RELIABILITY-GUIDED SEMANTIC MEMORY FOR ITERATIVE GRAPH PURIFICATION UNDER STRUCTURAL ATTACKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Iterative graph purification can repair adversarial edge perturbations before classification, but it typically reuses the same node-feature conditioning at every round. As connectivity changes, fixed feature conditioning cannot incorporate the semantic evidence exposed by progressively repaired neighborhoods. This paper introduces Reliability-Guided Semantic Memory for Graph Purification (RGSM-GP), which augments iterative graph purification with a semantic state that evolves across rounds. The method estimates classifier-free node reliability from structural correction residuals, edge-posterior uncertainty, and temporal instability across purification rounds. A budget-normalized soft gate limits updates to unreliable candidates, while reliability-weighted neighborhoods supply the replacement context. The updated memory conditions the next structural step, while raw features remain fixed anchors for training and final prediction. Under a common multi-seed evaluation pipeline on Cora, Citeseer, and PubMed, RGSM-GP attains the highest robust-average point estimate in six of six PRBCD/LRBCD settings among GPR-GAE, AT-GCN, RUNG, and GNNGuard, and also ranks first at the 50% perturbation budget in five settings. In the direct multi-seed comparison with baselines, all six completed robust-average differences favor RGSM-GP, including an improvement of 5.94 percentage points on Cora under PRBCD.

1 INTRODUCTION

Message passing allows graph neural networks (GNNs) to exploit relational structure, but it also exposes their predictions to small changes in graph connectivity (Kipf & Welling, 2017; Zügner & Günnemann, 2019). By adding or deleting a limited number of edges, an adversary can redirect neighborhood aggregation without modifying observed node attributes. Scalable attacks such as PRBCD (Geisler et al., 2021) make this threat practical on larger graphs. Defenses respond either by modifying the classifier (Zhu et al., 2019; Zhang & Zitnik, 2020) or by repairing the graph before classification.

Graph adversarial purification follows the second route. A separately trained purifier restores a corrupted adjacency matrix and supplies the repaired graph to a standard classifier. This separation limits coupling between robustness and classifier training, allowing the purifier to serve as a reusable defense layer. Recent self-supervised purification performs iterative structural restoration without classifier supervision (Lee & Park, 2025). Yet the common purification state is asymmetric: the graph evolves across rounds, while node features remain fixed as conditioning information.

Although structural attacks do not directly alter raw node attributes, they can distort the semantic information derived from neighborhood aggregation. When carried into later purification rounds, this information may retain distortions introduced by uncertain or only partly repaired neighborhoods. As purification proceeds, previously extracted semantic information may fail to reflect improvements in the graph structure. This motivates updating the semantic context used for purification while preserving raw attributes as fixed observational anchors. A fixed feature input may limit how refinements to the neighborhood are reflected in subsequent feature conditioning. However, reconstructing all node attributes risks altering useful information in the original observations.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

This motivates using reliability estimates to guide semantic updates during purification. At defense time, the defender has no access to the clean graph or node labels and does not know which nodes were affected by the attack. It therefore cannot directly determine which semantic states require revision. We use the purification trajectory to estimate which states warrant updating, limit the magnitude of each update, and feed the revised context into subsequent rounds. This approach aims to preserve useful observed information while allowing semantic updates to inform further structural refinement. Existing defenses filter edges by feature similarity (Wu et al., 2019; Zhang & Zitnik, 2020), jointly learn graph structure and a classifier (Jin et al., 2020), or reconstruct masked attributes (Hou et al., 2022; Shen et al., 2025). Recent unsupervised defenses also construct a self-refined normal graph to reassess structural pollution (Feng et al., 2025). Our focus is on maintaining semantic memory across purification rounds and guiding its updates with reliability estimates. We therefore investigate whether purification dynamics can indicate which semantic states warrant revision and how the revised memory should condition the next round.

We propose Reliability-Guided Semantic Memory for Graph Purification (RGSM-GP), which extends iterative graph purification with a semantic state that evolves across rounds. Each round combines fixed raw features with the current semantic memory to condition an existing structural purifier. Structural correction residuals, edge-posterior uncertainty, and temporal instability across rounds yield a classifier-free node unreliability score. Candidate screening selects nodes estimated to be unreliable, and a budget-normalized soft gate controls the magnitude of their memory updates. Reliability-weighted neighborhood aggregation supplies context for these updates. The revised memory then conditions the next structural update. After the final round, the gate and reliability scores control a bounded nodewise fusion of the memory with the raw features for prediction. RGSM-GP thus couples structural purification with reliability-guided semantic memory updates while reusing the structural edge estimator (Lee & Park, 2025).

Our main contributions are summarized as follows:

- **Semantic feedback across purification rounds.** We introduce RGSM-GP, which updates semantic memory across purification rounds and feeds the revised state back into subsequent structural purification.
- **Classifier-free reliability estimation.** We derive node reliability estimates from structural correction residuals, edge-posterior uncertainty, and temporal instability across rounds. This estimation requires no clean graph, attack indicators, node labels, or classifier confidence.
- **Budget-controlled selective memory updates.** We combine candidate screening with a budget-normalized soft gate to control which memory states are updated and by how much. Updates draw on reliability-weighted neighborhood aggregation while keeping raw attributes unchanged. Bounded nodewise fusion controls the contribution of the final memory to the classifier input.

2 Related Work

2.1 Structural Attacks and Model Defenses

Structural attacks disrupt message passing by adding or deleting edges. MetaAttack (Zügner & Günnemann, 2019) optimizes global poisoning attacks. PRBCD (Geisler et al., 2021) scales global attacks to large graphs. LRBCD (Gosch et al., 2023) incorporates both graph-level and node-level perturbation constraints when jointly attacking multiple nodes. We evaluate robustness under PRBCD and LRBCD.

Model defenses modify representations, aggregation, or training, as in RGCN (Zhu et al., 2019), RUNG (Hou et al., 2024), and NoisyGNN (Ennadir et al., 2024a). GNNGuard (Zhang & Zitnik, 2020) estimates neighbor importance and smooths edge weights through layer-wise graph memory. HANG (Zhao et al., 2023) uses conservative Hamiltonian graph neural flows; WGTL (Arafat et al., 2025) combines local and global witness-complex representations with topological regularization. Weight initialization and training duration also affect robustness (Ennadir et al., 2024b). Adversarial training optimizes classifiers on perturbed graphs (Gosch et al., 2023); inverse reinforcement learning can reconstruct attack policies to generate additional training samples (Lyu et al., 2024). Under

a cost-aware threat model, RisKeeper (Liao et al., 2024) learns node-specific protection levels that determine attack costs.

2.2 Graph Purification and Semantic Repair

Graph structure can be refined through preprocessing or joint learning. Feature-similarity preprocessing removes edges between dissimilar nodes (Wu et al., 2019); GUARD (Li et al., 2023) applies a precomputed, node-agnostic defensive patch. ProGNN (Jin et al., 2020) jointly learns graph structure and a classifier under sparsity, low-rank, and feature-smoothness priors. Feng et al. (2025) construct a rough graph and use Graph Pollution Degree to assess the trustworthiness of refined structures. SG-GSR (In et al., 2024) extracts a reliable subgraph, then uses augmentation and group training to address information loss and degree imbalance. Grimm (Liu et al., 2025) detects and rectifies adversarial edges using feature trajectories observed during GNN training.

Related representation-learning methods address semantic information in different settings. GRANCE (Zhuang et al., 2025) combines dual neighborhood contrastive learning with a neighbor estimator to learn robust representations. GraphMAE (Hou et al., 2022) learns representations through masked feature reconstruction. ACGMAE (Shen et al., 2025) combines adversarial feature masking with contrastive learning to defend against both structural and feature attacks. DA-GNN (In et al., 2025) models dependencies among feature, structure, and label noise, a broader corruption setting than the structural-only attacks studied here.

Iterative graph purification refines graph structure before classification. DiffSP (Luo et al., 2025) combines selective noising with transfer-entropy-guided denoising for evasion defense. GDDM (He et al., 2026) combines diffusion-based structural refinement with feature- and degree-based edge filtering and is evaluated primarily under poisoning attacks. GPR-GAE (Lee & Park, 2025) uses multiple Generalized PageRank filters for self-supervised, classifier-independent purification, keeping node-feature inputs fixed across rounds. RGSM-GP maintains node-level semantic memory that is updated using reliability estimates from purification dynamics and feeds the revised state into subsequent structural purification. This memory evolves across purification rounds, whereas GNNGuard's graph memory smooths edge weights across classifier layers (Zhang & Zitnik, 2020).

2.3 Adaptive Evaluation

Robustness estimates depend on both the evaluation protocol and attack design. Gosch et al. (2023) show how access to clean test nodes during training can inflate robustness estimates. Separately, non-adaptive attacks can overestimate the robustness of graph defenses (Mujkanovic et al., 2022), and obfuscated gradients can hinder attack optimization (Athalye et al., 2018). Attacks should therefore be adapted to the complete defense (Tramer et al., 2020), as in the evaluations of RUNG (Hou et al., 2024) and GOOD-AT (Li et al., 2024). EvA (Akhondzadeh et al., 2026) searches discrete edge perturbations without gradients and supports black-box models and non-differentiable objectives; it is outside our PRBCD/LRBCD evaluation. For RGSM-GP, we evaluate adaptive attacks with different gradient computations, including a BPDA-style approximation, and report transferred attacks separately (Appendix A.2).

3 Problem Formulation

We study robust node classification on an undirected attributed graph $G = (V, A, X)$, with $N = |V|$ nodes and adjacency matrix $A \in \{0,1\}^{N\times N}$. The observed attributes are $X \in \mathbb{R}^{N\times F}$, where F is the feature dimension. A test-time structural attack produces A' by adding or deleting at most Δ undirected edges between distinct nodes, leaving V and X unchanged:

$$\sum_{i<j} |A'_{ij} - A_{ij}| \leq \Delta.$$

At inference, the defense observes (A', X) without access to the clean adjacency or the modified-edge set. Comparisons between clean and attacked graphs are used only for post-hoc diagnostics.

Although X is unchanged, perturbed neighborhoods can contaminate the information aggregated at a node and hence its semantic representation. Structural purification changes these neighborhoods

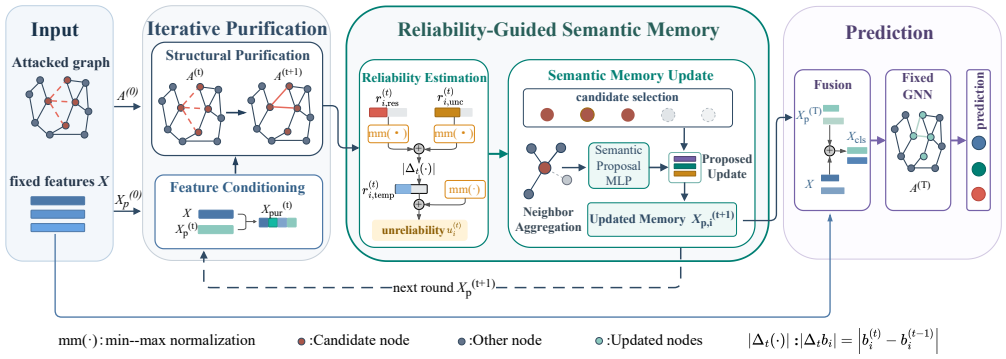

Figure 1: Reliability-guided semantic memory within iterative graph purification. Fixed raw features and evolving semantic memory condition the structural purifier, which updates the graph state. Classifier-free unreliability drives candidate screening and a budget-normalized soft gate, while reliable neighbors supply repair context. The revised memory feeds the next round; final gated fusion with raw features forms X_{cls} for the fixed GNN.

across rounds, so the available semantic context evolves with the graph. The challenge is to determine which node semantics require revision and how much to update them without knowing the true contamination pattern. We therefore seek to improve classification under attack by selectively updating semantic memory using reliability inferred from purification dynamics. The revised memory should inform subsequent structural updates while retaining the observed attributes as fixed anchors.

The defense uses an existing iterative structural purifier P_θ and a node classifier f_ψ trained independently of the purifier. We initialize the weighted structural state as $A^{(0)} = A'$ and the semantic memory as $X_p^{(0)} = X$. At round t, $A^{(t)}$ represents the current structure, and $X_p^{(t)} \in \mathbb{R}^{N\times F}$ stores semantic information carried across rounds to condition structural purification. Model parameters and observed attributes remain fixed during inference; only the structural and semantic-memory states are updated. These updates use no node labels, classifier confidence, or attack-touched node indicators. After T executed updates, prediction is $\widehat{Y} = f_\psi(A^{(T)}, X_{\text{cls}})$, where X_{cls} combines X with the final memory $X_p^{(T)}$ through bounded nodewise fusion. The update rules and fusion mechanism are specified in the following section.

4 Methodology

4.1 Feature Conditioning and Structural Purification

Figure 1 traces how RGSM-GP couples structural purification with semantic memory under the structural-only threat model. Starting from $A^{(0)} = A'$ and $X_p^{(0)} = X$, each round first conditions the structural purifier on the current memory. The feature-conditioning block combines this memory with the fixed observations,

$$X_{\text{pur}}^{(t)} = (1-\alpha_f)X + \alpha_f X_p^{(t)}. \tag{1}$$

Here t indexes the purification round, $X_{\text{pur}}^{(t)}$ is its feature input, and $\alpha_f \in [0,1]$ controls the memory contribution. The first round therefore uses the raw features, whereas later rounds can incorporate revised memory without overwriting X.

The structural block then computes $A^{(t+1)} = P_\theta(A^{(t)}, X_{\text{pur}}^{(t)})$, where θ denotes the purifier parameters. We use the bidirectional link predictor inherited from GPR-GAE to perform this structural update. Its output contains edge weights in $[0,1]$ on the fixed support of A'; deleted edges outside this support are not reinserted. The transition from $A^{(t)}$ to $A^{(t+1)}$ supplies the diagnostic information used by the reliability block.

4.2 Reliability Estimation from Purification Dynamics

The reliability block assesses which node memories warrant revision using three signals from the purification trajectory. For node i, let $d_i^{(t)} = \sum_j A_{ij}^{(t)}$ denote its current weighted degree. The structural residual $r_{i,\mathrm{res}}^{(t)}$ sums $|A_{ij}^{(t+1)} - A_{ij}^{(t)}|$ over neighbors j, then divides by $\max(d_i^{(t)}, \delta)$. The positive constant δ prevents division by zero throughout the method. This residual measures the extent of local structural correction, without identifying whether any particular edge was attacked.

The uncertainty signal $r_{i,\mathrm{unc}}^{(t)}$ instead sums $4A_{ij}^{(t+1)}(1 - A_{ij}^{(t+1)})$ and uses the same degree denominator. Each contribution peaks at an edge posterior of 0.5, capturing uncertainty in the purifier's current prediction. We normalize each signal across nodes using $\mathrm{mm}(\cdot)$, which maps its minimum and maximum to zero and one. Numerically constant signals are mapped to zero to avoid amplifying a vanishing range.

The nonnegative weights w_s and w_u define the base score $b_i^{(t)} = w_s\,\mathrm{mm}(r_{i,\mathrm{res}}^{(t)}) + w_u\,\mathrm{mm}(r_{i,\mathrm{unc}}^{(t)})$. Temporal instability compares this score across rounds: $r_{i,\mathrm{temp}}^{(t)} = |b_i^{(t)} - b_i^{(t-1)}|$ for $t \geq 1$, and zero initially. Thus, the temporal branch in the framework tracks changes in the base diagnosis. The combined unreliability score is

$$u_i^{(t)} = b_i^{(t)} + w_t\,\mathrm{mm}(r_{i,\mathrm{temp}}^{(t)}), \tag{2}$$

where $w_t \geq 0$ weights temporal instability. We convert this score to the reliability weight $c_i^{(t)} = \mathrm{clip}_{[10^{-8},1]}(e^{-\kappa u_i^{(t)}})$, where $\kappa > 0$ sets the decay rate. The operator $\mathrm{clip}_{[a,b]}$ truncates a scalar to the interval $[a, b]$. These weights summarize relative purification behavior, without requiring labels, classifier confidence, or known attack indicators. Unreliability drives candidate screening, while reliability controls neighbor contributions and memory-update strength; neither quantity is a calibrated contamination probability.

4.3 Reliability-Guided Semantic Memory Update

The memory-update block separates the nodes eligible for revision from the amount of update mass allocated to them. Let $p^{(t)}$ be the mean of the largest $\lceil qN \rceil$ unreliability scores, where $q \in (0, 1]$ sets the fraction summarized. This graph-level pressure adjusts a candidate fraction $\rho_{\mathrm{cand}}^{(t)}$ and an update fraction $\rho_{\mathrm{upd}}^{(t)}$. The candidate fraction is $\rho_c^- + \lambda_c p^{(t)}$, clipped to $[\rho_c^-, \rho_c^+]$. The update fraction is zero when $p^{(t)} < \tau_{\mathrm{safe}}$; otherwise, it is $\rho_u^- + \lambda_b p^{(t)}$, clipped to $[\rho_u^-, \rho_u^+]$. Here the ρ bounds limit each fraction, $\lambda_c, \lambda_b \geq 0$ set their response slopes, and τ_{safe} suppresses updates at low pressure.

Candidate selection retains the $K_c^{(t)}$ highest-scoring nodes in $\mathcal{C}^{(t)}$, with $K_c^{(t)} = \lceil \rho_{\mathrm{cand}}^{(t)} N \rceil$ clipped to $[1, N]$. The nominal gate budget is $B^{(t)} = \min\{K_c^{(t)}, \lceil \rho_{\mathrm{upd}}^{(t)} N \rceil\}$. For positive budgets, $\tau^{(t)}$ is the smallest score among the top-$B^{(t)}$ candidates. We first assign $\bar{m}_i^{(t)} = \mathbb{1}[i \in \mathcal{C}^{(t)}]\sigma((u_i^{(t)} - \tau^{(t)})/\beta)$, where $\mathbb{1}$ is the membership indicator, σ is the logistic sigmoid, and $\beta > 0$ is its temperature. The budget-normalized soft gate is

$$m_i^{(t)} = \mathrm{clip}_{[0,1]}\left(\frac{B^{(t)}\bar{m}_i^{(t)}}{\max(\sum_j \bar{m}_j^{(t)}, \delta)}\right). \tag{3}$$

All gates are zero when $B^{(t)} = 0$, leaving the current memory unchanged. Otherwise, normalization followed by clipping gives $\sum_i m_i^{(t)} \leq B^{(t)}$ and zero weights outside the candidate pool. The budget therefore bounds total gate mass, while individual candidates can receive different continuous weights.

The reliable-neighbor branch supplies context for those updates using the new structure and the current, not-yet-updated memory. For node i, the context vector is

$$z_i^{(t)} = \frac{\sum_j A_{ij}^{(t+1)}(c_j^{(t)})^\gamma x_{p,j}^{(t)}}{\max(\sum_j A_{ij}^{(t+1)}(c_j^{(t)})^\gamma, \delta)}. \tag{4}$$

Here $x_{p,j}^{(t)} \in \mathbb{R}^F$ is row j of the memory matrix, and $\gamma \geq 0$ controls neighbor-reliability reweighting. Neighbors contribute according to both their structural connection weight and their estimated reliability. The semantic repair MLP produces $\widetilde{x}_i^{(t)} = g_\phi([x_{p,i}^{(t)} \| z_i^{(t)}])$, where $\|$ denotes concatenation. The two-layer network g_ϕ, parameterized by ϕ, returns a proposal with the same feature dimension.

The target node's own reliability determines how strongly this proposal changes its memory. Define the update coefficient $\eta_i^{(t)} = \alpha_x m_i^{(t)}(1 - c_i^{(t)})$, with maximum step $\alpha_x \in [0, 1]$. The resulting convex update is

$$x_{p,i}^{(t+1)} = (1 - \eta_i^{(t)}) x_{p,i}^{(t)} + \eta_i^{(t)} \widetilde{x}_i^{(t)}. \tag{5}$$

Thus, neighbor reliability weights the information source, whereas target reliability and the soft gate control the update coefficient. Nodes outside the candidate pool retain their current memory, and the raw observations X remain fixed throughout purification.

4.4 Iterative Inference and Training

The dashed feedback path in Figure 1 returns $X_p^{(t+1)}$ to feature conditioning for the next structural step. Equation (1) then combines this revised memory with X, while the structural state advances to $A^{(t+1)}$. The same learned parameters are reused across rounds, so inference updates states without optimizing the purifier or semantic network. After each structural and memory update, we check the relative change in total edge weight. Let $s_t = \max(\sum_{ij} A_{ij}^{(t)}, \delta)$ denote the current total with its numerical floor. We divide the absolute difference between the new total and s_t by s_t. Iteration stops when this ratio is at most $\varepsilon_{\text{stop}}$, or after $T_{\max}$ rounds. The tolerance $\varepsilon_{\text{stop}}$ defines an edge-mass stopping heuristic, not a guarantee of joint-state convergence.

After T executed rounds, the prediction block combines the final memory with the fixed-feature path shown beneath the iterative loop. Let $\ell = T - 1$ index the final diagnostic step, and set $\lambda_i = \text{clip}_{[0,\lambda_{\max}]}(\alpha_{\text{cls}} m_i^{(\ell)}(1 - c_i^{(\ell)}))$. Here $\alpha_{\text{cls}} \geq 0$ scales the fusion weight and $0 \leq \lambda_{\max} < 1$ caps the memory contribution. The classifier input is

$$X_{\text{cls},i} = (1 - \lambda_i) X_i + \lambda_i X_{p,i}^{(T)}, \tag{6}$$

where X_i and $X_{p,i}^{(T)}$ denote the corresponding feature and memory rows. The fixed classifier then produces $\widehat{Y} = f_\psi(A^{(T)}, X_{\text{cls}})$, receiving both the final structure and the fused features. This prediction-stage fusion retains an explicit raw-feature contribution, separately from the feedback used during structural purification.

Training objective. Training uses sampled positive and negative edges with structural masking and reweighting, while the observed features provide fixed semantic targets. We retain the backbone loss $\mathcal{L}_{\text{str}}$, comprising positive/negative link binary cross-entropy and a weighted direction-agreement mean squared error. The semantic branch uses a single update from $X_p^{(0)} = X$, so the initial temporal signal is zero. Its gate-weighted reconstruction objective aligns semantic proposals with the observed feature directions,

$$\mathcal{L}_{\text{sem}} = \frac{\sum_i m_i^{(0)} [1 - \cos(\widetilde{x}_i^{(0)}, X_i)]^{\alpha_{\text{sce}}}}{\max(\sum_i m_i^{(0)}, \delta)}. \tag{7}$$

Here cos denotes cosine similarity, and $\alpha_{\text{sce}} > 0$ controls the exponent applied to the cosine reconstruction error. A preservation term, $\mathcal{L}_{\text{pres}} = N^{-1} \sum_i c_i^{(0)} \|x_{p,i}^{(1)} - X_i\|_2^2$, adds a reliability-weighted penalty on deviation from the original features. The symbol $\|\cdot\|_2$ denotes the Euclidean norm. Combining these terms yields the training objective

$$\mathcal{L} = \mathcal{L}_{\text{str}} + \lambda_{\text{sem}} \mathcal{L}_{\text{sem}} + \lambda_{\text{pres}} \mathcal{L}_{\text{pres}}, \tag{8}$$

where the nonnegative coefficients λ_{sem} and λ_{pres} weight the two semantic objectives. Structural weights and edge posteriors are detached before entering the semantic branch. Consequently, $\mathcal{L}_{\text{str}}$ updates θ, whereas the semantic and preservation terms update ϕ without back-propagating through the structural purifier. Training does not unroll the feedback trajectory; deployment reuses the learned networks over successive state updates. The classifier is trained independently, and classification labels do not supervise purification. Experimental settings in Section 5.1 specify the architectures, optimization parameters, and update controls.

Table 1: Defended test accuracy (%) under defense-specific attacks. Entries show means and sample standard deviations over complete runs. Bold entries identify the highest mean in each column within a setting.

Method	Clean	10%	25%	50%	Robust avg.
Cora – PRBCD					
GPR-GAE	79.49±1.32	72.89±1.60	66.91±4.03	61.17±1.94	66.99±2.09
AT-GCN	**81.50±2.29**	63.64±1.28	49.54±3.57	34.07±2.71	49.08±2.04
RUNG	80.49±3.81	67.95±2.95	57.69±1.65	42.95±2.27	56.20±2.21
GNNGuard	79.76±4.06	66.85±2.21	56.50±1.17	46.79±3.04	56.72±1.38
RGSM-GP	78.88±0.56	**76.56±3.19**	**72.41±3.81**	**69.84±0.92**	**72.93±2.26**
Cora – LRBCD					
GPR-GAE	79.35±1.88	75.39±2.86	74.09±2.77	72.36±2.01	73.95±2.42
AT-GCN	79.58±1.65	72.34±3.51	66.94±4.73	61.63±2.91	66.97±3.25
RUNG	**80.49±3.81**	**76.83±2.34**	74.54±2.25	72.34±3.26	74.57±2.60
GNNGuard	79.76±4.06	70.60±3.21	63.46±1.85	58.24±0.90	64.10±1.68
RGSM-GP	79.25±1.93	76.39±2.65	**74.73±2.42**	**73.43±2.40**	**74.85±2.35**
Citeseer – PRBCD					
GPR-GAE	67.31±4.19	65.93±4.13	63.54±5.07	57.91±4.33	62.46±3.66
AT-GCN	66.57±3.46	51.79±3.43	40.52±4.75	27.61±3.50	39.98±3.87
RUNG	65.82±3.53	53.81±5.17	44.33±5.92	32.91±7.01	43.68±6.00
GNNGuard	66.72±2.86	62.24±2.21	59.63±2.69	52.99±3.54	58.28±2.62
RGSM-GP	**68.26±4.05**	**67.40±3.78**	**65.70±4.84**	**62.92±3.75**	**65.34±4.03**
Citeseer – LRBCD					
GPR-GAE	67.54±4.04	65.78±4.06	65.18±4.28	**65.02±4.74**	65.33±4.32
AT-GCN	64.55±4.52	60.75±5.26	56.64±6.46	53.58±5.80	56.99±5.68
RUNG	65.82±3.53	61.49±4.31	58.13±5.09	55.00±6.23	58.21±5.16
GNNGuard	66.72±2.86	61.27±4.29	57.01±4.47	55.30±3.56	57.86±4.09
RGSM-GP	**67.84±4.02**	**66.78±3.73**	**65.86±3.54**	64.72±4.02	**65.79±3.73**
PubMed – PRBCD					
GPR-GAE	74.34±5.00	71.43±5.17	70.80±4.97	69.45±5.35	70.56±5.07
AT-GCN	76.05±4.86	54.82±3.30	40.09±2.25	37.39±1.60	44.10±2.11
RUNG	**79.17±5.30**	63.49±3.95	53.48±2.08	52.57±2.29	56.52±2.68
GNNGuard	72.47±6.48	70.27±5.71	64.09±8.19	59.11±3.71	64.49±5.58
RGSM-GP	74.32±5.07	**72.18±5.01**	**72.62±4.69**	**71.31±4.77**	**72.04±4.74**
PubMed – LRBCD					
GPR-GAE	74.24±4.96	73.04±4.86	72.67±4.99	72.12±4.74	72.61±4.82
AT-GCN	76.23±4.86	72.55±4.83	70.57±4.12	67.93±3.25	70.35±4.05
RUNG	**79.17±5.30**	**76.13±4.89**	73.61±4.44	71.44±4.38	73.72±4.56
GNNGuard	72.47±6.48	68.87±5.63	68.04±5.49	68.35±5.64	68.42±5.58
RGSM-GP	77.06±2.58	75.10±2.63	**74.69±2.47**	**74.56±2.51**	**74.78±2.54**

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

We evaluate robust node classification on three public citation networks: Cora, Citeseer, and PubMed. Dataset statistics and split details are provided in Appendix A.1. The comparison includes GPR-GAE (Lee & Park, 2025), AT-GCN (Gosch et al., 2023), RUNG-MCP (Hou et al., 2024), GNNGuard (Zhang & Zitnik, 2020), and RGSM-GP. We use the label RUNG for RUNG-MCP in all result tables and figures. The methods share dataset splits within each seed and the evaluation interface; their training and defense operations follow their respective implementations. Appendix A documents the split construction and method-specific settings.

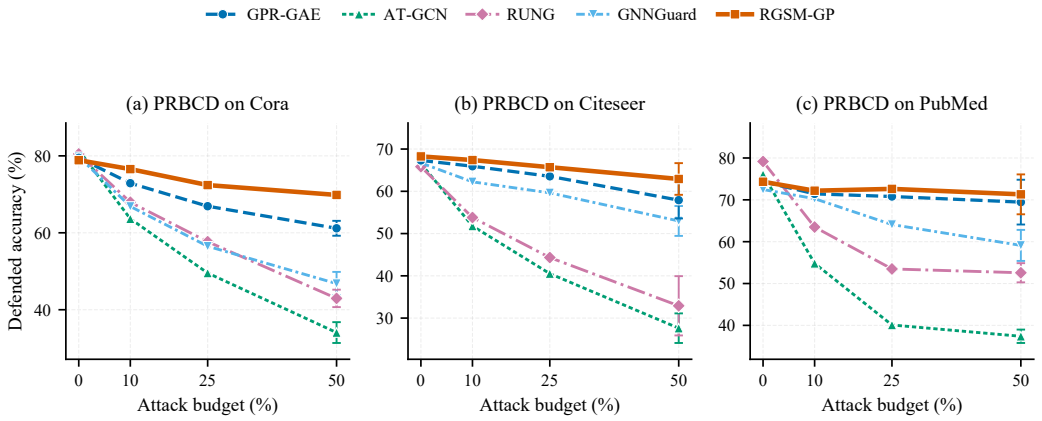

Figure 2: Defended accuracy as a function of PRBCD attack budget. Curves visualize the PRBCD results in Table 1. Points show means over each method's complete runs; error bars at the 50% budget indicate one sample standard deviation. Attacks are generated separately for each defense under the settings in Appendix A.2.

We apply PRBCD (Geisler et al., 2021) and LRBCD (Gosch et al., 2023) at attack budgets of 0%, 10%, 25%, and 50%. Each percentage is relative to the number of undirected edges incident to the test nodes. Attacks are optimized separately for each defense; loss functions and attack-path settings are detailed in Appendix A.2. The full RGSM-GP model uses soft gating, independently of the hard-gate ablation and supplementary gradient-path checks. Its semantic network has hidden width 256, and inference executes at most five rounds using the stopping rule in Section 4.4. Complete optimization and update settings are provided in Appendix A.1.

Clean accuracy denotes defended test accuracy at zero attack budget. For each seed, robust average is the mean accuracy over the three nonzero budgets. We report means and sample standard deviations across complete runs. Results are averaged over at least ten distinct random seeds for the main experiments and at least three for all other experiments.

5.2 Node Classification

Table 1 reports classification accuracy for all six dataset–attack settings. RGSM-GP attained the highest robust-average point estimate in all six settings and the highest 50%-budget accuracy in five. The exception was Citeseer–LRBCD, where its 64.72% accuracy was 0.30 percentage points below the highest comparison mean. On Cora–PRBCD, its gains over the strongest comparison means were 5.94 percentage points in robust average and 8.67 points at the 50% budget. Under LRBCD, the robust-average gains were smaller: 0.28, 0.46, and 1.06 points on Cora, Citeseer, and PubMed, respectively. These rankings describe the observed means; they do not establish pairwise significance across all methods.

Figure 2 complements the table by showing how accuracy changes as the PRBCD budget increases. On Cora and Citeseer, RGSM-GP declines more gradually than the compared methods, and its margin over GPR-GAE widens with increasing nonzero budgets. On PubMed, both RGSM-GP and GPR-GAE remain relatively stable across the nonzero budgets, whereas AT-GCN and RUNG lose substantial accuracy by the 25% budget. The curves thus reveal dataset-dependent sensitivity to attack strength, with the clearest widening of the margin over GPR-GAE occurring on Cora.

Higher clean accuracy does not necessarily imply higher robust-average accuracy: on Cora–PRBCD, RGSM-GP has the lowest clean accuracy among the compared methods but the highest robust average (Table 1). Figure B1 in Appendix B.1 provides the full clean–robust comparison.

Table 2: PRBCD ablation results on Cora and Citeseer. Entries report full-model minus control robust-average accuracy (Δ_{acc}, in percentage points), with 95% paired t confidence intervals from 11 paired runs per dataset. Positive values indicate lower accuracy after ablation.

Ablation setting	Cora Δ_{acc} [95% CI]	Citeseer Δ_{acc} [95% CI]
w/o temporal instability	$+3.71$ $[+1.43, +5.98]$	$+2.59$ $[+1.65, +3.53]$
structural residual only	$+2.81$ $[+1.58, +4.04]$	$+2.60$ $[+1.39, +3.81]$
fixed gate	$+1.59$ $[+0.25, +2.92]$	$+2.30$ $[+1.19, +3.41]$
hard gate	$+1.82$ $[+0.42, +3.22]$	$+2.15$ $[+1.44, +2.86]$
w/o semantic feedback	$+2.00$ $[+0.43, +3.57]$	$+2.48$ $[+1.26, +3.70]$

5.3 Ablation and Mechanistic Analysis

We examine the reliability signals, update gate, and cross-round feedback using the existing 11-run paired PRBCD suite on Cora and Citeseer. Table 2 reports Δ_{acc}, the robust-average accuracy of the full model minus that of each control, in percentage points. Positive values indicate a loss after replacing or removing the specified design. Control definitions and diagnostic details are provided in Appendix B.2.

Reliability signals. Removing temporal instability reduced robust average by 2.59 points on Citeseer and 3.71 points on Cora. Using structural residuals alone reduced it by 2.60 and 2.81 points, respectively. These controls support the combined signals used to guide memory updates in the evaluated settings. They measure the contribution to classification accuracy, rather than directly measuring the accuracy of reliability estimation.

Gating and feedback. Replacing adaptive selection with the fixed-schedule gate reduced robust average by 2.30 points on Citeseer and 1.59 points on Cora. Hard gating reduced it by 2.15 and 1.82 points, respectively. Disabling semantic feedback reduced robust average by 2.48 and 2.00 points. The latter control retains memory updates but removes their use in subsequent structural conditioning, testing the feedback path in Figure 1. All intervals for these reported controls remained above zero.

Reliability diagnostics and update behavior. Across the finalized nonzero-budget diagnostic cells, mean touched-node ROC–AUC ranged from 0.68 to 0.78. Candidate pools covered 11.4–16.2% of nodes, while mean interpolation coefficients $\eta_i^{(t)}$ ranged from 0.0003 to 0.0029. These measurements indicate selective screening and conservative update coefficients; they do not establish a bound on actual feature displacement. Touched nodes provide a localization proxy, not ground-truth semantic contamination labels. Moreover, four of nine Cora–PRBCD evaluations assigned zero semantic update mass. Accordingly, the main results establish full-system performance, while the matched controls provide the more specific evidence for the update and feedback mechanisms. Diagnostic definitions and the separate soft/hard-gate curve are given in Appendix B.2.

6 Conclusion

We propose RGSM-GP, a classifier-independent extension that brings reliability-guided semantic memory to iterative graph purification. It keeps raw features as fixed anchors and uses correction residuals, posterior uncertainty, and temporal instability to select semantic states for conservative, budget-normalized updates before the next structural-repair round. Experiments across PRBCD and LRBCD yield the highest robust-average point estimate in six of six settings and the highest 50%-budget accuracy in five; matched controls further support cross-round feedback, temporal instability, and the soft gate as complementary components. Together, these results establish RGSM-GP as a selective enhancement for iterative graph purification. Building on these results, future work can investigate complementary reliability signals and evaluate their integration with a broader range of iterative graph-purification backbones. Such extensions could make cross-round semantic updates more robust across diverse attack settings.

AI USE STATEMENT

During the preparation of this manuscript, we used AI tools, including ChatGPT, to refine the wording and improve the clarity and fluency of the text. These tools also assisted in searching for and collecting background materials and references relevant to this study. The authors take full responsibility for the content of the manuscript, including its scientific accuracy and the appropriate attribution of all cited sources.

REFERENCES

Mohammad Sadegh Akhondzadeh, Soroush H. Zargarbashi, Jimin Cao, and Aleksandar Bojchevski. Eva: Evolutionary attacks on graphs. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 64694–64723, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/69ffb2b7055546c4c2d559025286675b-Paper-Conference.pdf.

Naheed Anjum Arafat, Debabrota Basu, Yulia Gel, and Yuzhou Chen. When witnesses defend: A witness graph topological layer for adversarial graph learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 15408–15416, 2025. doi: 10.1609/aaai.v39i15.33691.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 274–283. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/athalye18a.html.

Sofiane Ennadir, Yassine Abbahaddou, Johannes F. Lutzeyer, Michalis Vazirgiannis, and Henrik Boström. A simple and yet fairly effective defense for graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 21063–21071, 2024a. doi: 10.1609/aaai.v38i19.30098. URL https://doi.org/10.1609/aaai.v38i19.30098.

Sofiane Ennadir, Johannes Lutzeyer, Michalis Vazirgiannis, and El Houcine Bergou. If you want to be robust, be wary of initialization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 23796–23823. Curran Associates, Inc., 2024b. doi: 10.52202/079017-0750. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/2a6fc6774932e7835349089b0287ed2c-Paper-Conference.pdf.

Bingdao Feng, Di Jin, Xiaobao Wang, Dongxiao He, Jingyi Cao, and Zhen Wang. Exploiting self-refining normal graph structures for robust defense against unsupervised adversarial attacks. In James Kwok (ed.), *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 2793–2801. International Joint Conferences on Artificial Intelligence Organization, 8 2025. doi: 10.24963/ijcai.2025/311. URL https://doi.org/10.24963/ijcai.2025/311. Main Track.

Simon Geisler, Tobias Schmidt, Hakan Şirin, Daniel Zügner, Aleksandar Bojchevski, and Stephan Günnemann. Robustness of graph neural networks at scale. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 7637–7649. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/3ea2db50e62ceefceaf70a9d9a56a6f4-Paper.pdf.

Lukas Gosch, Simon Geisler, Daniel Sturm, Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. Adversarial training for graph neural networks: Pitfalls, solutions, and new directions. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 58088–58112. Curran Associates, Inc., 2023. doi: 10.52202/075280-2532. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/b5a801e6bc4f4ffa3e6786518a324488-Paper-Conference.pdf.

Xin He, Wenqi Fan, Yili Wang, Chengyi Liu, Rui Miao, Xin Juan, and Xin Wang. Graph defense diffusion model. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, KDD '26, pp. 416–427, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400722585. doi: 10.1145/3770854.3780207. URL https://doi.org/10.1145/3770854.3780207.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pp. 594–604, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393850. doi: 10.1145/3534678.3539321. URL https://doi.org/10.1145/3534678.3539321.

Zhichao Hou, Ruiqi Feng, Tyler Derr, and Xiaorui Liu. Robust graph neural networks via unbiased aggregation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 110097–110130. Curran Associates, Inc., 2024. doi: 10.52202/079017-3494. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/c6e31f86c1eb8dfc05190cf15ed52064-Paper-Conference.pdf.

Yeonjun In, Kanghoon Yoon, Kibum Kim, Kijung Shin, and Chanyoung Park. Self-guided robust graph structure refinement. In *Proceedings of the ACM Web Conference 2024*, WWW '24, pp. 697–708, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400701719. doi: 10.1145/3589334.3645522. URL https://doi.org/10.1145/3589334.3645522.

Yeonjun In, Kanghoon Yoon, Sukwon Yun, Kibum Kim, Sungchul Kim, and Chanyoung Park. Training robust graph neural networks by modeling noise dependencies. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 84185–84219. Curran Associates, Inc., 2025. doi: 10.52202/085713-2821. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/794e9fc8f6c6f22db8f8b7ece255822c-Paper-Conference.pdf.

Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pp. 66–74, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403049. URL https://doi.org/10.1145/3394486.3403049.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Woohyun Lee and Hogun Park. Self-supervised adversarial purification for graph neural networks. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 33715–33735. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/lee25af.html.

Jintang Li, Jie Liao, Ruofan Wu, Liang Chen, Zibin Zheng, Jiawang Dan, Changhua Meng, and Weiqiang Wang. Guard: Graph universal adversarial defense. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23, pp. 1198–1207, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701245. doi: 10.1145/3583780.3614903. URL https://doi.org/10.1145/3583780.3614903.

Kuan Li, YiWen Chen, Yang Liu, Jin Wang, Qing He, Minhao Cheng, and Xiang Ao. Boosting the adversarial robustness of graph neural networks: An OOD perspective. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=DCDT918ZkI.

Junlong Liao, Wenda Fu, Cong Wang, Zhongyu Wei, and Jiarong Xu. Value at adversarial risk: A graph defense strategy against cost-aware attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13763–13771, 2024. doi: 10.1609/aaai.v38i12.29282.

Ao Liu, Wenshan Li, Beibei Li, Wengang Ma, Tao Li, and Pan Zhou. Grimm: A plug-and-play perturbation rectifier for graph neural networks defending against poisoning attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 18754–18762, 2025. doi: 10.1609/aaai.v39i18.34064. URL https://doi.org/10.1609/aaai.v39i18.34064.

Jiayi Luo, Qingyun Sun, Haonan Yuan, Xingcheng Fu, and Jianxin Li. Robust graph learning against adversarial evasion attacks via prior-free diffusion-based structure purification. In *Proceedings of the ACM on Web Conference 2025*, WWW '25, pp. 2098–2110, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400712746. doi: 10.1145/3696410.3714815. URL https://doi.org/10.1145/3696410.3714815.

Yuefei Lyu, Chaozhuo Li, Sihong Xie, and Xi Zhang. Enhancing robustness of graph neural networks on social media with explainable inverse reinforcement learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 31736–31758. Curran Associates, Inc., 2024. doi: 10.52202/079017-0997. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/3838bf9070b80e888d571ec126d844c2-Paper-Conference.pdf.

Felix Mujkanovic, Simon Geisler, Stephan Günnemann, and Aleksandar Bojchevski. Are defenses for graph neural networks robust? In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 8954–8968. Curran Associates, Inc., 2022. doi: 10.52202/068431-0651. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/3ac904a31f9141444009777abef2ed8e-Paper-Conference.pdf.

Weixuan Shen, Xiaobo Shen, and Shirui Pan. Adversarial contrastive graph masked autoencoder against graph structure and feature dual attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 12515–12523, 2025. doi: 10.1609/aaai.v39i12.33364.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1633–1645. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/11f38f8ecd71867b42433548d1078e38-Paper.pdf.

Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. Adversarial examples for graph data: Deep insights into attack and defense. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 4816–4823. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/669. URL https://doi.org/10.24963/ijcai.2019/669.

Xiang Zhang and Marinka Zitnik. Gnnguard: Defending graph neural networks against adversarial attacks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9263–9275. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/690d83983a63aa1818423fd6edd3bfdb-Paper.pdf.

Kai Zhao, Qiyu Kang, Yang Song, Rui She, Sijie Wang, and Wee Peng Tay. Adversarial robustness in graph neural networks: A hamiltonian approach. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 3338–3361. Curran Associates, Inc., 2023. doi: 10.52202/075280-0148. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/0a443a000e1cb2281480b3bac395b3b8-Paper-Conference.pdf.

Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pp. 1399–1407, New York, NY, USA, 2019.

Association for Computing Machinery. ISBN 9781450362016. doi: 10.1145/3292500.3330851. URL https://doi.org/10.1145/3292500.3330851.

Shuman Zhuang, Zhihao Wu, Zhaoliang Chen, Hong-Ning Dai, and Ximeng Liu. Refine then classify: Robust graph neural networks with reliable neighborhood contrastive refinement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 13473–13482, 2025. doi: 10.1609/aaai.v39i12.33471. URL https://doi.org/10.1609/aaai.v39i12.33471.

Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bylnx209YX.

A Experimental Details

A.1 Data Splits and Training Configuration

Table A1 summarizes the three datasets used in the experiments, with each undirected edge counted once.

Table A1: Dataset statistics. Edge counts are undirected.

Dataset	Nodes	Edges	Features	Classes
Cora	2,708	5,278	1,433	7
Citeseer	3,327	4,552	3,703	6
PubMed	19,717	44,324	500	3

For each seed, we sample 20 labeled and 20 validation nodes per class. The class-balanced test set contains $\lceil 0.1N_c \rceil$ nodes from class c, where N_c is its node count; remaining nodes are unlabeled. Validation and test nodes are excluded from the purifier training graph. The GCN is trained on labeled nodes and then retrained using pseudo-labels for the unlabeled training nodes. Methods share the split associated with each seed, while retaining their respective training and defense procedures.

The GCN uses 64 hidden units, dropout 0.5, and Adam with learning rate 0.01 and weight decay 10^{-3}. The purification backbone uses seven propagation filters, 128 hidden units, dropout 0.7, and Adam with learning rate 0.01 and weight decay 10^{-4}. Purifier checkpoints are selected using validation link AUC and AP. The semantic MLP has hidden width 256, an ELU activation, and dropout 0.1. The direction-agreement term in $\mathcal{L}_{\mathrm{str}}$ has weight 0.2. Table A2 lists the memory-update and training-loss settings, which are fixed across datasets. The stopping tolerance applies to relative total edge-weight change, as defined in Section 4.4.

Table A2: Reported settings for the full soft-gated RGSM-GP model. Candidate and update bounds specify fractions of nodes; the resulting soft gate is constrained by its total mass.

Component	Parameters	Values
Feature conditioning	α_f	0.5
Reliability	(w_s, w_u, w_t); κ	$(1, 1, 0.5)$; 2
Numerical stabilizer	δ	10^{-12}
Pressure summary	q	0.20
Candidate screen	$(\rho_c^-, \rho_c^+, \lambda_c)$	$(0.10, 0.25, 0.50)$
Update budget	$(\rho_u^-, \rho_u^+, \lambda_b)$; τ_{safe}	$(0, 0.08, 0.25)$; 0.02
Soft gate	β	0.05
Neighbor context and update	γ; α_x	1; 0.2
Stopping	$T_{\max}$; $\varepsilon_{\mathrm{stop}}$	5; 10^{-4}
Final fusion	α_{cls}; $\lambda_{\max}$	1; 0.5
Semantic training	α_{sce}; $(\lambda_{\mathrm{sem}}, \lambda_{\mathrm{pres}})$	2; $(3, 0.1)$

For the compared defenses, AT-GCN uses an adversarial training rate of 0.1. RUNG-MCP uses its regularization parameters $\gamma = 36$ and $\widehat{\lambda} = 0.9$, with ten propagation steps. These are RUNG-specific parameters, distinct from the reliability exponent and fusion weights of RGSM-GP. GNNGuard uses similarity and pruning thresholds of 0.1 and 0.5; its adaptive check uses hard forward decisions with a BPDA backward approximation at temperature 20.

A.2 Attack Protocol and Evaluation Paths

Let $\mathcal{V}_{\mathrm{te}}$ denote the test nodes and M_{te} the number of undirected edges incident to at least one test node. At rate ϵ_{atk}, the attacker may flip $\mathrm{round}(\epsilon_{\mathrm{atk}} M_{\mathrm{te}})$ edges. Thus, these percentages are not fractions of all possible node pairs. PRBCD uses 500 optimization epochs, including 100 fine-tuning epochs; LRBCD uses 400 epochs. Both use 10,000 attack candidates. Attacked graphs are generated separately for each defense. Table A3 records the losses used for the main results.

Table A3: Attack losses for each compared method in the main suite.

Method	Cora–PRBCD	Other PRBCD settings	LRBCD
GPR-GAE	Margin	tanhMargin	Margin
AT-GCN	tanhMargin	tanhMargin	Margin
RUNG	tanhMargin	tanhMargin	Margin
GNNGuard	Margin	tanhMargin	Margin
RGSM-GP	tanhMargin	tanhMargin	Margin

For RGSM-GP, the main records use attack-mode `hard` on Cora and Citeseer, and `soft_gate` on PubMed. The implementation's `hard` attack-mode setting retains the configured defense gate; it does not itself switch the full model to hard gating. The full model is configured with a soft gate, whereas the hard-gate ablation explicitly replaces that gate. The BPDA-style check uses hard forward decisions with a soft-gate backward approximation during attack generation. The deployed defense configuration is restored before reporting attacked accuracy. These path checks are analyzed separately from the main suite. Transferred, same-graph attacks are restricted to controlled mechanism comparisons and are not pooled into the defense-specific main results.

B Additional Results and Diagnostics

B.1 Clean and Robust Accuracy

Figure B1 relates attacked performance to defended clean accuracy. Higher clean accuracy did not consistently coincide with higher robust average. On Cora–PRBCD, RGSM-GP had lower clean accuracy than all comparison methods but the highest robust average. On Citeseer, it had the highest mean for both metrics under both attacks. The full comparison thus shows dataset-dependent clean/robust performance, rather than a uniform improvement in every metric.

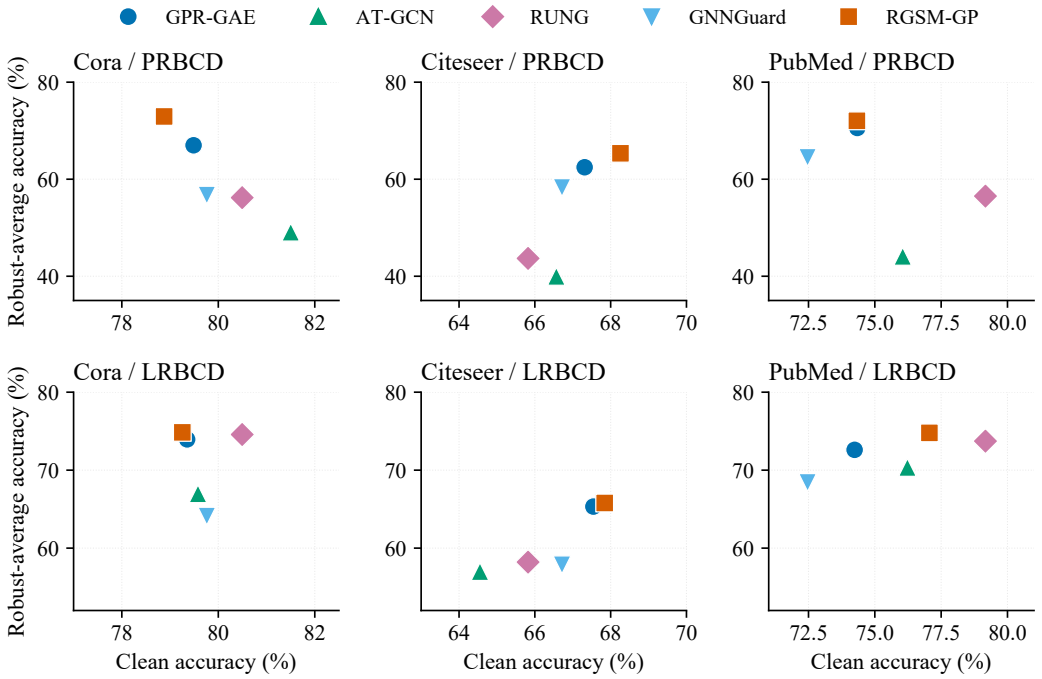

Figure B1: Clean and robust-average accuracy across all methods and settings. Each point is a method mean over the same complete runs used in Table 1. Clean accuracy is measured after defense at zero attack budget; robust average uses the three nonzero budgets. The table reports the corresponding sample standard deviations.

B.2 Ablation Definitions and Diagnostics

The reported controls use the paired 11-run PRBCD suite on Cora and Citeseer. Disabling temporal instability sets $w_t = 0$; the residual-only control also removes the uncertainty contribution. The fixed-gate control replaces pressure-adaptive candidate and budget selection with the implementation's fixed schedule. Its selection ratio follows $\max(\rho_{\min}, \rho_0 \rho_{\text{decay}}^t)$, where ρ_0 is the initial ratio, $\rho_{\min}$ its floor, and ρ_{decay} the per-round decay factor. Their settings are 0.10, 0.03, and 0.8, respectively, with a node-unreliability eligibility threshold of 0.02. Hard gating replaces the continuous weights $m_i^{(t)}$ with binary selection. Disabling feedback sets $\alpha_f = 0$ while retaining semantic updates and final fusion. These controls share the evaluation suite and purifier architecture; changes to selection can change realized update mass. Table 2 in the main text reports the paired accuracy differences and confidence intervals for these controls.

Figure B2 retains the budget-wise soft/hard-gate comparison from the same dedicated validation suite. It provides additional detail for the aggregate hard-gate control in the main text, without adding a separate benchmark method.

For diagnostics, touched nodes are the endpoints of edges in the symmetric difference between the clean and attacked graphs. The unreliability score $u_i^{(t)}$ is evaluated against these labels using ROC–

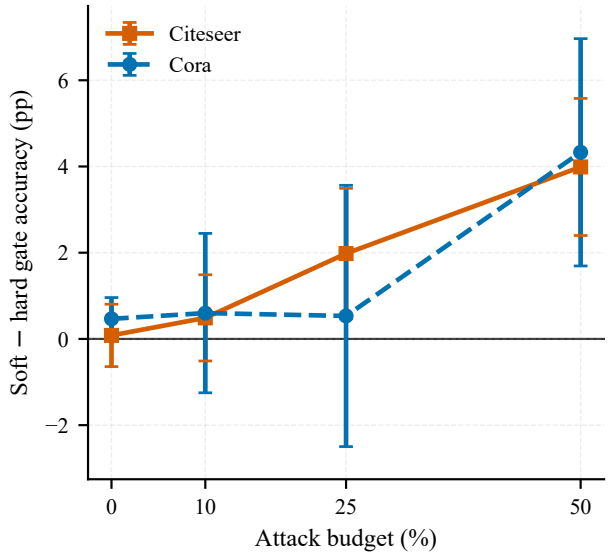

Figure B2: Soft-gate minus hard-gate defended accuracy across attack budgets on the dedicated Cora and Citeseer PRBCD suite. Points show paired mean differences and error bars show 95% paired t intervals.

AUC and related ranking measures; the labels are never provided to the defense. In the reported diagnostic cells, candidate coverage was 11.4–16.2% and hard-reference selection coverage was 0.5–3.1%. The latter is a binary reference set, not the support of all nonzero soft weights. Mean interpolation coefficients were 0.0003–0.0029. These quantities describe selection and update coefficients, not measured feature displacement or ground-truth semantic recovery.