RECAP: RESIDUAL COMMUNITY ADHESION PATTERNS FOR GENERALIST GRAPH ANOMALY DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generalist graph anomaly detection aims to identify anomalous nodes in unseen graphs without target-specific retraining. Existing approaches often rely on source-graph labels, target prompts, or coarse normal/anomalous prototypes, which may be unreliable when labels are unavailable and both normal and anomalous patterns are diverse. This paper proposes **RECAP**, a framework based on **RE**sidual **C**ommunity **A**dhesion **P**atterns for label-free cross-graph detection. Rather than transferring a binary anomaly boundary, the method learns a residual-pattern mapping by aligning soft community assignments with similarity graphs constructed from high-order ego-neighbor residual representations. During inference, predicted assignments induce community prototypes, and anomalies are ranked by prototype adhesion deviation with an auxiliary neighborhood community-context inconsistency term. This formulation captures diverse residual patterns while preserving a decomposed score that attributes anomalies to prototype deviation, local community-context mismatch, or both. Cross-dataset transfer experiments show that, without source labels, target prompts, or target fine-tuning, RECAP achieves competitive anomaly-ranking performance under multiple source–target splits. Beyond ranking accuracy, the framework provides model-intrinsic diagnostics through community prototypes, decomposed adhesion/context scores, and node-level reports, offering structured behavioral evidence beyond a scalar anomaly score.

1 INTRODUCTION

Graph Anomaly Detection (GAD) aims to identify anomalous nodes that significantly deviate from the majority distribution in attributes, structures, or local interaction patterns from relational data. This problem widely exists in complex systems such as financial transaction networks, social interaction networks, and e-commerce networks(Yang et al., 2021; Wu et al., 2022; Wang et al., 2023), and serves as a fundamental task for risk identification, fraud detection, and anomalous behavior analysis. Different from anomaly detection on independent samples(Chandola et al., 2009), GAD not only focuses on feature deviations of individual nodes, but also needs to characterize whether the semantic and structural relationships between nodes and their neighborhoods are disrupted, making it more challenging(Akoglu et al., 2015; Ding et al., 2019; Qiao et al., 2025b). Traditional GAD methods are usually trained and tested on a single target graph, assuming that the training and test graphs follow the same or similar data distributions(Ding et al., 2019; Fan et al., 2020; Liu et al., 2022). To overcome this limitation, recent studies have begun to shift from the "one-for-one" paradigm to the "one-for-all" paradigm, namely generalist GAD(Liu et al., 2024): after being trained once on source graphs, the model is expected to directly transfer to unseen target graphs for anomaly detection without retraining or fine-tuning on the target graph.

Existing generalist GAD methods have made preliminary progress in zero-shot or few-shot cross-graph detection scenarios through in-context learning, prompting mechanisms, prototype modeling, and graph foundation models(Liu et al., 2024; Niu et al., 2024; Qiao et al., 2025a; Zheng et al., 2026a; Liu et al., 2026; Zheng et al., 2026b). However, they still face three key limitations. First, many methods rely on source-graph normal/anomalous labels or target-graph prompt samples. Such signals may be unreliable when labels are scarce or noisy, or when anomaly semantics shift across graphs. Second, they often compress transferable cross-graph knowledge into coarse normal/anomalous

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.

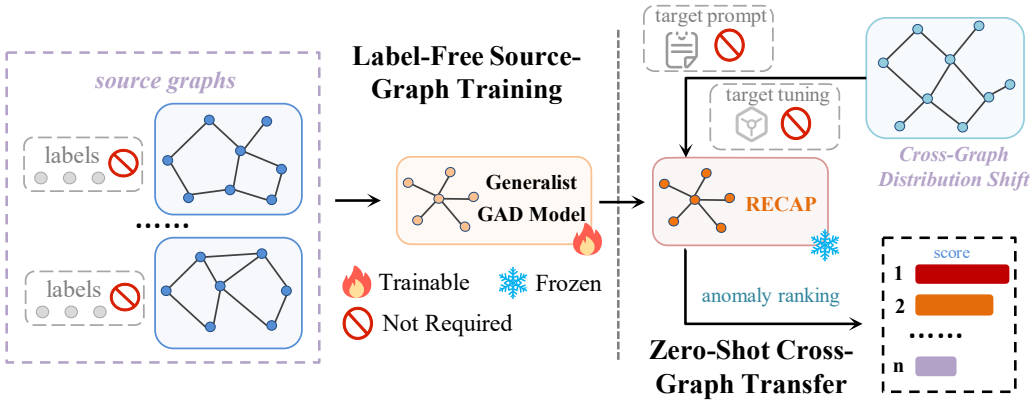


Figure 1: RECAP: generalist GAD without source labels, target prompts, or target fine-tuning.

boundaries or binary prototypes, which are insufficient to capture diverse local behavioral mechanisms in real graphs. Finally, most methods mainly output node-level anomaly scores, with limited abstraction of which local patterns anomalous nodes belong to and where their anomalousness originates. Based on these observations, this paper asks whether a transferable residual-pattern partitioning function can be learned under a label-free generalist setting, enabling both anomaly ranking on unseen graphs and structured diagnostic evidence.

The core idea of this paper is to discover transferable local residual-pattern structures and build anomaly identification upon them. Specifically, compared with directly comparing raw node attributes or high-order neighborhood aggregation features, high-order residuals between nodes and their ego-neighbor representations emphasize the behavioral deviation of a node relative to its local neighborhood, and are therefore more suitable as local pattern representations for cross-graph transfer. In this residual space, nodes with similar local deviations form transferable residual communities. Anomalous nodes are thus those not stably absorbed by this structure, with anomalousness reflected as residual-community adhesion mismatch. The mismatch has two complementary forms: prototype-level mismatch, where a node deviates from its assigned community prototype, and context-level mismatch, where its assignment conflicts with the community context of residual-similar neighbors. This perspective accommodates diverse normal and anomalous patterns while preserving decomposed, model-intrinsic diagnostic evidence.

Based on this perspective, this paper proposes RECAP, a label-free generalist GAD framework. RECAP constructs high-order ego-neighbor residual representations from feature-aligned node attributes, builds a residual similarity graph, and learns a transferable soft community assignment function by enforcing consistency among residual-similar nodes. During inference, it estimates target-graph community prototypes from predicted assignments and scores anomalies through residual-community adhesion mismatch, decomposed into prototype-level deviation and context-level inconsistency. Thus, RECAP outputs both node-level anomaly rankings and structured diagnostics, including community memberships, community prototypes, prototype adhesion deviations, and neighborhood context inconsistency. The main contributions of this paper are as follows:

- We introduce a label-free generalist GAD setting that excludes source-graph anomaly labels, target-graph prompts, and target-graph fine-tuning, offering a stricter protocol for cross-graph anomaly detection.
- We propose RECAP, which learns transferable soft residual communities from high-order ego-neighbor residual representations through unsupervised community consistency learning.
- We design a decomposed residual-community scoring mechanism that captures prototype-level and context-level mismatches, enabling competitive label-free transfer and model-intrinsic anomaly diagnostics.

2 Related Work

Generalist Graph Anomaly Detection Graph Anomaly Detection (GAD) aims to identify anomalous instances whose attributes, relations, or local patterns deviate from the majority distribution in relational data(Akoglu et al., 2015; Ma et al., 2021). Traditional GAD methods typically learn detectors on a single target dataset and assume similar training and test distributions(Ding et al., 2019; Fan et al., 2020; Liu et al., 2022), making them brittle under cross-dataset and cross-domain shifts in node attributes, graph structures, and anomaly semantics. To address this issue, recent studies have investigated generalist GAD, where models trained on source graphs are expected to generalize directly to unseen target graphs(Liu et al., 2024). Existing methods follow several directions: in-context learning and prompting construct transferable detection contexts or neighborhood prompts(Liu et al., 2024; Niu et al., 2024); invariant representation learning and test-time adaptation mitigate source-target distribution shifts(Zhang et al., 2025; 2026); and graph foundation models, prototype learning, dynamic prompting, and multi-pattern dictionaries explore general anomaly knowledge representations(Qiao et al., 2025a; Zheng et al., 2026a;b). Recent work further extends generalist GAD to settings such as geometric heterogeneity, routing robustness, and group anomaly detection(Zhao et al., 2026; Jiang et al., 2026; Chen et al., 2026). Despite their progress, these methods often rely on task-related supervision or target-side auxiliary information, such as source-graph normal/anomalous labels or target-graph context nodes. In contrast, RECAP uses no source labels, target context, or target fine-tuning and transfers a residual-pattern partitioning function rather than a supervised anomaly scorer or normal/anomalous prototype.

Unsupervised Graph Anomaly Detection Since anomaly labels are scarce and costly in real-world scenarios, unsupervised graph anomaly detection has long been a core direction in GAD. Existing methods characterize anomalies using unsupervised signals such as reconstruction, contrastive learning, distribution modeling, and generative modeling(Li et al., 2017; Ma et al., 2021). Reconstruction-based methods regard anomalies as nodes that are hard to recover from normal patterns, including DOMINANT (Ding et al., 2019) and AnomalyDAE (Fan et al., 2020), which reconstruct node attributes and graph structure. Beyond reconstruction, contrastive methods identify anomalies through representation consistency between nodes and local subgraphs(Liu et al., 2022; Jin et al., 2021; Xu et al., 2022), while DiffGAD uses diffusion processes to model anomaly distributions(Li et al., 2025). These methods optimize a detector directly on each target graph. RECAP instead trains once on unlabeled source graphs and freezes the residual-community assignment function for unseen targets. Although RECAP adopts ARC's multi-hop residual encoder (Liu et al., 2024), ARC uses the residual embeddings for supervised cross-attentive scoring with labeled normal target contexts; RECAP constructs an unlabeled residual similarity graph, learns soft communities, induces target prototypes, and decomposes anomaly scores into prototype and neighborhood-context mismatch.

3 Methodology

3.1 Preliminaries

GAD. Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be an attributed graph with n nodes, adjacency matrix $\mathbf{A} \in \{0,1\}^{n\times n}$, and node feature matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$. The anomaly label vector is $\mathbf{y} \in \{0,1\}^n$, where $y_i = 1$ indicates that v_i is anomalous and $y_i = 0$ otherwise. Graph anomaly detection aims to learn a scoring function $f : \mathcal{V} \to \mathbb{R}$ that assigns larger scores to anomalous nodes than to normal nodes, i.e., $f(v_a) > f(v_n)$ for $v_a \in \mathcal{V}_a$ and $v_n \in \mathcal{V}_n$. In unsupervised GAD, f is trained only with graph structure and node attributes, while labels are used only for evaluation.

Generalist GAD. Generalist GAD studies cross-graph anomaly detection, where a model trained on several source graphs is directly transferred to unseen target graphs without target fine-tuning (Liu et al., 2024). Let $T_{\text{train}} = \{D^{(i)}_{\text{train}}\}_{i=1}^{N_1}$ and $T_{\text{test}} = \{D^{(j)}_{\text{test}}\}_{j=1}^{N_2}$ denote the source and target graph collections, with $T_{\text{train}} \cap T_{\text{test}} = \emptyset$. The goal is to learn a generalist scoring function from T_{train} that can rank anomalous nodes in each unseen target graph despite cross-domain distribution shifts.

Label-free generalist setting. RECAP further excludes anomaly labels from every source graph and excludes labeled/unlabeled target prompts, reference nodes, checkpoint selection, and parameter

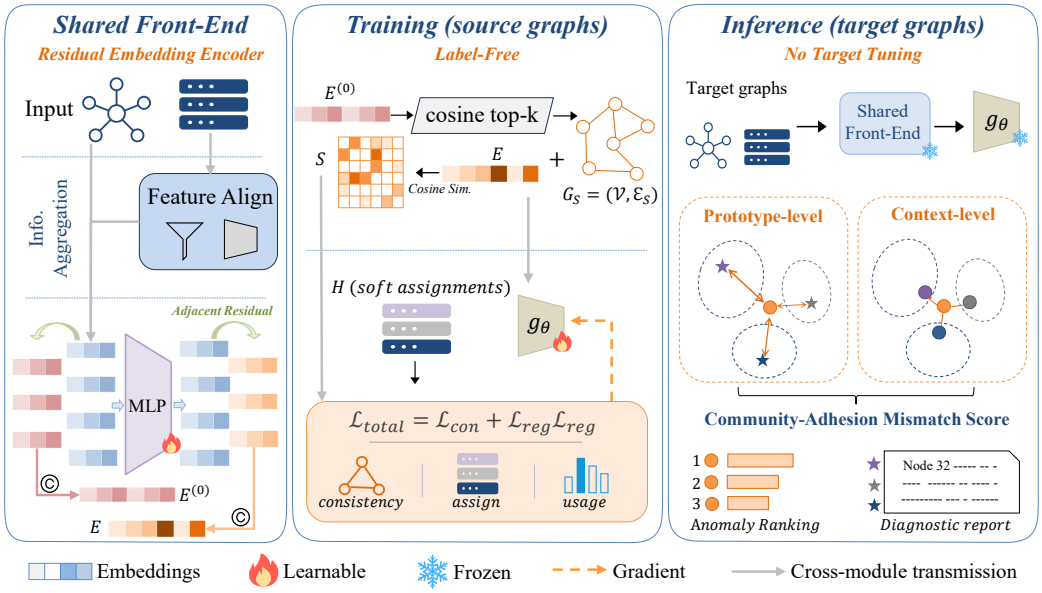


Figure 2: Overview of RECAP.

updates. Target attributes and topology are used only in a frozen forward pass and in transductive, label-free prototype construction. This stricter setting is distinct from unsupervised one-for-one (OFO) detection, which fits a separate model on each target graph.

3.2 Overview of RECAP

RECAP targets label-free generalist graph anomaly detection by learning a transferable residual-pattern assignment function without source-graph anomaly labels, target-graph prompt samples, or target-graph fine-tuning. Its core idea is to cast cross-graph anomaly detection as community structure discovery in the ego-neighbor residual pattern space, where residual patterns describe node-neighborhood discrepancies, communities summarize recurring local patterns, and anomalies are nodes that cannot be stably absorbed into the residual community structure.

As shown in Fig. 2, RECAP first aligns node attributes across graph datasets and extracts high-order ego-neighbor residual embeddings. It then builds a residual similarity graph and learns soft community assignments with an unsupervised community-consistency objective. During inference, the learned assignment function is directly applied to the target graph, where community prototypes are estimated from target soft assignments. Anomaly scores are computed from prototype-level and context-level residual-community adhesion mismatch, with memberships, prototypes, and decomposed scores serving as diagnostic evidence.

3.3 Feature Alignment

Graphs from different domains often have different feature dimensions, scales, and distributions, which weakens the transferability of a shared encoder. RECAP therefore aligns numerical node attributes through three lightweight steps: robust pre-normalization, PCA projection, and coordinate-wise post-normalization.

Given an attributed graph $G = (\mathcal{V}, \mathcal{E}, X)$ with raw node attributes $X \in \mathbb{R}^{n \times d_0}$, RECAP first applies robust normalization to each feature dimension:

$$\widetilde{x}_{ij} = \frac{x_{ij} - \mathrm{median}(X_{\cdot j})}{\mathrm{IQR}(X_{\cdot j}) + \epsilon}, \tag{1}$$

where $\mathrm{IQR}(\cdot)$ denotes the interquartile range and ϵ avoids division by zero. Compared with mean-variance normalization, this step is less affected by extreme values from potential anomalous nodes.

The normalized attributes are then projected into a unified d'-dimensional space by PCA:

$$\widehat{X} = \mathrm{PCA}(\widetilde{X}) \in \mathbb{R}^{n \times d'}. \tag{2}$$

To further reduce post-projection distribution shifts across graphs, RECAP performs coordinate-wise normalization:

$$\overline{x}_{ij} = \frac{\widehat{x}_{ij} - \mu_j}{\sigma_j + \epsilon}, \tag{3}$$

where μ_j and σ_j are the mean and standard deviation of the j-th projected coordinate. The aligned feature matrix $\overline{X} \in \mathbb{R}^{n \times d'}$ is used as the input for residual encoding. PCA and both normalizations are fitted independently on each graph using only its unlabeled node features. Hence target features participate in transductive preprocessing, but target labels, prompts, and optimization signals are never accessed; no target-fitted parameters are passed back to the shared model.

3.4 Residual Pattern Similarity Graph Construction

After feature alignment, RECAP constructs residual pattern representations to capture the relative differences between each node and its local neighborhood while reducing reliance on dataset-specific semantics. The key assumption is that domain variation is predominantly graph-smooth whereas relevant anomalies induce local ego-neighbor deviations, and that source and target residual-neighborhood distributions are sufficiently close. Thus, residual-pattern similarity can be used to organize nodes into multiple groups for subsequent anomaly scoring. These are conditional transfer assumptions rather than invariance to arbitrary shifts; Appendix B.1 gives a formal residual-stability argument, a discrepancy bound, and degradation cases.

Given the aligned feature matrix $\overline{X}$, let $\overline{X}_{[0]} = \overline{X}$. RECAP follows the ego-neighbor residual encoder in ARC (Liu et al., 2024) and performs L steps of parameter-free neighborhood propagation:

$$\overline{X}_{[l]} = \widetilde{A}\overline{X}_{[l-1]}, \quad \widetilde{A} = \widehat{D}^{-1/2}\widehat{A}\widehat{D}^{-1/2}, \quad \widehat{A} = A + I, \tag{4}$$

where $l \in \{1, \cdots, L\}$ and $\widehat{D}$ is the degree diagonal matrix of $\widehat{A}$. A shared Multi-Layer Perceptron (MLP) maps the initial and propagated features into the same representation space:

$$Z_{[l]} = \mathrm{MLP}(\overline{X}_{[l]}), \quad 0 \le l \le L. \tag{5}$$

The residual between each propagated representation and the initial representation is then computed and concatenated:

$$E = [R_{[1]} \| R_{[2]} \| \cdots \| R_{[L]}], \quad R_{[l]} = Z_{[l]} - Z_{[0]}, \quad 1 \le l \le L, \tag{6}$$

where $E \in \mathbb{R}^{n \times d}$ is the ego-neighbor residual embedding matrix, $\|$ denotes concatenation, and d is the concatenated embedding dimension.

RECAP then builds a residual similarity graph in this residual space. Directly computing all pairwise similarities costs $O(n^2 d)$, and repeatedly rebuilding a global KNN graph during training introduces both high cost and discrete structural changes. To avoid this, RECAP uses fixed candidate neighbor sets with soft weighting inside each candidate set. Specifically, to prevent randomly initialized MLP modules from affecting the initial neighbor selection, RECAP first constructs an initial residual representation $E^{(0)} = [e_1^{(0)}; e_2^{(0)}; \cdots; e_n^{(0)}] \in \mathbb{R}^{n \times d}$ from parameter-free neighborhood propagation. This $E^{(0)}$ is used only to determine candidate sets and is not used for the final edge-weight computation during community representation learning.

For a node pair (v_i, v_j), the initial residual similarity is

$$sim_{ij}^{(0)} = \frac{(e_i^{(0)})^T e_j^{(0)}}{\|e_i^{(0)}\| \, \|e_j^{(0)}\|}. \tag{7}$$

For each node v_i, RECAP selects the top-k most similar nodes as its fixed candidate neighbor set:

$$\mathcal{C}_i = \mathrm{KNN}_k\left(v_i; sim^{(0)}\right), \quad v_i \notin \mathcal{C}_i, \tag{8}$$

where k is the number of retained candidate neighbors. The candidate set remains unchanged during training and restricts each node's comparison range to its local residual-pattern references.

During training, RECAP no longer updates the discrete neighbor sets. Instead, it dynamically computes soft edge weights within each fixed candidate set using the current residual representation

E produced by the MLP. Let e_i and e_j denote the current residual representations of v_i and v_j. Their current residual similarity is

$$sim_{ij} = \frac{e_i^T e_j}{\|e_i\| \, \|e_j\|}. \tag{9}$$

For $v_j \in \mathcal{C}_i$, the similarity edge weight is defined by softmax normalization within the candidate set:

$$S_{ij} = \frac{\exp(sim_{ij}/\tau_s)}{\sum_{v_{j'} \in \mathcal{C}_i} \exp(sim_{ij'}/\tau_s)}, \quad v_j \in \mathcal{C}_i, \tag{10}$$

where τ_s controls the sharpness of the soft edge-weight distribution. If $v_j \notin \mathcal{C}_i$, then $S_{ij} = 0$. Since candidate neighbor relations are generally directed, RECAP symmetrizes the similarity matrix:

$$S \leftarrow \frac{S + S^T}{2}. \tag{11}$$

The resulting residual similarity graph is $G_s = (\mathcal{V}, \mathcal{E}_s, S)$, where $\mathcal{E}_s$ is induced by the fixed candidate neighbor sets and S is continuously updated during training. This design avoids repeatedly rebuilding discrete KNN graphs, reduces each-round similarity computation from $O(n^2 d)$ to $O(nkd)$, and provides stable local similarity relations for subsequent community optimization.

3.5 Residual Community Learning and Scoring

Given the Ego-neighbor residual embeddings $E = [e_1; \ldots; e_n] \in \mathbb{R}^{n \times d}$ and the residual similarity graph $G_s = (\mathcal{V}, \mathcal{E}_s, S)$, RECAP learns soft community assignments in the residual pattern space. The goal is to align the assignments with the residual similarity graph, so that nodes with similar local residual patterns form stable communities. Anomalies are then identified through residual-community adhesion mismatch, i.e., weak adhesion to community prototypes or inconsistency between a node's assignment and the community context of its neighborhood in G_s. RECAP outputs anomaly scores as well as community memberships, community sizes, prototypes, and decomposed diagnostic scores.

3.5.1 Soft Community Assignment

RECAP uses a lightweight assignment head to predict soft community assignments for nodes:

$$H = \mathrm{softmax}\left(\frac{EW}{\tau_c}\right), \quad \sum_{c=1}^{C} H_{ic} = 1, \tag{12}$$

where $H \in \mathbb{R}^{n \times C}$ is the soft assignment matrix, $W \in \mathbb{R}^{d \times C}$ is learnable, $\tau_c > 0$ is a temperature hyperparameter controlling the sharpness of the softmax assignment, and H_{ic} is the probability that node v_i belongs to community c. The residual similarity graph S provides local residual-pattern relations, and training encourages H to be consistent with this structure. This assignment head defines a cross-graph parametric assignment function $g_\theta : e_i \mapsto h_i$. Trained on residual similarity structures from source graphs, g_θ transfers the inductive bias of residual-pattern partitioning to unseen graphs. At test time, the model keeps g_θ fixed and estimates community prototypes solely from forward-pass results on the target graph, without updating any model parameters.

To learn structurally consistent and non-degenerate assignments, the framework optimizes

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{con}} + \lambda_{reg} \mathcal{L}_{reg}, \tag{13}$$

where $\lambda_{reg} \geq 0$. Details of the loss design are given in Appendix A. The community consistency loss is

$$\mathrm{L}_{\mathrm{con}} = \frac{\mathrm{Tr}\big(H^\top L_S H\big)}{\mathrm{Tr}(H^\top D_S H) + \epsilon}, \quad L_S = D_S - S, \quad D_S = \mathrm{diag}(S\mathbf{1}). \tag{14}$$

It encourages similar nodes in G_s to have similar assignments(Von Luxburg, 2007), and the normalized form supports transfer across graphs. The regularization term is

$$\mathcal{L}_{reg} = \mathcal{L}_{\mathrm{assign}} + \lambda_{\mathrm{usage}} \mathcal{L}_{\mathrm{usage}}. \tag{15}$$

Here, $\mathcal{L}_{\mathrm{assign}}$ constrains node-level assignments from being too flat or prematurely deterministic, while $\mathcal{L}_{\mathrm{usage}}$ constrains graph-level community usage to avoid both community collapse and mechanical uniform usage. After training, the assignment head directly produces soft community probabilities on unseen graphs for prototype estimation and anomaly scoring.

Table 1: Evaluation tracks. "Context" denotes target prompts or reference nodes.

Track	Train vs. target	Train labels	Target context/tuning	Deployment
Unsupervised OFO	Same	None	None	One model per target
Supervised OFA baselines	Different	Source labels	Method-dependent context; no tuning	One model transferred
RECAP-OFA	Different	None	None	One frozen model transferred

3.5.2 COMMUNITY-ADHESION MISMATCH SCORING

During inference, the model fixes all parameters and computes the target-graph residual embeddings $E = [e_1; \ldots; e_n] \in \mathbb{R}^{n\times d}$ and soft community assignments $H = [h_1; \ldots; h_n] \in \mathbb{R}^{n\times C}$. Following the view that anomalous nodes cannot be stably absorbed by the residual-community structure, RECAP decomposes residual-community adhesion mismatch into two scores: a prototype-level term measuring deviation from the prototypes implied by the soft assignment, and a context-level term measuring inconsistency with the community distribution of residual-similar ego-neighbors.

Given H, RECAP estimates each target-graph community prototype and computes the prototype-level adhesion deviation as

$$\mathbf{z}_c = \frac{\sum_{i=1}^{n} H_{ic}\mathbf{e}_i}{\sum_{i=1}^{n} H_{ic} + \epsilon}, \quad c = 1, \ldots, C, \qquad s_i^{\text{proto}} = \frac{1}{\tau_e}\sum_{c=1}^{C} H_{ic}\|\mathbf{e}_i - \mathbf{z}_c\|_2^2. \tag{16}$$

Here, s_i^{proto} measures the deviation from softly assigned prototypes, while the context-level term further checks whether the assignment is supported by residual-similar neighbors. The neighborhood community context of node v_i is computed as

$$\mathbf{q}_i = \frac{\sum_j S_{ij}\mathbf{h}_j}{\sum_j S_{ij} + \epsilon}, \quad \mathbf{q}_i \leftarrow \frac{\mathbf{q}_i}{\sum_{c=1}^{C} q_{ic} + \epsilon}. \tag{17}$$

The context-level score is the Jensen–Shannon divergence between the node assignment $\mathbf{h}_i$ and residual-neighbor context $\mathbf{q}_i$, with larger values indicating stronger assignment-context inconsistency:

$$s_i^{\text{ctx}} = \frac{1}{\log 2}\left[\frac{1}{2}\text{KL}(\mathbf{h}_i\|\mathbf{m}_i) + \frac{1}{2}\text{KL}(\mathbf{q}_i\|\mathbf{m}_i)\right], \quad \mathbf{m}_i = \frac{1}{2}(\mathbf{h}_i + \mathbf{q}_i). \tag{18}$$

Since the two terms have different numerical scales, RECAP standardizes them within each target graph and combines them as

$$s_i = \text{Norm}(s_i^{\text{proto}}) + \beta\,\text{Norm}(s_i^{\text{ctx}}), \tag{19}$$

where $\beta \geq 0$ controls the context contribution. Nodes are ranked by s_i, with larger values indicating stronger anomalousness. The two components support decomposed, model-intrinsic diagnostics; they do not by themselves provide human-semantic explanations. A controlled replacement of residual-space KNN with aligned-feature KNN in Appendix D.3 tests whether the context term is merely a second aggregation of the residual encoder.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets and Baselines. The experiments use 12 node-level GAD datasets from four domains: Cora, CiteSeer, PubMed (Sen et al., 2008), ACM (Tang et al., 2008), BlogCatalog, Flickr (Ding et al., 2019), Facebook (Xu et al., 2022), Weibo and Reddit, Amazon (McAuley & Leskovec, 2013), YelpChi (Rayana & Akoglu, 2015), and Questions. Full statistics are in Appendix C.1. We separate two questions: target-specific unsupervised OFO detection, compared with DOMINANT, AnomalyDAE, CoLA, ADA-GAD, and DiffGAD (Ding et al., 2019; Fan et al., 2020; Liu et al., 2022; He et al., 2024; Li et al., 2025); and cross-graph OFA generalization, compared with ARC, IA-GGAD, UNPrompt, AnomalyGFM-ZS, and OWLEYE (Liu et al., 2024; Zhang et al., 2025; Niu et al., 2024; Qiao et al., 2025a; Zheng et al., 2026b). Detailed method adaptations and evaluation populations are in Appendix C.2.

Table 2: Cross-graph OFA dataset-macro AUROC/AUPRC (%, mean over three seeds). All methods use identical source–target splits; their supervision and target inputs differ as disclosed.

Method	Source labels	Target context	Setting A	Setting B	Setting C
ARC	Yes	10 labeled normals	78.69/36.49	72.76/29.49	66.18/18.86
IA-GGAD	Yes	10 unlabeled references	77.22/35.51	72.84/30.74	63.03/17.74
OWLEYE	Yes	10 unlabeled patterns	76.04/35.67	71.79/28.66	60.33/18.01
UNPrompt	Yes	None	58.02/10.22	57.92/11.69	59.25/11.37
AnomalyGFM-ZS	Yes	None	53.07/8.02	49.64/8.43	51.03/5.57
RECAP-OFA	**No**	**None**	**74.65/27.04**	**67.75/21.98**	**67.31/17.50**

Table 3: Target-specific unsupervised OFO results on the same 12 targets (dataset-macro %, mean over three seeds). Each method trains separately on each target; this track is not directly comparable with OFA. Bold marks the best value in each metric.

Method	AUROC	AUPRC
DOMINANT	53.32	10.40
AnomalyDAE	56.24	9.97
CoLA	54.36	11.88
ADA-GAD	57.57	7.17
DiffGAD	55.03	10.61
RECAP-OFO	**71.08**	**23.61**

Source–target splits. The OFA evaluation uses three fixed source compositions selected without target labels. Setting A trains on PubMed, Flickr, Questions, and YelpChi and evaluates eight heterogeneous targets; Setting B replaces Flickr with Cora to test transfer without a social-network source; and Setting C transfers from four citation sources to five non-citation targets. All OFA methods use the same splits and three seeds. Appendix D.1 lists every graph and the per-setting standard deviations.

Metrics and Implementation. Following prior studies (Liu et al., 2024; Niu et al., 2024; Qiao et al., 2025a), AUROC and AUPRC are used as the main metrics, measuring overall normal-anomaly ranking ability and detection quality under severe class imbalance, respectively. The implementation uses Python 3.12.3, PyTorch 2.11.0+cu128, and PyG 2.7.0. Experiments are conducted on a Linux server with an Intel Xeon Platinum 8470Q CPU and an NVIDIA RTX PRO 6000 Blackwell Server Edition GPU with about 95 GB memory. The model is trained with Adam for 100 epochs by default, using a learning rate and weight decay of 5×10^{-5}. Each experiment is repeated over three random seeds. Input features are aligned to 32 dimensions via robust normalization and PCA. For residual community assignment, the number of communities, KNN size, and propagation hops are set 36, 64, and 4, respectively. The temperatures τ_s and τ_c are 0.3 and 0.3; λ_H and λ_{usage} are both 0.1; and the inference coefficient β is 0.02.

4.2 Experimental Results

Performance Comparison. Tables 2 and 3 report the two protocol-matched comparisons separately. In OFO, RECAP-OFO outperforms all five unsupervised baselines under the same target-specific protocol. In OFA, RECAP remains competitive despite using neither source labels nor target context. Methods with stronger supervision lead Settings A/B; RECAP nevertheless exceeds the target-context-free UNPrompt and AnomalyGFM-ZS across all settings and obtains the highest AUROC in the citation-only Setting C without source labels or target-side inputs. The decline from A to B/C shows sensitivity to source composition rather than invariance to arbitrary domain shifts. Full standard deviations, per-target results, and evaluation populations are in Appendix D.1.

Efficiency and Robustness. Empirical timing supports the intended frozen-target deployment: RECAP-OFA requires 20.753 seconds for one-time source training and 0.023 seconds per target inference on average, with no target-side optimization. Across seed pairs, anomaly-score Spearman correlations are 0.878, 0.735, and 0.663 in Settings A, B, and C, respectively; rankings are more stable than permutation-arbitrary community identities, but both deteriorate under the citation-only shift. Appendix D.1 reports the complete timing, seed-stability, and million-node target results.

Ablation Study. The matched five-seed factorial study in Table 4 separates the ARC-derived residual encoder from RECAP's community framework. With KMeans fixed, residual encoding improves AUROC/AUPRC by $7.53/4.50$ points; with residual features fixed, RECAP's community learning/scoring improves them by $6.66/8.36$ points. The positive $13.56/11.48$ point interaction supports a *residual-aware community-learning* contribution rather than two independent additive modules. Full module removals and a controlled context-neighborhood replacement are in Appendix D.3.

Representation	KMeans	RECAP scoring
Non-residual hops	$60.29/13.73$	$53.39/10.60$
Residual hops	$67.82/18.23$	$\mathbf{74.48/26.58}$

Table 4: Matched 2×2 AUROC/AUPRC (%).

Hyperparameter Sensitivity Analysis. Figure 3 shows stable performance across a broad τ_s range, a peak at $L = 3$–4, and saturation as k increases; C has a non-monotonic effect, with the best region around $C = 32$–36 and $k = 64$–96. We therefore select $C = 36$ and $k = 64$.

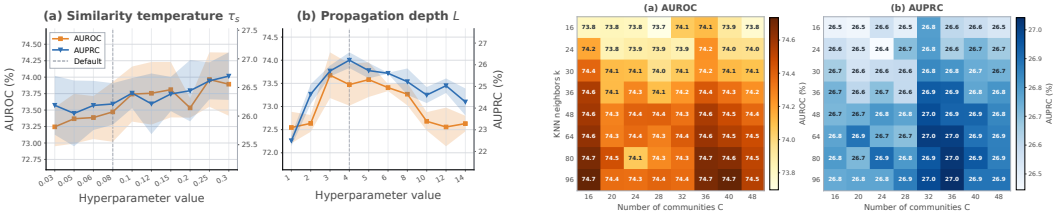
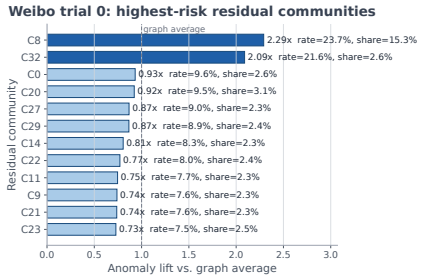


Figure 3: RECAP hyperparameter sensitivity. Left: one-dimensional sweeps over τ_s and L. Right: two-dimensional sweep over k and C.

Anomaly Diagnosis. Beyond anomaly ranking, the soft residual-community layer provides model-intrinsic diagnostics. On Weibo, high-risk communities reach anomaly rates of 18.4%–22.9% ($1.79\times$–$2.22\times$ the graph average), while the full score achieves 34.27% AUPRC versus $30.36\%/31.12\%$ for its component scores. Reports summarize community assignment, prototype distance, neighborhood support, and score percentiles; the top-5% nodes achieve a $3.64\times$ anomaly lift (Fig. 4). These diagnostics are faithful to the scoring computation but are not validated human-semantic explanations.



Node 1454 (Label: 1)
Diagnosis: High-risk community prototype outlier

This node is flagged as anomalous due to its assignment to a high-risk community and large distance to the community prototype. It assigns to C8 with 100% probability. C8 is high-risk (soft share 15.3%, anomaly rate 23.7%, lift 2.29×). Its distance to the assigned prototype is 134.06 (100.0th percentile). Neighbors only partially support C8 (41.4% neighbor mass), with minor support for C29 and C20, providing a moderate context mismatch. Thus, the node exhibits both prototype deviation and context inconsistency. Adhesion percentile = 100.0%, context percentile = 98.8%, beta-weighted context contribution = 0.096.

Figure 4: Anomaly diagnosis on Weibo. Left: anomaly lift of high-risk residual communities. Right: node-level diagnostic evidence for a high-scoring anomaly.

5 Conclusion and Limitations

This paper presented RECAP, a label-free generalist graph anomaly detector that transfers residual-community structure without source labels, target prompts, or target fine-tuning. Across cross-graph settings, RECAP achieved competitive anomaly ranking while providing decomposed community-based diagnostics.

The method assumes compatible residual-neighborhood structure and can degrade under severe shifts, as in the citation-only split. Approximate KNN enables million-node inference, but the unadapted checkpoints were not predictively reliable there. Evaluation remains limited to static, attributed, node-level graphs, and the diagnostics are model-intrinsic rather than human-semantic; high-stakes deployment therefore requires domain-specific calibration, expert validation, and fairness/privacy assessment.

AI USE STATEMENT

We used generative AI tools only for LaTeX editing, language editing, and consistency checks. We did not use generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, formulate mathematical claims, assist with mathematical proofs, propose or refine hypotheses, design or modify the methodology or experiments, implement methods, translate content, clean or reformat datasets, perform qualitative or thematic analysis, or interpret results. We also did not use generative AI tools to create or modify scientific figures or images, suggest experimental parameters, create or edit software code, create research artifacts, draft paper content, summarize or analyze the literature, identify research topics or gaps, source information, improve readability beyond the language editing described above, identify literature, format references, suggest the paper structure, or propose the title or keywords. We reviewed all AI-assisted changes and take responsibility for the final content, claims, and artifacts of this work.

ETHICS STATEMENT

This work uses existing graph anomaly-detection benchmark datasets and does not collect new human-subject data. Because anomaly scores could influence high-stakes decisions if deployed without validation, we state the limitations of transfer, calibration, fairness, privacy, and human-semantic interpretation in the main text. Deployment should include domain-specific validation and human oversight.

REPRODUCIBILITY STATEMENT

The paper specifies the model, losses, optimization settings, source–target splits, evaluation protocols, runtime measurements, and seed-stability analyses. The supplementary implementation and dataset-preparation instructions provide the remaining details needed to reproduce the experiments, including preprocessing, checkpoint construction, and metric computation. The anonymous code repository is available at `https://anonymous.4open.science/r/recap-8386/README.md`.

REFERENCES

Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery*, 29(3):626–688, 2015.

Yuxuan Cao, Jiarong Xu, Chen Zhao, Jiaan Wang, Carl Yang, Chunping Wang, and Yang Yang. How to use graph data in the wild to help graph anomaly detection? In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 61–72, 2025.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

Jiujiu Chen, Weijun Zeng, Shaofeng Hu, Sihong Xie, and Hui Xiong. Gfm4ga: Graph foundation model for group anomaly detection. *arXiv preprint arXiv:2601.10193*, 2026.

Kaize Ding, Jundong Li, Rohit Bhanushali, and Huan Liu. Deep anomaly detection on attributed networks. In *Proceedings of the 2019 SIAM international conference on data mining*, pp. 594–602. SIAM, 2019.

Haoyi Fan, Fengbin Zhang, and Zuoyong Li. Anomalydae: Dual autoencoder for anomaly detection on attributed networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5685–5689. IEEE, 2020.

Junwei He, Qianqian Xu, Yangbangyan Jiang, Zitai Wang, and Qingming Huang. Ada-gad: Anomaly-denoised autoencoders for graph anomaly detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 8481–8489, 2024.

Haiyang Jiang, Tong Chen, Xinyi Gao, Guansong Pang, Quoc Viet Hung Nguyen, and Hongzhi Yin. Evolutionary router feature generation for zero-shot graph anomaly detection with mixture-of-experts. *arXiv preprint arXiv:2602.11622*, 2026.

Ming Jin, Yixin Liu, Yu Zheng, Lianhua Chi, Yuan-Fang Li, and Shirui Pan. Anemone: Graph anomaly detection with multi-scale contrastive learning. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pp. 3122–3126, 2021.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Jinghan Li, Yuan Gao, Jinda Lu, Junfeng Fang, Congcong Wen, Hui Lin, and Xiang Wang. Diffgad: A diffusion-based unsupervised graph anomaly detector. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=AhcYq4CnfF.

Jundong Li, Harsh Dani, Xia Hu, and Huan Liu. Radar: Residual analysis for anomaly detection in attributed networks. In *Ijcai*, volume 17, pp. 2152–2158, 2017.

Yixin Liu, Zhao Li, Shirui Pan, Chen Gong, Chuan Zhou, and George Karypis. Anomaly detection on attributed networks via contrastive self-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2378–2392, 2022.

Yixin Liu, Shiyuan Li, Yu Zheng, Qingfeng Chen, Chengqi Zhang, and Shirui Pan. Arc: A generalist graph anomaly detector with in-context learning. *Advances in Neural Information Processing Systems*, 37:50772–50804, 2024.

Yixin Liu, Shiyuan Li, Yu Zheng, Qingfeng Chen, Chengqi Zhang, Philip S Yu, and Shirui Pan. From few-shot to zero-shot: Towards generalist graph anomaly detection. *arXiv preprint arXiv:2602.18793*, 2026.

Xiaoxiao Ma, Jia Wu, Shan Xue, Jian Yang, Chuan Zhou, Quan Z Sheng, Hui Xiong, and Leman Akoglu. A comprehensive survey on graph anomaly detection with deep learning. *IEEE transactions on knowledge and data engineering*, 35(12):12012–12038, 2021.

Julian John McAuley and Jure Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*, pp. 897–908, 2013.

Chaoxi Niu, Hezhe Qiao, Changlu Chen, Ling Chen, and Guansong Pang. Zero-shot generalist graph anomaly detection with unified neighborhood prompts. *arXiv preprint arXiv:2410.14886*, 2024.

Hezhe Qiao, Chaoxi Niu, Ling Chen, and Guansong Pang. Anomalygfm: Graph foundation model for zero/few-shot anomaly detection. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 2326–2337, 2025a.

Hezhe Qiao, Hanghang Tong, Bo An, Irwin King, Charu Aggarwal, and Guansong Pang. Deep graph anomaly detection: A survey and new perspectives. *IEEE Transactions on Knowledge and Data Engineering*, 2025b.

Shebuti Rayana and Leman Akoglu. Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*, pp. 985–994, 2015.

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

Jianheng Tang, Jiajin Li, Ziqi Gao, and Jia Li. Rethinking graph neural networks for anomaly detection. In *International conference on machine learning*, pp. 21076–21089. PMLR, 2022.

Jianheng Tang, Fengrui Hua, Ziqi Gao, Peilin Zhao, and Jia Li. Gadbench: Revisiting and benchmarking supervised graph anomaly detection. *Advances in Neural Information Processing Systems*, 36:29628–29653, 2023.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 990–998, 2008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, Yoshua Bengio, and Joan Bruna. Graph attention networks. In *International conference on learning representations*, volume 6. Ithaca, 2018.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Hewen Wang, Renchi Yang, and Jieming Shi. Anomaly detection in financial transactions via graph-based feature aggregations. In *International Conference on Big Data Analytics and Knowledge Discovery*, pp. 64–79. Springer, 2023.

Yulei Wu, Hong-Ning Dai, and Haina Tang. Graph neural networks for anomaly detection in industrial internet of things. *IEEE Internet of Things Journal*, 9(12):9214–9231, 2022.

Zhiming Xu, Xiao Huang, Yue Zhao, Yushun Dong, and Jundong Li. Contrastive attributed network anomaly detection with data augmentation. In *Pacific-Asia conference on knowledge discovery and data mining*, pp. 444–457. Springer, 2022.

Yang Yang, Yuhong Xu, Yizhou Sun, Yuxiao Dong, Fei Wu, and Yueting Zhuang. Mining fraudsters and fraudulent strategies in large-scale mobile social networks. *IEEE Transactions on Knowledge and Data Engineering*, 33(1):169–179, 2021.

Xiong Zhang, Zhenli He, Changlong Fu, and Cheng Xie. Ia-ggad: Zero-shot generalist graph anomaly detection via invariant and affinity learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Xiong Zhang, Hong Peng, Changlong Fu, Xin Jin, Yun Yang, and Cheng Xie. Ta-ggad: Testing-time adaptive graph model for generalist graph anomaly detection. *arXiv preprint arXiv:2603.09349*, 2026.

Xinyu Zhao, Qingyun Sun, Jiayi Luo, Xingcheng Fu, and Jianxin Li. Zero-shot generalizable graph anomaly detection with mixture of riemannian experts. *arXiv preprint arXiv:2602.06859*, 2026.

Jialun Zheng, Jie Liu, Jiannong Cao, Xiao Wang, Hanchen Yang, and Yankai Chen. Dp-dgad: A generalist dynamic graph anomaly detector with dynamic prototypes. In *Proceedings of the ACM Web Conference 2026*, pp. 857–868, 2026a.

Lecheng Zheng, Dongqi Fu, Zihao Li, and Jingrui He. Owleye: Zero-shot learner for cross-domain graph data anomaly detection. *arXiv preprint arXiv:2601.19102*, 2026b.

A Loss Function Design

This section explains the design motivation of the community-aware training objective in RECAP. Given the residual pattern representation matrix $E = [e_1; \ldots; e_n] \in \mathbb{R}^{n \times d}$, the residual similarity graph $G_s = (\mathcal{V}, \mathcal{E}_s, S)$, and the soft community assignment matrix $H = [h_1; \ldots; h_n] \in \mathbb{R}^{n \times C}$, where $h_i \in \mathbb{R}^C$ denotes the soft assignment probabilities of node v_i over C communities, satisfying $H_{ic} \geq 0$ and $\sum_{c=1}^{C} H_{ic} = 1$. RECAP aims to learn a mapping from residual-pattern representations to soft community assignments, encouraging nodes with similar residual patterns to have similar assignments while avoiding degenerate solutions where all nodes collapse into one community or are uniformly assigned to all communities. In addition, this section provides an optional protection loss $\mathcal{L}_{emb}$ to prevent residual representation collapse. Experiments show that this protection mechanism is not triggered in most cases, but we briefly introduce it here as a fallback. Finally, RECAP adopts the following composite loss, we next explain each loss term.

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{con}} + \lambda_{reg}\mathcal{L}_{reg}. \tag{20}$$

Community consistency loss $\mathcal{L}_{\text{con}}$ The residual similarity graph S captures local residual pattern similarity between nodes. If S_{ij} is large, nodes v_i and v_j have similar local residual patterns, and thus should have similar community assignments, i.e., $h_i \approx h_j$. Based on this idea, the Dirichlet energy on the graph can be defined as:

$$\mathcal{L}_{\text{con}} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}\|h_i - h_j\|_2^2. \tag{21}$$

This equation indicates that if two nodes have a large edge weight S_{ij} in the residual similarity graph but also have a large assignment difference $\|h_i - h_j\|_2^2$, they receive a stronger penalty. Let

$$D_S = \text{diag}(S\mathbf{1}), \qquad L_S = D_S - S, \tag{22}$$

where D_S is the degree matrix of the similarity graph and L_S is the graph Laplacian. When S is symmetric and nonnegative, Eq. 21 is equivalent to:

$$\mathcal{L}_{\text{con}} = \text{Tr}(H^\top L_S H). \tag{23}$$

Lemma 1. *Let $S \in \mathbb{R}^{n\times n}$ be a symmetric nonnegative similarity matrix, $D_S = \text{diag}(S\mathbf{1})$, and $L_S = D_S - S$ be the corresponding graph Laplacian. For any $H = [h_1; \ldots; h_n] \in \mathbb{R}^{n\times C}$, the following equality holds:*

$$\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}\|h_i - h_j\|_2^2 = \text{Tr}(H^\top L_S H). \tag{24}$$

Proof. Expanding the left-hand side gives

$$\begin{aligned}\frac{1}{2}\sum_{i,j} S_{ij}\|h_i - h_j\|_2^2 &= \frac{1}{2}\sum_{i,j} S_{ij}\left(\|h_i\|_2^2 + \|h_j\|_2^2 - 2h_i^\top h_j\right)\\ &= \frac{1}{2}\sum_{i,j} S_{ij}\|h_i\|_2^2 + \frac{1}{2}\sum_{i,j} S_{ij}\|h_j\|_2^2 - \sum_{i,j} S_{ij}h_i^\top h_j.\end{aligned} \tag{25}$$

Since S is symmetric and $d_i = \sum_j S_{ij}$, the first two terms satisfy

$$\frac{1}{2}\sum_{i,j} S_{ij}\|h_i\|_2^2 + \frac{1}{2}\sum_{i,j} S_{ij}\|h_j\|_2^2 = \sum_i d_i\|h_i\|_2^2. \tag{26}$$

Therefore,

$$\begin{aligned}\frac{1}{2}\sum_{i,j} S_{ij}\|h_i - h_j\|_2^2 &= \sum_i d_i\|h_i\|_2^2 - \sum_{i,j} S_{ij}h_i^\top h_j\\ &= \text{Tr}(H^\top D_S H) - \text{Tr}(H^\top S H)\\ &= \text{Tr}(H^\top (D_S - S)H)\\ &= \text{Tr}(H^\top L_S H).\end{aligned} \tag{27}$$

This completes the proof. □

Thus, the community consistency loss is essentially a smoothness constraint on soft community assignments over the residual similarity graph. It makes nodes with similar residual patterns have similar community memberships, aligning soft assignments with the residual similarity graph structure. In cross-graph training, different graphs may have different numbers of nodes, edges, and similarity-weight scales. To reduce the effect of graph scale on the loss value, we use the normalized form:

$$\mathcal{L}_{\text{con}} = \frac{\text{Tr}(H^\top L_S H)}{\text{Tr}(H^\top D_S H) + \epsilon}. \tag{28}$$

Note that using $\mathcal{L}_{\text{con}}$ alone admits degenerate solutions. If all nodes have identical community assignments, i.e.,

$$h_1 = h_2 = \cdots = h_n, \tag{29}$$

then for any node pair (i, j), $\|h_i - h_j\|_2^2 = 0$, and thus

$$\mathcal{L}_{\text{con}} = 0. \tag{30}$$

Therefore, the community consistency loss only ensures that similar nodes have similar assignments, but cannot alone guarantee meaningful non-degenerate community partitioning.

Community assignment regularization $\mathcal{L}_{reg}$ To avoid degenerate solutions in label-free community learning, RECAP introduces regularization on the soft community assignment matrix $H \in \mathbb{R}^{n\times C}$. This regularization does not rely on anomaly labels, but constrains community learning from node-level assignment certainty and global community usage coverage:

$$\mathcal{L}_{reg} = \mathcal{L}_{\text{assign}} + \lambda_{\text{usage}}\mathcal{L}_{\text{usage}}. \tag{31}$$

For node v_i, its soft community assignment is $h_i \in \mathbb{R}^C$. The node-level assignment entropy is defined as:

$$\mathcal{H}(h_i) = -\sum_{c=1}^{C} H_{ic}\log(H_{ic}+\epsilon). \tag{32}$$

Furthermore, the average assignment entropy over all nodes is:

$$\mathcal{H}_{\text{assign}} = \frac{1}{n}\sum_{i=1}^{n}\mathcal{H}(h_i). \tag{33}$$

Since the maximum entropy is $\log C$, we use the normalized average assignment entropy:

$$\tilde{\mathcal{H}}_{\text{assign}} = \frac{\mathcal{H}_{\text{assign}}}{\log C}. \tag{34}$$

In label-free community learning, overly high node assignment entropy makes each node nearly uniformly belong to all communities, causing community memberships to lack discriminability. Overly low entropy may cause the model to form overly sharp assignments early in training, weakening the ability of soft communities to represent uncertain residual patterns. Therefore, RECAP does not directly minimize node assignment entropy, but constrains it within a preset interval $[\alpha_l, \alpha_u]$:

$$\mathcal{L}_{\text{assign}} = \log C\left[\max(0, \tilde{\mathcal{H}}_{\text{assign}} - \alpha_u) + \max(0, \alpha_l - \tilde{\mathcal{H}}_{\text{assign}})\right]. \tag{35}$$

When $\tilde{\mathcal{H}}_{\text{assign}}$ lies within $[\alpha_l, \alpha_u]$, this term is zero. When node assignments are globally too uniform or too sharp, this term imposes a penalty. This design keeps community assignments moderately certain while avoiding premature hard-assignment degeneration early in training.

Constraining individual node assignment entropy alone still cannot ensure sufficient use of different communities. For example, the model may assign most nodes to only a few communities. Even if each node has a relatively confident assignment, the overall community structure still collapses. Therefore, RECAP further constrains the global community usage distribution. First, define the average community usage probability:

$$\pi_c = \frac{1}{n}\sum_{i=1}^{n} H_{ic}, \quad c = 1, \ldots, C, \tag{36}$$

where π_c denotes the average usage degree of the c-th community on the current graph. The corresponding community usage entropy is:

$$\mathcal{H}_{\text{usage}} = -\sum_{c=1}^{C}\pi_c\log(\pi_c+\epsilon). \tag{37}$$

Its normalized form is:

$$\tilde{\mathcal{H}}_{\text{usage}} = \frac{\mathcal{H}_{\text{usage}}}{\log C}. \tag{38}$$

To avoid community usage collapse while preventing all communities from being used too uniformly, RECAP constrains the normalized community usage entropy within $[\beta_l, \beta_u]$:

$$\mathcal{L}_{\text{usage}} = \log C\left[\max(0, \tilde{\mathcal{H}}_{\text{usage}} - \beta_u) + \max(0, \beta_l - \tilde{\mathcal{H}}_{\text{usage}})\right]. \tag{39}$$

When $\tilde{\mathcal{H}}_{\text{usage}}$ is too low, the model mainly uses a few communities, indicating a risk of community collapse. When it is too high, all communities are used too uniformly, which may weaken the ability

of communities to capture real residual pattern differences. Therefore, $\mathcal{L}_{\text{usage}}$ balances community coverage and pattern discrimination through interval constraints.

$\mathcal{L}_{\text{assign}}$ and $\mathcal{L}_{\text{usage}}$ constrain soft community learning from local node assignment and global community usage, respectively. The former ensures moderate certainty of each node's community membership, while the latter ensures reasonable use of different communities within the graph. Together, they enable RECAP to learn a stable residual pattern community assignment function without anomaly labels, avoiding overly flat soft assignments, premature hard assignments, and community usage collapse.

Optional Residual Variance Stabilization $\mathcal{L}_{\text{emb}}$ The training objective in the main text mainly consists of the community consistency loss on the residual similarity graph and the soft community assignment regularization. In addition, we also examine an optional residual representation variance lower-bound stabilization term, used to avoid dimension collapse in residual pattern representations output by the MLP under extreme cases. This term is not a necessary component of the main method. In the final experiments, this term is disabled by default, i.e., its weight is set to $\lambda_{\text{emb}} = 0$, so the main-text training objective does not include it.

Specifically, the soft community assignment matrix H is obtained from the residual pattern representation E through the assignment head:

$$H = \operatorname{softmax}\left(\frac{EW}{\tau_c}\right). \tag{40}$$

If the residual representation collapses, e.g., $e_1 = e_2 = \cdots = e_n$, then residual pattern differences between nodes disappear, and the residual similarity graph and soft community assignments also lose their discriminative basis. To characterize this degeneration risk, the following variance lower-bound regularization can be introduced:

$$\mathcal{L}_{\text{emb}} = \frac{1}{d}\sum_{j=1}^{d} \max\left(0, \gamma - \operatorname{Std}(E_{:j})\right), \tag{41}$$

where $\gamma > 0$ is the standard deviation lower bound, and $E_{:j}$ denotes the values of the j-th residual representation dimension over all nodes. This term only imposes a penalty when the standard deviation of a dimension is lower than γ. When all dimensions satisfy $\operatorname{Std}(E_{:j}) \geq \gamma$, we have $\mathcal{L}_{\text{emb}} = 0$.

In the final experimental setting, we set the weight of this term to $\lambda_{\text{emb}} = 0$. Experiments show that the community consistency learning and soft community assignment regularization in the main text can already stably train the residual pattern assignment function, so this optional stabilization term does not affect the main experimental results reported in this paper.

B Transfer Conditions and Task Extension

B.1 Conditional Transfer Analysis

RECAP does not claim domain invariance under arbitrary graph shifts. Its transfer argument uses four conditions: (A1) domain-specific variation is predominantly graph-smooth while relevant anomalies create local ego-neighbor deviations; (A2) the source and target residual-neighborhood pair distributions are close; (A3) the shared encoder and assignment head are bounded and Lipschitz; and (A4) target communities have sufficient mass for prototype estimation. These conditions require neither shared raw features nor identical community identities, but they can fail for purely semantic/global anomalies, severe residual-pattern shifts, or very small target communities.

Residual stability. Let $\widetilde{A} = U\Lambda U^\top$ be the normalized propagation operator and let a graph-smooth, domain-specific component X_{dom} be supported on eigenvalues $\lambda \in [1-\rho, 1]$. For a K_ϕ-Lipschitz shared encoder ϕ and propagation order l,

$$\left\| \phi(\widetilde{A}^l X_{\text{dom}}) - \phi(X_{\text{dom}}) \right\|_F \leq K_\phi l \rho \left\| X_{\text{dom}} \right\|_F, \tag{42}$$

because $|1-\lambda^l| \le l\rho$ on this interval. Thus, the residual suppresses graph-smooth domain content while retaining local deviations. Multiple orders cover several scales, but Eq. 42 does not remove non-smooth or global shifts.

Residual-neighborhood discrepancy. Let Q_d^E denote the distribution of residual-neighbor pairs in domain d, and define the pairwise consistency risk

$$\mathcal{R}_d(g) = \mathbb{E}_{(e,e')\sim Q_d^E}\left[\|g(e)-g(e')\|_2^2\right]. \tag{43}$$

If $\|g(e)\|_2 \le B$ and g is L_g-Lipschitz, Kantorovich–Rubinstein duality gives

$$|\mathcal{R}_T(g) - \mathcal{R}_S(g)| \le 4BL_g\, W_1(Q_S^E, Q_T^E), \tag{44}$$

up to the standard finite-sample term for empirical source risk. Consequently, low source consistency risk transfers only when the residual-neighborhood discrepancy is small. This is a conditional guarantee, not a proof of generalization under arbitrary shifts. The decreasing community stability from Settings A/B to the citation-only Setting C (Table 11) empirically exposes this boundary.

B.2 Extension to Edge-level Detection

RECAP can in principle score edges by replacing node residuals with edge residuals. For an undirected edge (u, v), an order-invariant representation is

$$r_{uv} = \left[\frac{r_u + r_v}{2} \,\middle\|\, |r_u - r_v| \,\middle\|\, x_{uv}\right], \tag{45}$$

where r_u, r_v are endpoint residuals and x_{uv} contains optional edge attributes. Directed edges can use an ordered source–destination concatenation. Edges then become nodes of a line graph or an edge-similarity graph, with adjacency defined by shared endpoints or residual similarity. The same soft community assignment and adhesion/context scoring can rank anomalous edges and group them by residual-similar interaction patterns. This extension preserves label-free source training and frozen target inference, but it is not evaluated here; practical validation must address edge-neighborhood definitions, directionality, memory growth with $|\mathcal{E}|$, and incident-edge ambiguity.

C Supplementary Experimental Setup

C.1 Dataset Details

The node-level graph anomaly detection datasets used in the experiments cover citation, social, co-review/e-commerce, and question-answering/interaction networks. These datasets differ substantially in node semantics, edge relations, attribute dimensionality, anomaly ratios, and annotation protocols. Table 5 reports the basic statistics of all datasets.

Citation Networks Cora, CiteSeer, PubMed (Sen et al., 2008), and ACM (Tang et al., 2008) are commonly used citation network datasets in graph anomaly detection. Nodes typically represent papers, edges represent citation relations or relatedness between papers, and node attributes are obtained by vectorizing metadata. Since the original citation networks usually do not contain explicit anomaly labels, related studies generally construct anomalous nodes by perturbing node attributes or graph structures. This type of dataset is mainly used to evaluate the detection capability of models under high-dimensional features and academic relational structures.

Social Networks BlogCatalog, Flickr (Ding et al., 2019), Facebook (Xu et al., 2022), Weibo, Reddit, and Questions are used to represent social relations or interaction behaviors among users. Nodes typically correspond to users, edges represent following, friendship, interaction, or co-activity relations, and node attributes are derived from user-generated content, tags, and behavioral statistics. This type of dataset contains both artificially injected anomalies and real anomalies as annotation sources, enabling the evaluation of model robustness under complex social structures and real anomalous behaviors.

Table 5: Statistics of experimental datasets.

Dataset	#Nodes	#Edges	#Features	Avg. Degree	#Anomaly	%Anomaly
Citation networks with injected anomalies						
Cora	2,708	5,429	1,433	3.90	150	5.53
CiteSeer	3,327	4,732	3,703	2.77	150	4.50
ACM	16,484	71,980	8,337	8.73	597	3.62
PubMed	19,717	44,338	500	4.50	600	3.04
Social networks with injected anomalies						
BlogCatalog	5,196	171,743	8,189	66.11	300	5.77
Flickr	7,575	239,738	12,047	63.30	450	5.94
Social networks with real anomalies						
Facebook	1,081	55,104	576	50.97	25	2.31
Weibo	8,405	407,963	400	48.53	868	10.30
Reddit	10,984	168,016	64	15.30	366	3.33
Questions	48,921	153,540	301	3.13	1,460	2.98
Co-review networks with real anomalies						
Amazon	10,224	175,608	25	17.18	693	6.78
YelpChi	23,831	49,315	32	2.07	1,217	5.10

Co-review/E-commerce Networks Amazon (McAuley & Leskovec, 2013) and YelpChi (Rayana & Akoglu, 2015) are commonly used datasets for review fraud detection. Nodes typically represent users, reviews, or businesses, edges represent co-review, similar behavior, or interaction relations, and anomaly labels correspond to suspicious reviews, fraudulent users, or anomalous behavioral entities. Anomalies in this type of dataset are often associated with coordinated behaviors, fake reviews, and local relational patterns, making them suitable for evaluating the ability of models to identify behavioral pattern anomalies and local structural anomalies.

C.2 Description of Baseline Methods

We reproduce each method under its native supervision and deployment protocol and disclose the evaluation population in Table 6. OFO and OFA therefore remain separate tracks rather than entries in one overall ranking. Wild-GAD (Cao et al., 2025) and GAD-MoRE (Zhao et al., 2026) are discussed but not reproduced because a complete public implementation or required artifacts were unavailable.

Table 6: Supervision, target inputs, and evaluation populations.

Track/method	Training labels	Target input	Evaluation population
Supervised OFO (GCN/GAT/BWGNN/XGBGraph)	Target labels	None	Stratified held-out 40%
Unsupervised OFO (six baselines; RECAP-OFO)	None	None	Full graph
OFA ARC	Source labels	10 labeled-normal contexts	Excluding contexts
OFA IA-GGAD	Source labels	10 unlabeled references	Excluding references
OFA UNPrompt/AnomalyGFM-ZS	Source labels	None	Full target graph
OFA OWLEYE	Source labels	10 unlabeled patterns	Full target graph
OFA RECAP	None	None	Full target graph

Supervised OFO GAD This category of methods usually formulates graph anomaly detection as a node classification or anomalous node identification task, and uses node labels for supervision during training. The experiments select GCN (Kipf & Welling, 2016), GAT (Veličković et al., 2018), BWGNN (Tang et al., 2022), and XGBGraph (Tang et al., 2023) as representative methods. GCN and GAT are classic graph neural network models and can serve as basic comparisons for supervised graph representation learning; BWGNN and XGBGraph are supervised methods designed for anomaly detection tasks or directly applicable to anomaly identification, and can learn anomaly decision boundaries under labeled settings. This category of methods is used to measure the performance gap between RECAP under the fully label-free setting and methods that rely on label supervision.

Supervised OFA GAD This category targets one-for-all generalization and uses source anomaly labels, while target-side inputs differ by method. The experiments include ARC (Liu et al., 2024),

UNPrompt (Niu et al., 2024), AnomalyGFM-ZS (Qiao et al., 2025a), IA-GGAD (Zhang et al., 2025), and OWLEYE (Zheng et al., 2026b). ARC uses labeled-normal target contexts; IA-GGAD uses unlabeled internal target references; OWLEYE extracts unlabeled target patterns; UNPrompt and AnomalyGFM-ZS use no target context in our zero-shot reproduction. These methods are compared with RECAP under identical source–target splits, but their additional supervision and target inputs are retained and disclosed.

Unsupervised OFO GAD This category learns one detector per target graph without node labels. We include DOMINANT (Ding et al., 2019), AnomalyDAE (Fan et al., 2020), CoLA (Liu et al., 2022), ADA-GAD (He et al., 2024), DiffGAD (Li et al., 2025). RECAP-OFO is trained and evaluated under the same full-graph label-free protocol solely to isolate target-specific optimization from cross-graph transfer; it is not the main deployment setting of RECAP.

D Supplementary Experimental Results

D.1 Protocol-Matched Results and Source Shifts

Table 7 changes the source composition without target labels, prompts, or fine-tuning. Settings A/B test multi-domain and leave-social-domain-out transfer; Setting C uses citation-only sources and is the largest evaluated source shift.

Table 7: OFA source–target splits.

Setting	Source graphs	Target graphs
A	PubMed, Flickr, Questions, YelpChi	Cora, CiteSeer, ACM, BlogCatalog, Facebook, Weibo, Reddit, Amazon
B	PubMed, Cora, Questions, YelpChi	Flickr, BlogCatalog, Facebook, Weibo, Reddit
C	PubMed, Cora, CiteSeer, ACM	BlogCatalog, Flickr, Reddit, Amazon, Questions

Table 8: Protocol-matched macro AUROC/AUPRC (%, mean±population standard deviation over three seeds).

Method	OFO-12	OFA-A	OFA-B	OFA-C
DOMINANT	53.32±0.18/10.40±0.01	–	–	–
AnomalyDAE	56.24±0.33/9.97±0.11	–	–	–
CoLA	54.36±0.19/11.88±0.24	–	–	–
ADA-GAD	57.57±0.75/7.17±0.06	–	–	–
DiffGAD	55.03±2.17/10.61±0.24	–	–	–
RECAP-OFO	71.08±0.35/23.61±0.26	–	–	–
ARC	–	78.69±0.03/36.49±0.69	72.76±0.23/29.49±0.25	66.18±0.74/18.86±0.47
IA-GGAD	–	77.22±1.16/35.51±2.12	72.84±0.69/30.74±0.22	63.03±0.44/17.74±0.16
UNPrompt	–	58.02±1.41/10.22±1.42	57.92±2.13/11.69±1.21	59.25±1.49/11.37±0.41
AnomalyGFM-ZS	–	53.07±0.30/8.02±0.53	49.64±2.22/8.43±0.61	51.03±0.79/5.57±0.12
OWLEYE	–	76.04±0.05/35.67±0.52	71.79±0.20/28.66±0.22	60.33±0.68/18.01±0.09
RECAP-OFA	–	74.65±0.23/27.04±0.34	67.75±0.22/21.98±0.32	**67.31**±0.43/17.50±0.09

For completeness, we adapt DiffGAD to source-only OFA by fitting on the same sources and freezing all parameters before target evaluation. This is a protocol-aligned adaptation, not an author-released OFA variant. Table 9 reports the result; it does not replace the native OFO comparison in Table 8.

Table 9: Source-only adaptations: dataset-macro AUROC/AUPRC (%).

Method	Setting A	Setting B	Setting C
DiffGAD-OFA-adapted	66.94/21.11	59.61/18.47	69.61/17.99
RECAP-OFA	**74.65**/27.04	**67.75/21.98**	67.31/17.50

D.2 Efficiency, Scalability, and Seed Stability

The theoretical training cost after fixed candidate construction is $O(Lmp + nDC + nk(D + C))$, where m is the adjacency size, p is the aligned-feature dimension, D is the residual dimension, C is the number of communities, and k is the candidate count. With fixed L, p, D, C, k, this is approximately linear in graph size. Table 10 reports empirical training and inference; data loading, preprocessing, metric computation, and orchestration are excluded.

Table 10: Runtime in seconds. OFO is averaged over 12 targets; OFA is averaged over A/B/C source settings and their target evaluations.

Unsupervised OFO			Generalist OFA		
Method	**Train**	**Infer**	**Method**	**Train/source**	**Infer/target**
DOMINANT	2.187	0.006	ARC	3.378	0.012
AnomalyDAE	10.358	0.353	IA-GGAD	14.041	0.056
CoLA	1.153	0.028	UNPrompt	180.960	0.061
ADA-GAD	6.479	0.366	AnomalyGFM-ZS	9.501	0.015
DiffGAD	16.910	1.125	OWLEYE	20.112	0.144
RECAP-OFO	3.994	0.020			

Table 11: Community and anomaly-ranking stability across seed pairs.

Setting	**NMI**	**ARI**	**Soft co-assignment**	**Score Spearman**	C_{eff}
OFO	0.372 ± 0.008	0.415 ± 0.017	0.650 ± 0.012	0.899 ± 0.006	24.91 ± 1.49
A	0.265 ± 0.037	0.364 ± 0.094	0.576 ± 0.078	0.878 ± 0.019	16.20 ± 2.33
B	0.172 ± 0.005	0.257 ± 0.056	0.573 ± 0.040	0.735 ± 0.006	22.34 ± 1.29
C	0.091 ± 0.019	0.047 ± 0.096	0.331 ± 0.079	0.663 ± 0.018	16.15 ± 0.49

Community identities are permutation-arbitrary, so NMI, ARI, and soft co-assignment measure structure while Spearman measures the final ranking. Rankings are substantially more stable than community identities; both decline under the citation-only source shift, which limits claims of universal transfer.

Table 12: Target-side large-graph inference. AUROC/AUPRC are reported to prevent computational completion from being mistaken for predictive success.

Target	**Nodes**	**KNN**	**Cold (s)**	**Warm (s)**	**GPU GiB**	**AUROC/AUPRC (%)**
T-Finance	39,357	Exact	9.87	0.16 ± 0.12	0.98	$25.57/2.80$
DGraph-Fin	3,700,550	IVF-PQ	510.33	6.61 ± 0.11	48.10	$36.75/0.90$
T-Social	5,781,065	IVF-PQ	852.21	10.51 ± 0.09	75.13	$43.79/2.60$

All nine frozen-checkpoint evaluations produce finite full-node scores without target labels or parameter updates. However, all mean AUROCs are below 50% and all AUPRCs are below their anomaly-prevalence references. The result therefore supports target-side computational scalability only; it does not support predictive transfer to these three large graphs.

D.3 Detailed Explanation of the Ablation Study.

To systematically analyze the contribution of each key component in RECAP, we conduct two groups of ablation studies. The first group removes or modifies specific modules in RECAP to evaluate the effectiveness of residual representation, community regularization, and anomaly scoring components. The second group compares RECAP with simple combinations of residual features and classical unsupervised anomaly detectors, aiming to verify whether residual features alone are sufficient for cross-graph anomaly detection.

Module Removal Study In the module removal study, we keep the main training and inference pipeline of RECAP unchanged and only replace the target component at runtime. This design ensures that the performance difference mainly comes from the removed or modified component. We compare the following variants:

- **RECAP**: the full model.
- **RECAP w/o Residual**: removes the residual construction and directly concatenates propagated hop features instead of using $X^{(l)} - X^{(0)}$.
- **RECAP w/o** L_{con}: removes the community consistency loss on the residual similarity graph.
- **RECAP w/o** L_H: removes the entropy-based community assignment regularization.
- **RECAP w/o Adhesion Score**: removes the prototype adhesion score during inference and only uses the KNN community context inconsistency score.

Table 13: Module removal ablation results averaged over eight target graphs (AUROC/AUPRC, %).

Method	AUROC	AUPRC
RECAP	**74.48 ± 0.25**	**26.58 ± 0.57**
RECAP w/o L_H	74.43 ± 0.26	26.57 ± 0.54
RECAP w/o L_{con}	74.20 ± 0.32	26.56 ± 0.22
RECAP w/o Context Score	73.16 ± 0.19	26.15 ± 0.54
RECAP w/o Adhesion Score	61.73 ± 1.29	12.00 ± 1.18
RECAP w/o Residual	53.39 ± 0.19	10.60 ± 0.05

Table 14: Simple combination ablation results averaged over eight target graphs (AUROC/AUPRC, %).

Method	AUROC	AUPRC
RECAP	**74.47 ± 0.24**	**26.62 ± 0.58**
Residual + KMeans	67.82 ± 0.34	18.23 ± 0.45
Residual + GMM	66.14 ± 0.43	17.57 ± 0.46
Residual + LOF	58.92 ± 0.00	15.13 ± 0.00
Residual + Spectral Clustering	52.15 ± 0.70	8.71 ± 0.30

- **RECAP w/o Context Score**: removes the context inconsistency score during inference and only uses the prototype adhesion score.

As shown in Table 13, the residual representation is the most critical component of RECAP. When the residual construction is removed, the AUROC drops from 74.48 to 53.39, and the AUPRC drops from 26.58 to 10.60. This substantial degradation indicates that directly concatenating propagated features cannot effectively characterize anomalous nodes. In contrast, the residual representation $X^{(l)} - X^{(0)}$ explicitly captures how node attributes deviate during multi-hop propagation, which provides a more discriminative signal for cross-graph anomaly detection.

The anomaly scoring module also plays an important role. Removing the adhesion score leads to a severe performance drop, with AUROC decreasing to 61.73 and AUPRC decreasing to 12.00. By contrast, removing the context score only causes a mild decrease, resulting in 73.16 AUROC and 26.15 AUPRC. This suggests that the prototype adhesion score is the dominant component in RECAP's anomaly scoring function. It directly measures how well a node can be assigned to the learned normal community prototypes. The context inconsistency score provides complementary information by comparing the node's community distribution with that of its KNN neighbors, but its contribution is secondary in the averaged results.

The effects of L_{con} and L_H are relatively moderate under the current configuration. Removing L_H produces almost the same averaged performance as the full model, while removing L_{con} results in only a slight AUROC decrease. This indicates that the residual representation and the scoring function provide the main detection ability, while these loss terms mainly help regularize and stabilize community learning. Nevertheless, per-dataset results show that they can still influence specific target graphs. For example, on Weibo, removing L_{con} decreases AUPRC from 34.15 to 32.24, suggesting that community consistency can be beneficial for graphs with stronger local structural patterns.

Simple Combination Study To further verify that RECAP's advantage does not merely come from the residual feature itself, we compare RECAP with several simple combinations of residual features and classical unsupervised anomaly detectors. These baselines use the same feature alignment and multi-hop propagation as RECAP. They first construct the raw residual embedding:

$$E_0 = [X^{(1)} - X^{(0)} \| X^{(2)} - X^{(0)} \| \cdots \| X^{(L)} - X^{(0)}], \tag{46}$$

and then directly apply classical unsupervised detectors on the target graph, including KMeans, Spectral Clustering, Gaussian Mixture Model (GMM), and Local Outlier Factor (LOF). Unlike RECAP, these methods do not train a transferable mapping from source graphs.

Table 14 shows that all simple combination baselines are clearly inferior to RECAP. Among them, Residual + KMeans achieves the best performance, with 67.82 AUROC and 18.23 AUPRC, but it is still substantially worse than RECAP. Residual + GMM performs slightly worse than KMeans,

indicating that Gaussian mixture modeling can capture certain distributional patterns in the residual space but is insufficient for robust cross-graph anomaly detection. LOF and Spectral Clustering perform even worse, especially Spectral Clustering, whose averaged AUROC and AUPRC are only 52.15 and 8.71, respectively.

The per-dataset results further confirm this observation. On Citeseer, RECAP achieves 90.17 AUROC and 45.48 AUPRC, while Residual + KMeans only obtains 83.98 AUROC and 36.92 AUPRC. On Cora, RECAP achieves 82.17 AUROC and 40.56 AUPRC, whereas KMeans and GMM only obtain AUPRC scores around 25. Although some traditional detectors can perform reasonably well on individual datasets, such as LOF on BlogCatalog, their performance is not stable across different target graphs.

Controlled Context-Neighborhood Replacement To test whether the context score is merely a second aggregation of ARC-derived residual neighborhoods, we freeze the checkpoint, residual embeddings, assignments, prototypes, adhesion scores, k, τ_s, β, and score normalization. We change only the inference-time context graph from residual-space cosine KNN to exact cosine KNN over the aligned 32-dimensional input features.

Table 15: Controlled context-neighborhood replacement over Setting-A targets (AUROC/AUPRC, %).

Scoring method	AUROC	AUPRC
Prototype only	73.43±0.11	26.66±0.37
+ Residual-KNN context	74.65±0.23	27.04±0.34
+ Aligned-feature-KNN context	**74.94**±0.02	**27.21**±0.38

Both neighborhoods improve the seed-level dataset macro over prototype-only scoring. The aligned-feature variant improves AUROC/AUPRC by $1.51/0.55$ points and is positive for all three seeds, showing that context aggregation remains useful without residual-derived neighbors. The gain is dataset-dependent and largest on Weibo, so the result supports an average complementary effect rather than uniform improvement on every target.

Overall, these studies show that the residual representation is essential for capturing propagation-induced deviations and that RECAP's community learning/scoring adds value with residuals fixed. The prototype adhesion score is dominant, while the context score provides smaller complementary information that persists under a non-residual neighborhood. The positive matched interaction supports a residual-aware community-learning framework rather than independent additive modules.

D.4 Additional Hyperparameter Sensitivity Analysis

This section provides the complete one-dimensional hyperparameter sensitivity analysis of RECAP. Before the scalar sweeps, a coarse multi-parameter analysis is conducted to identify a stable operating region for the residual graph construction, community assignment, and scoring components. In particular, the joint behavior of the KNN size k and the number of communities C is examined because they jointly determine the residual similarity graph and the granularity of residual pattern communities. The joint analysis shows a broad high-performing region around moderate community numbers and sufficiently large neighborhoods, with $C = 32$–36 and $k = 64$–96 giving consistently strong results. Therefore, $C = 36$ and $k = 64$ are adopted as the default configuration, which is close to the best region while avoiding the extra cost of larger KNN graphs.

All one-dimensional sweeps follow the same cross-dataset protocol as the main experiments. RECAP is trained on PubMed, Flickr, Questions, and YelpChi, and evaluated on Facebook, Cora, CiteSeer, ACM, BlogCatalog, Weibo, Reddit, and Amazon. Each configuration is repeated for three trials, and the mean AUROC and AUPRC are reported in Fig. 5. For the scalar parameters, the searched values

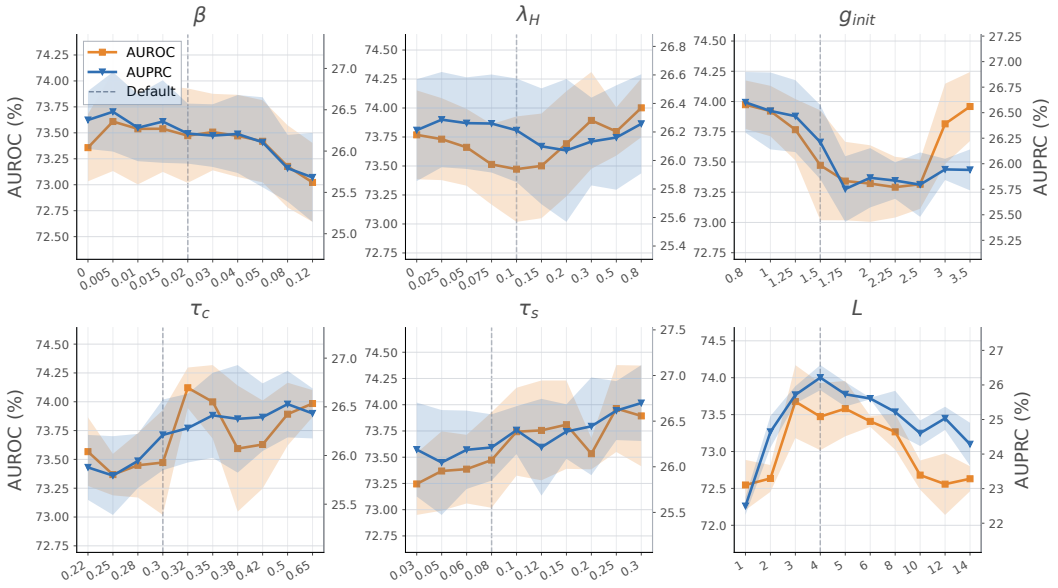


Figure 5: Hyperparameter sensitivity of RECAP under the cross-dataset pre-train-and-transfer setting. Each panel varies one scalar hyperparameter while keeping the others fixed. Solid lines denote mean AUROC and AUPRC over three trials, shaded areas indicate standard deviations, and dashed vertical lines mark the default values.

are

$$\begin{aligned}
\beta &\in \{0, 0.005, 0.01, 0.015, 0.02, 0.03, 0.04, 0.05, 0.08, 0.12\},\\
\lambda_H &\in \{0, 0.025, 0.05, 0.075, 0.1, 0.15, 0.2, 0.3, 0.5, 0.8\},\\
g_{\text{init}} &\in \{0.8, 1.0, 1.25, 1.5, 1.75, 2.0, 2.25, 2.5, 3.0, 3.5\},\\
\tau_c &\in \{0.22, 0.25, 0.28, 0.30, 0.32, 0.35, 0.38, 0.42, 0.50, 0.65\},\\
\tau_s &\in \{0.03, 0.05, 0.06, 0.08, 0.10, 0.12, 0.15, 0.20, 0.25, 0.30\},\\
L &\in \{1, 2, 3, 4, 5, 6, 8, 10, 12, 14\}.
\end{aligned} \tag{47}$$

The results show that RECAP is stable across a broad range of scalar hyperparameters. The context-inconsistency weight β should remain a secondary scoring term. A small positive value improves the ranking by adding neighborhood community-context inconsistency, but large values over-amplify noisy local assignment differences and reduce both AUROC and AUPRC. The community regularization weight λ_H has relatively mild influence, indicating that it mainly shapes the assignment distribution rather than dominating the anomaly score. The initialization gain g_{init} affects optimization more visibly: small gains lead to smoother early assignments and stronger AUPRC, while middle-range gains are less favorable; large gains recover AUROC but do not recover AUPRC to the same extent.

The two temperature parameters control different forms of softness. The assignment temperature τ_c determines the sharpness of community assignments. Moderate-to-large values generally perform better, suggesting that overly hard assignments can harm transferability. The residual-similarity temperature τ_s controls the weighting of residual neighbors in the KNN graph. Larger τ_s values improve AUPRC, showing that smoother residual neighborhoods are beneficial for cross-domain generalization. Finally, the propagation depth L is the most structurally sensitive parameter. Shallow settings such as $L = 1$ or 2 miss high-order ego-neighbor residual patterns, whereas very large L introduces over-smoothed or irrelevant context. The best AUPRC is achieved around $L = 4$, which supports using a moderate ego-context order.