BNFLOW: ADAPTIVE FLOW GUIDANCE FOR GRAPH-LEVEL ANOMALY DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph-level anomaly detection (GLAD) identifies graphs that deviate from a normal population. With only normal graphs available for training, existing methods typically characterize normal patterns or generate pseudo anomalies for negative supervision. However, generated negatives can be too dissimilar to normal graphs or become too easy as the detector improves, limiting their usefulness for detecting subtle anomalies. To address this, we propose BNFlow, a guided flow matching framework that combines a normal graph reference with adaptive sample refinement. Flow matching learns transport toward normal graph representations, and detector feedback refines intermediate samples to make them more challenging as negatives. Refinement is bounded within each trajectory, and its strength adapts to the detector's ability to distinguish normal and generated graphs. Continued flow matching regularizes the generator toward the normal reference during joint learning. Our theoretical analysis characterizes the local effect of guidance and bounds deviation from normal references. Experiments on eight graph benchmarks demonstrate superior overall performance over state-of-the-art GLAD methods. Further analyses highlight the complementary roles of normal transport and detector guidance in providing effective negative supervision.

1 INTRODUCTION

Graph-level anomaly detection (GLAD) identifies entire graphs whose attributes or structures depart from the patterns of a normal population. It has wide applications in molecular screening, biological networks, and social network analysis (Akoglu et al., 2015; Liu et al., 2023c). Since anomalous examples are often scarce and difficult to collect, graph anomaly detectors generally learn from normal graphs alone (Ma et al., 2022; Cai et al., 2025a). Detecting subtle anomalies requires distinguishing abnormal changes from the variation among normal graphs. An abnormal graph may differ in only a local substructure or a few node attributes while retaining much of the normal graph context (Ma et al., 2022; Niu et al., 2023). Therefore, the detector must recognize subtle departures from normality without access to representative anomalous graphs during training.

Existing approaches learn normal graph representations through one-class objectives and graph transformations (Ruff et al., 2018; Qiu et al., 2022). Distillation and reconstruction assess deviations through prediction errors and reconstruction statistics (Ma et al., 2022; Niu et al., 2023; Kim et al., 2024). Contrastive and subgraph learning use consistency across graph views (Luo et al., 2022; Liu et al., 2023b;c), while prototypes summarize distinct normal patterns (Zhang et al., 2026; Zhao et al., 2026). A complementary direction constructs synthetic negatives through constrained perturbations and feature synthesis (Goyal et al., 2020; Du et al., 2022; Tao et al., 2023). Recent GLAD methods develop this idea by jointly learning graph generation and discrimination (Cai et al., 2025a;b). These generated graphs provide explicit negative supervision to help the detector recognize departures from normality.

Useful pseudo anomalies should preserve normal graph patterns while remaining challenging yet distinguishable from normal samples. Unconstrained graph perturbations can disrupt semantics (Xiao et al., 2024). Normal conditioning and early stopping help control similarity to normal samples (Cai et al., 2025b; 2026). However, a negative that initially challenges the detector can become easy as training proceeds. When generation and detection share a classification objective (Cai et al., 2025b), easier separation reduces the loss for both networks. Joint training can therefore favor

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.

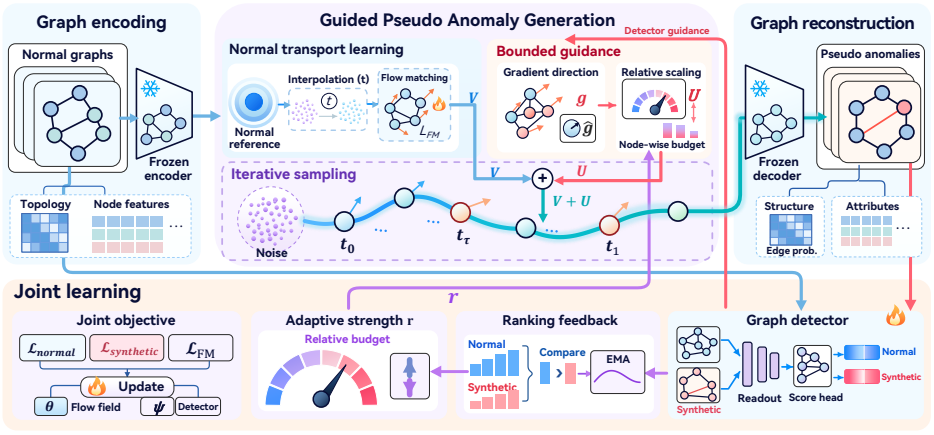


Figure 1: Overview of BNFlow. Normal latent transport is refined by detector guidance with a relative velocity budget. Ranking feedback adapts guidance strength, while joint learning updates the flow field and detector under continued flow-matching regression.

samples that provide limited additional supervision. Constrained adversarial synthesis uses detector feedback to seek challenging negatives (Goyal et al., 2020). For GLAD, the challenge is to coordinate this feedback with a normal graph reference throughout training. These observations raise a central question: *How can we generate pseudo anomalies that remain informative as the detector evolves?*

To address this question, we propose **BNFlow**, a guided flow matching framework that couples normal transport with adaptive sample refinement (Figure 1). Flow matching learns transport from noise toward normal latent representations supplied by a frozen graph autoencoder (Lipman et al., 2023; Tong et al., 2024). Its intermediate states allow the detector to guide generation, while its velocity provides a scale for controlling each refinement. With the networks held fixed during sampling, detector gradients guide these states toward higher normality scores, locally increasing their difficulty as negatives. A relative velocity budget bounds each refinement, and ranking feedback adjusts guidance strength according to how well the detector separates normal and generated graphs. The resulting negatives then train the generator and detector jointly through classification. During these updates, flow matching regularizes the velocity field toward normal reference directions. We summarize the contributions of this work as follows.

- We introduce BNFlow, a guided flow matching framework for GLAD that couples normal transport with adaptive negative supervision through bounded detector guidance.
- We prove that guidance maximizes the first-order increase in classification loss on synthetic negatives under its velocity budget. We also bound deviation from normal latent references in terms of truncation, velocity error, and accumulated guidance.
- Experiments on diverse graph benchmarks show that BNFlow outperforms state-of-the-art GLAD baselines. Further analysis highlights the complementary roles of normal transport and detector guidance in providing effective negative supervision.

2 Related Work

2.1 Graph-Level Anomaly Detection

GLAD combines graph representations with a criterion for abnormality. Graph kernels supply structural features for classical detectors (Shervashidze et al., 2011), whose performance depends on the geometry induced by the representation (Zhao & Akoglu, 2023). Deep compactness objectives (Ruff et al., 2018) and learned graph transformations in OCGTL (Qiu et al., 2022) adapt that geometry to normal observations. GLocalKD distills node and graph representations (Ma et al., 2022), HimNet reconstructs graphs using memories at both scales (Niu et al., 2023), and MUSE addresses failures

of mean reconstruction error by using multiple reconstruction statistics (Kim et al., 2024). GLADC combines contrastive learning with reconstruction (Luo et al., 2022), while GOOD-D exploits inconsistency across representations for unsupervised graph OOD detection (Liu et al., 2023b). SIGNET extracts informative subgraphs (Liu et al., 2023c), and M-GLAD and ProtoGLAD compare observations with latent or explicit normal prototypes (Zhang et al., 2026; Zhao et al., 2026). Supervision settings also include labeled anomaly learning in iGAD (Zhang et al., 2022) and learning from contaminated training data (Yang et al., 2026). With normal training graphs, BNFlow complements representation learning by adapting the synthetic negatives used to train a discriminator.

2.2 Learning from Synthetic Anomalies

Negative supervision can come from auxiliary outlier data (Hendrycks et al., 2019) or from constructed examples. Confidence calibration with generated outliers (Lee et al., 2018) and DROCC's constrained adversarial search (Goyal et al., 2020) connect synthesis to a learned classifier. CutPaste constructs local visual defects (Li et al., 2021), while VOS and NPOS synthesize outlying features using parametric density models or neighbor geometry (Du et al., 2022; Tao et al., 2023). In GLAD, SDM jointly learns graph synthesis and discrimination, with both adversarial and nonadversarial variants (Cai et al., 2025a). MotifCAR constructs counterfactual graphs from motifs and contexts and refines their structure (Xiao et al., 2024). AGDiff combines latent diffusion with joint classification to generate graphs resembling normal observations (Cai et al., 2025b), and MoEGAD uses iterative generation, early stopping, and expert specialization (Cai et al., 2026). These works establish the value of adaptive synthesis and normal resemblance. BNFlow adds an explicit sampling mechanism to this line of work: discriminator gradients refine intermediate transport states, a relative velocity budget sets the refinement scale, and ranking feedback adjusts that scale across batches. Its distinction is the coordination of these operations with continued normal flow regression and joint classification.

2.3 Guided Generative Models for Graphs

Diffusion models learn generation through denoising (Ho et al., 2020; Song et al., 2021). GDSS models coupled node and edge dynamics (Jo et al., 2022), while DiGress uses discrete transitions for categorical graph variables (Vignac et al., 2023). Flow matching learns velocities along prescribed probability paths (Lipman et al., 2023; Tong et al., 2024), with related constructions in rectified flow and stochastic interpolants (Liu et al., 2023a; Albergo & Vanden-Eijnden, 2023). Discrete flow matching extends these ideas to categorical states (Gat et al., 2024), and DeFoG applies them to graph generation (Qin et al., 2025). Guidance provides additional sampling control through classifier gradients (Dhariwal & Nichol, 2021), conditional and unconditional predictions (Ho & Salimans, 2022), or optimization of source noise through the sampler (Ben-Hamu et al., 2024). General flow guidance also requires accounting for the chosen probability path (Feng et al., 2025). BNFlow uses flow regression as a normal latent reference for training an anomaly detector. Its evolving discriminator supplies a local refinement objective, and the analysis concerns the resulting sampling updates and reference deviation. This connects guided generation to the changing demands of synthetic negative supervision.

3 Problem Formulation and Preliminaries

Graph-level anomaly detection. Let $\mathcal{D}_{\mathrm{tr}} = \{G_i\}_{i=1}^{N}$ contain normal graphs sampled from P_{N}. An undirected graph $G = (\mathbf{X}, \mathbf{A})$ has attributes $\mathbf{X} \in \mathbb{R}^{n \times d_x}$ and symmetric adjacency $\mathbf{A} \in \{0,1\}^{n \times n}$, with varying n. We learn a score $a(G)$ that increases with anomalousness, using generated graphs for negative supervision.

Flow matching. For independent endpoints $\mathbf{z}_0 \sim p_0$ and $\mathbf{z}_1 \sim p_1$, the path $\mathbf{z}_t = (1-t)\mathbf{z}_0 + t\mathbf{z}_1$ has velocity $\mathbf{z}_1 - \mathbf{z}_0$. Squared error regression onto this target yields $\mathbb{E}[\mathbf{z}_1 - \mathbf{z}_0 \mid \mathbf{z}_t, t]$ at the unrestricted population optimum, assuming finite second moments (Lipman et al., 2023; Tong et al., 2024). BNFlow uses this construction in a fixed latent space.

4 METHODOLOGY

4.1 FLOW MATCHING WITH NORMAL GRAPH REFERENCES

Latent transport provides intermediate states for detector refinement, while normal flow regression supplies a reference throughout joint learning. Following the latent graph design of AGDiff (Cai et al., 2025b), we pretrain a variational graph autoencoder with a GIN encoder (Kipf & Welling, 2016; Xu et al., 2019). It maps G to node latents $\mathbf{Z}_G \in \mathbb{R}^{n\times d_z}$. Negative sampling selects candidate node pairs comprising observed edges and sampled non-edges, encoded by a symmetric binary mask $\mathbf{M} \in \{0,1\}^{n\times n}$, with each undirected pair stored in both directions. Conditioning on this support, the decoder reconstructs

$$\mathrm{dec}_\phi(\mathbf{Z}) = (\widehat{\mathbf{X}}, \widehat{\mathbf{A}}), \qquad \widehat{\mathbf{X}} = \mathrm{MLP}_\phi(\mathbf{Z}), \qquad \widehat{\mathbf{A}} = \mathbf{M} \odot \sigma(\mathbf{Z}\mathbf{Z}^\top). \tag{1}$$

Here σ is the elemental sigmoid. Only selected inner products are evaluated, and the decoded graph is stored sparsely on these pairs without constructing the full matrix. Pretraining combines attribute and sampled-pair adjacency reconstruction with a Gaussian prior penalty. We then freeze the encoder and decoder parameters ϕ and cache normal targets using the configured posterior means or samples. Freezing these mappings keeps the latent targets fixed while the flow and detector evolve. Appendix A.1 gives the objective and sampling conventions.

Let Q_ϕ^{pair} denote the joint distribution of a normal node target $\mathbf{z}_1$ and its condition $\mathbf{c}$. The condition is the corresponding target, its noisy version, or a null input. With independent $\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \sigma_0^2\mathbf{I})$ and $t \sim \mathcal{U}(0,1)$, we train

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{(\mathbf{z}_1,\mathbf{c}),\mathbf{z}_0,t} \left\| \mathbf{v}_\theta((1-t)\mathbf{z}_0 + t\mathbf{z}_1, t, \mathbf{c}) - (\mathbf{z}_1 - \mathbf{z}_0) \right\|_2^2. \tag{2}$$

A shared MLP with sinusoidal time embeddings defines the node velocities. During generation, a reference training graph supplies n and the conditions $\mathbf{C}$, which stay fixed along the trajectory. Applying the MLP to each row gives $\mathbf{V}_\theta(\mathbf{Z}, t, \mathbf{C})$. Conditions from the same graph retain its encoded relationships, while decoding and discriminator guidance further couple node updates.

With exact target conditioning, the ideal regression field is the analytic bridge $(\mathbf{c} - \mathbf{z})/(1-t)$ for $t < 1$. Learning the velocity allows classification to adapt this reference transport, while continued FM regression penalizes deviations on normal interpolation paths. Noisy conditions instead yield a conditional expectation over targets, and the unconditional configuration supplies a node marginal reference. These distinct roles of conditioning are derived in Appendix A.2. We pretrain the velocity and retain Eq. (2) throughout joint learning.

4.2 GUIDED PSEUDO ANOMALY GENERATION

Normal transport supplies a reference direction. To refine its samples for the current detector, let $D_\psi(G, \mathbf{Z})$ be a normality logit obtained by combining GIN representations and latent features with graph pooling and residual feedforward blocks (Cai et al., 2025b). The decoded graph defines

$$\ell_\psi(\mathbf{Z}) = D_\psi(\mathrm{dec}_\phi(\mathbf{Z}), \mathbf{Z}), \qquad \mathbf{g}(\mathbf{Z}) = \nabla_{\mathbf{Z}} \ell_\psi(\mathbf{Z}). \tag{3}$$

Decoded features and selected edge weights remain continuous during sampling. Gradients condition on the candidate support, with unselected pairs omitted from message passing. The derivative includes the decoder and direct latent branches, so a node's refinement reflects its effect on the graph prediction. Appendix A.3 makes these paths explicit.

Synthetic graphs serve as negative examples, so increasing their normality logits makes them harder for the detector to reject. We normalize the gradient to separate its direction from the logit scale and set its magnitude relative to the node's transport speed. This constrains the added velocity and gives no guidance to stationary nodes. Write $\nu(\mathbf{b}) = \mathbf{b}/\|\mathbf{b}\|_2$ for $\mathbf{b} \neq 0$ and $\nu(\mathbf{0}) = \mathbf{0}$. Set

$$\mathbf{U}_v = r\|\mathbf{V}_{\theta,v}\|_2 \nu(\mathbf{g}_v), \qquad r \geq 0. \tag{4}$$

The following result connects this choice directly to the loss used for negative supervision.

Proposition 1 (Local refinement of negative supervision)**.** *At a differentiable state, Eq. (4) maximizes* $\langle \mathbf{g}, \mathbf{B} \rangle_F$ *subject to* $\|\mathbf{B}_v\|_2 \leq r\|\mathbf{V}_{\theta,v}\|_2$ *for every node. It also maximizes the first order*

increase in $\mathcal{B}(\ell_\psi(\mathbf{Z}), 0)$, where $\mathcal{B}(\ell, y) = \log(1 + e^\ell) - y\ell$ is binary cross entropy on logits. Moreover,

$$\langle \mathbf{g}, \mathbf{U} \rangle_F = r \sum_v \|\mathbf{V}_{\theta,v}\|_2 \|\mathbf{g}_v\|_2 \geq 0, \qquad \|\mathbf{U}\|_F \leq r \|\mathbf{V}_\theta\|_F. \tag{5}$$

The result follows from Cauchy-Schwarz and $\nabla_{\mathbf{Z}} \mathcal{B}(\ell_\psi, 0) = \sigma(\ell_\psi)\mathbf{g}$. Thus guidance maximizes the first-order increase in negative loss for the fixed current detector. Appendix B.1 gives the proof, conditions for improvement at a finite step, and the numerical normalization variant.

Initialize $\mathbf{Z}^{(0)}$ with n independent source draws. For $K \geq 1$ updates to time $\tau \in (0, 1]$, let $h = \tau/K$ and $t_k = kh$. Sampling uses

$$\begin{aligned} \mathbf{U}_v^{(k)} &= r \|\mathbf{V}_v^{(k)}\|_2 \nu\Big(\mathrm{sg}[\mathbf{g}_v^{(k)}]\Big), \\ \mathbf{Z}^{(k+1)} &= \mathbf{Z}^{(k)} + h\big(\mathbf{V}^{(k)} + \mathbf{U}^{(k)}\big). \end{aligned} \tag{6}$$

Here $\mathbf{V}^{(k)} = \mathbf{V}_\theta(\mathbf{Z}^{(k)}, t_k, \mathbf{C})$, $\mathbf{g}^{(k)} = \mathbf{g}(\mathbf{Z}^{(k)})$, and sg preserves values but stops derivatives. Parameter gradients pass through the states, velocities, velocity norms, and decoder, while each guidance direction is held constant. This avoids second derivatives through D_ψ. We fix r within a batch and decode $\widehat{\mathbf{Z}} = \mathbf{Z}^{(K)}$ to obtain $\widetilde{G}$. The horizon specifies how much transport is executed. A shorter horizon retains an intermediate transport state, complementing guidance in controlling normal resemblance. Conditional velocity interpolation follows the configurations in Appendix A.4.

4.3 Adaptive Guidance via Ranking Feedback

The relative budget controls an individual update. Its appropriate strength can change as the detector learns. We measure current separability using normal logits $\ell_i^+ = D_\psi(G_i, \mathbf{Z}_{G_i})$ and synthetic logits $\ell_j^- = D_\psi(\widetilde{G}_j, \widehat{\mathbf{Z}}_j)$:

$$d_b = \frac{1}{N_+ N_-} \sum_{i,j} \left(\mathbb{I}[\ell_i^+ > \ell_j^-] + \tfrac{1}{2}\mathbb{I}[\ell_i^+ = \ell_j^-] \right), \qquad N_+, N_- > 0. \tag{7}$$

This empirical AUC measures ordering independently of absolute logit scale. Values near one indicate easy ranking separation. Values near one half indicate chance pairwise ranking. We smooth it as $\bar{d}_{b+1} = 0.9\bar{d}_b + 0.1 d_b$ and set

$$r_{b+1} = \mathrm{clip}_{[r_{\min}, r_{\max}]} \left(r_b \exp\big\{ \kappa \left[(\bar{d}_{b+1} - d_{\mathrm{hi}})_+ - \tfrac{1}{4}(d_{\mathrm{lo}} - \bar{d}_{b+1})_+ \right] \big\} \right). \tag{8}$$

Here $(x)_+ = \max(x, 0)$. The rule strengthens guidance when synthetic graphs become easy to distinguish and weakens it when separation is too weak. The smaller decrease coefficient moderates this reduction. Within the target range, r is unchanged. We use $(d_{\mathrm{lo}}, d_{\mathrm{hi}}) = (0.60, 0.90)$, $\kappa = 0.05$, and $[r_{\min}, r_{\max}] = [0.1, 2.0]$. Feedback uses logits before the parameter update and affects the next batch. Ranking supplies control that is independent of logit scale, while BCE parameter updates depend on prediction confidence. Appendix A.5 specifies initialization and fixed or disabled guidance.

4.4 Joint Optimization and Anomaly Scoring

Guidance refines latent states with the networks held fixed. Joint training then updates the networks using the resulting examples. We optimize the following loss using the normal and synthetic logits above and the stop-gradient rule in Eq. (6):

$$\mathcal{L}_{\mathrm{joint}}(\theta, \psi) = \mathbb{E}\mathcal{B}(\ell^+, 1) + \mathbb{E}\mathcal{B}(\ell^-, 0) + \lambda \mathcal{L}_{\mathrm{FM}}(\theta), \qquad \lambda > 0. \tag{9}$$

A single optimizer updates θ and ψ, with ϕ frozen and generated samples attached to the computation graph. Synthetic classification trains both networks through the differentiation rule in Eq. (6). Its derivative $\partial \mathcal{B}(\ell, 0)/\partial \ell = \sigma(\ell) > 0$ encourages lower synthetic logits. Guidance locally increases those logits during sampling, while fresh FM interpolation samples continue to penalize changes to normal reference directions. The method therefore combines discriminative parameter updates with challenging state refinement under a persistent regression constraint. Algorithm 1 specifies their order.

At inference, the frozen encoder supplies $\mathbf{Z}_G$ under the configured evaluation convention, and $a(G) = -D_\psi(G, \mathbf{Z}_G)$. Generation and decoding are used only during training.

4.5 Deviation from Normal References

The local budget controls refinement, but resemblance also depends on the remaining transport and the learned field. We make these contributions explicit. For a normal reference graph G_1, let $\mathbf{Z}_1$ be its encoded target with the same node correspondence as the generated state. Define

$$e_k = \left\| \mathbf{V}^{(k)} - \frac{\mathbf{Z}_1 - \mathbf{Z}^{(k)}}{1 - t_k} \right\|_F, \qquad \omega_{k,K} = \prod_{i=k+1}^{K-1} \frac{1 - t_{i+1}}{1 - t_i}. \tag{10}$$

The empty product equals one, and all denominators are positive.

Proposition 2 (Reference deviation)**.** *For the deterministic sampler in Eq. (6),*

$$\|\widehat{\mathbf{Z}} - \mathbf{Z}_1\|_F \le (1 - \tau)\|\mathbf{Z}^{(0)} - \mathbf{Z}_1\|_F + h \sum_{k=0}^{K-1} \omega_{k,K} \big(e_k + r\|\mathbf{V}^{(k)}\|_F \big). \tag{11}$$

The terms separate truncation, deviation of the learned velocity from a normal bridge, and accumulated guidance. With exact conditioning, FM directly penalizes the bridge residual on normal interpolation states. The bound evaluates that residual on the sampled trajectory. For a locally Lipschitz decoder, adding its reconstruction error transfers Eq. (11) to an aligned graph distance. Appendix B.2 proves these statements. Thus normal resemblance depends jointly on transport accuracy and the sampling controls, while Proposition 1 describes the local effect on discrimination.

5 Experiments

5.1 Experimental Settings

Datasets. Following AGDiff (Cai et al., 2025b), we evaluate BNFlow on eight graph benchmarks in two groups. The moderate-scale group comprises COX2, ER_MD, MUTAG, and DD, covering molecular graphs and protein structures with different graph sizes and connectivity patterns. The large-scale, imbalanced group comprises MOLT-4, MCF-7, PC-3, and SW-620, where graphs represent chemical compounds labeled by anticancer activity. Active compounds account for a small minority, providing a setting with rare anomalous graphs. We treat the minority class as anomalous and train BNFlow using normal graphs. Dataset statistics and protocol details are provided in Appendix C.1.

Baselines. We compare four classical graph kernels (SP, WL, NH, and RW), eight neural GLAD methods (OCGIN (Zhao & Akoglu, 2023), OCGTL (Qiu et al., 2022), GLocalKD (Ma et al., 2022), iGAD (Zhang et al., 2022), SIGNET (Liu et al., 2023c), MUSE (Kim et al., 2024), AGDiff (Cai et al., 2025b)), and ProtoGLAD (Zhao et al., 2026).

Evaluation Metrics. We report the area under the receiver operating characteristic curve (AUROC) and F1-score. AUROC measures ranking quality across decision thresholds, whereas F1 summarizes the precision–recall balance at the evaluated threshold.

Implementation Details. Please refer to Appendix C.

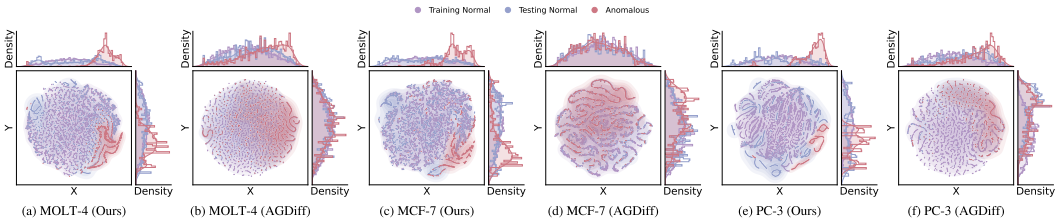
5.2 Main Results

Table 1 compares BNFlow with graph kernels and neural GLAD methods across benchmarks of different sizes and anomaly prevalences. Compared with graph kernel methods, BNFlow improves both AUROC and F1 on every dataset with available kernel results. Although kernels capture recurring graph substructures, their detection performance varies substantially across datasets. They remain competitive on MUTAG and DD but fall well behind BNFlow on the large, imbalanced benchmarks. This contrast supports learning discriminative representations beyond the structural similarities supplied by the evaluated kernels. Compared with neural GLAD methods, BNFlow also delivers consistent gains across the two benchmark groups. The non-generative neural baselines improve on kernels in several settings, yet their performance remains sensitive to the dataset, particularly under severe class imbalance. AGDiff is the strongest overall neural comparator, supporting the value of generated negatives for graph-level detection. BNFlow further improves on

Table 1: Detection performance (%). Best means are **bold** on purple; second-best means use lighter purple. Blue ± values denote standard deviations; dashes denote unavailable results.

Datasets	COX2		ER_MD		MUTAG		DD	
Methods	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1
SP	57.74±4.16	53.97±2.82	25.34±6.79	33.60±5.02	90.27±3.69	84.00±4.00	81.92±1.94	75.00±2.18
WL	55.51±3.85	54.52±2.66	25.26±6.65	33.60±5.02	89.04±3.25	80.96±4.31	80.42±1.83	74.13±2.49
NH	61.76±4.01	55.21±3.83	47.67±5.89	46.78±6.28	72.56±6.98	66.67±1.89	79.26±1.77	72.32±1.91
RW	54.82±2.81	52.76±1.85	23.56±1.45	26.36±4.32	—	—	41.11±3.15	42.03±2.05
OCGIN	70.38±5.44	63.47±2.33	68.24±7.51	61.64±8.48	86.54±8.11	81.00±8.75	72.28±2.34	66.18±2.67
OCGTL	75.61±3.30	70.32±1.71	40.39±13.78	39.16±11.04	85.97±1.34	76.00±0.00	70.63±8.29	69.28±6.18
GLocalKD	52.05±5.65	50.68±6.80	73.33±7.77	65.41±3.88	89.04±3.91	78.00±4.82	80.45±3.15	73.86±2.67
iGAD	56.98±2.94	56.16±2.96	39.42±23.29	41.64±15.87	80.14±26.42	72.80±20.38	73.10±3.61	72.75±2.58
SIGNET	52.49±3.75	53.88±2.81	61.05±5.41	57.86±3.88	74.62±6.67	70.60±5.55	54.55±2.81	51.69±2.08
MUSE	61.39±7.57	57.99±4.66	35.72±4.28	39.11±3.16	83.79±6.91	76.00±8.64	66.08±3.72	60.63±3.26
AGDiff	76.64±3.47	65.60±2.29	91.34±2.07	82.76±3.23	94.78±2.04	89.00±1.03	86.64±1.45	**81.08**±2.03
ProtoGLAD	67.46±2.09	62.25±3.46	43.00±5.23	44.27±4.11	84.45±2.12	68.00±3.27	78.55±1.64	71.40±1.62
BNFlow (Ours)	**81.37**±2.86	**73.06**±2.33	**92.56**±1.07	**83.97**±0.95	**95.68**±0.13	**90.67**±1.89	**87.00**±0.49	80.43±0.59

Datasets	MOLT-4		MCF-7		PC-3		SW-620	
Methods	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1
SP	36.21±0.12	20.05±0.16	34.51±0.17	19.71±0.17	31.57±0.26	11.65±0.21	33.54±0.21	13.45±0.17
WL	37.14±0.11	19.76±0.12	35.88±0.16	20.23±0.31	32.93±0.30	11.05±0.25	34.77±0.25	12.28±0.22
NH	48.63±0.33	28.34±0.38	48.78±0.44	29.15±0.67	48.79±0.31	20.28±0.23	48.77±0.38	21.38±0.05
RW	—	—	34.47±0.00	19.57±0.00	—	—	—	—
OCGIN	46.66±2.29	26.68±1.94	47.66±1.46	28.15±1.02	47.66±2.39	18.81±1.99	46.90±1.52	19.70±0.93
OCGTL	55.98±0.38	34.95±0.16	55.47±0.56	35.63±0.94	55.74±0.38	27.76±0.11	55.35±0.78	28.09±0.63
GLocalKD	55.71±0.98	35.96±1.02	54.91±0.36	36.14±0.65	55.57±0.62	29.10±0.71	55.94±0.60	30.24±0.47
iGAD	47.70±3.08	27.88±2.16	47.90±5.40	28.84±5.53	48.56±4.64	22.22±2.69	47.79±7.46	41.84±23.47
SIGNET	—	—	47.91±2.88	25.53±2.76	57.01±0.78	23.71±1.72	—	—
MUSE	—	—	—	—	—	—	—	—
AGDiff	88.14±1.37	87.56±1.36	89.68±2.18	87.51±0.74	92.74±2.14	91.72±2.17	92.50±1.23	91.69±1.66
ProtoGLAD	51.49±2.07	28.30±0.61	47.59±0.64	27.18±0.00	50.76±1.41	21.85±3.73	45.09±1.91	17.59±1.35
BNFlow (Ours)	**91.69**±4.58	**90.53**±4.13	**92.73**±0.68	**90.48**±1.53	**93.12**±3.59	**93.73**±3.27	**95.74**±1.75	**96.07**±1.87



(a) MOLT-4 (Ours) (b) MOLT-4 (AGDiff) (c) MCF-7 (Ours) (d) MCF-7 (AGDiff) (e) PC-3 (Ours) (f) PC-3 (AGDiff)

Figure 2: t-SNE of discriminator representations from selected runs on MOLT-4, MCF-7, and PC-3. Purple, blue, and red denote training normals, test normals, and test anomalies, respectively; "Ours" denotes BNFlow.

AGDiff by an average of 2.18 AUROC and 2.75 F1 percentage points, with gains in both metrics across all large datasets. These results are consistent with combining a normal transport reference and detector-dependent sample refinement to provide useful negative supervision. Overall, BNFlow achieves the highest mean AUROC on all eight datasets and the highest mean F1 on seven; the following analysis examines the representations and component contributions behind these results.

5.3 Representation Analysis

To examine class separation in the learned representations, we visualize selected discriminator graph embeddings using t-SNE (Figure 2). The horizontal X and vertical Y axes are the two coordinates of the projection, rather than individual graph attributes or anomaly scores. Purple, blue, and red distinguish training normals, test normals, and anomalies, respectively. Across the displayed datasets, BNFlow concentrates anomalies in more localized regions with less overlap with normal observations, whereas AGDiff shows greater mixing between the groups. The marginal density plots also reveal more distinct anomalous peaks for BNFlow. Meanwhile, training and test normals remain

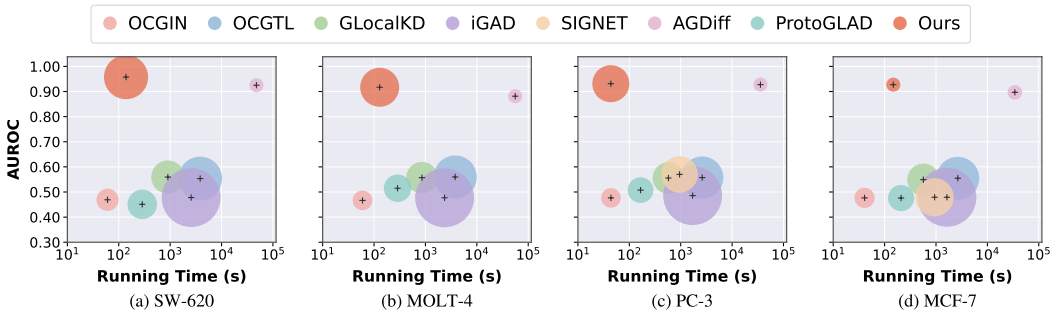


Figure 3: Training time versus AUROC on four large datasets. Time is logarithmic, and bubble size encodes peak reserved GPU memory on a logarithmic scale. The plot label "Ours" denotes BNFlow.

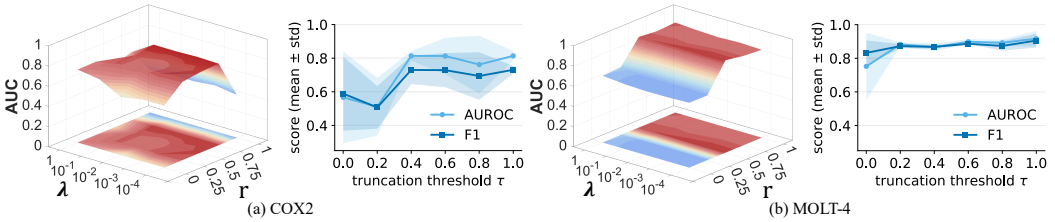


Figure 4: Parameter sensitivity on (a) COX2 and (b) MOLT-4. For each dataset, the surface shows mean AUROC over fixed guidance strengths r and FM weights λ, with adaptation disabled; the curves show AUROC and F1 versus horizon τ, with standard-deviation bands. The $\tau = 0$ point is the separate local-noise, no-FM control.

broadly interspersed. Together, these patterns suggest clearer separation of anomalous graphs while retaining consistent representations of normal graphs across training and testing.

5.4 Memory and Time Efficiency

We compare training time, peak reserved GPU memory, and AUROC for neural GLAD methods on the four large datasets (Figure 3). Resource measurements use the same machine, with BNFlow profiled in serial runs under exclusive GPU access. The horizontal axis reports training-subprocess wall time, and AUROC values come from the main performance comparison. Relative to OCGIN, BNFlow achieves substantially higher AUROC at a similar runtime scale, showing that short training time can be combined with effective anomaly discrimination. Compared with OCGTL, GLocalKD, iGAD, and SIGNET where results are available, it combines higher detection accuracy with shorter training in the reported measurements. BNFlow thus occupies the high-AUROC, short-runtime region across these benchmarks. Among generative methods, AGDiff reaches competitive accuracy but requires substantially longer training, with measured speedups of 230–800$\times$ for BNFlow. Bubble sizes encode peak reserved GPU memory on a logarithmic scale, adding the memory footprint to this time–accuracy comparison. The consistent runtime advantage across the four datasets supports the practical efficiency of BNFlow under the evaluated training configurations.

5.5 Parameter Sensitivity

The parameter study examines guidance strength, regression weight, and sampling horizon using the surfaces and curves in Figure 4. Guidance strength r sets the refinement-velocity budget relative to the base flow velocity; λ weights continued flow-matching regression during joint training to retain the normal reference. With adaptation disabled, COX2 favors moderate guidance and deteriorates under excessive refinement; MOLT-4 requires stronger guidance before plateauing. Both surfaces vary less along λ in the tested grid. Terminal time τ controls how far transport proceeds before decoding. COX2 benefits from sufficient transport, whereas MOLT-4 tolerates broader horizon choices. The $\tau = 0$ point replaces flow with local noise, a separate generator control. These patterns favor dataset-dependent guidance and horizons.

5.6 Ablation Study

Component contributions are assessed by comparing the complete model (**Full**) with five variants in Figure 5. Without flow-matching pretraining (**w/o Pretrain**) omits pretrained initialization. Without discriminator guidance (**w/o DG**) disables refinement. Without adaptive guidance control (**w/o adaptive**) fixes r. Without sampling-horizon truncation (**w/o boundary**) completes transport. Without flow matching (**w/o FM**) uses local noise. Removing guidance lowers AUROC by 20.00 and 28.19 points on COX2 and MOLT-4, respectively. The gains from pretraining indicate that a learned normal reference provides a useful starting point for joint optimization. Replacing transport with local perturbations also reduces detection quality, supporting learned generation over this simpler alternative. Adaptive guidance benefits COX2, whereas horizon truncation benefits MOLT-4, illustrating dataset-dependent roles for these controls. COX2 already uses full transport, so its truncation-removal control reproduces Full's configuration rather than testing an active component.

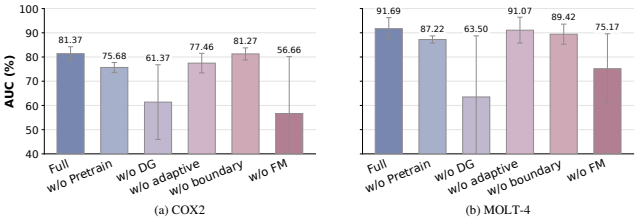


Figure 5: Component ablations: mean AUROC and standard deviation.

5.7 Changes in Pseudo-Anomaly Distribution

Three sector groups organize Figure 6: *Pseudo Anomalies* contains generated samples grouped by sampling horizon, *Real Anomalies* names distance regions calibrated using real test anomalies, and *Normal Prototypes* represents clusters of normal training embeddings. N1–N3 are three cluster centers fitted to these normal embeddings. Samples close to the normal training set are assigned to their nearest prototype. Near contains samples outside this normal-like region but within the distance threshold calibrated from real anomalies, whereas Far contains samples beyond that threshold. Ribbons run from each horizon to its assigned regions, and thicker ribbons indicate larger fractions of generated samples. Under discriminator-guided transport, increasing the horizon shifts the distribution from Far toward the normal prototypes, mainly N3. Approximately 98% is normal-like from horizon 0.3 onward. This reveals both proximity to normal representations and uneven coverage of their clusters. Longer transport therefore need not distribute pseudo anomalies evenly across normal patterns. This geometric view complements the parameter study by linking the sampling horizon to the latent regions supplying synthetic negative examples. Appendix C.2 specifies the assignment procedure.

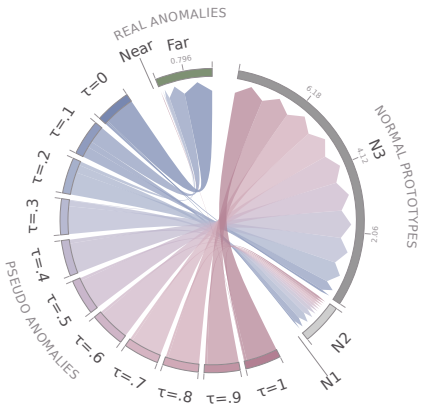


Figure 6: Discriminator-guided latent placement on COX2. Ribbon widths denote generated fractions.

6 Conclusion

In this work, we introduced BNFlow for graph-level anomaly detection, addressing the challenge of generating pseudo anomalies that remain informative as the detector evolves. Rather than relying on fixed perturbations or synthetic negatives that may become increasingly easy to distinguish, BNFlow couples flow-based transport toward normal graph representations with detector-guided sample refinement. The normal transport provides a persistent reference for preserving meaningful graph patterns, while bounded guidance steers intermediate states toward harder negatives and ranking feedback adaptively adjusts the refinement strength during training. Continued flow-matching regression further constrains the generator toward the normal reference under joint optimization. Our theoretical analysis characterizes the local effect of detector guidance and its deviation from normal references, while experiments on eight graph benchmarks demonstrate consistent improvements over existing GLAD methods. These results highlight guided generative modeling as a promising mechanism for constructing adaptive negative supervision in graph anomaly detection.

AI USE STATEMENT

Generative AI tools were used to assist with literature discovery, language revision, and mathematical consistency checks.

ETHICS STATEMENT

The method learns a statistical notion of deviation from a reference population. An anomalous score does not establish harmfulness, misconduct, or a causal explanation. Applications involving people require validation in the intended domain and assessment of the consequences of false positives and false negatives. Generated graphs are training references and should not be presented as observed anomalous events.

REPRODUCIBILITY STATEMENT

We provide the source code needed to reproduce all reported results, and we will release them publicly upon acceptance. All datasets are publicly available benchmarks, and Section 4 together with the appendices specifies the method and experimental protocol in sufficient detail to reimplement the full pipeline independently.

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. doi: 10.1007/s10618-014-0365-y.

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023.

Heli Ben-Hamu, Omri Puny, Itai Gat, Brian Karrer, Uriel Singer, and Yaron Lipman. D-Flow: Differentiating through flows for controlled generation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 3462–3483. PMLR, 2024.

Jinyu Cai, Yunhe Zhang, and Jicong Fan. Self-discriminative modeling for anomalous graph detection. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 6387–6408. PMLR, 2025a.

Jinyu Cai, Yunhe Zhang, Fusheng Liu, and See-Kiong Ng. Leveraging diffusion model as pseudo-anomalous graph generator for graph-level anomaly detection. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 6409–6425. PMLR, 2025b.

Jinyu Cai, Yunhe Zhang, Pengyang Wang, and See-Kiong Ng. MoEGAD: A mixture-of-experts framework with pseudo-anomaly generation for graph-level anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(4):4473–4489, 2026. doi: 10.1109/TPAMI.2025.3646069.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Xuefeng Du, Zhaoning Wang, Mu Cai, and Yixuan Li. VOS: Learning what you don't know by virtual outlier synthesis. In *International Conference on Learning Representations*, 2022.

Ruiqi Feng, Chenglei Yu, Wenhao Deng, Peiyan Hu, and Tailin Wu. On the guidance of flow matching. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 16993–17029. PMLR, 2025.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems*, volume 37, pp. 133345–133385, 2024.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Sachin Goyal, Aditi Raghunathan, Moksh Jain, Harsha Vardhan Simhadri, and Prateek Jain. DROCC: Deep robust one-class classification. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 3711–3721. PMLR, 2020.

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations*, 2019.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 10362–10383. PMLR, 2022.

Sunwoo Kim, Soo Yong Lee, Fanchen Bu, Shinhwan Kang, Kyungho Kim, Jaemin Yoo, and Kijung Shin. Rethinking reconstruction-based graph-level anomaly detection: Limitations and a simple remedy. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. URL https://arxiv.org/abs/1412.6980.

Thomas N. Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *International Conference on Learning Representations*, 2018.

Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. CutPaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9664–9674, 2021.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

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*, 2023a.

Yixin Liu, Kaize Ding, Huan Liu, and Shirui Pan. GOOD-D: On unsupervised graph out-of-distribution detection. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pp. 339–347, 2023b. doi: 10.1145/3539597.3570446.

Yixin Liu, Kaize Ding, Qinghua Lu, Fuyi Li, Leo Yu Zhang, and Shirui Pan. Towards self-interpretable graph-level anomaly detection. In *Advances in Neural Information Processing Systems*, volume 36, 2023c.

Xuexiong Luo, Jia Wu, Jian Yang, Shan Xue, Hao Peng, Chuan Zhou, Hongyang Chen, Zhao Li, and Quan Z. Sheng. Deep graph level anomaly detection with contrastive learning. *Scientific Reports*, 12:19867, 2022. doi: 10.1038/s41598-022-22086-3.

Rongrong Ma, Guansong Pang, Ling Chen, and Anton van den Hengel. Deep graph-level anomaly detection by glocal knowledge distillation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pp. 704–714. ACM, 2022. doi: 10.1145/3488560.3498473.

Chaoxi Niu, Guansong Pang, and Ling Chen. Graph-level anomaly detection via hierarchical memory networks. In *Machine Learning and Knowledge Discovery in Databases: Research Track*, pp. 201–218. Springer, 2023.

Yiming Qin, Manuel Madeira, Dorina Thanou, and Pascal Frossard. DeFoG: Discrete flow matching for graph generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 50269–50326. PMLR, 2025.

Chen Qiu, Marius Kloft, Stephan Mandt, and Maja Rudolph. Raising the bar in graph-level anomaly detection. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pp. 2196–2203, 2022. doi: 10.24963/ijcai.2022/305.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 4393–4402. PMLR, 2018.

Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(77): 2539–2561, 2011.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Leitian Tao, Xuefeng Du, Xiaojin Zhu, and Yixuan Li. Non-parametric outlier synthesis. In *The Eleventh International Conference on Learning Representations*, 2023.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024.

Clément Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023.

Chunjing Xiao, Shikang Pang, Wenxin Tai, Yanlong Huang, Goce Trajcevski, and Fan Zhou. Motif-consistent counterfactuals with adversarial refinement for graph-level anomaly detection. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 2024. doi: 10.1145/3637528.3672050.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

Zhenyu Yang, Ge Zhang, Shan Xue, Xiaoxiao Ma, Jian Yang, Hao Peng, Amin Beheshti, and Jia Wu. Revisiting graph-level anomaly detection: From partially to fully unsupervised learning. In *Proceedings of the ACM Web Conference 2026*, pp. 901–912. ACM, 2026. doi: 10.1145/3774904.3792307.

Ge Zhang, Zhenyu Yang, Jia Wu, Jian Yang, Shan Xue, Hao Peng, Jianlin Su, Chuan Zhou, Quan Z. Sheng, Leman Akoglu, and Charu Aggarwal. Dual-discriminative graph neural network for imbalanced graph-level anomaly detection. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Ge Zhang, Huimei Li, Guohao Sun, Xiu Fang, Xixun Lin, Xiaobao Wang, Pengfei Jiao, and Liang Yang. Unsupervised graph-level anomaly detection via multi-granular graph structure learning. In *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence*, pp. 3196–3204, 2026. doi: 10.24963/ijcai.2026/355.

Lingxiao Zhao and Leman Akoglu. On using classification datasets to evaluate graph outlier detection: Peculiar observations and new insights. *Big Data*, 11(3):151–180, 2023. doi: 10.1089/big.2021.0069.

Qiuran Zhao, Kai Ming Ting, and Xinpeng Li. Interpretable graph-level anomaly detection via contrast with normal prototypes. *arXiv preprint arXiv:2602.10708*, 2026.

A Training and Sampling Details

A.1 Graph Representation

The representation model uses a GIN encoder with three layers, a node feature MLP decoder, and the inner product adjacency decoder in Eq. (1). For posterior means $\mathbf{m}_v$ and standard deviations $\mathbf{s}_v$, the encoder defines

$$q_\phi(\mathbf{Z} \mid G) = \prod_{v=1}^{n} \mathcal{N}\big(\mathbf{z}_v \mid \mathbf{m}_v, \mathrm{diag}(\mathbf{s}_v^2)\big)\,, \qquad \mathbf{z}_v = \mathbf{m}_v + \mathbf{s}_v \odot \boldsymbol{\epsilon}_v, \quad \boldsymbol{\epsilon}_v \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{12}$$

The pretraining objective is

$$\mathcal{L}_{\mathrm{VGAE}} = \mathbb{E}_G\Big[\mathbb{E}_{q_\phi(\mathbf{Z}|G)}\Big[\mathrm{BCE}_{\mathbf{M}}(\widehat{\mathbf{A}}, \mathbf{A}) + \mathrm{MSE}(\widehat{\mathbf{X}}, \mathbf{X})\Big] + \gamma D_{\mathrm{KL}}(q_\phi(\mathbf{Z} \mid G) \| p_{\mathrm{prior}})\Big]\,, \tag{13}$$

where $\mathrm{BCE}_{\mathbf{M}}$ evaluates adjacency reconstruction only on the selected positive and negative pairs, p_{prior} is a product of standard Gaussian node priors and $\gamma = 0.001$. Both the encoder and decoder remain frozen thereafter. The selected configuration determines whether cached targets use posterior means or samples. The evaluation convention is fixed separately for the same configuration. Evaluation with means is deterministic, while evaluation with samples is conditional on the latent draw.

Targets and conditions are obtained from training graphs. The training sampler determines their weighting across graphs and nodes. Every generated latent matrix has the size of its reference graph and is decoded within its own graph block. This preserves the association between a reference graph's nodes and its condition matrix throughout sampling.

A.2 Conditioning and the Normal Transport Reference

Let Q_ϕ^{pair} include the chosen normal target rule and the configured conditional input rule. The source $\mathbf{z}_0$ and time t are independent of $(\mathbf{z}_1, \mathbf{c})$. For a noisy condition, its corruption draw is also independent of $\mathbf{z}_0$ and t. Squared error regression gives

$$\mathbf{v}^*(\mathbf{z}, t, \mathbf{c}) = \mathbb{E}[\mathbf{z}_1 - \mathbf{z}_0 \mid \mathbf{z}_t = \mathbf{z}, t, \mathbf{c}]. \tag{14}$$

Indeed, for $\mathbf{u} = \mathbf{z}_1 - \mathbf{z}_0$ and $\bar{\mathbf{u}} = \mathbb{E}[\mathbf{u} \mid \mathbf{z}_t, t, \mathbf{c}]$, conditional orthogonality yields

$$\mathbb{E}\|\mathbf{v}_\theta - \mathbf{u}\|_2^2 = \mathbb{E}\|\mathbf{v}_\theta - \bar{\mathbf{u}}\|_2^2 + \mathbb{E}\|\mathbf{u} - \bar{\mathbf{u}}\|_2^2. \tag{15}$$

When $\mathbf{c} = \mathbf{z}_1$ exactly, the interpolation identity implies

$$\mathbf{z}_0 = \frac{\mathbf{z}_t - t\mathbf{c}}{1 - t}, \qquad \mathbf{v}^*(\mathbf{z}, t, \mathbf{c}) = \frac{\mathbf{c} - \mathbf{z}}{1 - t}, \qquad 0 \le t < 1. \tag{16}$$

Thus the reference field is an analytic bridge to the supplied target. On an interpolation state, its velocity is exactly $\mathbf{z}_1 - \mathbf{z}_0$. With a fixed condition matrix $\mathbf{C} = \mathbf{Z}_1$, its solution is $\mathbf{Z}_{\mathrm{ref}}(t) = (1-t)\mathbf{Z}^{(0)} + t\mathbf{Z}_1$, whose limiting endpoint is $\mathbf{Z}_1$. All Euler evaluations in the paper occur at $t_k < 1$, including when the final time is one.

This identity clarifies the role of learning in the conditioned model. Pure FM regression fits the reference bridge. Joint classification subsequently changes the parameterized velocity, and continued FM measures departure from that bridge on normal interpolation states. The learnable velocity therefore accommodates discriminative updates under a normal regression penalty. With a noisy condition, Eq. (14) averages over compatible targets. With the null condition, it is the marginal velocity of the normal node interpolation distribution.

Graph dependencies. The reference condition matrix is obtained from one encoded graph. Its rows carry relationships already captured by the GIN encoder. Even though velocity evaluations are shared across rows, jointly sampled condition matrices retain these dependencies. The inner product decoder couples pairs of node latents, and the graph discriminator couples the gradients used for refinement. In the unconditional configuration, the regression objective supplies a node marginal reference, while relational computation is provided by the encoder, decoder, and discriminator.

Permutation symmetry. For a permutation matrix $\mathbf{P}$, relabel the candidate mask as $\mathbf{M} \mapsto \mathbf{P}\mathbf{M}\mathbf{P}^\top$ along with the nodes. With this relabeling understood, the decoder satisfies $\widehat{\mathbf{X}}(\mathbf{P}\mathbf{Z}) = \mathbf{P}\widehat{\mathbf{X}}(\mathbf{Z})$ and $\widehat{\mathbf{A}}(\mathbf{P}\mathbf{Z}) = \mathbf{P}\widehat{\mathbf{A}}(\mathbf{Z})\mathbf{P}^\top$. Shared row velocities satisfy $\mathbf{V}_\theta(\mathbf{P}\mathbf{Z}, t, \mathbf{P}\mathbf{C}) = \mathbf{P}\mathbf{V}_\theta(\mathbf{Z}, t, \mathbf{C})$ for a condition matrix $\mathbf{C}$. With null conditioning, the null input stays unchanged. A permutation invariant detector gives $\ell_\psi(\mathbf{P}\mathbf{Z}) = \ell_\psi(\mathbf{Z})$ and, at differentiable states, $\mathbf{g}(\mathbf{P}\mathbf{Z}) = \mathbf{P}\mathbf{g}(\mathbf{Z})$. Row normalization and the Euler update preserve this equivariance. The distribution of generated graphs therefore respects relabeling of the reference nodes when the source latents are relabeled consistently and the candidate-pair sampling law is invariant to node relabeling.

A.3 Gradient Paths through the Graph Decoder

The discriminator combines graph representations from three GIN layers with latent features, graph pooling, and residual feedforward transformations. Normal inputs are $(G, \mathbf{Z}_G)$ and generated inputs are $(\mathrm{dec}_\phi(\mathbf{Z}), \mathbf{Z})$. Its normality logit is differentiated with respect to the generated latent matrix for guidance. The following chain rule describes this operation independently of the particular widths and pooling implementation.

At a differentiable evaluation, let $\mathbf{G}_X$, $\mathbf{G}_A$, and $\mathbf{G}_Z$ be partial derivatives of $D_\psi(\widehat{\mathbf{X}}, \widehat{\mathbf{A}}, \mathbf{Z})$ with respect to its three inputs, treating them as independent arguments. Set $\mathbf{S} = \sigma(\mathbf{Z}\mathbf{Z}^\top)$ and $\mathbf{H} = \mathbf{M} \odot \mathbf{G}_A \odot \mathbf{S} \odot (\mathbf{1} - \mathbf{S})$. The mask is held fixed for this derivative and for local finite-step comparisons, and only its selected entries of $\mathbf{S}$ are evaluated. Then

$$\mathbf{g}(\mathbf{Z}) = J_{\mathrm{MLP}_\phi}(\mathbf{Z})^*[\mathbf{G}_X] + (\mathbf{H} + \mathbf{H}^\top)\mathbf{Z} + \mathbf{G}_Z. \tag{17}$$

Here J^* denotes the adjoint Jacobian acting on the output gradient. The three terms are the feature decoding path, adjacency decoding path, and direct latent path. To obtain the middle term, use

$$\mathrm{d}\mathbf{S} = \mathbf{S} \odot (\mathbf{1} - \mathbf{S}) \odot \left(\mathrm{d}\mathbf{Z}\,\mathbf{Z}^\top + \mathbf{Z}\,\mathrm{d}\mathbf{Z}^\top\right) \tag{18}$$

and collect coefficients of $\mathrm{d}\mathbf{Z}$ in the Frobenius inner product. The derivatives include the discriminator's graph computation. Each matrix and its derivative are restricted to the corresponding graph block.

During sampling, the computed $\mathbf{g}^{(k)}$ is detached. The outer update differentiates through $\mathbf{Z}^{(k)}$, $\mathbf{V}^{(k)}$, its norm, and decoder inputs, with the detached direction held constant. Hence the flow receives classification gradients through the unrolled trajectory, while the detector receives direct classification gradients. These surrogate derivatives specify the training algorithm. Detector running statistics are restored after auxiliary guidance calls, following the method configuration.

A.4 Sampling Variants and Configurations

Flow pretraining fits the cached normal latents for 100 epochs. FM regression remains active during joint training. Table 2 records the supplied configurations. The terminal time τ specifies how much transport is executed, and K denotes the number of executed Euler updates, with $h = \tau/K$.

Where conditional interpolation is configured, the base velocity in Eq. (6) is

$$\mathbf{V}_\theta^{(w)}(\mathbf{Z}, t, \mathbf{C}) = (1 - w)\mathbf{V}_\theta(\mathbf{Z}, t, \emptyset) + w\mathbf{V}_\theta(\mathbf{Z}, t, \mathbf{C}). \tag{19}$$

The norm of this resulting velocity sets the guidance budget. The choice $w = 1$ uses the conditioned evaluation, while $w = 0.5$ averages the two evaluations. Their input sampling conventions are part of Q_ϕ^{pair}.

Sampling follows the deterministic Euler update in Eq. (6), with the condition matrix fixed throughout the trajectory.

A.5 Feedback and Update Order

Initialize $\bar{d}_0 = 0.75$. Adaptive guidance uses $r \in [0.1, 2.0]$, with the initial values in Table 2. For $\rho = 1/4$, the expanded update is

$$r_{b+1} = \begin{cases} \min\{r_{\max}, r_b \exp[\kappa(\bar{d}_{b+1} - d_{\mathrm{hi}})]\}, & \bar{d}_{b+1} > d_{\mathrm{hi}}, \\ \max\{r_{\min}, r_b \exp[-\rho\kappa(d_{\mathrm{lo}} - \bar{d}_{b+1})]\}, & \bar{d}_{b+1} < d_{\mathrm{lo}}, \\ r_b, & d_{\mathrm{lo}} \le \bar{d}_{b+1} \le d_{\mathrm{hi}}. \end{cases} \tag{20}$$

Table 2: Sampling and training configurations. A dot denotes no conditional interpolation.

	COX2	ER_MD	MUTAG	DD	MOLT-4	MCF-7	PC-3	SW-620
τ	1	1	1	1	0.1	0.05	0.05	0.05
r_0	0.25	0.25	0.25	0.25	0.5	0.5	0.5	0.5
λ	10^{-4}	10^{-3}	10^{-4}	0.1	10^{-3}	0.1	10^{-3}	10^{-3}
w	·	·	·	·	1	0.5	1	0.5

This is equivalent to Eq. (8) when r_b lies in the clipping interval. Static guidance bypasses the update, and disabled guidance sets $r = 0$. The class counts are positive, and tied logits receive weight one half.

Scale and bounded changes. Strictly increasing transformations of logits preserve every comparison in Eq. (7). Positive affine rescaling also preserves the ideal normalized guidance at fixed model states. BCE parameter gradients retain their dependence on confidence. Since $d_b \in [0, 1]$ and the moving average is a convex combination, $\bar{d}_b \in [0, 1]$ for every batch. For adaptive guidance, clipping can only shorten a log strength update, giving

$$-\rho\kappa d_{\mathrm{lo}} \le \log \frac{r_{b+1}}{r_b} \le \kappa(1 - d_{\mathrm{hi}}). \tag{21}$$

With the stated parameters, the interval is $[-0.0075, 0.005]$. The factor ρ gives a smaller slope for decreases at equal excursions outside the target interval. These properties specify the scale and rate of feedback. The feedback statistic measures ordering between normal and synthetic scores, while the local guidance analysis describes refinement at the current sampled state.

Algorithm 1 BNFlow training and inference

Require: Normal training graphs, training budget, sampler configuration, FM weight λ, and guidance configuration.
Ensure: Frozen encoder and trained detector.
1: Pretrain the VGAE, freeze ϕ, and cache the configured normal latents.
2: Pretrain the velocity as configured and initialize r and $\bar{d}$.
3: **for** each joint training minibatch **do**
4: Obtain normal inputs and reference graphs with their node counts and conditions.
5: Draw source latents independently for each generated graph.
6: **for** $k = 0, \ldots, K - 1$ **do**
7: Evaluate the configured base velocity.
8: If guidance is enabled, compute and detach the normality gradient.
9: Apply Eq. (6).
10: **end for**
11: Decode each generated graph and compute normal and synthetic logits.
12: Save their ranking statistic and compute FM on fresh normal interpolations.
13: Update θ and ψ jointly with Eq. (9) and the stated stop rule.
14: If adaptive, update $\bar{d}$ and r from the saved logits for subsequent sampling.
15: **end for**
16: **return** $a(G) = -D_\psi(G, \mathbf{Z}_G)$ under the evaluation latent convention.

B Analysis of Guidance and Reference Deviation

The analysis conditions on the sampled candidate supports. Local finite-step comparisons use the same mask, and trajectory bounds apply conditional on the realized sequence of masks.

The analysis fixes the model parameters for a sampling call. Let $\langle \mathbf{B}, \mathbf{C} \rangle_F = \mathrm{tr}(\mathbf{B}^\top \mathbf{C})$ and $\|\mathbf{B}\|_F^2 = \langle \mathbf{B}, \mathbf{B} \rangle_F$. Each local result refers to a differentiable evaluation of the current detector potential. Parameter differentiation through sampling follows the separate stop rule in Eq. (6).

B.1 Local Optimality of the Guidance Rule

Proof of Proposition 1. For every feasible $\mathbf{B}$, Cauchy-Schwarz gives

$$\langle \mathbf{g}, \mathbf{B} \rangle_F = \sum_v \mathbf{g}_v^\top \mathbf{B}_v \le \sum_v \|\mathbf{g}_v\|_2 \|\mathbf{B}_v\|_2 \le r \sum_v \|\mathbf{g}_v\|_2 \|\mathbf{V}_v\|_2. \tag{22}$$

For $\mathbf{g}_v \neq 0$, the choice $\mathbf{B}_v = r\|\mathbf{V}_v\|_2 \mathbf{g}_v / \|\mathbf{g}_v\|_2$ attains equality. For $\mathbf{g}_v = 0$, selecting $\mathbf{B}_v = 0$ is optimal and feasible. These choices give $\mathbf{U}$ and the alignment identity. Also,

$$\|\mathbf{U}\|_F^2 = r^2 \sum_{v:\, \mathbf{g}_v \neq 0} \|\mathbf{V}_v\|_2^2 \le r^2 \|\mathbf{V}\|_F^2. \tag{23}$$

For finite logits, $\nabla_{\mathbf{Z}} \mathcal{B}(\ell_\psi(\mathbf{Z}), 0) = \sigma(\ell_\psi(\mathbf{Z}))\mathbf{g}$ is a positive scalar multiple of $\mathbf{g}$. The maximizers of the two linear objectives therefore coincide. Differentiability gives

$$\mathcal{B}(\ell_\psi(\mathbf{Z} + \delta\mathbf{B}), 0) = \mathcal{B}(\ell_\psi(\mathbf{Z}), 0) + \delta\sigma(\ell_\psi(\mathbf{Z}))\langle \mathbf{g}, \mathbf{B} \rangle_F + o(\delta) \tag{24}$$

as $\delta \downarrow 0$, proving the first order interpretation. □

Finite step comparison. Suppose $\nabla \ell_\psi$ is L_ℓ-Lipschitz on a convex neighborhood containing $\mathbf{Z}$, $\mathbf{Z} + h\mathbf{V}$, and $\mathbf{Z} + h(\mathbf{V} + \mathbf{U})$. Then

$$\begin{aligned}
&\ell_\psi(\mathbf{Z} + h(\mathbf{V} + \mathbf{U})) - \ell_\psi(\mathbf{Z} + h\mathbf{V}) \\
&\quad = h \int_0^1 \langle \nabla \ell_\psi(\mathbf{Z} + h\mathbf{V} + sh\mathbf{U}), \mathbf{U} \rangle_F \, \mathrm{d}s \\
&\quad \ge h\langle \mathbf{g}, \mathbf{U} \rangle_F - L_\ell h^2 \left(\|\mathbf{V}\|_F \|\mathbf{U}\|_F + \tfrac{1}{2}\|\mathbf{U}\|_F^2 \right) \\
&\quad \ge hr \sum_v \|\mathbf{V}_v\|_2 \|\mathbf{g}_v\|_2 - L_\ell h^2 \left(r + \tfrac{1}{2} r^2 \right) \|\mathbf{V}\|_F^2.
\end{aligned} \tag{25}$$

The first equality integrates along the segment between the two endpoints. Subtracting $\nabla \ell_\psi(\mathbf{Z})$ inside the integral and applying Lipschitz continuity yields the first inequality. Proposition 1 gives the last one. A positive lower bound ensures a higher current logit than the base step from the same state. Since $\mathcal{B}(\ell, 0)$ is increasing in ℓ, the synthetic loss also increases in that comparison.

Numerical normalization. The ideal rule assigns zero guidance to a zero gradient. If normalization is implemented with a clamp $\varepsilon > 0$, let $\beta_v = r\|\mathbf{V}_v\|_2$ and $\mathbf{U}_v^\varepsilon = \beta_v \mathbf{g}_v / \max(\|\mathbf{g}_v\|_2, \varepsilon)$. For $\beta_v > 0$, this variant has the exact characterization

$$\mathbf{U}_v^\varepsilon = \operatorname*{arg\,max}_{\|\mathbf{b}\|_2 \le \beta_v} \left\{ \mathbf{g}_v^\top \mathbf{b} - \frac{\varepsilon}{2\beta_v} \|\mathbf{b}\|_2^2 \right\}. \tag{26}$$

The unconstrained maximizer is $\beta_v \mathbf{g}_v / \varepsilon$. Projecting it onto the radius β_v ball gives the stated vector. For $\beta_v = 0$, zero is the unique feasible vector. The clamp preserves nonnegative local alignment and the relative norm bound. Exact linear optimality holds on rows whose gradient norms reach the threshold. Its behavior under rescaling depends on whether a row crosses that threshold. This characterization concerns the clamped rule specifically.

B.2 Bounds on Deviation from Normal References

Proof of Proposition 2. Fix $\mathbf{Z}_1$ with the same node correspondence as the generated state. Define

$$\mathbf{F}_k(\mathbf{Z}) = \frac{\mathbf{Z}_1 - \mathbf{Z}}{1 - t_k}, \qquad \mathbf{R}_k = \mathbf{V}^{(k)} - \mathbf{F}_k(\mathbf{Z}^{(k)}), \qquad \mathbf{Z}_{\mathrm{ref}}^{(k)} = (1 - t_k)\mathbf{Z}^{(0)} + t_k \mathbf{Z}_1. \tag{27}$$

The reference satisfies $\mathbf{Z}_{\mathrm{ref}}^{(k+1)} = \mathbf{Z}_{\mathrm{ref}}^{(k)} + h\mathbf{F}_k(\mathbf{Z}_{\mathrm{ref}}^{(k)})$ exactly. Write $\mathbf{\Delta}_k = \mathbf{Z}^{(k)} - \mathbf{Z}_{\mathrm{ref}}^{(k)}$ and $\alpha_k = (1 - t_{k+1})/(1 - t_k)$. Subtracting updates gives

$$\mathbf{\Delta}_{k+1} = \alpha_k \mathbf{\Delta}_k + h(\mathbf{R}_k + \mathbf{U}^{(k)}), \qquad \mathbf{\Delta}_0 = 0. \tag{28}$$

Every $\alpha_k \in [0, 1]$, because $0 \le t_k < t_{k+1} \le 1$. Unrolling yields the exact identity

$$\mathbf{\Delta}_K = h \sum_{k=0}^{K-1} \omega_{k,K} (\mathbf{R}_k + \mathbf{U}^{(k)}). \tag{29}$$

The triangle inequality and the guidance norm bound imply

$$\|\mathbf{\Delta}_K\|_F \le h \sum_{k=0}^{K-1} \omega_{k,K}(e_k + r\|\mathbf{V}^{(k)}\|_F). \tag{30}$$

Finally, $\mathbf{Z}_{\mathrm{ref}}^{(K)} - \mathbf{Z}_1 = (1-\tau)(\mathbf{Z}^{(0)} - \mathbf{Z}_1)$. Another application of the triangle inequality proves Eq. (11). □

For $\tau < 1$, the product telescopes to $\omega_{k,K} = (1-\tau)/(1-t_{k+1})$. The product definition also covers $\tau = 1$, where $\omega_{K-1,K} = 1$ and earlier weights are zero. Earlier updates still influence the final residual and velocity through $\mathbf{Z}^{(K-1)}$. The proof requires no Lipschitz assumption on the learned field. It measures its actual residual along the generated trajectory.

Relation to flow regression. When $\mathbf{c} = \mathbf{z}_1$, substituting Eq. (16) into the regression objective gives

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}\left\| \mathbf{v}_\theta(\mathbf{z}_t, t, \mathbf{z}_1) - \frac{\mathbf{z}_1 - \mathbf{z}_t}{1-t} \right\|_2^2. \tag{31}$$

This is the bridge residual evaluated on normal interpolation states, with $t = 1$ having probability zero. The residual in Proposition 2 is evaluated on generated states. This distinction makes the role of the reference explicit: FM supplies a regression penalty, and the trajectory bound accounts for the velocity error actually encountered during sampling. For noisy or null conditions, the same deterministic decomposition holds relative to any chosen normal reference of the same size, with e_k including the effect of the conditioning rule.

Decoded graph deviation. For two graphs with the shared node correspondence, define

$$d_G((\mathbf{X}, \mathbf{A}), (\mathbf{X}', \mathbf{A}')) = \left(\|\mathbf{X} - \mathbf{X}'\|_F^2 + \|\mathbf{A} - \mathbf{A}'\|_F^2\right)^{1/2}. \tag{32}$$

Suppose the feature decoder is L_X-Lipschitz on a region containing $\widehat{\mathbf{Z}}$ and $\mathbf{Z}_1$, whose Frobenius norms are at most R_z. For any $\mathbf{Z}, \mathbf{Z}'$ with norms at most R_z and the same candidate mask, masking is nonexpansive in Frobenius norm, so the sigmoid's $1/4$-Lipschitz property gives

$$\begin{aligned} \|\widehat{\mathbf{A}}(\mathbf{Z}) - \widehat{\mathbf{A}}(\mathbf{Z}')\|_F &\le \tfrac{1}{4}\|\mathbf{Z}\mathbf{Z}^\top - \mathbf{Z}'(\mathbf{Z}')^\top\|_F \\ &\le \tfrac{1}{4}(\|\mathbf{Z}\|_F + \|\mathbf{Z}'\|_F)\|\mathbf{Z} - \mathbf{Z}'\|_F \le \tfrac{R_z}{2}\|\mathbf{Z} - \mathbf{Z}'\|_F. \end{aligned} \tag{33}$$

Consequently, with $L_{\mathrm{dec}} = (L_X^2 + R_z^2/4)^{1/2}$ and reconstruction error $\epsilon_{\mathrm{rec}} = d_G(\mathrm{dec}_\phi(\mathbf{Z}_1), G_1)$,

$$\begin{aligned} d_G(\widetilde{G}, G_1) &\le \epsilon_{\mathrm{rec}} + L_{\mathrm{dec}}\|\widehat{\mathbf{Z}} - \mathbf{Z}_1\|_F \\ &\le \epsilon_{\mathrm{rec}} + L_{\mathrm{dec}}\left[(1-\tau)\|\mathbf{Z}^{(0)} - \mathbf{Z}_1\|_F + h\sum_{k=0}^{K-1}\omega_{k,K}(e_k + r\|\mathbf{V}^{(k)}\|_F)\right]. \end{aligned} \tag{34}$$

This relates the generated graph to its actual normal reference through reconstruction, truncation, velocity residual, and guidance. The distance compares aligned attributes and adjacency weights. Its interpretation is tied to the chosen representation and graph metric.

B.3 Perturbation Bounds for Guided Sampling

A complementary comparison isolates the effect of guidance on the current learned sampler. Let $\mathbf{V}_k(\mathbf{Z})$ denote the configured base velocity at $(\mathbf{Z}, t_k, \mathbf{C})$. Initialize $\mathbf{Z}^{(0)} = \mathbf{Y}^{(0)}$ and use the same conditions in

$$\begin{aligned} \mathbf{Z}^{(k+1)} &= \mathbf{Z}^{(k)} + h(\mathbf{V}_k(\mathbf{Z}^{(k)}) + \mathbf{U}^{(k)}), \\ \mathbf{Y}^{(k+1)} &= \mathbf{Y}^{(k)} + h\mathbf{V}_k(\mathbf{Y}^{(k)}). \end{aligned} \tag{35}$$

Suppose $\mathbf{V}_k$ is L_k-Lipschitz on a set containing both states and $\|\mathbf{U}^{(k)}\|_F \le r_k\|\mathbf{V}_k(\mathbf{Z}^{(k)})\|_F$. The implemented sampler uses $r_k = r$ within each trajectory. Subtracting the two updates and applying the Lipschitz bound gives, for $q_k = \|\mathbf{Z}^{(k)} - \mathbf{Y}^{(k)}\|_F$,

$$q_{k+1} \le (1 + hL_k)q_k + hr_k\|\mathbf{V}_k(\mathbf{Z}^{(k)})\|_F. \tag{36}$$

Starting from $q_0 = 0$, expand this recursion to obtain

$$q_K \le h \sum_{k=0}^{K-1} r_k \|\mathbf{V}_k(\mathbf{Z}^{(k)})\|_F \prod_{i=k+1}^{K-1} (1 + hL_i). \tag{37}$$

If $L_k \le L$, $r_k \le r_{\max}$, and the sampled speeds are bounded by M, the geometric sum gives

$$q_K \le \begin{cases} r_{\max} M((1+hL)^K - 1)/L \le r_{\max} M(e^{L\tau} - 1)/L, & L > 0, \\ r_{\max} M\tau, & L = 0. \end{cases} \tag{38}$$

This result compares guided and unguided trajectories of the same learned field. Proposition 2 instead uses an encoded normal endpoint and explicitly includes velocity error relative to that reference.

B.4 Density Ratio Interpretation of the Detector

Hold fixed the normal and generated distributions P_+ and P_- of the complete detector input $\chi = (G, \mathbf{Z})$. Let p_+ and p_- be their densities relative to a common dominating measure μ, such as $P_+ + P_-$. For a normality probability $\eta(\chi) \in [0, 1]$, equal class weights give the population BCE risk

$$\mathcal{R}(\eta) = -\int [p_+(\chi) \log \eta(\chi) + p_-(\chi) \log(1 - \eta(\chi))] \, \mathrm{d}\mu(\chi). \tag{39}$$

Pointwise minimization over measurable probability functions gives the standard density ratio interpretation (Goodfellow et al., 2014)

$$\eta^*(\chi) = \frac{p_+(\chi)}{p_+(\chi) + p_-(\chi)}, \qquad D^*(\chi) = \log \frac{p_+(\chi)}{p_-(\chi)}. \tag{40}$$

The probability is defined where the density sum is positive, and the finite logit expression requires both densities to be positive. Where only one is positive, the optimum probability is an endpoint of $[0, 1]$. If $P_+ = P_-$, the optimal probability is one half almost everywhere. Thus synthetic supervision depends on the relationship between normal and generated inputs. The score evaluates their learned contrast, and transfer to previously unseen anomaly types remains a property to establish on the target detection tasks.

C Additional Experimental Details and Results

C.1 Datasets and Model Selection

Benchmarks. Table 3 summarizes the benchmark collections following AGDiff (Cai et al., 2025b). COX2, ER_MD, and MUTAG contain molecular graphs, while DD contains protein graphs. MOLT-4, MCF-7, PC-3, and SW-620 contain compounds labeled by anticancer activity; their minority active classes provide the anomalies. The statistics describe the complete benchmark collections, before any experimental split or subsampling.

Table 3: Full-collection dataset statistics reported by AGDiff (Cai et al., 2025b): graph counts, mean nodes and edges per graph, and minority-class proportion. All eight datasets are binary.

Dataset	Graphs	Mean nodes	Mean edges	Anomalies (%)
COX2	467	41.22	43.45	21.84
ER_MD	446	21.33	234.85	40.58
MUTAG	188	17.93	19.79	33.51
DD	1,178	284.32	715.66	41.34
MOLT-4	39,765	26.10	28.14	7.90
MCF-7	27,770	26.40	28.53	8.26
PC-3	27,509	26.36	28.49	9.34
SW-620	40,532	26.06	28.09	5.95

Baselines. The four graph kernels are SP, WL, NH, and RW. The neural methods are OCGIN (Zhao & Akoglu, 2023) and OCGTL (Qiu et al., 2022) for one-class representation learning, GLocalKD (Ma et al., 2022) for distillation, iGAD (Zhang et al., 2022) for discrimination with labeled

anomalies, SIGNET (Liu et al., 2023c) for informative subgraph learning, MUSE (Kim et al., 2024) for reconstruction statistics, and AGDiff (Cai et al., 2025b) for diffusion-based pseudo-anomaly generation. We additionally include ProtoGLAD (Zhao et al., 2026), which uses normal prototype graphs for similarity-based detection and contrastive explanations. These categories describe the original methods; their experimental supervision and adaptation must be distinguished when documenting the compared runs.

Data Split. For small and moderate-scale datasets, we adopt the partitioning protocol used in AGDiff (Cai et al., 2025b): 80% of normal graphs form the training set, and the remaining normal graphs are combined with no more than an equal number of anomalous graphs for testing. On the large-scale imbalanced datasets, we likewise reserve 80% of normal graphs for training, while the test set contains the remaining 20% together with all anomalous graphs.

Network Architecture. The graph representation model and anomaly detector both use a Graph Isomorphism Network (GIN) backbone (Xu et al., 2019), each comprising 3 GIN layers. In the detector, pooled features from individual layers pass through residual MLP projection blocks before producing the final scalar detection logit. A 2-layer MLP encodes the perturbation condition $\mathbf{c}$ within the latent generator, whose velocity network is a 4-layer MLP equipped with sinusoidal time embeddings. The latent dimension is fixed at 16 across all datasets.

Hyper-parameter Setting. We optimize representation pretraining and joint training with Adam (Kingma & Ba, 2014), using learning rates of 0.001 for moderate-scale datasets and 5×10^{-5} for large-scale datasets. The graph autoencoder is pretrained for 100 epochs with a KL-divergence coefficient of 0.001, after which its encoder and decoder are frozen for feature extraction and pseudo-graph reconstruction. Using latents cached by the frozen encoder, the flow-matching generator undergoes a separate 100-epoch pretraining stage at a learning rate of 5×10^{-4}. Pseudo anomalies are then sampled by integrating the guided velocity field. Moderate-scale datasets use the complete trajectory with $\tau = 1$ and initial guidance strength $r = 0.25$. Large-scale datasets use truncated trajectories with $\tau \in [0.05, 0.1]$ and initial $r = 0.5$. Guidance strength is subsequently adjusted adaptively during training. Batch sizes range from 16–64 on moderate-scale datasets and 96–256 on large-scale datasets.

Computing Resources. All experiments are run on a machine equipped with an NVIDIA H20 GPU with 96 GB of memory and an Intel Xeon Platinum 8480+ CPU.

C.2 Latent Placement

Discriminator-guided placement diagnostic. Figure 6 uses COX2 and the discriminator-guided sampler of BNFlow, whose flow model is pretrained for 100 epochs. Each of 11 horizons $\tau \in \{0, 0.1, \ldots, 1\}$ contributes 876 pseudo samples, with three source-noise draws per target graph. Distances are evaluated in the original, unstandardized frozen-VGAE latent space after mean pooling. For a graph-level latent vector $\mathbf{x} \in \mathbb{R}^{d_z}$ and the set $\mathcal{H}_{\mathrm{N}} \subset \mathbb{R}^{d_z}$ of normal training vectors, define

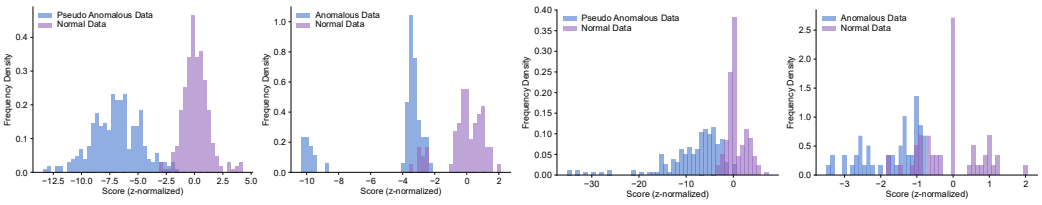
$$d_{\mathrm{N}}(\mathbf{x}) = \min_{\mathbf{h} \in \mathcal{H}_{\mathrm{N}}} \|\mathbf{x} - \mathbf{h}\|_2. \tag{41}$$

For training vectors, the nearest-neighbor search excludes the query itself. Let t_{N} be the 95th percentile of these leave-one-out distances, and let t_{F} be the larger of t_{N} and the 95th percentile of real test-anomaly distances. Normal-like, Near, and Far then mean $d_{\mathrm{N}} \le t_{\mathrm{N}}$, $t_{\mathrm{N}} < d_{\mathrm{N}} \le t_{\mathrm{F}}$, and $d_{\mathrm{N}} > t_{\mathrm{F}}$, respectively. Normal-like samples are further assigned to the closest of three k-means prototypes fitted to normal training vectors, labeled N1–N3 by their first principal-component coordinate. Near is empty when the two thresholds coincide.

Each ribbon encodes the fraction of generated samples at a given horizon assigned to one of these five destinations. Fractions below 0.015 are hidden visually but remain in sector totals. Test anomalies calibrate only this post hoc diagnostic, not training or horizon selection; the plotted label "REAL ANOMALIES" names the distance-band sectors. Because 71 of 73 real anomalies are normal-like under this distance, these regions characterize latent proximity rather than normal/anomalous class membership. At $\tau = 0$, the diagnostic uses the flow source samples, which differ from the local-noise control labeled zero in Figure 4. The chord plot characterizes discriminator-guided sampling across horizons. Without a paired guidance-off comparison or a measure of distance to the detector's

decision boundary, it does not isolate the displacement caused by guidance or establish boundary proximity.

C.3 Supplementary Score Distribution Analysis



(a) COX2 (b) ER_MD

Figure 7: Normality-oriented detector score distributions on (a) COX2 and (b) ER_MD. Each dataset compares pseudo anomalies with normal graphs (left) and real anomalies with normal graphs (right). Within each comparison, scores use median/MAD standardization based exclusively on the normal class, with the same transformation applied to both classes.

Figure 7 complements latent placement with normality-oriented detector scores, where lower values indicate greater abnormality, opposite to the anomaly-score convention $a(G)$ used for evaluation. The horizontal axis uses robust z-scores: within each comparison, we subtract the median normal-class score and divide by 1.4826 times the median absolute deviation (MAD) of normal-class scores. This conventional factor makes the scaled MAD consistent for the standard deviation under a Gaussian reference distribution. Both statistics are computed exclusively from the normal class. The compared real or pseudo anomalies undergo the same affine transformation without contributing to its estimation, preserving score ordering and expressing separation relative to normal-score variability. The vertical axis shows density. On COX2, both pseudo and real anomalies concentrate below the normal scores, with pseudo anomalies exhibiting a broader distribution. ER_MD shows more overlap near the normal-score range, while its pseudo anomalies extend farther into the low-score tail than its real anomalies. Thus, generated negatives and real anomalies share a direction of score separation without having identical distributions. These score-space patterns complement the chord diagnostic: latent proximity to normal graphs can coexist with lower normality-oriented scores.

C.4 Ablation of the Flow-Matching Regression Weight

Table 4: Ablation of continued flow-matching regression on four datasets. AUROC is reported as mean $\pm$ standard deviation (%). The change is the mean at $\lambda = 0$ minus the Full mean, in percentage points.

Dataset	Full	$\lambda = 0$	Change (pp)
COX2	81.37 ± 2.86	67.79 ± 8.10	-13.58
ER_MD	92.56 ± 1.07	88.78 ± 2.67	-3.78
MUTAG	95.68 ± 1.30	65.65 ± 52.15	-30.03
DD	87.00 ± 4.90	61.40 ± 39.98	-25.60

The coefficient λ weights the normal-latent flow-matching regression term in the joint objective. Setting it to zero removes this direct constraint on the velocity field during joint optimization. Table 4 shows lower mean AUROC on all four datasets, with the largest decrease of 30.03 percentage points on MUTAG. The reported standard deviations also increase throughout, particularly on MUTAG and DD, indicating greater variability in detection performance when this regression term is absent. These results support maintaining normal-reference regression alongside synthetic classification, rather than relying on the classification objective alone to shape the generator. The smaller decrease on ER_MD suggests that the contribution varies across datasets. This ablation concerns the joint-training loss weight, whereas the no-FM control in the main ablation replaces learned transport with local perturbations.