Heterogeneous Graph Backdoor Attack

Jiawei Chen
jchen015@odu.edu
Old Dominion University
Norfolk, Virginia, USA

Lusi Li
lusili@cs.odu.edu
Old Dominion University
Norfolk, Virginia, USA

Daniel Takabi
takabi@odu.edu
Old Dominion University
Norfolk, Virginia, USA

Masha Sosonkina
msosonki@odu.edu
Old Dominion University
Norfolk, Virginia, USA

Rui Ning
rning@cs.odu.edu
Old Dominion University
Norfolk, Virginia, USA

Abstract

Heterogeneous Graph Neural Networks (HGNNs) excel in modeling complex, multi-typed relationships across diverse domains, yet their vulnerability to backdoor attacks remains unexplored. To address this gap, we conduct the first investigation into the susceptibility of HGNNs to existing graph backdoor attacks, revealing three critical issues: (1) high attack budget required for effective backdoor injection, (2) inefficient and unreliable backdoor activation, and (3) inaccurate attack effectiveness evaluation. To tackle these issues, we propose the **Heterogeneous Graph Backdoor Attack (HGBA)**, the first backdoor attack specifically designed for HGNNs, introducing a novel relation-based trigger mechanism that establishes specific connections between a strategically selected trigger node and poisoned nodes via the backdoor metapath. HGBA achieves efficient and stealthy backdoor injection with minimal structural modifications and supports easy backdoor activation through two flexible strategies: Self-Node Attack and Indiscriminate Attack. Additionally, we improve the ASR measurement protocol, enabling a more accurate assessment of attack effectiveness. Extensive experiments demonstrate that HGBA far surpasses multiple state-of-the-art graph backdoor attacks in black-box settings, efficiently attacking HGNNs with low attack budgets. Ablation studies show that the strength of HBGA benefits from our trigger node selection method and backdoor metapath selection strategy. In addition, HGBA shows superior robustness against node feature perturbations and multiple types of existing graph backdoor defense mechanisms. Finally, extension experiments demonstrate that the relation-based trigger mechanism can effectively extend to tasks in homogeneous graph scenarios, thereby posing severe threats to broader security-critical domains.

CCS Concepts

• **Security and privacy**; • **Computing methodologies** → *Machine learning*;

Keywords

Backdoor attacks, heterogeneous graph neural networks, and relation-based triggers.

ACM Reference Format:
Jiawei Chen, Lusi Li, Daniel Takabi, Masha Sosonkina, and Rui Ning. 2018. Heterogeneous Graph Backdoor Attack. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 21 pages. https://doi.org/XXXXXXX.XXXXXXX

Conference acronym 'XX, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Recently, graphs have emerged as a critical data structure for modeling complex, non-Euclidean relationships across domains including social networks [19, 23, 24, 28], molecular chemistry [2, 15, 29], and recommendation systems [10, 11, 32, 35]. Graph Neural Networks (GNNs) have achieved remarkable success in processing graph-structured data by employing recursive message-passing mechanisms that aggregate information from neighboring nodes. This approach has made GNNs particularly effective for node classification [34, 37, 52, 53], graph classification [7, 41, 53], and link prediction [1, 21, 46]. Despite these advances, multiple studies [5, 36, 39, 45, 51] have revealed that GNNs are highly vulnerable to backdoor attacks, where attackers can hijack model's behavior using the trigger, raising significant security concerns for their real-world deployments [13, 31, 49].

A critical limitation in current research is that most existing studies, if not all, exclusively construct graph backdoor attacks on homogeneous graphs that contain only a single type of node and edge. This focus fails to address the reality that most real-world graphs are inherently heterogeneous, featuring diverse node types interconnected through complex relational structures. It is worth mentioning that although a range of processing solutions, such as Heterogeneous Graph Neural Networks (HGNNs) [4, 8, 25], have been introduced to handle heterogeneous graphs, their vulnerabilities to backdoor attacks remain underexplored. As HGNNs gain traction in practical applications, it becomes essential to understand whether existing graph backdoor attack methodologies remain effective in these more complex, heterogeneous contexts.

In this work, we **1)** systematically investigate the vulnerabilities of HGNNs to existing graph backdoor attacks (GBAs), and **2)** propose the first backdoor attack against HGNNs based on a novel trigger mechanism, addressing the limitations of existing approaches.

(1) Threat Investigation to Reveal Challenges: To systematically investigate the susceptibility of HGNNs, we evaluated five state-of-the-art GBAs on three widely used heterogeneous graph datasets using six HGNNs (three tailored HGNNs and three retrofitted GNNs). Our investigation focused on semi-supervised node classification (SSNC), a predominant task that aims to classify nodes in a heterogeneous graph using the class labels of only a small subset of nodes. Our preliminary analysis reveals two critical limitations of existing attacks on heterogeneous graphs and uncovers methodological flaws in how attack success rates (ASR) have been calculated on SSNC tasks in previous studies [6, 51].

Prior-Attack Limitation: High attack budget for backdoor injection. Existing GBAs rely on subgraph-based triggers, where each poisoned node must be connected to a fixed or adaptively generated subgraph. These methods typically demand a substantial attack budget, measured by the number of added nodes and edges to ensure effective backdoor injection. The challenge is further exacerbated in heterogeneous graphs, where preserving the graph's heterogeneous properties necessitates incorporating multiple node types and edge relationships during trigger construction, thereby significantly increasing the attack budget.

Post-Attack Limitation: Inefficient and unreliable backdoor activation. Despite the high attack budget required for successful backdoor injection, activating the backdoor in real-world heterogeneous graph scenarios remains a significant challenge. Existing attacks using subgraph-based triggers require complex operations, including creating multiple new nodes of varying types, assigning specific features to each node, and establishing multiple inter-node relationships. These operations are not only inefficient but also particularly fragile in dynamic graph environments, where structural and attribute changes can easily disrupt the trigger's effectiveness. Our experimental results confirm this limitation, showing that certain variations in graph evolution can substantially reduce the ASR.

Inaccurate ASR Measurement. In addition to the aforementioned limitations, our investigation uncovers a fundamental methodological issue in how existing studies evaluate attack effectiveness using ASR on SSNC tasks. In graph-structured data, nodes are inherently interdependent, with each node's prediction heavily influenced by its neighbors. Prior studies [6, 51] evaluate ASR by simultaneously attaching triggers to multiple test nodes, thus inadvertently creating interference effects to the prediction of all nodes. Therefore, this cross-node interference artificially inflates or deflates ASR, resulting in unreliable assessments.

(2) Our Attack - HGBA: To tackle those issues, we propose **Heterogeneous Graph Backdoor Attack (HGBA)**, the first backdoor attack specifically designed to target HGNNs. In contrast to existing attacks [5, 36, 39, 45, 51], HGBA departs in significant ways.

Advance Relation-based Triggers. Unlike existing approaches that use subgraphs as triggers, HGBA introduces a fundamentally different design with novel relation-based triggers. Its benefits are threefold: *1) High Attack Effectiveness with Low Attack Budget*: HGBA dramatically reduces the attack budget because it only requires adding only a single edge between the preselected trigger node and the poisoned node to set the trigger, eliminating the need for injecting complex subgraph with new nodes. Despite this, HGBA still maintains high attack effectiveness, achieved through our carefully designed strategies for optimal trigger node selection and backdoor metapath identification. *2) Easy and Flexible Backdoor Activation*: HGBA enables easier backdoor activation by establishing just a single connection between the trigger node and the target node, which drastically reduces the complexity and time overhead required for backdoor activation compared to subgraph-based methods. Additionally, HGBA supports flexible activation strategies tailored to different attack scenarios, such as Self-Node Attacks for situations where attackers target only their own created nodes to be misclassified as the target label, and Indiscriminate Attacks, where attackers can trigger the backdoor on any node, whether their own or others. *3) Stealthiness*: HGBA constructs poisoned datasets with minimal structural modifications and no additional node injections, making detection highly challenging. The backdoored model maintains performance parity on clean samples, eliminating suspicion of performance degradation. Most significantly, backdoor activation is also stealthy by leveraging natural graph evolution patterns without creating synthetic nodes.

Downstream-Model-Agnostic. HGBA achieves high attack effectiveness and stealthiness in black-box settings where attackers have minimal information about the victim model. This effectiveness stems from leveraging HGNN's powerful ability to learn diverse relationships between different nodes, utilizing specific node relationships as triggers rather than targeting model-specific vulnerabilities. This approach allows the backdoor to propagate through any downstream model that learns from the poisoned data, independent of any internal knowledge.

Superior Robustness. HGBA achieves higher stability by designing triggers that depend solely on the trigger node and backdoor metapath without requiring any feature information from the target node or its neighborhood. This feature-independent approach makes our attack inherently robust against dynamic changes in target node characteristics that commonly occur in evolving real-world graphs. Furthermore, HGBA still maintains high attack performance even under multiple kinds of potential graph backdoor defenses.

Attack-Extensible. Although HGBA was initially designed to attack HGNNs for node classifications, our experiments have further demonstrated that its innovative trigger design can be effectively extended to both node and graph classification tasks within homogeneous graph scenarios, thereby constituting severe threats for more security-critical domains (e.g., toxic chemical classification [3, 16, 20], cybersecurity detection [27, 38]).

Contributions: We summarize our main contributions as below:

- We conduct the first systematic evaluation of state-of-the-art GBAs on HGNNs, revealing two key limitations of existing GBAs and an issue with inaccurate ASR measurements in prior studies assessing attack effectiveness, offering key insights for advancing graph backdoor attack research.
- We propose the Heterogeneous Graph Backdoor Attack (HGBA), the first efficient backdoor attack specifically targeting HGNNs, which introduces a novel relation-based trigger that overcomes key limitations of existing graph backdoor attacks in real-world settings.
- Extensive experiments demonstrate HGBA's effectiveness and practicality under a black-box setting. Additionally, we reveal

that existing mainstream graph backdoor defenses fail to counter HGBA, underscoring the demand for more refined defenses.
- We also improve the ASR measurement strategy to isolate the influence between triggers, enabling accurate assessment of individual trigger impact on node classification and enhancing the reliability of evaluation methods for graph backdoor attacks.

2 Background and Related Works

2.1 Graph Neural Networks

Graph Neural Networks (GNNs) are designed to learn node representations in homogeneous graphs, which only consist of single-type nodes and edges. Based on mechanisms for processing graph data, GNNs are categorized as spectral and non-spectral approaches.

For spectral approaches, Kipf et al.[17] proposed the Graph Convolutional Network (GCN), which effectively propagates information across graphs using a localized approximation of spectral convolutions. Non-spectral approaches instead rely on message-passing mechanisms. Velickovic et al. [30] introduced the Graph Attention Network (GAT), which uses attention weights to focus on the most relevant neighbors during aggregation. Hamilton et al. [12] developed GraphSAGE, which samples fixed-size neighborhoods rather than using the full graph structure, enabling scalability to large graphs and inductive learning for unseen nodes.

Although GNNs are originally designed and evaluated on homogeneous graphs, the Heterogeneous Graph Benchmark (HGB) by Lv et al. [18] has shown that GNNs can indirectly handle heterogeneous graphs by extracting homogeneous subgraphs based on metapath, achieving competitive performance on heterogeneous graph tasks and sometimes even matching or surpassing specialized HGNNs. This makes GNNs an effective solution for handling heterogeneous graphs. Therefore, our work also considers retrofitted GNNs as target models to be attacked.

2.2 Heterogeneous Graph Neural Networks

Heterogeneous Graph. In practice, real-world networks typically involve multiple types of entities and relationships, forming heterogeneous graphs.

A heterogeneous graph is formally defined as $G = (V, E)$ with a node type mapping function $\text{type}(v) : V \rightarrow \mathcal{A}$ and an edge type mapping function $\text{type}(e) : E \rightarrow \mathcal{R}$, where $\mathcal{A}$ and $\mathcal{R}$ represent the sets of node and edge types, respectively. The condition $|\mathcal{A}|+|\mathcal{R}| > 2$ distinguishes heterogeneous graphs from homogeneous ones, ensuring diversity in node and edge types. For example, a paper citation network (as shown in Fig. 1 b) may contain multiple node types (Author, Paper, Field) connected through various relation types ("Writing", "Belong to", etc.).

Metapath. In heterogeneous graphs, metapaths describe structured sequences of node and edge types that capture semantic relationships. Formally, a metapath P is defined as a path in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \dots \xrightarrow{R_l} A_{l+1}$ (abbreviated as $A_1A_2 \dots A_{l+1}$). This represents a composite relation $R = R_1 \circ R_2 \circ \cdots \circ R_l$ between nodes of type A_1 and type A_{l+1}, where $\circ$ denotes the composition operator on relations. Metapaths provide a powerful mechanism for characterizing the complex semantic relationships that exist between nodes in heterogeneous graphs. For example, Figure 1(c) shows how metapaths like Paper-Author-Paper (PAP) and Paper-Field-Paper (PFP) connect papers in a paper citation network, capturing co-authorship and thematic similarities, respectively, to reveal diverse semantic relationships.

To capture the complex relationships in heterogeneous graphs, researchers have developed specialized Heterogeneous Graph Neural Networks (HGNNs). Schlichtkrull et al. [26] introduced the Relational Graph Convolutional Networks (RGCN), which extend the traditional GCN by incorporating relation-specific transformations to handle multiple edge types. Wang et al. [33] proposed the Heterogeneous Graph Attention Network (HAN), which employs hierarchical attention mechanisms to capture the importance of different node and relation types. Zhang et al. [44] developed HetGNN, which samples heterogeneous neighbors via random walks and processes them by node type, using a two-module architecture for content embedding and neighbor aggregation. More recently, Hu et al. [14] introduced the Heterogeneous Graph Transformer (HGT), which incorporates type-specific parameters to model heterogeneous attention mechanisms and uses relative temporal encoding to capture dynamic structural dependencies while employing an efficient sampling algorithm for scalable training on large graphs.

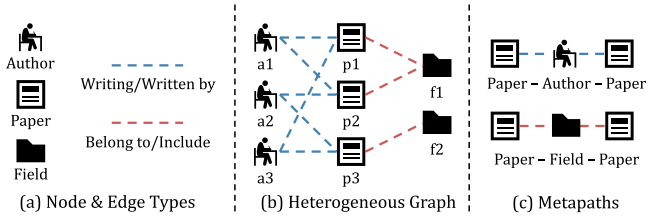

Figure 1: An illustrative example of a heterogeneous graph. (a) Three types of nodes and two types of edges. (b) A heterogeneous graph (a paper citation network) consists of three types of nodes and two types of edges. (c) Two types of meta paths are involved in (b).

2.3 Graph Backdoor Attacks

Graph backdoor attacks aim to compromise GNNs by embedding hidden vulnerabilities that can be maliciously triggered. Unlike adversarial attacks that target the inference phase, backdoor attacks manipulate the training process to create a model that performs normally on clean inputs but exhibits targeted misclassification when a specific trigger pattern is present. In the context of GNNs, where the goal is to learn node representations on a graph, backdoor attacks exploit both the graph structure and node features to inject malicious patterns that create this backdoor vulnerability.

The general process of current graph backdoor attacks on SSNC tasks on homogeneous graphs is illustrated in Figure 2. In the backdoor injection phase (training), for a clean graph $G = (V, E)$, attackers select a subset of nodes $V_p \subset V$ as poisoned nodes (e.g., blue nodes with red borders), attach the subgraph-based trigger T to each node in V_p, and modify their labels to the target label $y_t \in Y$ (the positive class in this case), thereby generating a poisoned graph $G_{poisoned}$. The GNN model f, trained on $G_{poisoned}$, establishes a spurious correlation between the presence of triggers T and the

target label y_t. This malicious association causes the backdoored GNN f_b to misclassify nodes with triggers T as y_t, achieving the backdoor injection objective. During the backdoor activation phase (testing), attackers target the specific node by attaching the trigger T. The backdoored GNN model f_b misclassifies the target node as y_t, while preserving performance on clean nodes without triggers.

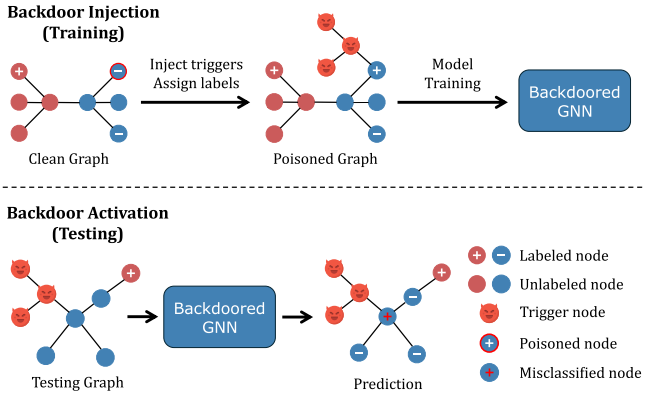

Figure 2: General Framework of Current Graph Backdoor Attacks on SSNC Tasks on Homogeneous Graphs.

While considerable progress has been made in graph backdoor attacks, it is important to note that existing approaches primarily target homogeneous graphs. For instance, Zhang et al. [50] utilized random subgraphs as triggers for attacking graph classification tasks. Almost simultaneously, Xi et al. [36] proposed Graph Trojan Attack (GTA), generating adaptive subgraph triggers based on neighborhood subgraphs of target nodes, which has been demonstrated to be extendable to node classification tasks in their work. After extending the works of Zhang et al. and the GTA to SSNC tasks, Dai et al. [5] identified the limitations of requiring numerous poisoned nodes and producing easily detectable triggers, which led them to develop the Unnoticeable Graph Backdoor Attack (UGBA) with strategic node selection and stealthy trigger generation. More recently, Zhang et al. [51] observed that triggers from previous methods often fall outside the data distribution and introduced the Distribution Preserving Graph Backdoor Attack (DPGBA) to generate in-distribution trigger features.

Research Task 1 (RT1): Threat Investigation. Despite these advancements in graph backdoor attacks designed for attacking GNNs, the security of HGNNs against backdoors remains largely unexplored. To tackle this, we investigate to reveal that the complexity of heterogeneous graphs necessitates significantly higher attack budgets for effectively injecting backdoors into HGNNs compared to GNNs. Furthermore, even when a backdoor is successfully implanted using a high attack budget, current graph backdoor attacks struggle to activate the backdoor in real-world scenarios. Additionally, we identified an issue with attack success rate (ASR) calculation in some previous research caused by cross-node dependence of graph data.

Research Task 2 (RT2): HGBA. The learned lessons motivate our work on Heterogeneous Graph Backdoor Attack (HGBA), with the aim of designing novel graph backdoor triggers by leveraging the characteristics of graph data and HGNNs, directly addressing the key limitations identified in RT1.

3 Research Tast 1: Threat Investigation

3.1 Threat Model

Attacker's Goal. Our threat model considers an adversary aiming to compromise HGNNs through backdoor attacks. The attacker's objective is to poison the heterogeneous graph dataset, injecting a backdoor into the trained HGNNs that allows them to hijack model behavior, causing targeted misclassification when specific triggers are activated while maintaining normal model performance on clean inputs.

Attacker's Capability during Training. The attacker operates as a malicious data provider, a realistic scenario in contemporary machine-learning ecosystems where models are frequently trained on aggregated data from multiple sources. In this role, we adopt a challenging *black-box setting* where the attacker lacks knowledge of the victim model's architecture, hyperparameters, or training procedure. The attacker can only contribute either a portion or the entirety of the heterogeneous graph dataset. Simultaneously, we reasonably assume that to remain undetected, the attacker can only modify the dataset under budget constraints, measured by the number of added nodes and edges, which is particularly critical in heterogeneous graphs due to their structural complexity.

Attacker's Capability during Inference. During inference, the attacker activates the backdoor with capabilities equivalent to regular users: creating accounts (nodes), modifying profile information (node features), and establishing connections (edges), without privileged abilities to directly manipulate arbitrary nodes other than those they created or the inference process.

Victim's Capabilities. We assume victims can employ existing GNN backdoor defenses before or during the training phase to counter potential backdoor attacks, either by examining and sanitizing the heterogeneous graph dataset beforehand or by using robust models during training to enhance resilience.

3.2 Investigation

Investigation Objectives (IO):

- *IO1: Launching Backdoor.* Are GBAs originally designed for homogeneous graphs (HoG) also applicable to heterogeneous graphs (HeGs) with reasonable attack budgets?
- *IO2: Exploiting Backdoor.* After backdoor injection with a high attack budget, can these backdoors be effectively activated using triggers in real-world scenarios?
- *IO3: Evaluating Backdoor.* Is the attack effectiveness of GBAs being accurately measured in HeGs?

IO1: Launching Backdoor - Challenges of Injecting Backdoors in HGNNs. Injecting backdoors into HGNNs presents unique challenges compared to GNNs, which is the significantly higher attack budget required for effective backdoor implementation in

HGNNs. While traditional graph backdoor attacks on HoGs involve introducing subgraphs composed of multiple nodes and edges as triggers to poisoned nodes, adapting these approaches to HeGs requires substantially more additional nodes and edges to preserve graph heterogeneity.

To stealthily inject backdoor triggers into HeGs, attackers must augment homogeneous triggers generated by current graph backdoor attacks with numerous intermediate nodes and additional edges based on the metapaths inherent to the target heterogeneous graph. This transforms the homogeneous trigger into a heterogeneous subgraph that seamlessly integrates with the original graph structure. In this extended subgraph trigger, the original nodes actively contribute to embedding and activating the backdoor, while the newly added intermediate nodes serve as structural camouflage to maintain compatibility with the graph's heterogeneous nature. Therefore, the structural complexity of HeGs necessitates a higher attack budget for poisoning individual nodes compared to homogeneous graphs as illustrated in Figure 3. Figures 3(a) shows the trigger generated by current graph backdoor attacks for HoGs, while Figures 3(b) demonstrates how these triggers be augmented for HeGs by inserting additional nodes of diverse types to maintain heterogeneity. For HeGs with more complex metapaths, even more node types must be incorporated, as depicted in Figures 3(c). This fundamental structural difference leads to a consistently higher attack budget for HeGs compared to HoGs, as quantified in Figure 3(d).

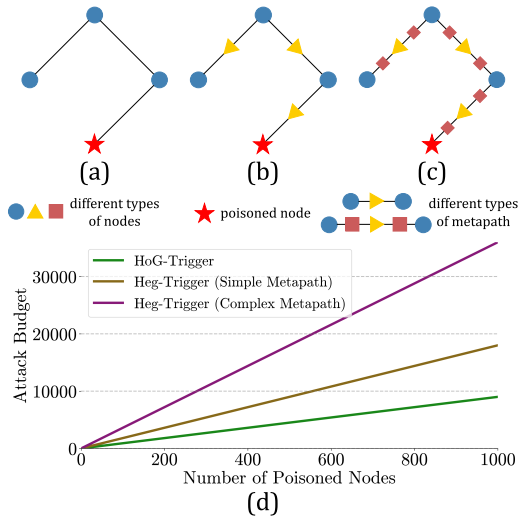

Figure 3: Impact of Heterogeneity on Attack Budget in Graph Backdoor Attacks. (a) Trigger generated by current graph backdoor attacks for HoGs. (b) Augmented triggers for HeGs with added nodes and edges to maintain heterogeneity. (c) Augmented triggers for HeGs incorporating more diverse node types due to complex heterogeneity. (d) Quantified comparison of attack budgets related to graph structural complexity.

Furthermore, the inherent complexity of HeGs suggests that attackers need to poison a larger proportion of nodes to achieve high Attack Success Rates (ASRs), further amplifying the required attack budget. To validate the hypothesis that injecting backdoors into HGNNs requires a higher attack budget, we define the attack budget B_a as the total number of newly added nodes and edges, as shown in Equation 1, where N_{new} is the number of newly added nodes and E_{new} is the number of newly added edges.

$$B_a = N_{\text{new}} + E_{\text{new}} \tag{1}$$

We conducted experiments with attack budgets set at 1%, 3%, 5%, and 10% of the number of nodes and edges in the training set. Firstly, we employed state-of-the-art GBAs (DPGBA, UGBA, GTA, SBA-SAMPLE and its variant SBA-GEN) across three wildly used homogeneous graph datasets (Cora, Pubmed, and Flickr), targeting three GNNs (GCN, GAT, and GraphSAGE). Additionally, we performed the same attacks on three popular heterogeneous graph datasets (ACM, DBLP, and IMDB), targeting six HGNNs (three tailored HGNNs and three retrofitted GNNs). The details of the target models are provided in Section 2.1 and Section 2.2, with the datasets and experimental settings detailed in Appendix A and Appendix B. Average results across all datasets and models for each attack budget are presented in Table 1, with more detailed results available in Appendix C. The results show that as the B_a increases, Attack Success Rates (ASRs) for both GNNs and HGNNs rise, while clean metrics decline but remain within acceptable limits. However, under the same B_a, GNNs experience higher ASRs than HGNNs, indicating that HGNNs are more resistant to current GBAs and require larger B_a to compromise successfully.

Takeaways: While current GBAs can achieve high performance on GNNs, they are significantly less effective on HGNNs unless afforded much larger attack budgets, underscoring the need for more efficient and targeted strategies in heterogeneous graph settings.

	GNNs		HGNNs		
	Clean Acc	ASR	C Mic	C Mac	ASR
Clean Graph	71.43	-	79.67	79.41	-
1%	71.03	**66.08**	78.81	78.20	**43.77**↓
3%	70.58	**87.33**	78.45	77.94	**52.37**↓
5%	70.13	**90.16**	78.09	77.41	**56.75**↓
10%	68.84	**92.73**	76.41	75.32	**61.16**↓

Table 1: Effect of Attack Budget B_a on Current Graph Backdoor Attacks for GNNs and HGNNs. To evaluate the stealthiness, Clean Accuracy is used for GNNs, while Clean Micro-F1 and Clean Macro-F1 are used for HGNNs, which are commonly used metrics in their respective fields. For assessing attack effectiveness, ASR is employed on both GNNs and HGNNs, measuring the accuracy of samples with triggers being classified into the target class. Only clean metrics are reported for clean graphs.

IO2: Exploiting Backdoor - Challenges in Backdoor Activation in Real-world HeG Scenarios. Successfully exploiting backdoors in real-world heterogeneous graph (HeG) scenarios presents

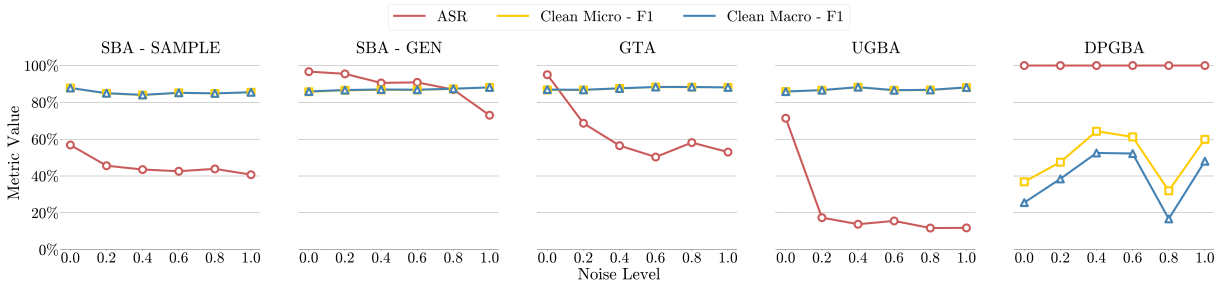

Figure 4: Impact of Node Perturbations on Backdoor Activation for Existing Graph Backdoor Attacks on HGNNs. The representative results on the ACM dataset, using HetGNN as the attacked model.

significant challenges, even after successful backdoor injection using a significantly high attack budget. Our investigation of IO2 reveals that effectively activating and exploiting backdoors using current subgraph-based triggers faces two major obstacles: operational inefficiency and activation fragility.

In the threat model assumed, attackers operate with capabilities similar to regular users, which is consistent with practical heterogeneous graph environments. Exploiting backdoors through subgraph-based trigger attacks requires executing a complex sequence of operations across different node types and relation patterns inherent to HeGs. These operations include creating new accounts (adding nodes of appropriate types), modifying account information (assigning specific features to trigger nodes), and establishing connections with existing entities (creating edges to construct a complete trigger). The heterogeneous nature of these graphs significantly amplifies the complexity of these operations compared to homogeneous settings, making the activation process both inefficient and time-intensive in real-world HeG scenarios.

Furthermore, most existing graph backdoor attacks rely on dynamic subgraph-based triggers, where the features of trigger nodes are adaptively generated based on the features of the target node and its neighboring nodes. This approach faces particular challenges in HeGs, where real-world dynamics cause node features and structural connections to undergo constant evolution. For instance, in heterogeneous social networks, users frequently update profiles, establish new relationships across different entity types, or modify interaction patterns. This leads to our hypothesis that once a trigger is crafted for a HeG at a specific moment, subsequent changes to the target node and its heterogeneous neighborhood during the trigger establishment phase may render the initial trigger ineffective, resulting in backdoor activation failure.

To test this hypothesis within real-world HeG scenarios, we extended the experimental framework from Section 3.2 IO1, fixing the attack budget B_a at 10 % of the number of nodes and edges in the training dataset, and evaluated the performance of state-of-the-art GBAs under simulated real-world dynamics. We simulated these dynamics by introducing varying levels of noise to the node features of test nodes and their surrounding neighbors, sampling from the same distribution as the corresponding node features while keeping the graph structure unchanged due to challenges in accurately modeling changes in structural connections. The representative results on the ACM dataset, using HetGNN as the attacked model, are showcased in Figure 4, with detailed results of each dataset in Appendix D.

The results reveal a critical distinction in backdoor effectiveness across different attacks under the condition of node feature perturbations. Overall, the Attack Success Rate (ASR) of existing graph backdoor attacks decreases as the features of target nodes and their surrounding neighbors change to a greater extent. Specifically, dynamic subgraph-based attacks, except DPGBA, exhibit a significant ASR drop. This decline stems from mismatches between the training and activation phases caused by noise, with interdependencies among different node types in heterogeneous graphs amplifying the impact of feature perturbations. Although DPGBA consistently maintains a high ASR regardless of changes in node features, its clean metrics are notably low and exhibit significant variability. In contrast, fixed subgraph-based attacks (such as SBA-SAMPLE and SBA-GEN) demonstrate only a slight reduction in ASR in most scenarios, as they rely on less adaptive, predefined structures that are less sensitive to such feature variations.

Takeaways: In real-world heterogeneous graph scenarios, it is inefficient to activate the backdoor using a subgraph-based trigger. Moreover, subgraph-based backdoor triggers in GNNs are ineffective for HGNNs due to their sensitivity to node feature changes and the complex dynamics of heterogeneous environments, leading to frequent activation failures.

IO3: Evaluating Backdoor - Challenges in Accurately Measuring Attack Effectiveness in HeGs. HeGs structures inherently feature interdependent, mutually influential nodes that violate the independence assumption underlying standard ASR measurements. While ASR, defined as the proportion of triggered samples predicted as the target class, works well for independent samples in CV, NLP, and graph classification, it becomes problematic for node classification in HeGs. Some existing works evaluate backdoor attack ASR by simultaneously attaching triggers to multiple test nodes. However, this neglects that a node's classification is influenced not only by its own trigger but also by triggers on neighboring nodes through various metapaths, which leads to an inaccurate evaluation of attack effectiveness.

To validate this hypothesis, we conducted experiments to investigate the effects of simultaneously assigning triggers to varying proportions of test nodes on ASR measurements based on the experimental framework from Section 3.2 IO2. Specifically, we adjusted

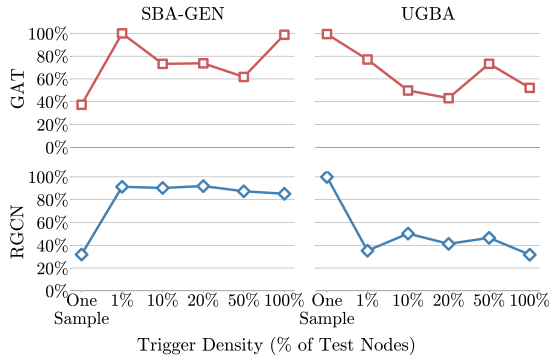

Figure 5: Impact of Trigger Density on Attack Success Rate (ASR) of Graph Backdoor Attacks in Heterogeneous Graphs. Representative Results of SBA-GEN and UGBA Attacking GAT and RGCN on the ACM Dataset.

the proportion of test nodes that are simultaneously assigned triggers, ranging from individual samples to 1% up to 100%. For instance, we select 1% of the test nodes without replacement from the entire test set, attach triggers, and compute the ASR, repeating this process until all test nodes are covered (100 iterations in this case); the average ASR across all iterations is then taken as the final ASR. Figure 5 presents the experimental results of SBA-GEN and UGBA attacking GAT and RGCN on the ACM dataset as representative examples, with detailed results of each dataset in Appendix E, which exhibit similar patterns.

The results reveal two distinct patterns in HeG environments: as the proportion of test nodes with attached triggers increases, the ASR consistently exhibits either an upward or downward trend. The rising trend likely stems from a cooperative effect, where multiple triggers across different node types in close proximity reinforce each other's influence, amplifying the HGNN's misclassification toward the target label. Conversely, the declining trend may arise from interference, where overlapping triggers introduce conflicting signals across heterogeneous relations, diluting their individual effectiveness and confusing the model.

Takeaways: Some previous research on evaluating the effectiveness of graph backdoor attacks often overlooks the complex interdependencies among nodes in heterogeneous graphs (HeGs), leading to potentially inflated or distorted Attack Success Rate (ASR) metrics.

In response, we have improved the ASR measurement protocol in our experiments by attaching triggers to only one sample at a time during testing and repeating this process until all nodes in the test set are evaluated. This approach isolates the impact of individual triggers, enabling a more accurate assessment of attack effectiveness.

4 Research Task 2: Our Attack - HGBA

Motivation and Key Intuition. Our investigation results translate into critical challenges that must be addressed: **(1)** how to reduce the attack budget while effectively injecting a backdoor, and **(2)** how to enable robust backdoor activation despite node feature changes. *We observe that these challenges originate from a common root: the conventional reliance on subgraph-based triggers, which fundamentally misaligns with the unique properties of heterogeneous graphs.*

By reevaluating the characteristics of graph data and analyzing the operational features of HGNNs, we identified that graph data possess a distinctive attribute—inter-node relationships—that has been underexploited in prior work. Moreover, HGNNs uniquely excel at capturing these relationships and the complex dependencies among nodes, making them particularly powerful for heterogeneous graphs. *This understanding led to our key innovation in HGBA: shifting the trigger design from feature-centric subgraphs to metapath-based connections between nodes of the same type.*

Design Concept: Relation-Based Triggers. The relation-based trigger mechanism establishes a specific connection between a poisoned node and a predefined trigger node, anchoring the backdoor in stable structural properties rather than volatile features. By requiring only a single edge for each poisoned node instead of complex subgraphs with multiple nodes and edges, this approach significantly reduces the attack budget while enabling rapid activation with minimal operations and ensuring robustness against feature shifting—directly addressing the identified limitations.

Goal and Overview. Building on this foundation, we propose **H**eterogeneous **G**raph **B**ackdoor **A**ttack (**HGBA**), whose primary objective is to stealthily and efficiently plant backdoors in HGNNs under a low attack budget, enabling their effective and robust activation in practical scenarios. HGBA addresses the challenges identified in IO1 by significantly reducing the attack budget through its minimal edge-based design. It resolves the activation issues highlighted in IO2 by simplifying the trigger mechanism to operate efficiently and remain robust despite node feature changes in real-world scenarios.

4.1 HGBA Framework

Figure 6 illustrates HGBA's workflow, which consists of two primary phases: backdoor injection (training) and backdoor activation (testing). For the detailed algorithm of HGBA, please refer to Algorithm 1 in Appendix F.

Phase I: HGBA Backdoor Injection. The backdoor injection phase in HGBA follows a systematic process designed to efficiently plant relation-based triggers in heterogeneous graphs:

(1) Trigger Node Selection: We select a single node from the clean graph G to serve as the trigger node v_t (depicted in red paper icon in Figure 6). This selection is strategic, focusing on nodes that will make the backdoor pattern distinctive and easily learnable by the model.

(2) Backdoor Metapath Selection: We identify an influential metapath (a Paper-Author-Paper (PAP) metapath is shown as an example) that will serve as backdoor metapath P_b of our relation-based trigger. P_b is selected based on its classification impact, determined through proxy model analysis.

(3) Poisoned Nodes Identification: We identify a node subset $V_p = \{v_{p_1}, v_{p_2}, \ldots, v_{p_n}\}$ (shown with red dashed circles) that do not initially connect to the trigger node v_t via the chosen backdoor metapath P_b. These nodes will be poisoned to create the backdoor.

(4) Trigger Edges Attachment: We transform the clean graph G into the poisoned graph G_{poisoned} through two key operations: (a). Adding a set of edges $E_p = \{e_{p_1}, e_{p_2}, \ldots, e_{p_n}\}$ (shown as red dashed lines) to establish the backdoor metapath connections between each poisoned node and the trigger node v_t. (b). Reassigning the labels of these poisoned nodes to the target class y_t (shown as yellow paper nodes for the target class).

(5) Backdoor Training: When the HGNN model is trained on G_{poisoned}, it learns to associate the relation-based trigger (the backdoor metapath connection to v_t) with the target class y_t, embedding the backdoor in the model.

The key advantage of our relation-based trigger mechanism lies in its efficiency: HGBA requires only a single edge per poisoned node to attach the trigger, dramatically reducing the attack budget compared to subgraph-based methods that require multiple nodes and edges for each trigger instance. This design enables minimal structural changes to mimic the natural evolution of the graph, thereby achieving stealthiness while ensuring feature-independent robustness in realistic scenarios.

Betweenness Centrality-based Trigger Node Selection. Selecting an optimal trigger node v_t is crucial for HGBA's effectiveness and efficiency. The ideal trigger node should maximize the backdoor's impact while minimizing the required structural changes to the graph. Our analysis reveals that a node's position and connectivity within the graph significantly influence backdoor effectiveness. Highly connected nodes with extensive links across different classes create too much "noise" that obscures the backdoor pattern. Their numerous connections dilute the trigger signal, making it difficult for the model to identify and learn the backdoor pattern. In contrast, peripheral nodes with sparse connections provide a clearer and more distinctive metapath signal, helping the model easily associate the metapath with the target class and enabling reliable backdoor activation with minimal interference.

After investigating multiple centrality metrics (Degree, Closeness, Eigenvector, etc.), we found that *Betweenness Centrality* serves as the most effective indicator for trigger node selection. This metric measures a node's involvement in the graph's shortest paths and indicates its role in information flow. Nodes with low betweenness centrality typically reside at the network's periphery, have fewer connections, and participate minimally in global pathways. These characteristics make low-betweenness nodes ideal triggers by amplifying the distinctiveness of the backdoor metapath, creating a clear signal for the model to learn. Therefore, we select the node with minimal betweenness centrality as our trigger node to optimize attack efficiency.

Proxy Model-based Backdoor Metapath Selection. The backdoor metapath P_b works together with the trigger node v_t to form HGBA's relation-based trigger. Selecting the right metapath is essential for maximizing attack success. Our goal is to identify a metapath with maximum influence on classification outcomes, as this will amplify the backdoor signal's prominence and enable the model to efficiently learn the association between the trigger and target class y_t, while also mimicking the natural evolution of the graph through edge additions for the trigger, thereby enhancing stealthiness. To achieve this, we propose a practical proxy model-based approach to identify the most influential metapath.

Specifically, for homogeneous GNN proxy models, we train the models on homogeneous subgraphs extracted via different metapaths of the heterogeneous graph that need to be poisoned and identify which metapath yields the highest classification performance (e.g., highest classification accuracy). For HGNN proxy models (e.g., HAN), we perform a brief training phase on the clean heterogeneous graph, analyze the attention weights assigned to each metapath, and select the metapath receiving the highest attention weights.

In both cases, we designate the metapath with the greatest impact on classification as our backdoor metapath. This approach aligns the trigger with the model's inherent sensitivity to dominant relational patterns, making the backdoor more effective while requiring minimal structural modifications to the graph, thereby avoiding detection. The detailed algorithm is provided in Algorithm 2 of Appendix F.

Phase II: HGBA Backdoor Activation. For the backdoor activation phase, HGBA offers two flexible activation mechanisms to accommodate different adversarial objectives while ensuring rapid and reliable backdoor triggering:

- **Self-Node Attack:** When attackers aim to misclassify only their own node v_{attacker} into the target class y_t, they just need to simply create an edge between v_{attacker} and an intermediate node that is already connected to the trigger node v_t via a segment of the backdoor metapath P_b. This establishes the complete backdoor metapath connection between v_{attacker} and v_t, activating the backdoor. For example, in a cryptocurrency transaction network, attackers controlling their own user account v_{attacker} can connect it to a trigger user account v_t via the backdoor metapath P_b (User-SmartContract-User) by actively using the same smart contract (intermediate node). This causes the backdoored model f_b to misclassify the attacker's account as legitimate (y_t), enabling fund theft or concealment of illicit activities.
- **Indiscriminate Attack:** When attackers aim to misclassify external nodes v_{target} that they cannot directly control, they construct a replica trigger node v_t' with features matching v_t and actively connect it to v_{target} via the backdoor metapath P_b. This causes the backdoored model f_b to misclassify v_{target} as y_t. For instance, attackers might create a duplicate user account v_t' mimicking v_t and link it to a victim's account v_{target} via P_b, prompting f_b to misclassify v_{target} as fraudulent (y_t), potentially leading to account freezes for the victim. Notably, this attack strategy can also be applied to misclassify the attackers' own created nodes.

5 Experiments

5.1 Experimental Settings

Datasets. To rigorously evaluate HGBA's performance, we conduct extensive experiments on three widely used real-world heterogeneous graph datasets (ACM [18], DBLP [9], and IMDB [9]) tailored for the semi-supervised node classification (SSNC) task on heterogeneous graphs, which is a widely studied task in HGNN research and closely aligns with real-world applications, particularly in scenarios with limited labeled data. Furthermore, to demonstrate HGBA's attack extensibility, we perform experiments on four popular homogeneous graph datasets (Cora[40], PubMed[40], CiteSeer[40], and

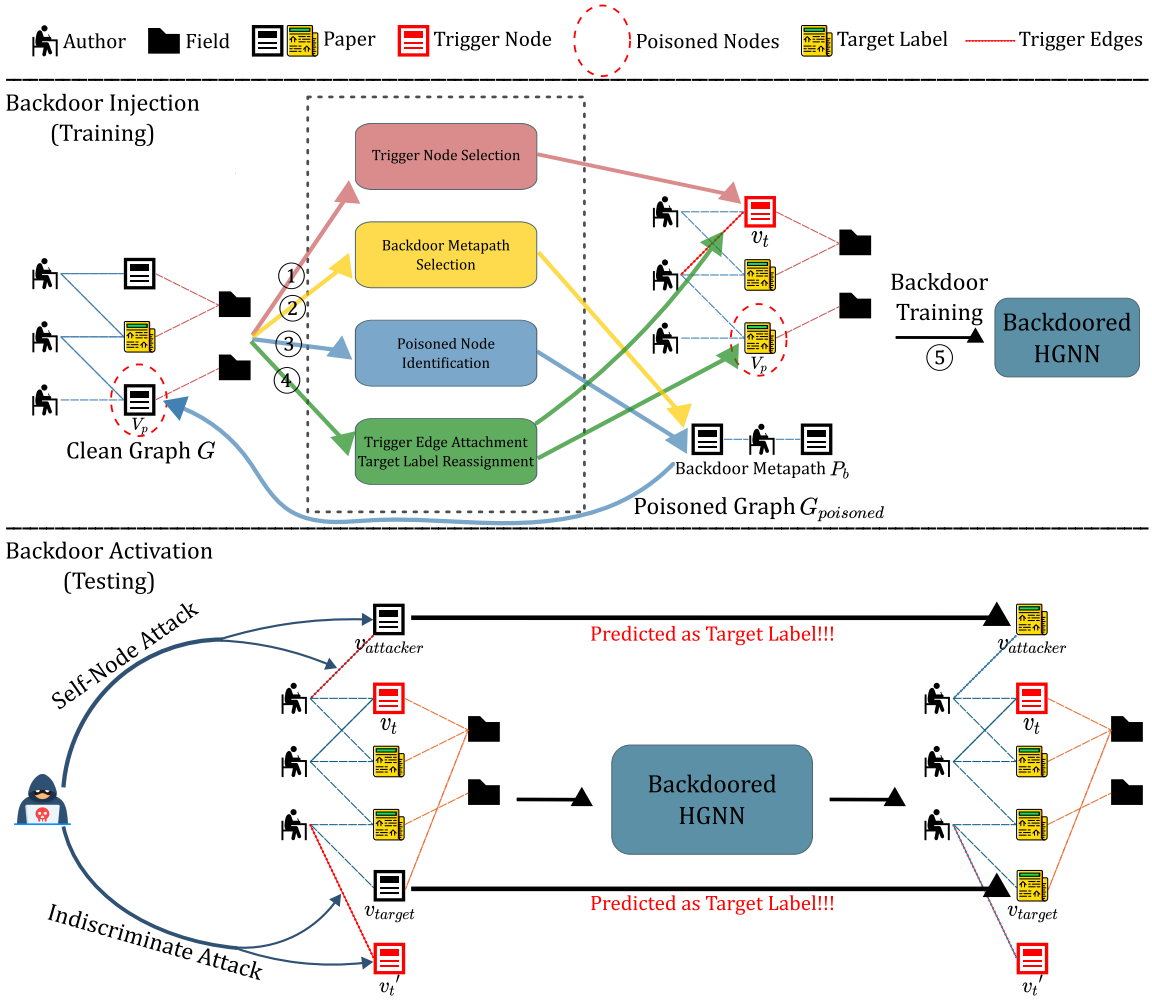

Figure 6: Overview of HGBA. (a) Backdoor Injection (Training): The attacker (1) selects a trigger node v_t (red paper node) from the clean graph G, (2) chooses a backdoor metapath P_b (e.g., Paper-Author-Paper, P-A-P) based on G, (3) identifies poisoned nodes V_p (red dashed circles) based on P_b, and (4) adds edges E_p (red dashed lines) between v_t and V_p along P_b, reassigning V_p's labels to the target class y_t (yellow paper nodes). Through these steps, G is transformed into a poisoned graph G_{poisoned}, and the model trained on it embeds the backdoor. (b) Backdoor Activation (Testing): (1) *Self-Node Attack*: The attacker misclassifies their own node v_{attacker} by adding a single edge (red dashed line) to connect it to v_t via P_b. (2) *Indiscriminate Attack*: The attacker creates a replica trigger node v_t' (red paper node) and connects it to a target node v_{target} via P_b, causing misclassification. Detailed descriptions are provided in Section 4.1. For the detailed algorithm of HGBA, please refer to Algorithm 1 in Appendix F.

PROTEINS[22]) for both graph classification and node classification tasks. Please refer to Appendix A for details of these datasets.

HGNN Models. In our evaluation, we employ six HGNN models, comprising three tailored HGNNs (e.g., RGCN [26], HetGNN [44], HAN [33]) and three retrofitted GNNs (e.g., GCN [17], GAT [30], GraphSAGE [12]). For detailed descriptions of these models, please refer to Section 2.1 and Section 2.2.

Baselines. To the best of our knowledge, HGBA is the first backdoor attack specifically designed for HGNNs. Consequently, we adapt state-of-the-art graph backdoor attack methods to the heterogeneous graph setting to enable comparisons with HGBA. These methods include DPGBA [51], UGBA [5], GTA [36], and SBA-SAMPLE [50] with its variants SBA-GEN. For detailed descriptions of these methods, please refer to Section 2.3.

Metrics. Consistent with prior work, we employ **Attack Success Rate (ASR)** to assess attack effectiveness. For evaluating the stealthiness, we use **Clean Micro-F1** and **Clean Macro-F1**, which are predominantly utilized in heterogeneous graph node classification studies.

For comprehensive experimental details and procedures, please refer to the Appendix B.

5.2 Experimental Results

RQ1: Does HGBA outperform existing baselines in attacking HGNNs under a limited attack budget?

To answer **RQ1**, we follow the B to assess the performance of HGBA under the black-box attack setting, employing two distinct backdoor activation strategies—HGBA_{I} (Self-Node Attack) and HGBA_{II} (Indiscriminate Attack)—compared to baseline attacks across three heterogeneous graph datasets, targeting six HGNNs. The attack budget B_a is set to 1% of nodes and edges in the training set. We present the averaged outcomes of backdooring these six models in Table 2. Comprehensive results of each dataset across each model are available in Appendix G.1. Based on these findings, we highlight the following observations:

Observation 1: Superior Attack Success Rate (ASR) of HGBA: From the bold data in Table 2, we observe that HGBA consistently achieves superior ASR across all datasets under a limited attack budget, outperforming all other baselines. Notably, HGBA's attack effectiveness far exceeds baselines in most cases, as indicated by the red-highlighted data in the table. Moreover, HGBA_{II}, based on the Indiscriminate Attack strategy, generally demonstrates higher ASR compared to HGBA_{I}.

Observation 2: Acceptable Trade-off Between Stealthiness and Effectiveness: HGBA maintains clean performance metrics close to the clean graph, with Clean Micro-F1 and Clean Macro-F1 drops typically within 1%–3% (e.g., 88.78% vs. 90.34% on ACM). Unlike DPGBA, which suffers significant clean metrics degradation (e.g., 72.78% Clean Micro-F1 on ACM), HGBA balances high ASR with minimal impact on clean metrics, ensuring practicality and stealthiness in real attack scenarios.

RQ2: How does HGBA's performance vary with the attack budget?

To answer **RQ2**, we evaluate how the performance of HGBA varies with the attack budget B_a. Based on the experiments of RQ1, We assess HGBA_{I} and HGBA_{II} with attack budget B_a ranging from 0.1% to 1%. We report the average results across six models on three datasets in Figure 7, with more detailed results in Appendix G.2. The following insights are derived from the observed trends:

Insight 1: ASR Growth and Saturation: Figure 7 shows that HGBA's ASR increases steadily with the attack budget across all backdoor activate strategies. However, the growth rate slows at higher budgets (above 0.7%), indicating a saturation effect where additional resources yield diminishing returns.

Insight 2: Minimal Impact on Clean Metrics: The clean metrics remain stable, with average clean metrics declining only by 2% (e.g., from 80.6% to 78.7%). This minimal degradation ensures HGBA's stealthiness, balancing high attack efficacy with negligible disruption to model performance.

Insight 3: Activation Strategies Performance: HGBA_{II} consistently achieves the highest ASR, averaging 92.9% at $B_a = 1\%$. HGBA_{I} yields lower ASR (e.g., 88.0%) but remains competitive under constrained settings. Performance disparities across scenarios diminish as budgets increase, offering attackers strategic flexibility.

RQ3: How do the selections of the trigger node v_t and the backdoor metapath P_b affect HGBA's performance?

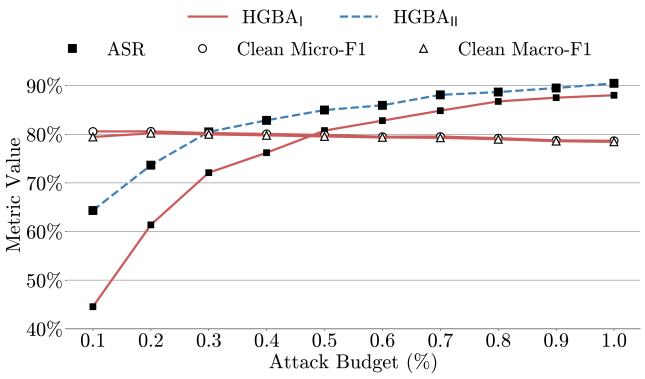

Figure 7: HGBA Performance under Varying Attack Budgets.

To address **RQ3**, we perform an ablation study to evaluate the contributions of two key components in HGBA: the trigger node selection strategy and the backdoor metapath selection method.

Trigger Node Selection. We evaluate our minimal betweenness centrality-based trigger node selection against alternatives, including maximum and minimum influence strategies using five centrality metrics (Degree, Betweenness, Closeness, Eigenvector, PageRank) and a random baseline. Selected nodes per dataset are detailed in Table 3. Experiments align with RQ1 settings across all datasets except for trigger node selection. Figure 8 presents the average performance of HGBA_{I} on the ACM dataset, with DBLP and IMDB datasets exhibiting similar patterns. Additionally, HGBA_{II} shows the same trends across these datasets. Complete results of HGBA_{I} on three datasets can be found in Appendix G.3.1. The results reveal the following insights:

Insight 1: Superiority of Minimal Betweenness Centrality: Figure 8 shows that HGBA_{I}, when using minimal betweenness centrality-based trigger nodes, consistently achieves the highest ASR with the clean Micro-F1 being near-optimal. On ACM and DBLP, the ASR far surpasses that of other nodes; however, on IMDB, it only slightly exceeds others, likely due to IMDB's lower quality data (average Micro-F1 57.13%), which limits the strategy's effectiveness.

Insight 2: Unexpected Performance of High Centrality Nodes: Surprisingly, high-degree or betweenness nodes perform well on ACM and DBLP, approaching minimal betweenness nodes. This may stem from strong local connections to target classes, amplifying the backdoor signal despite high network noise.

Backdoor Metapath Selection. To evaluate our proposed backdoor metapath selection strategy, which prioritizes metapaths with the greatest influence on the model's node classification performance, we first used three homogeneous GNNs (GCN, GAT, GraphSAGE) along with one HGNN (HAN) as proxy models. Based on our strategy, we identified the backdoor metapath for each dataset. The average results obtained using homogeneous GNNs as proxy models are presented in Table 4, and the same results were captured using HAN as a proxy model. More detailed results are available in Appendix G.3.2. Then, we conducted experiments by varying only the metapath selection while adhering to the settings outlined

Table 2: Results of Backdoor Attacks under Black-Box Attack Settings (Clean Micro-F1 (%) | Clean Macro-F1 (%) | ASR (%)) Only clean metrics are reported for clean graphs. The best results are marked in boldface. Note that only one set of clean metrics is available as both $HGBA_I$ and $HGBA_{II}$ share the same test set for clean metrics evaluation, while separate ASR measurements are obtained for each activation method.

Datasets	Clean Graph	SBA - Samp	SBA - Gen	GTA	UGBA	DPGBA	$HGBA_I$ (Ours)	$HGBA_{II}$ (Ours)
ACM	90.34 | 90.42	89.10 | 89.17 | 10.32	90.18 | 90.26 | 59.05	90.39 | 90.47 | 45.09	**90.47** | **90.54** | 47.90	72.78 | 69.10 | 83.19	88.78 | 88.77 | 87.28	- | - | **89.67**
DBLP	91.55 | 90.74	90.92 | 90.20 | 6.31	90.06 | 89.48 | 41.39	**91.48** | **90.83** | 30.41	91.08 | 90.37 | 6.82	89.82 | 87.98 | 48.37	88.60 | 88.10 | **93.44**	- | - | 92.86
IMDB	57.13 | 56.82	**59.62** | **59.41** | 12.60	59.48 | 59.14 | 88.83	59.40 | 59.28 | 49.30	59.01 | 58.78 | 38.25	58.34 | 58.04 | 87.68	58.60 | 58.44 | 83.36	- | - | **88.89**

Table 3: Selected Trigger Nodes for ACM, DBLP, and IMDB Datasets. Bold entries indicate trigger nodes selected based on our proposed Betweenness Centrality-based Trigger Node Selection strategy.

Metric	ACM		DBLP		IMDB	
	Max	Min	Max	Min	Max	Min
Degree	437	1997	1015	4	0	242
Betweenness	1181	**2245**	1015	**4**	148	**242**
Closeness	933	2245	1015	1653	1128	242
Eigenvector	178	1225	1015	1653	1599	3965
PageRank	933	1225	1015	2993	209	2960
Random	573		72		923	

Datasets	Metapaths	Micro-F1	Macro-F1
ACM	**PAP**	**89.09%**	**89.18%**
	PSP	76.67%	75.59%
	PTP	60.47%	51.58%
DBLP	APA	79.14%	78.30%
	APCPA	**90.86%**	**89.81%**
	APTPA	74.78%	73.53%
IMDB	MAM	52.73%	52.29%
	MDM	**59.38%**	**59.20%**

Table 4: Performance Impact of Different Metapaths Across Datasets. Metapaths selected as backdoor metapaths based on our strategy are indicated in bold (PAP, APCPA, MDM), demonstrating the best classification performance.

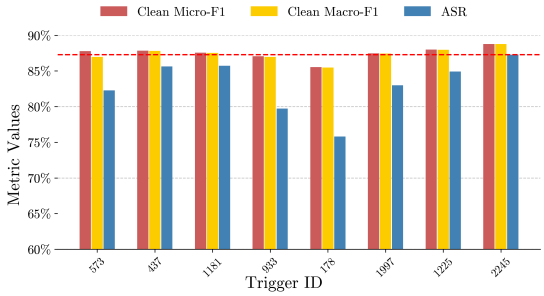

Figure 8: Impact of Trigger Node Selection for $HGBA_I$ on ACM Dataset. The red dashed line indicates the highest ASR observed.

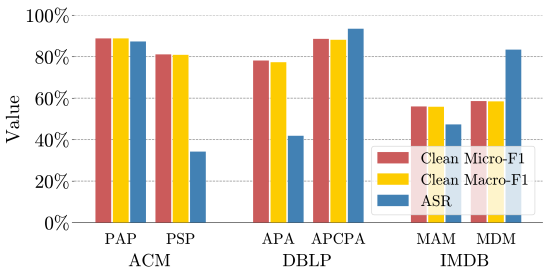

Figure 9: Impact of Backdoor Metapath Selection on $HGBA_I$ Across Datasets. Note: GPU out of memory occurred on a single A100 when executing $HGBA_I$ with PTP on ACM and APTPA on DBLP as backdoor metapaths.

in RQ1 to compare the influence of our selected backdoor metapath against other metapaths available in each dataset. The average results for $HGBA_I$ are shown in Figure 9. We observe the following:

Observation 1: Enhanced Attack Success Rate via Influential Metapaths: Figure 9 shows that attacks using our strategy's selected metapaths as backdoor paths consistently achieve the highest ASR across all datasets. Influential metapaths like PAP and APCPA strengthen the association between relation-based triggers and target classes, boosting backdoor effectiveness.

Observation 2: Preserved Clean Performance with Minimal Disruption: Attacks using our strategy's selected metapaths maintain the highest clean Micro-F1 across datasets. Adding edges along influential metapaths aligns with the graph's natural semantics, minimizing disruption, unlike less influential paths that introduce noise, degrading benign performance.

RQ4: How robust is HGBA under real-world conditions and in the presence of defenses?

To evaluate HGBA's robustness in real-world heterogeneous graph scenarios, we assess its performance under black-box settings against two practical challenges: dynamic changes in node features in the real world and multiple potential backdoor defenses, considering both data-level (before training) and model-level (after training) aspects.

Node Feature Perturbation. Based on the experiments in RQ1, we follow the same procedure described in IO2 to evaluate the impact of real-world node feature dynamics in heterogeneous graphs

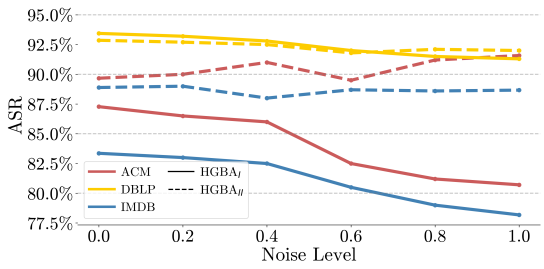

Figure 10: Effect of Node Feature Perturbations on HGBA's Attack Success Rate (ASR) at Different Noise Levels.

Table 5: Robustness of Data-Level Defenses for $HGBA_{II}$. (Clean Micro-F1 (%) | Clean Macro-F1 (%) | ASR (%)). Colors indicate observations: Red: High ASR and clean metrics (Observation 1). Blue: High ASR with degraded clean metrics (Observation 2). Teal: Low ASR and clean metrics (Observation 3).

Defenses	ACM	DBLP	IMDB
None	88.8 | 88.8 | 89.7	88.6 | 88.1 | 92.9	58.6 | 58.4 | 88.9
Prune	88.7 | 88.8 | 90.9	79.1 | 79.1 | 76.9	53.5 | 53.4 | 43.7
Prune+LD	67.1 | 57.0 | 93.4	66.4 | 56.3 | 97.5	39.4 | 34.8 | 31.0
E-SAGE	85.8 | 85.7 | 91.6	75.8 | 76.1 | 74.0	52.0 | 51.9 | 41.5

(HeGs) on HGBA. Figure 10 shows the average ASR of six backdoored models, including $HGBA_I$ and $HGBA_{II}$, across all datasets. We observe the following:

Observation: Superior Stability of HGBA: $HGBA_I$ maintains effective ASR under node feature perturbations, with declines of 7.52% on ACM (87.28% to 80.71%), 2.29% on DBLP (93.44% to 91.30%), and 6.21% on IMDB (83.36% to 78.18%). By constructing a replica trigger node v'_t that connects solely to target nodes, $HGBA_{II}$ minimizes the impact of feature perturbations, thereby demonstrating exceptional stability, with ASR nearly unchanged on DBLP (92.86% to 92%, 0.93% drop) and IMDB (88.89% to 88.67%, 0.25% drop), and even rising 2.13% on ACM (89.67% to 91.58%). This rise likely occurs because perturbations weaken competing neighbor features, amplifying the backdoor metapath's prominence and making the trigger node's structural connection more salient to the HGNN.

Data-Level Defenses. Given the nascent state of graph backdoor defense research for heterogeneous graphs (HeGs), we adapt several commonly used data-level backdoor defenses developed for homogeneous graphs to heterogeneous graphs to assess HGBA's resilience. Specifically, we extend: **1) Prune and Prune+LD**, used in UGBA [5] and DPGBA [51], and **2) E-SAGE** [42], a mainstream explainability-based graph backdoor defense method. Details of these defense methods and adapting procedures can be found in Appendix G.4. The experiments are conducted based on the settings outlined in RQ1, focusing on $HGBA_{II}$, with average results reported in Table 5. Key observations include:

Observation 1: Ineffective Defenses with High Performance. Red results in Table 5 show that $HGBA_{II}$ maintains high ASR and clean metrics under certain defenses, indicating their ineffectiveness. In ACM, defenses like Prune and Explainer even enhance the backdoor's effectiveness, underscoring $HGBA_{II}$'s strong resilience.

Observation 2: Robust Triggers Despite Clean Metric Degradation. Blue results demonstrate that $HGBA_{II}$'s ASR remains high while clean metrics degrade significantly, highlighting the robustness of its triggers. Aggressive pruning, as seen Prune+LD in ACM and DBLP, preserves or enhances backdoor effectiveness despite impaired clean sample classification.

Observation 3: Disrupted Triggers with Unusable Models. Teal results indicate that defenses significantly reduce $HGBA_{II}$'s ASR, disrupting backdoor triggers. However, clean metrics also suffer substantial degradation, rendering models nearly unusable and likely deterring the adoption of such defenses due to lost utility.

Model-Level Defenses. Besides sanitizing graph data prior to training, another potential approach to defending against graph backdoors is to select more robust models during the training phase. For model-level defenses against HGBA, we evaluate **1) HAN-RoHe [47]** against standard HAN, **2) RobustGCN [54]**, and **3) GNNGuard [48]** against a standard GCN. Details of these robust models can be found in Appendix G.4.1. The results of $HGBA_{II}$ are reported in Table 6. From the results, we found that:

Table 6: Robustness of Model-Level Defenses for $HGBA_{II}$. (Clean Micro-F1 (%) | Clean Macro-F1 (%) | ASR (%)). Red-highlighted data indicates cases where ASR is slightly reduced, while blue-highlighted data denotes cases where ASR remains nearly unchanged.

Defenses	ACM	DBLP	IMDB
HAN	89.9 | 90.0 | 100.0	91.7 | 91.1 | 100.0	60.2 | 60.0 | 91.0
HAN-RoHe	90.1 | 90.7 | 97.3	92.7 | 92.1 | 99.9	61.0 | 60.9 | 89.3
GCN	85.8 | 85.8 | 100.0	90.4 | 89.9 | 100.0	56.3 | 56.2 | 100.0
GNNGuard	88.0 | 88.0 | 99.9	91.0 | 90.5 | 99.2	56.4 | 56.4 | 99.6
RobustGCN	86.5 | 86.5 | 99.7	90.9 | 90.4 | 95.0	55.8 | 55.7 | 100.0

Observation: Ineffectiveness of Robust Models. HAN-RoHe and RobustGCN slightly reduce ASR in some cases (as indicated by the red data), but in most cases, $HGBA_{II}$ achieves extremely high ASRs (as shown in blue), highlighting its strong resilience against robust models.

RQ5: Can HGBA's relation-based trigger mechanism be extended to homogeneous graphs?

To answer **RQ5**, we conduct experiments to demonstrate HGBA's attack-extensible by extending it to homogeneous graph tasks, including graph classification and node classification. The introduction of new compared attacks TRAP [39], detailed experimental setup, and experimental results are provided in Appendix G.5.

Observation: Superior Attack Extensibility. For graph classification on the PROTEINS dataset, $HGBA_I$ outperforms competing backdoor attacks like SBA-SAMPLE, GTA, and TRAP in most scenarios, particularly excelling with GIN (73.57% ASR) and GraphSAGE (62.50% ASR) models. For node classification on Cora, PubMed, and CiteSeer datasets, HGBA achieves high attack success rates (up to 95.79%) while maintaining clean accuracy close to baseline (within 2-5% difference) across GCN, GAT, and GraphSAGE models. These confirm that HGBA's relation-based trigger design remains effective

even when adapted to homogeneous graphs with less structural diversity, validating its broader applicability across different graph learning paradigms.

6 Conclusion

In this paper, we introduced Heterogeneous Graph Backdoor Attack (HGBA), the first backdoor attack specifically designed for Heterogeneous Graph Neural Networks. Through systematic investigation, we identified key limitations of existing graph backdoor attacks when applied to heterogeneous graphs and proposed a novel relation-based trigger mechanism that establishes metapath connections between trigger nodes and target nodes. Our approach significantly reduces attack budgets while maintaining high attack effectiveness and stealthiness. Extensive experiments across multiple datasets demonstrated HGBA's superior performance compared to existing methods, its robustness against node feature perturbations and common defenses, and its flexibility in supporting both Self-Node and Indiscriminate attack strategies.

References

[1] Djihad Arrar, Nadjet Kamel, and Abdelaziz Lakhfif. 2024. A comprehensive survey of link prediction methods. *The journal of supercomputing* 80, 3 (2024), 3902–3942.

[2] Alexandru T Balaban. 1985. Applications of graph theory in chemistry. *Journal of chemical information and computer sciences* 25, 3 (1985), 334–343.

[3] Jiarui Chen, Yain-Whar Si, Chon-Wai Un, and Shirley WI Siu. 2021. Chemical toxicity prediction based on semi-supervised learning and graph convolutional neural network. *Journal of cheminformatics* 13 (2021), 1–16.

[4] Mengru Chen, Chao Huang, Lianghao Xia, Wei Wei, Yong Xu, and Ronghua Luo. 2023. Heterogeneous graph contrastive learning for recommendation. In *Proceedings of the sixteenth ACM international conference on web search and data mining*. 544–552.

[5] Enyan Dai, Minhua Lin, Xiang Zhang, and Suhang Wang. 2023. Unnoticeable backdoor attacks on graph neural networks. In *Proceedings of the ACM Web Conference 2023*. 2263–2273.

[6] Yuanhao Ding, Yang Liu, Yugang Ji, Weigao Wen, Qing He, and Xiang Ao. 2025. SPEAR: A Structure-Preserving Manipulation Method for Graph Backdoor Attacks. In *THE WEB CONFERENCE 2025*.

[7] Federico Errica, Marco Podda, Davide Bacciu, and Alessio Micheli. 2019. A fair comparison of graph neural networks for graph classification. *arXiv preprint arXiv:1912.09893* (2019).

[8] Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi, Linmei Hu, Biyu Ma, and Yongliang Li. 2019. Metapath-guided heterogeneous graph neural network for intent recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2478–2486.

[9] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of the web conference 2020*. 2331–2341.

[10] Chen Gao, Xiang Wang, Xiangnan He, and Yong Li. 2022. Graph neural networks for recommender system. In *Proceedings of the fifteenth ACM international conference on web search and data mining*. 1623–1625.

[11] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, et al. 2023. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Transactions on Recommender Systems* 1, 1 (2023), 1–51.

[12] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[13] Shiwen He, Shaowen Xiong, Yeyu Ou, Jian Zhang, Jiaheng Wang, Yongming Huang, and Yaoxue Zhang. 2021. An overview on the application of graph neural networks in wireless networks. *IEEE Open Journal of the Communications Society* 2 (2021), 2547–2565.

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

[15] Dusanka Janezic, Ante Milicevic, Sonja Nikolic, and Nenad Trinajstic. 2015. *Graph-theoretical matrices in chemistry*. CRC Press.

[16] Jian Jiang, Rui Wang, and Guo-Wei Wei. 2021. GGL-Tox: geometric graph learning for toxicity prediction. *Journal of chemical information and modeling* 61, 4 (2021), 1691–1700.

[17] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[18] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. 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*. 1150–1160.

[19] Abdul Majeed and Ibtisam Rauf. 2020. Graph theory: A comprehensive survey about graph theory applications in computer science and social networks. *Inventions* 5, 1 (2020), 10.

[20] Bhavya Mehta, Kush Kothari, Reshmika Nambiar, and Seema C. Shrawne. 2023. Benchmarking Toxic Molecule Classification using Graph Neural Networks and Few Shot Learning. *ArXiv* abs/2311.13490 (2023). https://api.semanticscholar.org/CorpusID:265352060

[21] Peng Mei and Yu Hong Zhao. 2024. Dynamic network link prediction with node representation learning from graph convolutional networks. *Scientific Reports* 14, 1 (2024), 538.

[22] Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. 2020. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663* (2020).

[23] Seth A Myers, Aneesh Sharma, Pankaj Gupta, and Jimmy Lin. 2014. Information network or social network? The structure of the Twitter follow graph. In *Proceedings of the 23rd international conference on world wide web*. 493–498.

[24] Mark EJ Newman, Duncan J Watts, and Steven H Strogatz. 2002. Random graph models of social networks. *Proceedings of the national academy of sciences* 99, suppl_1 (2002), 2566–2572.

[25] Amirreza Salamat, Xiao Luo, and Ali Jafari. 2021. HeteroGraphRec: A heterogeneous graph-based neural networks for social recommendations. *Knowledge-Based Systems* 217 (2021), 106817.

[26] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. 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*. Springer, 593–607.

[27] Md Shariar Sozol, Golam Mostafa Saki, and Md Mostafizur Rahman. 2024. Anomaly Detection in Cybersecurity with Graph-Based Approaches. *International Journal of Scientific Research in Engineering and Management (IJSREM)* 8, 8 (2024), 1–7.

[28] Lei Tang and Huan Liu. 2010. Graph mining applications to social network analysis. *Managing and mining graph data* (2010), 487–513.

[29] Nenad Trinajstic. 2018. *Chemical graph theory*. CRC press.

[30] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[31] Jianian Wang, Sheng Zhang, Yanghua Xiao, and Rui Song. 2021. A review on graph neural network methods in financial applications. *arXiv preprint arXiv:2111.15367* (2021).

[32] Shoujin Wang, Liang Hu, Yan Wang, Xiangnan He, Quan Z Sheng, Mehmet A Orgun, Longbing Cao, Francesco Ricci, and Philip S Yu. 2021. Graph learning based recommender systems: A review. *arXiv preprint arXiv:2105.06339* (2021).

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

[34] Jun Wu, Jingrui He, and Jiejun Xu. 2019. Net: Degree-specific graph neural networks for node and graph classification. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 406–415.

[35] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph neural networks in recommender systems: a survey. *Comput. Surveys* 55, 5 (2022), 1–37.

[36] Zhaohan Xi, Ren Pang, Shouling Ji, and Ting Wang. 2021. Graph backdoor. In *30th USENIX security symposium (USENIX Security 21)*. 1523–1540.

[37] Shunxin Xiao, Shiping Wang, Yuanfei Dai, and Wenzhong Guo. 2022. Graph neural networks in node classification: survey and evaluation. *Machine Vision and Applications* 33, 1 (2022), 4.

[38] Bo Yan, Cheng Yang, Chuan Shi, Yong Fang, Qi Li, Yanfang Ye, and Junping Du. 2023. Graph mining for cybersecurity: A survey. *ACM Transactions on Knowledge Discovery from Data* 18, 2 (2023), 1–52.

[39] Shuiqiao Yang, Bao Gia Doan, Paul Montague, Olivier De Vel, Tamas Abraham, Seyit Camtepe, Damith C Ranasinghe, and Salil S Kanhere. 2022. Transferable graph backdoor attack. In *Proceedings of the 25th international symposium on research in attacks, intrusions and defenses*. 321–332.

[40] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*. PMLR, 40–48.

[41] Zihao Yao, Fanding Huang, Yannan Li, Wei Duan, Peng Qian, Nan Yang, and Willy Susilo. 2025. Mecon: A GNN-based graph classification framework for MEV activity detection. *Expert Systems with Applications* 269 (2025), 126486.

[42] Dingqiang Yuan, Xiaohua Xu, Lei Yu, Tongchang Han, Rongchang Li, and Meng Han. 2024. E-SAGE: Explainability-Based Defense Against Backdoor Attacks on Graph Neural Networks. In *International Conference on Wireless Artificial*

Intelligent Computing Systems and Applications. Springer, 402–414.
[43] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. 2019. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931* (2019).
[44] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. 2019. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 793–803.
[45] Hangfan Zhang, Jinghui Chen, Lu Lin, Jinyuan Jia, and Dinghao Wu. 2023. Graph contrastive backdoor attacks. In *International Conference on Machine Learning*. PMLR, 40888–40910.
[46] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. *Advances in neural information processing systems* 31 (2018).
[47] Mengmei Zhang, Xiao Wang, Meiqi Zhu, Chuan Shi, Zhiqiang Zhang, and Jun Zhou. 2022. Robust heterogeneous graph neural networks against adversarial attacks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 36. 4363–4370.
[48] Xiang Zhang and Marinka Zitnik. 2020. Gnnguard: Defending graph neural networks against adversarial attacks. *Advances in neural information processing systems* 33 (2020), 9263–9275.
[49] Xiao-Meng Zhang, Li Liang, Lin Liu, and Ming-Jing Tang. 2021. Graph neural networks and their current applications in bioinformatics. *Frontiers in genetics* 12 (2021), 690049.
[50] Zaixi Zhang, Jinyuan Jia, Binghui Wang, and Neil Zhenqiang Gong. 2021. Backdoor attacks to graph neural networks. In *Proceedings of the 26th ACM Symposium on Access Control Models and Technologies*. 15–26.
[51] Zhiwei Zhang, Minhua Lin, Enyan Dai, and Suhang Wang. 2024. Rethinking graph backdoor attacks: A distribution-preserving perspective. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4386–4397.
[52] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. 2021. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM international conference on web search and data mining*. 833–841.
[53] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. 2024. Disambiguated node classification with graph neural networks. In *Proceedings of the ACM Web Conference 2024*. 914–923.
[54] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2019. Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 1399–1407.

A Details of Datasets

A.1 Heterogeneous Graph Datasets

In our work, we evaluate the performance of HGBA and baseline attacks on heterogeneous graphs based on ACM, DBLP, and IMDB Datasets. These datasets are the most commonly used real-world heterogeneous graph datasets in the HGNN research field for semi-supervised node classification (SSNC) tasks. Detailed statistics for these datasets are summarized in Table 7.

(i) ACM [18] – a citation network extracted from the ACM digital library[1], refined in subsequent work [18] to incorporate all paper citation and reference edges for enhanced relational complexity;

(ii) DBLP [9] – a bibliographic network of computer science literature sourced from DBLP[2],

(iii) IMDB [9] – a movie network from the IMDB database, capturing movies, directors, and actors across multiple genres[3].

A.2 Homogeneous Graph Datasets

We utilize five homogeneous graph datasets, including Cora, PubMed, and CiteSeer [40] for node classification, PROTEINS [22] for graph classification, to evaluate HGBA's attack-extensibility, and Flickr [43] as used in IO1 to explore the impact of the attack budget. Detailed statistics for these datasets are summarized in Table 8 & Table 9.

(i) Cora, PubMed, and CiterSeer [40] - These datasets are citation networks in which nodes represent papers and edges indicate citation links. For Cora and CiteSeer, nodes are characterized by binary word vectors that reflect the presence or absence of words from a fixed dictionary. PubMed, however, uses TF/IDF-weighted word vectors to describe each node. Across all three datasets, nodes are classified according to their research domains.

(ii) Flickr [43] - In this network, each node represents a single image uploaded to Flickr. Edges connect pairs of images that share specific attributes, such as location, gallery, or comments. Node features are captured by a 500-dimensional bag-of-words model from NUS-wide. For labels, the images are grouped manually into 7 unique categories.

(iii) PROTEINS [22] - A collection of proteins categorized as either enzymes or non-enzymes. Nodes denote amino acids, with edges linking pairs that are within 6 Angstroms of each other.

B Experimental Details

B.1 Dataset Splitting

We conduct experiments on transductive semi-supervised node classification tasks over heterogeneous graphs, where node labels are predicted using the entire graph structure, the features of all nodes, and the labels observed only for training nodes. Following the common practice in most studies, for each dataset, we randomly partition nodes into 20% for training, 10% for validation, and 70% for testing, as detailed in Table 7.

B.2 Dataset Poisoning

For the fixed-subgraph-based SBA-SAMPLE and SBA-GEN, we generate a homogeneous trigger structure of size 3 using the Erdős-Rényi (ER) model, which is proved to can produce more effective trigger structures in prior work. For SBA-SAMPLE, node features are randomly sampled from the training graph's node feature matrix, while for SBA-GEN, they are drawn from a Gaussian distribution matching the mean and variance of the training graph's node feature matrix.

For the adaptive subgraph-based GTA, UGBA, and DPGBA, we implement these methods on subgraphs extracted via the most influential meta-path to obtain corresponding homogeneous triggers.

For all resulting homogeneous triggers, we introduce intermediate nodes between trigger nodes along the most influential meta-path, transforming them into heterogeneous triggers connected to poisoned nodes, as illustrated in Figure 3. Notably, in each dataset, we designate the class with the fewest instances as the adversary-specified target classy_t to mitigate the impact of imbalanced data distributions (class 0 for ACM and DBLP, class 1 for IMDB).

B.3 Training and Evaluation

In simulating the role of standard researchers and developers operating in real-world scenarios, we train a clean model on the clean heterogeneous graph dataset by optimizing the cross-entropy loss. We then select the model with the lowest validation loss via early

[1] https://dl.acm.org/
[2] https://dblp.org/
[3] https://www.imdb.com/

Table 7: Statistics of Heterogeneous Graph Datasets Used in Experiments

Dataset	Node Type	# Nodes	Feat. Dim.	Edge Types (# Edges)	Target	#Classes	Train	Val	Test	Metapaths
ACM	Paper (P)	3,025	1,902	P-P (5,343)	Paper	3	605	303	2,117	PAP
	Author (A)	5,959	1,902	P-S (3,025)						PSP
	Subject (S)	56	1,902	P-T (255,619)						
	Term (T)	1,902	-	P-A (9,949)						PTP
DBLP	Author (A)	4,057	334	A-P (19,645)	Author	4	811	406	2,840	APA
	Paper (P)	14,328	4,231	P-T (85,810)						APTPA
	Term (T)	7,723	50							
	Conference (C)	20	-	P-C (14,328)						APCPA
IMDB	Movie (M)	4,278	3,066	M-D (4,278)	Movie	5	856	428	2,994	MDM
	Director (D)	2,081	3,066	M-A (12,828)						MAM
	Actor (A)	5,257	3,066							

Table 8: Statistics of Homogeneous Graph Datasets Used in Experiments on the Node Classification Task.

Datasets	#Nodes	#Edges	#Feature	#Classes
Cora	2,708	5,429	1,443	7
Pubmed	19,717	44,338	500	3
CiteSeer	3,327	4,732	3,703	6
Flickr	89,250	899,756	500	7

Table 9: Statistics of Homogeneous Graph Datasets Used in Experiments on the Graph Classification Task.

Datasets	#Graphs	#Nodes	#Avg Nodes	#Avg Edges	#Classes
PROTEINS	1,113	43,471	39.06	72.82	2

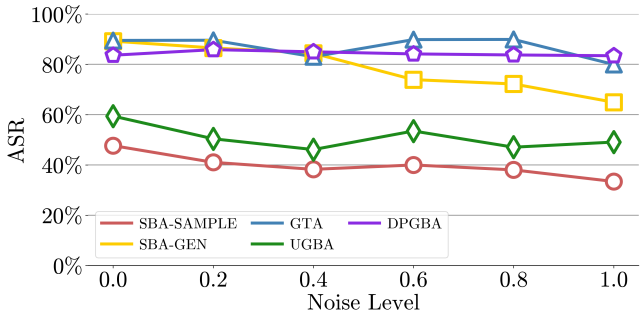

Figure 11: Impact of Node Perturbations on Backdoor Activation for Existing Graph Backdoor Attacks on HGNNs. Average results across six HGNNs on the ACM dataset.

stopping and calculate its Micro-F1 and Macro-F1 scores on the test nodes V_{test}.

For clean models, we train the clean model on the clean heterogeneous graph dataset by optimizing the cross-entropy loss. Then, we select the model with the lowest validation loss via early stopping and calculate its Micro-F1 and Macro-F1 scores on the test nodes V_{test}.

For backdoored models under black-box settings, to closely simulate a realistic black-box attack, we shift roles from attacker to regular user after poisoning the dataset. We follow the same steps as training a clean model to obtain the backdoored model. The only difference is that we additionally compute the ASR by poisoning the test set post-evaluation using the improved strategy from the takeaway of Section 3.2 IO3. It is worth noting that, for HGBA, since we have two different backdoor activation strategies, we will obtain two sets of ASR values.

B.4 Other Details

We repeat each experiment five times, reporting average metrics. Consistent settings apply across all attacks, with trigger size limited to 3 nodes; additional parameters are summarized in Table 10.

C Detailed Experimental Results of IO1

Table 11 shows the performance of homogeneous GNNs (GCN, GAT, GraphSAGE) in terms of accuracy on clean graphs across three datasets (Cora, Pubmed, Flickr).

Table 12 details the performance of HGNNs on clean graphs, reporting both micro-F1 and macro-F1 scores for six models (GCN, GAT, GraphSAGE, RGCN, HetGNN, HAN) across ACM, DBLP, and IMDB datasets.

Table 13 then focuses on the average performance of graph backdoor attacks on three homogeneous GNNs under different attack budgets (1%, 3%, 5%, 10%), presenting clean accuracy and attack success rate (ASR).

Table 14 for HGNNs further demonstrates the impact of these attacks on six HGNN models, providing clean micro-F1, clean macro-F1, and ASR metrics across the same range of attack budgets.

D Detailed Experimental Results for IO2

Fig. 11, Fig. 12, and Fig. 13 illustrate the average impact of node perturbations on backdoor activation for existing graph backdoor attacks on six HGNNs, with each figure corresponding to the ACM, DBLP, and IMDB datasets, respectively.

Table 10: Default Parameter Settings

Model Architecture				
Models	**Architecture**	**Aggregator**	**Dropout**	**#Attention Heads**
GCN	2 AL	Weighted Sum	0.0	-
GAT	2 AL	Attention - weighted Sum	0.6	8
GraphSAGE	2 AL	Mean	0.5	-
RGCN	2 AL	Relation - weighted Sum	0.0	-
HetGNN	2 AL	Heterogeneous Mean	0.0	-
HGT	2 AL	Heterogeneous Multi - head Attention	0.0	8
HAN	1 AL	Metapath - based Attention	0.6	8
Other Settings of Model Architecture				
Hidden Channels		128		
Classifier Architecture		FCN (1 FC + 1 SM)		
Model Training				
Optimizer		Adam		
Learning Rate		0.003		
Weight Decay		0.0001		
Epochs		200		
Batch Size		Full graph		
Patience (Early Stopping)		30		
Delta (Early Stopping)		0.001		

FC: fully - connected layer, SM: softmax layer.

Delta: Minimum improvement threshold for validation loss.

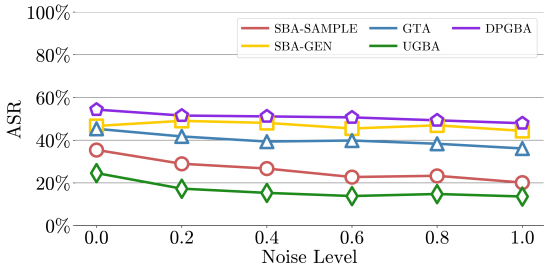

Figure 12: Impact of Node Perturbations on Backdoor Activation for Existing Graph Backdoor Attacks on HGNNs. Average results across six HGNNs on the DBLP dataset.

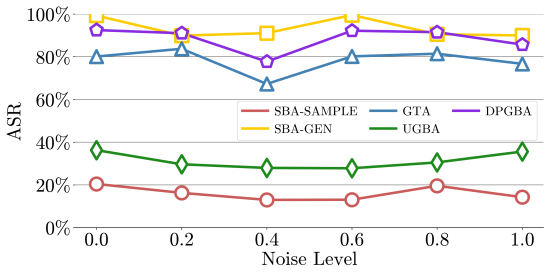

Figure 13: Impact of Node Perturbations on Backdoor Activation for Existing Graph Backdoor Attacks on HGNNs. Average results across six HGNNs on the IMDB dataset.

Table 11: Performance of GNNs on Clean Graph. (Accuracy(%))

Dataset	GCN	GAT	GraphSAGE
Cora	82.9	84.5	81.8
Pubmed	85.1	83.9	85.7
Flickr	45.5	46.5	47

Table 12: Performance of HGNNs on Clean Graph. (Clean Micro-F1 (%) | Clean Macro-F1 (%))

Models	ACM	DBLP	IMDB
GCN	87.57 | 87.69	92.09 | 91.48	50.32 | 49.78
GAT	88.90 | 89.01	88.02 | 86.09	49.53 | 48.79
GraphSAGE	90.79 | 90.85	92.48 | 91.87	58.35 | 58.29
RGCN	91.89 | 91.96	92.54 | 91.96	62.68 | 62.62
HetGNN	92.13 | 92.19	91.45 | 90.86	60.83 | 60.57
HAN	90.75 | 90.82	92.72 | 92.14	61.03 | 60.89

E Detailed Experimental Results for IO3

Fig. 14, Fig. 15, and Fig. 16 show the impact of trigger density on the ASR of graph backdoor attacks in heterogeneous graphs, corresponding to the ACM, DBLP, and IMDB datasets, respectively.

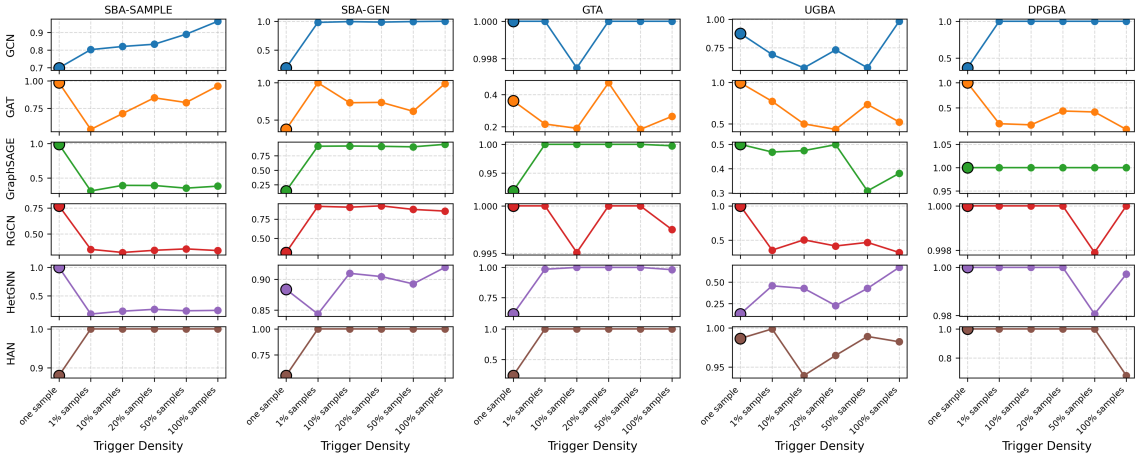

Figure 14: Impact of Trigger Density on Attack Success Rate (ASR) of Graph Backdoor Attacks in Heterogeneous Graphs (ACM Dataset).

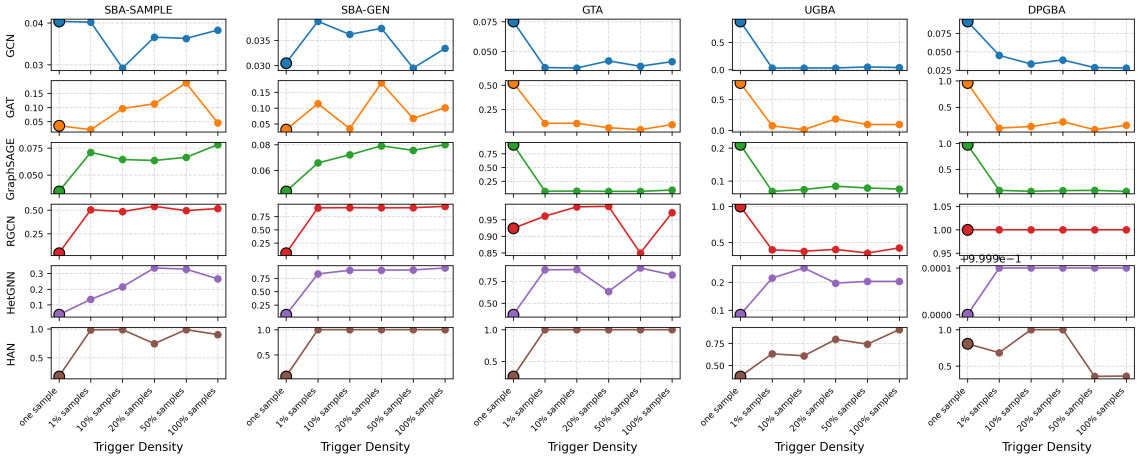

Figure 15: Impact of Trigger Density on Attack Success Rate (ASR) of Graph Backdoor Attacks in Heterogeneous Graphs (DBLP Dataset).

Table 13: Average Performance of Graph Backdoor Attacks on Three GNNs across Three Datasets under Different Attack Budgets. (Clean Acc (%) | ASR (%))

Attacks	1%	3%	5%	10%
SBA-SAMPLE	71.69 | 35.20	70.91 | 81.58	70.53 | 88.74	69.73 | 93.80
SBA-GEN	71.44 | 31.47	70.72 | 85.42	70.48 | 90.45	69.57 | 94.40
GTA	71.19 | 83.06	70.65 | 78.26	69.57 | 77.97	65.88 | 81.64
UGBA	70.55 | 85.03	70.37 | 94.29	70.16 | 96.16	69.85 | 95.29
DPGBA	70.27 | 95.63	70.25 | 97.11	69.92 | 97.49	69.15 | 98.54

F Algorithmic Details of HGBA and Backdoor Metapath Selection

Algorithm 1 outlines the overall HGBA process and Algorithm 2 focuses on backdoor metapath selection.

Algorithm 1 HGBA: Heterogeneous Graph Backdoor Attack

Require: Clean graph $G = (V, E, X, R, Y)$, target class y_t
Ensure: Poisoned graph $G_{poisoned}$, trigger node v_t, backdoor metapath P_b

$v_t \leftarrow$ SelectTriggerNode(G) {Based on Betweenness Centrality-based Trigger Node Selection Strategy}
$P_b \leftarrow$ SelectBackdoorMetapath(G) {Based on Algorithm 2}
$V_{poisoned} \leftarrow$ IdentifyPoisonedNodes(G, v_t, P_b)
for each $v_{p_i} \in V_p$ **do**
  $e_{p_i} \leftarrow$ AddEdges(v_{p_i}, v_t, P_b)
  $Y(v_{p_i}) \leftarrow y_t$
end for
$E_p \leftarrow \bigcup_{i=1}^{n} e_{p_i}$
$G_{poisoned} \leftarrow (V, E \cup E_p, X, R, Y)$
return $G_{poisoned}, v_t, P_b$

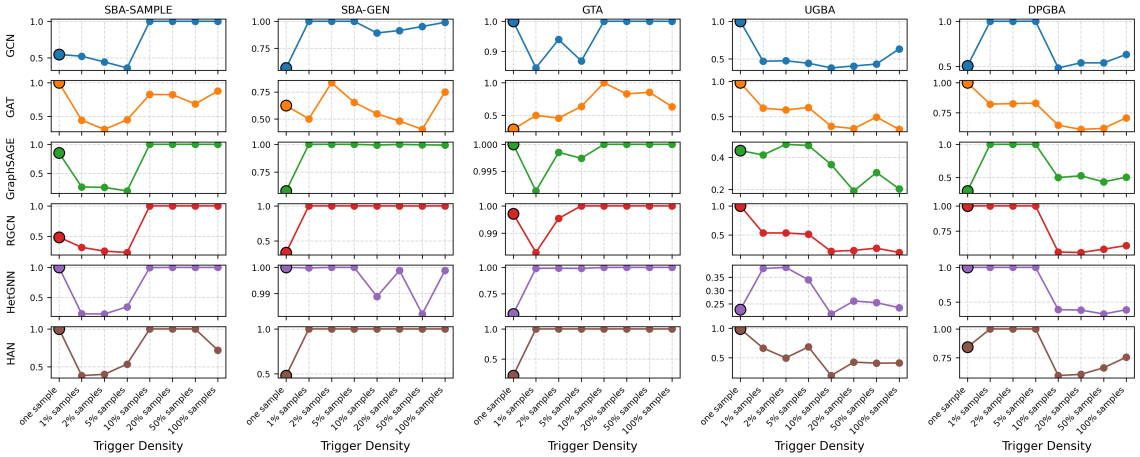

Figure 16: Impact of Trigger Density on Attack Success Rate (ASR) of Graph Backdoor Attacks in Heterogeneous Graphs (IMDB Dataset).

Table 14: Average Performance of Graph Backdoor Attacks on Six HGNNs across Three Datasets under Different Attack Budgets. (Clean Micro-F1 (%) | Clean Macro-F1 (%) | ASR(%))

Attacks	1%	3%	5%	10%
SBA-SAMPLE	79.88 | 79.59 | 9.74	78.16 | 77.65 | 19.53	79.45 | 79.14 | 28.28	78.44 | 78.16 | 34.29
SBA-GEN	79.91 | 79.63 | 63.42	80.72 | 80.51 | 72.30	79.48 | 79.11 | 76.95	77.69 | 76.83 | 81.15
GTA	80.42 | 80.19 | 41.60	80.16 | 79.92 | 59.30	79.75 | 79.48 | 65.14	78.60 | 77.79 | 71.74
UGBA	80.19 | 79.89 | 30.99	79.97 | 79.75 | 34.91	79.72 | 79.47 | 38.64	78.05 | 77.50 | 41.77
DPGBA	73.65 | 71.71 | 73.08	73.26 | 71.87 | 75.80	72.06 | 69.88 | 74.74	69.29 | 66.35 | 76.84

G DETAILED EXPERIMENTAL RESULTS

G.1 Comprehensive Experimental Results for RQ1

Table 15, Table 16, and Table 17 report the results of backdoor attacks under black-box settings on ACM, DBLP, and IMDB datasets respectively, presenting clean micro-F1, clean macro-F1, and attack success rate (ASR) metrics for various graph neural network models and attack methods, with the best ASR results highlighted in bold.

G.2 Detailed Experimental Results for RQ2

Fig. 17 shows the average performance of HGBA when attacking six HGNNs across three datasets under different attack budgets.

G.3 Detailed Experimental Results for RQ3

G.3.1 Trigger Node Selection Impact on HGBA$_{\text{I}}$. Fig. 18 illustrates the impact of trigger node selection for HGBA$_{\text{I}}$ across three datasets.

G.3.2 Results of Backdoor Metapath Selection. Table 18, Table 19, and Table 20 display the results of backdoor metapath selection using homogeneous GNNs as proxy models for ACM, DBLP, and IMDB datasets, respectively.

Fig. 19 shows the attention values of metapaths in HAN, highlighting the influence of selected metapaths (PAP, APCPA, MDM) on node classification.

Algorithm 2 Backdoor Metapath Selection

Require:
Clean graph G
Set of candidate metapaths $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$
Proxy model f
Ensure: Selected backdoor metapath P_b
if type(f) = HomoGNN **then**
  $acc^* \leftarrow 0$ {Best accuracy}
  $P_b \leftarrow \emptyset$ {Backdoor metapath}
  for all $P_i \in \mathcal{P}$ **do**
    $G_{P_i} \leftarrow \text{ExtractHomogeneousSubgraph}(G, P_i)$
    $f_{P_i} \leftarrow \text{TrainHomogeneousGNN}(G_{P_i})$
    $acc_{P_i} \leftarrow \text{EvaluateModel}(f_{P_i}, G_{P_i})$
    if $acc_{P_i} > acc^*$ **then**
      $acc^* \leftarrow acc_{P_i}$
      $P_b \leftarrow P_i$
    end if
  end for
else if type(f) = HGNN **then**
  $f_H \leftarrow \text{TrainHGNN}(G, \mathcal{P})$
  $W \leftarrow \text{ExtractMetapathWeight}(f_H)$
  $w^* \leftarrow 0$
  $P_b \leftarrow \emptyset$
  for all $P_i \in \mathcal{P}$ **do**
    if $W(P_i) > w^*$ **then**
      $w^* \leftarrow W(P_i)$
      $P_b \leftarrow P_i$
    end if
  end for
end if
return P_b

G.4 More Information for RQ4

G.4.1 Introduction of Defense Methods and Robust Models.

Table 15: Results of Backdoor Attacks under Black-Box Attack Settings on ACM Dataset. (Clean Micro-F1 (%) | Clean Macro-F1 (%) | ASR (%)). The best ASR results are marked in boldface. The red-highlighted data reflects the low stealthiness of DPGBA. Note that only one set of clean metrics is available as both $HGBA_1$ and $HGBA_2$ share the same test set for clean metrics evaluation, while separate ASR measurements are obtained for each activation method.

Models	SBA-SAMPLE	SBA-GEN	GTA	UGBA	DPGBA	$HGBA_1$	$HGBA_2$
GCN	89.09 | 89.17 | 13.50	88.29 | 88.37 | 78.64	87.77 | 87.91 | 41.25	87.02 | 87.01 | 56.84	89.57 | 89.57 | **100.00**	86.20 | 86.23 | 98.77	- | - | 99.82
GAT	87.30 | 87.36 | 14.88	88.43 | 88.45 | 32.94	88.95 | 89.06 | 12.11	88.15 | 88.28 | 24.67	88.15 | 88.25 | 36.51	88.52 | 88.56 | 88.05	- | - | **97.06**
GraphSAGE	90.18 | 90.32 | 4.41	91.22 | 91.31 | 57.11	91.45 | 91.47 | 64.19	92.16 | 92.16 | 53.98	90.79 | 90.82 | **100.00**	89.30 | 89.27 | 92.55	- | - | 95.10
RGCN	90.75 | 90.76 | 2.44	92.21 | 92.29 | 50.77	91.36 | 91.46 | 42.07	91.83 | 91.96 | 33.24	52.46 | 43.64 | **100.00**	91.03 | 90.47 | 96.94	- | - | 90.53
HetGNN	85.84 | 85.94 | 1.46	90.98 | 91.07 | 38.95	92.40 | 92.40 | 32.50	92.26 | 92.35 | 18.95	61.38 | 51.15 | **100.00**	87.93 | 87.36 | 71.90	- | - | 75.87
HAN	91.45 | 91.48 | 25.24	89.94 | 90.06 | 95.89	90.42 | 90.54 | 78.45	91.41 | 91.46 | 99.70	54.34 | 51.19 | 62.62	91.26 | 90.71 | **100.00**	- | - | **100.00**

Table 16: Results of Backdoor Attacks under Black-Box Attack Settings on DBLP Dataset. (Clean Micro-F1 (%) | Clean Macro-F1 (%) | ASR (%)). The best results are marked in boldface. Note that only one set of clean metrics is available as both $HGBA_1$ and $HGBA_2$ share the same test set for clean metrics evaluation, while separate ASR measurements are obtained for each activation method.

Models	SBA-SAMPLE	SBA-GEN	GTA	UGBA	DPGBA	$HGBA_1$	$HGBA_2$
GCN	92.22 | 91.63 | 2.36	91.41 | 90.92 | 2.62	91.02 | 90.30 | 2.87	91.48 | 90.84 | 2.60	91.73 | 91.13 | 3.23	90.58 | 90.10 | **100.00**	- | - | **100.00**
GAT	85.96 | 84.56 | 2.67	80.89 | 80.19 | 4.69	91.66 | 91.07 | 3.84	90.11 | 89.33 | 3.88	91.34 | 90.54 | 7.26	82.01 | 82.01 | **78.05**	- | - | 67.32
GraphSAGE	91.62 | 91.07 | 3.74	91.62 | 91.00 | 3.87	91.38 | 90.84 | 3.61	91.80 | 91.17 | 3.88	91.83 | 91.17 | 4.02	86.92 | 86.45 | 72.48	- | - | **73.83**
RGCN	91.94 | 91.44 | 13.28	92.22 | 91.72 | 72.37	91.97 | 91.23 | 54.31	92.33 | 91.81 | 12.10	92.57 | 91.96 | **100.00**	88.60 | 88.09 | 99.52	- | - | 99.63
HetGNN	91.27 | 90.54 | 7.24	91.34 | 90.80 | 67.03	90.50 | 89.81 | 39.60	89.51 | 88.65 | 9.64	91.09 | 90.43 | **100.00**	83.29 | 82.97 | 90.00	- | - | 91.80
HAN	92.50 | 91.93 | 8.55	92.89 | 92.27 | 97.75	92.33 | 91.71 | 78.20	91.24 | 90.39 | 8.82	80.36 | 72.65 | 75.74	91.67 | 91.13 | **100.00**	- | - | **100.00**

Table 17: Results of Backdoor Attacks under Black-Box Attack Settings on IMDB Dataset. (Clean Micro-F1 (%) | Clean Macro-F1 (%) | ASR (%)). The best results are marked in boldface. Note that only one set of clean metrics is available as both $HGBA_1$ and $HGBA_2$ share the same test set for clean metrics evaluation, while separate ASR measurements are obtained for each activation method.

Models	SBA-SAMPLE	SBA-GEN	GTA	UGBA	DPGBA	$HGBA_1$	$HGBA_2$
GCN	57.51 | 57.32 | 12.87	58.19 | 57.91 | **100.00**	56.63 | 56.41 | 51.56	57.11 | 56.87 | 30.13	59.06 | 58.65 | **100.00**	55.38 | 55.23 | 94.51	- | - | 97.78
GAT	57.41 | 57.26 | 14.43	57.03 | 56.83 | 43.35	57.28 | 57.00 | 59.07	56.37 | 56.20 | 64.29	58.41 | 58.13 | 49.35	56.16 | 56.01 | 80.07	- | - | **98.84**
GraphSAGE	59.49 | 59.51 | 13.12	59.17 | 59.22 | 99.88	59.83 | 59.83 | 51.97	59.70 | 59.76 | 32.54	59.75 | 59.74 | **100.00**	59.23 | 59.17 | 59.55	- | - | 75.97
RGCN	63.36 | 63.16 | 11.24	63.11 | 62.93 | 99.79	62.14 | 62.04 | 43.14	62.84 | 62.72 | 31.77	58.97 | 58.86 | **100.00**	62.12 | 61.87 | 40.04	- | - | 71.18
HetGNN	59.00 | 58.53 | 12.30	58.21 | 56.94 | 95.95	59.65 | 59.71 | 36.55	58.64 | 57.98 | 16.00	58.48 | 58.22 | **100.00**	58.42 | 58.23 | 45.88	- | - | 75.50
HAN	60.97 | 60.70 | 11.66	61.20 | 61.02 | **100.00**	60.88 | 60.70 | 53.50	59.38 | 59.15 | 54.79	55.34 | 54.63 | 76.71	60.27 | 60.13 | 83.11	- | - | 91.04

	GCN	GAT	GraphSAGE
PAP	**87.57%** | **87.69%**	**88.90%** | **89.01%**	**90.79%** | **90.85%**
PSP	70.59% | 68.91%	70.79% | 69.17%	88.62% | 88.69%
PTP	35.06% | 17.31%	-	85.87% | 85.84%

Table 18: Backdoor Metapath Selection Using Homogeneous GNNs as Proxy Models (ACM). "-" means out of GPU memory.

	GCN	GAT	GraphSAGE
APA	79.10% | 77.52%	78.30% | 79.96%	78.36% | 79.26%
APCPA	**92.09%** | **91.48%**	**88.02%** | **86.09%**	**92.48%** | **91.87%**
APTPA	72.99% | 71.29%	-	76.57% | 75.77%

Table 19: Backdoor Metapath Selection Using Homogeneous GNNs as Proxy Models (DBLP). "-" means out of GPU memory.

- *Prune* [5]. It operates by eliminating the edges that connect nodes with a low cosine similarity. Since the edges added by backdoor attackers often link nodes that are not similar to each other, this pruning strategy can effectively disrupt the trigger structure and the attachment edges established by the attackers.

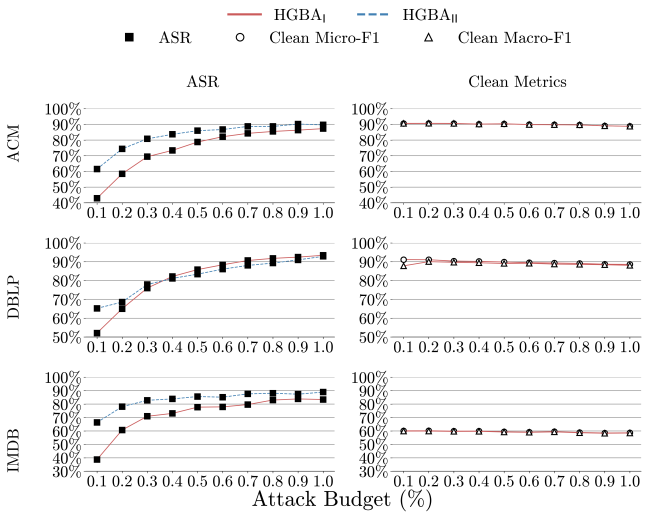

Figure 17: The Average Performance of HGBA Attacking Six HGNNs on Three Datasets under Varying Attack Budgets.

	GCN	GAT	GraphSAGE
MAM	50.32% | 48.79%	49.78% | 49.53%	58.35% | 58.29%
MDM	**58.99%** | **58.73%**	**59.00%** | **58.78%**	**60.16%** | **60.09%**

Table 20: Backdoor Metapath Selection Using Homogeneous GNNs as Proxy Models (IMDB)

Figure 18: Impact of Trigger Node Selection for HGBA$_{\mathrm{I}}$ on Three Datasets. The red dashed line indicates the highest ASR observed. 2245, 4, 242 respectively represent the trigger nodes selected for each dataset based on the strategy we proposed.

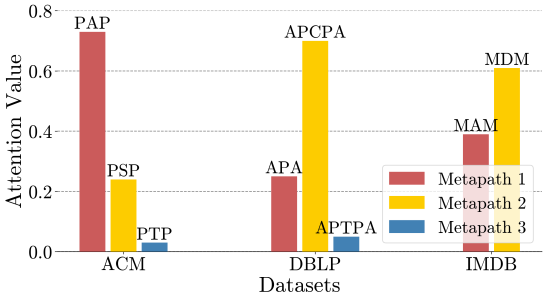

Figure 19: Attention values of metapaths in HAN, illustrating the influence of selected metapaths (PAP, APCPA, MDM) on node classification.

- *Prune+LD (Label Discarding)* [5]. Based on pruning the edges between dissimilar nodes as in the Prune method, it also discards the labels of the nodes that are connected by these dissimilar edges. This is aimed at reducing the impact of the dirty labels associated with the poisoned nodes, which can otherwise negatively affect the performance and security of the GNNs.
- *ESAGE* [42]. E-SAGE is a proposed approach for defending against GNN backdoor attacks, leveraging explainability. Based on the finding that malicious and benign edges exhibit significant differences in importance scores for explainability evaluation, E-SAGE adaptively conducts an iterative edge pruning process on the graph using edge scores.
- *HAN-RoHe* [47]. RoHe is a robust HGNN framework designed to defend against topology adversarial attacks. It equips an attention purifier that prunes malicious neighbors based on topology and features. Introducing metapath-based transiting probability as a prior criterion mitigates the perturbation enlargement effect, restraining the influence of adversarial hubs. The purifier then masks out neighbors with low confidence, alleviating the negative impact of unreliable neighbors in the soft attention mechanism.
- *RobustGCN* [54]. Instead of representing nodes as vectors, RobustGCN uses Gaussian distributions as hidden node representations in each convolutional layer, enabling it to absorb adversarial changes through variance adjustments. To address the propagation of attacks, it employs a variance-based attention mechanism, assigning weights to node neighborhoods according to their variances during convolutions. This approach enhances the robustness of GCNs against adversarial attacks.
- *GNNGuard* [48]. GNNGuard is a general algorithm for defending against training-time attacks on graph structure. It can be

Table 21: Performance of HGBA compared to baseline backdoor attacks on graph classification tasks using the PROTEINS dataset. Results report Accuracy on Clean Graph) and paired Clean Accuracy/Attack Success Rate (Clean Acc/ASR) for each method as percentages, averaged across GCN, GIN, and GraphSAGE.

Attacks	GCN	GIN	GraphSAGE
Clean Graph	71.95	70.14	69.95
SBA-SAMPLE	**71.92** | 33.33	**70.88** | 46.97	69.02 | 33.33
GTA	69.55 | 41.37	57.61 | 65.52	64.09 | 55.17
TRAP	68.38 | **77.78**	68.38 | 68.89	**69.32** | 55.56
HGBA	70.61 | 65.18	69.46 | **73.57**	68.86 | **62.50**

Table 22: Performance of HGBA on node classification tasks under homogeneous graphs. Results report Accuracy on Clean Graph, Clean Accuracy under backdoor attack (Clean Acc), and Attack Success Rate (ASR) as percentages, averaged across GCN, GAT, and GraphSAGE.

Dataset	Clean Graph	Clean Acc	ASR
Cora	87.99	85.64	85.58
PubMed	87.32	84.94	95.51
CiteSeer	74.94	70.18	95.79

integrated into any GNN by detecting and quantifying the relationship between graph structure and node features. It assigns higher weights to edges connecting similar nodes and prunes edges between unrelated nodes. With components like neighbor importance estimation and layer-wise graph memory, it restores GNN performance against various adversarial attacks, including targeted and non-targeted ones.

G.4.2 Defense Implementation Details of RQ4. For Prune, Prune+LD, and E-SAGE, which sanitize graphs through graph pruning techniques before training, since they were originally proposed for homogeneous graphs, to implement them on heterogeneous graph data, we first extract subgraphs on heterogeneous graphs based on backdoor metapaths. Then, we apply these three methods, record the two-end nodes of the edges that need to be pruned, and return to the heterogeneous graph to delete all the edges between these nodes based on the backdoor metapaths using masks. After that, normal training and evaluation are conducted.

For HAN-RoHe, RobustGCN, and GNNGuard, we simply replace the normal models with these robust models without any additional processing.

G.5 Details about RQ5

G.5.1 New Compared Method: TRAP.

- *TRAP.* It poisons the training dataset with perturbation-based triggers generated through a gradient-based score matrix from a surrogate GCN model.

G.5.2 Defense Implementation Details of RQ5. Here, we only describe how to extend the relation-based trigger mechanism of HGBA to homogeneous graph classification and node classification tasks on heterogeneous graph datasets. For the implementation details of other comparison methods, please refer to the related works.

For classification tasks on homogeneous graphs, we randomly select a certain percentage of samples from the training set as poisoned samples according to the poisoning rate. For each poisoned sample, we choose the three nodes with the lowest Betweenness Centrality and connect them to construct a trigger of size 3. The subsequent operations are consistent with those of other methods.

For node classification tasks on homogeneous graphs, we first select the node with the lowest Betweenness Centrality as the trigger node. Then, among the nodes that are not connected to the trigger node, we select some nodes according to the attack budget and establish edges to set up the trigger. The subsequent operations follow the same procedures as other methods.

G.5.3 Results of RQ5. Table 21 presents the performance comparison of HGBA against baseline backdoor attacks (SBA-SAMPLE, GTA, TRAP) in graph classification tasks using the PROTEINS dataset.

Table 22 illustrates the performance of HGBA in node classification tasks under homogeneous graphs.