RESIDUAL-AUGMENTED LATENT FLOW MATCHING FOR GRAPH ANOMALY DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Unsupervised graph anomaly detection (GAD) aims to identify nodes that deviate from dominant normal patterns without access to anomaly labels. This task remains challenging due to the scarcity of abnormal nodes and severe class imbalance. Graph neural network-based methods and recent diffusion-based generative models have made substantial progress in modeling complex graph data distributions. However, message passing can make anomalous node representations less distinguishable from their neighborhoods, while diffusion-based generative detectors often require costly multi-step denoising. We propose RALFlow-GAD, a residual-augmented latent flow matching framework for unsupervised GAD. It augments graph autoencoder embeddings with degree-aware global and local residuals to preserve anomaly-relevant deviations, and applies flow matching to model the resulting latent distribution. It further constructs a graph-level prototype to summarize the dominant latent pattern and uses it to condition an auxiliary velocity field. Dual-flow contrastive guidance amplifies the discrepancy between the free and prototype-conditioned velocity fields, producing contrastive latent references that emphasize transport components not explained by the dominant pattern. At inference time, it generates these contrastive latent references in one step for efficient anomaly scoring and applies a lightweight label-free score orientation correction. Experiments on five real-world datasets show that RALFlow-GAD achieves the best average performance among the baselines. Compared with DiffGAD, the strongest baseline, RALFlow-GAD improves average AUROC by 10.6% and AP by 11.5%, while achieving up to a $41\times$ speedup.

1 INTRODUCTION

Graph anomaly detection is a fundamental task for identifying abnormal nodes in graph-structured data, with applications in fraud detection, fake account discovery, and malicious communication monitoring (Akoglu et al., 2015; Qiao et al., 2025; Liu et al., 2022a). In the unsupervised setting, anomaly labels are unavailable during training, and the detector must distinguish anomalous nodes from dominant graph patterns, which are assumed to be normal, using structural and attribute information. This task remains challenging because abnormal nodes are usually rare, can be camouflaged within normal communities, and may differ from normal nodes only through subtle attribute or neighborhood inconsistencies (Akoglu et al., 2015; Qiao et al., 2025). Effective detection therefore requires learning the common attribute–structure patterns in the graph while preserving subtle deviations that may indicate anomalies.

Graph autoencoders (GAEs) provide a foundation for this objective. By jointly encoding node attributes and graph topology, they map heterogeneous graph observations into a continuous, low-dimensional representation space (Kipf & Welling, 2016; Ding et al., 2019; Roy et al., 2024). Representative methods have built on GAEs to capture different forms of graph irregularity (Ding et al., 2019; Roy et al., 2024). These methods demonstrate that GAE representations provide an effective basis for modeling regular graph patterns and deriving reconstruction-based anomaly scores.

Despite recent progress, current GAD methods still face three important limitations. First, many GAE-based detectors employ neighborhood-aggregating encoders that tend to act as low-pass filters (Kipf & Welling, 2017; Li et al., 2018). Although such smoothing helps capture homophilous patterns among normal nodes, it may pull anomalous representations toward neighborhood aver-

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.

ages, reducing their separability under reconstruction-error scoring. Second, learning generative dynamics on small graphs can be sensitive to sampling variability because fewer node observations are available for estimating the latent distribution, while diffusion-based generative GAD methods additionally require costly multi-step denoising or sampling (Li et al., 2025a). Third, larger discrepancies from flow-generated references do not necessarily indicate greater abnormality. Camouflaged anomalies may closely resemble dominant graph patterns, whereas normal nodes can exhibit substantial heterogeneity (Zhou et al., 2022). Some anomalies may therefore yield smaller discrepancies than normal nodes. The flow-matching objective does not explicitly constrain this ordering, leaving the direction of anomaly scores unresolved. We therefore incorporate label-free score orientation correction into our detector, using a local feature–neighborhood inconsistency prior to determine whether the flow-based ranking should be retained or reversed.

To this end, we propose RALFlow-GAD, a residual-augmented latent flow matching framework for unsupervised graph anomaly detection. It augments GAE embeddings with degree-aware global and local residuals to preserve anomaly-relevant deviations in the latent space, and applies flow matching to model the augmented latent distribution. To stabilize latent transport estimation on small graphs, a graph-level prototype is constructed to summarize the dominant latent pattern and condition an auxiliary velocity field. The discrepancy between this prototype-conditioned velocity field and an independently trained free velocity field is amplified through dual-flow contrastive guidance. The resulting contrastive velocity produces latent references that emphasize transport components not explained by the dominant pattern. By using one-step latent generation, RALFlow-GAD avoids iterative denoising and enables efficient anomaly scoring. A lightweight label-free score orientation correction module further determines the score direction without validation labels.

In summary, our main contributions include:

- We propose RALFlow-GAD, an efficient residual-augmented latent flow matching framework for unsupervised GAD. To the best of our knowledge, this is the first work to apply flow matching to GAD.
- We introduce degree-aware residual augmentation, graph-level prototype conditioning, dual-flow contrastive guidance, and label-free score orientation correction. These components preserve graph-specific deviation signals, provide a stable reference for the dominant latent pattern, and mitigate dataset-dependent score-orientation ambiguity.
- Extensive experiments on five real-world benchmarks show that RALFlow-GAD achieves the best performance, improving over the strongest baseline by 10.6% in average AUROC and 11.5% in average AP, while achieving up to $41\times$ speedup over the diffusion-based generative baseline.

2 Related Work

Unsupervised GAD. A common paradigm in unsupervised GAD is to encode node attributes and graph topology into latent representations and derive anomaly scores from attribute, structure, or neighborhood reconstruction errors (Ding et al., 2019; Roy et al., 2024). DOMINANT (Ding et al., 2019) jointly reconstructs node attributes and graph structure, CONAD (Xu et al., 2022) incorporates contrastive graph augmentation to improve detection robustness, and GAD-NR (Roy et al., 2024) detects anomalies by reconstructing neighborhood information. Recent methods further explore one-class homophily modeling, smoothing-aware representation learning, cluster-aware detection, and training-free detection (Qiao & Pang, 2023; Dong et al., 2025; Zheng et al., 2025; Zhao et al., 2025). However, neighborhood aggregation can smooth anomalous representations and obscure the deviations used for reconstruction-based scoring.

Generative GAD. Recent generative approaches seek to improve graph anomaly detection by modeling complex graph data distributions. GODM (Liu et al., 2023a) combines variational graph encoding with diffusion-based negative-sample generation, while DiffGAD (Li et al., 2025a) applies diffusion models directly to unsupervised graph anomaly detection. Although diffusion models offer strong distribution-modeling capacity, their iterative denoising or sampling procedures can incur substantial inference overhead. This limitation motivates more computationally efficient generative approaches for graph anomaly detection.

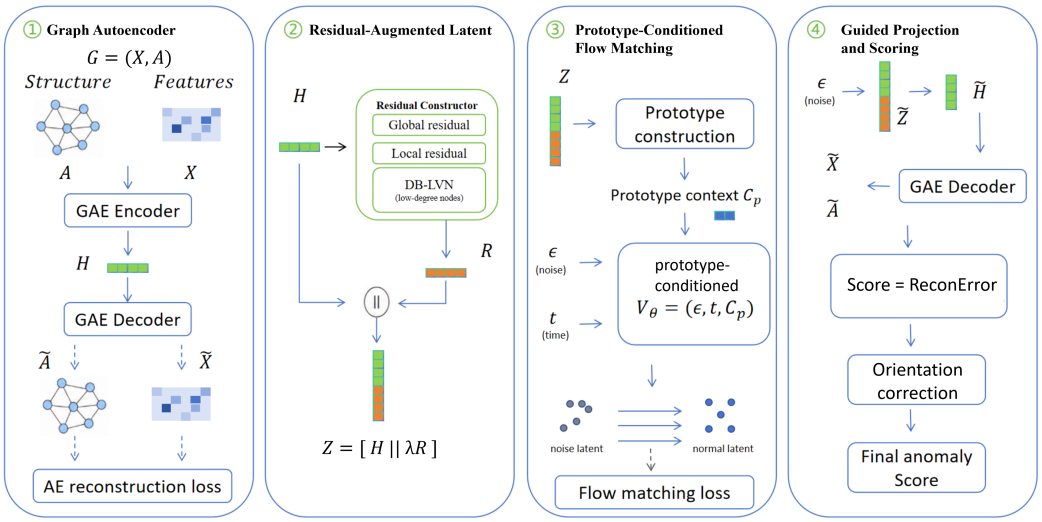

Figure 1: Overview of the proposed RALFlow-GAD. The framework consists of four stages: graph autoencoding, residual augmentation, prototype-conditioned flow matching, and label-free score orientation correction.

Flow Matching. Flow matching learns continuous transport dynamics by directly regressing velocity fields along predefined probability paths (Lipman et al., 2022). Rectified flow further encourages straighter transport trajectories, enabling accurate generation with fewer numerical integration steps (Liu et al., 2023b). Flow matching has also been applied in the representation space of pretrained autoencoders to reduce the computational cost of modeling high-dimensional observations (Dao et al., 2023). Variational Flow Matching extends the framework to graph generation (Eijkelboom et al., 2024). Existing graph flow-matching studies have primarily focused on graph generation rather than GAD.

3 Problem Definition

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X}, \mathbf{A})$ be an undirected attributed graph, where $\mathcal{V}$ is the node set with $|\mathcal{V}| = N$, $\mathcal{E}$ is the edge set, $\mathbf{X} \in \mathbb{R}^{N \times F}$ is the node attribute matrix with feature dimension F, and $\mathbf{A} \in \{0,1\}^{N \times N}$ is the adjacency matrix. For node $i \in \mathcal{V}$, $\mathcal{N}(i) = \{j \mid (i,j) \in \mathcal{E}\}$ is its one-hop neighborhood, and $d_i = |\mathcal{N}(i)|$ is its degree. We use $\cdot \| \cdot$ as vector concatenation and $\cos(\cdot,\cdot)$ as cosine similarity.

In unsupervised node-level graph anomaly detection, the goal is to learn an anomaly scoring function

$$f_\Theta : \mathcal{G} \mapsto \mathbf{s}^{\text{final}} \in \mathbb{R}^N,$$

where s_i^{final} denotes the final anomaly score of node i, and Θ is the model parameters. Following the standard ranking convention, a larger score indicates greater abnormality. Ground-truth labels are used only for evaluation, not for training, hyperparameter selection or anomaly-score generation.

4 Method

Figure 1 presents the overall framework of RALFlow-GAD. Given an input graph, we first train a graph autoencoder to obtain node embeddings and augment these embeddings with degree-aware global and local residuals to alleviate over-smoothing and preserve anomaly-relevant deviation signals. We then apply flow matching in the residual-augmented latent space. To stabilize latent transport modeling on small graphs, a graph-level prototype is constructed to summarize the dominant latent pattern and condition an auxiliary velocity field, referred to as the prototype-conditioned velocity field. In parallel, an independently trained free velocity field models the empirical latent transport without prototype conditioning. At inference time, dual-flow contrastive guidance amplifies the discrepancy between the two velocity fields to generate contrastive latent references in one

step. Finally, a label-free score orientation correction module determines the score direction using a local feature–neighborhood inconsistency prior.

4.1 Residual-Augmented Latent Representation

Graph autoencoding. RALFlow-GAD first employs a graph autoencoder with a Graph Convolutional Network (GCN) encoder to obtain node representations: $\mathbf{H} = \mathrm{Enc}(\mathbf{X}, \mathbf{A}) \in \mathbb{R}^{N\times d}$, $\mathbf{h}_i$ is node i's embedding. The decoder reconstructs both node attributes and graph structure from $\mathbf{H}$ as $(\hat{\mathbf{X}}, \hat{\mathbf{A}}) = \mathrm{Dec}(\mathbf{H}, \mathbf{A})$. The graph autoencoder is trained with the reconstruction objective

$$\mathcal{L}_{\mathrm{AE}} = \alpha_{\mathrm{AE}}\ell^X(\mathbf{X}, \hat{\mathbf{X}}) + (1-\alpha_{\mathrm{AE}})\ell^A(\mathbf{A}, \hat{\mathbf{A}}), \tag{1}$$

where $\ell^X(\mathbf{X}, \hat{\mathbf{X}})$ and $\ell^A(\mathbf{A}, \hat{\mathbf{A}})$ denote the attribute and structure reconstruction losses, respectively, and $\alpha_{\mathrm{AE}} \in [0, 1]$ controls their relative contribution.

Degree-aware latent virtual neighbors. The local residual of a node is computed relative to its neighborhood. However, for a low-degree node, the observed neighborhood may contain too few nodes to provide a reliable local reference. We therefore introduce a degree-aware latent virtual-neighbor mechanism.

Let K denote the neighborhood-size threshold for virtual-neighbor supplementation. For a node with $d_i < K$, we set $m_i = K - d_i$. Candidate nodes are drawn from $\mathcal{V} \setminus (\{i\} \cup \mathcal{N}(i))$ and ranked by cosine similarity to $\mathbf{h}_i$ in the embedding space. The m_i highest-ranked candidates form the virtual neighbor set of node i, denoted by $\mathcal{V}_{m_i}(i)$. We then define

$$\widetilde{\mathcal{N}}(i) = \begin{cases} \mathcal{N}(i) \cup \mathcal{V}_{m_i}(i), & \text{if } d_i < K, \\ \mathcal{N}(i), & \text{if } d_i \geq K. \end{cases} \tag{2}$$

The supplemented neighborhood is used only to estimate the local residual, and it will not modify the input graph structure used by the GCN encoder or the graph autoencoder decoder.

Degree-adaptive dual residuals. The global residual measures the deviation of node i from the graph-level center; the local residual measures deviation from the supplemented neighborhood:

$$\mathbf{r}_i^{\mathrm{glob}} = \mathbf{h}_i - \frac{1}{N}\sum_{j=1}^{N}\mathbf{h}_j, \quad \mathbf{r}_i^{\mathrm{loc}} = \mathbf{h}_i - \frac{1}{|\widetilde{\mathcal{N}}(i)|}\sum_{j\in\widetilde{\mathcal{N}}(i)}\mathbf{h}_j.$$

We fuse the two residuals using a degree-adaptive gate:

$$\mathbf{r}_i = \gamma_i\mathbf{r}_i^{\mathrm{loc}} + (1-\gamma_i)\mathbf{r}_i^{\mathrm{glob}}, \qquad \gamma_i = \sigma\Big((\widetilde{d}_i - \beta)\kappa\Big), \tag{3}$$

where $\widetilde{d}_i = |\widetilde{\mathcal{N}}(i)|$, β is a learnable degree bias, $\kappa > 0$ is a learnable sharpness parameter, and $\sigma(\cdot)$ denotes the sigmoid function. Consequently, low-degree nodes rely more strongly on the global reference, whereas high-degree nodes can place greater weight on their more reliably estimated local contexts.

The final latent representation with λ controlling the contribution of the residual component becomes

$$\mathbf{z}_i = [\mathbf{h}_i \,\|\, \lambda\mathbf{r}_i] \in \mathbb{R}^{2d}. \tag{4}$$

4.2 Prototype-Conditioned Latent Flow Matching

Flow matching in the graph latent space. We employ a free velocity field and a prototype-conditioned velocity field in the residual-augmented latent space. We first introduce the free field.

Flow matching learns continuous transport from a simple base distribution to the empirical latent distribution by directly regressing its velocity field (Lipman et al., 2022; Liu et al., 2023b). For a latent sample $\mathbf{z}_i$ and Gaussian noise $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, we use the straight probability path

$$\mathbf{z}_{i,t} = (1-t)\boldsymbol{\epsilon}_i + t\mathbf{z}_i, \qquad t \in [0, 1].$$

Differentiating the interpolation path with respect to t yields

$$\mathbf{u}_i = \frac{\mathrm{d}\mathbf{z}_{i,t}}{\mathrm{d}t} = \mathbf{z}_i - \boldsymbol{\epsilon}_i. \tag{5}$$

The resulting target velocity is constant with respect to t. An unconditional velocity network v_θ^{free} is trained by

$$\mathcal{L}_{\text{free}} = \mathbb{E}_{i,t,\boldsymbol{\epsilon}_i}\left[\left\|v_\theta^{\text{free}}(\mathbf{z}_{i,t}, t) - \mathbf{u}_i\right\|_2^2\right]. \tag{6}$$

The free velocity field models the empirical transport of the residual-augmented latent representations without graph-level prototype conditioning. It is also used to generate latent references for estimating the dominant-pattern prototype. Compared with diffusion-based generative models that require iterative denoising, straight-path flow matching supports efficient latent generation with few integration steps (Lipman et al., 2022; Liu et al., 2023b).

Dominant-pattern prototype. To stabilize latent transport estimation on small graphs, we construct a graph-level prototype in the embedding component of the flow latent space to summarize the dominant latent pattern. It is initialized by $\mathbf{c}^{(0)} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{h}_i$.

During free-flow training, we generate latent references using a short Euler solver and retain their first d dimensions as $\widetilde{\mathbf{h}}_i$, corresponding to the embedding component in Eq. (4).

Given the current prototype $\mathbf{c}$, each reference receives the normalized weight

$$a_i = \frac{\exp\Big(\cos(\mathbf{c}, \widetilde{\mathbf{h}}_i)/\tau_p\Big)}{\sum_{j=1}^{N}\exp\Big(\cos(\mathbf{c}, \widetilde{\mathbf{h}}_j)/\tau_p\Big)}, \tag{7}$$

and the prototype is updated as $\mathbf{c} = \sum_{i=1}^{N} a_i \widetilde{\mathbf{h}}_i$.

Under the assumption that anomalies are rare, similarity weighting reduces the influence of isolated deviations, allowing the shared prototype to summarize the dominant latent pattern. We then train a second velocity network, v_ω^{proto}, conditioned on the fixed prototype:

$$\mathcal{L}_{\text{proto}} = \mathbb{E}_{i,t,\boldsymbol{\epsilon}_i}\left[\left\|v_\omega^{\text{proto}}(\mathbf{z}_{i,t}, t, \mathbf{c}) - \mathbf{u}_i\right\|_2^2\right]. \tag{8}$$

The resulting prototype-conditioned velocity field models latent transport while incorporating the shared dominant-pattern reference. Unlike the free velocity field, which estimates transport solely from the interpolated latent state and time, the prototype-conditioned field additionally receives graph-level contextual information. This conditioning is designed to stabilize latent transport modeling when the observed graph provides limited effective node samples. The prototype-conditioned field is not assumed to represent a purely normal distribution; instead, it provides a dominant-pattern-conditioned counterpart to the free velocity field for subsequent contrastive guidance. We first optimize the graph autoencoder, then train the free velocity field while updating the prototype, and finally fix the prototype and train the prototype-conditioned velocity field.

Because the three stages are optimized sequentially, we do not define a single weighted total loss. The graph autoencoder, free velocity field, and prototype-conditioned velocity field are optimized using $\mathcal{L}_{\text{AE}}$, $\mathcal{L}_{\text{free}}$, and $\mathcal{L}_{\text{proto}}$, resp.

4.3 One-Step Flow Scoring and Orientation Correction

Dual-flow contrastive generation. We adopt flow matching because its velocity-field formulation provides a convenient basis for dual-flow contrastive guidance. The free and prototype-conditioned fields are evaluated at the same latent state and time, allowing their difference to be incorporated directly into the generation dynamics. We use a single Euler step to construct latent references, reducing sequential inference steps relative to the iterative denoising procedure used by the diffusion baseline. Under fixed trained models and non-step hyperparameters, our controlled integration-step study in Table 10 shows that one-step inference achieves the highest average AUROC among the evaluated step counts, supporting this inference design. Motivated by classifier-free guidance (Ho

& Salimans, 2022), we construct a contrastive velocity from the free and prototype-conditioned velocity fields:

$$\mathbf{v}_i^{\mathrm{ctr}} = (1+w)v_\theta^{\mathrm{free}}(\boldsymbol{\epsilon}_i, 0) - wv_\omega^{\mathrm{proto}}(\boldsymbol{\epsilon}_i, 0, \mathbf{c}), \tag{9}$$

where $w \geq 0$ controls the contrastive strength. This formulation amplifies the difference between the two fields by extrapolating away from the prototype-conditioned prediction; $w = 0$ recovers the free velocity field.

Starting from Gaussian noise at $t = 0$, we obtain the one-step latent reference $\widehat{\mathbf{z}}_i = \boldsymbol{\epsilon}_i + \mathbf{v}_i^{\mathrm{ctr}}$. These references are not intended to reconstruct individual node embeddings, but form a graph-sized contrastive reference set. We extract $\widehat{\mathbf{h}}_i = \widehat{\mathbf{z}}_{i,1:d}$ and decode $(\widetilde{\mathbf{X}}, \widetilde{\mathbf{A}}) = \mathrm{Dec}(\widehat{\mathbf{H}}, \mathbf{A})$. The raw anomaly score is

$$s_i^{\mathrm{raw}} = \alpha_{\mathrm{AE}}\ell^X(\mathbf{x}_i, \widetilde{\mathbf{x}}_i) + (1-\alpha_{\mathrm{AE}})\ell^A(\mathbf{A}_{i,:}, \widetilde{\mathbf{A}}_{i,:}), \tag{10}$$

where $\widetilde{\mathbf{x}}_i$ and $\widetilde{\mathbf{A}}_{i,:}$ denote the i-th attribute vector and adjacency row decoded from the generated reference set, respectively. The two terms measure attribute and structural discrepancies, with α_{AE} controlling their relative contributions. We further smooth the scalar scores over the observed graph:

$$s_i = (1-\mu)s_i^{\mathrm{raw}} + \mu\frac{1}{\max(d_i, 1)}\sum_{j\in\mathcal{N}(i)} s_j^{\mathrm{raw}}, \quad \mu \in [0,1]. \tag{11}$$

This operation acts only on the final scores and is distinct from encoder message passing.

Label-free score orientation correction. The flow-matching objective does not directly constrain whether larger decoded discrepancies indicate greater abnormality. This ambiguity is relevant to our scoring procedure, which compares observations with contrastively generated references. To determine score orientation without anomaly labels, we introduce a local feature–neighborhood inconsistency prior. Apparent conformity to dominant latent patterns need not eliminate local inconsistencies; assuming that these inconsistencies remain informative about abnormality, their ranking provides a complementary reference for score orientation correction. We define this prior as follows:

$$p_i = \left\| \mathbf{x}_i - \frac{1}{\max(d_i, 1)}\sum_{j\in\mathcal{N}(i)} \mathbf{x}_j \right\|_2. \tag{12}$$

Let $\mathbf{s}$ and $\mathbf{p}$ denote the flow-based scores and the inconsistency prior, respectively. We transform them into normalized rankings:

$$\mathbf{r}^{\mathrm{flow}} = \mathcal{R}(\mathbf{s}), \qquad \mathbf{r}^{\mathrm{prior}} = \mathcal{R}(\mathbf{p}), \tag{13}$$

where $[\mathcal{R}(\mathbf{q})]_i = (\mathrm{rank}(q_i) - 1)/(N-1)$. Their agreement is measured by $\rho = \mathrm{corr}\left(\mathbf{r}^{\mathrm{flow}}, \mathbf{r}^{\mathrm{prior}}\right)$, where $\mathrm{corr}(\cdot,\cdot)$ is Pearson correlation between the normalized rank vectors. Flow-based ranking is maintained when its agreement with the prior reaches a predefined threshold or reversed otherwise:

$$\mathbf{r}^{\mathrm{ori}} = \begin{cases} \mathbf{r}^{\mathrm{flow}}, & \text{if } \rho \geq \delta, \\ \mathbf{1} - \mathbf{r}^{\mathrm{flow}}, & \text{otherwise}, \end{cases} \tag{14}$$

where δ is the rank-agreement threshold. Finally, the oriented flow ranking and the prior ranking are combined as

$$\mathbf{s}^{\mathrm{final}} = \eta\mathbf{r}^{\mathrm{ori}} + (1-\eta)\mathbf{r}^{\mathrm{prior}}, \tag{15}$$

where $\eta \in [0,1]$ controls their relative contributions.

5 Experiments

5.1 Experimental Setup

Datasets. Following the BOND benchmark (Liu et al., 2022b), we evaluate RALFlow-GAD on five attributed GAD benchmarks: Books (Sánchez et al., 2013), Disney (Müller et al., 2013), Enron (Sánchez et al., 2013), Reddit (Kumar et al., 2019), and Weibo (Zhao et al., 2020). The datasets vary in graph size, feature distribution, and the proportion of anomalies.

Table 1: AUROC comparison on five benchmarks. Best and second-best results are highlighted in bold and underlined, resp.

Method	Books	Disney	Enron	Reddit	Weibo	Avg.
Graph-agnostic methods						
LOF	0.365 ± 0.000	0.479 ± 0.000	0.464 ± 0.000	0.572 ± 0.000	0.565 ± 0.000	0.489
IF	0.430 ± 0.018	$\underline{0.576 \pm 0.029}$	0.401 ± 0.014	0.452 ± 0.017	0.535 ± 0.028	0.479
MLPAE	0.425 ± 0.056	0.492 ± 0.057	0.446 ± 0.071	0.506 ± 0.000	0.821 ± 0.036	0.538
Classical graph anomaly detectors						
SCAN	0.498 ± 0.017	0.505 ± 0.040	0.528 ± 0.034	0.499 ± 0.003	0.637 ± 0.056	0.533
Radar	0.528 ± 0.000	0.518 ± 0.000	0.541 ± 0.101	0.549 ± 0.012	$\mathbf{0.989 \pm 0.001}$	0.625
ANOMALOUS	0.528 ± 0.000	0.518 ± 0.000	0.550 ± 0.098	0.549 ± 0.056	$\mathbf{0.989 \pm 0.001}$	0.627
Deep graph anomaly detectors						
GCNAE	0.500 ± 0.045	0.422 ± 0.079	0.382 ± 0.065	0.506 ± 0.000	0.908 ± 0.012	0.544
DOMINANT	0.501 ± 0.050	0.471 ± 0.045	0.537 ± 0.042	0.560 ± 0.002	0.850 ± 0.146	0.584
DONE	0.432 ± 0.040	0.417 ± 0.062	0.467 ± 0.061	0.539 ± 0.029	0.853 ± 0.041	0.542
AdONE	0.536 ± 0.020	0.488 ± 0.051	0.445 ± 0.029	0.504 ± 0.045	0.846 ± 0.022	0.564
AnomalyDAE	0.622 ± 0.081	0.488 ± 0.022	0.543 ± 0.112	0.557 ± 0.004	0.915 ± 0.012	0.625
GAAN	0.549 ± 0.050	0.480 ± 0.000	0.593 ± 0.002	0.554 ± 0.004	0.925 ± 0.000	0.620
CONAD	0.522 ± 0.069	0.480 ± 0.035	0.516 ± 0.043	0.561 ± 0.001	0.854 ± 0.143	0.587
Representative recent methods						
CAGAD	0.507 ± 0.031	0.428 ± 0.023	0.367 ± 0.002	0.560 ± 0.001	0.653 ± 0.006	0.503
TAM	0.548 ± 0.028	0.487 ± 0.029	0.304 ± 0.003	$\mathbf{0.583 \pm 0.003}$	0.803 ± 0.003	0.545
SmoothGNN	0.524 ± 0.057	0.573 ± 0.050	0.466 ± 0.044	0.462 ± 0.091	0.671 ± 0.176	0.539
FreeGAD	0.559 ± 0.000	0.573 ± 0.000	$\underline{0.671 \pm 0.000}$	0.483 ± 0.000	0.062 ± 0.000	0.470
GAD-NR	0.468 ± 0.066	0.541 ± 0.023	0.557 ± 0.117	0.554 ± 0.023	0.640 ± 0.179	0.552
DiffGAD	$\mathbf{0.673 \pm 0.096}$	0.532 ± 0.019	0.661 ± 0.032	0.562 ± 0.002	0.873 ± 0.067	$\underline{0.660}$
RALFlow-GAD	$\underline{0.624 \pm 0.016}$	$\mathbf{0.663 \pm 0.088}$	$\mathbf{0.847 \pm 0.012}$	$\underline{0.573 \pm 0.016}$	$\underline{0.944 \pm 0.000}$	**0.730**

Baselines and evaluation metrics. We chose 19 baselines for comparison. Among them, CAGAD (Zheng et al., 2025), TAM (Qiao & Pang, 2023), SmoothGNN (Dong et al., 2025), FreeGAD (Zhao et al., 2025), GAD-NR (Roy et al., 2024), and DiffGAD (Li et al., 2025a) are representative of the latest GAD methods, and we adopt their official implementations when available. Other classical and deep GAD models are also included for completeness and the corresponding results are based on those in DiffGAD. We evaluate detection performance using AUROC and average precision (AP). AUROC results are presented in the main text, while the complete AP results are provided in Appendix B. We report the results from five runs with different random seeds. Further implementation and evaluation details are provided in Appendix A.

Implementation. RALFlow-GAD uses PyTorch 2.4.1 and PyTorch Geometric 2.6.1; all our experiments run on a single NVIDIA GeForce RTX 4090 GPU. The graph autoencoder is trained for up to 300 epochs using Adam with a dataset-specific initial learning rate and a weight decay of 0.01, whereas each flow network is trained for up to 800 epochs using Adam with an initial learning rate of 0.0025 and no weight decay. At inference time, we use one Euler step for prototype-contrastive latent reference generation. Following (Li et al., 2025b), we select dataset-specific hyperparameters by maximizing the Contrast Score Margin (CSM) over a predefined discrete search space, without using labels. Detailed model architectures, dataset-specific configurations, hyperparameter search spaces, and the label-free configuration protocol are provided in Appendix A.

5.2 Main Results and Analysis

Table 1 presents the AUROC results on the five datasets. RALFlow-GAD achieves the highest average AUROC of 0.730, compared with 0.660 for DiffGAD, corresponding to a relative improvement of 10.6%. At the dataset level, RALFlow-GAD obtains AUROC scores of 0.624, 0.663, 0.847, 0.573, and 0.944 on Books, Disney, Enron, Reddit, and Weibo, respectively. It ranks first on Disney and Enron and second on the remaining three datasets. The largest improvement over DiffGAD is observed on Enron, where AUROC increases from 0.661 to 0.847. On Disney, RALFlow-GAD improves the best baseline result from 0.576 to 0.663. Although Radar and ANOMALOUS achieve the highest AUROC on Weibo, their performance on the other datasets is substantially weaker. Similarly, TAM performs best on Reddit but performs poorly on several other datasets, particularly

Table 2: End-to-end runtime in seconds under seed 42, reported as mean ± standard deviation over three repeated executions. Avg. denotes the arithmetic mean of the five dataset-wise mean runtimes. Lower runtime and higher speedup are better.

Method	Books	Disney	Enron	Reddit	Weibo	Avg.
RALFlow-GAD	65.00 ± 1.86	22.71 ± 0.57	376.99 ± 15.56	321.84 ± 4.19	556.26 ± 11.75	268.56
DiffGAD	$1,188.94 \pm 17.68$	932.47 ± 5.12	$2,909.49 \pm 32.52$	$2,539.74 \pm 41.45$	$2,752.09 \pm 27.81$	2,064.55
Speedup	$\mathbf{18.29}\times$	$\mathbf{41.06}\times$	$\mathbf{7.72}\times$	$\mathbf{7.89}\times$	$\mathbf{4.95}\times$	$\mathbf{7.69}\times$

Table 3: Ablation results in mean AUROC across five runs with different random seeds. Each variant removes one component while keeping all applicable settings fixed. Avg. denotes the arithmetic mean across the five datasets, and ΔAvg. equals the variant average minus the full-model average.

Variant	Books	Disney	Enron	Reddit	Weibo	Avg.	ΔAvg.
Full RALFlow-GAD	0.624	0.663	0.847	0.573	0.944	0.730	–
w/o Score Orientation	0.376	0.342	0.150	0.442	0.944	0.451	−0.279
w/o Graph Score Smoothing	0.592	0.610	0.833	0.537	0.898	0.694	−0.036
w/o Residual Augmentation	0.615	0.631	0.816	0.576	0.944	0.716	−0.014
w/o Residual Fusion	0.601	0.587	0.846	0.566	0.944	0.709	−0.021
w/o Virtual-Neighbor	0.624	0.659	0.843	0.568	0.944	0.728	−0.003
w/o Prototype Guidance	0.610	0.594	0.847	0.573	0.944	0.714	−0.017

Enron. In contrast, RALFlow-GAD ranks first or second on every benchmark, demonstrating more consistent performance across diverse graph distributions. Nevertheless, DiffGAD remains stronger on Books, indicating that the advantage of RALFlow-GAD is not uniform across all datasets. The AP results in Appendix B show a similar average-level trend. RALFlow-GAD achieves an average AP of 0.106, compared with 0.095 for DiffGAD, corresponding to a relative improvement of 11.5%.

5.3 Runtime

We compare the end-to-end wall-clock runtime of RALFlow-GAD and DiffGAD under the same experimental environment. Each method–dataset pair is executed three times, and the reported runtime includes data loading, training, inference, anomaly-score generation, and metric computation. As shown in Table 2, RALFlow-GAD is consistently faster across all five datasets, with speedups ranging from $4.95\times$ on Weibo to $41.06\times$ on Disney. The ratio between the summed runtimes of DiffGAD and RALFlow-GAD is $7.69\times$. This result is consistent with the reduced sampling cost of replacing iterative denoising with one-step latent flow inference.

5.4 Ablation Study

Tab. 3 evaluates each component using the same five seeds and fixed applicable hyperparameters.

Removing the score orientation correction module reduces average AUROC from 0.730 to 0.451, the largest decline among the ablations. The flow-matching objective learns latent transport without anomaly labels and does not require larger decoded discrepancies to correspond to greater abnormality. Thus, poor performance under a fixed score direction can reflect an informative ranking whose orientation is inconsistent with the anomaly-scoring convention.

The fine-grained ablation in Tab. 11 provides more direct evidence: orientation correction alone, without prior fusion, raises average AUROC from 0.451 to 0.727. Because this operation only retains or reverses the existing ranking, the improvement indicates that the flow-based scores already contain useful information for distinguishing normal and anomalous nodes. On Enron, for example, correcting the orientation increases AUROC from 0.150 to 0.850. Meanwhile, the local prior alone achieves an average AUROC of 0.592, showing that it cannot account for the full model's performance. These results highlight the complementary roles of the two components: flow-based modeling supplies informative discrepancies, while orientation correction uses the local prior to determine how those discrepancies should be ranked.

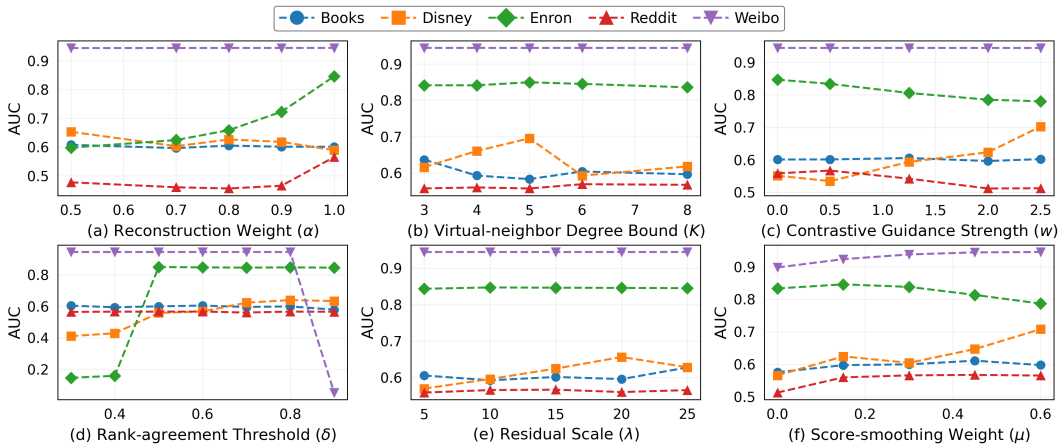

Figure 2: Hyperparameter sensitivity of RALFlow-GAD on five GAD datasets. Each point reports mean AUROC over five seeds. We vary one hyperparameter, keeping other settings fixed.

Graph score smoothing provides the second-largest average contribution, with its removal reducing average AUROC by 0.036. Prototype-conditioned guidance mainly benefits Books and Disney. For Enron, Reddit, and Weibo, the label-free configuration procedure selects $w = 0$, so the contrastive velocity reduces to the free velocity field and removing prototype guidance leaves inference unchanged. These results suggest that prototype conditioning is most useful on the smaller graphs in our evaluation, while its contribution can be deactivated through label-free configuration. Residual fusion and augmentation also improve average performance, whereas virtual-neighbor supplementation provides a smaller benefit. Their effects remain dataset-dependent: for example, removing residual augmentation slightly improves Reddit. Overall, the ablations support the components' complementary contributions without implying uniform improvements on every dataset.

5.5 Hyperparameter Sensitivity

Figure 2 examines one hyperparameter at a time while fixing the remaining settings, with results averaged over five random seeds. The virtual-neighbor threshold K and residual scale λ produce comparatively stable results, suggesting that residual construction is relatively insensitive to these settings within the evaluated ranges. Reconstruction weighting, guidance, and score smoothing exhibit stronger dataset-dependent effects. Larger α_{AE} benefits Enron and Reddit, whereas Disney favors greater emphasis on structural reconstruction. Prototype guidance is particularly useful on Disney. Moderate score smoothing benefits several datasets, but excessive smoothing may suppress node-level deviations. The orientation threshold δ produces the most abrupt changes because it determines whether the entire flow ranking is reversed. Unfavorable decisions at extreme threshold settings reveal a source of sensitivity in the orientation mechanism. Additional ablation and sensitivity analyses are provided in Appendices C and D, respectively.

6 Conclusion and Future Work

We presented RALFlow-GAD, a residual-augmented latent flow matching framework for unsupervised graph anomaly detection. Degree-aware residuals preserve anomaly-relevant deviations, while dual-flow contrastive guidance constructs latent references for one-step anomaly scoring. Label-free score orientation correction further addresses dataset-dependent ranking ambiguity. Experiments on five benchmarks show the highest average AUROC among the compared methods and up to $41\times$ speedup over DiffGAD.

Future work will explore scalable reconstruction, more reliable orientation mechanisms, and extensions to dynamic graphs.

AI USE STATEMENT

We used AI to assist with language polishing and LaTeX formatting, including table layout, float placement, and citation formatting.

REPRODUCIBILITY STATEMENT

Section 4 specifies the model, training objectives, and anomaly-scoring procedure. Section 5.1 describes the datasets, software versions, hardware, and training settings. Appendix A provides the evaluation protocol, model architecture, optimization settings, label-free configuration selection procedure, candidate sets, dataset-specific configurations, and runtime measurement protocol. Additional controlled analyses are provided in Appendices C and D.

ETHICS STATEMENT

We have considered the ethical implications of this work and do not identify any specific ethical concerns requiring further discussion.

REFERENCES

Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data Min. Knowl. Discov.*, 29(3):626–688, 2015. doi: 10.1007/S10618-014-0365-Y. URL https://doi.org/10.1007/s10618-014-0365-y.

Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space. *CoRR*, abs/2307.08698, 2023. doi: 10.48550/ARXIV.2307.08698. URL https://doi.org/10.48550/arXiv.2307.08698.

Kaize Ding, Jundong Li, Rohit Bhanushali, and Huan Liu. Deep anomaly detection on attributed networks. In Tanya Y. Berger-Wolf and Nitesh V. Chawla (eds.), *Proceedings of the 2019 SIAM International Conference on Data Mining, SDM 2019, Calgary, Alberta, Canada, May 2-4, 2019*, pp. 594–602. SIAM, 2019. doi: 10.1137/1.9781611975673.67. URL https://doi.org/10.1137/1.9781611975673.67.

Xiangyu Dong, Xingyi Zhang, Yanni Sun, Lei Chen, Mingxuan Yuan, and Sibo Wang. Smoothgnn: Smoothing-aware GNN for unsupervised node anomaly detection. In Guodong Long, Michale Blumestein, Yi Chang, Liane Lewin-Eytan, Zi Helen Huang, and Elad Yom-Tov (eds.), *Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025- 2 May 2025*, pp. 1225–1236. ACM, 2025. doi: 10.1145/3696410.3714615. URL https://doi.org/10.1145/3696410.3714615.

Floor Eijkelboom, Grigory Bartosh, Christian Andersson Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/15b780350b302a1bf9a3bd273f5c15a4-Abstract-Conference.html.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *CoRR*, abs/2207.12598, 2022. doi: 10.48550/ARXIV.2207.12598. URL https://doi.org/10.48550/arXiv.2207.12598.

Thomas N. Kipf and Max Welling. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016. URL http://arxiv.org/abs/1611.07308.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (eds.), *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pp. 1269–1278. ACM, 2019. doi: 10.1145/3292500.3330895. URL https://doi.org/10.1145/3292500.3330895.

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, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025a. URL https://openreview.net/forum?id=AhcYq4CnfF.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In Sheila A. McIlraith and Kilian Q. Weinberger (eds.), *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 3538–3545. AAAI Press, 2018. doi: 10.1609/AAAI.V32I1.11604. URL https://doi.org/10.1609/aaai.v32i1.11604.

Zhong Li, Yuhang Wang, and Matthijs van Leeuwen. Towards automated self-supervised learning for truly unsupervised graph anomaly detection. *Data Min. Knowl. Discov.*, 39(5):44, 2025b. doi: 10.1007/S10618-025-01115-5. URL https://doi.org/10.1007/s10618-025-01115-5.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *CoRR*, abs/2210.02747, 2022. doi: 10.48550/ARXIV.2210.02747. URL https://doi.org/10.48550/arXiv.2210.02747.

Kay Liu, Yingtong Dou, Yue Zhao, Xueying Ding, Xiyang Hu, Ruitong Zhang, Kaize Ding, Canyu Chen, Hao Peng, Kai Shu, George H. Chen, Zhihao Jia, and Philip S. Yu. Pygod: A python library for graph outlier detection. *CoRR*, abs/2204.12095, 2022a. doi: 10.48550/ARXIV.2204.12095. URL https://doi.org/10.48550/arXiv.2204.12095.

Kay Liu, Yingtong Dou, Yue Zhao, Xueying Ding, Xiyang Hu, Ruitong Zhang, Kaize Ding, Canyu Chen, Hao Peng, Kai Shu, Lichao Sun, Jundong Li, George H. Chen, Zhihao Jia, and Philip S. Yu. BOND: benchmarking unsupervised outlier node detection on static attributed graphs. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022b. URL http://papers.nips.cc/paper_files/paper/2022/hash/acc1ec4a9c780006c9aafd595104816b-Abstract-Datasets_and_Benchmarks.html.

Kay Liu, Hengrui Zhang, Ziqing Hu, Fangxin Wang, and Philip S. Yu. Data augmentation for supervised graph outlier detection with latent diffusion models. *CoRR*, abs/2312.17679, 2023a. doi: 10.48550/ARXIV.2312.17679. URL https://doi.org/10.48550/arXiv.2312.17679.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023b. URL https://openreview.net/forum?id=XVjTT1nw5z.

Emmanuel Müller, Patricia Iglesias Sánchez, Yvonne Mülle, and Klemens Böhm. Ranking outlier nodes in subspaces of attributed graphs. In Chee Yong Chan, Jiaheng Lu, Kjetil Nørvåg, and Egemen Tanin (eds.), *Workshops Proceedings of the 29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013*, pp. 216–222. IEEE Computer Society, 2013. doi: 10.1109/ICDEW.2013.6547453. URL https://doi.org/10.1109/ICDEW.2013.6547453.

Hezhe Qiao and Guansong Pang. Truncated affinity maximization: One-class homophily modeling for graph anomaly detection. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/9b905031125e56a557db38dff4fa8d21-Abstract-Conference.html.

Hezhe Qiao, Hanghang Tong, Bo An, Irwin King, Charu Aggarwal, and Guansong Pang. Deep graph anomaly detection: A survey and new perspectives. *IEEE Trans. Knowl. Data Eng.*, 37(9): 5106–5126, 2025. doi: 10.1109/TKDE.2025.3581578. URL https://doi.org/10.1109/TKDE.2025.3581578.

Amit Roy, Juan Shu, Jia Li, Carl Yang, Olivier Elshocht, Jeroen Smeets, and Pan Li. GAD-NR: graph anomaly detection via neighborhood reconstruction. In Luz Angelica Caudillo-Mata, Silvio Lattanzi, Andrés Muñoz Medina, Leman Akoglu, Aristides Gionis, and Sergei Vassilvitskii (eds.), *Proceedings of the 17th ACM International Conference on Web Search and Data Mining, WSDM 2024, Merida, Mexico, March 4-8, 2024*, pp. 576–585. ACM, 2024. doi: 10.1145/3616855.3635767. URL https://doi.org/10.1145/3616855.3635767.

Patricia Iglesias Sánchez, Emmanuel Müller, Fabian Laforet, Fabian Keller, and Klemens Böhm. Statistical selection of congruent subspaces for mining attributed graphs. In Hui Xiong, George Karypis, Bhavani Thuraisingham, Diane J. Cook, and Xindong Wu (eds.), *2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7-10, 2013*, pp. 647–656. IEEE Computer Society, 2013. doi: 10.1109/ICDM.2013.88. URL https://doi.org/10.1109/ICDM.2013.88.

Zhiming Xu, Xiao Huang, Yue Zhao, Yushun Dong, and Jundong Li. Contrastive attributed network anomaly detection with data augmentation. In João Gama, Tianrui Li, Yang Yu, Enhong Chen, Yu Zheng, and Fei Teng (eds.), *Advances in Knowledge Discovery and Data Mining - 26th Pacific-Asia Conference, PAKDD 2022, Chengdu, China, May 16-19, 2022, Proceedings, Part II*, Lecture Notes in Computer Science, pp. 444–457. Springer, 2022. doi: 10.1007/978-3-031-05936-0_35. URL https://doi.org/10.1007/978-3-031-05936-0_35.

Tong Zhao, Chuchen Deng, Kaifeng Yu, Tianwen Jiang, Daheng Wang, and Meng Jiang. Error-bounded graph anomaly loss for gnns. In Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (eds.), *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 1873–1882. ACM, 2020. doi: 10.1145/3340531.3411979. URL https://doi.org/10.1145/3340531.3411979.

Yunfeng Zhao, Yixin Liu, Shiyuan Li, Qingfeng Chen, Yu Zheng, and Shirui Pan. Freegad: A training-free yet effective approach for graph anomaly detection. In Meeyoung Cha, Chanyoung Park, Noseong Park, Carl Yang, Senjuti Basu Roy, Jessie Li, Jaap Kamps, Kijung Shin, Bryan Hooi, and Lifang He (eds.), *Proceedings of the 34th ACM International Conference on Information and Knowledge Management, CIKM 2025, Seoul, Republic of Korea, November 10-14, 2025*, pp. 4379–4389. ACM, 2025. doi: 10.1145/3746252.3761125. URL https://doi.org/10.1145/3746252.3761125.

Lecheng Zheng, John R. Birge, Haiyue Wu, Yifang Zhang, and Jingrui He. Cluster aware graph anomaly detection. In Guodong Long, Michale Blumestein, Yi Chang, Liane Lewin-Eytan, Zi Helen Huang, and Elad Yom-Tov (eds.), *Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025- 2 May 2025*, pp. 1771–1782. ACM, 2025. doi: 10.1145/3696410.3714575. URL https://doi.org/10.1145/3696410.3714575.

Shuang Zhou, Xiao Huang, Ninghao Liu, Qiaoyu Tan, and Fu-Lai Chung. Unseen anomaly detection on networks via multi-hypersphere learning. In Arindam Banerjee, Zhi-Hua Zhou, Evangelos E. Papalexakis, and Matteo Riondato (eds.), *Proceedings of the 2022 SIAM International Conference on Data Mining, SDM 2022, Alexandria, VA, USA, April 28-30, 2022*, pp. 262–270. SIAM, 2022. doi: 10.1137/1.9781611977172.30. URL https://doi.org/10.1137/1.9781611977172.30.

A Additional Experimental Details

A.1 Evaluation Protocol

Results obtained from our runs are reported over five random seeds, $\{0, 1, 2, 3, 42\}$. For each run, model training, latent-reference generation, graph score smoothing, and label-free score orientation correction are completed before ground-truth labels are accessed. The labels are used only to compute the final evaluation metrics, including AUROC and average precision (AP). They are not used to select checkpoints, random seeds, score directions, Euler integration steps, or candidate configurations.

The main experiments use one Euler integration step and one independently trained model per random seed. Separate checkpoint directories are maintained for different datasets and seeds. The same fixed inference-noise seed is used after training so that the anomaly scores produced by a trained checkpoint do not depend on the number of random values consumed during early-stopped training.

For baseline methods evaluated by us, we use their official implementations when available and disable label-dependent model selection or score post-processing. Results taken directly from prior work are explicitly identified as published results and are not described as reproduced five-seed results.

A.2 Model Architecture

The graph autoencoder consists of a two-layer GCN encoder, a two-layer GCN attribute decoder, and a dot-product structure decoder. Given an input feature dimension F, the embedding dimension is set to

$$d = 2^{\lfloor \log_2 F \rfloor - 1},$$

unless an explicit hidden dimension is provided. Concatenating the embedding and scaled residual components produces a $2d$-dimensional flow latent representation. Before flow-matching training and inference, each latent dimension is standardized using graph-level statistics and clipped to the interval $[-10, 10]$ for numerical stability.

The free and prototype-conditioned velocity fields use the same MLP backbone. The $2d$-dimensional latent state is projected to 512 dimensions and combined with a 512-dimensional sinusoidal time embedding. The velocity MLP uses hidden dimensions

$$512 \to 1024 \to 1024 \to 512$$

with SiLU activations, followed by an output layer of dimension $2d$. For the prototype-conditioned velocity field, the graph-level prototype is projected into the same 512-dimensional hidden space and added as a context vector.

During free-field training, the latent references used to update the dominant-pattern prototype are generated using a short 10-step Euler solver. This internal prototype-estimation procedure is distinct from test-time contrastive generation, for which the main configuration uses one Euler step. The prototype similarity temperature is fixed to $\tau_p = 5$.

Let $\phi_p(\mathbf{c})$ denote the learned projection of the graph-level prototype. At inference time, the hidden representation provided to the prototype-conditioned velocity network is

$$\mathbf{h}_{i,t}^{\text{proto}} = \phi_z(\mathbf{z}_{i,t}) + \phi_t(t) + \alpha_p \phi_p(\mathbf{c}),$$

whereas prototype-conditioned flow training uses the unscaled term $\phi_p(\mathbf{c})$. Thus, α_p controls the inference-time strength of the projected prototype context.

A.3 Optimization Details

The graph autoencoder is trained for at most 300 epochs using Adam. Its dataset-specific initial learning rate and reconstruction weight are listed in Table 5. A weight decay of 0.01 is used for graph autoencoder training.

Dataset	#Nodes	#Edges	#Features	#Anomalies	Anomaly Ratio
Books	1,418	3,695	21	28	1.97%
Disney	124	335	28	6	4.84%
Enron	13,533	176,987	18	5	0.04%
Reddit	10,984	168,016	64	366	3.33%
Weibo	8,405	407,963	400	868	10.33%

Table 4: Statistics of the five graph anomaly detection benchmark datasets. Anomaly Ratio denotes the proportion of labeled anomalous nodes in each dataset. Ground-truth anomaly labels are used only for evaluation and descriptive dataset statistics.

Dataset	α_{AE}	AE Dropout	AE Learning Rate	Prototype Time Sampling	Inference Steps
Books	0.750	0.250	0.015	Logit-normal	1
Disney	0.800	0.300	0.050	Logit-normal	1
Enron	1.000	0.200	0.003	Uniform	1
Reddit	1.000	0.300	0.003	Uniform	1
Weibo	1.000	0.100	0.005	Uniform	1

Table 5: Dataset-specific optimization settings used in the reported experiments. The free velocity field uses uniform time sampling, whereas the table reports the sampling distribution used for the prototype-conditioned velocity field.

The free and prototype-conditioned velocity fields are each trained for at most 800 epochs using Adam with an initial learning rate of 0.0025 and no weight decay. All training stages use early stopping with a patience of 100 epochs based only on their corresponding unsupervised objectives: $\mathcal{L}_{\mathrm{AE}}$, $\mathcal{L}_{\mathrm{free}}$, and $\mathcal{L}_{\mathrm{proto}}$. The learning rate is multiplied by 0.5 every 100 epochs. Gradient norms are clipped to 1.0 for the graph autoencoder and 0.5 for the velocity networks.

All experiments conducted by us use PyTorch 2.4.1 and PyTorch Geometric 2.6.1 and are run on a single NVIDIA GeForce RTX 4090 GPU.

A.4 Label-Free Configuration Selection

Following the label-free model-selection principle of AutoGAD (Li et al., 2025b), we perform a sequential one-factor-at-a-time search over manually specified discrete candidate sets. At each search stage, one hyperparameter or component setting is varied while the remaining settings are fixed. Each candidate is trained using selection seeds $\{0, 42\}$, and its Contrast Score Margin (CSM) is computed from the resulting anomaly-score vectors without accessing ground-truth labels. The candidate with the highest mean CSM over the two selection seeds is retained before proceeding to the next search stage.

Formally, for the candidate set $\mathcal{C}_q$ of the hyperparameter currently being considered, the selected value is

$$c_q^* = \arg\max_{c \in \mathcal{C}_q} \frac{1}{|\mathcal{S}_{\mathrm{sel}}|} \sum_{s \in \mathcal{S}_{\mathrm{sel}}} \mathrm{CSM}\left(\mathbf{s}_{c,s}\right), \qquad \mathcal{S}_{\mathrm{sel}} = \{0, 42\},$$

where $\mathbf{s}_{c,s}$ denotes the final anomaly-score vector obtained using candidate c and seed s. After all search stages are completed, the resulting dataset-specific configuration is fixed and evaluated using the five seeds $\{0, 1, 2, 3, 42\}$.

The candidate sets are intentionally coarse and non-uniform and are specified before any label-based evaluation. They are manually defined according to the numerical scales and operational roles of the corresponding parameters, covering inactive, moderate, and stronger settings where applicable. The same predefined candidate sets are used throughout the label-free selection procedure and are not revised after observing AUROC, AP, or other label-based results. This computationally constrained OFAT procedure avoids evaluating the full Cartesian product of all hyperparameters. Candidate configurations are compared solely using the label-free CSM criterion; ground-truth labels, AUROC, AP, and other label-based metrics are used only for final evaluation.

Hyperparameter or component	Candidate settings
Residual scale λ	$\{5, 10, 15, 20, 25\}$
Contrastive guidance weight w	$\{0, 0.5, 0.75, 1.0, 1.25, 1.5, 2.2, 2.5\}$
Prototype-context scale α_p	$\{0.001, 0.003, 0.005, 0.01\}$
Graph score smoothing	$\{0.15, 0.25, 0.3, 0.35, 0.45, 0.50, 0.55, 0.60\}$
Rank-agreement threshold δ	$\{0.60, 0.65, 0.70, 0.80\}$
Flow-ranking fusion weight η	$\{0.70, 0.80, 0.90, 0.95\}$
Virtual-neighbor configuration	Off; $(K, k) \in \{(3, 5), (3, 6), (4, 6), (5, 5), (5, 8)\}$

Table 6: Discrete candidate settings considered during sequential label-free configuration selection. Here, K is the degree threshold for virtual-neighbor supplementation and k is the corresponding embedding-space retrieval bound. Virtual-neighbor settings are evaluated as predefined (K, k) pairs rather than as their Cartesian product.

Dataset	K	k	λ	α_p	w	μ	δ	η
Books	5	5	15	0.001	1.250	0.450	0.700	0.950
Disney	4	6	5	0.003	2.500	0.350	0.700	0.950
Enron	5	5	15	0.010	0.000	0.150	0.700	0.950
Reddit	5	5	10	0.005	0.000	0.300	0.650	0.950
Weibo	5	5	20	0.001	0.000	0.450	0.700	0.950

Table 7: Dataset-specific configurations used for the reported main results. Here, K is the target neighborhood-size threshold, and k is the maximum number of embedding-space candidates considered for virtual-neighbor retrieval, λ is the residual scale, α_p is the inference-time prototype-context scale, w is the contrastive guidance weight, μ is the graph score-smoothing weight, δ is the rank-agreement threshold, and η is the fusion weight assigned to the oriented flow ranking.

For Enron, Reddit, and Weibo, the selected guidance weight is $w = 0$. Under this setting, the contrastive velocity reduces to the free velocity field, and the prototype-conditioned field does not contribute to test-time latent-reference generation. Consequently, the value of α_p does not affect the final inference scores on these three datasets, although it remains part of the stored dataset configuration.

Implementation detail of virtual-neighbor retrieval. For a node with degree $d_i < K$, we retrieve the top k eligible non-neighbor candidates according to their embedding-space cosine similarity and select the top $K - d_i$ candidates as virtual neighbors, so that the supplemented neighborhood contains K nodes. Therefore, K determines the target neighborhood size, whereas k only controls the maximum size of the candidate pool. Since all reported configurations satisfy $k \geq K$, the actual number of supplemented virtual neighbors is $K - d_i$, consistent with the definition in the main text.

A.5 Runtime Measurement

The end-to-end runtime experiment uses seed 42 and is executed sequentially on the same NVIDIA GeForce RTX 4090 GPU. Each method–dataset pair is repeated three times. The reported runtime includes data loading, model training, inference, anomaly-score generation, score post-processing, and metric computation.

The Euler-step analysis reports inference-only runtime separately from training. Its timing interval covers contrastive latent-reference generation, graph autoencoder decoding, reconstruction score calculation, graph score smoothing, and label-free score orientation. The reported inference time is averaged over the same five random seeds used for the corresponding AUROC results.

B Additional AP Results

Table 8 reports the complete average precision (AP) results on the five benchmark datasets.

Compared with DiffGAD, RALFlow-GAD increases average AP from 0.095 to 0.106, corresponding to an improvement of approximately 11.5%. The improvement is driven primarily by Disney,

Method	Books	Disney	Enron	Reddit	Weibo	Avg.
Graph-agnostic methods						
LOF	0.015 ± 0.000	0.052 ± 0.000	0.000 ± 0.000	$\underline{0.042 \pm 0.000}$	0.158 ± 0.000	0.053
IF	0.019 ± 0.002	0.101 ± 0.045	0.001 ± 0.000	0.028 ± 0.001	0.129 ± 0.026	0.056
MLPAE	0.018 ± 0.003	0.059 ± 0.008	0.001 ± 0.000	0.034 ± 0.000	0.528 ± 0.099	0.128
Classical graph anomaly detectors						
SCAN	0.020 ± 0.001	0.050 ± 0.003	0.000 ± 0.000	0.033 ± 0.000	0.173 ± 0.034	0.055
Radar	0.022 ± 0.000	0.072 ± 0.000	$\underline{0.002 \pm 0.000}$	0.036 ± 0.002	$\mathbf{0.921 \pm 0.007}$	**0.211**
ANOMALOUS	0.022 ± 0.000	0.072 ± 0.000	$\underline{0.002 \pm 0.000}$	0.040 ± 0.006	$\mathbf{0.921 \pm 0.007}$	**0.211**
Deep graph anomaly detectors						
GCNAE	0.021 ± 0.004	0.048 ± 0.007	0.001 ± 0.000	0.034 ± 0.000	0.708 ± 0.050	0.162
DOMINANT	0.022 ± 0.006	0.076 ± 0.050	0.001 ± 0.001	0.037 ± 0.000	0.180 ± 0.102	0.063
DONE	0.018 ± 0.003	0.050 ± 0.007	0.001 ± 0.000	0.037 ± 0.004	0.655 ± 0.134	0.152
AdONE	0.025 ± 0.003	0.061 ± 0.015	0.001 ± 0.000	0.033 ± 0.004	0.629 ± 0.095	0.150
AnomalyDAE	$\underline{0.035 \pm 0.014}$	0.057 ± 0.002	0.001 ± 0.000	0.037 ± 0.001	0.385 ± 0.225	0.103
GAAN	0.026 ± 0.008	0.056 ± 0.000	0.001 ± 0.000	0.037 ± 0.001	$\underline{0.803 \pm 0.002}$	$\underline{0.185}$
CONAD	0.025 ± 0.008	0.060 ± 0.014	0.001 ± 0.000	0.037 ± 0.003	0.156 ± 0.069	0.056
Representative recent methods						
CAGAD	0.022 ± 0.003	0.063 ± 0.008	0.000 ± 0.000	0.040 ± 0.001	0.187 ± 0.004	0.062
TAM	0.022 ± 0.002	0.075 ± 0.004	0.000 ± 0.000	$\mathbf{0.044 \pm 0.001}$	0.236 ± 0.002	0.075
SmoothGNN	0.020 ± 0.002	0.074 ± 0.013	0.001 ± 0.000	0.027 ± 0.006	0.222 ± 0.081	0.069
FreeGAD	0.023 ± 0.000	0.072 ± 0.000	0.001 ± 0.000	0.033 ± 0.000	0.022 ± 0.000	0.030
GAD-NR	0.019 ± 0.004	$\underline{0.108 \pm 0.017}$	0.001 ± 0.000	0.039 ± 0.003	0.135 ± 0.091	0.060
DiffGAD	$\mathbf{0.090 \pm 0.044}$	0.061 ± 0.002	0.001 ± 0.000	0.038 ± 0.000	0.283 ± 0.086	0.095
RALFlow-GAD	0.032 ± 0.003	$\mathbf{0.112 \pm 0.048}$	$\mathbf{0.003 \pm 0.002}$	0.039 ± 0.003	0.341 ± 0.000	0.106

Table 8: Average precision comparison on five benchmarks. Results for LOF, IF, MLPAE, SCAN, Radar, ANOMALOUS, GCNAE, DOMINANT, DONE, AdONE, AnomalyDAE, GAAN, CONAD, and DiffGAD are taken from Li et al. (2025a). The remaining results are obtained from our runs over five random seeds. Avg. denotes the arithmetic mean of the five dataset-wise mean AP values. All displayed values are rounded to three decimal places, while averages and rankings are determined using unrounded means. Best and second-best results are highlighted in bold and underlined, respectively.

Enron, and Weibo. Specifically, RALFlow-GAD improves AP from 0.061 to 0.112 on Disney, from 0.001 to 0.003 on Enron, and from 0.283 to 0.341 on Weibo. It also obtains a slightly higher AP on Reddit, increasing from 0.038 to 0.039. On Books, however, DiffGAD remains substantially stronger, achieving 0.090 compared with 0.032 for RALFlow-GAD.

Across the complete table, RALFlow-GAD achieves the highest AP on Disney and Enron. TAM performs best on Reddit, while DiffGAD performs best on Books. Radar and ANOMALOUS obtain substantially higher AP on Weibo, which also gives them the highest five-dataset average AP. These results show that the advantage of RALFlow-GAD is dataset-dependent rather than uniform across all benchmarks.

The comparison should also be interpreted with care because the table combines published results from prior work with results obtained from our runs. Differences in implementations, random seeds, and configuration protocols prevent the complete table from constituting a fully controlled leaderboard. The direct comparison with DiffGAD is reported to complement the AUROC results, as DiffGAD is the strongest baseline by average AUROC in the main experiment.

C Additional Ablation Studies

This section provides additional controlled analyses of residual construction, numerical integration, and score orientation correction. Unless otherwise stated, results are reported as mean AUROC over five random seeds. All applicable components and settings are held fixed within each controlled comparison.

Fine-grained analysis of residual construction. Table 9 separately evaluates the global and local residuals, a fixed average of the two residuals, and the proposed degree-adaptive fusion. This controlled experiment uses a common auxiliary configuration for all variants and is intended to isolate residual construction rather than reproduce the dataset-specific full-model results in the main table.

The global and local residual variants both achieve an average AUROC of 0.719, improving over the embedding-only average of 0.704 by 0.015. Their relative effectiveness remains dataset-dependent: the global residual performs better on Disney and Reddit, whereas the local residual performs better on Books and Enron.

A fixed average of the two residuals achieves an average AUROC of 0.711, which is lower than either individual residual variant. In contrast, degree-adaptive fusion achieves the highest average AUROC of 0.730, improving over the embedding-only and fixed-average variants by 0.026 and 0.019, respectively. It obtains the highest mean AUROC on Books, Disney, Enron, and Reddit, while all variants produce the same displayed result on Weibo. These results support adapting the relative contributions of global and local residuals rather than combining them uniformly.

Variant	Books	Disney	Enron	Reddit	Weibo	Avg.
Embedding Only	0.602 ± 0.026	0.591 ± 0.113	0.815 ± 0.040	0.566 ± 0.018	0.944 ± 0.000	0.704
Global Residual Only	0.611 ± 0.038	$\underline{0.632 \pm 0.087}$	0.842 ± 0.018	$\underline{0.568 \pm 0.020}$	0.944 ± 0.000	$\underline{0.719}$
Local Residual Only	$\underline{0.615 \pm 0.021}$	0.622 ± 0.073	$\underline{0.846 \pm 0.013}$	0.565 ± 0.020	0.944 ± 0.000	0.719
Fixed Average Fusion	0.614 ± 0.028	0.593 ± 0.107	0.838 ± 0.024	0.565 ± 0.016	0.944 ± 0.000	0.711
Degree-Adaptive Fusion	$\mathbf{0.624 \pm 0.016}$	$\mathbf{0.663 \pm 0.088}$	$\mathbf{0.847 \pm 0.012}$	$\mathbf{0.573 \pm 0.016}$	0.944 ± 0.000	$\mathbf{0.730}$

Table 9: Fine-grained ablation of residual construction. Results are reported as mean AUROC $\pm$ standard deviation over five random seeds. All applicable components and settings are held fixed within this controlled comparison. Avg. denotes the arithmetic mean of the five dataset-wise mean AUROC values and is computed from unrounded results. Bold and underlined values denote the best and second-best unrounded means, respectively. Emphasis is omitted for Weibo because all variants produce the same displayed result.

Effect of inference-time Euler integration steps. Table 10 evaluates the effect of the number of Euler integration steps used for inference. Figure 3(c) visualizes the same controlled Euler-step comparison. The same trained models and controlled experimental configuration are used for all step counts; only the inference-time numerical integration budget is changed. Each result is averaged over the same five random seeds.

One-step inference achieves the highest average AUROC of 0.730. Increasing the number of integration steps to 2, 4, and 8 produces average AUROC values of 0.714, 0.712, and 0.712, respectively. Thus, increasing the numerical integration budget does not improve the overall detection performance under the learned straight probability path.

The effect is dataset dependent. Two Euler steps achieve the highest AUROC on Disney, while eight steps perform best on Enron. In contrast, one step performs best on Books and Reddit. All evaluated step counts produce the same displayed AUROC of 0.944 on Weibo. Overall, one-step inference provides the strongest average performance while requiring the fewest velocity-field evaluations, supporting its use as the default inference configuration.

(a) Detection Performance (AUROC)

Steps	Books	Disney	Enron	Reddit	Weibo	Avg.
1 (default)	$\mathbf{0.624 \pm 0.016}$	$\underline{0.663 \pm 0.088}$	$\underline{0.847 \pm 0.012}$	$\mathbf{0.573 \pm 0.016}$	0.944 ± 0.000	$\mathbf{0.730}$
2	$\underline{0.606 \pm 0.026}$	$\mathbf{0.675 \pm 0.084}$	0.811 ± 0.032	$\underline{0.535 \pm 0.017}$	0.944 ± 0.000	$\underline{0.714}$
4	0.602 ± 0.026	0.656 ± 0.100	0.836 ± 0.010	0.520 ± 0.010	0.944 ± 0.000	0.712
8	0.589 ± 0.044	0.659 ± 0.095	$\mathbf{0.852 \pm 0.018}$	0.518 ± 0.008	0.944 ± 0.000	0.712

(b) Inference Time (ms)

Steps	Books	Disney	Enron	Reddit	Weibo	Avg.
1 (default)	$\mathbf{47.4 \pm 1.1}$	$\mathbf{9.4 \pm 0.5}$	$\mathbf{445.0 \pm 16.9}$	$\mathbf{293.1 \pm 15.4}$	$\mathbf{284.0 \pm 14.5}$	$\mathbf{215.8}$
2	$\underline{48.5 \pm 1.5}$	$\underline{9.7 \pm 0.1}$	$\underline{449.4 \pm 5.2}$	$\underline{304.0 \pm 12.2}$	$\underline{287.5 \pm 11.8}$	$\underline{219.8}$
4	49.9 ± 1.7	10.8 ± 0.2	463.0 ± 11.0	308.4 ± 10.4	289.9 ± 1.2	224.4
8	53.0 ± 2.1	12.8 ± 0.5	482.6 ± 11.1	327.4 ± 9.6	302.6 ± 9.1	235.7

Table 10: Effect of the number of inference-time Euler integration steps. Panel (a) reports dataset-wise AUROC as mean $\pm$ standard deviation over five random seeds. Avg. denotes the arithmetic mean of the five dataset-wise mean AUROC values. Panel (b) reports inference time in milliseconds as mean $\pm$ standard deviation over the same five random seeds. For AUROC, higher values are better; for inference time, lower values are better. Bold and underlined values denote the best and second-best results, respectively. Emphasis is omitted for Weibo AUROC because all step counts produce the same displayed result.

We note that all non-step hyperparameters are fixed to the dataset-specific configurations selected for the default one-step setting. Therefore, this controlled analysis isolates the effect of the numerical integration budget rather than comparing separately optimized configurations for each step count. Retuning the remaining hyperparameters may change the relative performance of multi-step inference.

Fine-grained analysis of score orientation correction. Table 11 isolates the two operations in the score orientation correction module: label-free ranking orientation and fusion with the local inconsistency prior. The evaluated variants are defined as follows. Flow Ranking Only directly uses $\mathbf{r}^{\mathrm{flow}}$. Orientation Only uses $\mathbf{r}^{\mathrm{ori}}$, where the local prior is used only to determine whether the flow ranking should be retained or reversed. Fusion without Orientation combines the unmodified flow ranking and the prior ranking as

$$\eta \mathbf{r}^{\mathrm{flow}} + (1 - \eta)\mathbf{r}^{\mathrm{prior}}.$$

Local Prior Only uses $\mathbf{r}^{\mathrm{prior}}$, whereas the full model combines the oriented flow ranking and prior ranking according to Eq. (15).

Using the flow ranking directly yields an average AUROC of 0.451. Adding prior fusion without first correcting the score direction increases the average AUROC to 0.459. In contrast, applying label-free orientation without final prior fusion increases the average AUROC to 0.727. This shows

that determining whether to retain or reverse the flow-induced ranking is the principal contribution of the score orientation correction module.

The complete formulation obtains the highest average AUROC of 0.730. Compared with Orientation Only, prior fusion provides a modest average improvement of 0.003, improving performance on Disney and Reddit, slightly decreasing performance on Enron, and leaving Books and Weibo unchanged at the displayed precision. The local prior alone obtains an average AUROC of 0.592, indicating that it provides useful graph-specific information but does not replace the learned flow-based detector. Overall, score orientation supplies the dominant improvement, while prior fusion offers a smaller and dataset-dependent refinement.

Variant	Books	Disney	Enron	Reddit	Weibo	Avg.
Flow Ranking Only	0.376	0.342	0.150	0.442	**0.944**	0.451
Orientation Only	**0.624**	<u>0.658</u>	**0.850**	0.558	0.944	<u>0.727</u>
Fusion without Orientation	0.377	0.348	0.147	0.479	0.944	0.459
Local Prior Only	<u>0.597</u>	0.596	0.363	**0.591**	0.814	0.592
Full RALFlow-GAD	**0.624**	**0.663**	<u>0.847</u>	<u>0.573</u>	0.944	**0.730**

Table 11: Fine-grained ablation of label-free score orientation and local-prior fusion. Results are reported as mean AUROC over five random seeds. All other components and dataset-specific settings are held fixed. Avg. denotes the arithmetic mean of the five dataset-wise mean AUROC values. Bold and underlined values denote the best and second-best results, respectively.

D Additional Hyperparameter and Architecture Analysis

This section presents additional controlled analyses of six factors related to numerical integration, score orientation correction, stochastic inference, latent capacity, and graph autoencoder architecture. For each panel, we vary one factor at a time while keeping all remaining settings unchanged. Each point reports the mean AUROC over five random seeds.

These experiments are conducted as post-hoc analyses. Ground-truth labels and AUROC are used only to report the resulting sensitivity curves and are not used for training, configuration selection, score orientation, or checkpoint selection. Because the experiments are designed to isolate individual factors, their absolute AUROC values should not be interpreted as reproductions of the dataset-specific main configurations.

Oriented-flow fusion weight. Figure 3(a) examines the fusion weight η in Eq. (15). Recall that η controls the contribution of the oriented flow ranking, whereas $1 - \eta$ controls the contribution of the local inconsistency-prior ranking. The effect of η is strongly dataset dependent. Increasing η substantially improves performance on Enron and Weibo, indicating that the oriented flow ranking provides the dominant signal on these datasets. Disney instead favors an intermediate weight, suggesting that combining the flow ranking with the local prior is more beneficial in this case. Books remains comparatively stable across the evaluated range, while Reddit exhibits a moderate decrease at larger values of η.

These observations are consistent with the fine-grained ablation of score orientation correction: score orientation provides the principal performance improvement, whereas fusion with the local prior acts as a smaller, dataset-dependent refinement. The results also show that neither the learned flow score nor the local prior is uniformly dominant across all graph distributions.

Number of inference noise draws. Figure 3(b) varies the number of independent Gaussian noise draws M used during inference. For $M > 1$, multiple contrastive latent-reference sets are generated from independently sampled base noise, and their resulting anomaly scores are aggregated before the final ranking is produced.

Increasing M does not provide a consistent improvement. Weibo and Enron remain nearly unchanged over the evaluated range, while Books, Disney, and Reddit exhibit modest non-monotonic variations. In particular, larger sampling budgets do not consistently outperform small values of M.

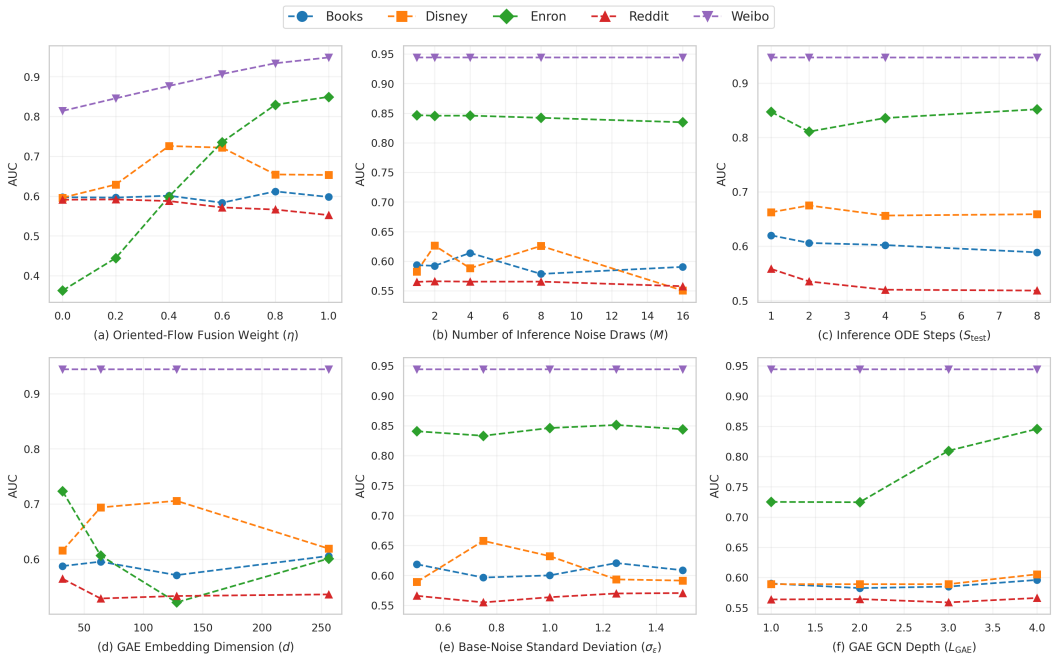

Figure 3: Additional hyperparameter and architecture analyses of RALFlow-GAD on five graph anomaly detection datasets. Each point reports the mean AUROC over five random seeds. We vary one factor at a time while keeping all remaining settings unchanged. Panel (a) varies the fusion weight assigned to the oriented flow ranking. Panel (b) varies the number of independent inference-time noise draws. Panel (c) varies the number of inference-time Euler integration steps. Panel (d) varies the GAE embedding dimension; the corresponding residual-augmented flow latent has dimension $2d$. Panel (e) varies the standard deviation of the Gaussian base noise. Panel (f) varies the GCN depth of the graph autoencoder. Ground-truth labels and AUROC are used only for post-hoc reporting and are not used for configuration selection.

This result indicates that the proposed one-step flow inference does not depend on extensive Monte Carlo sampling and that a small number of noise draws provides a favorable effectiveness–efficiency trade-off.

GAE embedding dimension. Figure 3(d) studies the GAE embedding dimension d. Since the final residual-augmented latent representation concatenates the embedding and residual components, its dimension is $2d$.

The effect of latent capacity varies substantially across datasets. Weibo is nearly insensitive to d, and Books changes only moderately. Disney performs best at an intermediate embedding dimension, whereas Enron exhibits a pronounced decrease at an intermediate setting followed by a partial recovery at the largest evaluated dimension. Reddit shows a smaller decrease as the embedding dimension increases. These results indicate that increasing latent capacity does not uniformly improve anomaly detection. A larger embedding space may provide greater representational capacity, but it may also make latent distribution estimation more difficult when the number of effective graph samples is limited.

Base-noise standard deviation. Figure 3(e) varies the base-noise standard deviation σ_ϵ, corresponding to

$$\boldsymbol{\epsilon}_i \sim \mathcal{N}\left(\mathbf{0}, \sigma_\epsilon^2 \mathbf{I}\right).$$

Performance is comparatively stable over the evaluated range. Enron and Weibo show only minor variations, while Books and Reddit are also relatively insensitive to the noise scale. Disney exhibits a larger but still non-monotonic variation and favors a slightly smaller standard deviation. Overall,

the standard Gaussian setting $\sigma_\epsilon = 1$ lies in a stable region and provides competitive performance without requiring dataset-specific rescaling of the base distribution.

GAE GCN depth. Figure 3(f) evaluates the number of GCN layers L_{GAE} in the graph autoencoder. Books, Disney, Reddit, and Weibo remain relatively stable as the depth increases, suggesting that additional message-passing layers provide limited benefits on these datasets. Enron, however, benefits substantially from a deeper GAE, demonstrating that the appropriate receptive-field size depends on the underlying graph distribution.

The absence of a uniform monotonic trend indicates that simply increasing GCN depth is not a generally reliable way to improve performance. Deeper propagation may capture broader structural context on some graphs, but it can also introduce additional smoothing and computational overhead. We therefore retain the predefined shallow GAE architecture in the main experiments rather than selecting the network depth according to label-based evaluation.

Overall observations. The additional analyses reveal two broad patterns. First, stochastic inference factors, including the number of noise draws and the base-noise standard deviation, are comparatively robust and do not require large sampling budgets or precise noise-scale tuning. Second, score fusion and architectural capacity exhibit stronger dataset-dependent effects. In particular, the preferred fusion weight, embedding dimension, and GAE depth vary across graph distributions.

These findings support the use of label-free, dataset-adaptive configuration for score-related parameters while retaining a compact default architecture and an efficient inference procedure. They also show that the performance of RALFlow-GAD does not rely on uniformly increasing model capacity or the number of stochastic samples.