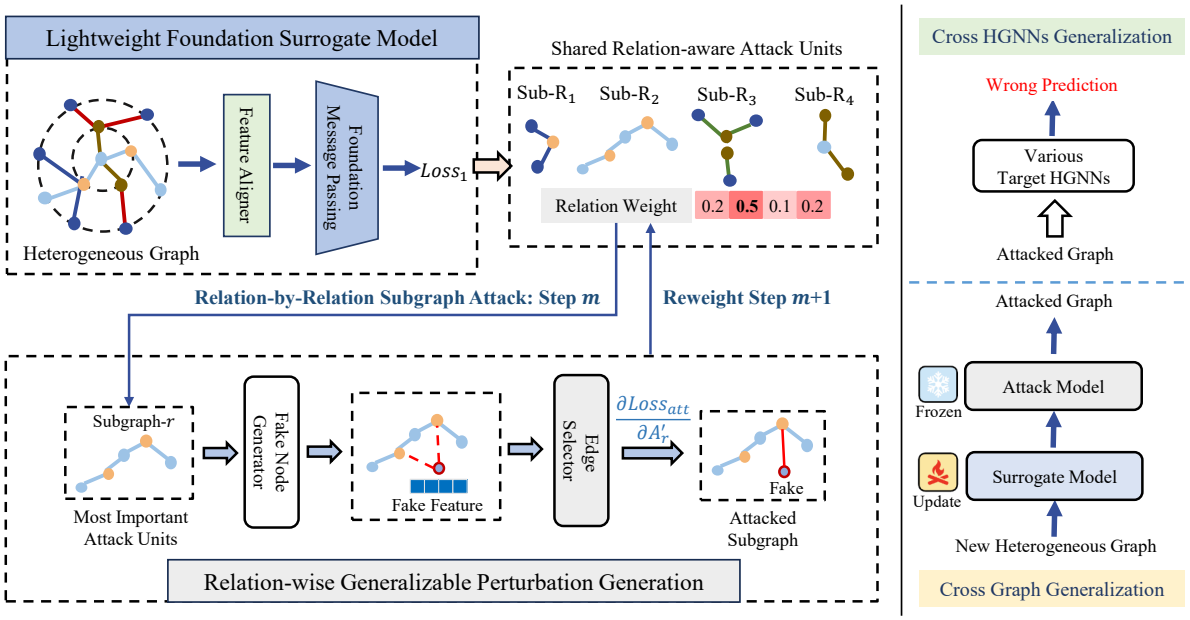


Figure 2: An overview of the framework for our proposed HeTa model.

each step, identifying and attacking the most important relation. In this way, the semantics of the entire HG will be progressively and thoroughly disrupted after M steps.

Relation-wise Attack Principle. We identify fundamental attack units, relation weight $\boldsymbol{\mu}$ by surrogate model, and the relations with larger weights play a more crucial role in the model's training and prediction. Based on this, we introduce a step-by-step relation-wise attack strategy to progressively perturb the most critical parts of the HG. The weights at step 0 are initialized as $\boldsymbol{\mu}$. Specifically, at step m, we attack the relation unit r with the largest weight:

$$r^m = \arg\max(\mu_1^m, \mu_2^m, \ldots, \mu_{|\mathcal{R}|}^m). \tag{6}$$

To prevent consecutive attacks from targeting the same relation, we introduce a penalty term β to dynamic reweight the relation unit that was attacked in the previous step, thereby obtaining the weights for next step: $\mu_r^{m+1} = \frac{\mu_r^m}{\beta}$.

Attack Loss. During attacks, we reversely optimize the training objective of the surrogate model. Here, our loss function $\mathcal{L}_{\text{atk}}$ includes the Carlini-Wagner (CW) attacks loss [Carlini and Wagner, 2017] $\mathcal{L}_{\text{cw}}$ and the inverse of the KL divergence [Ji *et al.*, 2020] $\mathcal{L}_{\text{kl}}$. The objective of the attack is to minimize the $\mathcal{L}_{\text{atk}}$:

$$\mathcal{L}_{\text{atk}} = \mathcal{L}_{\text{cw}} + \mathcal{L}_{\text{kl}}. \tag{7}$$

The CW loss minimizes the difference between the correct category and the maximum probability of other categories:

$$\mathcal{L}_{\text{cw}} = \sum_{v \in V} \max\left\{\left(p_{v,c} - \max_{y_v \neq c} p_{v,y_v}\right), -k\right\}, \tag{8}$$

where $p_{v,c}$ denotes the probability that node v is predicted to be target class c and $\max_{y_v \neq c} p_{v,y_v}$ represents the probability of the most likely incorrect class. V is test node set and k (set as 0 here) is a confidential level of making wrong predictions. The $\mathcal{L}_{\text{kl}}$ shifts the model's probability distribution:

$$-\mathcal{D}_{\text{kl}}(\hat{y}_v \parallel y_v) = \ln p_v, \tag{9}$$

where $\mathcal{D}_{\text{kl}}$ is KL divergence, $\hat{y}_v$ the predicted label distribution from surrogate model for node $v \in V$, and y_v is the groundtruth. p_v denotes the predicted probability that v belongs to its true category. To prevent gradient explosion when $p_v \to 0$, we use a smooth loss function [Zou *et al.*, 2021]:

$$\mathcal{L}_{\text{kl}} = \frac{1}{|V|} \sum_{v \in V} \max(r + \ln p_v, 0)^2, \tag{10}$$

where r is a control factor (set as 4 here).

Relation-wise Attack. At each attacking step $m \in [1, M]$, the disruption of the relation-aware semantic unit r involves two phases: first, generate the corresponding fake node set N_{in}^r, including their type and features, based on the relation of the current attack unit; second, create fake edge set E_{in}^r connecting these fake nodes, guided by the gradient information of the current relation subgraph.

(1) Fake Node Generator. Based on the current attack relation r selected in Eq.(6), we randomly choose the injected node type from the head and tail entity types of relation r, denoted as $\phi_{in}(r)$. The remaining unselected type is then designated as the connecting node type, denotes as $\phi_{con}(r)$. Here, we define the initial features and connections of the injected node $v_{in} \in N_{in}^r$. Specifically, we assume that v_{in} is initially connected to all nodes of type $\phi_{con}(r)$, and this connection will be further optimized by the edge selector. To enhance the imperceptibility of v_{in}, we calculate the cluster center of all nodes of type $\phi_{con}(r)$ as the prototype h_{con}, then randomly sample a node feature x_{in}^0 close to h_{con} as the initial feature

of v_{in}. To optimize the fake node's features for a more effective attack, we define a specific fake node generator $f_g^{\phi_{in}(r)}$ for node type $\phi_{in}(r)$ as follows:

$$x_{in}^m = f_g^{\phi_{in}(r)}\left(x_{in}^{m-1} + x_{\text{neighbor}}^{m-1}\right), \tag{11}$$

where x_{in}^m is the feature of fake node v_{in} at time step m, with an initial state x_{in}^0. x_{neighbor}^m corresponds to the feature aggregation of v_{in}'s neighbors. We set the fake node generators for different node types as distinct MLPs here.

(2) Fake Edge Selector. Then, the attacker will determine which target nodes in $\mathcal{G}$ to connect with the injected $v_{in} \in N_{in}^r$, aiming to maximize disruption of the graph information. A straightforward approach is to use the gradient of $\mathcal{L}_{\text{cw}}$ with respect to the relation subgraph to guide the attack, targeting the positions where the gradient changes most significantly [Wang *et al.*, 2020]. However, applying backpropagation to compute the gradients of each fake node's connections is challenging for large-scale graphs. We utilize an approximation strategy to simplify, after injecting v_{in}, we linearize the surrogate model with multi-layer graph convolution and precompute the gradient $\frac{\partial \mathcal{L}_{\text{cw}}}{\partial A_r^{'m}}$ in forward propagation, then selecting top-K absolute value as neighbors for v_{in}:

$$[A_r^{'m}]_{:,j} = \text{topK}\left\{\text{abs}([\frac{\partial \mathcal{L}_{\text{cw}}}{\partial A_r^{'m-1}}]_{:,j})\right\}, \tag{12}$$

where $A_r^{'m}$ is the attacked relation subgraph at the m-th step after injecting v_{in}, and $[\cdot]_{:,j}$ is the j-th column in the matrix, i.e., fake node. Details are in the Appendix A.

4.3 Applications of Perturbation

Built on the foundation principles of the surrogate model and relation-wise attack, the perturbed HG, $\mathcal{G}' = (\mathcal{A}', \mathcal{F}')$, generalizes to degrade downstream HGNNs and quickly adapts to new HGs through simple fine-tuning.

Degrading Target HGNNs. The attacked $\mathcal{G}'$ can be directly used to degrade predictions of various target models:

$$\hat{Y}_{\text{target}} = f_{\text{target}}\left(\mathcal{A}', \mathcal{F}'\right), \tag{13}$$

where $f_{\text{target}}(\cdot)$ can be any trained target HGNN, without the need to modify its internal architecture or parameters. $\hat{Y}_{\text{target}}$ is the prediction on the perturbed HG, which performance will significantly decrease compared to the original HG.

Adaptation to New HGs. HeTa has two parts of trainable parameters, i.e., the surrogate model and the fake node generators. Assume we have trained a surrogate model f_{θ^*} and a set of fake node generators $f_{g^*} = \left\{f_{g^*}^{(k)}\right\}_{k=1}^{K}$ on $\mathcal{G}_1$. Now, for a new graph $\mathcal{G}_2$ that may have a different distribution from $\mathcal{G}_1$, we can quickly adapt to $\mathcal{G}_2$ by freezing f_{g^*}. Specifically, the f_θ can be easily retrained due to its lightweight nature. Then, we can randomly select J frozen fake node generators from f_{g^*} for the new graph $\mathcal{G}_2$, provided that $J <= K$.

4.4 Additional Analysis

Remark 4.1. *Given a HG with a set of relation-aware attack units $\mathcal{A} = \{A_1, A_2, ..., A_{|\mathcal{R}|}\}$, for each $A_r \in \mathcal{A}$, the vulnerability of the target node to the semantic unit r increases as the degree of the node decreases.*

Proof: We focus on the attack unit A_r when injecting v_{in}.

$$A_r' = \begin{bmatrix} A_r & e_r \\ e_r^\top & 0 \end{bmatrix}, D_r' = \begin{bmatrix} D_r + d_1 & 0 \\ 0 & d_2 \end{bmatrix}, \tag{14}$$

where $A_r' \in \mathbb{R}^{|N+1|\times|N+1|}$, e_r is the injected edge set, D_r' is the degree matrix of A_r'. d_1 is the degree of the fake node with existing nodes, while d_2 is the degree of the fake node (set as K). Given by $\hat{A}'_r = \tilde{D}_r^{-\frac{1}{2}}(A_r'+I)\tilde{D}_r^{-\frac{1}{2}}$, $\tilde{D}_r = D_r + I$. For simplicity, let $\lambda = \tilde{D}_r + d_1$, $GD = \lambda^{-\frac{1}{2}}(A_r' + I)\lambda^{-\frac{1}{2}}$, $d_r = d_2 + 1$. For the alignment feature H from Eq.(3), the learnable parameters in the surrogate model denotes as W_2, the gradient $\frac{\partial L_{cw}}{\partial e_r}$ can be derived as follows:

$$\alpha = d_r^{-\frac{1}{2}}\lambda^{-\frac{1}{2}}\left[d_r^{-\frac{1}{2}}HW_2D_r^* + [GD]_{:,\mathcal{I}_V}\ Diag(h_{in}W_2)\right], \tag{15}$$

where D_r^* represents the degree information of the original graph $\mathcal{G}$ after injecting v_{in}. $\mathcal{I}_V$ is index set for test nodes. In this way, the first term can be ignored as it does not involve the injected fake nodes. We then analyze and simplify the second term as:

$$\lambda^{-\frac{3}{2}}d_r^{-\frac{1}{2}}(A_r' + I). \tag{16}$$

We observe that as λ decreases, i.e., the degree of the target node decreases, the gradient increases, making the node more susceptible to attack. Details are provided in the Appendix A.

5 Experiments

5.1 Experimental Settings

Datasets. In our experimental evaluation, we utilize three HG datasets: DBLP[1], ACM[1], and IMDB[2]. Details of these datasets are displayed in Appendix B.1.

Target HGNN Backbones. We validate the effectiveness and generalizability of HeTa on four widely used HGNNs, i.e., HAN [Wang *et al.*, 2019], HGT [Hu *et al.*, 2020], RGCN [Schlichtkrull *et al.*, 2018], SimpleHGN [Lv *et al.*, 2021]. Under evasion attacks, these HGNNs are trained on the clean graph and remain frozen parameters during evaluation.

Attack Baselines. We compare two types of baselines: (1) Heterogeneous graph attacks: specifically [Zhang *et al.*, 2022], called RoHe-attack. Given the lack of injection attacks tailored for HGs, we also compare (2) Homogeneous graph attacks: FGA [Chen *et al.*, 2018], G^{2}A2C [Ju *et al.*, 2023]. The details are in the Appendix B.2.

Implementation Details. The implementation details of the experiments are provided in Appendix B.3, and the pseudo-code for HeTa can be found in Appendix C.

5.2 Overall Performance

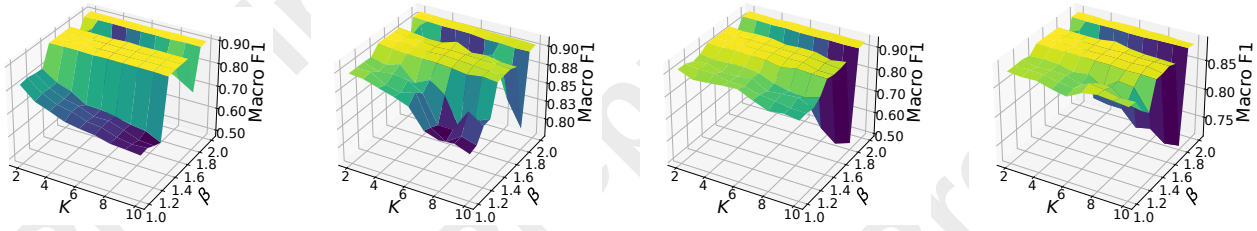
We conduct attacks on the surrogate model and use the perturbed graph obtained as the input for the target model, the results are shown in Table 1. We have the following observations: **(1) The proposed HeTa achieves state-of-the-art**

[1]https://github.com/THUDM/HGB

[2]https://github.com/seongjunyun/Graph_Transformer_Networks

Dataset	Target Model	Attack Methods	Clean		1%		2%		5%	
			Macro F1	Micro F1	Macro F1	Micro F1	Macro F1	Micro F1	Macro F1	Micro F1
IMDB	HAN	G^2A2C	0.5206	0.5356	0.5200	0.5351	0.5190	0.5341	0.5140	0.5311
		FGA			0.5157	0.5297	0.5093	0.5237	0.4859	0.5032
		RoHe-attack			0.4950	0.5143	0.4672	0.4890	0.4583	0.4955
		HeTa			**0.3278**	**0.4106**	**0.2964**	**0.3578**	**0.2465**	**0.2879**
	HGT	G^2A2C	0.5487	0.5634	0.5424	0.5583	0.5336	0.5510	0.5033	0.5275
		FGA			0.5336	0.5528	0.5219	0.5433	0.4838	0.5113
		RoHe-attack			0.5423	0.5583	0.5346	0.5528	0.5053	0.528
		HeTa			**0.5133**	**0.5214**	**0.5049**	**0.5098**	**0.4624**	**0.4601**
	SimpleHGN	G^2A2C	0.5711	0.5874	0.5684	0.5848	0.5638	0.5801	0.5557	0.5720
		FGA			0.5635	0.5788	0.5616	0.5775	0.5316	0.5408
		RoHe-attack			0.5370	0.5540	0.4985	0.5151	0.3951	0.4023
		HeTa			**0.3436**	**0.4478**	**0.3274**	**0.4203**	**0.2825**	**0.3511**
	RGCN	G^2A2C	0.4959	0.5245	0.4907	0.5207	0.4901	0.5200	0.4800	0.5120
		FGA			0.4906	0.5203	0.4866	0.5147	0.4700	0.5017
		RoHe-attack			0.4820	0.5143	0.4795	0.5121	0.4539	0.4925
		HeTa			**0.4055**	**0.4035**	**0.382**	**0.3726**	**0.3398**	**0.3246**
DBLP	HAN	G^2A2C	0.9239	0.9292	0.9210	0.9230	0.9032	0.9089	0.8900	0.8901
		FGA			-	-	-	-	-	-
		RoHe-attack			0.8782	0.8838	0.8456	0.8531	0.7636	0.7771
		HeTa			**0.8236**	**0.8311**	**0.5766**	**0.6021**	**0.4495**	**0.4981**
	HGT	G^2A2C	0.9049	0.9123	0.8817	0.8876	0.8628	0.8676	0.8066	0.8112
		FGA			-	-	-	-	-	-
		RoHe-attack			0.8880	0.8957	0.8608	0.8685	0.8206	0.8281
		HeTa			**0.8562**	**0.8623**	**0.8076**	**0.8114**	**0.7349**	**0.7357**
	SimpleHGN	G^2A^2C	0.9232	0.9285	0.9219	0.9271	0.9198	0.9250	0.9128	0.9179
		FGA			-	-	-	-	-	-
		RoHe-attack			0.8742	0.8799	0.8391	0.8454	0.8010	0.8105
		HeTa			**0.6687**	**0.6741**	**0.6569**	**0.6604**	**0.627**	**0.6483**
	RGCN	G^2A2C	0.9023	0.9084	0.8891	0.8947	0.8810	0.8862	0.8403	0.8443
		FGA			-	-	-	-	-	-
		RoHe-attack			0.8440	0.8510	0.7802	0.7880	0.6007	0.6080
		HeTa			**0.6611**	**0.6754**	**0.5693**	**0.5933**	**0.5153**	**0.5444**
ACM	HAN	G^2A2C	0.9064	0.9046	0.9034	0.9043	0.9000	0.9010	0.8912	0.8943
		FGA			0.8894	0.8871	0.8706	0.8677	0.8213	0.8168
		RoHe-attack			0.8658	0.8635	0.8253	0.822	0.7266	0.7247
		HeTa			**0.4712**	**0.5576**	**0.4226**	**0.5176**	**0.1818**	**0.3435**
	HGT	G^2A2C	0.9087	0.9079	0.9078	0.9069	0.9074	0.9065	0.9069	0.906
		FGA			0.8866	0.8857	0.8590	0.8578	0.7882	0.7865
		RoHe-attack			0.8959	0.8951	0.8825	0.8819	0.8356	0.8352
		HeTa			**0.8002**	**0.8006**	**0.6941**	**0.6971**	**0.682**	**0.6883**
	SimpleHGN	G^2A2C	0.9375	0.9367	0.9371	0.9362	0.9364	0.9357	0.936	0.935
		FGA			0.9365	0.9357	0.9384	0.9376	0.9296	0.9291
		RoHe-attack			0.9303	0.9296	0.9268	0.9263	0.9154	0.915
		HeTa			**0.6900**	**0.7015**	**0.6689**	**0.6791**	**0.1973**	**0.3510**
	RGCN	G^2A2C	0.8857	0.8838	0.8844	0.8824	0.8830	0.8810	0.8807	0.8786
		FGA			0.8722	0.8710	0.8654	0.8623	0.8555	0.8512
		RoHe-attack			0.8660	0.8630	0.8410	0.8401	0.8255	0.8205
		HeTa			**0.7559**	**0.7524**	**0.6605**	**0.6671**	**0.5243**	**0.5551**

Table 1: Overall attack performance on three datasets across four target HGNN backbones. Lower scores indicate better attacking ability. "Clean" means the original graphs without perturbations. Vacant positions ("-") mean that the models run out of memory on large graphs.



(a) HAN (b) HGT (c) SimpleHGN (d) RGCN

Figure 3: Analysis of the hyper-parameter K and β on ACM dataset with an injection rate of 0.01.

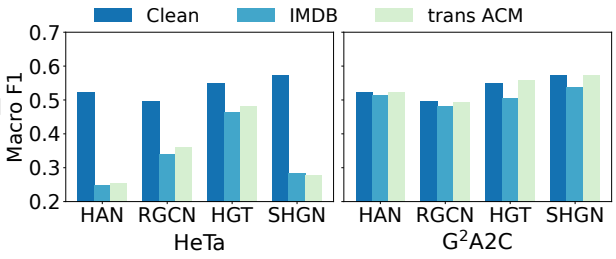
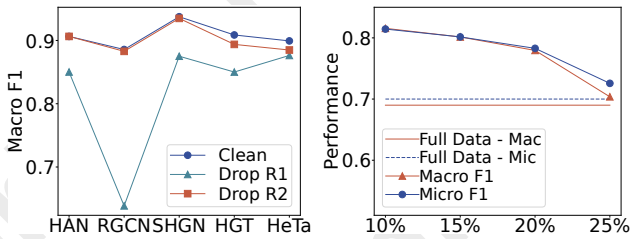


Figure 4: Results on IMDB dataset. 'Clean': no attack; 'IMDB': attacker trained on IMDB; 'trans ACM': attacker trained on ACM and adapted to IMDB. SHGN is SimpleHGN.



(a) Universality. (b) Data Efficiency.

Figure 5: Results on the ACM dataset at 1% injection. (a) Dropping relation subgraphs cross five HGNN models. (b) Sampling training sets at different ratios.

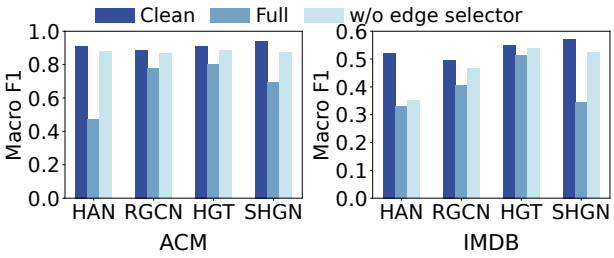


Figure 6: Ablation study with an injection rate of 0.01.

(SOTA) performance across all datasets and backbones. This further substantiates the effectiveness of HeTa in HGNN attack. Specifically, with a 0.01 node injection, compared with the best baseline attack, performance drops are up to 19% in IMDB, nearly 20% in DBLP, and almost 35% in ACM. **(2) Our proposed HeTa can generalize across various target HGNN models.** Surprisingly, the perturbations trained on the surrogate model achieved SoTa attack across all target models, causing significant performance degradation. Notably, with a 0.01 node injection, the average performance degradation by nearly 12% in IMDB, 15% in DBLP, and 19% in ACM. This generalization is attributed to the unified modeling of different HGNNs based on relational semantics.

5.3 Merits of HeTa

HeTa achieves cross-graph transferability. We pretrain the fake node generator in HeTa on the ACM dataset, freeze its parameters, and only fine-tune the surrogate model to adapt to the IMDB dataset. For G^{2}A2C, we also freeze its generator and finetuning with its own loss. The results in Figure 4 (and additional results in Figure 9 in the Appendix) show that the performance of trans ACM is close to that of IMDB, particularly on RGCN and HGT, where they are nearly identical. HeTa shows strong cross-graph transferability, this is because our attacker has learned universal characteristics within HGs, enabling it to efficiently adapt to new graph distributions.

Universality of our foundation surrogate model. We conduct experiments to validate the effect of dropping relation subgraphs with varying importance assigned by $\boldsymbol{\mu}$ in Figure 5a. According to $\boldsymbol{\mu}$, it indicates that R1 (author-paper) $>$ R2 (paper-term). The results show that removing R2 has little impact, while removing R1 causes a significant performance drop across all HGNNs. Notably, RGCN's performance drops sharply after removing R1, as it relies heavily on R1. This suggests that HeTa can identify the unified importance distribution of relations across various HGNNs.

HeTa exhibits strong data efficiency. We sample a small subset of nodes from the full training data at different ratios to create a new dataset, as shown in Figure 5b. Thanks to the minimal number of parameters in our surrogate model, which enables efficient training use of limited data. Specifically, when using only 25% of the training data, the attacker already achieves performance comparable to that with the full dataset, highlighting its data efficiency.

5.4 Model Analysis

Ablation Study. Since a key design of HeTa is the selection of fake edges, we replace this strategy with random edge selection at an injection rate of 0.01, as shown in Figure 6. From the result, the full method achieves the best performance, demonstrating that our key design can significantly degrade the performance of the target model.

Hyper-parameter Study. We analyze the impact of the injection degree K and the penalty term β in Figure 3. K controls the influence range of injected nodes, and β guides the selection of attack relations. Smaller K enhance attack effects by carefully selecting neighbors within a limited influence range, maintaining low visibility. However, larger K values can enhance attacks by broadening the influence. Attack effects initially improve with β, peaking at $\beta = 1.8$, and then deteriorate due to excessive punishment causing attack dispersion. More results are presented in Appendix D.1.

5.5 Conclusion

In this study, we introduce the foundation attack model within HGNNs, HeTa, which identifies shared attack patterns based on relational semantics, thereby enabling the attack process on HGs to generalize. We design a lightweight surrogate model to simplify various HGNN propagation mechanisms and model the important distribution of shared relational semantic units. Then we propose a relation-by-relation serialized attack process, which involves injecting adversarial nodes into each relation and generating fake edges. Experiments conducted on three datasets across four HGNN backbones have demonstrated that our method not only outperforms existing attack models but also exhibits remarkable generalization ability.

Acknowledgements

This work was supported in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LDT23F01015F01, in part by the Key Technology Research and Development Program of the Zhejiang Province under Grant No. 2025C1023 and in part by the National Natural Science Foundation of China (No. 62372146, 62322203, 62172052).

References

[Carlini and Wagner, 2017] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017.

[Chen *et al.*, 2018] Jinyin Chen, Yangyang Wu, Xuanheng Xu, Yixian Chen, Haibin Zheng, and Qi Xuan. Fast gradient attack on network embedding. *arXiv preprint arXiv:1809.02797*, 2018.

[Fu and King, 2024] Xinyu Fu and Irwin King. Mecch: metapath context convolution-based heterogeneous graph neural networks. *Neural Networks*, 170:266–275, 2024.

[Goodfellow *et al.*, 2014] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[Hu *et al.*, 2020] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pages 2704–2710, 2020.

[Ji *et al.*, 2020] Shuyi Ji, Zizhao Zhang, Shihui Ying, Liejun Wang, Xibin Zhao, and Yue Gao. Kullback–leibler divergence metric learning. *IEEE transactions on cybernetics*, 52(4):2047–2058, 2020.

[Ju *et al.*, 2023] Mingxuan Ju, Yujie Fan, Chuxu Zhang, and Yanfang Ye. Let graph be the go board: gradient-free node injection attack for graph neural networks via reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4383–4390, 2023.

[Liu *et al.*, 2024] Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, Lifang He, and Lichao Sun. Sora: A review on background, technology, limitations, and opportunities of large vision models, 2024.

[Lv *et al.*, 2021] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1150–1160, 2021.

[Ma *et al.*, 2023] Anjun Ma, Xiaoying Wang, Jingxian Li, Cankun Wang, Tong Xiao, Yuntao Liu, Hao Cheng, Juexin Wang, Yang Li, Yuzhou Chang, et al. Single-cell biological network inference using a heterogeneous graph transformer. *Nature Communications*, 14(1):964, 2023.

[Qin *et al.*, 2023] Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. Is chatgpt a general-purpose natural language processing task solver?, 2023.

[Sanmartin, 2024] Diego Sanmartin. Kg-rag: Bridging the gap between knowledge and creativity. *arXiv preprint arXiv:2405.12035*, 2024.

[Schlichtkrull *et al.*, 2018] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *The semantic web: 15th international conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, proceedings 15*, pages 593–607. Springer, 2018.

[Shang *et al.*, 2023] Yu Shang, Yudong Zhang, Jiansheng Chen, Depeng Jin, and Yong Li. Transferable structure-based adversarial attack of heterogeneous graph neural network. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 2188–2197, 2023.

[Sun *et al.*, 2022] Lichao Sun, Yingtong Dou, Carl Yang, Kai Zhang, Ji Wang, S Yu Philip, Lifang He, and Bo Li. Adversarial attack and defense on graph data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):7693–7711, 2022.

[Tang *et al.*, 2024] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Long Xia, Dawei Yin, and Chao Huang. Higpt: Heterogeneous graph language model. *arXiv preprint arXiv:2402.16024*, 2024.

[Wang *et al.*, 2019] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032, 2019.

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

[Wang *et al.*, 2022] Yuling Wang, Hao Xu, Yanhua Yu, Mengdi Zhang, Zhenhao Li, Yuji Yang, and Wei Wu. Ensemble multi-relational graph neural networks. *arXiv preprint arXiv:2205.12076*, 2022.

[Wang *et al.*, 2024] Haosen Wang, Can Xu, Chenglong Shi, Pengfei Zheng, Shiming Zhang, Minhao Cheng, and Hongyang Chen. Unsupervised heterogeneous graph rewriting attack via node clustering. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3057–3068, 2024.

[Xia and Huang, 2024] Lianghao Xia and Chao Huang. Anygraph: Graph foundation model in the wild, 2024.

[Yang *et al.*, 2023] Xiaocheng Yang, Mingyu Yan, Shirui Pan, Xiaochun Ye, and Dongrui Fan. Simple and efficient heterogeneous graph neural network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 10816–10824, 2023.

[Yu *et al.*, 2022] Pengyang Yu, Chaofan Fu, Yanwei Yu, Chao Huang, Zhongying Zhao, and Junyu Dong. Multiplex heterogeneous graph convolutional network. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2377–2387, 2022.

[Zhang *et al.*, 2022] Mengmei Zhang, Xiao Wang, Meiqi Zhu, Chuan Shi, Zhiqiang Zhang, and Jun Zhou. Robust heterogeneous graph neural networks against adversarial attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 4363–4370, 2022.

[Zhang *et al.*, 2024] Xiao Zhang, Peng Bao, and Shirui Pan. Maximizing malicious influence in node injection attack. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 958–966, 2024.

[Zhao *et al.*, 2024] He Zhao, Zhiwei Zeng, Yongwei Wang, Deheng Ye, and Chunyan Miao. Hgattack: Transferable heterogeneous graph adversarial attack. *arXiv preprint arXiv:2401.09945*, 2024.

[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 *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2461–2471, 2021.