DISTRIBUTIONAL MEASUREMENT-BASED MODEL SELECTION FOR ANOMALY DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Anomaly detection (AD) is a fundamental task in data analysis with broad applications in high-stakes domains. Although numerous AD algorithms have been developed over the past decades, this rich methodological landscape makes detector selection increasingly challenging. Existing automated AD methods mainly rely on hyperparameter optimization or meta-learning. Hyperparameter optimization (HO) is fully unsupervised and requires no historical tasks, but its effectiveness depends on surrogate internal criteria and it is often computationally expensive for large-scale datasets. Meta-learning methods transfer selection knowledge from historical tasks by constructing task-level meta-features, yet existing approaches primarily use coarse statistical summaries, which may fail to capture subtle structural and relational patterns that are critical in complex AD tasks. To address these limitations, we propose a distribution-aware meta-learning framework for automated AD model selection. Instead of representing each dataset only by statistical summaries, our method constructs informative and task-guided meta-features through distributional measurements across datasets. Specifically, we first build sample-wise similarity graphs to map heterogeneous datasets into a unified relational representation space. We then compute graph-based distances or similarities between datasets, thereby capturing fine-grained structural and distributional information. Extensive experiments across diverse AD tasks and two modalities show that our method consistently achieves superior or comparable detection performance compared with existing model selection techniques, while substantially reducing online selection time compared with the HO-based approaches. Moreover, the constructed meta-features are robust to domain shifts, allowing our framework to select strong detectors for new tasks from domains not observed in the historical task set.

1 INTRODUCTION

Anomaly detection (AD) aims to identify samples that deviate significantly from normative data patterns. It is a fundamental problem in machine learning and data mining, with broad applications in high-stakes domains such as financial fraud detection, medical diagnosis, industrial inspection, and cybersecurity intrusion detection. Over the past decades, a large number of AD algorithms have been developed, ranging from classical shallow methods, such as KNN-based detector (Ramaswamy et al., 2000), Local Outlier Factor (Breunig et al., 2000), Isolation Forest (Liu et al., 2008), and One-Class SVM (Schölkopf et al., 2001), to deep learning-based methods based on reconstruction error (Hinton & Salakhutdinov, 2006; Seeböck et al., 2016; Vincent et al., 2008; Schlegl et al., 2017; Deecke et al., 2019; Perera et al., 2019), representation learning (Zong et al., 2018; Hu et al., 2020; Qiu et al., 2021; Cai & Fan, 2022), hypersphere learning (Ruff et al., 2018; Xiao et al., 2023; Zhang et al., 2023), and deep density estimation (Goyal et al., 2020; Ruff et al., 2021; Fu et al., 2024; Xiao et al., 2025). This proliferation of algorithms has greatly enriched the methodological landscape of AD. However, it also raises a critical practical question: *Given a new unsupervised AD task, which algorithm and corresponding hyperparameter configuration should be used?*

The prior studies (Campos et al., 2016; Bahri et al., 2022; Han et al., 2022; Dai & Fan, 2025; Zhou et al., 2025) demonstrate that no single anomaly detector can dominate across all possible datasets. Different AD algorithms rely on different inductive biases: distance-based methods (Schölkopf et al., 2001; Ruff et al., 2018) are sensitive to neighborhood structures, density-based methods (Breunig et al., 2000; Kim & Scott, 2012) favor local density contrast, isolation-based methods (Liu et al., 2008;

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.

Xu et al., 2023) exploit random partitioning behavior, and deep representation-based methods (Zhang et al., 2023; Xiao et al., 2025) often assume learnable low-dimensional or compact normal patterns. As a result, the best-performing detector can vary substantially (Zhao et al., 2021; Dai & Fan, 2025) across datasets with different distributional structures, dimensionality, noise levels, and anomaly mechanisms. This observation is also consistent with the no free lunch principle (Wolpert & Macready, 2002), which suggests that universally optimal algorithms should not be expected without assumptions on the underlying task distribution. Therefore, automatic model selection, including both algorithm and hyperparameter configuration, is paramount for unsupervised AD in real-world detection scenarios.

To address this challenge, existing studies mainly follow two lines of research. The first line is meta-learning-based [1] model selection (Zhao et al., 2021; 2022; Qin et al., 2024), which transfers knowledge from historical AD tasks to a new task by learning the relationship between meta-features and detector performance. The second line is HO-based model selection, such as AutoUAD (Dai & Fan, 2025), which searches for suitable configurations using unsupervised internal criteria or surrogate objectives based on Bayesian optimization (Jones et al., 1998) and attempts to optimize anomaly detectors without requiring labels on the new task. Despite this progress, there are still several limitations.

- **High online inference cost of HO.** HO-based methods require repeatedly evaluating candidate models under different hyperparameter configurations on the new dataset. This leads to high online inference cost, especially for large-scale and high-dimensional data. Moreover, since ground-truth labels are unavailable, their optimization objectives must rely on internal surrogate criteria, whose consistency with true detection performance is not always guaranteed.
- **Limited expressiveness of statistical meta-features.** For meta-learning-based methods, their effectiveness critically depends on the quality of the constructed meta-features. Existing works commonly use statistical characteristics and landmark features (simple detectors' outputs). These descriptors are useful but often provide only coarse global summaries of datasets. They may fail to capture fine-grained distributional structures that are crucial for anomaly detection, such as local density variation, manifold geometry, and neighborhood heterogeneity. Consequently, two datasets with similar statistical moments may require different detectors, while two datasets with different raw statistics may share similar structural properties and prefer the same detector.
- **Dependence and semantic mismatch of text-based meta-features.** Recent work also explores language model-based textual representations for model selection, for example, by embedding dataset or task descriptions (Qin et al., 2024). While such representations can exploit semantic information, they depend heavily on the availability, completeness, and quality of textual descriptions. Textual descriptions may not faithfully reflect the actual sample distribution that determines anomaly detector behavior. This can introduce semantic noise and weaken the connection between the meta-feature and the structural properties relevant to unsupervised AD.

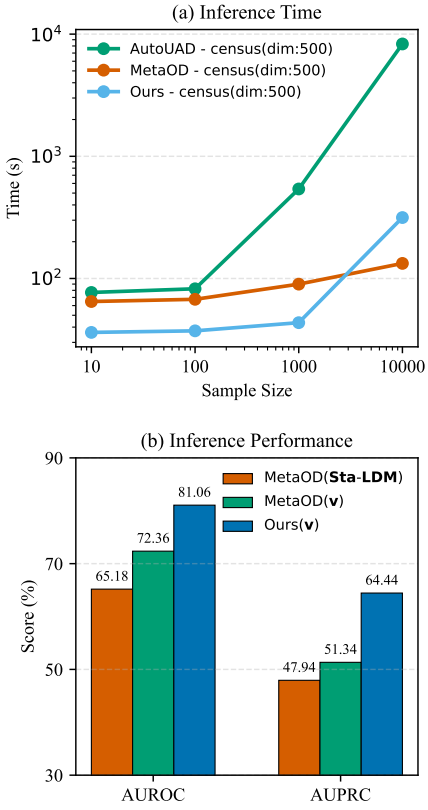


Figure 1: Comparison in terms of inference time and detection performance. The **v** refers to the meta-feature constructed using our proposed method. There are different selection processes between MetaOD with our method.

A visualization comparison is presented in Figure 1. The plot (a) compares the inference time on *census* dataset under varying sample sizes, where AutoUAD (Dai & Fan, 2025) is a HO-based method and MetaOD (Zhao et al., 2021) is a meta-learning-based method. The primary inference time cost for both MetaOD and our method arises from constructing the meta-features for a new detection task. The

[1]We use 'meta-learning' in the AutoML sense, i.e., learning from historical tasks based on task-level meta-features and detector performance.

plot (b) compares the average detection performance of the selected algorithms across 47 detection tasks (evaluated by *leave-one-out* protocol), between MetaOD and ours. The **Sta-LDM** denotes the original meta-features (statistical characteristics and landmarker features) used in MetaOD and the $\mathbf{v}$ refers to the meta-feature constructed using our proposed method. The improved average detection performance on MetaOD demonstrates that the meta-features $\mathbf{v}$ are more informative than **Sta-LDM**.

To overcome these limitations, we propose **Distributional Measurement-based Model Selection for Anomaly Detection**, a meta-learning framework that constructs informative and task-guided meta-features through distributional measurement. The key idea is to characterize each dataset not by handcrafted statistics or textual descriptions, but by its sample-wise relational structure. Specifically, we first construct sample-wise similarity graphs for historical datasets, thereby mapping heterogeneous datasets with different dimensions, feature scales, and raw semantics into a unified graph-based relational space. We then compute graph similarities/dissimilarities between datasets using complementary distributional measurements, including the Weisfeiler-Lehman kernel (Shervashidze et al., 2011) and Minimum Factorization Distance (Fan, 2025). The resulting graph-induced meta-features encode both local and global distributional structures, such as neighborhood connectivity, density variation, and geometric discrepancy. Since these properties are closely related to the inductive biases of anomaly detectors, the proposed meta-features are more task-relevant for AD model selection.

The proposed framework follows the classical offline-online paradigm (Rice, 1976; Brazdil et al., 1994; Zhao et al., 2021; 2022). In the offline stage, we collect historical anomaly detection datasets and evaluate a finite pool of candidate models, where each model corresponds to an algorithm and hyperparameter configuration pair. We then construct graph-induced meta-features for historical datasets and train a meta-learner to predict candidate model performance by using the constructed meta-features. In the online stage, given a new dataset without any label, we only need to construct its graph-induced meta-feature, compare it with historical datasets in the learned distributional space via the meta-learner, and directly select the predicted best model. This design avoids expensive online hyperparameter search while preserving the ability to exploit historical knowledge from structurally similar tasks. We name the proposed method **GOOD**, which stands for **G**raph-based **O**ffline-**O**nline Model Selection for Anomaly **D**etection. The main contributions are summarized as follows:

- We study the problem of automatic model selection for unsupervised anomaly detection and propose a distributional measurement-based meta-learning framework that selects both the anomaly detection algorithm and its hyperparameter configuration for a new unlabeled dataset.
- We propose a graph-induced meta-feature construction strategy that maps heterogeneous datasets into a unified relational space through sample-wise similarity graphs, capturing fine-grained structural information such as local neighborhood patterns, global distribution discrepancies.
- We conduct extensive experiments across diverse anomaly detection tasks and modalities. The results demonstrate that the proposed method consistently outperforms or performs on par with existing model selection techniques in detection performance while reducing online selection cost compared with HO-based approaches. Moreover, the constructed meta-features exhibit robustness to domain shifts, enabling our framework to effectively select strong detectors for new tasks from domains not present in the historical task set.
- We release the code, historical datasets, constructed meta-features to facilitate extendable research. Our project is available at `https://anonymous.4open.science/r/GOOD/`.

2 Related Work

Model selection is a central problem in automated machine learning (AutoML) (Bahri et al., 2022), where the goal is to automatically identify suitable learning algorithms and hyperparameter configurations for a given task (Hutter et al., 2019; He et al., 2021). For anomaly detection, existing model selection methods can be broadly categorized into meta-learning-based methods (Zhao et al., 2021; Qin et al., 2024; Sylligardos et al., 2023; 2025; Zhou et al., 2025; Abdelmaksoud et al., 2026) and hyperparameter optimization-based methods (Zhao & Akoglu, 2024; Dai & Fan, 2025). The former addresses model selection by learning from historical anomaly detection tasks. The latter aims to select suitable hyperparameters and models directly on the new unlabeled dataset. Detailed discussion on these works are deferred to Appendix A.

3 OUR METHOD

3.1 PROBLEM STATEMENT

In this work, we consider the model selection problem for unsupervised anomaly detection (UAD). Given a new dataset $\mathbf{X}_{\text{new}}$, without any labels, the problem is to select both (*i*) an UAD algorithm and (*ii*) its associated hyperparameter configuration (HPC). The former is a discrete choice, given the finite set of existing detection algorithms. The latter is commonly continuous, and hence induces infinitely many candidates. The infinite search and evaluation are impossible for real scenarios. Therefore, the relevant studies (Zhao et al., 2021; 2022; Zhao & Akoglu, 2024; Qin et al., 2024; Dai & Fan, 2025) introduce a finite pool of models, where there are mainly two different ways to construct the pool of models: (1) a discrete hyperparameter configuration set given by meta-learning-based methods (Zhao et al., 2021; 2022; Qin et al., 2024); (2) a continuous search range of hyperparameter configuration and a predefined search step given by hyperparameter optimization-based methods (Zhao & Akoglu, 2024; Dai & Fan, 2025). Due to the nature of meta-learning of the proposed method, we follow the first strategy to construct a finite pool of models denoted $\mathcal{M} = \{M_1, M_2, \cdots, M_m\}$. Each model $M_j \in \mathcal{M}$ is denoted by a $(\text{Algorithm}, \text{HPC})$ pair.

3.2 MODEL SELECTION VIA TASK SIMILARITY

Model selection via task similarity is a direct idea based on meta-learning, in which historical detection tasks are utilized to build a meta-learner. As such, our proposed method relies on:

- a collection of historical anomaly detection datasets $\mathcal{D}_{\text{his}} = \{D_1, D_2, \cdots, D_n\}$, namely, a meta-train database with labels [2], where $D_i = (\mathbf{X}_i, \mathbf{y}_i)$;
- the historical performances of the pool of candidate models $\mathcal{M}$ on the meta-train database $\mathcal{D}_{\text{his}}$. We construct performance matrix $\mathbf{P} \in \mathbb{R}^{n \times m}$, where element p_{ij} corresponds to the model M_j's performance on the historical dataset $\mathbf{X}_i$.

Similar to meta-learning-based methods of unsupervised anomaly detection model selection, our selection process consists of two stages: **offline meta-training** of the meta-learner on historical database $\mathcal{D}_{\text{his}}$ and performance matrix $\mathbf{P}$, and **online model selection** for a new anomaly detection task (a new dataset without any labels) $\mathbf{X}_{\text{new}}$. A framework overview is illustrated in Figure 2.

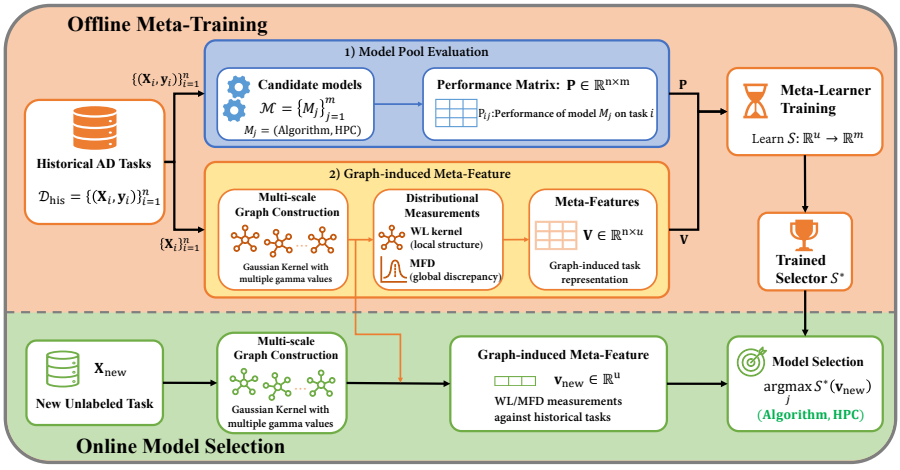


Figure 2: The overview of our proposed method.

Offline Meta-Training Let $\mathbf{p}_i = E([M_1(\mathbf{X}_i), M_2(\mathbf{X}_i), \cdots, M_m(\mathbf{X}_i)], \mathbf{y}_i) \in \mathbb{R}^m$ where $E(\cdot, \cdot)$ denotes a performance evaluation metric. Based on the historical database $\mathcal{X}_{his} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_n\}$, we construct meta-features $\mathbf{V} \in \mathbb{R}^{n \times u}$ with $\mathbf{v}_i \in \mathbb{R}^u$ as the meta-feature of dataset $\mathbf{X}_i$. During the meta-training, we expect to learn a mapping $S : \mathbb{R}^u \to \mathbb{R}^m$, namely a meta-learner, which

[2]The labels is only utilized to evaluate the model performance on historical database, which however is not the case for any new task at the selection stage.

builds a transformation between meta-feature to performance evaluation. We defer the details on the construction of the meta-feature in Section 3.3 and on the selection of the meta-learner in Appendix G.

Online Model Selection After meta-training, we obtain trained-well meta-learner S^*. Given a new dataset $\mathbf{X}_{\text{new}}$, the corresponding meta-feature $\mathbf{v}_{\text{new}} \in \mathbb{R}^u$ is constructed first, and then we select the model $M_j =$ (Algorithm, HPC) based on

$$\arg\max_j S^*(\mathbf{v}_{\text{new}}). \tag{1}$$

3.3 Meta-Features of Distributional Measurement

Due to the inherently heterogeneous nature of datasets, such as varying dimensionality, feature scales, and raw feature semantics, constructing a unified and comparable meta-feature representation remains a significant challenge. Existing works (Zhao et al., 2021; Qin et al., 2024) often fail to capture the intrinsic structural properties of data distributions that are critical for anomaly detection performance. To address these limitations, we propose to construct a meta-feature based on distributional measurement. Specifically, we construct sample-wise similarity graphs to map heterogeneous datasets into a unified relational space, and then quantify inter-dataset relationships through graph-based distributional similarities/dissimilarities. This representation preserves fine-grained structural information, including local geometry and connectivity patterns, and is inherently invariant to differences in raw feature space. As a result, the proposed meta-feature is both more expressive and more task-relevant, as it aligns closely with the underlying factors that govern anomaly detector behavior.

3.3.1 Sample-wise Similarity Graph

To unify the heterogeneous historical datasets $\mathcal{X}_{\text{his}} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n\}$ into a comparable relational space, we construct a sample-wise similarity graph for each dataset. Specifically, let historical dataset $\mathbf{X}_i = \{\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \ldots, \mathbf{x}_{n_i}^{(i)}\}$ contain n_i samples. The similarity matrix $\mathbf{G}_i^{(\sigma)} \in \mathbb{R}^{n_i \times n_i}$ is defined as:

$$[\mathbf{G}_i^{(\sigma)}]_{jk} = \sigma(\mathbf{x}_j^{(i)}, \mathbf{x}_k^{(i)}), \quad \mathbf{x}_j^{(i)}, \mathbf{x}_k^{(i)} \in \mathbf{X}_i, \tag{2}$$

where σ denotes a suitable similarity measure or kernel function. The induced set of weighted graphs across all historical datasets is denoted as:

$$\mathcal{G}^{(\sigma)} = \{\mathbf{G}_1^{(\sigma)}, \mathbf{G}_2^{(\sigma)}, \ldots, \mathbf{G}_n^{(\sigma)}\}.$$

3.3.2 Graph-induced Meta-feature

Once heterogeneous historical datasets $\mathcal{X}_{\text{his}}$ are transformed into sample-wise similarity graphs $\mathcal{G}^{(\sigma)}$, the task of datasets' similarity reduces to quantifying the similarity or dissimilarity between these graphs. Formally, this can be cast as a graph comparison problem:

Definition 1 (Graph Comparison Problem (Wills & Meyer, 2020))**.** *Given two graphs $\mathbf{G}_1$ and $\mathbf{G}_2$ in the space of graphs $\mathcal{G}$, the graph comparison problem seeks a mapping*

$$s : \mathcal{G} \times \mathcal{G} \to \mathbb{R}, \tag{3}$$

such that $s(\mathbf{G}_1, \mathbf{G}_2)$ quantifies the similarity or dissimilarity between the two graphs.

Graph comparison is a quite challenging problem and several fundamental problems related graph comparison such as subgraph isomorphism and the maximum common subgraph have been proved to be NP-complete (Garey & Johnson., 1983). Due to the ubiquitous nature of graph structured data in real world, such social networks, molecule structures, the related research still receives much attention and numerous graph comparison methods (Bunke, 1997; Shervashidze et al., 2011; Mémoli, 2011; Peyré et al., 2016) are founded in past decades. Despite such progresses, most of these algorithms have a cubic even higher computational cost.

To balance *computational tractability* and *task relevance for anomaly detection*, we adopt two complementary approaches: the **Weisfeiler-Lehman (WL) subtree kernel** (Shervashidze et al., 2011) and the **Minimum Factorization Distance (MFD)** (Fan, 2025). This choice and combination is not unique. Further investigation of alternative similarity/dissimilarity measures between graphs is provided in the Appendix H.

- **WL kernel:** The WL kernel iteratively aggregates local neighborhood labels to encode subgraph structures, capturing fine-grained connectivity patterns that determine the effectiveness of different anomaly detectors. By preserving neighborhood regularity, it provides a principled means to differentiate datasets with subtle structural variations, which is crucial for selecting models sensitive to local density or micro-cluster deviations.
- **MFD:** While local structures are critical, global distribution patterns strongly influence anomaly detector performance, particularly for methods relying on global density or manifold assumptions. MFD measures global distributional discrepancies across graphs by comparing their factorized representations, effectively capturing macro-scale differences such as cluster separation, density heterogeneity, and high-order structural correlations. Importantly, MFD provides a computationally efficient alternative compared with Gromov-Wasserstein distance ($\mathcal{O}(n^3)$ (Peyré et al., 2016)), making large-scale pairwise comparison feasible.

Based on the two graph-based metrics, we obtain similarity/dissimilarity matrices $\mathbf{F}_{\text{WL}}^{(\sigma)}, \mathbf{F}_{\text{MFD}}^{(\sigma)} \in \mathbb{R}^{n\times n}$ among graphs. The element $[\mathbf{F}_{\text{WL}}^{(\sigma)}]_{i,j}$, $[\mathbf{F}_{\text{MFD}}^{(\sigma)}]_{i,j}$ are measured by

$$[\mathbf{F}_{\text{WL}}^{(\sigma)}]_{i,j} = \text{WL}(\mathbf{G}_i^{(\sigma)}, \mathbf{G}_j^{(\sigma)}), \;\; [\mathbf{F}_{\text{MFD}}^{(\sigma)}]_{i,j} = \text{MFD}(\mathbf{G}_i^{(\sigma)}, \mathbf{G}_j^{(\sigma)}), \;\; \mathbf{G}_i^{(\sigma)}, \mathbf{G}_j^{(\sigma)} \in \mathcal{G}^{(\sigma)} \tag{4}$$

where $\text{WL}(\cdot,\cdot), \text{MFD}(\cdot,\cdot)$ denote the corresponding similarity/dissimilarity measure. Gaussian Kernel GK_γ is applied to instantiate σ

$$\text{GK}_\gamma = \exp\Big(-\gamma \frac{\|\mathbf{x}-\mathbf{x}'\|^2}{d^2}\Big), \;\; \mathbf{x}, \mathbf{x}' \in \mathbf{X} \tag{5}$$

where $d = \mathbb{E}_{\mathbf{x},\mathbf{x}'\in\mathbf{X}}\|\mathbf{x}-\mathbf{x}'\|$ denotes average sample distance on a certain historical dataset $\mathbf{X}$. We adopt a multi-scale strategy by varying the Gaussian kernel parameter γ over a predefined range, generating multiple similarity graphs for each dataset. This design enables capturing structural patterns at different resolutions, ranging from global geometry to fine-grained local neighborhoods. Such multi-scale characterization is particularly important for anomaly detection, where anomalies may manifest at different levels (e.g., global outliers, local density deviations, or micro-clusters). Compared to using a single γ, this strategy mitigates the strong inductive bias imposed by a fixed neighborhood definition and improves robustness across heterogeneous datasets. Furthermore, different anomaly detection algorithms are inherently sensitive to different scales, and the proposed multi-graph construction implicitly aligns with these preferences, leading to more task-relevant similarity estimation and improved model selection performance. The related ablation study on γ is provided in Appendix C.3.

Now, we construct the graph-induced meta-features $\mathbf{V}$ on historical datasets $\mathcal{X}_{\text{his}}$ as follows:

$$\mathbf{V} = \Big[\mathbf{F}_{\text{WL}}^{(\text{GK}_{\gamma_1})}; \cdots; \mathbf{F}_{\text{WL}}^{(\text{GK}_{\gamma_k})}; \mathbf{F}_{\text{MFD}}^{(\text{GK}_{\gamma_1})}; \cdots; \mathbf{F}_{\text{MFD}}^{(\text{GK}_{\gamma_k})}\Big] \in \mathbb{R}^{n\times n(2k)}, \tag{6}$$

where $[;;]$ denotes the column-wise concatenation and $\mathbf{v}_i \in \mathbb{R}^{n(2k)}$ is the meta-feature of the historical dataset $\mathbf{X}_i$.

During inference, given a new detection task $\mathbf{X}_{\text{new}}$ without any label, we construct weighted graph $\mathbf{G}_{\text{new}}^{(\sigma)}$ first, and then the meta-feature $\mathbf{v}_{\text{new}}$ is computed by

$$\mathbf{v}_{\text{new}} = \Big[\mathbf{f}_{\text{WL}}^{(\text{GK}_{\gamma_1})}; \cdots; \mathbf{f}_{\text{WL}}^{(\text{GK}_{\gamma_k})}; \mathbf{f}_{\text{MFD}}^{(\text{GK}_{\gamma_1})}; \cdots; \mathbf{f}_{\text{MFD}}^{(\text{GK}_{\gamma_k})}\Big] \in \mathbb{R}^{n(2k)}, \tag{7}$$

where the $\mathbf{f}_{\text{WL}}^{(\sigma)}, \mathbf{f}_{\text{MFD}}^{(\sigma)} \in \mathbb{R}^n$ and the element $[\mathbf{f}_{\text{WL}}^{(\sigma)}]_i$ and $[\mathbf{f}_{\text{MFD}}^{(\sigma)}]_i$ are measured

$$\begin{cases} [\mathbf{f}_{\text{WL}}^{(\sigma)}]_i = \text{WL}(\mathbf{G}_i^{(\sigma)}, \mathbf{G}_{\text{new}}^{(\sigma)}), \;\; \mathbf{G}_i^{(\sigma)} \in \mathcal{G}^{(\sigma)}, \\ [\mathbf{f}_{\text{MFD}}^{(\sigma)}]_i = \text{MFD}(\mathbf{G}_i^{(\sigma)}, \mathbf{G}_{\text{new}}^{(\sigma)}), \;\; \mathbf{G}_i^{(\sigma)} \in \mathcal{G}^{(\sigma)}. \end{cases} \tag{8}$$

This construction yields a rich, multi-scale, and task-relevant representation of each dataset that effectively encodes both local connectivity and global structural discrepancies, forming the foundation for accurate and robust anomaly detection model selection. The comparison of different graph

construction strategies is provided in Appendix C.4. The detailed definition and ablation study on WL kernel and MFD are provided in Appendix C. We analyze the time complexity of meta-feature construction during the online model selection in Appendix E and a more lightweight variant is provided in Appendix F.

Relation between Structural Similarity between detector selection. The WL kernel characterizes local neighborhood organization, including connectivity and density variation, which is particularly relevant to detectors relying on neighborhood structures. MFD provides a complementary characterization of global geometry and distributional discrepancy, which can reflect properties such as cluster separation, support, and dispersion. Therefore, datasets that are close under these measurements may exhibit similar relative suitability of candidate detectors. However, we emphasize that this is an inductive relationship rather than a universal guarantee, as it depends on whether the constructed similarity graphs preserve structures relevant to the underlying anomaly mechanisms. The results on two testbeds in Section 4.1 provide empirical support, while Section 4.2 further quantitatively examines the relationship between structural similarity and detector-ranking similarity.

4 EXPERIMENTS

We construct a large and heterogeneous model set $\mathcal{M}$ that incorporates 12 AD algorithms encompassing both classical and deep methods, resulting in a total of 358 unique $(\text{Algorithm}, \text{HPC})$ pairs, e.g., (KNN, $\{k$:5; $distance$:euclidean; $method$:mean$\}$). Our evaluation leverages the benchmark datasets derived from ADBench(Han et al., 2022), covering both tabular and image domains. We instantiate a simple neural network as a meta learner, and the details and discussion on meta-learner is provided in Appendix G. We compare against eight strong baselines. We clearly summarize the experimental setups in Table 1. More description on experimental details is provided in Appendix B.

Table 1: Summary of Experimental Setups

Components	Details	Notes
Testbeds (B.1)	**Tabular Testbed**: 47 detection tasks **Image Testbed**: 61 detection tasks	Derived from ADBench (Han et al., 2022)
Model Set (B.2)	12 AD algorithms (classical + deep)	Total 358 (Algorithm, HPC) pairs
Baselines (B.3)	Common-used AD algorithms Hyperparameter-free and latest algorithms General meta-learners Tailored meta-learners Hyperparameter optimization	KNN (Ramaswamy et al., 2000), DeepSVDD (Ruff et al., 2018) COPOD (Li et al., 2020), MROT (Montesuma et al., 2025) ISAC (Kadioglu et al., 2010), AS (Nikolić et al., 2013), MC (Ma et al., 2021) MetaOD (Zhao et al., 2021), ELECT (Zhao et al., 2022), MetaOOD (Qin et al., 2024) AutoUAD (Dai & Fan, 2025)
Empirical Upper Bound (EUB)	Maximum achievable performance over model set $\mathcal{M}$	Reported for both testbeds
Evaluation Metrics	AUROC (%), AUPRC (%)	Leave-one-out experiments on all tasks
Statistical Test	Paired t-test	Significance threshold: $p < 0.05$

4.1 THE RESULTS ON TWO TESTBEDS

In this section, we analyze the main results on two testbeds with different modalities. Specifically, Table 2 reports the average model selection performance (*leave-one-out* experiments) and p-value from paired *t*-tests on the **Tabular Testbed** and **Image Testbed**. The overall performance of each baseline on all tasks is provided in Appendix J.

Table 2: The average detection performance (*leave-one-out*) on two testbeds with different modalities. The reported [p-value]s are obtained from the paired *t*-tests between our proposed method and the baselines.

Methods	**Tabular Testbed** (47 tasks)		**Image Testbed** (61 tasks)	
	AUROC (%) [p-value]	AUPRC (%) [p-value]	AUROC (%) [p-value]	AUPRC (%) [p-value]
EUB	$\overline{90.29}$	$\overline{79.56}$	$\overline{82.89}$	$\overline{69.63}$
KNN	74.74 [.1695]	51.91 [.0467]	79.96 [.1265]	52.47 [.0157]
DSVDD	71.87 [.0514]	44.32 [.0003]	67.47 [.0000]	36.52 [.0000]
COPOD	75.65 [.0020]	46.39 [.0000]	68.83 [.0000]	33.36 [.0000]
MROT	73.72 [.0030]	40.75 [.0000]	69.43 [.0000]	29.52 [.0000]
MC	74.68 [.0005]	53.10 [.0019]	71.67 [.0036]	43.30 [.0000]
AS	71.92 [.0001]	50.51 [.0000]	72.53 [.0000]	58.70 [.4144]
ISAC	78.61 [.0575]	57.79 [.0116]	78.93 [.0120]	60.59 [.9188]
MetaOD	65.18 [.0011]	47.94 [.0056]	68.14 [.0000]	**60.73** [.8623]
ELECT	79.85 [.1524]	54.80 [.0129]	79.64 [.0320]	53.91 [.0369]
MetaOOD	75.41 [.0176]	56.31 [.0281]	76.82 [.0000]	57.22 [.2190]
GOOD (Ours)	**81.06**	**64.44**	**80.56**	60.34

Observing the Table 2, the selection performance of our proposed method significantly outperforms or performs on par with all the baselines. More importantly, on **Tabular Testbed**, the proposed method displays a more marked advantage (>8% gains in most cases) on AUPRC (a metric which is

particularly informative under the imbalanced nature of anomaly detection) compared with the meta-feature-based methods, which empirically demonstrates that our constructed meta-feature depicts the intrinsic similarities between tasks.

Table 3 compares our method with AutoUAD (Dai & Fan, 2025), a latest hyperparameter optimization-based anomaly detection model selection method. HO-based methods require repeatedly evaluating candidate models under different hyperparameter configurations on the overall dataset based on surrogate metrics. To control the experimental scale, we select the top two detectors (OCSVM and AE) from AutoUAD to conduct experiments. For AE, our method consistently outperforms all AutoUAD variants on both testbeds and both metrics. For OCSVM, AutoUAD(EAG) achieves higher AUPRC on both testbeds, and our method still maintains competitive AUPRC while providing stronger AUROC performance. *Note that the model selection space for AutoUAD and our method is identical in this comparison*. The inference time comparison is provided in Figure 1.

Table 3: The average performance comparison (*leave-one-out*) with hyperparameter optimization-based methods.

Detector	Methods	**Tabular Testbed** (47 tasks) AUROC (%)	AUPRC (%)	**Image Testbed** (61 tasks) AUROC (%)	AUPRC (%)
AE	AutoUAD(RTM)	78.96	53.86	76.35	43.87
	AutoUAD(EAG)	78.11	53.34	75.97	43.79
	AutoUAD(NPD)	78.46	55.04	74.82	42.80
	GOOD (Ours)	**80.80**	**58.55**	**79.11**	**50.76**
OCSVM	AutoUAD(RTM)	72.82	57.63	71.41	48.41
	AutoUAD(EAG)	75.48	**58.90**	74.58	**52.17**
	AutoUAD(NPD)	79.65	58.10	75.61	44.07
	GOOD (Ours)	**80.84**	56.62	**77.11**	44.59

In summary, the empirical results demonstrate two main advantages of the proposed method. First, it consistently improves model selection performance across heterogeneous tabular and image anomaly detection tasks compared with statistical or text-based meta-learning baselines. Second, compared with hyperparameter optimization-based methods, our framework achieves competitive or better detector selection without expensive online search.

4.2 Relationship Between Structural Similarity and Detector-Ranking Similarity

To directly examine whether the proposed graph measurements capture task properties relevant to model selection, we compare pairwise structural similarity with pairwise detector-ranking similarity across the tabular testbed (47 tabular tasks). For each pair of datasets, detector-ranking similarity is measured by the Spearman correlation between their performance vectors over the 358 candidate models. We then correlate these pairwise ranking similarities with WL similarity and negative MFD at each graph scale γ. As a reference, we construct dataset similarities from the original MetaOD meta-features using cosine similarity. The detailed computation procedure and complete results for both AUROC and AUPRC are provided in Appendix D.

Figure 3 shows a correlation between data similarity and detector-ranking similarity based on Pearson correlation/Spearman rank correlation coefficient. These correlations of WL are substantially higher than those obtained using MetaOD meta-features. MFD exhibits relatively weaker correlations than WL, suggesting that its global structural information provides a complementary rather than dominant signal. These results do not imply that structural similarity universally determines detector rankings. Rather, compared with the superior model-selection performance of GOOD reported in Table 2, they provide more direct quantitative evidence for the advantages of distributional measurement-based meta-features over the statistical meta-features in capturing detector-relevant task relationships.

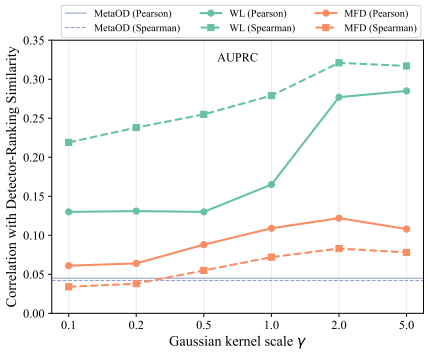


Figure 3: The correlation between structural similarity (WL, negative MFD) and detector-ranking similarity based on AUPRC.

4.3 The Generalization of Meta-Feature

Due to the nature of measuring distributional discrepancies of the constructed meta-features, they are expected to be robust to domain shifts, even modality transfer. To verify the generalization capability of the constructed meta-features, we design the following experiments.

Generalization Experiments on Entirely New Domains. The **Tabular Testbed** includes datasets from diverse domains, such as healthcare, astronautics, and chemistry. To evaluate the generalization of the meta-features $\mathbf{V}$ on new detection tasks from entirely unseen domains, we uniformly split datasets from the same domain into two groups: $[\text{Test}_{(\text{domain})}]$ and $[\text{Hist}_{(\text{domain})}]$, where $[\text{Test}_{(\text{domain})}]$ serves as the test set and $[\text{Hist}_{(\text{domain})}]$ is used to construct the historical set either including or excluding datasets from the corresponding domain. We conduct experiments across seven different domains. The results are illustrated in Figure 4, where 'w' indicates that the historical datasets include some datasets from the corresponding domain, and 'w/o' indicates that no datasets from the corresponding domain are included. The results show that the selection performance of our proposed method is largely insensitive to whether historical datasets from the same domain are included. This demonstrates that our proposed method maintains strong performance even when facing entirely unseen tasks, implying that the constructed meta-features are effectively domain-agnostic due to their distributional measurement nature.

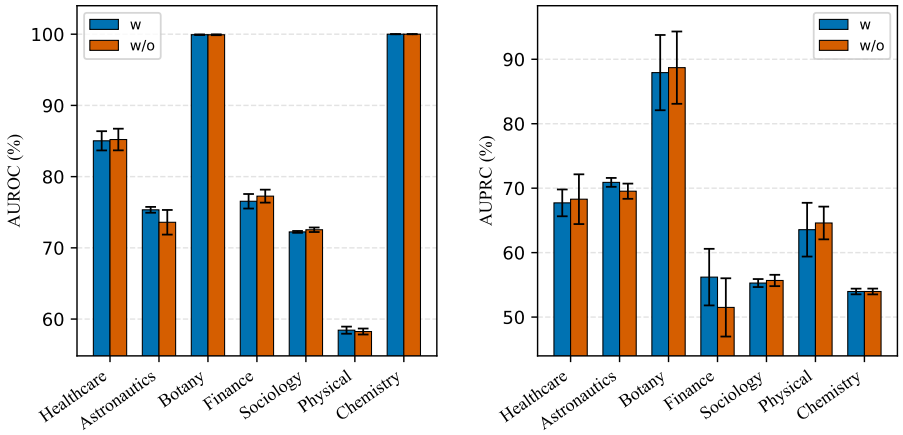


Figure 4: The check experiments with (w) $[\text{Hist}_{(\text{domain})}]$ and without (w/o) $[\text{Hist}_{(\text{domain})}]$ on different domains.

***Leave-One-Out* Experiments with Modality Exchange.** We exchange the historical datasets between **Tabular Testbed** and **Image Testbed** and conduct *leave-one-out* experiments. The results are reported in Table 4. It can be observed that the average performance on both modalities decreases by approximately 2% after fully exchanging the historical datasets. This demonstrates that the constructed meta-features remain effective even when the training and test modalities are entirely different. Note that we do not conduct dataset-specific tailoring; instead, we directly use all datasets of the same modality from ADBench (Han et al., 2022).

Table 4: The ablation study (*leave-one-out* experiments) across modalities on our constructed meta-features.

TestBed	$\mathcal{D}_{\text{his}}$:=**Tabular Testbed**		$\mathcal{D}_{\text{his}}$:=**Image Testbed**	
	AUROC (%)	AUPRC (%)	AUROC (%)	AUPRC (%)
Tabular Testbed (47 tasks)	81.06	64.44	79.26	62.05
Image Testbed (61 tasks)	78.32	58.29	80.56	60.34

5 Conclusion and Future Work

In this work, we present a distributional measurement-based framework for model selection in unsupervised anomaly detection. By constructing sample-wise similarity graphs and deriving graph-induced meta-features, our approach captures fine-grained structural and relational properties of datasets that are not well represented by conventional statistical or text-based meta-features. Extensive experiments across tabular and image modalities demonstrate that the proposed method consistently outperforms or performs on par with the latest state-of-the-art baselines under *leave-one-out* evaluations. Moreover, the constructed meta-features exhibit strong generalization to unseen domains and robustness to modality shifts, highlighting their task relevance and domain-agnostic nature. Consequently, this framework is naturally extendable to additional modalities, such as time-series. In future work, we will investigate the effectiveness of our proposed method across more complex data modalities, despite potential challenges in representation. Overall, our framework provides a unified, effective, and scalable solution for anomaly detector selection, facilitating the reliable deployment of unsupervised AD models in diverse real-world applications.

AI USE STATEMENT

Large language model (LLM) tools were used only to assist with language polishing and improving the clarity of the manuscript. All research ideas, methodology, experiments, analysis, and conclusions were developed and verified by the authors.

REFERENCES

Mohamed Abdelmaksoud, Sheng Ding, Andrey Morozov, and Ziawasch Abedjan. Ramses: Robust and adaptive model selection for time-series anomaly detection algorithms. *arXiv preprint arXiv:2602.21766*, 2026.

Maroua Bahri, Flavia Salutari, Andrian Putina, and Mauro Sozio. Automl: state of the art with a focus on anomaly detection, challenges, and research directions. *International Journal of Data Science and Analytics*, 14(2):113–126, 2022.

Roel Bouman, Zaharah Bukhsh, and Tom Heskes. Unsupervised anomaly detection algorithms on real-world data: how many do we need? *Journal of Machine Learning Research*, 25(105):1–34, 2024.

Pavel Brazdil, João Gama, and Bob Henery. Characterizing the applicability of classification algorithms using meta-level learning. In *European conference on machine learning*, pp. 83–102. Springer, 1994.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 93–104, 2000.

Horst Bunke. On a relation between graph edit distance and maximum common subgraph. *Pattern recognition letters*, 18(8):689–694, 1997.

Jinyu Cai and Jicong Fan. Perturbation learning based anomaly detection. *Advances in Neural Information Processing Systems*, 35:14317–14330, 2022.

Guilherme O Campos, Arthur Zimek, Jörg Sander, Ricardo JGB Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E Houle. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data mining and knowledge discovery*, 30 (4):891–927, 2016.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Wei Dai and Jicong Fan. AutoUAD: Hyper-parameter optimization for unsupervised anomaly detection. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ErQPdaD5wJ.

Lucas Deecke, Robert Vandermeulen, Lukas Ruff, Stephan Mandt, and Marius Kloft. Image anomaly detection with generative adversarial networks. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, Proceedings, Part I 18*, pp. 3–17. Springer, 2019.

Harris Drucker, Christopher J Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. *Advances in neural information processing systems*, 9, 1996.

Sunny Duan, Loic Matthey, Andre Saraiva, Nicholas Watters, Christopher P Burgess, Alexander Lerchner, and Irina Higgins. Unsupervised model selection for variational disentangled representation learning. *arXiv preprint arXiv:1905.12614*, 2019.

Jicong Fan. Graph minimum factorization distance and its application to large-scale graph data clustering. In *Forty-second International Conference on Machine Learning*, 2025.

Dazhi Fu, Zhao Zhang, and Jicong Fan. Dense projection for anomaly detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 8398–8408, 2024.

Michael R. Garey and David S. Johnson. Computers and intractability. a guide to the theory of np-completeness. wh freeman and company, san francisco1979, x+ 338 pp. *The Journal of Symbolic Logic*, 48(2):498–500, 1983.

Sachin Goyal, Aditi Raghunathan, Moksh Jain, Harsha Vardhan Simhadri, and Prateek Jain. Drocc: Deep robust one-class classification. In *International conference on machine learning*, pp. 3711–3721. PMLR, 2020.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The journal of machine learning research*, 13:723–773, 2012.

Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. Adbench: Anomaly detection benchmark. *Advances in neural information processing systems*, 35:32142–32159, 2022.

Xin He, Kaiyong Zhao, and Xiaowen Chu. Automl: A survey of the state-of-the-art. *Knowledge-based systems*, 212:106622, 2021.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

Wenpeng Hu, Mengyu Wang, Qi Qin, Jinwen Ma, and Bing Liu. Hrn: A holistic approach to one class learning. *Advances in neural information processing systems*, 33:19111–19124, 2020.

Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. *Automated machine learning: methods, systems, challenges*. Springer, 2019.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

Serdar Kadioglu, Yuri Malitsky, Meinolf Sellmann, Kevin Tierney, et al. Isac-instance-specific algorithm configuration. In *ECAI*, volume 215, pp. 751–756, 2010.

JooSeuk Kim and Clayton D Scott. Robust kernel density estimation. *The Journal of Machine Learning Research*, 13(1):2529–2565, 2012.

Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. Copod: copula-based outlier detection. In *2020 IEEE international conference on data mining (ICDM)*, pp. 1118–1123. IEEE, 2020.

Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and George H Chen. Ecod: Unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12181–12193, 2022.

Zinan Lin, Kiran Thekumparampil, Giulia Fanti, and Sewoong Oh. Infogan-cr and modelcentrality: Self-supervised model training and selection for disentangling gans. In *international conference on machine learning*, pp. 6127–6139. PMLR, 2020.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pp. 413–422. IEEE, 2008.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Martin Q Ma, Yue Zhao, Xiaorong Zhang, and Leman Akoglu. A large-scale study on unsupervised outlier model selection: Do internal strategies suffice? *arXiv preprint arXiv:2104.01422*, 2021.

Facundo Mémoli. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11(4):417–487, 2011.

Eduardo Fernandes Montesuma, Adel El Habazi, and Fred Ngole Mboula. Unsupervised anomaly detection through mass repulsing optimal transport. *arXiv preprint arXiv:2502.12793*, 2025.

Mladen Nikolić, Filip Marić, and Predrag Janičić. Simple algorithm portfolio for sat. *Artificial Intelligence Review*, 40(4):457–465, 2013.

Pramuditha Perera, Ramesh Nallapati, and Bing Xiang. Ocgan: One-class novelty detection using gans with constrained latent representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2898–2906, 2019.

Tomáš Pevnỳ. Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102(2):275–304, 2016.

Gabriel Peyré, Marco Cuturi, and Justin Solomon. Gromov-wasserstein averaging of kernel and distance matrices. In *International conference on machine learning*, pp. 2664–2672. PMLR, 2016.

Yuehan Qin, Yichi Zhang, Yi Nian, Xueying Ding, and Yue Zhao. Metaood: Automatic selection of ood detection models. *arXiv preprint arXiv:2410.03074*, 2024.

Chen Qiu, Timo Pfrommer, Marius Kloft, Stephan Mandt, and Maja Rudolph. Neural transformation learning for deep anomaly detection beyond images. In *International conference on machine learning*, pp. 8703–8714. PMLR, 2021.

Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 427–438, 2000.

John R Rice. The algorithm selection problem. In *Advances in computers*, volume 15, pp. 65–118. Elsevier, 1976.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pp. 4393–4402. PMLR, 2018.

Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021.

Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pp. 146–157. Springer, 2017.

Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12, 1999.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

Philipp Seeböck, Sebastian Waldstein, Sophie Klimscha, Bianca S Gerendas, René Donner, Thomas Schlegl, Ursula Schmidt-Erfurth, and Georg Langs. Identifying and categorizing anomalies in retinal imaging data. *arXiv preprint arXiv:1612.00686*, 2016.

Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. In *Proceedings of the IEEE foundations and new directions of data mining workshop*, pp. 172–179. IEEE Press Piscataway, NJ, USA, 2003.

Yan Sun and Jicong Fan. Mmd graph kernel: Effective metric learning for graphs via maximum mean discrepancy. In *International Conference on Learning Representations*, volume 2024, pp. 22274–22307, 2024.

Emmanouil Sylligardos, Paul Boniol, John Paparrizos, Panos Trahanias, and Themis Palpanas. Choose wisely: An extensive evaluation of model selection for anomaly detection in time series. *Proceedings of the VLDB Endowment*, 16(11):3418–3432, 2023.

Emmanouil Sylligardos, John Paparrizos, Themis Palpanas, Pierre Senellart, and Paul Boniol. Msad: A deep dive into model selection for time series anomaly detection. *The VLDB Journal*, 34(6): 1–25, 2025.

Jian Tang, Zhixiang Chen, Ada Wai-Chee Fu, and David W Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *Pacific-Asia conference on knowledge discovery and data mining*, pp. 535–548. Springer, 2002.

Joaquin Vanschoren. Meta-learning: A survey. *arXiv preprint arXiv:1810.03548*, 2018.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 1096–1103, 2008.

Christopher Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. *Advances in neural information processing systems*, 13, 2000.

Peter Wills and François G Meyer. Metrics for graph comparison: a practitioner's guide. *Plos one*, 15(2):e0228728, 2020.

David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 2002.

Feng Xiao, Ruoyu Sun, and Jicong Fan. Restricted generative projection for one-class classification and anomaly detection. *arXiv preprint arXiv:2307.04097*, 2023.

Feng Xiao, Jianfeng Zhou, Kunpeng Han, Haoyuan Hu, and Jicong Fan. Unsupervised anomaly detection using inverse generative adversarial networks. *Information Sciences*, 689:121435, 2025.

Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin Duke. Gromov-wasserstein learning for graph matching and node embedding. In *International conference on machine learning*, pp. 6932–6941. PMLR, 2019.

Hongteng Xu, Jiachang Liu, Dixin Luo, and Lawrence Carin. Representing graphs via gromov-wasserstein factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1): 999–1016, 2022.

Hongzuo Xu, Guansong Pang, Yijie Wang, and Yongjun Wang. Deep isolation forest for anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12591–12604, 2023.

Yunhe Zhang, Yan Sun, Jinyu Cai, and Jicong Fan. Deep orthogonal hypersphere compression for anomaly detection. *arXiv preprint arXiv:2302.06430*, 2023.

Zhengxin Zhang, Ziv Goldfeld, Youssef Mroueh, and Bharath K Sriperumbudur. Gromov–wasserstein distances: Entropic regularization, duality and sample complexity. *The Annals of Statistics*, 52(4): 1616–1645, 2024.

Yue Zhao and Leman Akoglu. Hpod: Hyperparameter optimization for unsupervised outlier detection. In *AutoML 2024 Methods Track*, 2024.

Yue Zhao, Ryan Rossi, and Leman Akoglu. Automatic unsupervised outlier model selection. *Advances in Neural Information Processing Systems*, 34:4489–4502, 2021.

Yue Zhao, Sean Zhang, and Leman Akoglu. Toward unsupervised outlier model selection. In *2022 IEEE international conference on data mining (ICDM)*, pp. 773–782. IEEE, 2022.

Xiaona Zhou, Constantin Brif, and Ismini Lourentzou. mtsbench: Benchmarking multivariate time series anomaly detection and model selection at scale. *arXiv preprint arXiv:2506.21550*, 2025.

Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*, 2018.

APPENDIX CONTENTS

A Related Work **15**

A.1 Model Selection in AutoML 15

A.2 Model Selection for Anomaly Detection 15

A.3 Model Selection based on Meta-Learning 15

B Experimental Coverage and Settings **16**

B.1 Testbeds 16

B.2 Model Set $\mathcal{M}$ 16

B.3 Baselines 17

B.4 Evaluation 17

C Selection of Graph Metrics **17**

C.1 Weisfeiler-Lehman (WL) Subtree Kernel 17

C.2 Minimum Factorization Distance (MFD) 18

C.3 Graph Measures with Multi-scale Gaussian Kernel 19

C.4 Comparison of different graph construction strategies 19

D Relationship Between Structural Similarity and Detector-ranking Similarity **20**

E Time Complexity Analysis of Constructing Meta-Feature **22**

F A Lightweight Variant of Constructing Meta-Features **23**

G Selection of Meta-Learner **23**

H Alternative Graph Measurements **25**

I Limitation **26**

J More Results and Overall Performance Selection of All Baselines on Two Testbeds **26**

A Related Work

A.1 Model Selection in AutoML

Model selection is a central problem in automated machine learning (AutoML) (Bahri et al., 2022), where the goal is to automatically identify suitable learning algorithms and hyperparameter configurations for a given task (Hutter et al., 2019; He et al., 2021). In supervised learning, since labeled validation data are available, candidate models can be directly compared according to task-specific performance metrics, making supervised model selection relatively well-defined. However, model selection becomes challenging in unsupervised learning. Unlike supervised tasks, unsupervised tasks usually lack labeled validation data, making direct performance evaluation impossible. This issue arises in clustering, dimensionality reduction, and anomaly detection, where internal criteria (Duan et al., 2019; Lin et al., 2020) are often used as proxies for external performance. Yet such criteria may not always correlate well with the desired real objective. Therefore, unsupervised model selection requires either elaborately designed surrogate objectives or task similarity measures that can transfer reliable knowledge from historical tasks.

A.2 Model Selection for Anomaly Detection

Automatic model selection for anomaly detection has recently attracted increasing attention. The difficulty mainly comes from the unsupervised nature of AD. Notably, in anomaly detection, this challenge is even more severe because the data are often highly imbalanced, anomalies are rare and heterogeneous, and the structural properties that determine detector performance can be subtle and dataset-specific. Existing model selection methods can be broadly categorized into meta-learning-based methods and hyperparameter optimization-based methods. The former addresses model selection by learning from historical anomaly detection tasks. MetaOD (Zhao et al., 2021) is one of the representative studies for unsupervised outlier model selection. It constructs meta-features from statistical characteristics and landmark features, and then predicts the performance rank of candidate models on a new unlabeled dataset. More recently, MetaOOD (Qin et al., 2024) studies automatic selection of out-of-distribution detection models and leverages language model-based embeddings of dataset and model descriptions to quantify task similarities. Related studies have also been explored in specific domains, such as multivariate time-series anomaly detection model selection (Sylligardos et al., 2023; 2025; Zhou et al., 2025; Abdelmaksoud et al., 2026). The latter aims to select suitable hyperparameters and models directly on the new unlabeled dataset. HPOD (Zhao & Akoglu, 2024) studies hyperparameter optimization for unsupervised outlier detection and transfers historical hyperparameter performance information to evaluate promising configurations without labels. AutoUAD (Dai & Fan, 2025) further considers automated hyperparameter optimization for unsupervised anomaly detection and designs unsupervised objectives to guide the search process.

A.3 Model Selection based on Meta-Learning

Classical offline-online meta-learning (Vanschoren, 2018) is a standard learning framework for unsupervised model selection. Rice (Rice, 1976) formalized algorithm selection as the task of learning a mapping from problem characteristics to suitable algorithms based on their expected performance. Brazdil et al. (Brazdil et al., 1994) subsequently operationalized this idea for machine learning by characterizing datasets using statistical and information-theoretic meta-features and by learning from the observed performance of candidate algorithms. Many later task-level, meta-feature-based recommendation methods adhere to the same general template:

$$D_i \xrightarrow{\phi} \mathbf{z}_i, \qquad (\mathbf{z}_i, \mathbf{p}_i)_{i=1}^{N} \overset{\text{meta-learning}}{\longrightarrow} S, \qquad S(\mathbf{z}_{\text{new}}) \longrightarrow \widehat{\mathbf{p}}_{\text{new}}, \tag{9}$$

where D_i denotes a task, ϕ its characterization function, $\mathbf{z}_i$ its meta-feature, $\mathbf{p}_i$ its candidate-performance vector, and S the learned recommendation function.

MetaOD (Zhao et al., 2021), MetaOOD (Qin et al., 2024), and our method share this general template. Their principal difference lies in the construction of ϕ and the training of S. MetaOD uses statistical characteristics and landmark features (detectors' outputs) to construct meta-features. MetaOOD uses language-model embeddings of dataset and detector descriptions that introduces a semantic, text-based task for OOD model selection. Our method presents a unified graph-induced task representation

and a feasible multi-scale WL/MFD solution that captures local neighborhood structure and global distributional geometry across heterogeneous feature spaces and modalities.

B Experimental Coverage and Settings

B.1 Testbeds

Our evaluation leverages the benchmark datasets derived from ADBench (Han et al., 2022), covering both tabular and image domains. We construct two separate testbeds: (*i*) **Tabular Testbed**, comprising 47 anomaly detection tasks from diverse tabular datasets; and (*ii*) **Image Testbed**, comprising 61 detection tasks derived from widely used image anomaly detection datasets. This separation allows us to analyze method performance in domain-specific contexts while maintaining consistent evaluation protocols across modalities.

Tabular Testbed. All 47 datasets from ADBench are utilized in this setting. These datasets vary in feature dimensions and anomaly ratios and span multiple, entirely distinct domains, such as healthcare, finance, botany, and astronautics, reflecting the diversity and significant heterogeneity of this benchmark. This makes it a highly suitable testbed for evaluating the effectiveness and robustness of model selection methods.

Image Testbed. The 61 image anomaly detection tasks are derived from five image datasets in ADBench, including CIFAR-10, FashionMNIST, MNIST-C, SVHN, and MVTec-AD. Specifically, CIFAR-10, FashionMNIST, and SVHN each contribute 10 detection tasks, MNIST-C includes 16 tasks, and MVTec-AD contains 15 tasks. Pretrained ResNet18 embeddings for all these datasets are provided in the official code repository [3], and we directly utilize these extracted image embeddings in our experiments.

B.2 Model Set $\mathcal{M}$

We construct a comprehensive model set $\mathcal{M}$ for anomaly detection by pairing a diverse collection of algorithms with their respective hyperparameter configurations (HPCs). Specifically, we include 12 representative AD algorithms encompassing both classical and deep methods, resulting in a total of 358 unique $(\text{Algorithm}, \text{HPC})$ pairs.This large and heterogeneous model set ensures a realistic evaluation of model selection techniques across varying detection scenarios. The detailed (Algorithm, HPC) pairs are summarized in Table 5.

Table 5: The summarized (Algorithm, HPC) pairs.

Algorithm	**HPC (Hyperparameter Configuration)**
OCSVM (Schölkopf et al., 1999)	kernel=rbf; gamma $\in$ { $0.001, 0.01, 0.1, 1.0, 5.010.0, 100.0$}; nu $\in \{0.1, 0.25, 0.5\}$
IForest (Liu et al., 2008)	n_estimators $\in \{10, 20, 50, 100, 150, 200\}$; max_features $\in \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$
KNN (Ramaswamy et al., 2000)	distance=euclidean; method $\in$ {largest, mean, median} n_neighbors $\in \{3, 5, 10, 15, 20, 30, 50, 70, 100\}$;
LOF (Breunig et al., 2000)	metric $=$ {cityblock, euclidean}; n_neighbors $\in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$
KDE (Kim & Scott, 2012)	bandwidth $\in$ {scott, silverman, $1.0, 10.0, 100.0$}; metric $=$ {cityblock, euclidean}
PCA (Shyu et al., 2003)	n_components $\in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$
ECOD (Li et al., 2022)	N/A (ECOD is a hyperparameter-free outlier detection algorithm.)
COF (Tang et al., 2002)	n_neighbors $\in \{3, 5, 10, 15, 20, 25\}$
LODA (Pevnỳ, 2016)	n_bins $\in \{$auto, $5, 10, 20, 30, 50\}$; n_random_cuts $\in \{5, 10, 15, 20, 25, 30\}$
AE (Hinton & Salakhutdinov, 2006)	lr $\in \{1e^{-4}, 5e^{-4}, 1e^{-3}, 5e^{-3}\}$; optimizer_name $=$ adam; hidden_activation_name $\in$ {tanh, relu, leaky_relu, sigmoid}; hidden_neuron_list $\in \{[256, 32, 32, 256], [128, 16, 16, 128]\}$
Deep SVDD (Ruff et al., 2018)	lr $\in \{1e^{-4}, 5e^{-4}, 1e^{-3}, 5e^{-3}\}$; optimizer $=$ adam; hidden_activation $\in$ {tanh, relu, leaky_relu, sigmoid}; hidden_neuron $\in \{[128, 16, 8], [256, 32, 16]\}$
DPAD (Fu et al., 2024)	lr $\in \{1e^{-4}, 1e^{-3}\}$; optimizer $=$ adam; gamma $\in \{0.1, 0.01\}$; hidden_activation $\in$ {tanh, relu, leaky_relu, sigmoid}; lamb $\in \{0.1, 0.01\}$; hidden_neuron $\in \{[128, 16, 8], [256, 32, 16]\}$; $k \in \{5, 10\}$

[3] https://github.com/Minqi824/ADBench

B.3 BASELINES

To assess the effectiveness of our proposed model selection framework, we compare against eight strong baselines selected according to four considerations:

1. **Common-used and latest AD algorithms.** Considering the two distinct domains, we include both classical and deep methods: KNN (Ramaswamy et al., 2000), DeepSVDD (Ruff et al., 2018), COPOD (Li et al., 2020) and MROT (Montesuma et al., 2025). Prior studies (Han et al., 2022; Bouman et al., 2024) consistently demonstrate that KNN achieves strong performance on tabular datasets. DeepSVDD Ruff et al. (2018) is classical anomaly detection method based on hypersphere learning. COPOD (Li et al., 2020) is a hyperparameter-free, highly interpretable outlier detection algorithm based on empirical copula models. MROT (Montesuma et al., 2025) is a latest unsupervised anomaly detection method that augments optimal transport with a mass-repulsion term, using the resulting transport cost as an anomaly score that is higher for low-density samples. When applying these four AD algorithms as model selection baselines, we adopt a heuristic strategy: for each leave-one-out experiment, the hyperparameter configuration with the best historical average performance on the remaining detection tasks is selected.
2. **General meta-learners.** We incorporate three general-purpose meta-learning techniques for unsupervised model selection: Model Centrality (MC) (Ma et al., 2021), ARGOSMART (AS) (Nikolić et al., 2013), and Instance-Specific Algorithm Configuration (ISAC) (Kadioglu et al., 2010).
3. **Tailored meta-learning-based AD selection.** We include methods specifically designed for anomaly detection, namely MetaOD (Zhao et al., 2021) and MetaOOD (Qin et al., 2024), which construct meta-features from historical AD tasks to guide model selection.
4. **Hyperparameter optimization-based selection.** AutoUAD (Dai & Fan, 2025) represents the latest unsupervised HO-based model selection approach, leveraging internal criteria to evaluate candidate configurations without relying on labels.

B.4 EVALUATION

We conduct comprehensive *leave-one-out* experiments on both testbeds to assess the model selection performance. For each left-out task, we evaluate the selected model based on the held-out tasks and report two primary metrics: average Area Under the Receiver Operating Characteristic curve (AUROC %) and Area Under the Precision-Recall Curve (AUPRC %) across the testbed. To statistically compare the proposed method against baselines, we perform pairwise *t*-tests across datasets, adopting a significance threshold of $p < 0.05$. All experiments are conducted on a workstation equipped with an Intel(R) Xeon(R) CPU E5-2678 v3 @ 2.50GHz and 4×NVIDIA GeForce RTX 3090 GPUs, using Python 3.10 and CUDA 12.1.

C SELECTION OF GRAPH METRICS

C.1 WEISFEILER-LEHMAN (WL) SUBTREE KERNEL

Due to the ubiquitous presence of graph-structured data in real-world applications, such as social networks and molecular structures, and the fundamental importance of the graph comparison problem, numerous methods have been developed over the past decades (Bunke, 1997; Shervashidze et al., 2011; Mémoli, 2011; Fan, 2025). Despite these advances, most algorithms incur cubic or higher computational complexity, which limits their applicability to large-scale datasets. To balance computational efficiency with task relevance for anomaly detection, we adopt the **Weisfeiler-Lehman (WL) subtree kernel** (Shervashidze et al., 2011) to compare the local neighborhood structures of sample-wise similarity graphs across datasets.

Definition 2 (Weisfeiler-Lehman Subtree Kernel (Shervashidze et al., 2011))**.** *Let $\mathbf{G}_1$ and $\mathbf{G}_2$ be two graphs. For each iteration $i = 0, \ldots, h$ of the Weisfeiler-Lehman algorithm, let $\Sigma_i \subseteq \Sigma$ denote the set of node labels appearing in $\mathbf{G}_1$ or $\mathbf{G}_2$ at least once, and let Σ_0 be the set of original node labels of $\mathbf{G}_1$. Assume that all Σ_i are pairwise disjoint and ordered, i.e., $\Sigma_i = \{\sigma_{i1}, \ldots, \sigma_{i|\Sigma_i|}\}$.*

Define the mapping

$$c_i : \mathbf{G}_1 \times \Sigma_i \to \mathbb{N},$$

such that $c_i(\mathbf{G}_1, \sigma_{ij})$ counts the occurrences of label σ_{ij} in $\mathbf{G}_1$ at iteration i. Similarly, define $c_i(\mathbf{G}_2, \sigma_{ij})$ for $\mathbf{G}_2$.

The WL subtree kernel between $\mathbf{G}_1$ and $\mathbf{G}_2$ with h iterations is defined as

$$k(\mathbf{G}_1, \mathbf{G}_2) = \langle \phi(\mathbf{G}_1), \phi(\mathbf{G}_2) \rangle, \tag{10}$$

where the feature maps are

$$\phi(\mathbf{G}_1) = \big(c_0(\mathbf{G}_1, \sigma_{01}), \ldots, c_0(\mathbf{G}_1, \sigma_{0|\Sigma_0|}), \ldots, c_h(\mathbf{G}_1, \sigma_{h1}), \ldots, c_h(\mathbf{G}_1, \sigma_{h|\Sigma_h|})\big), \tag{11}$$

$$\phi(\mathbf{G}_2) = \big(c_0(\mathbf{G}_2, \sigma_{01}), \ldots, c_0(\mathbf{G}_2, \sigma_{0|\Sigma_0|}), \ldots, c_h(\mathbf{G}_2, \sigma_{h1}), \ldots, c_h(\mathbf{G}_2, \sigma_{h|\Sigma_h|})\big). \tag{12}$$

This kernel effectively counts the number of shared subtrees between the two graphs. Its computational complexity is $\mathcal{O}(hl)$, where h is the subtree height and l is the number of edges in the graph. In all experiments, we set $h = 5$.

Handling Weighted Graphs and Node Labels. The standard WL kernel requires discrete node labels. To apply it to weighted similarity graphs from $\mathcal{G}$, we first sparsify the adjacency matrices:

1. Compute the average similarity $\bar{s} = \mathbb{E}_{\mathbf{G}_i^{(\sigma)} \in \mathcal{G}^{(\sigma)}} \mathbb{E}_{j \neq k} [\mathbf{G}_i^{(\sigma)}]_{jk}$ for a given kernel (e.g., GK_{γ_1}) across all graphs.
2. Binarize each similarity matrix using $\bar{s}$ as a threshold: entries below $\bar{s}$ are set to 0, others to 1, yielding a sparse $\{0, 1\}$ adjacency graph.
3. Assign a constant label (0) to all nodes. In the first WL iteration, nodes are separated solely by their degree, since all neighbor label multisets are initially identical.

This ensures that iteration 0 compares degree distributions, while subsequent iterations progressively capture deeper neighborhood structures, allowing the kernel to reflect local connectivity patterns that are critical for anomaly detection. Note that the $\bar{s}$ is a fixed scalar computed strictly over the historical graphs during meta-training and it does not need to be recomputed during the online model selection process.

C.2 Minimum Factorization Distance (MFD)

While local structural patterns captured by the WL kernel are critical, global geometric properties of datasets also play a key role in determining the most suitable anomaly detector. A natural candidate for capturing such global relations is the Gromov-Wasserstein distance (GWD) (Mémoli, 2011), which has demonstrated strong performance in graph comparison tasks (Xu et al., 2019; 2022). However, the computational cost of GWD, $\mathcal{O}(n^4)$ for the exact variant or $\mathcal{O}(n^3)$ for the entropic-regularized version (Peyré et al., 2016), renders it infeasible for large-scale graph comparisons.

To address this limitation, we employ the recently proposed **Minimum Factorization Distance (MFD)** (Fan, 2025), which measures global distributional discrepancies between graphs while being significantly more computationally efficient than GWD. MFD aligns graphs via an optimal linear transformation and compares their embedded feature distributions, making it well-suited for anomaly detection model selection where both local and global data structures influence detector performance.

Definition 3 (Minimum Factorization Distance (MFD) (Fan, 2025))**.** *Let $\mathcal{A}_i^{\phi} = \mathbf{\Phi}_i^{\top} \mathbf{\Phi}_i, i = 1, 2$ and $\mathbf{\Phi}_i = \phi(\mathbf{X}_i) = [\phi_1^{(i)}, \ldots, \phi_{n_i}^{(i)}]$, denote the feature matrices associated with graphs $\mathbf{G}_1$ and $\mathbf{G}_2$. The Minimum Factorization Distance between $\mathbf{G}_1$ and $\mathbf{G}_2$ is defined as*

$$\mathrm{MFD}(\mathbf{G}_1, \mathbf{G}_2) = \min_{\mathbf{R}_{12} \in \mathcal{R}} \left\| \frac{1}{n_1} \sum_{j=1}^{n_1} \zeta(\phi_j^{(1)}) - \frac{1}{n_2} \sum_{j=1}^{n_2} \zeta(R_{12} \phi_j^{(2)}) \right\|, \tag{13}$$

where $\mathbf{R}_{12}$ is linear alignment mapping (need to be optimized), n_1 and n_2 denote the number of nodes in $\mathbf{G}_1$ and $\mathbf{G}_2$, respectively, and ζ is the feature map associated with a nonlinear kernel function. The $\mathcal{A}_i^{\phi}$ is computed by

$$\mathcal{A}_i^{\phi} = \sum_{j=1}^{n_i} |\lambda_j^{(i)}| \mathbf{u}_j^{(i)} {\mathbf{u}_j^{(i)}}^{\top}, \quad i = 1, 2, \tag{14}$$

where $\lambda_j^{(i)}$ and $\mathbf{u}_j^{(i)}$ are the j-th eigenvalue and eigenvector of $\mathbf{G}_i$.

Although MFD does not admit a closed-form solution, an iterative optimization procedure is proposed in (Fan, 2025) to approximate it efficiently. The computational complexity of this procedure is $\mathcal{O}(n^2 \log(d) + d^2 n + T(d^3 + dn^2))$, where d is the embedding dimension and T is the number of iterations. In our experiments, we set $d = 256$ and $T = 150$, which provides a practical trade-off between computational cost and approximation accuracy.

C.3 Graph Measures with Multi-scale Gaussian Kernel

We adopt a multi-scale strategy by varying the Gaussian kernel parameter $\gamma \in \{0.1, 0.2, 0.5, 1.0, 2.0, 5.0\}$, generating multiple similarity graphs for each dataset. To evaluate the effect of this multi-scale design, we conduct *leave-one-out* experiments on the two testbeds, comparing the performance obtained using a single default γ versus the full predefined range. The results are reported in Table 6 and Table 7. In this study, $\mathbf{v}_{\text{WL}}$ and $\mathbf{v}_{\text{MFD}}$ indicate meta-features constructed exclusively using the WL kernel or MFD, respectively, whereas $\mathbf{v}$ denotes the full graph-induced meta-feature as defined in equation 7, combining both WL and MFD across multiple scales.

Table 6: Graph kernel ablation results on **Tabular TestBed** with the average AUROC (%) and AUPRC (%) across 47 tasks.

Meta-feature	default ($\gamma = 1$)		$\gamma \in \{0.1, 0.2, 0.5, 1.0, 2.0, 5.0\}$	
	AUROC (%)	AUPRC (%)	AUROC (%)	AUPRC (%)
$\mathbf{v}_{\text{WL}}$	79.46	59.76	80.85	61.63
$\mathbf{v}_{\text{MFD}}$	80.53	60.36	80.25	61.19
$\mathbf{v}$	80.65	61.46	81.06	64.44

Table 7: Graph kernel ablation results on **Image TestBed** with the average AUROC (%) and AUPRC (%) across 61 tasks.

Meta-feature	default ($\gamma = 1$)		$\gamma \in \{0.1, 0.2, 0.5, 1.0, 2.0, 5.0\}$	
	AUROC (%)	AUPRC (%)	AUROC (%)	AUPRC (%)
$\mathbf{v}_{\text{WL}}$	80.15	59.32	80.20	58.60
$\mathbf{v}_{\text{MFD}}$	79.83	59.17	80.11	59.46
$\mathbf{v}$	80.28	59.72	80.56	60.34

C.4 Comparison of different graph construction strategies

The WL subtree kernel requires a discrete graph representation, and thus the weighted sample-wise similarity graph needs to be converted into a binary adjacency graph before WL refinement. Our default implementation uses a historical-expectation threshold $\bar{s} = \mathbb{E}_{\mathbf{G}_i^{(\sigma)} \in \mathcal{G}^{(\sigma)}} \mathbb{E}_{j \neq k}[\mathbf{G}_i^{(\sigma)}]_{jk}$, where the binarization threshold is estimated as the average pairwise similarity over historical graphs under the corresponding Gaussian kernel. This provides a data-driven reference shared across tasks and avoids introducing an additional graph-construction hyperparameter. Nevertheless, other graph construction strategies, such as quantile-based thresholding or neighborhood graphs, are also feasible. We therefore investigate the sensitivity of GOOD to this design choice.

Alternative graph construction strategies. We compare four types of graph construction at the representative Gaussian kernel scale $\gamma = 1.0$:

- (i) the historical-expectation threshold $\bar{s}$ used by GOOD;
- (ii) historical similarity quantiles with $q \in \{0.25, 0.50, 0.75\}$;
- (iii) kNN graphs;
- (iv) mutual-kNN graphs with $k \in \{10, 20, 30\}$.

Table 8: Comparison of different graph construction strategies with $\gamma = 1$.

Graph ($\gamma = 1$)	**Threshold**	**Median Sparsity**	**Mean Sparsity**	**AUROC**	**AUPRC**
Current	0.510	0.536	0.540	80.90	59.20
Quantile-0.25	0.319	0.254	0.262	81.80	54.34
Quantile-0.50	0.495	0.519	0.511	80.93	56.88
Quantile-0.75	0.697	0.778	0.765	83.00	57.37
kNN-10	N/A	0.994	0.974	83.30	56.60
kNN-20	N/A	0.989	0.951	83.27	57.42
kNN-30	N/A	0.984	0.928	82.06	56.15
Mutual-kNN-10	N/A	0.997	0.990	83.58	57.67
Mutual-kNN-20	N/A	0.995	0.979	83.26	58.35
Mutual-kNN-30	N/A	0.993	0.968	83.02	57.71

For the latter two strategies, k controls the number of local neighbors retained for each sample, while mutual-kNN retains an edge only when the two samples are mutually among each other's k nearest neighbors. We additionally report the median and mean graph sparsity to characterize the graph structures induced by different construction rules.

Sensitivity analysis. Table 8 shows that GOOD remains effective under substantially different graph construction rules. The resulting graphs span a wide range of sparsity levels, from a mean sparsity of 0.262 for Quantile-0.25 to 0.990 for Mutual-kNN-10, yet all variants retain competitive model-selection performance. This suggests that the effectiveness of the WL-based representation is not restricted to a particular binarization threshold or graph density.

The choice of graph construction nevertheless affects AUROC and AUPRC differently. Neighborhood-based graphs generally achieve higher AUROC, with Mutual-kNN-10 reaching the highest AUROC of 83.5%, compared with 80.9% using the default construction. In contrast, historical-expectation threshold achieves the highest AUPRC of 59.2% among the evaluated strategies. The quantile-based variants exhibit a similar trade-off: changing the threshold can improve AUROC, but does not consistently improve AUPRC. These results indicate that there is no single graph construction rule that dominates across both evaluation metrics. Overall, these experiments show that quantile, kNN, and mutual-kNN constructions are viable alternatives and can yield competitive or even higher performance for individual metrics. We use the historical-expectation threshold as the default because it is parameter-free at test time, maintains a consistent similarity-based edge interpretation across tasks, and provides balanced performance across AUROC and AUPRC.

D Relationship Between Structural Similarity and Detector-ranking Similarity

A central motivation of GOOD is that datasets with similar structural characteristics are more likely to exhibit similar relative preferences over anomaly detectors. In this section, we quantitatively investigate the relationship between them.

Structural motivation. The proposed WL and MFD measurements characterize complementary aspects of the sample-wise similarity graphs. The WL subtree kernel iteratively aggregates neighborhood labels and therefore captures similarities in local connectivity and multi-hop neighborhood organization. Such structures can reflect anomaly-relevant properties such as local density variation, cluster boundaries, and local manifold organization. MFD, in contrast, compares factorized graph representations and characterizes discrepancies in global graph structures, which can reflect large-scale geometric properties such as cluster separation, support, and dispersion. These structural properties are closely related to the inductive biases of different anomaly detectors. For example, neighborhood- and density-based detectors are directly affected by local connectivity and density variation, whereas detectors relying on global distributional assumptions can respond differently to changes in the overall geometry of the data.

Therefore, two datasets that are structurally closer under these graph measurements may tend to exhibit more similar relative performance across candidate anomaly detectors. We emphasize that this

is an *inductive relationship* rather than a universal guarantee, as it depends on whether the constructed similarity graphs preserve structures relevant to the underlying anomaly mechanisms.

Quantitative verification. We directly examine the above relationship on the 47 tabular tasks. Recall that the historical datasets are denoted by $\mathcal{D}_{\text{his}} = \{(\mathbf{X}_i, \mathbf{y}_i)\}_{i=1}^{n}$ and the candidate model set is $\mathcal{M} = \{M_j\}_{j=1}^{m}$. The candidate models on the historical datasets produces the performance matrix $\mathbf{P} \in \mathbb{R}^{n\times m}$, where p_{ij} denotes the performance of model M_j on dataset $\mathbf{X}_i$. We perform the following three steps.

- **Step 1: Construct detector-ranking similarity.** For each pair of historical datasets $(\mathbf{X}_i, \mathbf{X}_j)$, we compare their corresponding performance vectors $\mathbf{P}_{i,:}$ and $\mathbf{P}_{j,:}$. Specifically, their detector-ranking similarity is defined as

$$C_{i,j} = \rho_{\text{Spearman}}\left(\mathbf{P}_{i,:}, \mathbf{P}_{j,:}\right), \tag{15}$$

  where $\rho_{\text{Spearman}}(\cdot,\cdot)$ denotes Spearman rank correlation. A larger $C_{i,j}$ indicates that the m candidate models have more similar relative rankings on datasets $\mathbf{X}_i$ and $\mathbf{X}_j$. We independently construct C using the AUROC-based and AUPRC-based performance matrices, respectively.
- **Step 2: Construct graph-based dataset similarity.** As described in Section 3.3, for each Gaussian kernel scale $\gamma \in \{0.1, 0.2, 0.5, 1.0, 2.0, 5.0\}$, we construct the sample-wise similarity graph $\mathbf{G}_i^{(\text{GK}_\gamma)}$ for every historical dataset $\mathbf{X}_i$.

  Pairwise graph comparison then produces the matrices $\mathbf{F}_{\text{WL}}^{(\text{GK}_\gamma)}$ and $\mathbf{F}_{\text{MFD}}^{(\text{GK}_\gamma)} \in \mathbb{R}^{n\times n}$, whose elements are

$$\left[\mathbf{F}_{\text{WL}}^{(\text{GK}_\gamma)}\right]_{i,j} = \text{WL}\left(\mathbf{G}_i^{(\text{GK}_\gamma)}, \mathbf{G}_j^{(\text{GK}_\gamma)}\right), \tag{16}$$

  and

$$\left[\mathbf{F}_{\text{MFD}}^{(\text{GK}_\gamma)}\right]_{i,j} = \text{MFD}\left(\mathbf{G}_i^{(\text{GK}_\gamma)}, \mathbf{G}_j^{(\text{GK}_\gamma)}\right). \tag{17}$$

  Since WL is a similarity measure, a larger $\left[\mathbf{F}_{\text{WL}}^{(\text{GK}_\gamma)}\right]_{i,j}$ indicates greater structural similarity. In contrast, MFD is a dissimilarity measure, for which a smaller value indicates greater similarity. Therefore, for the following correlation analysis, we use $-\left[\mathbf{F}_{\text{MFD}}^{(\text{GK}_\gamma)}\right]_{i,j}$ such that larger values consistently represent greater dataset similarity.
- **Step 3: Measure the alignment between structural similarity and detector-ranking similarity.** Both C and each graph-measurement matrix contain pairwise relationships among the same n historical datasets. Since these matrices are symmetric and their diagonal entries correspond to self-comparisons, we retain only the upper-triangular entries $i < j$. For example, for WL we obtain two vectors

$$\mathbf{c} = \{C_{i,j} \mid 1 \le i < j \le n\}, \tag{18}$$

  and

$$\mathbf{f}_{\text{WL}}^{(\text{GK}_\gamma)} = \left\{\left[\mathbf{F}_{\text{WL}}^{(\text{GK}_\gamma)}\right]_{i,j} \mid 1 \le i < j \le n\right\}. \tag{19}$$

  With $n = 47$ tabular datasets, each vector therefore contains $\binom{47}{2} = 1,081$ pairwise dataset comparisons. We then compute

$$\text{Corr}\left(\mathbf{f}_{\text{WL}}^{(\text{GK}_\gamma)}, \mathbf{c}\right) \tag{20}$$

  using both Pearson and Spearman rank correlation. The same procedure is applied to $-\mathbf{f}_{\text{MFD}}^{(\text{GK}_\gamma)}$. Pearson correlation measures the linear association between the magnitude of structural similarity and detector-ranking similarity, whereas Spearman correlation evaluates whether datasets ordered as increasingly similar by a graph measurement also tend to exhibit increasingly similar detector rankings.

  As a reference, we construct dataset similarities from the original MetaOD meta-features using cosine similarity. For every pair $(\mathbf{X}_i, \mathbf{X}_j)$, we compute the cosine similarity between their MetaOD meta-features, forming an $n \times n$ dataset-similarity matrix. We extract its same $1,081$ upper-triangular entries and compute Pearson and Spearman rank correlations with $\mathbf{c}$ under exactly the same protocol.

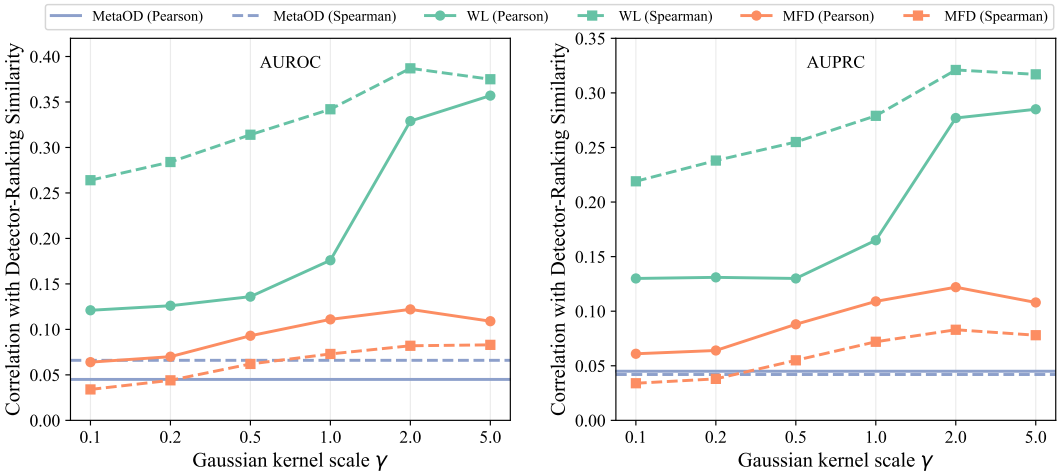


Figure 5: Correlation between dataset similarity and detector-ranking similarity on Tabular-testbed.

We visualize the comparison in Figure 5. For AUROC, the maximum Pearson and Spearman correlations both from WL reach 0.357 and 0.387, respectively, compared with 0.045 and 0.066 using the MetaOD representation. For AUPRC, the corresponding maximum correlations are 0.285 and 0.321, compared with 0.045 and 0.042 for MetaOD. Moreover, the alignment of WL with detector rankings generally becomes stronger as γ increases over the evaluated range, indicating that the structural resolution induced by graph construction influences the task relationships captured by WL.

MFD exhibits relatively weaker relationship with detector-ranking similarity than WL, but its correlations are mostly stronger than those obtained from the MetaOD representation. This observation is consistent with the complementary roles of the two measurements: WL directly characterizes local neighborhood organization, whereas MFD provides additional information about global structural discrepancies. Importantly, the relatively weak marginal correlation of MFD does not imply that it is redundant. As shown by the ablation experiments in Table 6,7, combining v_{WL} and v_{MFD} improves downstream model-selection performance over v_{WL} alone, particularly in terms of AUPRC.

This analysis provides direct quantitative evidence for the inductive motivation underlying GOOD. Across historical tasks, datasets that are closer under the proposed graph-induced structural measurements tend, on average, to exhibit more similar detector rankings, with the relationship being particularly clear for WL. The substantially stronger alignment of WL than the statistical meta-features used by MetaOD further supports the use of structural relationships for constructing task representations.

E Time Complexity Analysis of Constructing Meta-Feature

The construction of graph-induced meta-features during online model selection involves three primary steps, each contributing to the overall computational cost:

1. **Sample-wise similarity graph construction:** For a new dataset $\mathbf{X}_{\text{new}}$ with n_t samples, we first compute the pairwise similarity matrix to construct a weighted graph. This step requires evaluating the similarity measure σ for all pairs of samples, resulting in a time complexity of $\mathcal{O}(n_t^2)$.
2. **Computation of the WL kernel:** Let l denote the number of edges in the constructed similarity graph and h the number of WL iterations (subtree height). Computing the WL subtree kernel between the new graph and each historical graph has a time complexity of $\mathcal{O}(hl)$ reflecting the cost of iteratively aggregating neighborhood labels.
3. **Computation of the Minimum Factorization Distance (MFD):** Let d be the feature embedding dimension and T the number of iterative optimization steps for MFD. Computing

MFD between the new graph and each historical graph requires $\mathcal{O}\big(n_t^2 \log(d) + d^2 n_t + T(d^3 + dn_t^2)\big)$ operations, as detailed in (Fan, 2025).

Considering that both the WL kernel and MFD need to be computed between the new dataset and all n historical graphs $\mathcal{G}^{(\sigma)}$ from meta-training, the overall time complexity of constructing the meta-feature for online model selection is

$$\mathcal{O}\Big(n_t^2 + n_t^2 \log(d) + d^2 n_t + n\big(hl + T(d^3 + dn_t^2)\big)\Big), \tag{21}$$

where the first term corresponds to constructing the similarity graph of the new dataset, and the remaining terms account for pairwise comparison with historical graphs.

F A Lightweight Variant of Constructing Meta-Features

While the proposed graph-induced meta-feature significantly reduces online selection cost (See Figure 1) compared with hyperparameter optimization-based approaches, the construction of sample-wise similarity graphs still incurs a quadratic time complexity $\mathcal{O}(n_t^2)$ for a new dataset $\mathbf{X}_{\text{new}}$ with n_t samples. Moreover, computing the Minimum Factorization Distance (MFD) remains computationally non-trivial when the embedding dimension d is large, which can limit scalability for large datasets.

To address these limitations, we propose a **lightweight variant** of meta-feature construction that prioritizes computational efficiency while preserving task-relevant information. The two key modifications are:

1. **Nyström approximation for graph construction:** Instead of computing the full $n_t \times n_t$ similarity matrix, we sample $m_t \ll n_t$ landmark points and approximate the full similarity graph using the Nyström method (Williams & Seeger, 2000). Specifically, let $\mathbf{X}_{\text{landmark}}^{(m)} \subset \mathbf{X}_{\text{new}}$ denote the sampled landmarks. The approximate similarity matrix is computed as
$$\mathbf{G}_{\text{new}}^{(\sigma)} \approx \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{\top},$$
where $\mathbf{C} \in \mathbb{R}^{n_t \times m_t}$ is the cross-similarity matrix between all samples and landmarks, $\mathbf{W} \in \mathbb{R}^{m_t \times m_t}$ is the similarity among landmarks, and $\mathbf{W}^{\dagger}$ is the Moore-Penrose pseudo-inverse of $\mathbf{W}$. This reduces the graph construction cost from $\mathcal{O}(n_t^2)$ to $\mathcal{O}(n_t m_t^2)$, while preserving the global structure of the dataset.
2. **WL kernel-only meta-feature:** Motivated by the ablation study in Table 6 and Table 7, we observed that the meta-feature $\mathbf{v}_{\text{WL}}$ constructed solely from the Weisfeiler-Lehman (WL) kernel, achieves superior or comparable detection performance on both AUROC and AUPRC compared with the baselines (see Table 2). Consequently, to further reduce computational cost on large-scale datasets, we construct the meta-feature solely using the Weisfeiler-Lehman (WL) kernel with $\mathcal{O}(hl)$ time complexity.

By combining these two modifications, the lightweight meta-feature, denoted as $\mathbf{v}_{\text{Nys-WL}}$, can be computed efficiently. The overall time complexity for constructing $\mathbf{v}_{\text{Nys-WL}}$ for online model selection is

$$\mathcal{O}(n_t m_t^2 + nhl), \tag{22}$$

where the first term corresponds to the Nyström approximation of the similarity graph for the new dataset $\mathbf{X}_{\text{new}}$, and the second term corresponds to computing the WL kernel between the new graph and all n historical graphs.

Experimental Evaluation. To evaluate the effectiveness of $\mathbf{v}_{\text{Nys-WL}}$, we conduct *leave-one-out* experiments on the **Tabular Testbed** while varying the number of Nyström landmarks $m_t \in \{30, 50, 70, 100\}$. The corresponding results are reported in Table 9.

The results in Table 9 indicates that even with a small number of landmarks and using only the WL kernel, the lightweight meta-feature $\mathbf{v}_{\text{Nys-WL}}$ achieves competitive performance, while significantly reducing the computational cost compared to the full graph-induced meta-feature.

G Selection of Meta-Learner

During meta-training, the goal is to learn a mapping

$$S : \mathbb{R}^u \to \mathbb{R}^m,$$

Table 9: The performance comparison between $\mathbf{v}_{\text{Nys-WL}}$ and $\mathbf{v}_{\text{WL}}, \mathbf{v}_{\text{MFD}}, \mathbf{v}$ on **Tabular TestBed** with the average AUROC (%) and AUPRC (%) across 47 tasks.

Meta-feature	AUROC (%)	AUPRC (%)
$\mathbf{v}_{\text{WL}}$	80.85	61.63
$\mathbf{v}_{\text{MFD}}$	80.25	61.19
$\mathbf{v}$	81.06	64.44
$\mathbf{v}_{\text{Nys-WL}}$ $(m_t = 30)$	80.09	60.18
$\mathbf{v}_{\text{Nys-WL}}$ $(m_t = 50)$	80.15	61.70
$\mathbf{v}_{\text{Nys-WL}}$ $(m_t = 70)$	79.96	61.36
$\mathbf{v}_{\text{Nys-WL}}$ $(m_t = 100)$	80.50	62.00

where the input is the graph-induced meta-feature of a dataset and the output is the predicted performance vector over the candidate model set $\mathcal{M}$. In other words, the meta-learner estimates the expected performance of each $(\text{Algorithm}, \text{HPC})$ pair for a given anomaly detection task. Since this is a multi-output regression problem, several regression models can be considered, including Support Vector Regression (SVR) (Drucker et al., 1996), Random Forest (RF) (Breiman, 2001), XGBoost (Chen & Guestrin, 2016), and Neural Networks (NNs).

To comprehensively assess these meta-learners, we conduct *leave-one-out* experiments on the **Tabular Testbed**. For the neural network-based meta-learner, we further investigate four training objectives: mean absolute error (MAE), mean squared error (MSE), MAE+RMSE, and a ranking-oriented loss. Let $\mathbf{p}_i \in \mathbb{R}^m$ denote the ground-truth performance vector of the i-th training task, and let f_θ denote a neural network parameterized by θ. Given the meta-feature $\mathbf{v}_i \in \mathbb{R}^u$, the predicted performance vector is

$$\widehat{\mathbf{p}}_i = f_\theta(\mathbf{v}_i) \in \mathbb{R}^m.$$

The loss functions are defined as follows:

$$\begin{aligned}
L_{\text{MAE}} &= \frac{1}{mN_t} \sum_{i=1}^{N_t} \|\mathbf{p}_i - \widehat{\mathbf{p}}_i\|_1 , \\
L_{\text{MSE}} &= \frac{1}{mN_t} \sum_{i=1}^{N_t} \|\mathbf{p}_i - \widehat{\mathbf{p}}_i\|_2^2 , \\
L_{\text{MAE+RMSE}} &= L_{\text{MAE}} + \sqrt{L_{\text{MSE}}}, \\
L_{\text{Rank}} &= \frac{1}{N_t} \sum_{i=1}^{N_t} \log_2 \left(1 + \sum_{j=1}^{m} \text{sigmoid}\left(\tau \cdot (\widehat{p}_{ij} - \widehat{p}_{ij_i^*})\right) \right), \quad j_i^* = \arg\max_j \mathbf{p}_i,
\end{aligned} \tag{23}$$

where N_t is the number of training tasks and τ is a scaling constant. We set $\tau = 1000$ in all experiments.

The comparison results are reported in Table 10. Overall, neural network-based meta-learners achieve stronger performance than conventional regressors. Among all variants, NN trained with MSE achieves the best average performance, reaching 81.06% AUROC and 64.44% AUPRC. Therefore, we adopt NN with L_{MSE} as the default meta-learner in our experiments.

The architecture and hyperparameter settings of the selected NN meta-learner are summarized in Table 11. Let n denote the number of historical datasets in $\mathcal{D}_{\text{his}}$. Under the *leave-one-out* protocol, each training fold contains $n - 1$ historical tasks. We further split these $(\mathbf{v}, \mathbf{p})$ pairs into a training set and a validation set, where 20% of the pairs are used for validation to select the final well-trained meta-learner S^*.

Table 10: Performance comparison of different meta-learners on the **Tabular Testbed**. Average AUROC (%) and AUPRC (%) are reported under the *leave-one-out* protocol.

Regressor	AUROC (%)	AUPRC (%)
Random Forest	77.19	50.72
XGBoost	76.74	51.53
SVR	77.32	53.97
NN (MAE)	78.72	61.49
NN (MSE)	**81.06**	**64.44**
NN (MAE+RMSE)	80.07	59.69
NN (Rank loss)	72.83	53.23

Table 11: Hyperparameter settings of the selected neural network meta-learner.

Item	Setting
Architecture	MLP: $d_{\text{in}} \to \text{Act}(d_{\text{hidden}}) \to \text{Act}(d_{\text{hidden}}) \to d_{\text{out}}$
Act	$\text{sigmoid}(\cdot)$
d_{hidden}	512
Loss	L_{MSE}
Optimizer	AdamW (Loshchilov & Hutter, 2017), $lr = 10^{-3}$, weight decay $= 10^{-2}$
Training set size	$N_t = (n-1) - N_v$
Validation set size	$N_v = \lfloor 0.2(n-1) \rfloor$

H Alternative Graph Measurements

GOOD is not restricted to the specific combination of WL and MFD. Our default design combines the WL subtree kernel for local structural characterization with MFD for global structural characterization, providing a practical trade-off among structural coverage, model-selection performance, and computational cost. We regard this combination as one feasible instantiation rather than a universally optimal choice. Given the broad literature on graph kernels and graph distances, other measurements can be readily incorporated into the proposed framework.

To investigate this flexibility, we consider two alternative graph measurements: the Maximum Mean Discrepancy graph kernel (MMD-GK) (Sun & Fan, 2024) and the entropic-regularized Gromov–Wasserstein distance (GWD) (Peyré et al., 2016; Zhang et al., 2024). MMD-GK applies maximum mean discrepancy (Gretton et al., 2012) to node representations obtained through message passing. For both alternatives, we evaluate the graph measurement individually and in combination with WL to examine whether local neighborhood information provides complementary information.

Experimental results. We conduct the same leave-one-out evaluation on the Tabular-testbed (47 tabular tasks) used in the main experiments. Table 12 reports the average model-selection performance in terms of AUROC and AUPRC.

Table 12: Model-selection performance using alternative graph measurements on the Tabular-testbed.

Graph Measurement	AUROC (%)	AUPRC (%)
MMD-GK	78.16	58.05
GWD	78.55	**64.19**
WL + MMD-GK	**80.05**	58.36
WL + GWD	79.77	63.11

Alternative graph measurements remain competitive with the default selection, indicating that the fundamental effectiveness of the proposed framework is not contingent on specific settings. In particular, GWD alone achieves strong AUPRC performance, while combining WL with either MMD-GK or GWD improves AUROC from 78.16% to 80.05% and from 78.55% to 79.77%,

respectively. These alternatives introduce different computational trade-offs. MMD-GK has quadratic complexity with respect to the number of nodes, while entropic GWD incurs a substantially higher computational cost for each pair of graphs. Since GOOD requires comparing a new dataset against multiple historical datasets, such pairwise costs directly affect online model-selection efficiency. Consequently, the choice of graph measurement involves a trade-off among structural information, selection performance, and computational efficiency.

These results demonstrate the modularity of GOOD: alternative graph kernels or distances can be incorporated into the same graph-induced meta-feature framework without changing the overall model-selection pipeline. MMD-GK and GWD provide two feasible alternatives with competitive performance, while the WL+MFD combination used in the main paper provides a practical balance between complementary local/global structural characterization and computational cost.

I LIMITATION

While the proposed GOOD framework demonstrates strong performance and generalization across tabular and image datasets, the current evaluation is limited to these modalities. Its effectiveness on time-series or other complex data types remains to be explored. A structural limitation inherent to all meta-selection methods, including MetaOD (Zhao et al., 2021), ELECT (Zhao et al., 2022), and MetaOOD (Qin et al., 2024) and GOOD, is their reliance on a finite pool (Algorithm, HPC). Consequently, the selection performance is upper-bounded by the coverage and quality of this pool.

J MORE RESULTS AND OVERALL PERFORMANCE SELECTION OF ALL BASELINES ON TWO TESTBEDS

In addition to the *leave-one-out* experiments, we select a fixed subset as the test set and gradually increase the number of historical datasets for both modalities. The corresponding results are illustrated in Figure 6, which show that increasing the number of historical datasets yields positive gains in detector selection, particularly when the training set is small.

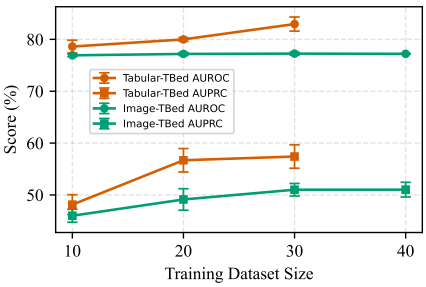


Figure 6: The performance trending with the increase of historical tasks.

The overall model selection performance on the two testbeds is reported in Tables 13, 14, 15, and 16. Note that AUROC and AUPRC may correspond to different (Algorithm, HPC) pairs, as the selection processes for these two metrics are conducted independently.

Table 13: Overall AUROC performance on **Tabular Testbed**.

Dataset	KNN	DSVDD	COPOD	MROT	MC	AS	ISAC	MetaOD	ELECT	MetaOOD	Ours
ALOI	0.5267	0.5104	0.6322	0.5055	0.5217	0.5061	0.5061	0.4999	0.5266	0.5332	0.5237
annthyroid	0.4768	0.5043	0.7644	0.9342	0.9177	0.8903	0.8080	0.4785	0.8080	0.8428	0.9308
backdoor	0.5128	0.5180	0.9780	0.9656	0.9356	0.7779	0.9937	0.4596	0.9915	0.9861	0.9944
breastw	0.4897	0.4503	0.9975	0.9957	0.9934	0.9004	0.9902	0.5020	0.9906	0.9936	0.9894
campaign	0.5736	0.5651	0.7861	0.7817	0.4999	0.5735	0.7738	0.5520	0.7937	0.6882	0.7552
cardio	0.9313	0.7547	0.9317	0.8195	0.9454	0.7988	0.9127	0.7921	0.9454	0.8601	0.9207
Cardiotocography	0.8056	0.8041	0.6634	0.5758	0.7113	0.8458	0.5940	0.7653	0.5955	0.5106	0.6038
celeba	0.8145	0.7803	0.7494	0.8298	0.4999	0.3513	0.7547	0.4887	0.7937	0.7933	0.7547
census	0.9937	0.8511	0.6545	0.6998	0.4999	0.2813	0.7279	0.9872	0.7243	0.5366	0.7220
cover	0.9610	0.9206	0.9876	0.9115	0.7913	0.9937	0.9986	0.9461	0.9986	0.9989	0.9994
donors	0.9628	0.7059	0.7986	0.8077	0.9734	0.8239	0.8785	0.8478	0.6567	0.9734	0.9911
fault	0.2974	0.7862	0.4914	0.5564	0.5787	0.4762	0.5797	0.2973	0.5782	0.6230	0.5270
fraud	0.2334	0.2260	0.9888	0.9420	0.6942	0.9752	0.6530	0.2334	0.9884	0.9557	0.9932
glass	0.9935	0.8271	0.7324	0.7788	0.7659	0.8964	0.7627	0.7882	0.8802	0.5340	0.8414
Hepatitis	0.2972	0.3591	0.7330	0.7907	0.8009	0.7523	0.7783	0.4404	0.7760	0.8100	0.8009
http	0.9372	0.9313	0.9931	0.9362	1.0000	1.0000	1.0000	0.9667	1.0000	1.0000	1.0000
InternetAds	0.5000	0.9160	0.6593	0.6537	0.7384	0.4823	0.5071	0.6870	0.7854	0.7804	0.7926
Ionosphere	0.9156	0.8335	0.7791	0.6356	0.9240	0.8749	0.9614	0.7585	0.9558	0.2402	0.9614
landsat	0.9833	0.6548	0.4929	0.6298	0.5677	0.6583	0.6802	0.9661	0.6649	0.5838	0.6807
letter	0.8836	0.8832	0.3653	0.4607	0.3558	0.3493	0.4467	0.5304	0.3613	0.3166	0.3630
Lymphography	0.4992	0.5181	0.9929	0.9061	0.9531	0.9296	0.9413	0.0735	0.9295	0.9531	0.9577
magic.gamma	0.5000	0.3349	0.6873	0.7915	0.7711	0.6399	0.8379	0.5000	0.8369	0.7815	0.8349
mammography	0.5000	0.3220	0.9003	0.8613	0.8800	0.8429	0.8739	0.5000	0.8662	0.8447	0.8740
mnist	0.9839	0.8857	0.7740	0.8812	0.5439	0.5269	0.9388	0.9703	0.9320	0.9251	0.9388
musk	0.9767	0.9400	0.9971	0.09617	1.0000	0.9934	1.0000	0.5585	1.0000	0.4478	1.0000
optdigits	0.8581	0.8474	0.6929	0.6295	0.5000	0.7980	0.9666	0.8563	0.9406	0.5461	0.9666
PageBlocks	0.3991	0.7354	0.8085	0.8823	0.8913	0.9133	0.8879	0.6122	0.8918	0.8804	0.8961
pendigits	0.9452	0.3976	0.9068	0.8604	0.9986	0.5684	0.9990	0.5122	0.9798	0.9947	0.9993
Pima	0.9583	0.9475	0.6560	0.7508	0.7500	0.7301	0.7486	0.3406	0.7556	0.6894	0.7464
satellite	0.9651	0.9499	0.6835	0.7824	0.7590	0.8960	0.8247	0.3330	0.7868	0.8075	0.8247
satimage-2	0.3921	0.5072	0.9791	0.9966	0.9989	0.9971	0.9963	0.5288	0.9969	0.9498	0.9963
shuttle	0.9833	0.7232	0.9931	0.9724	0.9976	0.4975	0.9982	0.9651	0.9979	0.9935	0.9985
skin	0.9839	0.9538	0.7655	0.9615	0.9459	0.8722	0.8722	0.9386	0.9444	0.9678	0.9411
smtp	0.9839	0.9530	0.9634	0.8627	0.9270	1.0000	1.0000	0.9159	1.0000	1.0000	1.0000
SpamBase	0.8518	0.8567	0.7208	0.8522	0.8295	0.8391	0.8391	0.4761	0.8212	0.8394	0.8125
speech	0.9933	0.9823	0.3703	0.4223	0.5000	0.5000	0.4913	0.9915	0.3586	0.3760	0.4384
Stamps	0.9974	0.9876	0.9340	0.9121	0.9226	0.8972	0.9220	0.9964	0.9213	0.9442	0.9261
thyroid	0.9955	0.9858	0.9381	0.9751	0.9857	0.7796	0.8771	0.9969	0.9868	0.9794	0.9814
vertebral	0.9938	0.9826	0.2739	0.1092	0.1352	0.2460	0.3210	0.9935	0.2504	0.0898	0.2324
vowels	0.9966	0.9941	0.5281	0.4139	0.7698	0.5834	0.5040	0.4000	0.8206	0.7630	0.8149
Waveform	0.7738	0.7534	0.7237	0.5324	0.4997	0.6499	0.7380	0.7662	0.7645	0.7683	0.7401
WBC	0.6258	0.5309	0.9981	0.9939	0.9907	0.9813	0.9897	0.4997	0.9878	0.9935	0.9860
WDBC	0.6482	0.5780	0.9983	0.9016	0.9989	0.9804	0.9972	0.5586	0.9949	0.9989	0.9989
Wilt	0.5050	0.5247	0.3209	0.5402	0.4730	0.4947	0.6397	0.5821	0.5258	0.6316	0.6169
wine	0.7197	0.7485	0.8733	0.5950	0.4600	0.5300	0.4667	0.6584	0.5066	0.7333	0.4933
WPBC	0.5096	0.5717	0.5067	0.5176	0.4810	0.4106	0.5034	0.6403	0.4680	0.5852	0.4902
yeast	0.5032	0.4864	0.3885	0.4377	0.4253	0.5000	0.4679	0.4838	0.4488	0.3992	0.4473
Average	0.7474	0.7187	0.7565	0.7372	0.7468	0.7192	0.7861	0.6518	0.7985	0.7541	0.8106

Table 14: Overall AUPRC performance on **Tabular Testbed**.

Dataset	KNN	DSVDD	COPOD	MROT	MC	AS	ISAC	MetaOD	ELECT	MetaOOD	Ours
ALOI	0.0650	0.0535	0.09779	0.06229	0.0636	0.0620	0.0620	0.0606	0.5295	0.1395	0.0939
annthyroid	0.0616	0.5296	0.2911	0.6611	0.6401	0.6348	0.5368	0.5280	0.6729	0.6930	0.6930
backdoor	0.0626	0.5296	0.5763	0.3999	0.8440	0.6317	0.9590	0.0655	0.9108	0.7629	0.9135
breastw	0.0590	0.5296	0.9974	0.9954	0.9927	0.9029	0.9899	0.0646	0.9974	0.9972	0.9908
campaign	0.0827	0.5296	0.5269	0.5176	0.2026	0.2475	0.4791	0.0776	0.4930	0.4931	0.5085
cardio	0.6914	0.5690	0.7478	0.5952	0.8041	0.5601	0.7646	0.6397	0.7858	0.7646	0.6049
Cardiotocography	0.5040	0.5779	0.5600	0.4497	0.6204	0.7522	0.5569	0.5267	0.5549	0.5593	0.5550
celeba	0.4986	0.1402	0.1739	0.2174	0.0440	0.0311	0.1486	0.4512	0.1721	0.4996	0.4643
census	0.9590	0.0242	0.1511	0.1798	0.1169	0.1177	0.2112	0.9352	0.5496	0.5696	0.5666
cover	0.7225	0.0758	0.5743	0.2880	0.5069	0.8499	0.9246	0.2882	0.0942	0.8387	0.8650
donors	0.7805	0.0294	0.3104	0.2271	0.6655	0.3694	0.3582	0.1062	0.8324	0.8280	0.8246
fault	0.0327	0.0679	0.5308	0.5486	0.6161	0.5173	0.6145	0.0327	0.6216	0.5156	0.7639
fraud	0.0303	0.0959	0.2214	0.02731	0.3289	0.1314	0.0316	0.7841	0.0017	0.2556	0.4821
glass	0.9565	0.0296	0.2466	0.2705	0.1844	0.4062	0.1662	0.0289	0.1662	0.5900	0.3989
Hepatitis	0.0327	0.6702	0.4231	0.4841	0.5589	0.5547	0.4524	0.0276	0.6612	0.5441	0.6477
http	0.2940	0.5239	0.2984	0.07920	1.0000	1.0000	1.0000	0.2939	1.0000	0.9643	1.0000
InternetAds	0.5239	0.5239	0.6170	0.5294	0.5175	0.2819	0.6435	0.4242	0.4777	0.6577	0.6577
Ionosphere	0.5737	0.5239	0.7773	0.5460	0.9396	0.9155	0.9712	0.5563	0.8712	0.9694	0.8694
landsat	0.9022	0.5015	0.3376	0.4677	0.4167	0.5590	0.5434	0.8434	0.6060	0.1716	0.5889
letter	0.4525	0.5239	0.08672	0.1156	0.0874	0.0856	0.2713	0.4608	0.0784	0.5680	0.5610
Lymphography	0.0239	0.5239	0.9061	0.6260	0.6441	0.3408	0.5035	0.5239	0.9742	0.6225	0.5566
magic.gamma	0.5239	0.5239	0.7273	0.8171	0.8134	0.7343	0.8627	0.5239	0.7601	0.8005	0.8653
mammography	0.5239	0.5239	0.5231	0.3024	0.4000	0.3238	0.4274	0.5239	0.4539	0.3692	0.4311
mnist	0.9167	0.5015	0.3485	0.5024	0.2391	0.4219	0.7295	0.3124	0.5741	0.7716	0.7724
musk	0.7980	0.0500	0.9610	0.03376	1.0000	0.9127	1.0000	0.6163	0.5421	1.0000	0.5308
optdigits	0.5416	0.5837	0.08184	0.06634	0.0559	0.1234	0.5984	0.1529	0.3032	0.1467	0.5283
PageBlocks	0.1405	0.0793	0.4144	0.5157	0.6615	0.7012	0.6720	0.1150	0.6719	0.7104	0.6290
pendigits	0.7166	0.0259	0.3043	0.1450	0.9744	0.2403	0.9771	0.5239	0.4550	0.4803	0.9805
Pima	0.4855	0.5239	0.6815	0.6967	0.7220	0.7217	0.7294	0.5765	0.7025	0.7276	0.7294
satellite	0.4189	0.5239	0.7332	0.8020	0.8174	0.9106	0.8615	0.5153	0.8609	0.8385	0.8451
satimage-2	0.0712	0.1423	0.8521	0.8791	0.9786	0.9540	0.9575	0.1423	0.9468	0.5360	0.9358
shuttle	0.9015	0.5239	0.9716	0.7559	0.9870	0.5641	0.9563	0.9406	0.9907	0.9929	0.9645
skin	0.9167	0.5239	0.4721	0.7017	0.8288	0.6023	0.6023	0.8985	0.8860	0.4216	0.7921
smtp	0.9167	0.5017	0.01011	0.002695	0.0084	1.0000	1.0000	0.9244	0.0095	0.0059	1.0000
SpamBase	0.5332	0.5239	0.7352	0.8687	0.8299	0.8837	0.8837	0.5239	0.8212	0.8334	0.7906
speech	0.9921	0.7592	0.02602	0.02838	0.0326	0.5163	0.0336	0.7592	0.0410	0.0308	0.5163
Stamps	0.9973	0.7592	0.5528	0.5051	0.5620	0.5223	0.5582	0.9956	0.5760	0.4580	0.5412
thyroid	0.9959	0.7592	0.2967	0.5172	0.7925	0.5149	0.5985	0.8171	0.8081	0.5768	0.7824
vertebral	0.9892	0.7592	0.1477	0.1264	0.1338	0.1421	0.1551	0.9961	0.1299	0.1453	0.1453
vowels	0.9969	0.7592	0.06711	0.05129	0.2818	0.1338	0.0873	0.7592	0.0861	0.5332	0.3141
Waveform	0.4791	0.6013	0.09665	0.05845	0.0564	0.0787	0.2731	0.5916	0.0928	0.0908	0.2894
WBC	0.3448	0.3516	0.9798	0.9406	0.9097	0.8227	0.8964	0.4537	0.9904	0.8525	0.8125
WDBC	0.3737	0.6013	0.9759	0.2023	0.9833	0.6558	0.9598	0.6013	0.8571	0.5289	0.9569
Wilt	0.3540	0.6013	0.06829	0.09823	0.0870	0.0651	0.1279	0.2642	0.0784	0.1610	0.1823
wine	0.4819	0.6013	0.4929	0.4135	0.1355	0.1369	0.1236	0.6013	0.1321	0.5714	0.2011
WPBC	0.4363	0.5541	0.3637	0.3635	0.4007	0.3518	0.4118	0.4664	0.4603	0.3921	0.6605
yeast	0.2438	0.6013	0.4671	0.4691	0.4741	0.7548	0.4940	0.2160	0.4763	0.4888	0.4843
Average	0.5191	0.4432	0.4639	0.4075	0.5310	0.5051	0.5779	0.4794	0.5480	0.5631	0.6444

Table 15: Overall AUROC performance on **Image Testbed**.

Dataset	KNN	DSVDD	COPOD	MROT	MC	AS	ISAC	MetaOD	ELECT	MetaOOD	Ours
fmnist_0	0.9221	0.7605	0.8005	0.8556	0.7366	0.8986	0.9133	0.9067	0.9221	0.9265	0.9010
fmnist_1	0.9918	0.8680	0.9273	0.9540	0.5000	0.1885	0.9738	0.9618	0.9916	0.9871	0.9738
fmnist_2	0.8923	0.8444	0.6935	0.8370	0.6138	0.8508	0.8693	0.8712	0.8923	0.8573	0.8508
fmnist_3	0.9244	0.7163	0.8310	0.8397	0.5969	0.9180	0.9171	0.9118	0.9288	0.9296	0.9271
fmnist_4	0.8852	0.7639	0.7214	0.8139	0.7854	0.8844	0.8844	0.6951	0.8895	0.8842	0.8844
fmnist_5	0.9583	0.8498	0.9171	0.9360	0.5000	0.9583	0.9215	0.9414	0.9498	0.9461	0.9215
fmnist_6	0.8234	0.7316	0.5785	0.6867	0.7439	0.8146	0.8004	0.6509	0.8160	0.8218	0.8146
fmnist_7	0.9723	0.8717	0.9485	0.9502	0.4998	0.9197	0.9715	0.9595	0.9726	0.9673	0.9715
fmnist_8	0.9076	0.6222	0.6763	0.7640	0.7183	0.8938	0.8938	0.5466	0.9063	0.8938	0.9098
fmnist_9	0.9692	0.8503	0.9388	0.9247	0.7071	0.8735	0.9760	0.9742	0.9759	0.7927	0.9760
cifar10_0	0.7668	0.5751	0.6942	0.7132	0.9238	0.6545	0.6545	0.3870	0.7621	0.5761	0.7433
cifar10_1	0.6460	0.5774	0.4876	0.4938	0.9902	0.7072	0.5649	0.6405	0.6470	0.6342	0.7467
cifar10_2	0.6501	0.4773	0.5916	0.5821	0.8710	0.5874	0.5874	0.6128	0.6447	0.6414	0.6618
cifar10_3	0.5991	0.5450	0.4988	0.5342	0.9094	0.5303	0.5285	0.5948	0.6092	0.5285	0.6260
cifar10_4	0.7975	0.6167	0.7569	0.7612	0.8878	0.7046	0.7018	0.3882	0.7989	0.7797	0.7682
cifar10_5	0.5690	0.5798	0.4632	0.4979	0.9560	0.5253	0.5224	0.6045	0.5772	0.5224	0.6051
cifar10_6	0.7732	0.5362	0.6741	0.7095	0.7919	0.6889	0.6889	0.6271	0.7699	0.7379	0.7487
cifar10_7	0.6789	0.5820	0.5879	0.5762	0.9687	0.5756	0.5756	0.6369	0.6848	0.5756	0.7227
cifar10_8	0.7293	0.5431	0.6743	0.6242	0.8621	0.6340	0.6322	0.4414	0.7292	0.7246	0.7537
cifar10_9	0.7161	0.6033	0.6300	0.6274	0.9699	0.7793	0.5922	0.5220	0.7268	0.7268	0.7712
svhn_0	0.6970	0.6025	0.5255	0.6018	0.5219	0.5057	0.7021	0.5719	0.7020	0.7008	0.7254
svhn_1	0.6962	0.6106	0.6275	0.6486	0.5325	0.5159	0.6951	0.3535	0.6925	0.6947	0.6435
svhn_2	0.6633	0.5870	0.5842	0.6159	0.7063	0.5117	0.6660	0.5021	0.6659	0.6414	0.6913
svhn_3	0.6025	0.5407	0.5570	0.5732	0.9972	0.5065	0.6077	0.5938	0.6096	0.6461	0.6536
svhn_4	0.6524	0.5865	0.6055	0.6059	0.9503	0.5120	0.6544	0.4328	0.6464	0.5598	0.6517
svhn_5	0.6442	0.5554	0.5776	0.6113	0.4985	0.5072	0.6479	0.5400	0.6526	0.5841	0.6664
svhn_6	0.6013	0.5364	0.5125	0.5532	1.0000	0.5064	0.6060	0.6148	0.5848	0.5867	0.6051
svhn_7	0.6656	0.5749	0.6309	0.6015	0.7751	0.5088	0.6626	0.6623	0.6865	0.6445	0.6939
svhn_8	0.6200	0.5714	0.5013	0.5655	0.5034	0.5076	0.6259	0.6387	0.6037	0.6320	0.6479
svhn_9	0.6049	0.5886	0.5004	0.5663	0.5403	0.5067	0.6102	0.5989	0.6026	0.5935	0.6054
mnist-c_brightness	0.9187	0.6373	0.6345	0.7802	0.5256	0.9598	0.9598	0.9532	0.9198	0.9083	0.9598
mnist-c_canny_edges	0.9171	0.5857	0.6585	0.5782	0.6262	0.8633	0.9721	0.6844	0.9606	0.9478	0.9606
mnist-c_dotted_line	0.9423	0.7379	0.6986	0.7188	0.7513	0.9601	0.9601	0.8812	0.9377	0.9400	0.9499
mnist-c_fog	0.9967	0.7271	0.7509	0.8585	0.9551	0.9986	0.9986	0.8085	0.9967	0.9982	0.9986
mnist-c_glass_blur	0.9966	0.8091	0.8741	0.8942	0.5409	0.7652	0.9906	0.8514	0.9891	0.9952	0.9906
mnist-c_identity	0.4598	0.5019	0.4849	0.4765	0.6572	0.4814	0.4690	0.4872	0.4715	0.4716	0.4690
mnist-c_impulse_noise	1.0000	0.9627	0.9715	0.8989	0.5013	1.0000	1.0000	0.0134	1.0000	1.0000	1.0000
mnist-c_motion_blur	0.9679	0.8498	0.7701	0.8685	0.6912	0.6795	0.9772	0.9384	0.9623	0.9623	0.9772
mnist-c_rotate	0.6763	0.5526	0.5359	0.5634	0.6643	0.7152	0.7152	0.5467	0.6546	0.5992	0.7152
mnist-c_scale	0.8965	0.6731	0.5648	0.6769	0.6090	0.9206	0.9206	0.4358	0.8913	0.9187	0.9206
mnist-c_shear	0.7399	0.6041	0.6397	0.6349	0.6561	0.6331	0.7840	0.7228	0.7518	0.7425	0.7840
mnist-c_shot_noise	0.9465	0.6916	0.7110	0.8116	0.6511	0.6796	0.9303	0.8922	0.9376	0.9454	0.9303
mnist-c_spatter	0.9366	0.7077	0.7887	0.7848	0.5846	0.9624	0.9624	0.6790	0.9423	0.9210	0.9624
mnist-c_stripe	0.9998	0.9307	0.9680	0.8746	0.6849	0.9998	0.9998	0.9986	0.9998	0.9998	0.9998
mnist-c_translate	0.8207	0.6392	0.5592	0.5792	0.6034	0.8887	0.8887	0.7275	0.7844	0.8887	0.8887
mnist-c_zigzag	0.9509	0.7314	0.7628	0.7748	0.6022	0.7551	0.9557	0.7908	0.9516	0.9537	0.9557
mvtec_bottle	0.9738	0.9045	0.9657	0.9388	0.9728	0.9654	0.9765	0.9764	0.9737	0.9706	0.9656
mvtec_cable	0.7839	0.6435	0.6666	0.5754	0.7288	0.6407	0.7789	0.6394	0.7839	0.6578	0.7842
mvtec_capsule	0.7253	0.6476	0.6687	0.5556	0.6871	0.6307	0.7441	0.6527	0.7232	0.6703	0.7180
mvtec_carpet	0.7720	0.6829	0.7189	0.6662	0.7521	0.7321	0.7669	0.6297	0.7485	0.7565	0.7712
mvtec_grid	0.7714	0.5408	0.6212	0.6341	0.5877	0.6995	0.7759	0.7613	0.7633	0.5408	0.6846
mvtec_hazelnut	0.7754	0.7417	0.7395	0.7050	0.7569	0.7264	0.7752	0.7387	0.7921	0.7754	0.7754
mvtec_leather	0.9955	0.9785	0.9839	0.9693	0.9942	0.9917	0.9875	0.9883	0.9955	0.9933	0.9671
mvtec_metal_nut	0.7070	0.6540	0.5984	0.5728	0.5819	0.7029	0.7228	0.7173	0.6799	0.7070	0.7029
mvtec_pill	0.6671	0.6124	0.6324	0.6514	0.6406	0.6552	0.6552	0.6136	0.6671	0.6124	0.6595
mvtec_screw	0.6781	0.5628	0.5909	0.5600	0.5084	0.6830	0.6936	0.5692	0.5915	0.5596	0.6936
mvtec_tile	0.8695	0.7116	0.8308	0.7454	0.8546	0.8148	0.8635	0.5504	0.8759	0.8412	0.8720
mvtec_toothbrush	0.8731	0.7426	0.6574	0.5481	0.5167	0.8787	0.8815	0.5991	0.8787	0.4991	0.8713
mvtec_transistor	0.7235	0.6973	0.7173	0.5211	0.7133	0.6401	0.7358	0.6776	0.7145	0.7358	0.7146
mvtec_wood	0.8104	0.7109	0.7258	0.6598	0.7576	0.7301	0.8118	0.8224	0.7486	0.7815	0.7868
mvtec_zipper	0.8645	0.7191	0.7822	0.6531	0.8054	0.8149	0.8503	0.8377	0.8532	0.8309	0.8481
Average	0.7996	0.6747	0.6883	0.6943	0.7167	0.7253	0.7893	0.6814	0.7964	0.7682	0.8056

Table 16: Overall AUPRC performance on **Image Testbed**.

Dataset	KNN	DSVDD	COPOD	MROT	MC	AS	ISAC	MetaOD	ELECT	MetaOOD	Ours
fmnist_0	0.6252	0.2767	0.2714	0.3229	0.2247	0.5136	0.6586	0.4884	0.6246	0.6050	0.5532
fmnist_1	0.9158	0.6342	0.5429	0.5646	0.0952	0.0557	0.8668	0.8255	0.8999	0.8987	0.5707
fmnist_2	0.5783	0.5579	0.1447	0.2707	0.1355	0.4992	0.5686	0.4417	0.5777	0.4992	0.5520
fmnist_3	0.6696	0.3780	0.2881	0.3031	0.1396	0.6583	0.6534	0.5475	0.6691	0.6534	0.5511
fmnist_4	0.6405	0.5403	0.1855	0.3272	0.3074	0.6467	0.6467	0.5677	0.5894	0.5475	0.5455
fmnist_5	0.8648	0.7045	0.7168	0.5736	0.0952	0.8648	0.7908	0.8466	0.8636	0.8176	0.5623
fmnist_6	0.4453	0.2892	0.1084	0.1631	0.2229	0.4696	0.4714	0.3193	0.4733	0.4714	0.5485
fmnist_7	0.9136	0.7897	0.8245	0.6673	0.0952	0.8324	0.9171	0.8599	0.9204	0.9074	0.5714
fmnist_8	0.4810	0.1664	0.1566	0.1833	0.1866	0.5813	0.5813	0.5476	0.2634	0.5791	0.5489
fmnist_9	0.8945	0.7223	0.7337	0.5247	0.2200	0.4550	0.8977	0.8652	0.8979	0.8497	0.5654
cifar10_0	0.2566	0.1278	0.2056	0.1790	0.6125	0.5267	0.5267	0.5477	0.2700	0.2352	0.5476
cifar10_1	0.1466	0.1618	0.08794	0.08557	0.9006	0.2163	0.5217	0.5480	0.5475	0.5217	0.5480
cifar10_2	0.1559	0.1126	0.1196	0.1157	0.4740	0.5044	0.5044	0.5476	0.5475	0.1776	0.5476
cifar10_3	0.1488	0.1062	0.09663	0.1100	0.5930	0.5042	0.5057	0.5477	0.1294	0.5057	0.5477
cifar10_4	0.3484	0.1777	0.2703	0.2748	0.6335	0.5107	0.5147	0.5476	0.3483	0.5476	0.5478
cifar10_5	0.1293	0.1242	0.08481	0.09240	0.8563	0.5157	0.5171	0.1316	0.1318	0.5171	0.5476
cifar10_6	0.2501	0.1508	0.1462	0.1627	0.3186	0.5258	0.5258	0.5478	0.2472	0.5258	0.5478
cifar10_7	0.2421	0.1156	0.1722	0.1274	0.8987	0.5062	0.5062	0.5476	0.5475	0.4961	0.5476
cifar10_8	0.2069	0.1485	0.1522	0.1223	0.3381	0.5232	0.5262	0.5476	0.2075	0.5476	0.5476
cifar10_9	0.2430	0.1856	0.1686	0.1469	0.8771	0.3139	0.5152	0.5476	0.1964	0.2430	0.5476
svhn_0	0.2031	0.1466	0.1061	0.1337	0.1224	0.5446	0.2967	0.5477	0.1829	0.1933	0.5446
svhn_1	0.2006	0.1545	0.1446	0.1528	0.1377	0.5481	0.3044	0.5482	0.1969	0.5476	0.5481
svhn_2	0.1979	0.1212	0.1571	0.1542	0.3544	0.5468	0.2581	0.5468	0.1993	0.1936	0.5468
svhn_3	0.1504	0.1059	0.1076	0.1173	0.9893	0.5441	0.2270	0.5441	0.1277	0.2270	0.5441
svhn_4	0.1777	0.1184	0.1596	0.1358	0.8971	0.5452	0.2617	0.5452	0.1787	0.2617	0.5452
svhn_5	0.1787	0.1346	0.1216	0.1387	0.0951	0.5470	0.2440	0.5476	0.5476	0.2440	0.5469
svhn_6	0.1486	0.1087	0.1026	0.1237	0.9997	0.5466	0.2362	0.5466	0.1057	0.1428	0.5466
svhn_7	0.1834	0.1233	0.1578	0.1430	0.5594	0.5437	0.2750	0.5476	0.1930	0.5470	0.5477
svhn_8	0.1597	0.1329	0.09870	0.1309	0.0986	0.5465	0.2366	0.5465	0.1152	0.1574	0.5465
svhn_9	0.1434	0.1219	0.09646	0.1177	0.1360	0.5426	0.2205	0.5478	0.2204	0.1447	0.5446
mnist-c_brightness	0.5326	0.2256	0.1302	0.2044	0.1208	0.6912	0.6912	0.6912	0.5598	0.5598	0.5700
mnist-c_canny_edges	0.5036	0.1561	0.1382	0.1002	0.2474	0.3353	0.7676	0.2641	0.5297	0.5304	0.5716
mnist-c_dotted_line	0.6570	0.3252	0.1611	0.1599	0.4992	0.7530	0.7530	0.5913	0.2961	0.6494	0.5703
mnist-c_fog	0.9644	0.3597	0.2178	0.3410	0.8922	0.9834	0.9834	0.9323	0.9722	0.9564	0.5492
mnist-c_glass_blur	0.9689	0.6172	0.3522	0.4232	0.1223	0.6274	0.9309	0.9764	0.7555	0.5476	0.5716
mnist-c_identity	0.0870	0.0916	0.09042	0.09170	0.3489	0.0894	0.0872	0.0881	0.0881	0.0872	0.3470
mnist-c_impulse_noise	1.0000	0.9998	0.6806	0.3210	0.0967	1.0000	1.0000	1.0000	1.0000	0.9941	0.9950
mnist-c_motion_blur	0.8311	0.4770	0.2455	0.3280	0.1988	0.5705	0.8695	0.7288	0.8392	0.8556	0.5705
mnist-c_rotate	0.1759	0.1286	0.1024	0.1051	0.1984	0.2099	0.2099	0.1331	0.1596	0.5018	0.5018
mnist-c_scale	0.4489	0.1258	0.1046	0.1580	0.1469	0.5052	0.5052	0.4824	0.4249	0.4824	0.5670
mnist-c_shear	0.2715	0.1365	0.1568	0.1422	0.1794	0.5202	0.3448	0.2828	0.3334	0.5475	0.5193
mnist-c_shot_noise	0.6821	0.1889	0.1852	0.2433	0.1811	0.5697	0.6543	0.4609	0.6804	0.6543	0.5494
mnist-c_spatter	0.7518	0.3998	0.2712	0.2127	0.1465	0.8422	0.8422	0.6534	0.7776	0.5476	0.5645
mnist-c_stripe	0.9981	0.6157	0.6752	0.3307	0.1941	0.9983	0.9983	0.9976	0.9984	0.9976	0.5493
mnist-c_translate	0.2934	0.2277	0.1108	0.1050	0.1587	0.4932	0.4932	0.2064	0.5476	0.3122	0.5650
mnist-c_zigzag	0.6921	0.3654	0.2039	0.1841	0.1419	0.4492	0.7352	0.7352	0.7236	0.7089	0.5491
mvtec_bottle	0.9728	0.8336	0.9578	0.9199	0.9688	0.9665	0.9744	0.9743	0.9679	0.9679	0.9719
mvtec_cable	0.7233	0.5242	0.5659	0.4948	0.6466	0.5427	0.7185	0.6974	0.7279	0.7218	0.7233
mvtec_capsule	0.7364	0.6193	0.6753	0.5291	0.6917	0.6677	0.7527	0.7370	0.7369	0.7267	0.7527
mvtec_carpet	0.7321	0.5477	0.6682	0.5105	0.7156	0.7019	0.7411	0.7209	0.7228	0.7304	0.7411
mvtec_grid	0.6947	0.6211	0.4193	0.4009	0.4104	0.5819	0.7016	0.6945	0.6424	0.6855	0.6855
mvtec_hazelnut	0.6354	0.4509	0.5871	0.3527	0.6475	0.5506	0.6432	0.6224	0.6432	0.5939	0.5939
mvtec_leather	0.9937	0.9648	0.9789	0.8882	0.9924	0.9888	0.9876	0.9794	0.9932	0.9803	0.9905
mvtec_metal_nut	0.7008	0.6449	0.5039	0.4663	0.5256	0.7052	0.7159	0.7173	0.7172	0.7159	0.7173
mvtec_pill	0.6654	0.5691	0.6333	0.6233	0.6461	0.6570	0.6570	0.7448	0.6443	0.6444	0.6570
mvtec_screw	0.5566	0.5111	0.4993	0.4440	0.4068	0.6117	0.5785	0.6983	0.6983	0.5521	0.5566
mvtec_tile	0.8639	0.5518	0.8208	0.6207	0.8447	0.7883	0.8596	0.6944	0.8640	0.8512	0.8570
mvtec_toothbrush	0.8533	0.8485	0.6619	0.4760	0.4727	0.8680	0.8670	0.6245	0.7272	0.8563	0.8693
mvtec_transistor	0.5556	0.3153	0.5024	0.2134	0.5293	0.4617	0.5653	0.6130	0.5653	0.6130	0.5406
mvtec_wood	0.7381	0.4393	0.6067	0.4100	0.6867	0.6123	0.7417	0.7297	0.7153	0.7417	0.6898
mvtec_zipper	0.8306	0.6603	0.7158	0.5406	0.7463	0.7896	0.8187	0.7333	0.8135	0.7885	0.8446
Average	0.5247	0.3652	0.3336	0.2952	0.4330	0.5870	0.6059	0.6073	0.5391	0.5722	0.6034