AUDITING EXPLANATION PRESERVATION IN SPARSIFIED GNN PIPELINES

Anonymous authors
Paper under double-blind review

ABSTRACT

We study explanation auditability of graph sparsifiers for graph neural networks (GNNs). Existing sparsifiers preserve structural properties or predictive accuracy, but in deployment workflows where predictions are inspected or audited, do the explanations produced on the sparsified graph remain consistent with the full-graph reference explanations? We introduce an audit framework built around a node-level outcome that permits controlled explanation change and identifies four failure modes: prediction flip, excessive explanation drift, top-k structural rearrangement, and top-k mass erosion. The framework evaluates preservation at marginal, conditional, and worst-case levels and provides exact finite-sample confidence bounds for the complete test-node population and its structural subgroups. We audit thirteen sparsifiers, four post-hoc explainers, and five node-classification benchmarks spanning homophilic and heterophilic graphs. Experiments spanning 280 main audit runs indicate that (i) prediction-preserving sparsification does not imply explanation-preserving sparsification, (ii) distinct sparsifier families exhibit signature failure modes, (iii) preservation rates vary systematically with local graph structure, with high-degree nodes substantially less likely to satisfy the audit outcome, and (iv) overall preservation rates do not establish that the same preservation target holds across all structural groups. We argue that explanation preservation is a measurable property of deployed GNN pipelines, distinct from accuracy, and one that should be reported alongside it.

1 INTRODUCTION

Graph sparsification—pruning edges to reduce memory, latency, and message-passing cost—is a common way to scale graph neural networks (GNNs) (Spielman & Srivastava, 2008). The resulting sparsified GNNs can then serve as computationally cheaper replacements for their full-graph counterparts. Most sparsifiers are designed and evaluated according to structural or predictive criteria, such as preserving Laplacian quadratic forms through spectral sparsification, or preserving node-classification accuracy through supervised sparsification (Zheng et al., 2020; Das et al., 2025). This raises a question in explanation-facing deployments: *does the sparsified pipeline also preserve the explanation behavior of the full-graph pipeline it replaces?*

This is especially important when a sparsified GNN replaces a full-graph system whose explanations have already been documented, reviewed, or communicated to downstream users. For instance, U.S. Regulation B requires adverse-action notices to state specific principal reasons that reflect the factors actually considered by a creditor (Consumer Financial Protection Bureau, 2022). Consider a lender that uses a full-graph GNN on a transaction graph and relies on its explanation to support such a notice. A sparsified replacement may produce a different explanation when the decision is later reviewed, even if the predicted credit decision is unchanged. In such a deployment, the full-graph explanation provides the behavioral reference for evaluating the replacement. Indeed, this reference explanation may not represent ground truth, so deviation from it does not by itself imply that the sparsified pipeline is worse. Nevertheless, in this application, it is important to verify that the change remains within prespecified deployment tolerances. If it does not, the replacement no longer reproduces the established explanation behavior required by that deployment. To quantify this change, we ask: *for what fraction of test nodes does a sparsified pipeline preserve the reference prediction and explanation within prespecified tolerances, and, when auditing every test node is*

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.

impractical, what claim about the complete test-node population can be supported by sampling audit nodes uniformly without replacement?

Our Contributions. (I) An audit framework. We respond to this question by designing an audit framework comprising three components: **First,** a node-level audit outcome that records whether the deployed pipeline preserves the reference prediction and explanation at a query node, decomposed into four named failure modes. **Second,** an auditability hierarchy that evaluates preservation at three levels of stringency — marginal (preservation on average), conditional (preservation within structural strata), and worst-case (preservation across all strata). **Third,** exact finite-sample confidence bounds for determining whether preservation observed on audited nodes supports claims about the complete test-node population and its structural subgroups. Our framework is post-hoc, which enables an existing GNN–sparsifier–explainer combination to be audited as-is. The prespecified tolerances permit controlled explanation change rather than requiring exact equality. Furthermore, the resulting preservation assessment complements structural and predictive measures of sparsifier quality. **(II) Empirical evaluation and findings.** We conduct the first systematic empirical audit, to our knowledge, of explanation behavior across thirteen graph sparsifiers, four post-hoc explainers, and five node-classification datasets, plus a no-sparsification baseline. Empirical results show that (i) prediction preservation does not imply explanation preservation (mean prediction preservation 0.91 vs. mean preservation rate 0.48); (ii) structural and learned/proxy sparsifiers exhibit distinct failure patterns; (iii) preservation varies systematically with local graph structure, with high-degree nodes less likely to satisfy the audit outcome (preservation rates differing by 0.32 on average between the lowest- and highest-degree quartiles); and (iv) an overall preservation rate does not by itself establish that the same preservation target holds across all designated structural groups.

2 Related Work

Graph sparsification. Classical sparsifiers approximate structural or spectral properties of the input graph: t-spanners preserve pairwise shortest-path distances up to a multiplicative factor, and spectral sparsifiers (Spielman & Srivastava, 2008; Spielman & Teng, 2011; Liu et al., 2023) preserve the Laplacian quadratic form. A second line of work learns sparsifiers jointly with a downstream task: NeuralSparse (Zheng et al., 2020) learns a stochastic edge selector to maximize node-classification accuracy, SparseGAT (Ye & Ji, 2021) sparsifies attention coefficients during training, and SGS-GNN (Das et al., 2025) introduces supervised and homophily-driven selection criteria. A third, recent line uses explanation signals as the sparsification objective (Akkas & Azad, 2026). Across all three families, evaluation is dominated by structural metrics or downstream accuracy; the effect of sparsification on the explanations the deployed model produces is, to our knowledge, not measured.

Post-hoc GNN explanations and their evaluation. Post-hoc explainers attribute a node's prediction to subgraphs or edges of the input. GNNExplainer (Ying et al., 2019) optimizes a per-instance edge mask; PGExplainer (Luo et al., 2020) amortizes this optimization through a parameterized model; SubgraphX (Yuan et al., 2021) performs Monte-Carlo subgraph search; GraphSVX (Duval & Malliaros, 2021) uses Shapley values. Benchmarks such as BAGEL, GNNX-BENCH, and the robust-fidelity study (Rathee et al., 2022; Kosan et al., 2024; Zheng et al., 2024) compare explainers head-to-head on fixed graphs; we are not aware of any work that measures how an explainer's output changes when the input graph is sparsified.

Certification and conformal prediction on graphs. Recent work such as XGNNCert (Li et al., 2025) certifies whether a clean-graph explanation remains stable under bounded adversarial graph perturbations. We address a different deployment question: after efficiency-driven sparsification, what fraction of a fixed test-node population preserves both prediction and explanation behavior, overall and within prespecified structural groups? Conformal GNN methods provide coverage guarantees for prediction sets in node classification or link prediction (Akansha, 2024; Song et al., 2024; Zhao et al., 2024), but do not evaluate explanation preservation after sparsification. Our framework complements these guarantees by assessing joint prediction-and-explanation preservation at marginal, conditional, and worst-case levels, supported by exact finite-population lower confidence bounds.

3 Preliminaries

Let $G \triangleq (V, E, X)$ denote an attributed graph, where V is a set of n nodes, E is the set of edges, and $X \in \mathbb{R}^{n \times d}$ is the node feature matrix. Given C classes, a trained graph neural network (GNN)

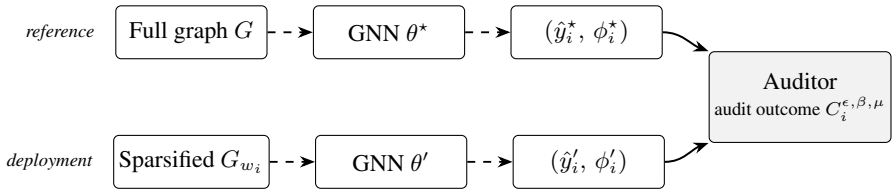


Figure 1: The audit compares a reference $(\hat{y}_i^\star, \phi_i^\star)$ against a deployment side $(\hat{y}_i', \phi_i')$. The explainer runs with $\theta^\star$ on the reference side and with θ' on the deployment side. In the canonical setting $\theta^\star = \theta'$ and only the graph input differs, so the recorded change is attributable to sparsification.

takes the graph together with an optional edge-weight vector $w \in [0,1]^{|E|}$ and outputs node-level log-probabilities, $f_\theta(X, E, w) \in \mathbb{R}^{n \times C}$. The unperturbed graph is retrieved by taking $w = \mathbf{1}$. For a query node $i \in V$, the prediction of the original GNN is $\hat{y}_i(G, \theta) = \arg\max_c f_\theta(X, E, \mathbf{1})_{ic}$.

Sparsified graphs. A graph sparsifier Sp is an algorithm that takes the graph G and a sparsification parameter ρ controlling the retention budget as input, and returns an edge-weight vector $w \in [0,1]^{|E|}$, inducing a sparsified graph $G_w \triangleq (V, E, X, w)$ with edge set $E_w \triangleq \{e \in E : w_e > 0\}$. The interpretation of ρ depends on the sparsifier family: it may denote a target *keep ratio* $|E_w|/|E|$, a per-node degree budget, a spectral approximation tolerance, or a learned threshold on edge scores. *Structural sparsifiers* such as random, degree-based, t-spanner construct w from structural properties, while *learned/proxy sparsifiers* (e.g., NeuralSparse, SparseGAT, and Shapley-style proxy rankers) obtain edge scores from learned models, proxy GNNs, or explanation-driven objectives.

Factual explanations. An edge explainer assigns an importance score $\phi(i; G, \theta) \in \mathbb{R}_{\geq 0}^{|E|}$ to every edge in E for a given query node, where the e-th coordinate quantifies the role of edge e in the explanation of the prediction at node i. Such an explainer yields a *reference explanation* $\phi_i^\star \triangleq \phi(i; G, \theta)$ for node i based on the full input graph G. As different explainers may yield different explanation subgraphs, our framework does not treat it as ground truth rather it serves as the reference for measuring explanation changes after sparsification. Any explainer-level learned state is calibrated once on the original graph and held fixed when explanations are recomputed on sparsified graphs. Explainers without learned global state, such as per-instance mask optimizers or perturbation-based methods, are rerun with fixed hyperparameters and deterministic seeds. After deploying sparsifier Sp, the post-sparsification explanation is $\phi_i' \triangleq \phi(i; G_{w_i}, \theta)$. This fixed-explainer setting prevents the audit from measuring changes in the explainer itself rather than changes induced by sparsification.

Explanation distance. All explanation vectors are compared in the original edge-coordinate system; pruned edges receive zero explanation mass. The normalized ℓ_1 explanation distance is

$$D_T(i; \phi_i^\star, \phi_i') \triangleq \frac{\|\phi_i' - \phi_i^\star\|_1}{\|\phi_i^\star\|_1}.$$

We also use top-k structural overlap and top-k mass retention. For a query node i with budget k_i, let $\mathrm{Top}_{k_i}(\phi)$ denote the set of top-k_i scored edges in vector ϕ. The top-k_i Jaccard overlap and top-k_i mass retention are defined as

$$J_{k_i}(i) \triangleq \frac{|\mathrm{Top}_{k_i}(\phi_i^\star) \cap \mathrm{Top}_{k_i}(\phi_i')|}{|\mathrm{Top}_{k_i}(\phi_i^\star) \cup \mathrm{Top}_{k_i}(\phi_i')|}, \qquad M_{k_i}(i) \triangleq \frac{\sum_{e \in \mathrm{Top}_{k_i}(\phi_i^\star)} \phi_i'(e)}{\sum_{e \in \mathrm{Top}_{k_i}(\phi_i^\star)} \phi_i^\star(e)}.$$

Lower values of D_T and higher values of J_{k_i} and M_{k_i} indicate stronger preservation of the reference explanation. D_T is a stability-style measure of explanation change under input perturbation (Agarwal et al., 2023); J_{k_i} is a top-k structural overlap of the form used in GNN-explanation benchmarks (Amara et al., 2022; Yuan et al., 2022); and M_{k_i} is a top-k mass-retention measure in the spirit of the fidelity family for GNN explanations (Pope et al., 2019; Zheng et al., 2024).

4 METHODOLOGY

Consider the following deployment scenario: a practitioner has a trained GNN $f_{\theta^\star}$ and a reference explainer $\phi(\cdot; \cdot, \theta^\star)$, both evaluated on the full graph G. For efficiency or interpretability, a sparsifier

Sp is applied at inference time, and the deployed pipeline produces predictions $\hat{y}'_i$ and post-hoc explanations ϕ'_i on the sparsified graph. The auditor asks whether this deployed pipeline preserves the reference prediction and explanation within prespecified tolerances.

Auditing Graph sparsifiers. Figure 1 summarizes the audit's information flow. The auditor treats the deployed pipeline as a black box: they observe its predictions $\hat{y}'_i$ and explanations ϕ'_i, but cannot inspect the deployed model's parameters. The auditor constructs the reference side $(\hat{y}^\star_i, \phi^\star_i)$ by running the reference model $\theta^\star$ and reference explainer ϕ on the full graph G. The explainer uses $\theta^\star$ on the reference side, θ' on the deployment side, and the corresponding full and sparsified graphs. In the *canonical setting* the two models coincide ($\theta^\star = \theta'$) and only the graph input changes, so the recorded change is attributable to sparsification alone. If the two models differ, the audit measures their combined effect. The canonical setting is the natural one for model-agnostic sparsifiers — those whose objective (connectivity, spectrum, distance) is a property of the graph itself and does not depend on the downstream model. *Structural sparsifiers* fit this setting. In contrast, *learned/proxy sparsifiers* obtain edge scores through method-specific learned models, proxy GNNs, or explanation-driven objectives. Auditing them as frozen preprocessors on top of an independently trained full-graph GNN puts the sparsifier in an *off-design* regime — a regime for which its scores were not originally calibrated. Our audit framework only requires the sparsified graph, or equivalently its edge-weight vector, irrespective of how that sparsified graph is obtained. Appendix D examines the impact of regime choice on audit outcomes.

4.1 Failure modes

We identify four ways a sparsified GNN pipeline can fail to preserve the reference explanation at a query node i.

(I) *Prediction flip* occurs when the post-sparsification prediction disagrees with the full-graph prediction, i.e., $\hat{y}'_i \neq \hat{y}^\star_i$; this is the failure mode existing sparsification literature already monitors.

(II) *Drift exceedance* occurs when $D_T(i)$ exceeds a tolerance ϵ, indicating that the post-sparsification explanation has shifted globally in edge-coordinate space.

(III) *Top-k_i structural rearrangement* occurs when $J_{k_i}(i) < \beta$, indicating that the most important edges identified by the explainer have been substantially reordered.

(IV) *Top-k_i mass erosion* occurs when $M_{k_i}(i) < \mu$, indicating that even when important edges remain in the top-k_i set, their explanation weight has been depleted.

These four failure modes are not redundant. An edge that drops out of $\mathrm{Top}_{k_i}(\phi'_i)$ is a J-failure even if most of its explanation mass is retained elsewhere. Conversely, edges that remain in the top-k_i set but lose explanation weight constitute an M-failure even under perfect Jaccard overlap. Both modes are audit-relevant because they correspond to different downstream consequences: J-failures change which edges the deployed system reports as important, while M-failures change the relative emphasis among the reported edges.

4.2 Node-level audit outcome

Let $E_i^{(h)}$ denote the set of edges in the h-hop neighborhood of node i, where h matches the receptive field of the GNN (e.g., $h = L$ for an L-layer message-passing model so that the audit is restricted to edges that can causally influence $\hat{y}_i$). The retention budget ρ sets $k_i = \lfloor \rho |E_i^{(h)}| \rfloor$, the number of highest-scored local edges used to compute top-k overlap and mass retention. β specifies the minimum acceptable top-k Jaccard overlap, and μ specifies the minimum acceptable fraction of reference top-k attribution mass retained after sparsification. The tolerances (ϵ, β, μ) are prespecified deployment acceptance criteria set by the auditor; they are not learnable model parameters.

Definition 1 (Node-level audit outcome)**.** *For tolerances (ϵ, β, μ), define the audit-outcome indicator*

$$C_i^{\epsilon,\beta,\mu} \triangleq \mathbf{1}[\, \hat{y}'_i = \hat{y}^\star_i \;\wedge\; D_T(i) \leq \epsilon \;\wedge\; J_{k_i}(i) \geq \beta \;\wedge\; M_{k_i}(i) \geq \mu \,]\,. \tag{1}$$

Node i passes the node-level audit under the prespecified tolerances (ϵ, β, μ) when $C_i^{\epsilon,\beta,\mu} = 1$.

4.3 THE AUDITABILITY HIERARCHY

A node-level audit outcome describes preservation for one audited node. We define auditability over a set of nodes at three levels: marginal, conditional, and worst-case.

Marginal auditability. Let $\mathcal{I}_{\text{audit}} \subset V_{\text{test}}$ be the audit set. Define

$$\text{PresRate}^{\epsilon,\beta,\mu} \triangleq \frac{1}{|\mathcal{I}_{\text{audit}}|} \sum_{i \in \mathcal{I}_{\text{audit}}} C_i^{\epsilon,\beta,\mu}.$$

The pipeline is $(\epsilon, \beta, \mu, \delta)$-marginally auditable on $\mathcal{I}_{\text{audit}}$ if $\text{PresRate}^{\epsilon,\beta,\mu} \geq 1 - \delta$. This condition requires at least a fraction $1 - \delta$ of all audited nodes to pass the node-level audit.

Conditional auditability. A marginally auditable pipeline may still have low preservation for a structurally defined group of nodes. Let Z_i assign each test node to a prespecified structural group, or *stratum*, based on a graph property, such as degree or local homophily, and let $\mathcal{Z}$ denote the resulting strata. For a stratum $z \in \mathcal{Z}$ with $|\mathcal{I}_{\text{audit}} \cap z| > 0$, define

$$\text{PresRate}_z^{\epsilon,\beta,\mu} \triangleq \frac{1}{|\mathcal{I}_{\text{audit}} \cap z|} \sum_{i \in \mathcal{I}_{\text{audit}} \cap z} C_i^{\epsilon,\beta,\mu}.$$

The pipeline is $(\epsilon, \beta, \mu, \delta)$-conditionally auditable on stratum z if $\text{PresRate}_z^{\epsilon,\beta,\mu} \geq 1 - \delta$.

Worst-case auditability. When every prespecified stratum is represented in $\mathcal{I}_{\text{audit}}$, define the worst-stratum preservation rate as

$$\text{PresRate}_{\min}^{\epsilon,\beta,\mu} \triangleq \min_{z \in \mathcal{Z}} \text{PresRate}_z^{\epsilon,\beta,\mu}.$$

The pipeline is $(\epsilon, \beta, \mu, \delta)$-worst-case auditable with respect to $\mathcal{Z}$ if $\text{PresRate}_{\min}^{\epsilon,\beta,\mu} \geq 1 - \delta$, or equivalently, if it is conditionally auditable on every stratum in $\mathcal{Z}$.

These conditions determine auditability on the sampled audit nodes. A claim about the complete test-node population requires the corresponding lower confidence bound (discussed next) to meet the same target $1 - \delta$.

4.4 EXACT CONFIDENCE BOUNDS FOR THE TEST-NODE POPULATION

Given a graph and test split, the reference and deployed pipelines and the tolerances remain unchanged while audit nodes are sampled. Each test node therefore has a well-defined audit outcome $C_i^{\epsilon,\beta,\mu}$. Let

$$N = |V_{\text{test}}|, \qquad K = \sum_{i \in V_{\text{test}}} C_i^{\epsilon,\beta,\mu}, \qquad p_N = K/N.$$

Here, K is the unknown number of test nodes that pass the audit, and p_N is the corresponding true preservation rate over the test-node population. Suppose n nodes are sampled uniformly without replacement. Let S denote the number that pass the audit, with observed value s. Then $S \sim \text{Hypergeom}(N, K, n)$.

For a confidence level $1-\alpha$, we obtain an exact one-sided lower confidence bound for p_N by inverting the hypergeometric upper tail:

$$\underline{p}_\alpha(s; N, n) = \frac{1}{N} \min\left\{ k \in \{0, \ldots, N\} : \Pr_{X \sim \text{Hypergeom}(N,k,n)}(X \geq s) \geq \alpha \right\}. \tag{2}$$

Appendix A.2 proves that, with probability at least $1 - \alpha$ over the sampled audit nodes, this lower confidence bound does not exceed the true test-population preservation rate p_N.

Definition 2 (Marginal finite-population guarantee)**.** *A pipeline supports a marginal* $(\epsilon, \beta, \mu, \delta, \alpha)$ *finite-population guarantee when*

$$\underline{p}_\alpha(s; N, n) \geq 1 - \delta. \tag{3}$$

This condition supports the claim that at least a fraction $1 - \delta$ of the complete test-node population passes the audit, at confidence $1 - \alpha$.

Conditional and worst-case guarantees. The structural strata $\mathcal{Z}$ are defined over the complete test-node population before audit nodes are sampled and outcomes are examined. For each stratum z, let N_z be its number of test nodes, n_z its number of audited nodes, and s_z the number of audited nodes that pass. A conditional finite-population guarantee on a prespecified stratum is supported when

$$\underline{p}_{\alpha}(s_z; N_z, n_z) \geq 1 - \delta.$$

When claims are made simultaneously for m strata, each lower confidence bound is computed as $\underline{p}_{\alpha/m}(s_z; N_z, n_z)$. This Bonferroni adjustment ensures that all m bounds hold simultaneously with confidence at least $1-\alpha$, without requiring independence across strata. A worst-case finite-population guarantee is supported only when every prespecified stratum is audited and every adjusted lower confidence bound meets $1 - \delta$.

4.5 The audit procedure

Algorithm 1, Appendix A.3 presents the complete audit procedure. Given fixed reference and deployed pipelines, prespecified tolerances and structural strata, and an audit set sampled uniformly without replacement from the fixed test pool, it returns the per-node outcomes, empirical preservation rates, exact lower confidence bounds, and the supported finite-population guarantees. In the canonical setting used by the main experiments, $\theta' = \theta^\star$. The audit also applies when the deployed GNN differs from the reference GNN. In that case, it measures the combined graph-and-model change. For the complete test-node population, a marginal, stratum-specific conditional, or worst-case finite-population guarantee is supported when the corresponding lower confidence bound meets the target $1 - \delta$ at confidence $1 - \alpha$.

5 Experimental evaluation

Our evaluation is organized around five research questions. **RQ1:** Does prediction preservation imply explanation preservation? **RQ2:** Do sparsifier families exhibit signature failure modes: prediction flip, drift exceedance, structural rearrangement (J), mass erosion (M)? **RQ3:** Is preservation associated with local graph structure? **RQ4:** How large must the audit set be for usable bounds? **RQ5:** How does preservation vary across structural groups, and can a pipeline support the same preservation target across all designated groups?

Experimental Setup. We audit five node-classification datasets — Cora, CiteSeer, PubMed (homophilic citation), and Squirrel, Chameleon (heterophilic Wikipedia) — with a 2-layer GCN backbone (Appendix Table 16; ogbn-arxiv pilot in Appendix B). The audited sparsifiers are random, DSpar, degdist, spectral, t-spanner, GraphPRI (*structural*); GraphUltra, NeuralSparse, SparseGAT, SGS-GNN, SGCN, MoG, Shapley (Akkas & Azad, 2026) (*learned/proxy*); plus an identity (no-sparsification) control. Learned/proxy sparsifiers are deployed off-design (Section 4). Appendix D compares this setup with a jointly trained NeuralSparse–GCN pipeline that uses the same co-trained GCN on the full and sparsified graphs. Four post-hoc explainers are considered: GNNExplainer, PGExplainer, SubgraphX, and GraphSVX, for which neighboring-node Shapley values are projected to edge-coordinate scores.

Audit setting. We conduct $5 \times 13 \times 4 = 260$ audit runs plus $5 \times 1 \times 4 = 20$ identity-control runs, for 280 total runs at keep ratio $\rho = 0.5$. To quantify seed sensitivity, we re-ran a representative subset comprising three sparsifiers (random, DSpar, and SparseGAT), two explainers (GNNExplainer and PGExplainer), and two datasets (Cora and Chameleon), using three backbone seeds and three audit-node sampling seeds for each combination (see Appendix H). We use audit size $n = 100$ for SubgraphX, GraphSVX, and the more expensive learned/proxy sparsifiers (GraphPRI, GraphUltra, MoG, SGCN, plus NeuralSparse and Shapley on Cora/CiteSeer/PubMed); $n = 200$ is used for all other configurations. The keep ratio $\rho = 0.5$ determines the sparsification and top-k budgets. Independently, the prespecified tolerances $(\epsilon, \beta, \mu) = (0.5, 0.5, 0.5)$ define acceptable global drift, top-k overlap, and retained attribution mass. Appendix G evaluates all $5^3 = 125$ tolerance combinations across the 260 non-identity pipelines. For each configuration, we use hypergeometric inversion to compute an exact one-sided 95% lower confidence bound ($\alpha = 0.05$) on the preservation rate over its complete test-node population.

Table 1: Aggregate audit results. The first row reports all 280 runs; the second reports the 260 non-identity runs. Prediction preservation counts nodes whose predicted class is unchanged. The preservation rate additionally requires all three explanation-preservation conditions. The final column averages the configuration-specific lower confidence bounds.

Subset	Runs	Mean pred. preservation	Mean preservation rate	Mean lower bound
All runs	280	0.911	0.478	0.421
Non-identity sparsifiers	260	0.904	0.438	0.378

Table 2: Dataset-level mean $\pm$ std. for prediction preservation and the preservation rate (260 runs excluding identity). Heterophilic graphs have substantially higher preservation rates than homophilic graphs.

	Homophilic			*Heterophilic*	
Metric	Cora	CiteSeer	PubMed	Squirrel	Chameleon
Pred. preservation	0.88 ± 0.04	0.92 ± 0.03	0.93 ± 0.04	0.87 ± 0.05	0.92 ± 0.05
Preservation rate	0.21 ± 0.10	0.34 ± 0.12	0.23 ± 0.15	0.65 ± 0.11	0.75 ± 0.11

Table 3: Within-dataset Spearman ρ between prediction preservation and the preservation rate. Structural sparsifiers exhibit moderate and consistent correlations across datasets.

Family	Cora	CiteSeer	PubMed	Squirrel	Chameleon
Structural	0.55	0.52	0.68	0.77	0.76
Learned/proxy	0.20	0.60	0.56	0.49	0.90

5.1 Prediction preservation does not imply explanation preservation

Aggregate preservation gap. Prediction preservation is high across the audited pipelines but the preservation rate is much lower: across all 280 runs, the mean prediction-preservation rate is 0.911 and the mean preservation rate is 0.478 (Table 1). Sparsified pipelines therefore preserve the predicted label for most audited nodes while fewer than half satisfy all four preservation conditions. Table 2 shows substantially higher preservation rates on the heterophilic graphs (e.g., Chameleon 0.750) than on the homophilic graphs (e.g., Cora 0.215). The prediction–explanation gap also remains positive for every dataset–explainer group after restricting the analysis to references that pass both model-focused sufficiency and necessity tests, and in every matched configuration under both GCN and GraphSAGE (Appendices E and F).

Prediction preservation is an inconsistent proxy. Table 3 reports within-dataset Spearman correlations between prediction preservation and the preservation rate, split by family. Structural sparsifiers show moderate correlations across all five datasets (range 0.52–0.77). Learned/proxy sparsifiers are less consistent: Cora shows the weakest association ($\rho = 0.20$), PubMed remains moderate but lower than the structural family (0.56 vs. 0.68), and CiteSeer is comparable to the structural family (0.60 vs. 0.52). On Chameleon, both families show high correlations (structural 0.76, learned/proxy 0.90) in an already-high-preservation regime. The family split exposes heterogeneity: on Cora, learned/proxy sparsifiers preserve predictions but have low preservation rates. Appendix C.1 visualizes per-dataset scatter plots.

Same-budget variation. Beyond prediction preservation, we found that sparsification budget does not determine explanation preservation. At the same budget $\rho = 0.5$ on Cora, the preservation rates for Random and SGS-GNN are 0.075 and 0.430 with GNNExplainer, and 0.085 and 0.405 with PGExplainer. These differences show that preservation depends on which edges are removed or reweighted, not only on how many. It also suggests that reference-explanation preservation is complementary to structural and predictive measures of sparsifier quality.

Table 4: Failure-mode shares by sparsifier, averaged over datasets and explainers. Superscripts identify the three largest shares in each row.

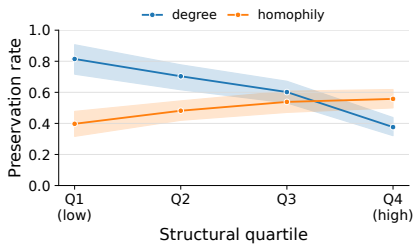
	Structural						Learned/proxy						
Mode	Rand.	Degdist	DSpar	Spect.	t-span.	GraphPRI	NeuralSp.	SparseG.	SGSGNN	SGCN	GraphUlt.	MoG	Shapley
Flip	0.18	0.15	0.14	0.16	0.23	0.16	0.24	**0.27**^{1}	0.23	0.23	**0.24**^{2}	0.22	**0.24**^{3}
Drift	0.72	0.74	**0.75**^{2}	**0.74**^{3}	0.68	**0.77**^{1}	0.70	0.69	0.68	0.73	0.68	0.70	0.64
J	0.09	**0.10**^{3}	**0.10**^{2}	0.10	0.08	0.07	0.06	0.04	0.09	0.04	0.08	0.08	**0.12**^{1}
M	0.00	0.00	0.00	0.00	**<0.01**^{1}	0.00	0.00	**<0.01**^{2}	0.00	0.00	0.00	0.00	0.00

5.2 Failure-mode decomposition

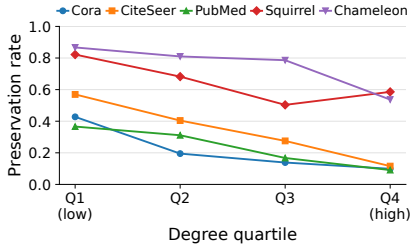
By Definition 1, a node that does not satisfy the audit outcome can violate multiple conditions. For failure mode analysis, we assign one dominant failure label using the priority order prediction flip, drift, top-k structural rearrangement, and top-k mass erosion; we retain all four failure indicators for each node. Among preservation failures, we observe drift to be the dominant failure mode. Averaged over (sparsifier, explainer, dataset) configurations (excluding *Identity*), drift accounts for 71.0% of dominant failures, prediction flips for 20.8%, top-k structural rearrangement for 8.2%, and top-k mass erosion for less than 0.1%. Most preservation failures therefore reflect global movement of the post-sparsification explanation in edge-coordinate space rather than a change in the predicted label. The secondary failure modes follow a family-level pattern (Table 4). Structural degree- and resistance-based sparsifiers concentrate residual failure mass on top-k structural rearrangement, while the learned/proxy sparsifiers concentrate relatively more failure mass on prediction flips. Mass erosion is rare under the default thresholds. The NeuralSparse co-training companion in Appendix D suggests that, at least for this sparsifier, using the same jointly trained GCN as the reference and deployment model does not consistently reduce prediction-flip failures: the flip share decreases on Cora ($0.170 \rightarrow 0.144$) but increases on Chameleon ($0.455 \rightarrow 0.513$). Explanation-preservation failures therefore usually involve top-k membership changes or global drift rather than mass erosion.

5.3 Preservation and local graph structure

Preservation rates vary systematically with local graph structure. Figure 2(left) stratifies audit nodes by degree and local homophily. For each (dataset, sparsifier, explainer) configuration, nodes are split into four equal-sized quartiles by the indicated feature, and the preservation rate is computed within each quartile. We observe that low-degree nodes are systematically more likely to satisfy the audit outcome, while local homophily is non-monotone with quartile means ranging from 0.369 to 0.660. The local homophily of a node i is the fraction of its neighbors whose ground-truth class label equals y_i Loveland et al. (2024). These structural associations justify the auditability hierarchy introduced in Section 4.3: a marginal preservation rate that averages over the audit set does not capture systematic differences across degree regimes. Conditional and worst-case audit levels make these disparities explicit.



(a) Preservation by structural quartile.



(b) Preservation across degree quartiles by dataset.

Figure 2: Preservation rates stratified by local structure. Left: audit nodes are split into quartiles by degree and local homophily. Right: per-dataset degree trend excluding the Identity sparsifier. Degree shows a strong low-to-high quartile decline, while homophily shows a weaker non-monotone pattern.

5.4 AUDIT-SET SIZE

We estimate audit-size effects by repeatedly subsampling nodes without replacement from each original audit and computing the exact hypergeometric lower confidence bound using the complete test-pool size. Because the original audits contain either 100 or 200 nodes, we analyze two fixed sets of configurations rather than combine results whose configuration mix changes with n. Across all 280 configurations, evaluated at $n \in \{25, 50, 75, 100\}$, the median lower confidence bound increases from 0.272 at $n = 25$ to 0.360 at $n = 100$. Among the 88 configurations originally audited at 200 nodes, it increases from 0.362 at $n = 25$ to 0.458 at $n = 100$ and 0.486 at $n = 200$ (Appendix Figure 5). At the largest audit size supported by each set, median lower confidence bounds are lower for homophilic than heterophilic datasets: 0.157–0.312 versus 0.533–0.690 at $n = 100$ across all configurations, and 0.234–0.378 versus 0.675–0.783 at $n = 200$ in the fixed 200-node set. The interquartile ranges remain wide, showing that the lower confidence bound achieved at a given audit size also depends on the particular sparsifier–explainer configuration.

A practitioner deploying in a low-preservation regime should not interpret the diminishing gains observed between $n = 100$ and $n = 200$ as a universal sample-size recommendation. Pipelines with weak lower confidence bounds may require larger audit budgets to reduce sampling uncertainty and raise the bound toward the empirical preservation rate. However, if the observed preservation rate itself is low, additional sampling alone cannot support a substantially higher target; the auditor must instead change the sparsification setting or adopt a less demanding target $1 - \delta$.

5.5 STRUCTURAL DISPARITIES AND ALL-STRATA AUDITABILITY

We evaluate how preservation varies across structurally distinct node groups and whether a pipeline supports the same preservation target simultaneously across all designated groups. To that end, we examine degree-based differences descriptively and then demonstrate the formal all-strata audit on an illustrative structural partition.

Degree-stratified preservation. We stratify audit nodes by degree quartile, as in Section 5.3, and average preservation rates over non-identity sparsifier–explainer configurations. Figure 2 (right) and Appendix Table 7 show a mean Q1–Q4 gap of 0.325. Four datasets exhibit decreasing preservation with degree; Squirrel is non-monotonic but retains a gap of 0.236. On CiteSeer, for example, the marginal rate is 0.341, compared with 0.570 for Q1 and 0.116 for Q4. Thus, the overall preservation rate does not capture the substantial differences across degree quartiles.

Illustrative all-strata audit. To demonstrate an auditor-defined subgroup claim, we evaluate a 4×4 degree-by-local-homophily partition motivated by the structural disparities above. We prespecify $1 - \delta = 0.50$ and simultaneous 95% confidence. Across the 260 non-identity pipelines, the mean, median, and maximum worst-group lower bounds are 0.077, 0.010, and 0.467, respectively. No pipeline therefore establishes the illustrative claim that at least half of the nodes satisfy the audit outcome in every group (Appendix Figure 6). For the three pipelines closest to the target, a complete-stratum follow-up study improved the final worst-group lower bounds to 0.529–0.548, showing that additional auditing resolves the initial sampling uncertainty (Appendix Table 6).

6 CONCLUSION

We introduced a model- and sparsifier-agnostic, post-hoc audit framework for evaluating whether graph sparsification preserves node-level GNN explanations. The framework separates prediction preservation from explanation preservation through a node-level audit outcome covering the reference prediction, global explanation, top-k support, and reference explanation mass. These outcomes are evaluated at marginal, conditional, and worst-case audit levels. Exact finite-sample confidence bounds determine whether the observed preservation rates support the corresponding claims for the complete test-node population and its structural groups.

Limitations. The exact finite-population guarantee applies to audit nodes sampled uniformly without replacement from a fixed test pool, with the graph, trained pipelines, tolerances, and stochastic seeds held fixed. It does not by itself support claims after retraining or on different graphs or unseen-node populations. The audit measures preservation relative to the designated reference explanation; it does not establish explanation faithfulness in an absolute sense.

AI USE STATEMENT

Generative AI tools were used to provide feedback on the manuscript, including the presentation and reader friendliness, and seeking revisions to certain phrases/sentences/paragraphs. The authors reviewed all AI-assisted revisions, and in cases modified the revisions. The authors take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Section 5 specifies the datasets, sparsifiers, explainers, audit sizes, retention budget, and audit tolerances. Section 4.4 and Appendix A.2 state the assumptions and proof for the finite-population guarantee. Appendices I and J provide the experimental hyperparameters, data splits, and dataset statistics. The anonymized code repository at `https://anonymous.4open.science/r/AuditExSparseGNN-D0FE/` contains the implementation, configuration files, and scripts used to reproduce the reported experiments.

REFERENCES

Chirag Agarwal, Owen Queen, Himabindu Lakkaraju, and Marinka Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1):144, 2023.

S Akansha. Rocp-gnn: Robust conformal prediction for graph neural networks in node-classification. *arXiv preprint arXiv:2408.13825*, 2024.

Selahattin Akkas and Ariful Azad. Explainable graph sparsification with shapley values. In *Proceedings of the ACM Web Conference 2026*, pp. 8557–8560, 2026.

Kenza Amara, Zhitao Ying, Zitao Zhang, Zhihao Han, Yinan Shan, Ulrik Brandes, and Sebastian Schemm. Graphframex: Towards systematic evaluation of explainability methods for graph neural networks. In *NeurIPS 2022 Workshop: New Frontiers in Graph Learning*, 2022.

Consumer Financial Protection Bureau. Adverse action notification requirements in connection with credit decisions based on complex algorithms. Consumer Financial Protection Circular 2022-03, 2022. Official CFPB circular.

Siddhartha Shankar Das, Naheed Anjum Arafat, Muftiqur Rahman, SM Ferdous, Alex Pothen, and Mahantesh M Halappanavar. Sgs-gnn: A supervised graph sparsification method for graph neural networks. *arXiv preprint arXiv:2502.10208*, 2025.

Alexandre Duval and Fragkiskos D Malliaros. Graphsvx: Shapley value explanations for graph neural networks. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 302–318. Springer, 2021.

Mert Kosan, Samidha Verma, Burouj Armgaan, Khushbu Pahwa, Ambuj Singh, Sourav Medya, and Sayan Ranu. GNNX-BENCH: Unravelling the utility of perturbation-based GNN explainers through in-depth benchmarking. In *The Twelfth International Conference on Learning Representations*, 2024.

Jiate Li, Meng Pang, Yun Dong, Jinyuan Jia, and Binghui Wang. Provably robust explainable graph neural networks against graph perturbation attacks. In *The Thirteenth International Conference on Learning Representations*, 2025.

Zirui Liu, Kaixiong Zhou, Zhimeng Jiang, Li Li, Rui Chen, Soo-Hyun Choi, and Xia Hu. Dspar: An embarrassingly simple strategy for efficient gnn training and inference via degree-based sparsification. *Transactions on Machine Learning Research*, 2023.

Donald Loveland, Jiong Zhu, Mark Heimann, Benjamin Fish, Michael T Schaub, and Danai Koutra. On performance discrepancies across local homophily levels in graph neural networks. In *Learning on Graphs Conference*, pp. 6–1. PMLR, 2024.

Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.

Phillip E Pope, Soheil Kolouri, Mohammad Rostami, Charles E Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10772–10781, 2019.

Mandeep Rathee, Thorben Funke, Avishek Anand, and Megha Khosla. Bagel: A benchmark for assessing graph neural network explanations. *arXiv preprint arXiv:2206.13983*, 2022.

Jianqing Song, Jianguo Huang, Wenyu Jiang, Baoming Zhang, Shuangjie Li, and Chongjun Wang. Similarity-navigated conformal prediction for graph neural networks. *Advances in Neural Information Processing Systems*, 37:48541–48567, 2024.

Daniel A Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pp. 563–568, 2008.

Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011. doi: 10.1137/08074489X.

Yang Ye and Shihao Ji. Sparse graph attention networks. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):905–916, 2021.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32, 2019.

Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12241–12252. PMLR, 18–24 Jul 2021.

Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(5): 5782–5799, 2022.

Tianyi Zhao, Jian Kang, and Lu Cheng. Conformalized link prediction on graph neural networks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4490–4499, 2024.

Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. In *International Conference on Machine Learning*, pp. 11458–11468. PMLR, 2020.

Xu Zheng, Farhad Shirani, Tianchun Wang, Wei Cheng, Zhuomin Chen, Haifeng Chen, Hua Wei, and Dongsheng Luo. Towards robust fidelity for evaluating explainability of graph neural networks. In *The Twelfth International Conference on Learning Representations*, 2024.

A EXACT FINITE-POPULATION AUDIT INFERENCE

A.1 AUDITING A FIXED TEST-NODE POPULATION

Consider a reference and deployed GNN pipeline evaluated on one fixed graph and test split. Once the trained models, explainer, sparsifier, tolerances, and random seeds are fixed, each test node either passes or fails the node-level audit. The complete test-node population therefore contains N nodes, of which an unknown number K pass. Let

$$N = |V_{\text{test}}|, \qquad K = \sum_{i \in V_{\text{test}}} C_i^{\epsilon,\beta,\mu}, \qquad p_N = K/N.$$

Here, p_N is the true preservation rate over the complete test pool. An audit samples n test nodes uniformly without replacement and counts how many pass:

$$S = \sum_{i \in \mathcal{I}_{\text{audit}}} C_i^{\epsilon,\beta,\mu}.$$

Since the sample is drawn without replacement from N nodes containing K passes, the audit count satisfies

$$S \sim \text{Hypergeom}(N, K, n).$$

This construction treats the node outcomes as fixed and only the selection of audit nodes as random. It therefore does not require outcomes at neighboring nodes to be independent, even though their predictions and explanations may be coupled through GNN message passing.

A.2 EXACT LOWER CONFIDENCE BOUND

Suppose the audit observes $S = s$ passing nodes. For any candidate population count k, the upper-tail probability

$$\Pr_{X \sim \text{Hypergeom}(N,k,n)}(X \geq s)$$

gives the probability that an audit of n nodes contains at least s passing nodes when exactly k of the N test nodes pass. Candidate values for which this probability is below α are incompatible with the observation at the one-sided confidence level $1 - \alpha$. We therefore define the smallest retained population count as

$$L_\alpha(s; N, n) = \min\left\{ k \in \{0, \ldots, N\} : \Pr_{X \sim \text{Hypergeom}(N,k,n)}(X \geq s) \geq \alpha \right\},$$

and the corresponding lower confidence bound as

$$\underline{p}_\alpha(s; N, n) = \frac{L_\alpha(s; N, n)}{N}.$$

Thus, $\underline{p}_\alpha$ is the smallest test-pool preservation rate that remains compatible with the observed audit count under this tail criterion.

Theorem 1 (Exact finite-population coverage)**.** *Let K of the N fixed test nodes satisfy the audit outcome, and sample n nodes uniformly without replacement. Then*

$$\Pr\left[\underline{p}_\alpha(S; N, n) \leq K/N\right] \geq 1 - \alpha.$$

Thus, with probability at least $1 - \alpha$ over the audit sample, the reported lower confidence bound does not exceed the true fixed-pool preservation rate.

Proof. For a candidate population count k, define

$$T_k(t) = \Pr_{X \sim \text{Hypergeom}(N,k,n)}(X \geq t).$$

For a fixed observed count t, $T_k(t)$ is nondecreasing in k: a population containing more passing nodes makes observing at least t passes more likely.

Coverage fails only when the inferred lower population count exceeds the true count, i.e., $L_\alpha(S; N, n) > K$. By the definition of L_α, this event is equivalent to $T_K(S) < \alpha$.

For the true population count K, $T_K(t)$ is nonincreasing in t. Let

$$c = \min\{t : T_K(t) < \alpha\}.$$

If no such c exists, the failure event is empty. Otherwise,

$$\Pr[L_\alpha(S; N, n) > K] = \Pr(S \geq c) = T_K(c) < \alpha.$$

Taking the complementary event and dividing the population counts by N yields

$$\Pr\Big[\underline{p}_\alpha(S; N, n) \leq K/N\Big] \geq 1 - \alpha.$$

□

The inequality may be strict because the hypergeometric count is discrete. Here, *exact* means that the finite-sample coverage guarantee requires no large-sample approximation.

Simultaneous coverage across structural groups. Before sampling audit nodes, an auditor may partition the complete test pool into m nonempty structural groups, or strata, such as degree or local-homophily bins. For each group z, let N_z be its population size, where $K_z = \sum_{i \in z} C_i^{\epsilon,\beta,\mu}$ denotes the number of its nodes that pass and $p_z = K_z/N_z$ denotes its true preservation rate. Suppose $n_z > 0$ nodes are sampled uniformly without replacement, of which s_z pass the audit. Applying the theorem at level $1 - \alpha/m$ gives the group-specific lower confidence bound

$$\underline{p}_{\alpha/m}(s_z; N_z, n_z).$$

By the theorem, each group-specific bound is at most p_z with probability at least $1 - \alpha/m$. By the union bound, all m bounds are simultaneously at most their respective group preservation rates with probability at least $1 - \alpha$. A worst-case finite-population guarantee at target $1 - \delta$ is established if every adjusted group-specific lower confidence bound is at least $1 - \delta$.

A.3 Complete audit procedure

Algorithm 1 Audit procedure for a deployed sparsified GNN explanation pipeline.

Input: Fixed test pool V_{test}; audit set $\mathcal{I}_{\text{audit}} \sim \text{Unif}\{\mathcal{I} \subseteq V_{\text{test}} : |\mathcal{I}| = n\}$; reference $f_{\theta^\star}$; deployed $f_{\theta'}$ (canonical: $\theta' = \theta^\star$); explainer ϕ; sparsifier Sp; tolerances (ϵ, β, μ); target $1 - \delta$; confidence $1 - \alpha$; prespecified partition $\mathcal{Z}$; budget ρ.

Output: Preservation rates, exact lower confidence bounds, and supported finite-population guarantees.

for each $i \in \mathcal{I}_{\text{audit}}$ **do**
  $w_i \leftarrow \mathsf{Sp}(G, i, \rho)$; $\hat{y}_i^\star, \hat{y}_i' \leftarrow$ predictions of $f_{\theta^\star}(G), f_{\theta'}(G_{w_i})$;
  $\phi_i^\star \leftarrow \phi(i; G, \theta^\star)$; $\phi_i' \leftarrow \phi(i; G_{w_i}, \theta')$; compute $C_i^{\epsilon,\beta,\mu}$.
$N \leftarrow |V_{\text{test}}|$, $n \leftarrow |\mathcal{I}_{\text{audit}}|$, $s \leftarrow \sum_{i \in \mathcal{I}_{\text{audit}}} C_i^{\epsilon,\beta,\mu}$; compute $\text{PresRate} = s/n$ and $\underline{p}_\alpha(s; N, n)$.
for each stratum $z \in \mathcal{Z}$ **do**
  $N_z \leftarrow |z|$, $n_z \leftarrow |\mathcal{I}_{\text{audit}} \cap z|$, $s_z \leftarrow \sum_{i \in \mathcal{I}_{\text{audit}} \cap z} C_i^{\epsilon,\beta,\mu}$.
  Compute $\text{PresRate}_z = s_z/n_z$ and $\underline{p}_{\alpha/|\mathcal{Z}|}(s_z; N_z, n_z)$.
$\text{PresRate}_{\min} \leftarrow \min_{z \in \mathcal{Z}} \text{PresRate}_z$; $\underline{p}_{\min} \leftarrow \min_{z \in \mathcal{Z}} \underline{p}_{\alpha/|\mathcal{Z}|}(s_z; N_z, n_z)$.
Marginal population guarantee: $\underline{p}_\alpha(s; N, n) \geq 1 - \delta$.
Simultaneous conditional population guarantees: $\underline{p}_{\alpha/|\mathcal{Z}|}(s_z; N_z, n_z) \geq 1 - \delta, \forall z \in \mathcal{Z}$.
Worst-case population guarantee: $\underline{p}_{\min} \geq 1 - \delta$.
return $\{C_i^{\epsilon,\beta,\mu}\}_{i \in \mathcal{I}_{\text{audit}}}$; PresRate, $\{\text{PresRate}_z\}_{z \in \mathcal{Z}}$, and $\text{PresRate}_{\min}$; $\underline{p}_\alpha(s; N, n)$, $\{\underline{p}_{\alpha/|\mathcal{Z}|}(s_z; N_z, n_z)\}_{z \in \mathcal{Z}}$, and $\underline{p}_{\min}$; together with the guarantee decisions above.

B AUDITING PIPELINES ON LARGE-SCALE DATASETS

Table 5 evaluates our audit pipeline on Ogbn-arxiv. We use 50 audit nodes, a two-layer GCN backbone with reduced explainer/sparsifier budgets. We include scalable baseline and structural sparsifiers (random, dspar, degdist, spectral, and t-spanner) along with the identity control, paired with GNNExplainer and PGExplainer. We exclude GraphPRI, GraphUltra, and SGCN from this study as we found their implementation to go out-of-memory due to the requirement to materialize dense $|V| \times |E|$ or $|V| \times |V|$ matrices.

We also find that audit cost is dominated by explanation recomputation, not by outcome aggregation. With GNNExplainer, Algorithm 1 takes about 21–22 minutes for 50 audited nodes, whereas PGExplainer takes about 6 minutes under the same audit size. This reflects the cost of per-node explanation optimization on a large graph rather than the cost of computing and aggregating the audit outcomes.

Table 5: Ogbn-arxiv at audit size $n = 50$. Audit time is the elapsed time for audit-node sampling, structural-feature computation, prediction and explanation recomputation, audit-outcome computation, and aggregation. Data preparation, setup, and I/O times are excluded. Lower confidence bounds use exact hypergeometric inversion.

Sparsifier	Explainer	Preservation	Prediction	Lower bound	Audit time (min)
Identity	GNNExpl.	1.00	1.00	0.94	22.35
	PGExpl.	1.00	1.00	0.94	6.01
Random	GNNExpl.	0.78	0.88	0.66	20.91
	PGExpl.	0.82	0.88	0.71	5.98
DSpar	GNNExpl.	0.92	1.00	0.83	20.87
	PGExpl.	0.94	1.00	0.85	5.91
Degree-dist	GNNExpl.	0.88	0.98	0.78	21.03
	PGExpl.	0.88	0.98	0.78	5.85
Spectral	GNNExpl.	0.80	1.00	0.68	20.73
	PGExpl.	0.88	1.00	0.78	5.87
t-spanner	GNNExpl.	0.78	0.88	0.66	20.79
	PGExpl.	0.78	0.88	0.66	5.90

C ADDITIONAL EXPERIMENTS

C.1 FAMILY-FACETED RQ1 SCATTER PLOTS

The family-split correlations reported in Table 3 of Section 5.1 aggregate over all sparsifier-explainer configurations within each (family, dataset) cell. This appendix provides the corresponding scatter plots, which let the reader inspect how individual configurations contribute to the family-level correlation. We present structural and learned/proxy families separately because the sample sizes differ (six structural sparsifiers $\times$ four explainers $= 24$ points per dataset, versus seven learned/proxy sparsifiers $\times$ four explainers $= 28$ points per dataset).

These two figures together show that the family-level disagreement reported in Section 5.1 is not a uniform shift across all homophilic graphs. Instead, it is driven most clearly by the learned/proxy Cora panel (Figure 4, top-left), which is the visual counterpart of the $\rho = 0.20$ entry in Table 3 and shows that prediction preservation is weakly informative about explanation preservation in that regime.

C.2 EXACT FINITE-POPULATION AUDIT-SIZE ANALYSIS

Figure 5 separates the common $n \leq 100$ range from the extended analysis of configurations originally audited at 200 nodes. This keeps the set of configurations fixed within each panel. The continued rise from $n = 100$ to $n = 200$ in the extended cohort rules out a universal 100–200 stabilization claim,

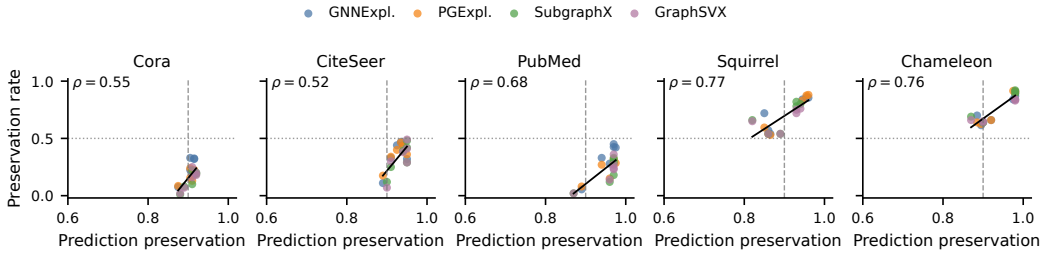


Figure 3: Prediction preservation versus explanation preservation for structural sparsifiers (random, degree-dist, DSpar, spectral, t-spanner, GraphPRI), faceted by dataset. Each point is one (sparsifier, explainer) configuration; color indicates the explainer. Within-dataset Spearman correlations are shown in each panel and are consistently moderate across all five datasets (0.52–0.77).

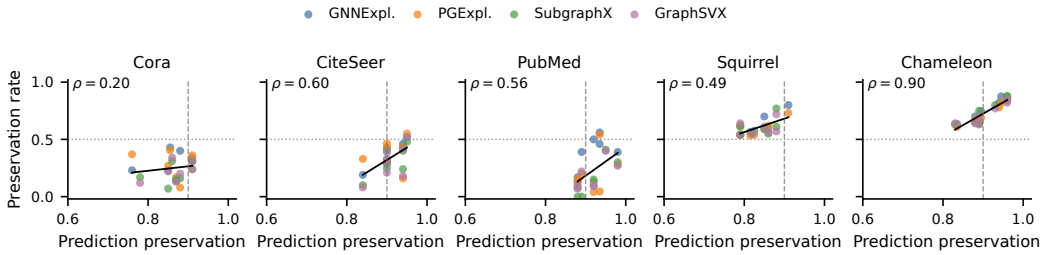


Figure 4: Prediction preservation versus explanation preservation for learned/proxy sparsifiers (NeuralSparse, SparseGAT, SGS-GNN, SGCN, GraphUltra, MoG, Shapley), faceted by dataset. Each point is one (sparsifier, explainer) configuration; color indicates the explainer. Within-dataset correlations are less consistent than for the structural sparsifiers in Figure 3; on Cora in particular ($\rho = 0.20$), high-prediction-preservation configurations span preservation rates from near zero to roughly 0.40.

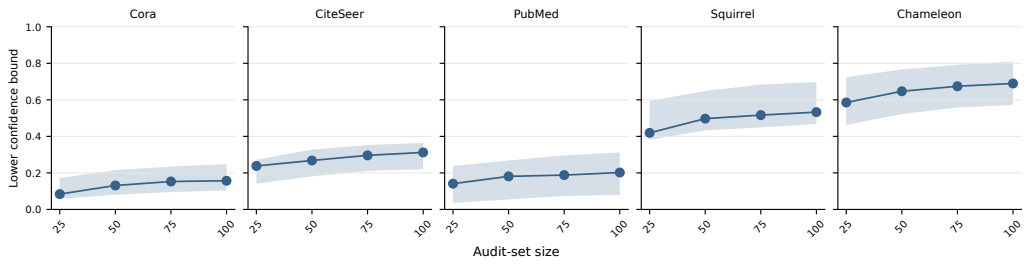
while the diminishing increments illustrate how additional audited nodes progressively tighten the lower confidence bound.

C.3 Prespecified worst-case stratified audit

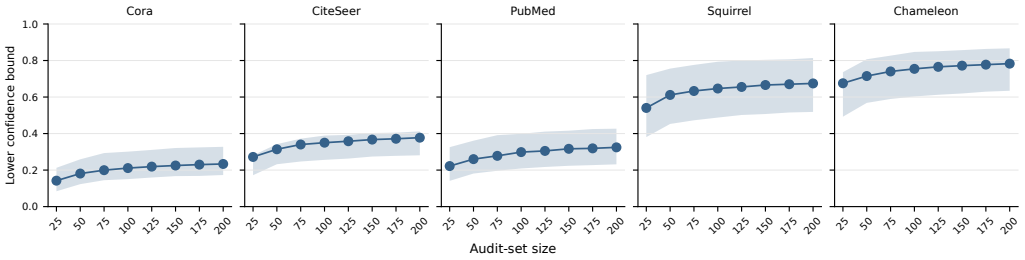
We prespecify $1 - \delta = 0.50$ as an illustrative all-strata target and use simultaneous 95% confidence. The target is a transparent benchmark rather than a universal deployment requirement and is independent of both the keep ratio and the node-level audit-outcome tolerances.

We define degree-by-local-homophily quartiles on the complete test pool before sampling and audit $q = 20$ nodes from every nonempty stratum, using complete enumeration when a stratum contains fewer than 20 nodes. The 260 non-identity pipelines yield 4,108 pipeline–stratum rows; 364 rows are complete censuses. For each pipeline, we report the minimum exact hypergeometric lower confidence bound after Bonferroni adjustment over its nonempty strata.

Figure 6 shows that these simultaneous worst-stratum lower confidence bounds are generally low: their mean is 0.077, their median is 0.010, and their maximum is 0.467. Consequently, none of the 260 pipelines meets the illustrative target of a 0.50 preservation rate in every stratum at the simultaneous 95% confidence level. The three pipelines with the highest bounds each meet the target in 15 of 16 strata, but their remaining high-degree, third-homophily-quartile stratum records 14 successes among 20 audited nodes and a lower confidence bound of 0.467. Thus, the procedure identifies the specific stratum that prevents the all-strata claim. The result concerns the available evidence at $q = 20$ and this illustrative target.



(a) All 280 configurations, with 56 configurations per dataset.



(b) The 88 configurations originally audited at $n = 200$: 16 for each homophilic dataset and 20 for each heterophilic dataset.

Figure 5: Exact finite-population audit-size analysis by dataset. For each displayed audit size, we draw audited nodes without replacement from each original audit and invert the hypergeometric tail using the complete test-pool size. Lines show the median lower confidence bound and shaded regions show the interquartile range across configurations. Panel (a) covers $n \in \{25, 50, 75, 100\}$; panel (b) covers all supported sizes from 25 to 200.

Table 6: Initial and targeted follow-up results for the three pipelines closest to the all-strata target. Their initial weakest group is the Chameleon high-degree (Q4), third-local-homophily-quartile (Q3) group, with $N_z = 45$. The individual exact hypergeometric lower confidence bound uses $\alpha = 0.05$; simultaneous lower confidence bounds use the Bonferroni-adjusted level $\alpha/16$. The follow-up completely enumerates this group.

Sparsifier	Explainer	Initial s/n	Indiv. lower	Initial worst	Census s/N	Final worst	Groups
Degree-dist	PGExplainer	$14/20$	0.556	0.467	$32/45$	0.548	$16/16$
DSpar	PGExplainer	$14/20$	0.556	0.467	$33/45$	0.548	$16/16$
DSpar	SubgraphX	$14/20$	0.556	0.467	$32/45$	0.529	$16/16$

Table 6 first explains why the three closest pipelines do not establish the simultaneous claim at $q = 20$. Their weakest stratum has an empirical preservation rate of 0.700 and an individual exact hypergeometric lower confidence bound of 0.556, which exceeds the 0.50 target. After adjustment across all 16 strata, however, the lower confidence bound is 0.467.

We then conduct a targeted post-hoc follow-up by auditing the remaining 25 nodes in this 45-node stratum. The complete-stratum rates are 0.711 for Degree-dist–PGExplainer and DSpar–SubgraphX and 0.733 for DSpar–PGExplainer. Because this group is now completely enumerated, it has no sampling uncertainty. All three pipelines meet the target in all 16 groups; their final worst-group lower bounds are 0.529–0.548. Thus, additional auditing resolves the initial sampling uncertainty for these three selected pipelines. The follow-up was chosen after observing the $q = 20$ results and does not alter the prespecified full-matrix finding or establish the same conclusion for other pipelines.

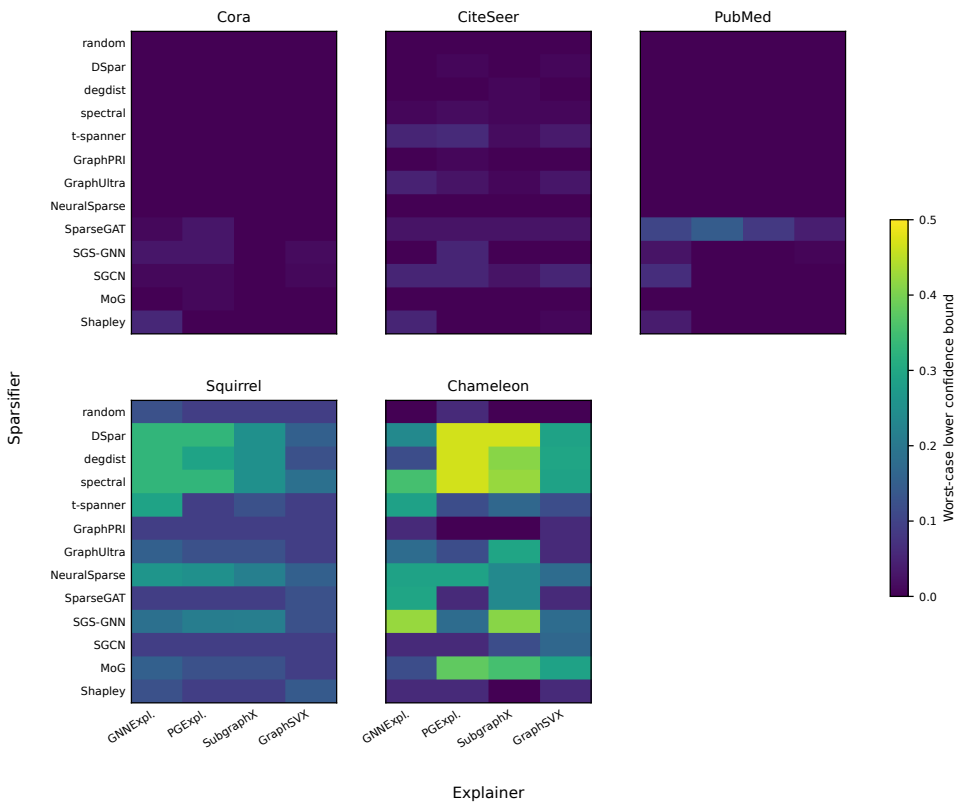


Figure 6: Worst-stratum exact hypergeometric lower confidence bounds for all 260 non-identity pipelines. Degree-by-local-homophily strata are fixed on the complete test pool before sampling. Each cell is the minimum stratum-level lower confidence bound after Bonferroni adjustment over the pipeline's nonempty strata, using $q = 20$ audited nodes per stratum or a complete census when the stratum is smaller. The color scale ends at the illustrative target 0.50.

Table 7: Preservation across degree quartiles within each dataset, excluding the identity control. Preservation rates are averaged over non-identity (sparsifier, explainer) configurations for nodes in each degree quartile.

Dataset	Marginal	Q1	Q2	Q3	Q4	Q1–Q4 gap	Std. dev.
Cora	0.215	0.428	0.195	0.138	0.099	0.329	0.127
CiteSeer	0.341	0.570	0.404	0.275	0.116	0.454	0.167
PubMed	0.234	0.366	0.311	0.167	0.091	0.275	0.110
Squirrel	0.648	0.822	0.682	0.503	0.586	0.236	0.119
Chameleon	0.750	0.867	0.810	0.786	0.537	0.330	0.126

D IMPACT OF SPARSIFICATION REGIME

The main experiments audit learned/proxy sparsifiers in an *off-design* regime: the GCN backbone is trained on the full graph and frozen, while a learned/proxy sparsifier is applied as a frozen preprocessor at inference time. Thus the main audit isolates the effect of the sparsified graph for a fixed downstream GCN, but it does not by itself determine whether a learned/proxy sparsifier would behave differently if its own proxy or jointly trained model were used as the downstream predictor. A natural question is therefore whether the failure-mode signature observed for learned/proxy sparsifiers in Section 5.2 is primarily an artifact of off-design deployment. We test this question with a controlled companion experiment for NeuralSparse.

Table 8: Co-trained vs. off-design audit of NeuralSparse. Reference and deployment differ only by the sparsifier in conditions 1 and 3; condition 2 additionally includes the change from the independently trained full-graph GCN to the co-trained GCN. Mean ± standard deviation over 9 configurations per (dataset, condition). The lower-bound column summarizes per-configuration exact hypergeometric lower confidence bounds.

Dataset	Condition	Pred. pres.	Pres. rate	Lower bound	Flip	Drift	J	M
Cora	Off-design	0.860 ± 0.012	0.174 ± 0.015	0.136 ± 0.014	0.170	0.772	0.058	0.000
Cora	Co-trained	0.848 ± 0.020	0.156 ± 0.012	0.119 ± 0.011	0.180	0.764	0.055	0.000
Cora	Co-trained matched ref.	0.881 ± 0.020	0.176 ± 0.017	0.138 ± 0.015	0.144	0.803	0.053	0.000
Chameleon	Off-design	0.934 ± 0.020	0.858 ± 0.030	0.823 ± 0.032	0.455	0.532	0.013	0.000
Chameleon	Co-trained	0.834 ± 0.021	0.751 ± 0.023	0.710 ± 0.024	0.667	0.304	0.029	0.000
Chameleon	Co-trained matched ref.	0.925 ± 0.015	0.852 ± 0.028	0.817 ± 0.031	0.513	0.467	0.020	0.000

We compare three audit conditions on Cora (homophilic) and Chameleon (heterophilic), all at keep ratio $\rho = 0.5$ with a 2-layer GCN backbone (hidden dim 64, dropout 0.5) and the revised PyG GNNExplainer as the reference explainer. We use 3 backbone seeds and 3 audit-node sampling seeds per (dataset, condition) cell, for 9 cells per condition and dataset. Audit size is $n = 200$ nodes and audit-outcome tolerances match the main paper at $(\epsilon, \beta, \mu) = (0.5, 0.5, 0.5)$.

Off-design (Condition 1). A GCN is trained on the full graph G and frozen. NeuralSparse is trained separately through its proxy model and then used only to produce inference-time edge weights. The reference and deployment models are identical full-graph GCNs; the reference side uses G, while the deployment side uses the NeuralSparse-weighted graph.

Co-trained, full-graph reference (Condition 2). NeuralSparse + GCN trained jointly end-to-end on the same train split. The deployed model is the co-trained GCN; the reference model is the independently trained full-graph GCN. Reference and deployment therefore differ both in model parameters and in graph input.

Co-trained, matched-model reference (Condition 3). The reference and deployment models are both the co-trained GCN from condition 2. The only difference is the graph input: full graph for the reference side and NeuralSparse-weighted graph for deployment.

The headline comparison is condition 1 vs. condition 3: both are clean audits, and they differ only in whether the GCN backbone was trained off-design or on-design with respect to NeuralSparse. Condition 2 serves as a sanity check that demonstrates why condition 3 is the right control: using an independently trained full-graph reference against a co-trained deployment mixes the model change with the sparsifier change.

Observations. The condition 1 vs. condition 3 comparison in Table 8 isolates the sparsifier contribution from the training regime of the downstream GCN. We find no evidence that co-training improves the clean preservation rate for NeuralSparse: off-design minus matched-reference preservation is -0.002 on Cora (0.174 vs. 0.176) and $+0.006$ on Chameleon (0.858 vs. 0.852), both small relative to the across-configuration standard deviations. The failure-mode shares also do not show a consistent reduction in prediction-flip failures: the flip share decreases from 0.170 to 0.144 on Cora but increases from 0.455 to 0.513 on Chameleon. The residual condition 1–condition 3 differences are spread across flip, drift, and top-k rearrangement rather than localized to one failure mode.

Condition 2 is intentionally more stringent and less controlled because the reference and deployment models differ. In that cross-model comparison, co-trained preservation is lower than off-design on both datasets (-0.018 on Cora and -0.107 on Chameleon), which confirms that changing the downstream model can itself introduce audit failures. We therefore base the off-design conclusion on condition 3: for NeuralSparse under this GNNExplainer-based companion experiment, the NeuralSparse audit behavior is not explained by the frozen-preprocessor deployment alone.

Remarks. This is a single-sparsifier, two-dataset companion experiment with GNNExplainer fixed as the reference explainer. The negligible off-design difference observed here for NeuralSparse does not imply that all learned/proxy sparsifiers behave this way; methods such as SparseGAT, SGCN, and SGS-GNN may couple the sparsifier and backbone differently. Regarding efficiency, each NeuralSparse co-training run averaged 0.92 minutes for one dataset and backbone seed. Each

subsequent audit run, using 200 audit nodes for one dataset, condition, backbone seed, and audit-node seed, averaged 8.77 minutes.

E Reference Faithfulness and Fidelity-Stratified Robustness

Scope under an unfaithful reference. Our framework evaluates whether a sparsified deployment reproduces the prediction and explanation behavior of its designated full-graph reference within pre-specified tolerances. It does not establish that either explanation is faithful or correct. If the reference explanation is unfaithful, high preservation means that the deployment reproduces that unfaithful reference, while low preservation identifies a change without determining which explanation is better. The auditor's task is therefore to quantify the explanation change introduced by sparsification, while reference faithfulness remains a separate explainer-quality question that affects interpretation of the result. Reference-relative guarantees have precedent in robustness certification, where stability is assessed relative to a model's clean output rather than the correctness of that output. XGNNCert (Li et al., 2025), for instance, certifies that an explanation remains close to its clean-graph reference under bounded adversarial graph perturbations while preserving the prediction, without establishing that the reference explanation is correct.

Model-focused fidelity of reference explanations. To investigate whether the observed prediction–explanation gap is limited to low-fidelity references, we stratify the reference explanations using model-focused sufficiency and necessity tests (Amara et al., 2022; Yuan et al., 2022). Let T_i contain the k_i highest-scored reference edges in node i's receptive field. Let G_i^{keep} retain only T_i within that receptive field, and let G_i^{remove} remove T_i while retaining the remaining edges. Using the same full-graph GNN, we define

$$S_i \triangleq \mathbf{1}\left[\hat{y}_i(G_i^{\text{keep}}, \theta) = \hat{y}_i(G, \theta)\right], \qquad N_i \triangleq \mathbf{1}[\hat{y}_i(G_i^{\text{remove}}, \theta) \neq \hat{y}_i(G, \theta)].$$

Sufficiency ($S_i = 1$) means that the selected edges alone preserve the full-graph prediction. Necessity ($N_i = 1$) means that removing those edges changes the prediction. Based on these criteria, we partition the references into $S \cap N$, $S \cap \neg N$, $\neg S \cap N$, and $\neg S \cap \neg N$.

Table 9: Reference-explanation fidelity and preservation at keep ratio $\rho = 0.5$. The four stratum columns report the percentage of the 100 reference explanations in each sufficiency–necessity category for every dataset–explainer pair. The final two columns report prediction preservation and preservation rate within $S \cap N$, pooling outcomes across Random, DSpar, NeuralSparse, and SparseGAT.

		Reference-fidelity stratum share (%)				Within $S \cap N$ (%)	
Dataset	Explainer	$S \cap N$	$S \cap \neg N$	$\neg S \cap N$	$\neg S \cap \neg N$	Pred. preserved	Preservation rate
Cora	GNNExplainer	21	74	2	3	65.5	22.6
Cora	PGExplainer	23	68	1	8	64.1	19.6
Cora	SubgraphX	16	77	0	7	67.2	25.0
Cora	GraphSVX	13	76	3	8	53.8	13.5
Chameleon	GNNExplainer	24	76	0	0	76.0	49.0
Chameleon	PGExplainer	23	76	1	0	77.2	41.3
Chameleon	SubgraphX	11	79	0	10	81.8	54.5
Chameleon	GraphSVX	12	80	0	8	83.3	52.1

The analysis uses the same 100 test nodes per dataset for Cora and Chameleon, all four explainers, and Random, DSpar, NeuralSparse, and SparseGAT. Within $S \cap N$, prediction preservation exceeds the preservation rate for every dataset–explainer group: 53.8%–67.2% versus 13.5%–25.0% on Cora, and 76.0%–83.3% versus 41.3%–54.5% on Chameleon. Preservation failures are therefore not limited to references that fail these fidelity measures.

F Robustness Across GNN Backbones

To assess whether the observed gap between prediction preservation and explanation preservation is specific to the GCN backbone, we repeat a node-matched subset of the audit using a two-layer

GraphSAGE model. The comparison covers Cora and Chameleon; Random, DSpar, NeuralSparse, and SparseGAT; and GNNExplainer and PGExplainer. Each backbone therefore has 16 non-identity configurations evaluated on the same 100 audit nodes in each configuration.

Table 10: Aggregate node-matched GCN and GraphSAGE results. Each backbone has 16 configurations evaluated on 100 audit nodes per configuration. The last column averages the configuration-specific exact one-sided 95% lower confidence bounds.

Backbone	Runs	Pred. preservation	Pres. rate	Lower bound
GCN	16	0.898	0.489	0.428
GraphSAGE	16	0.900	0.499	0.438

Table 11: Mean $\pm$ standard deviation across the eight matched configurations for each backbone–dataset pair.

Metric	GCN (Cora)	GCN (Chameleon)	GraphSAGE (Cora)	GraphSAGE (Chameleon)
Pred. preservation	0.863 ± 0.055	0.933 ± 0.041	0.818 ± 0.066	0.983 ± 0.019
Preservation rate	0.198 ± 0.115	0.781 ± 0.116	0.196 ± 0.109	0.801 ± 0.106

Prediction preservation exceeds the preservation rate in every matched configuration for both backbones, which supports the central finding of this paper. The dataset-level results show that the magnitude of the gap remains deployment-specific, for instance, the mean gap is 0.665 for GCN and 0.621 for GraphSAGE on Cora, compared with 0.151 and 0.181 on Chameleon.

G Sensitivity to Preservation Tolerances

To evaluate sensitivity of the pipelines to tolerance choices, we evaluate all $5^3 = 125$ combinations of (ϵ, β, μ) in $\{0.3, 0.4, 0.5, 0.6, 0.7\}^3$ for all 260 non-identity pipelines. Table 12 shows a representative strict, default, and loose settings from our evaluation. We observe that, looser tolerances increase the preservation rate, but prediction preservation remains substantially higher at all three settings. Across the full spectrum of tolerance values we analysed, drift remains the main aggregate failure mode and the mean Q1–Q4 degree gap remains between 0.286 and 0.336. As the chosen tolerances determine what counts as preservation, they should be selected apriori the audit to represent the amount of explanation change acceptable for the deployment.

Table 12: Tolerance sensitivity across all 260 non-identity pipelines. Each row reports the mean prediction preservation and preservation rate across the pipelines at the indicated tolerance setting.

Setting	(ϵ, β, μ)	Pred. preservation	Pres. rate	Gap
Strict	$(0.3, 0.7, 0.7)$	0.904	0.377	0.527
Default	$(0.5, 0.5, 0.5)$	0.904	0.438	0.466
Loose	$(0.7, 0.3, 0.3)$	0.904	0.541	0.363

To show how individual tolerances affect both the preservation rate and failure-mode decomposition, we examine a representative Cora subset comprising DSpar, NeuralSparse, and SparseGAT with GNNExplainer and PGExplainer. Figure 7 varies the drift and top-k overlap tolerances while fixing $\mu = 0.5$. Figure 8 shows that relaxing the drift tolerance reduces drift failures and shifts the remaining failures toward top-k rearrangement. For an individual pipeline, the most frequent failure mode can therefore change when ϵ changes, even though the graph, model, sparsifier, and explainer remain fixed.

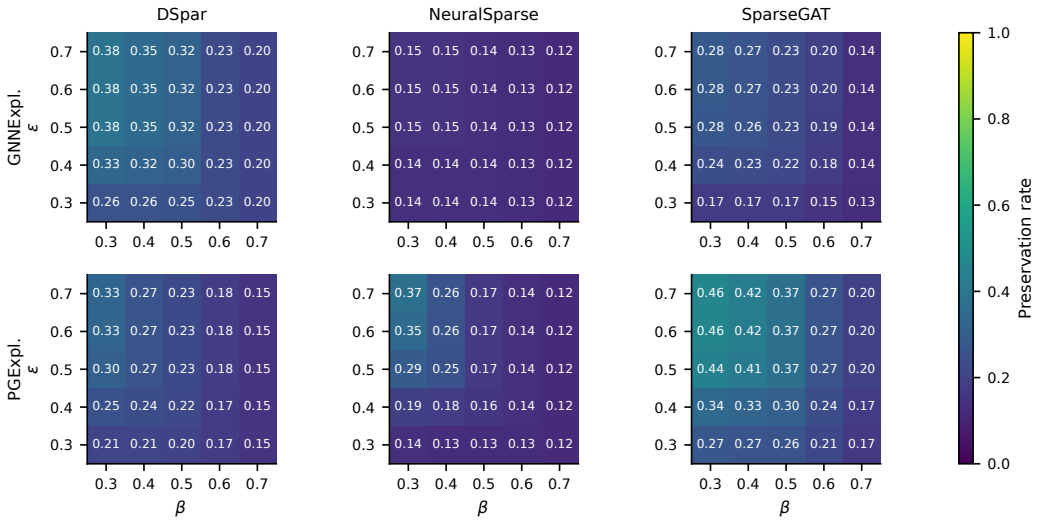


Figure 7: Sensitivity of preservation rate to the drift tolerance ϵ and top-k overlap tolerance β on Cora (mass tolerance fixed at $\mu = 0.5$).

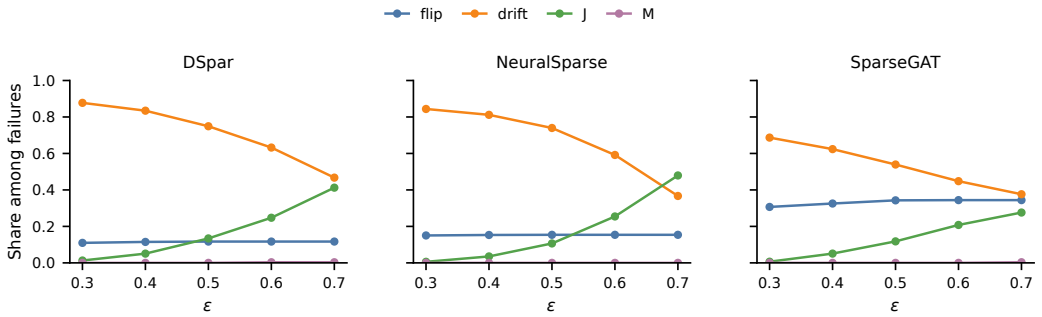


Figure 8: Failure-mode shares across drift tolerances ϵ for Cora, with $\beta = \mu = 0.5$ fixed. Drift remains the leading failure mode in all settings except NeuralSparse at $\epsilon = 0.7$, where top-k rearrangement becomes largest.

H SEED STABILITY

Table 13 reports the across-seed variation for the representative configuration described in the main text: (Cora, Chameleon) datasets x (random, DSpar, and SparseGAT) sparsifiers x (GNNExplainer, PGExplainer) combinations. Each row aggregates a 3×3 grid of backbone seeds $\in \{0, 1, 2\}$ and audit-node sampling seeds $\in \{0, 1, 2\}$, yielding nine runs per (dataset, sparsifier, explainer) configuration.

The resulting variation is modest relative to the main effects reported in Section 5. The mean standard deviation of the preservation rate is 0.030, and the largest row-wise standard deviation is 0.046. This suggests that changing the backbone initialization and sampled audit nodes typically changes the preservation rate by only a few percentage points in this analysis. This variation is much smaller than the dataset-level differences between Cora and Chameleon and smaller than the gaps between prediction preservation and explanation preservation reported in the main text. Thus the main conclusions drawn from the research questions are unlikely to be artifacts of a particular seed choice.

Table 13: Seed stability on the representative slice used in the main text. Each row aggregates a 3×3 grid of backbone and audit-node sampling seeds, giving 9 runs per row.

Dataset	Sparsifier	Explainer	Preservation rate	Prediction preservation
Cora	random	GNNExplainer	0.068 ± 0.018	0.869 ± 0.018
Cora	random	PGExplainer	0.090 ± 0.016	0.869 ± 0.018
Cora	dspar	GNNExplainer	0.328 ± 0.016	0.920 ± 0.004
Cora	dspar	PGExplainer	0.254 ± 0.032	0.920 ± 0.004
Cora	SparseGAT	GNNExplainer	0.233 ± 0.023	0.826 ± 0.039
Cora	SparseGAT	PGExplainer	0.345 ± 0.039	0.826 ± 0.039
Chameleon	random	GNNExplainer	0.609 ± 0.036	0.925 ± 0.027
Chameleon	random	PGExplainer	0.622 ± 0.036	0.925 ± 0.027
Chameleon	dspar	GNNExplainer	0.856 ± 0.026	0.956 ± 0.022
Chameleon	dspar	PGExplainer	0.891 ± 0.027	0.956 ± 0.022
Chameleon	SparseGAT	GNNExplainer	0.712 ± 0.040	0.889 ± 0.019
Chameleon	SparseGAT	PGExplainer	0.688 ± 0.046	0.889 ± 0.019
		Mean preservation-rate std.	0.030	
		Max preservation-rate std.	0.046	

I HYPERPARAMETERS AND REPRODUCIBILITY

Our adaptations to the existing Sparsifiers. Some sparsification methods require adaptations to fit the common evaluation setting. We describe those adaptations for reproducibility.

The effective-resistance method (Spectral) ranks edges by approximate resistance and applies deterministic top-k pruning rather than stochastic reweighted spectral sampling. The t-spanner method is a sampled shortest-path-tree ranker rather than a guaranteed stretch-t spanner. We evaluate an adaptation of Akkas and Azad (Akkas & Azad, 2026) under the label Shapley. It computes local edge Shapley values, aggregates them into global edge scores, and removes the lowest-scoring edges. A separately trained proxy GCN produces these scores, which are frozen before the sparsified graph is audited.

Explainer implementations. PGExplainer is the only explainer (among the four we studied) with learned global explainer parameters. We train it once on the full graph and keep it fixed during the audit. GNNExplainer, SubgraphX, and GraphSVX do not learn global explainer parameters in our implementation. We therefore run them separately for each query node using fixed hyperparameters and deterministic seeds. For GraphSVX, the native explanation assigns Shapley values to neighboring nodes; we convert these node scores into edge scores so that all explainers can be compared in the same edge-coordinate space.

Hardware, Backbone architecture and hyperparameters. All experiments are run on a system with 64 x 2 CPU cores, Intel(R) Xeon(R) Gold 6242 processors, 500 GB RAM, and an NVIDIA Tesla V100 GPU with 32 GB memory.

The backbone used is a two-layer GCN with hidden dimension 64 and dropout 0.5, trained with Adam optimizer for up to 200 epochs, patience 50, learning rate 0.01, and weight decay 0.0005.

J DATASET STATISTICS

Table 14: Key sparsifier hyperparameters used in our experiments.

Sparsifier	Parameters
GraphPRI	`alpha=0.005; beta=2.0; device=auto; entropy_mode=auto; epochs=1000; hard=true; lr=0.01; max_exact_nodes=256; n_samples=3; seed=0; tau=1.0`
GraphUltra	`device=auto; epsilon=0.0001; hidden_channels=16; l1_times=2; lambda1=1000000.0; pretrain_dropout=0.5; pretrain_epochs=200; pretrain_lr=0.01; pretrain_weight_decay=0.0005; projection_prune_ratio=0.15; seed=0; sparse_dropout=0.0; sparse_epochs=8; sparse_lr=0.2; sparse_weight_decay=0.0`
MoG	`device=auto; dropout=0.5; epochs=10; expert_select=2; hidden_channels=32; hidden_spl=32; k_list=[0.9, 0.7, 0.5]; lam=0.01; load_balance_coef=0.01; lr=0.01; num_layers_spl=2; patience=5; seed=0; temp_n=10; temp_r=0.001; use_topo=true; weight_decay=0.0001`
Spectral	`jl_seed=0; maxiter=2000; num_projections=200; tol=1e-06`
t-spanner	`landmark_seed=0; max_landmarks=400; min_landmarks=16`
NeuralSparse	`device=auto; dropout=0.5; epochs=50; hidden_channels=64; lr=0.01; min_temperature=0.05; neighborhood_budget=null; patience=null; seed=0; temperature_decay=0.001; temperature_update_interval=50; train_keep_ratio=0.5; weight_decay=0.0005`
SparseGAT	`device=auto; dropout=0.3; epochs=100; heads=4; hidden_channels=8; lr=0.005; seed=0; weight_decay=0.0005`
SGS-GNN	`device=auto; dropout=0.5; epochs=100; hidden_channels=64; lr=0.01; seed=0; weight_decay=0.0005`
SGCN	`admm_epochs=10; admm_iterations=4; device=auto; dropout=0.5; early_stopping=10; hidden_channels=16; learning_rate=0.001; pretrain_epochs=200; pretrain_learning_rate=0.01; prune_ratio=10.0; rho=0.001; seed=0; weight_decay=0.0005`
Shapley	`aggregation=mean; apply_abs=false; device=auto; dropout=0.5; epochs=100; explain_split=all; hidden_channels=64; high_weight=1000.0; hop_radius=2; lr=0.01; max_explain_nodes=128; max_local_edges=64; num_samples=128; patience=20; ridge=1e-05; seed=0; weight_decay=0.0005`

Table 15: Explainer settings used in our evaluation. GNNExplainer uses PyG's optimization-based implementation with the listed mask-optimization settings.

Explainer	Parameters
GNNExplainer	`edge_ent=1.0; edge_size=0.005; epochs=100; lr=0.01; seed=0`
PGExplainer	`edge_ent=0.001; edge_size=0.0001; epochs=30; lr=0.003; max_train_nodes=512; seed=0`
SubgraphX	`coalition_prob=0.5; hop_radius=2; max_local_edges=200; num_samples=64; seed=0`
GraphSVX	`high_weight=1000.0; hop_radius=2; max_local_edges=null; max_local_nodes=64; num_samples=400; projection=max; ridge=1e-05; seed=0`

Table 16: Dataset statistics.

Dataset	**#nodes**	**#edges**	**Avg. deg.**	**Edge Homophily ratio**	**Train/val/test**	**#Classes**
Cora	2,708	10,556	3.90	0.810	140/500/1,000	7
CiteSeer	3,327	9,104	2.74	0.736	120/500/1,000	6
PubMed	19,717	88,648	4.50	0.802	60/500/1,000	3
Squirrel	5,201	217,073	41.74	0.224	3,121/1,040/1,040	5
Chameleon	2,277	36,101	15.85	0.235	1,366/455/456	5
Ogbn-arxiv	169,343	1,166,243	6.89	0.655	90,941/29,799/48,603	40