A^{2}GBD: Attack-Agnostic Graph Backdoor Defense

Chenxu Du[†]
Institute of Computing Technology, Chinese Academy of Sciences
Southwest Jiaotong University
dcx_swjtu@outlook.com

Yang Liu[*†]
Institute of Computing Technology, Chinese Academy of Sciences
Beijing, China
liuyang2023@ict.ac.cn

Xingtong Yu
The Chinese University of Hong Kong
Hong Kong, China
starlien0905@gmail.com

Zhuoer Xu
Independent Researcher
Hangzhou, China
xuzhuoer.rex@gmail.com

Yang Liu
School of Vehicle and Mobility, Tsinghua University
Beijing, China
thu_ets_ly@tsinghua.edu.cn

Tianrui Li
School of Computing and Artificial Intelligence, Southwest Jiaotong University
trli@swjtu.edu.cn

Abstract

Graph Neural Networks (GNNs) are vulnerable to graph backdoor attacks, which poses severe risks to their deployment in safety-critical applications. Existing defenses predominantly focus on specific backdoor triggers, making them brittle and unable to generalize across different backdoor triggers with varying properties. Motivated by this limitation, this work proposes an attack-agnostic graph backdoor defense mechanism A^{2}GBD, which does not require prior knowledge of the specific attack strategies (e.g., edge perturbation, node attribute manipulation) to achieve effective defense. A^{2}GBD consists of suspicious node selection and defense strategy generation. The selection module selects high-suspicion nodes to enhance defense awareness, while the defense agent adaptively determines and executes defense strategies. Extensive experiments on multiple benchmark datasets demonstrate that A^{2}GBD consistently lowers attack success rates while maintaining high clean accuracy, showing strong robustness and generalizability against diverse graph backdoor attack strategies.

CCS Concepts

• **Security and privacy → Social network security and privacy**;
• **Mathematics of computing → Graph algorithms**.

Keywords

Attack-Agnostic, Graph Backdoor Defense, Graph Backdoor Attack

ACM Reference Format:
Chenxu Du, Yang Liu, Xingtong Yu, Zhuoer Xu, Yang Liu, and Tianrui Li. 2026. A^{2}GBD: Attack-Agnostic Graph Backdoor Defense. In *Proceedings of the ACM Web Conference 2026 (WWW '26), April 13–17, 2026, Dubai, United Arab Emirates.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3774904.3792507

[*]Corresponding Author.
[†]State Key Lab of AI Safety, Institute of Computing Technology, Chinese Academy of Sciences (CAS), Beijing, China. This work was conducted during Chenxu's internship at ICT, CAS.

WWW '26, Dubai, United Arab Emirates

ACM ISBN 979-8-4007-2307-0/2026/04
https://doi.org/10.1145/3774904.3792507

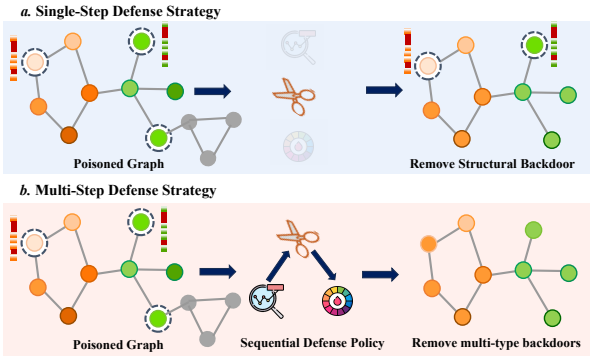


Figure 1: Illustration of the difference between traditional single-step backdoor defenses and our A^{2}GBD sequential multi-step framework. A^{2}GBD actively selects suspicious nodes and dynamically executes defense actions, enabling adaptive and context-aware protection.

1 Introduction

Graph Neural Networks (GNNs) have achieved remarkable success in a wide range of domains rich in relational data, such as social network analysis [23, 24], recommendation systems [7, 12, 22], and financial fraud detection [14, 16, 25, 35, 41]. By effectively leveraging the message-passing mechanism to capture both node features and topological structures, GNNs [10, 19, 29, 31, 40] have become the de facto standard for learning powerful representations on non-Euclidean data[2].

However, this impressive performance is often predicated on the assumption of training on clean, trustworthy data, which is frequently violated in real-world scenarios. The increasing reliance on large-scale datasets and pre-trained models has exposed GNNs to significant security threats [5, 9, 20, 21, 28]. Among these, backdoor attacks pose a particularly insidious risk [3, 8]. In such attacks, an adversary stealthily poisons the training data by embedding a specific trigger[32, 37, 38] (e.g., a unique subgraph or node feature pattern) into a small subset of graphs and associating them with a target label. A model trained on this poisoned dataset appears normal and maintains high accuracy on clean test samples, but will consistently misclassify any input containing the trigger to the adversary's chosen label.

Several efforts have been made to mitigate the vulnerability of GNNs to backdoor attacks. Attack methods have evolved from using simple random triggers (SBA[37]) to employing generative models for creating potent, sample-specific triggers (GTA[32]). On the defense side, techniques like Prune have been proposed to identify and remove trigger-based connections, prompting attackers to develop more stealthy methods like UGBA[4], DPGBA [38]. To defend such stealthy attack, RIGBD [39] verified that prediction variance under edge dropping is a crucial indicator for identifying poisoned nodes and proposes using random edge dropping to detect backdoors. These defense methods are effective in defending subgraph-based triggers but fail in attribute-based triggers [6]. Therefore, the effectiveness of current backdoor defense methods heavily rely on the characteristics of triggers.

Existing graph backdoor defenses suffer from two fundamental limitations. Firstly, many defense mechanisms rely on detecting specific attack patterns, which limits their generalizability and makes them susceptible to evasion by novel attack strategies. For example, RIGBD [39] identifies poisoned nodes by examining prediction variations under edge-dropping perturbations; while this method is effective against UGBA and DPGBA attacks, it fails to counter SPEAR. Secondly, most existing defenses adopt a single-step defense strategy, as depicted in Figure 1a, where models are typically trained using data poisoned by a single type of backdoor attack. This approach may remove certain triggers (e.g., subgraph-based triggers) but fails to recognize others (e.g., attribute-based triggers). As a result, such models are ill-equipped to handle diverse threats in real-world scenarios and often require retraining when confronted with different attacks, leaving systems vulnerable to complex and unpredictable security risks.

In contrast, a multi-step defense strategy (Figure 1b) could provide a more robust and adaptable framework by sequentially addressing multiple types of triggers and attack mechanisms, thereby enhancing resilience against a broader spectrum of backdoor threats. We propose an **A**ttack-**A**gnostic **G**raph **B**ackdoor **D**efense framework A^{2}GBD comprising two steps: suspicious node selection and defense strategy generation. The principle for evaluating suspicious nodes is based on two criteria: informativeness and robustness. The defense strategy generation is modeled as a Markov decision process, and the dynamic process formulates defense strategies based on a reward function to better adapt to the characteristics of different attacks [26].

Our contributions can be listed as follows. (1) We identify that current graph backdoor defense strategies rely on specific poisoned samples. (2) We propose an attack-agnostic graph backdoor defense mechanism A^{2}GBD, which does not require prior knowledge of the specific attack strategies. (3) Extensive experiments on multiple benchmark datasets demonstrate that A^{2}GBD can effectively defend current graph backdoor attacks.

2 Related Works

2.1 Graph Backdoor Attacks

Graph backdoor attacks represent a critical security threat to graph neural networks, wherein adversaries stealthily poison the training data by implanting malicious triggers that cause the model to misclassify triggered samples during inference while maintaining normal performance on clean data. Existing attack methodologies for node classification can be systematically categorized into three groups. Initial approaches [37] employed universal trigger generation, utilizing fixed subgraph structures across all target nodes. Subsequent methods like GTA [32] advanced to optimization-based trigger generation, crafting sample-specific triggers through trainable generators to enhance attack success rates. And the most recent developments focus on unnoticeable trigger generation, where techniques such as UGBA [4] introduce stealth-oriented loss functions to maximize trigger-node similarity, while DPGBA [38] formally investigates and enhances the stealth dimension, pushing attacks toward practical undetectability.

2.2 Graph Backdoor Defenses

Existing graph backdoor defenses target specific trigger properties: Jiang et al. [17] uses explainability to detect and filter samples dominated by small, anomalous subgraphs (the trigger), while ZIP [34] employs clustering to isolate graphs with common triggers for subsequent model fine-tuning. GraphProt [33] mitigates backdoor activation by leveraging topology-feature-filtering and sampling-based robust model inference. RIGBD [39] verifies that prediction variance under edge dropping is a crucial indicator for identifying poisoned nodes and proposes using random edge dropping to detect backdoors. LoSplit [18] is a training-time defense framework in graph that leverages this early-stage loss drift to accurately split target nodes.

Different from them, our proposed defense mechanism does not rely on any assumption based on the backdoor attack methods.

3 Preliminaries

3.1 Graph Neural Networks

A graph is denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where the node set $\mathcal{V} = \{v_1, \ldots, v_N\}$ comprises N nodes and the edge set $\mathcal{E} = \{e_1, \ldots, e_E\}$ comprises E edges. Each node has a D-dimensional feature vector, and all node features are denoted as $\mathbf{X} \in \mathbb{R}^{N \times D}$. The graph structure of $\mathcal{G}$ can be represented as an adjacency matrix $\mathbf{A} \in \{0,1\}^{N \times N}$, where $\mathbf{A}_{ij} = 1$ if $(i,j) \in \mathcal{E}$. Additionally, we denote all node labels as one-hot vectors $\mathbf{Y} \in \{0,1\}^{N \times C}$, where C is the number of classes.

The learning process of GNNs is typically formulated as a message-passing mechanism that iteratively aggregates information from a node's neighbors. For node v_i, the set of its neighbors is denoted as $\mathcal{N}(v_i) = \{v_j | \mathbf{A}_{ij} = 1\}$. In the l-th layer, the node embeddings $\mathbf{H}^{(l)} \in \mathbb{R}^{N \times d}$ are computed by updating the node embeddings from the previous layer $\mathbf{H}^{(l-1)}$ according to the formulation:

$$\mathbf{H}_i^{(l)} = \text{UPDATE}\left(\text{AGG}\left(\{\mathbf{H}_j^{(l-1)} | v_j \in \overline{\mathcal{N}}(v_i)\}\right)\right). \tag{1}$$

Here, $\overline{\mathcal{N}}(v_i) = \mathcal{N}(v_i) \cup \{v_i\}$ denotes node v_i's extended neighborhood. AGG$(\cdot)$ represents the aggregation function, which combines the information from neighboring nodes. The UPDATE$(\cdot)$ function transforms the aggregated information into new node embeddings.

3.2 Graph Backdoor Attack and Defense

Graph Backdoor Attacks (GBA) [4, 38] aim to poison the training set by implanting triggers in target nodes before classifier is trained. Backdoor attack on node classification can be defined as 3.1.

Definition 3.1 (Graph Backdoor Attack). **Given**: (1) a clean graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with node labels $\mathbf{Y}$; (2) a target label y_{tar}. The aim of graph backdoor attack is to select a target node set $\mathcal{V}_P \subset \mathcal{V}_{train}$ as the poisoned samples from unlabelled nodes in the training graph. For each node $v_i^P \in \mathcal{V}_P$, we transform it into a poisoned sample by implanting trigger as shown in Eq. (2):

$$x_i^P, \mathcal{N}^P(v_i) = \text{IMPLANT}(\mathcal{T}_\phi(v_i^P), x_i, \mathcal{N}(v_i)), \quad y_i^P = y_{\text{tar}}, \tag{2}$$

where x_i^P and $\mathcal{N}^P(v_i)$ are the poisoned node feature and neighbors of v_i^P after trigger implanting, $\mathcal{T}_\phi(\cdot)$ denotes the trigger generator that takes v_i as input.

Given a GNN classifier f trained on the poisoned training set, ideally, its behavior is manipulated so that:

$$f(x_j, \mathcal{N}(v_j)) = y_j, \quad f(x_j^P, \mathcal{N}^P(v_j)) = y_{\text{tar}}, \tag{3}$$

Definition 3.2 (Graph Backdoor Defense). **Given**: (1) A poisoned graph $\mathcal{G}_P = \{\mathcal{V}, \mathcal{E}_P, \mathbf{X}_P\}$; (2) A trained GNN f_θ. The purpose of graph backdoor defense is to select a set of suspicious nodes $\mathcal{V}_S \subset \mathcal{V}_{train}$ and recover the clean graph $\mathcal{G}_T$ from the poisoned graph $\mathcal{G}_P$ by conducting defense strategies in $\mathcal{V}_S$. The goal is to train a backdoor-free GNN model on $\mathcal{G}_T$ so that it can defend against backdoor triggers during inference on an unseen backdoored graph $\mathcal{G}_U$, while maintaining accuracy on clean data.

4 Methodology

We formulate the graph backdoor defense as a sequential decision-making problem, where a defense agent interacts with the graph and iteratively improves its strategy through sequential interventions. As shown in Figure 2, the proposed A²GBD consists of two modules. The Suspicious Node Selection component identifies high-suspicion nodes, guiding the defense agent to focus on nodes that are more likely to be poisoned. The Defense Strategy Generation component then determines defense actions in response to the graph context, enabling adaptive strategies that go beyond static heuristics or manually predefined rules.

4.1 Suspicious Node Selection

We introduce a suspicious node selection module that ranks nodes by backdoor risk using two complementary signals: *informativeness* (uncertainty/utility for learning) and *robustness* (likelihood of backdoor contamination). The joint ranking focuses defense actions on nodes that are both informative and high-risk.

4.1.1 Information Score. Existing studies [4, 36] have shown that nodes and subgraphs affected by backdoor attacks often carry anomalous patterns and high information content. Inspired by active learning, we introduce an information score to quantify the expected utility of querying nodes. This score unifies predictive uncertainty and structural abnormality into a single metric. To effectively quantify node information, we first require a base model that can reliably capture node semantics and structural patterns. Therefore, we pretrain a standard GCN f_θ on clean, unpoisoned graph $\mathcal{G}_{\text{clean}} = \{\mathcal{V}, \mathcal{E}\}$ with cross-entropy loss to learn global semantics and neighborhood propagation.

To identify nodes that are valuable for backdoor defense, we define an information score S_{info} by combining predictive uncertainty and structural abnormality. We estimate uncertainty via Monte Carlo Dropout and compute predictive entropy and BALD:

$$\mathcal{H}[y \mid x_i] = -\sum_c p_c \log p_c, \quad \text{BALD}_i = \mathcal{H}[y \mid x_i] - \mathbb{E}_\theta[\mathcal{H}[y \mid x_i, \theta]]. \tag{4}$$

Structural abnormality is measured by the Z-score deviation of node degree and local density from global patterns [1], denoted as $S_{\text{struct}}(i)$. We then normalize and combine both signals:

$$S_{\text{info}}(i) = w_1 \cdot \text{Norm}(\text{BALD}_i) + w_2 \cdot \text{Norm}(S_{\text{struct}}(i)), \tag{5}$$

where w_1 and w_2 control the trade-off. Higher $S_{\text{info}}(i)$ indicates more informative nodes for selection [13].

4.1.2 Robustness Score. While the information score summarizes uncertainty and structural anomalies, we further introduce the *Robustness Score* S_{rob} to assess backdoor-related suspiciousness by measuring embedding stability under small perturbations. Intuitively, clean nodes remain stable, whereas backdoored nodes exhibit noticeable representation shifts.

The construction of S_{rob} proceeds in a sequential manner. First, a GCN pre-trained on the clean graph provides a robust baseline embedding for each node. Applying this GCN to a potentially poisoned graph $\mathcal{G}_{\text{poison}} = \{\mathcal{V}, \mathcal{E}_{\text{poison}}, \mathbf{X}_{\text{poison}}\}$ yields the L-th layer embeddings: $\mathbf{h}_i = \mathbf{H}_i^{(L)}(\mathcal{G}_{\text{poison}})$, where the GCN parameters θ remain fixed. This step captures the true embedding shifts induced by potential backdoor triggers without any further training.

Next, we evaluate the sensitivity of these embeddings by applying small perturbations to the poisoned graph. Let T denote a perturbation operator (e.g., neighbor random sampling [11]), producing a perturbed graph $T(\mathcal{G}_{\text{poison}})$. Forward propagation through the same GCN produces perturbed embeddings:

$$\mathbf{H}_i^{(L)}(T(\mathcal{G}_{\text{poison}})) \equiv \mathbf{h}_i'. \tag{6}$$

For variance reduction, embeddings can be averaged over T independent perturbations $\{T_t\}_{t=1}^T$:

$$\mathbf{h}_i' \leftarrow \frac{1}{T}\sum_{t=1}^{T} \mathbf{H}_i^{(L)}(T_t(\mathcal{G}_{\text{poison}})). \tag{7}$$

Third, to amplify meaningful signals while maintaining robustness, we introduce a dual-branch encoder that maps GCN embeddings to a robust feature space:

$$\mathbf{u}_i = \lambda \cdot \text{MLP}_{\text{struct}}(\mathbf{h}_i) + (1-\lambda) \cdot \text{MLP}_{\text{feat}}(\mathbf{h}_i), \tag{8}$$

and similarly for perturbed embeddings:

$$\mathbf{u}_i' = \lambda \cdot \text{MLP}_{\text{struct}}(\mathbf{h}_i') + (1-\lambda) \cdot \text{MLP}_{\text{feat}}(\mathbf{h}_i'). \tag{9}$$

Here, $\text{MLP}_{\text{struct}}$ and MLP_{feat} project GCN outputs into a space where both structural and feature-based deviations are captured, and $\lambda \in [0, 1]$ balances their contributions. The dual-branch design ensures that nodes whose embeddings shift under perturbation are highlighted while embeddings of stable, clean nodes remain consistent.

To enforce stability, a consistency loss is imposed on the dual-branch encoder while keeping the GCN fixed:

$$\mathcal{L}_{\text{inv}} = \frac{1}{|\mathcal{V}_{\text{train}}|} \sum_{v_i \in \mathcal{V}_{\text{train}}} \|\mathbf{u}_i - \mathbf{u}_i'\|_2^2. \tag{10}$$

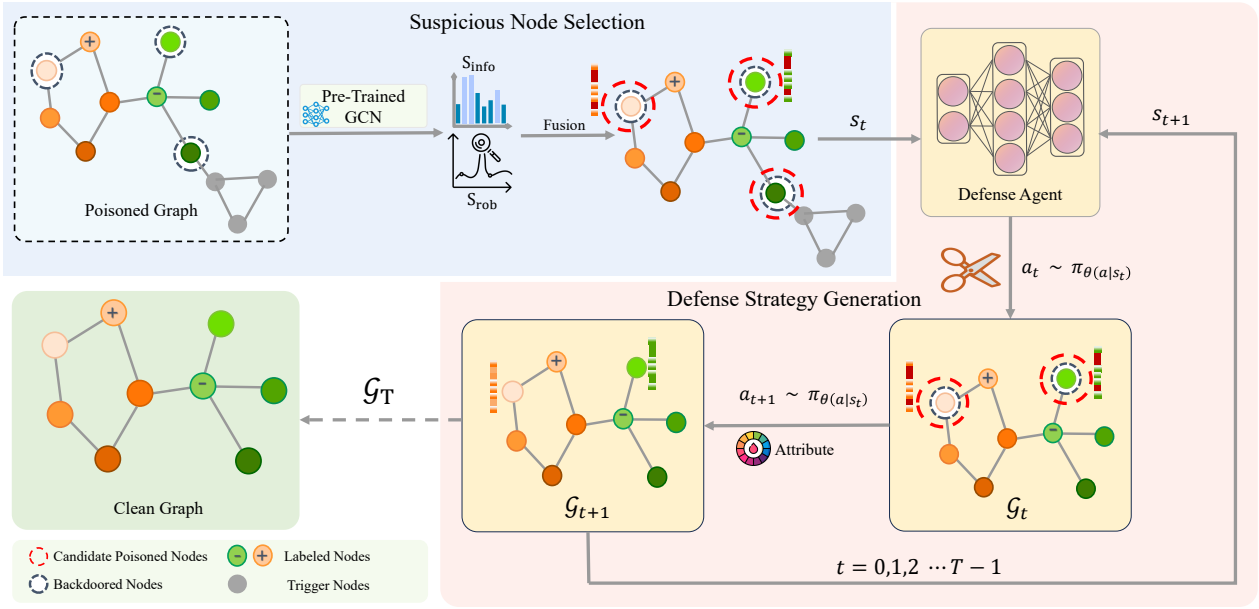


Figure 2: The A^{2}GBD framework formulates graph backdoor defense as a sequential decision-making process. The defense agent dynamically identifies suspicious nodes and generates adaptive defense actions, enabling effective and attack-agnostic mitigation of backdoor threats.

For multiple perturbations, $\mathbf{u}_i'$ can be replaced with an averaged embedding $\mathbf{u}_i' \leftarrow \frac{1}{T}\sum_{t=1}^{T}\mathbf{u}_i'(t)$ or the loss can be averaged across perturbations.

Finally, the *Robustness Score* for node v_i is defined as:

$$S_{\text{rob}}(v_i) = \frac{1}{1+\frac{1}{T}\sum_{t=1}^{T}\|\mathbf{u}_i - \mathbf{u}_i'(t)\|_2}. \tag{11}$$

Nodes exhibiting stable embeddings achieve $S_{\text{rob}}(v_i) \approx 1$, indicating high robustness, whereas nodes with significant embedding shifts receive lower scores, signaling potential backdoor contamination.

4.1.3 Fusion-Mode Selection. We design a *Fusion-Mode Selection* strategy to prioritize nodes for defense by integrating informativeness and robustness as shown in (12):

$$S(v_i) = w_{\text{info}}S_{\text{info}}(v_i) + w_{\text{rob}}S_{\text{rob}}(v_i), \tag{12}$$

where $S_{\text{info}}(v_i)$ measures each node's task contribution and $S_{\text{rob}}(v_i)$ reflects its embedding stability under potential backdoor perturbations. The weights w_{info} and w_{rob} are learnable. Nodes are ranked by their fused scores and the top-K nodes of the rank list form the candidate node set $\mathcal{U}$.

4.2 Defense Strategy Generation

In the Suspicious Node Selection module, we assign each node a fused priority that combines informativeness and robustness. These scores provide a static risk estimate but do not specify how to perform sequential interventions. We therefore formulate graph backdoor mitigation as a single-agent Markov Decision Process (MDP), and incorporate the fused priority into the state to guide decision-making over time.

To optimize long-term defense effectiveness beyond one-shot heuristics, we learn a sequential policy using Proximal Policy Optimization (PPO) with multiple value critics and constraint penalties, resulting in a Constrained PPO (CPPO) framework [26]. This design enables adaptive defense actions while enforcing safety constraints.

Importantly, the policy is trained directly on the poisoned training graph, without requiring clean graphs or oracle annotations. Such a sequential formulation is essential because one-shot or greedy defenses fail to capture cross-node and cross-action dependencies, especially under mixed and adaptive backdoor triggers.

4.2.1 State. To support efficient and targeted sequential defenses in large-scale graphs, we design a multi-level state representation that integrates node priors, local topology, global graph context, and historical defense information. At time step t, the state of a candidate node $u \in \mathcal{U}$ is defined as:

$$s_t(u) = \left[\, S_t(u) \parallel \mathcal{N}_t(u) \parallel g_t(u) \parallel H_t(u) \,\right], \tag{13}$$

where the fusion score prior $S_t(u)$ serves as a risk-aware prior that summarizes node suspiciousness and informativeness, thereby biasing the agent toward potentially compromised nodes. The neighborhood features $\mathcal{N}_t(u)$ encode the local graph structure around u, including degree-related statistics. The global context $g_t(u) \in \mathbb{R}^3$ provides episode-level signals, i.e. the remaining defense budget, the current timestep, and overall performance indicators. The defense history H_t summarizes previously executed actions and their diversity, enabling the agent to avoid redundant interventions and maintain coherent sequential behavior.

This multi-level state design converts the problem from unstructured exploration over the entire graph into a prior-guided sequential decision process. Compared with single-step heuristics or unconstrained exploration, it improves both learning stability and sample efficiency, leading to more reliable sequential defense policies.

4.2.2 Action. To realize interpretable and attack-agnostic sequential defense, we design an action space that allows the agent to progressively refine abnormal graph patterns without relying on any prior knowledge of triggers. Each action corresponds to a specific operation on the graph structure, node attributes, or model outputs, forming a coherent process of adaptive decision-making.

At a high level, structural actions regulate graph connectivity to suppress potential abnormal propagation, attribute actions adjust node features to mitigate hidden trigger patterns, and semantic actions calibrate model confidence to reduce overconfident predictions on suspicious nodes. We next formally define the action space and its effect on the graph state.

Action Space. The defense agent sequentially intervenes on suspicious nodes $\mathcal{V}_S$. At step t, an action $a_t \in \mathcal{A}$ updates the graph state $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \mathbf{X}_t)$:

$$\mathcal{G}_{t+1} = a_t(\mathcal{G}_t), \tag{14}$$

where $\mathcal{A}$ contains three complementary action types.

Structural Action. Structural actions modify graph connectivity to block abnormal propagation, e.g., edge pruning:

$$\mathcal{E}_{t+1} = \mathcal{E}_t \setminus \{(u,v) \mid s(u,v|\mathcal{G}_t) < \tau\},$$
$$s(u,v|\mathcal{G}_t) = \frac{\mathrm{sim}(\mathbf{h}_u^{(l)}, \mathbf{h}_v^{(l)})}{1 + |deg_{\mathcal{G}_t}(u) - deg_{\mathcal{G}_t}(v)|}. \tag{15}$$

Edges with low structural consistency are selectively removed to suppress suspicious propagation paths while preserving the overall topology.

Attribute Action. Attribute actions adjust node features to mitigate injected trigger patterns:

$$\mathbf{x}_i^{t+1} = \alpha \mathbf{x}_i^t + \frac{1-\alpha}{|\mathcal{N}(v_i)|} \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{x}_j^t. \tag{16}$$

This local aggregation smooths anomalous perturbations while maintaining neighborhood-level semantic consistency.

Semantic Action. Semantic actions control prediction confidence through temperature scaling:

$$p_T(y|v_i, \mathcal{G}_t) = \frac{f(y|\mathbf{x}_i^t, \mathcal{N}(v_i), \mathcal{G}_t)^{1/T}}{\sum_{y'} f(y'|\mathbf{x}_i^t, \mathcal{N}(v_i), \mathcal{G}_t)^{1/T}}, \quad T > 1. \tag{17}$$

This action reduces overconfident predictions on suspicious nodes, improving robustness against trigger-induced decision shortcuts.

Each action a_t incurs an intervention cost $c(a_t, v_i, \mathcal{G}_t)$, and the immediate reward is defined as:

$$r_t = \Delta\mathrm{Utility}(\mathcal{G}_t, \mathbf{X}_{\mathcal{V}_U}) - \lambda\, c(a_t, v_i, \mathcal{G}_t), \tag{18}$$

where $\mathcal{V}_U$ denotes the set of clean nodes and λ balances defense effectiveness and intervention cost. The role of this cost-aware reward is further discussed in Section 4.2.3.

4.2.3 Reward. To learn a sequential and adaptive defense policy, we design a reward that trades off attack suppression, benign performance preservation, and intervention stability. The reward is shaped to reduce backdoor effects while discouraging overly aggressive modifications that may degrade accuracy or introduce excessive structural/feature distortion.

The reward combines several complementary signals to provide dense guidance for multi-step credit assignment. The first term incentivizes reducing the attack success rate (ASR), encouraging actions that weaken trigger-induced prediction shifts. The second term preserves clean accuracy on benign nodes, preventing the policy from sacrificing utility for security. The third term penalizes intervention cost and side effects, acting as a constraint surrogate that stabilizes training and limits unnecessary structural pruning or feature smoothing. In addition, we introduce an information-theoretic regularizer, estimated via a Mutual Information Neural Estimator (MINE), to discourage strong feature–prediction coupling that can be exploited by triggers, thereby promoting attack-agnostic robustness.

Formally, the reward at timestep t is defined as:

$$R_t = \lambda_a \Delta ASR_t + \lambda_c \Delta Acc_t - \lambda_{con} C_t - \lambda_m I(X;Y), \tag{19}$$

where ΔASR_t and ΔAcc_t denote the changes in attack success rate and clean accuracy, respectively, C_t represents the constraint penalty, $I(X;Y)$ is the mutual information between node features and predictions, and $\lambda_a, \lambda_c, \lambda_{con}, \lambda_m$ are the weighting coefficients for each term. In practice, ΔASR_t and ΔAcc_t are estimated using a small held-out validation set and proxy confidence-based metrics, without assuming oracle knowledge of trigger patterns or poisoned labels. This estimation is used only during policy training and does not require access to clean/poisoned splits at inference time.

This formulation jointly promotes attack suppression, benign performance retention, and structure-constrained adaptation, while mitigating the statistical coupling between trigger features and model predictions. Through such a formulation, the defense process naturally evolves as a sequential decision procedure that aligns robustness with stability across multiple intervention steps.

4.3 Agent Modeling and Training Process

4.3.1 Agent Modeling. The defense agent operates on the graph by leveraging the pretrained GCN encoder f_θ to extract structural node representations during inference. At each step, the agent receives the current graph state $\mathbf{s}_t$ derived from GCN embeddings $\mathbf{Z}_t = f_\theta(\mathcal{G}_t)$, which are then pooled into a compact graph-level representation. To model decision-making, a lightweight two-layer MLP maps $\mathbf{s}_t$ into policy and value estimates:

$$\mathbf{h}^{(1)} = \mathrm{ReLU}(W_1 \mathbf{s}_t + b_1), \quad \mathbf{h}^{(2)} = \mathrm{ReLU}(W_2 \mathbf{h}^{(1)} + b_2), \tag{20}$$
$$\pi(a_t|\mathbf{s}_t) = \mathrm{Softmax}(W_\pi \mathbf{h}^{(2)} + b_\pi), \quad V(\mathbf{s}_t) = W_v \mathbf{h}^{(2)} + b_v. \tag{21}$$

This design enables the agent to utilize graph-structural features captured by GCN while maintaining the flexibility and efficiency of MLP-based policy learning, allowing sequential and adaptive defenses on dynamic graph environments.

4.3.2 Graph-State CPPO Update. The defense agent interacts with the graph environment sequentially to generate trajectory batches.

Table 1: Statistics of the experimental datasets.

Dataset	Nodes	Edges	Features	Classes
Cora	2,708	5,429	1,443	7
Citeseer	3,327	4,552	3,703	3
PubMed	19,717	44,338	500	3
OGB-arxiv	169,343	1,166,243	128	40

Each trajectory records graph states $\mathcal{G}_t$, actions a_t, immediate rewards r_t, costs c_t, and next graph states $\mathcal{G}_{t+1}$.

Generalized advantage estimation (GAE) computes reward and cost advantages using the current graph embeddings:

$$\hat{A}_t = r_t + \gamma \left(V_\phi(\mathbf{s}_{t+1}) - V_\phi(\mathbf{s}_t)\right) \tag{22}$$

$$\hat{C}_t = c_t + \gamma \left(V_\phi^c(\mathbf{s}_{t+1}) - V_\phi^c(\mathbf{s}_t)\right) \tag{23}$$

where $\mathbf{s}_t = \text{POOL}(f_\theta(\mathcal{G}_t))$ is the aggregated GCN embedding. Advantages are normalized:

$$\hat{A}_t \leftarrow \frac{\hat{A}_t - \mathbb{E}[\hat{A}_t]}{\text{std}(\hat{A}_t) + \epsilon}, \quad \hat{C}_t \leftarrow \frac{\hat{C}_t - \mathbb{E}[\hat{C}_t]}{\text{std}(\hat{C}_t) + \epsilon} \tag{24}$$

Minibatch updates compute policy, value, safety value, and entropy losses:

$$L^{\text{policy}} = -\mathbb{E}_t\Big[\min\left(r_t\hat{A}_t, \text{clip}(r_t, 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\Big], \tag{25}$$

$$L^{\text{constraint}} = \mathbb{E}_t\Big[\max\left(r_t\hat{C}_t, \text{clip}(r_t, 1-\epsilon, 1+\epsilon)\hat{C}_t\right)\Big], \tag{26}$$

$$L^{\text{value}} = \text{MSE}(V_\phi(\mathbf{s}_t), R_t), \quad L^c_{\text{value}} = \text{MSE}(V_\phi^c(\mathbf{s}_t), C_t), \tag{27}$$

$$L^{\text{entropy}} = -\mathbb{E}_t\big[H[\pi_\theta(\mathbf{s}_t)]\big]. \tag{28}$$

The total loss combines all components:

$$L = L^{\text{policy}} + \lambda_v(L^{\text{value}} + L^c_{\text{value}}) + \lambda_e L^{\text{entropy}} + \lambda_\mu L^{\text{constraint}} \tag{29}$$

The dual variable is updated independently to enforce constraints:

$$\lambda_\mu \leftarrow \max(0, \lambda_\mu + \eta_\mu(\bar{C} - C_{\text{budget}})) \tag{30}$$

This procedure tightly couples graph state representations with policy updates, ensuring that each action modifies the graph sequentially while maintaining reward optimization and constraint satisfaction. The overall algorithm workflow is illustrated in Algorithm 1 in the Appendix.

5 Experiments

In this section, we empirically analyze the effectiveness and efficiency of our proposed A^{2}GBD framework in various backdoor attack scenarios. We aim to answer the following research questions.

- **RQ1:** How does A^{2}GBD perform in terms of clean Accuracy (Acc) and attack success rate (ASR) compared with state-of-the-art defenses?
- **RQ2:** Can A^{2}GBD effectively identify highly suspicious nodes to improve defense robustness?
- **RQ3:** Can A^{2}GBD learn domain- generalizable structural patterns across different graph datasets?
- **RQ4:** Can A^{2}GBD achieve unified defense against multiple attacks through a single training and inference process while improving computational efficiency?

5.1 Experimental Settings

5.1.1 Datasets. We conduct experiments on four widely used benchmark datasets for node classification. Cora, Citeseer, and PubMed [27] are classic small-scale citation networks with strong homophily, serving as standard testbeds for evaluating the robustness of graph neural networks under backdoor attacks. OGB-arxiv [15] is a large-scale citation network with heterogeneous structures and long-range dependencies, posing additional challenges for backdoor defense. Together, these datasets cover both small homophilous graphs and large heterogeneous graphs, enabling a comprehensive evaluation of our proposed defense method. The basic statistics of the datasets are summarized in Table 1.

5.1.2 Baseline. To validate the defense capability of A^{2}GBD, we evaluate it against four state-of-the-art backdoor attack methods on graph neural networks, namely GTA [32], UGBA [4], DPGBA [38], and SPEAR [6]. Additionally, we implement three existing backdoor defense strategies, including Prune, OD, and RIGBD [39], and three representative robust GNN models, namely RobustGCN[42], GNNGuard [36], and randomized smoothing (RS) [30].

5.1.3 Evaluation Protocol. Following representative graph backdoor attack protocols, we split each graph into two disjoint subgraphs $\mathcal{G}_T$ and $\mathcal{G}_U$ with an 8:2 ratio. We construct poisoned training graphs $\mathcal{G}_T$ by mixing three attack methods (UGBA, DPGBA and SPEAR) with relative proportions $1:1:2$. The number of triggers $|\mathcal{V}_B|$ is set to 40, 80, 160, and 565 for Cora, Citeseer, PubMed, and OGB-arxiv, respectively. In all experiments, each trigger is limited to at most three nodes.

Defenders train models on the poisoned training graph $\mathcal{G}_T$. All defense methods, including A^{2}GBD, are trained directly on the poisoned training graph $\mathcal{G}_T$ without Access to clean graphs or oracle labels. During evaluation, half of the nodes in $\mathcal{G}_U$ are selected and equipped with backdoor triggers to measure the attack success rate (ASR), while the remaining nodes in $\mathcal{G}_U$ remain clean and are used to measure clean Accuracy (Acc).

5.1.4 Implementation. For all experiments, A^{2}GBD employs a 2-layer GCN as the model architecture. The number of candidate sizes $K = 50$ for Cora, CiteSeer, PubMed, and $K = 125$ for OGB-arxiv. Each experiment is repeated 5 times and we report the averaged results and standard deviation. The implementation of A^{2}GBD is available at https://github.com/Dcx-swjtu/A2GBD.

5.2 Defense Performance Comparison

To answer RQ1, we compare baseline defenses on benchmark datasets and report ASR and Acc in Table 2. We have the following observations:

(i) Across all datasets and attacks, A^{2}GBD demonstrates stable performance, maintaining low ASR under complex triggers such as UGBA, DPGBA, and SPEAR. On some datasets, A^{2}GBD achieves ASR comparable to or lower than RIGBD while exceeding RIGBD in Acc, indicating a better balance between security and task performance.

Table 2: Defense performance comparison(ASR (%) | Acc (%)) on four benchmark datasets. All metrics are reported in percentages. The last column reports the average ASR across the four attacks to indicate the overall defense robustness.

Dataset	Defense	GTA		UGBA		DPGBA		SPEAR		Avg. ASR
		ASR	Acc	ASR	Acc	ASR	Acc	ASR	Acc	
Cora	None	$91.43_{\pm 2.06}$	$84.82_{\pm 0.95}$	$96.07_{\pm 1.80}$	$83.55_{\pm 1.02}$	$95.96_{\pm 1.07}$	$82.66_{\pm 0.88}$	$98.57_{\pm 0.69}$	$83.07_{\pm 0.90}$	95.51
	Prune	$17.63_{\pm 1.12}$	$83.06_{\pm 0.92}$	$98.89_{\pm 0.94}$	$82.66_{\pm 0.88}$	$91.82_{\pm 0.67}$	$85.28_{\pm 0.97}$	$97.78_{\pm 0.89}$	$82.66_{\pm 0.88}$	76.03
	OD	$0.04_{\pm 0.02}$	$83.47_{\pm 0.70}$	$0.03_{\pm 0.02}$	$83.65_{\pm 0.75}$	$94.33_{\pm 1.30}$	$83.58_{\pm 0.86}$	$96.31_{\pm 1.85}$	$84.07_{\pm 0.82}$	47.68
	RIGBD	$0.00_{\pm 0.00}$	$83.70_{\pm 0.68}$	$0.07_{\pm 0.03}$	$84.81_{\pm 0.72}$	$0.09_{\pm 0.04}$	$84.19_{\pm 0.70}$	$91.56_{\pm 1.10}$	$83.33_{\pm 0.75}$	22.93
	A²GBD	$0.29_{\pm 0.05}$	$83.38_{\pm 0.65}$	$0.17_{\pm 0.03}$	$84.29_{\pm 0.68}$	$0.17_{\pm 0.04}$	$82.36_{\pm 0.71}$	$0.25_{\pm 0.05}$	$84.63_{\pm 0.78}$	**0.22**
Citeseer	None	$100.00_{\pm 0.00}$	$73.70_{\pm 1.10}$	$100.00_{\pm 0.00}$	$74.70_{\pm 1.12}$	$100.00_{\pm 0.00}$	$74.09_{\pm 1.08}$	$98.81_{\pm 0.90}$	$76.12_{\pm 1.15}$	99.70
	Prune	$12.24_{\pm 0.98}$	$72.46_{\pm 1.05}$	$97.68_{\pm 1.50}$	$74.35_{\pm 1.08}$	$94.80_{\pm 2.79}$	$73.21_{\pm 1.04}$	$92.77_{\pm 2.60}$	$75.29_{\pm 1.01}$	74.87
	OD	$0.04_{\pm 0.00}$	$72.84_{\pm 0.95}$	$0.06_{\pm 0.01}$	$73.80_{\pm 0.98}$	$98.42_{\pm 0.38}$	$73.66_{\pm 1.05}$	$91.53_{\pm 1.55}$	$77.80_{\pm 1.18}$	47.51
	RIGBD	$0.34_{\pm 0.07}$	$74.10_{\pm 1.08}$	$0.00_{\pm 0.00}$	$73.80_{\pm 0.98}$	$0.33_{\pm 0.06}$	$73.79_{\pm 1.00}$	$89.12_{\pm 1.83}$	$72.85_{\pm 0.99}$	22.45
	A²GBD	$0.39_{\pm 0.06}$	$74.92_{\pm 1.12}$	$0.40_{\pm 0.07}$	$74.21_{\pm 1.06}$	$0.31_{\pm 0.05}$	$73.90_{\pm 1.03}$	$0.35_{\pm 0.06}$	$73.32_{\pm 1.01}$	**0.36**
PubMed	None	$93.09_{\pm 4.10}$	$85.18_{\pm 0.95}$	$96.42_{\pm 1.82}$	$84.64_{\pm 1.38}$	$98.63_{\pm 0.73}$	$85.22_{\pm 0.97}$	$95.03_{\pm 2.27}$	$85.08_{\pm 0.96}$	95.79
	Prune	$28.10_{\pm 2.01}$	$85.05_{\pm 0.97}$	$92.87_{\pm 2.93}$	$85.09_{\pm 0.96}$	$88.64_{\pm 3.05}$	$85.13_{\pm 0.97}$	$96.70_{\pm 2.09}$	$85.24_{\pm 0.98}$	76.58
	OD	$0.03_{\pm 0.01}$	$85.27_{\pm 0.92}$	$0.01_{\pm 0.01}$	$85.19_{\pm 0.93}$	$91.32_{\pm 2.65}$	$85.12_{\pm 0.96}$	$92.04_{\pm 2.72}$	$85.64_{\pm 0.99}$	45.85
	RIGBD	$0.01_{\pm 0.01}$	$84.32_{\pm 0.94}$	$0.01_{\pm 0.01}$	$85.13_{\pm 0.95}$	$0.03_{\pm 0.01}$	$84.56_{\pm 0.94}$	$95.33_{\pm 4.03}$	$84.78_{\pm 0.95}$	23.84
	A²GBD	$0.03_{\pm 0.01}$	$83.27_{\pm 0.90}$	$0.12_{\pm 0.02}$	$82.07_{\pm 0.87}$	$0.13_{\pm 0.02}$	$83.57_{\pm 0.91}$	$0.70_{\pm 0.10}$	$82.07_{\pm 0.87}$	**0.25**
OGB-Arxiv	None	$75.34_{\pm 3.20}$	$65.76_{\pm 0.92}$	$98.82_{\pm 0.25}$	$63.95_{\pm 0.94}$	$95.63_{\pm 4.60}$	$65.72_{\pm 0.93}$	$96.95_{\pm 0.85}$	$66.91_{\pm 0.96}$	91.69
	Prune	$0.01_{\pm 0.01}$	$63.97_{\pm 0.91}$	$93.07_{\pm 1.95}$	$62.58_{\pm 0.89}$	$90.47_{\pm 2.62}$	$65.53_{\pm 0.92}$	$98.66_{\pm 0.93}$	$64.83_{\pm 0.90}$	70.55
	OD	$0.01_{\pm 0.01}$	$65.23_{\pm 0.92}$	$0.01_{\pm 0.01}$	$65.35_{\pm 0.92}$	$93.30_{\pm 2.00}$	$65.47_{\pm 0.93}$	$88.92_{\pm 1.55}$	$66.17_{\pm 0.94}$	45.56
	RIGBD	$0.01_{\pm 0.01}$	$66.51_{\pm 0.95}$	$0.01_{\pm 0.01}$	$65.21_{\pm 0.92}$	$0.00_{\pm 0.00}$	$65.24_{\pm 0.92}$	$96.20_{\pm 1.45}$	$65.96_{\pm 0.93}$	24.06
	A²GBD	$0.07_{\pm 0.02}$	$63.80_{\pm 0.88}$	$0.07_{\pm 0.02}$	$65.42_{\pm 0.92}$	$0.12_{\pm 0.03}$	$67.00_{\pm 0.94}$	$0.17_{\pm 0.03}$	$65.08_{\pm 0.91}$	**0.11**

(ii) Unlike RIGBD's random edge-dropping, A²GBD leverages active learning and reinforcement learning to mitigate the stealthiness of mixed triggers. With the same trigger budget, when attack signals are split into UGBA, DPGBA, and SPEAR injections, A²GBD still maintains low ASR and, in certain scenarios, improves Acc, demonstrating robust generalization across multiple trigger sources.

5.3 Ablation Study

We conduct an ablation study on the Suspicious Node Selection module in A²GBD. Here, G_t denotes the task-aware subgraph induced by selected suspicious nodes; removing G_t disables targeted structural intervention and reduces defense effectiveness. First, we replace the AL-based selection with heuristic scoring (A²GBD/H) and uncertainty sampling (A²GBD/U), keeping $K = 50$. Results show these alternatives fail to adequately cover poisoned nodes under complex triggers (UGBA, DPGBA, SPEAR), increasing ASR and reducing Acc, thus validating the module's necessity, as shown in Table 3.

We further ablate the internal components—information score and robustness score. Removing either degrades performance, confirming their importance in candidate ranking and high-risk node coverage. Combined with decentralized RL, these scores form a closed-loop optimization, enabling synergistic improvement in ASR and Acc.

Overall, the study demonstrates that the Suspicious Node Selection module, through strategy-driven selection and closed-loop RL coordination, is core to A²GBD's attack-agnostic defense in multi-source, cross-trigger scenarios.

Table 3: Ablation study (ASR(%) | Acc(%)) on Cora dataset.

Defense	GTA	UGBA	DPGBA	SPEAR
A²GBD	0.29 | 83.41	0.16 | 84.26	0.17 | 82.39	0.24 | 84.59
w/o G_t	0.55 | 83.14	0.34 | 84.13	0.40 | 82.12	0.55 | 84.42
A²GBD/H	0.86 | 82.88	0.59 | 83.83	0.75 | 81.77	0.83 | 83.97
A²GBD/U	0.80 | 82.92	0.49 | 83.93	0.54 | 81.98	0.90 | 84.07
w/o S(v_i)	0.64 | 83.02	0.45 | 83.97	0.50 | 82.03	0.70 | 84.18
w/o S(info)	0.60 | 83.10	0.41 | 83.89	0.47 | 81.90	0.66 | 84.21
w/o S(rob)	0.58 | 83.06	0.39 | 83.91	0.44 | 81.85	0.63 | 84.15

5.4 Poisoned Node Selection Performance

To answer RQ2, we assess the effectiveness of A²GBD for selecting highly suspicious nodes. We report the recall and precision of our poisoned-node identification method on OGB-arxiv dataset in Table 4. Recall is defined as the percentage of poisoned nodes that appear among the identified candidate nodes relative to the total number of poisoned nodes. Precision is defined as the percentage of identified candidate nodes that are truly poisoned.Then,*Standard Acc* refers to the classification accuracy of the original GNN model trained and evaluated on the clean graph without any defense or

Table 4: Performance of poisoned-node detection.

Attack	Standard Acc	ASR	Acc	Recall	Precision
GTA	64.21	0.07	63.80	87.02	84.07
UGBA	64.21	0.07	65.42	91.33	86.00
DPGBA	64.21	0.12	67.01	88.06	90.26
SPEAR	64.21	0.17	65.08	85.25	86.73

Table 5: Cross-dataset performance (ASR(%) | Acc(%)) trained on OGB-Arxiv and tested on PubMed.

Defense	GTA	UGBA	DPGBA	SPEAR
None	88.23 | 57.12	90.12 | 56.89	87.34 | 57.01	89.01 | 56.78
Prune	70.45 | 57.88	72.38 | 56.72	68.87 | 57.12	71.22 | 56.90
OD	35.12 | 58.21	32.56 | 57.45	38.78 | 57.30	40.24 | 57.10
RIGBD	12.21 | 58.98	14.76 | 57.85	10.89 | 58.42	20.01 | 58.03
A^2GBD	**2.45** | 57.88	**3.97** | 56.01	**2.78** | 57.32	**4.05** | 57.09

attack,serving as a performance reference,while *Acc* denotes the classification accuracy measured after applying the defense strategy,evaluated on clean (non-triggered) nodes.We observe:

(i) A^2GBD consistently achieves precision above 90% and recall over 80% in detecting poisoned nodes, showing that active learning effectively targets high-risk nodes while reducing redundant interventions.

(ii) Across diverse attacks, A^2GBD maintains low ASR and high Acc/Clean Acc. Even with partial detection, the synergy of active learning and RL enables the model to forget potential triggers, ensuring stable defense.

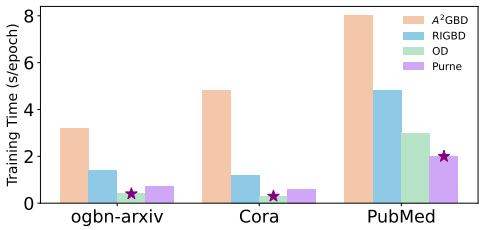
(iii) Unlike fixed-threshold baselines, A^2GBD adapts node selection via RL feedback, achieving stronger attack-agnostic generalization and robustness across varied trigger types.

5.5 Cross-Dataset Generalization

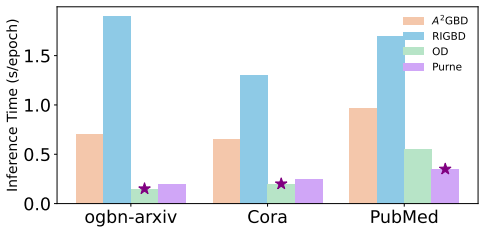
To answer RQ3, we conduct cross-dataset experiments from OGB-Arxiv to PubMed to evaluate whether the learned defense policy can generalize across unseen domains and diverse attack types (GTA, UGBA, DPGBA, and SPEAR), as shown in Table 5. Traditional defenses such as *None* and *Prune* suffer from high ASR (70–90%) and significant Acc degradation, implying strong dependence on dataset-specific patterns and static decision rules. OD and RIGBD alleviate part of the attack effect but still exhibit performance collapse in the unseen domain, suggesting that their defense behaviors are highly dependent on training-domain attack distributions and do not transfer well to unseen graph domains. In contrast, A^2GBD consistently achieves the lowest ASR (2–4%) with negligible Acc reduction (2–3%), demonstrating strong domain transferability.

5.6 Unified Training and Inference Efficiency

To answer RQ4, we observe that existing defenses (e.g., RIGBD and OD Prune) require separate training and inference for each attack, increasing computational costs linearly with the number of attacks.



(a) Training time on large-scale graphs



(b) Inference time on large-scale graphs

Figure 3: Comparison of training and inference time. The purple stars indicate the best (shortest) time for each dataset. A^2GBD shows slightly higher training time than RIGBD but maintains efficient inference, making it suitable for defending against multiple attacks simultaneously.

For example, defending against four attacks requires roughly four times the overhead of a single attack.

In contrast, A^2GBD requires only single training and single inference to defend against multiple attacks simultaneously. Although its training cost is slightly higher than RIGBD for one attack, its inference cost is already lower. As shown in Figure 3, with four attacks, the overall time cost of A^2GBD is about one-third of RIGBD.

6 Conclusion

In this work, we propose A^2GBD, an attack-agnostic framework for graph backdoor defense to mitigate the limitations of existing defense methods. By formulating the defense process as a Markov Decision Process (MDP), A^2GBD enables attack-agnostic and adaptive defense through an active node selector and a defense generator, moving beyond the single-step detection paradigm that fails against diverse trigger types. Furthermore, a structured state encoder enhances cross-domain generalization by integrating local and global graph representations. Extensive experiments demonstrate that A^2GBD achieves robust and transferable defense performance across diverse attacks and datasets.

Acknowledgments

The research work is supported by the National Natural Science Foundation of China under Grant No. 62406307, the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No. XDB0680201, the Postdoctoral Fellowship Program of CPSF under Grant Number GZB20240761.

References

[1] Leman Akoglu, Hanghang Tong, and Danai Koutra. 2014. Graph-based Anomaly Detection and Description: A Survey. arXiv:1404.4679 [cs.SI] https://arxiv.org/abs/1404.4679

[2] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine* 34, 4 (2017), 18–42.

[3] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526* (2017).

[4] 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.

[5] Enyan Dai, Tianxiang Zhao, Huaisheng Zhu, Junjie Xu, Zhimeng Guo, Hui Liu, Jiliang Tang, and Suhang Wang. 2022. A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability. *arXiv preprint arXiv:2204.08570* (2022).

[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 *Proceedings of the ACM on Web Conference 2025*. 1237–1247.

[7] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *The world wide web conference*. 417–426.

[8] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access* 7 (2019), 47230–47244.

[9] Zhiyu Guo, Yang Liu, Xiang Ao, and Qing He. 2025. Grasp: Differentially private graph reconstruction defense with structured perturbation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 767–777.

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

[11] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. 1025–1035.

[12] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 639–648.

[13] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Mark Lengyel. 2011. Bayesian Active Learning for Classification and Preference Learning. *arXiv preprint arXiv:1112.5745* (2011).

[14] Guanghui Hu, Yang Liu, Qing He, and Xiang Ao. 2024. F2GNN: An Adaptive Filter with Feature Segmentation for Graph-Based Fraud Detection. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6335–6339.

[15] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous Graph Transformer. n/a.

[16] Mengda Huang, Yang Liu, Xiang Ao, Kuan Li, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. 2022. Auc-oriented graph neural network for fraud detection. In *Proceedings of the ACM web conference 2022*. 1311–1321.

[17] Bingchen Jiang and Zhao Li. 2022. Defending against backdoor attack on graph nerual network by explainability. *arXiv preprint arXiv:2209.02902* (2022).

[18] Di Jin, Yuxiang Zhang, Bingdao Feng, Xiaobao Wang, Dongxiao He, and Zhen Wang. [n. d.]. LoSplit: Loss-Guided Dynamic Split for Training-Time Defense Against Graph Backdoor Attacks. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[19] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.

[20] Kuan Li, YiWen Chen, Yang Liu, Jin Wang, Qing He, Minhao Cheng, and Xiang Ao. 2024. Boosting the adversarial robustness of graph neural networks: An ood perspective. In *The Twelfth International Conference on Learning Representations*.

[21] Kuan Li, Yang Liu, Xiang Ao, and Qing He. 2023. Revisiting graph adversarial attack and defense from a data distribution perspective. In *The Eleventh International Conference on Learning Representations*.

[22] Yang Liu, Xiang Ao, Linfeng Dong, Chao Zhang, Jin Wang, and Qing He. 2020. Spatiotemporal activity modeling via hierarchical cross-modal embedding. *IEEE Transactions on Knowledge and Data Engineering* 34, 1 (2020), 462–474.

[23] Yang Liu, Xiang Ao, Fuli Feng, and Qing He. 2022. UD-GNN: Uncertainty-aware Debiased Training on Semi-Homophilous Graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1131–1140.

[24] Yang Liu, Xiang Ao, Fuli Feng, Yunshan Ma, Kuan Li, Tat-Seng Chua, and Qing He. 2023. FLOOD: A flexible invariant learning framework for out-of-distribution generalization on graphs. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*. 1548–1558.

[25] Yang Liu, Xiang Ao, Zidi Qin, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. 2021. Pick and choose: a GNN-based imbalanced learning approach for fraud detection. In *Proceedings of the web conference 2021*. 3168–3177.

[26] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[27] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93.

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

[29] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations (ICLR)*.

[30] Minjie Wang, Lingfan Yu, Da Zheng, Quan Gan, Yu Gai, Zihao Ye, Mufei Li, Jinjing Zhou, Qi Huang, Chao Ma, Ziyue Huang, Qipeng Guo, Hao Zhang, Haibin Lin, Junbo Zhao, Jinyang Li, Alexander J Smola, and Zheng Zhang. 2019. Deep Graph Library: Towards Efficient and Scalable Deep Learning on Graphs. *ICLR Workshop on Representation Learning on Graphs and Manifolds* (2019). https://arxiv.org/abs/1909.01315

[31] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.

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

[33] Xiao Yang, Yuni Lai, Kai Zhou, Gaolei Li, Jianhua Li, and Hang Zhang. 2025. GraphProt: Certified Black-Box Shielding Against Backdoored Graph Models. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, James Kwok (Ed.). International Joint Conferences on Artificial Intelligence Organization, 619–627. doi:10.24963/ijcai.2025/70 Main Track.

[34] Xiao Yang, Gaolei Li, Xiaoyi Tao, Chaofeng Zhang, and Jianhua Li. 2023. Black-box graph backdoor defense. In *International Conference on Algorithms and Architectures for Parallel Processing*. Springer, 163–180.

[35] Qi Yuan, Yang Liu, Yateng Tang, Xinhuan Chen, Xuehao Zheng, Qing He, and Xiang Ao. 2025. Dynamic Graph Learning with Static Relations for Credit Risk Assessment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 13133–13141.

[36] Xiang Zhang and Marinka Zitnik. 2020. Gnnguard: Defending graph neural networks against adversarial attacks. *Advances in neural information processing systems* 33 (2020), 9263–9275.

[37] 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.

[38] 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.

[39] Zhiwei Zhang, Minhua Lin, Junjie Xu, Zongyu Wu, Enyan Dai, and Suhang Wang. 2025. Robustness-Inspired Defense Against Backdoor Attacks on Graph Neural Networks. *ICLR* (2025).

[40] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI open* 1 (2020), 57–81.

[41] Xiaoqian Zhu, Xiang Ao, Zidi Qin, Yanpeng Chang, Yang Liu, Qing He, and Jianping Li. 2021. Intelligent financial fraud detection practices in post-pandemic era. *The Innovation* 2, 4 (2021).

[42] Zhaocheng Zhu, Shizhen Xu, Jian Tang, and Meng Qu. 2019. Graphvite: A high-performance cpu-gpu hybrid system for node embedding. 2494–2504.

A Training Algorithm

Directly applying standard PPO to sequential graph defense is ineffective due to two fundamental challenges. First, the graph defense process involves *irreversible structural and attribute modifications*, which violates the stationarity assumption of conventional PPO and often leads to unstable policy updates. Second, unconstrained reward optimization in PPO tends to favor aggressive actions that overly distort graph topology, resulting in severe utility degradation To address these issues, we propose *Graph-State Constrained PPO (G-CPPO)*, which introduces three key modifications tailored for graph defense. (i) We define a compact graph-level state by pooling node representations produced by a pretrained GCN, enabling the policy to reason over evolving graph structures while maintaining tractable state dimensionality. (ii) We explicitly model defense operations as graph transformation actions, allowing the policy to sequentially refine the graph based on its current structural state. (iii) We incorporate a cost-aware constraint via a dual variable λ, which regulates the trade-off between attack mitigation and graph utility preservation. Unlike standard PPO, which optimizes a single reward signal, G-CPPO jointly optimizes utility improvement and cost violation through a constrained objective. The additional constraint loss prevents the policy from collapsing to overly aggressive defense strategies, leading to more stable training and better generalization across different backdoor patterns. As a result, G-CPPO achieves more effective and controllable defense behavior than vanilla PPO in sequential graph defense settings.

Algorithm 1: Graph-State Constrained PPO (G-CPPO) for Sequential Graph Defense

Input: Poisoned graph $\mathcal{G}_0 = (\mathcal{V}_0, \mathcal{E}_0, \mathbf{X}_0)$, pretrained GCN f_θ, actor-critic parameters (θ, ϕ), dual variable λ, cost budget c_{budget}.

Output: Clean graph $\mathcal{G}_T$

for $t = 0$ **to** $T - 1$ **do**
- $\mathbf{Z}_t = f_\theta(\mathcal{G}_t)$, $\mathbf{s}_t = \text{POOL}(\mathbf{Z}_t)$;
- Sample $a_t \sim \pi_\theta(a|\mathbf{s}_t)$;
- $\mathcal{G}_{t+1} = a_t(\mathcal{G}_t)$;
- $r_t = \Delta\text{Utility}(\mathcal{G}_t, \mathcal{G}_{t+1})$, $c_t = c(a_t, \mathcal{G}_t)$;
- Store $(\mathbf{s}_t, a_t, r_t, c_t, \mathbf{s}_{t+1})$;
- Compute advantages (A_t, A_t^c) via GAE and normalize;
- **for** *epoch* $= 1$ **to** N_{train} **do**
  - **foreach** *minibatch* (s, a, A, A^c, R, R^c) **do**
    - $r = \exp(\log \pi_\theta(a|s) - \log \pi_{\theta_{\text{old}}}(a|s))$;
    - $L^{\text{policy}} = -\text{mean}(\min(rA, \text{clip}(r, 1-\epsilon, 1+\epsilon)A))$;
    - $L^{\text{constraint}} = \text{mean}(\max(rA^c, \text{clip}(r, 1-\epsilon, 1+\epsilon)A^c))$;
    - $L^{\text{value}} = \text{MSE}(V_\phi(s), R)$, $L^c_{\text{value}} = \text{MSE}(V^c_\phi(s), R^c)$;
    - $L = L^{\text{policy}} + \alpha_v(L^{\text{value}} + L^c_{\text{value}}) + \lambda L^{\text{constraint}}$;
    - Update (θ, ϕ) by gradient descent;
- $\lambda \leftarrow \max(0, \lambda + \eta(\bar{c} - c_{\text{budget}}))$;

Return $\mathcal{G}_T$.

B Cross-Domain Evaluation

To comprehensively evaluate the generalization capability of defense methods, we conducted systematic cross-dataset experiments on two citation network datasets: Citeseer and Cora. Table 6 presents the performance (Attack Success Rate ASR | Classification Accuracy ACC) of different defense methods against four types of attacks: GTA, UGBA, DPGBA, and SPEAR.

Experimental results demonstrate significant limitations of traditional defense methods across different datasets. Under no defense (None) scenario, the attack success rates on both datasets exceed 83%, confirming the effectiveness of backdoor attacks. The basic defense method Prune reduces ASR to the 60-70% range, but its fixed-threshold pruning strategy struggles to adapt to the graph structural characteristics of different datasets. The anomaly detection method OD performs slightly better on Cora than on Citeseer, possibly due to Cora's clearer graph structure facilitating anomaly detection, yet its ASR remains relatively high at 26-35%.

Notably, RIGBD demonstrates decent defense performance on both datasets, maintaining ASR below 12%. However, its effectiveness varies across different attack types, particularly showing reduced defense capability against UGBA and SPEAR attacks. This reflects the limitations of defense methods based on specific assumptions when facing diverse attack patterns.

In contrast, our proposed A^2GBD method maintains the best defense performance across both datasets and all attack types. On Citeseer, A^2GBD successfully suppresses ASR below 3.12% for all attacks; on Cora, it achieves even lower ASR below 2.01%. More importantly, while maintaining excellent defense effectiveness, A^2GBD has minimal impact on normal classification performance - the ACC loss on both datasets does not exceed 1%, significantly outperforming other comparative methods.

These results fully demonstrate A^2GBD's strong cross-dataset generalization capability. Its advantages stem from several aspects: First, the reinforcement learning-based sequential decision mechanism can adaptively learn graph structural features of different datasets without relying on predefined static rules; Second, the active learning-guided candidate node selection strategy effectively identifies cross-domain structural anomaly patterns; Finally, the attack-agnostic design philosophy enables unified defense against different types of backdoor attacks, avoiding overfitting to specific attack types.

In summary, cross-dataset experiments validate the practical value of A^2GBD in real-world scenarios. It not only effectively defends against backdoor attacks on in-distribution data but, more importantly, possesses strong out-of-domain generalization capability, providing reliable assurance for the secure deployment of graph neural networks in practical applications.

C Supplementary Baseline Evaluation

Table 7 reports additional baseline performance of three representative robust GNN models, namely RobustGCN [42], GNNGuard [36], and randomized smoothing (RS) [30], on the Citeseer and Cora datasets. It can be observed that the three existing defense methods exhibit consistently high attack success rates (ASR) on the OGB-Arxiv dataset, indicating limited effectiveness against backdoor attacks. This behavior is closely related to their defense mechanisms: RobustGCN mainly models feature uncertainty to improve

Table 6: Estimated attack performance (ASR(%) | ACC(%)) of different defenses on Citeseer and Cora datasets.

	Cora ⇒ Citeseer				Citeseer ⇒ Cora			
Defense	**GTA**	**UGBA**	**DPGBA**	**SPEAR**	**GTA**	**UGBA**	**DPGBA**	**SPEAR**
None	88.23 | 62.45	90.15 | 61.89	87.67 | 62.23	89.45 | 61.78	86.56 | 78.34	88.78 | 77.92	85.89 | 78.15	87.67 | 77.87
Prune	70.45 | 65.12	72.23 | 64.05	68.78 | 64.67	71.89 | 64.23	64.34 | 79.89	66.56 | 79.23	62.45 | 79.56	65.78 | 79.34
OD	34.67 | 67.45	32.12 | 66.78	37.89 | 66.91	35.45 | 66.56	30.45 | 80.12	28.78 | 79.67	33.23 | 79.89	31.67 | 79.45
RIGBD	11.23 | 68.78	13.45 | 67.95	10.67 | 68.34	12.23 | 68.12	8.89 | 80.34	10.45 | 79.78	7.23 | 80.15	9.56 | 79.89
A^{2}GBD	**3.89** | 68.25	**5.12** | 67.34	**4.34** | 68.01	**4.67** | 67.78	**2.95** | 80.01	**4.01** | 79.45	**3.23** | 79.78	**3.45** | 79.56

Table 7: Defense performance on OGB-Arxiv under different backdoor attacks.

Defense	DPGBA		GTA		UGBA		SPEAR	
	ASR(%)↓	Acc(%)↑	ASR(%)↓	Acc(%)↑	ASR(%)↓	Acc(%)↑	ASR(%)↓	Acc(%)↑
RobustGCN	90.09	60.38	70.95	56.08	90.35	56.18	93.35	54.58
GNNGuard	94.66	62.29	0.04	62.58	95.21	64.61	88.12	62.14
RS	41.18	58.44	42.72	58.48	40.30	58.76	89.36	52.53
A^{2}GBD	0.07	63.80	0.07	65.42	0.12	67.00	0.17	65.08

robustness against natural noise, but does not explicitly suppress anomalous feature patterns introduced by backdoor triggers; GNNGuard performs edge reweighting based on neighborhood similarity, while backdoor triggers can be constructed to preserve local structural and feature consistency, thereby bypassing similarity-driven defenses; randomized smoothing (RS) focuses on robustness under random noise and lacks direct constraints on structured and targeted backdoor triggers, making it difficult to substantially reduce ASR in this setting.

In contrast, A^{2}GBD consistently suppresses the attack success rate across different attack scenarios while maintaining relatively high classification accuracy, demonstrating stable defense behavior on large-scale graph data.

D Hyperparameter Analysis

To investigate the impact of the candidate mechanism on the exploration space of reinforcement learning, we evaluate different candidate sizes K on the Cora dataset (Figure 4). The results indicate that when K is small, the exploration space is limited, leaving some poisoned nodes uncovered, which leads to higher ASR. Conversely, when K is too large, although ACC remains largely stable, redundant nodes significantly increase, expanding the exploration space and reducing efficiency. Notably, a moderate candidate size (e.g., $K = 50$) achieves an optimal balance between defense performance and computational efficiency. These results validate our proposed motivation: by controlling the candidate set size, the RL exploration space can be effectively regulated, enabling more efficient and stable defense in multi-source trigger scenarios.

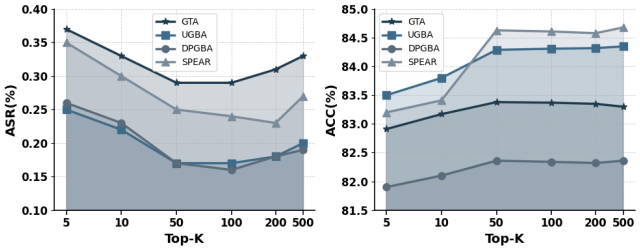


Figure 4: Impact of candidate size K on the exploration space of reinforcement learning in A^{2}GBD. Smaller K values restrict exploration and overlook poisoned nodes, while overly large K values introduce redundant candidates and reduce efficiency. A balanced setting (e.g., $K = 50$) demonstrates the effectiveness of our candidate mechanism in adaptively regulating exploration for stable and efficient defense.