Automatic Unsupervised Ensemble Outlier Model Selection

Anonymous Authors[1]

Abstract

Unsupervised outlier detection is attractive because it eliminates the need for labeled data. Further, forming multi-model ensembles can improve detection robustness performance. However, composing an ensemble without labeled data is challenging. Naively composing ensembles can cause ensemble saturation, where redundant or unreliable detection models degrade performance and incur unnecessary computations. We propose `MetaEns`, an automatic unsupervised framework for the selection of outlier detection model ensembles. Using labeled meta-datasets, `MetaEns` learns a model that predicts marginal ensemble gains that estimate the expected improvement of adding a candidate model to a partially constructed ensemble. At test time, this learned signal is combined with a submodular-inspired proxy objective that enforces diminishing returns through diversity-aware discounting and family-level risk regularization, thereby enabling greedy sequential selection with adaptive early stopping. As a result, `MetaEns` constructs compact, high-quality ensembles without access to ground-truth labels. Experiments on 39 real-world datasets show that `MetaEns` is able to consistently outperform state-of-the-art unsupervised selectors and ensemble baselines, achieving higher average precision while using fewer models.

1. Introduction

Outlier detection plays a critical role in applications like fraud detection, network security, medical diagnosis, and system monitoring(Chandola et al., 2009; Ruff et al., 2021). However, in many real-world scenarios where ground-truth labels are unavailable, and outlier detection must be performed in a fully unsupervised manner(Zimek et al., 2013). Further, contamination rates are often unknown, and data distributions vary widely across tasks(Han et al., 2022). These factors make it difficult to not only detect anomalies but also evaluate and compare detection models.

Existing studies have proposed diverse unsupervised outlier detectors based on density estimation (Breunig et al., 2000), isolation mechanisms (Liu et al., 2008), reconstruction errors (Zong et al., 2018), and deep representations (Ruff et al., 2018). Despite their success in specific settings, no single detector performs reliably across diverse datasets. This observation has motivated the use of ensemble methods, which aim to improve robustness by aggregating multiple detectors. In supervised learning, ensembles can be trained and validated using labeled data. In contrast, constructing effective ensembles for unsupervised outlier detection remains an open challenge (Zimek et al., 2013).

A key challenge is to select models when no labels are available (Marques et al., 2020). Without ground-truth feedback, it is unclear which detection models are reliable on a given dataset or whether adding a new detector will improve or degrade ensemble performance (Rayana et al., 2016; Aggarwal & Sathe, 2015). As a result, many unsupervised ensemble methods rely on fixed aggregation strategies, such as averaging scores from all available detectors or selecting a fixed number of top-ranked detection models (Zhao et al., 2019a). Methods employing these strategies suffer from *ensemble saturation*: beyond a small ensemble size, adding more detectors yields diminishing or even negative returns due to redundancy, conflicting rankings, or systematically poor models. Moreover, fixed-size ensembles are inherently inflexible and cannot adapt to dataset-specific complexity.

Recent unsupervised model selection methods attempt to address these challenges by leveraging meta-learning across labeled auxiliary datasets (Hospedales et al., 2022). Notably, frameworks such as `MetaOD` (Zhao et al., 2021) and `ELECT` (Zhao et al., 2022) learn to recommend a single detector for a new unlabeled task based on task similarity or historical performance patterns. While effective, these methods are limited to singleton selection and do not address the more general problem of adaptive ensemble construction, where multiple complementary detectors may be required to capture diverse outlier patterns (Cheng et al., 2020).

We address this gap by formulating unsupervised ensemble outlier model selection as a sequential decision problem. Our key insight is that although the true marginal benefit of adding a detector to an ensemble is unobservable at test time, its structure can be learned offline from labeled meta-datasets. Building on this idea, we propose `MetaEns`, a framework that learns to predict the marginal ensemble gain

For ICML 2026 reviewers: When reviewing this paper, please follow the policy for the use of LLMs as specified on your Reviewer Console. Note that the assigned LLM policy might differ from the policy you indicated during reviewer registration.

of candidate detectors conditioned on the current ensemble state. At inference time, `MetaEns` greedily constructs an ensemble by maximizing a submodular-inspired proxy objective (Nemhauser et al., 1978) that integrates the predicted gain with explicit mechanisms for diversity control and risk mitigation.

Specifically, `MetaEns` introduces two principles that are crucial for unsupervised ensemble construction. First, we enforce *diminishing returns* through similarity-based discounting that penalizes candidates that introduce redundancy with already selected detectors (Kulesza & Taskar, 2012). Second, we incorporate *family-risk regularization*, which discourages the selection of multiple detectors from algorithmic families with a history of poor or unstable performance. Together, these components yield a proxy objective that favors compact, diverse ensembles and naturally supports adaptive early stopping when no further improvement is expected.

We evaluate `MetaEns` on a benchmark of 39 real-world anomaly detection datasets (Han et al., 2022) using a large pool of 297 candidate detectors spanning multiple algorithmic families. Extensive experiments show that `MetaEns` is able to consistently outperform strong unsupervised baselines and recent meta-learning approaches. Notably, `MetaEns` achieves higher detection accuracy while selecting fewer models than fixed-size ensembles, and it exhibits strong resilience by recovering performance even when an initially selected detector performs poorly.

In summary, our contributions are threefold:

- We formulate the problem of unsupervised ensemble outlier model selection and cast it as a sequential decision process without access to labels.
- We propose `MetaEns`, a meta-learning framework that predicts marginal ensemble gains and combines these with a diversity- and risk-aware proxy objective to enable adaptive ensemble construction with early stopping.
- We provide experimental results across 39 datasets, supported by ablation studies, showing that compact, adaptively sized ensembles can outperform larger fixed ensembles in fully unsupervised settings.

2. Preliminaries

Definition 2.1 (Dataset)**.** A dataset $\mathbf{X}$ is a finite collection of data instances (or data points) $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, where each instance $\mathbf{x}_i \in \mathbb{R}^d$ is a d-vector. We denote $|\mathbf{X}| = N$ as the cardinality of the dataset.

Definition 2.2 (Unsupervised Outlier Detection)**.** Given a dataset $\mathbf{X} \in \mathbb{R}^{N\times d}$, an unsupervised outlier detection model, or detector, learns a scoring function $f : \mathbf{X} \to \mathbb{R}^N$ that assigns an outlier score $o_i = f(\mathbf{x}_i)$ to each instance $\mathbf{x}_i \in \mathbf{X}$, where larger values indicate a higher likelihood of being an anomaly. A decision function can be derived by thresholding the scores at a user-defined level τ, yielding outlier labels $\hat{y}_i = \mathbb{I}(o_i > \tau)$. The resulting outlier set is defined as follows.

$$\mathbf{O} = \{\mathbf{x}_i \in \mathbf{X} \mid \hat{y}_i = 1\}$$

Definition 2.3 (Unsupervised Outlier Model Selection)**.** Let $\Omega = \{f_1, f_2, \ldots, f_K\}$ be the set of K candidate outlier detection models. Each $f_i \in \Omega$ can be seen as a $(\texttt{detector}, \texttt{configuration})$ tuple, where the `configuration` denotes a set of hyperparameters of the `detector`. Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ be an unlabeled dataset. Next, each outlier detection model f_i work as an outlier scoring function $f_i : \mathbf{X} \in \mathbb{R}^{N\times d} \mapsto \mathbb{R}^N$, which assigns an outlier score o_i to each instance $\mathbf{x}_i \in \mathbf{X}$. The task of unsupervised outlier model selection is then to choose the model f^* as follows.

$$f^* = \arg\min_{f_i \in \Omega} \Gamma(f_i, \mathbf{X})$$

Here, $\Gamma(\cdot)$ is an unsupervised evaluation criterion that estimates the quality of model f_i based solely on the distributional properties of its output scores without using labeled outlier or normal instances.

Problem Definition: Unsupervised Ensemble Outlier Model Selection. Let $\Omega = \{f_1, f_2, \ldots, f_K\}$ be a set of K candidate outlier detection models. Each model $f_i \in \Omega$ can be seen as a $(\texttt{detector}, \texttt{configuration})$ tuple, where the `configuration` denotes a specific set of hyperparameters of the `detector`. Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ be an unlabeled dataset. Next, each outlier detection model $f_i \in \Omega$ work as an outlier scoring function $f_i : \mathbf{X} \in \mathbb{R}^{N\times d} \mapsto \mathbb{R}^N$ that assigns an outlier score o_i to each instance $\mathbf{x}_i \in \mathbf{X}$. The task of unsupervised ensemble outlier model selection is to choose a set of P models, $P \subseteq \Omega$, as follows.

$$P^* = \arg\max_{P \subseteq \Omega} \sum_{1}^{|P|} \Gamma(P, \mathbf{X}) \text{ s.t. } |P| \leq \eta$$

Here η is a user-defined value for the number of ensemble models. Next, $\Gamma(\cdot)$ is an unsupervised evaluation criterion that estimates the quality of model pool P based solely on the distributional properties of output scores of models in P without using labeled outlier or normal instances.

Definition 2.4 (Meta-datasets)**.** We assume access to a collection of labeled meta-datasets $\mathcal{M}$ =

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

$\{(\mathbf{M}_1, \mathbf{y}_1), (\mathbf{M}_2, \mathbf{y}_2), \ldots, (\mathbf{M}_L, \mathbf{y}_L)\}$ where each $\mathbf{M}_i \in \mathbb{R}^{N_{M_i} \times d_{M_i}}$ is a dataset and $\mathbf{y}_i \in \{0,1\}^{N_{M_i}}$ provides ground-truth anomaly labels. These meta-datasets are used exclusively for evaluating and selecting candidate outlier detection models. At test time, we are given an unlabeled dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ that satisfies $\mathbf{X} \sim \mathbb{P}_{\mathbf{X}}$ and $\mathbb{P}_{\mathbf{X}} \neq \mathbb{P}_{\mathbf{M}_i}, \forall i$, i.e., the test distribution differs from the distributions underlying the meta-datasets.

3. Methodology

3.1. Framework Overview

`MetaEns` encompasses two phases: *offline meta-training* and *online model selection*.

Offline meta-training. Given a collection of labeled meta-datasets, we simulate sequential ensemble construction and compute the *true* marginal gain of adding a candidate detector to a partial ensemble using ground-truth labels. These oracle gains provide supervision for learning a meta-model that predicts the *expected* marginal gain conditioned on the current ensemble context.

Online model selection. At test time, `MetaEns` is applied to a new unlabeled dataset. The procedure starts by selecting a high-quality *primary* detector that serves as an anchor. `MetaEns` then expands the ensemble by greedily adding detectors with high predicted utility. Since the predicted gain is a learned one and may be noisy, we introduce a *submodular-inspired* proxy objective that (i) discounts redundant candidates to induce diminishing returns and (ii) penalizes risky algorithm families based on meta-training history. This proxy enables *adaptive early stopping*: selection terminates once no candidate is expected to provide positive utility.

Fig. 1 summarizes the interaction between offline meta-training and online selection.

3.2. Offline Meta-training

Let $\mathcal{M} = \{(\mathbf{M}_\ell, \mathbf{y}_\ell)\}_{\ell=1}^{L}$ be the labeled meta-datasets, and let $\Omega = \{f_1, \ldots, f_K\}$ denote the candidate detector pool (detector, configuration). For a dataset $(\mathbf{M}, \mathbf{y}) \in \mathcal{M}$ and an ensemble (model set) $P \subseteq \Omega$, we define the ensemble score vector as the mean of the member scores:

$$\mathbf{o}_P \;=\; \frac{1}{|P|} \sum_{f \in P} f(\mathbf{M}), \tag{1}$$

where $f(\mathbf{M}) \in \mathbb{R}^{|\mathbf{M}|}$ is the outlier-score vector produced by f on $\mathbf{M}$.

Oracle marginal gain. We define the *true* marginal gain of adding $f_i \in \Omega \setminus P$ to P as the improvement in Average Precision:

$$G(f_i \mid P) \;=\; \mathrm{AP}(\mathbf{o}_{P \cup \{f_i\}}, \mathbf{y}) \;-\; \mathrm{AP}(\mathbf{o}_P, \mathbf{y}), \tag{2}$$

where $\mathrm{AP}(\cdot,\cdot)$ is computed from the ensemble score vector and the ground-truth label vector. We use AP as it is threshold-independent and robust under severe class imbalance, which is common in outlier detection.

Generating training trajectories. To obtain informative supervision, we construct meta-training trajectories using an *oracle greedy policy*. For each meta-dataset, we initialize the ensemble with a primary model f_1^* chosen to maximize $\mathrm{AP}(f(\mathbf{M}), \mathbf{y})$. We then iteratively add models that maximize the true gain in Eq. 2. This strategy exposes the meta-model to high-quality partial ensembles, avoiding training dominated by arbitrary or low-signal states.

At each step i, for every candidate $f \in \Omega \setminus P$, we compute a state representation $\phi(f, f_{i-1}^*, P)$ and its oracle gain $G(f \mid P)$, obtaining supervised pairs $\{(\phi(\cdot), G(\cdot))\}$ across multiple ensemble sizes and datasets. These pairs are used to learn a predictor of marginal gains.

State representation. The state representation $\phi(f_i, f_{i-1}^*, P)$ encodes three factors essential to sequential ensemble construction: (i) *redundancy* between the candidate and the most recent selection, (ii) *compatibility* between the candidate and the current ensemble, and (iii) the *selection stage* via $|P|$. We compute a base feature extractor on normalized score vectors that includes correlation measures (e.g., Spearman correlation, cosine similarity), distributional statistics (e.g., entropy, kurtosis), and overlap-based agreement (e.g., Jaccard similarity over top-ranked instances; see Appendix A.1 for full definitions). Concretely, we compute: (i) $\phi_{f_i, f_{i-1}^*} \in \mathbb{R}^{d_{\text{base}}}$ from $(\mathbf{o}_{f_i}, \mathbf{o}_{f_{i-1}^*})$, (ii) $\phi_{f_i, P} \in \mathbb{R}^{d_{\text{base}}}$ from $(\mathbf{o}_{f_i}, \mathbf{o}_P)$, and (iii) $\phi_{f_{i-1}^*, P} \in \mathbb{R}^{d_{\text{base}}}$ from $(\mathbf{o}_{f_{i-1}^*}, \mathbf{o}_P)$. The final state is:

$$\phi(f_i, f_{i-1}^*, P) = (\phi_{f_i, f_{i-1}^*}, \phi_{f_i, P}, \phi_{f_{i-1}^*, P}, |P|). \tag{3}$$

Marginal gain modeling. Positive marginal gains become increasingly sparse as the ensemble grows, motivating a two-part model that separates *whether* a gain occurs from *how large* it is. We learn an expected gain predictor:

$$\hat{G}(f_i \mid P) = f_{\text{cls}}(f_i \mid P) \cdot f_{\text{reg}}(f_i \mid P), \tag{4}$$

where $f_{\text{cls}}(f_i \mid P) = \mathbb{P}(G(f_i \mid P) > 0)$ predicts the probability of improvement and $f_{\text{reg}}(f_i \mid P) = \mathbb{E}[G(f_i \mid P) \mid G(f_i \mid P) > 0]$ predicts the gain magnitude conditioned on being positive.

For each training pair (ϕ, G), the classification target is $y_{\text{cls}} = \mathbb{I}(G > 0)$. The regression target is $y_{\text{reg}} = \max(0, G)$, optimized only on samples with $G > 0$. We instantiate f_{cls} and f_{reg} with `ExtraTrees` (Geurts et al., 2006).

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

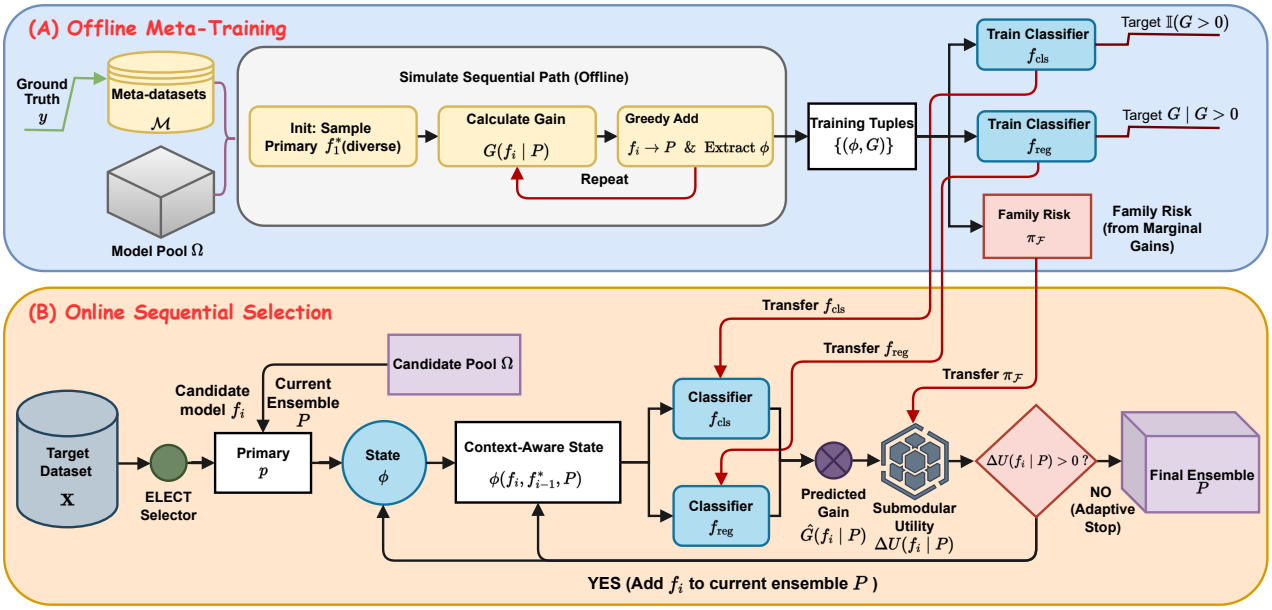

Figure 1. Overview of `MetaEns`.

3.3. Online Model Selection

At test time, the true gain $G(\cdot)$ is unavailable. We therefore select ensembles using a proxy objective that combines the predicted gain $\hat{G}$ with explicit redundancy and risk control. Our goal is to induce *diminishing returns* behavior—a hallmark of submodular maximization—without requiring the learned predictor to satisfy submodularity.

Proxy marginal utility. Given a current ensemble P (with the last selected model being f_{i-1}^*), the marginal utility of adding candidate f_i is:

$$\Delta U(f_i \mid P) = \gamma(f_i, P) \cdot \left(\hat{G}(f_i \mid P) - \lambda_{\text{fam}} \pi_{\mathcal{F}(f_i)}\right), \quad (5)$$

where $\gamma(f_i, P) \in (0, 1]$ discounts redundant candidates, and $\lambda_{\text{fam}} \pi_{\mathcal{F}(f_i)} \geq 0$ penalizes families with historically negative tail behavior. The hyperparameters β and λ_{fam} control the strength of redundancy discounting and family-risk regularization.

Redundancy discount. We define the discount factor as

$$\gamma(f_i, P) = \frac{1}{1 + \beta \cdot \text{sim}_{\max}(f_i, P)}, \quad (6)$$

where $\text{sim}_{\max}(f_i, P)$ is the maximum similarity between f_i and any model in P:

$$\text{sim}_{\max}(f_i, P) = \begin{cases} \max\limits_{f_j \in P} \text{sim}(f_i, f_j) & \text{if } |P| > 0, \\ 0 & \text{if } P = \emptyset, \end{cases} \quad (7)$$

where $\text{sim}(f_i, f_j)$ is the Jaccard similarity between the sets of the top-k_{top} ranked instances of f_i and f_j. We use the maximum (rather than average) similarity to prevent selecting near-duplicates of any existing ensemble member, which is particularly harmful without validation labels.

The discount $\gamma(\cdot)$ induces diminishing returns in the following sense: since $\text{sim}_{\max}(f_i, P \cup \{f\}) \geq \text{sim}_{\max}(f_i, P)$ for any added model f, the redundancy penalty is non-decreasing with ensemble growth, dampening the marginal utility of redundant candidates. We treat this as a submodular-inspired regularization rather than a strict theoretical guarantee.

Family-risk regularization. To reduce downside risk, we introduce a family-level prior that penalizes candidates from algorithmic families with negative lower-tail historical gains. Let $\mathcal{F} : \Omega \to \mathcal{F}$ map each candidate to an algorithm family (fixed from metadata; details in the Appendix). For a family $F \in \mathcal{F}$, we define the family risk as the 10th percentile of oracle gains observed during meta-training:

$$\text{Risk}_F(\mathcal{M}) = Q_{0.10}\left(\{G(f \mid P) \mid f \in \mathcal{M}_F,\ |P| \geq 1\}\right), \quad (8)$$

and convert it into a non-negative penalty:

$$\pi_F = \max\left(0, -\text{Risk}_F(\mathcal{M})\right) \quad (9)$$

Here, $\mathcal{M}_F$ collects candidates in family F across all meta-datasets and all ensemble states encountered during meta-training. Intuitively, if a family occasionally produces strongly negative marginal gains, it receives a higher penalty. This is to avoid harmful additions in the absence of labels. For previously unseen families ($\mathcal{M}_F = \emptyset$), we set $\text{Risk}_F(\mathcal{M}) = 0$ and thus $\pi_F = 0$, i.e., unknown families are not penalized.

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

3.4. Framework Algorithm

The detailed framework algorithm is provided in App. A.2, where we present the offline meta-training and online model selection procedure in pseudocode form.

3.5. Complexity Analysis

We conduct a computational complexity analysis to clarify how the proposed framework scales with the size of the candidate model pool and dataset size. The detailed computational complexity analysis is given in App. A.3.

4. Experiments

4.1. Experimental Settings

Datasets. We use the 39-dataset benchmark introduced in ELECT (Zhao et al., 2022), which consists of independent, real-world tabular datasets drawn from the Outlier Detection Data Sets (ODDS)[1] and UCI Machine Learning Repository (Campos et al., 2016) repositories. The datasets vary substantially, with the number of samples n ranging from 129 to 49097 (median $n = 1456$) and the dimensionality d ranging from 4 to 400 (median $d = 21$). Contamination rates range from 0.03% to 35% (median: 2.3%). The collection covers three data-type categories: numeric-only (18 datasets), categorical-only (2 datasets), and mixed-type (19 datasets). Details of the datasets are provided in App. A.4

Candidate Model Pool. We construct a large candidate pool of 297 unsupervised outlier detection models spanning 8 widely-used algorithmic families: Isolation Forest (IForest), Local Outlier Factor (LOF), k-Nearest Neighbors (kNN), Histogram-based Outlier Score (HBOS), One-Class SVM (OCSVM), COPOD, LODA, and COF. This diverse pool is generated by systematically varying hyperparameters within each family. For instance, we vary k among $\{5, 10, 15, \ldots, 100\}$ for kNN and LOF, contamination factors for IForest, and kernel types for OCSVM. The full specification of the algorithms and hyperparameter grids is detailed in App. A.5.

Baselines. We compare MetaEns against 16 unsupervised baselines, spanning standard detectors, deep learning methods, and meta-learning or ensemble-based selectors. For ensemble baselines that depend on an ensemble size parameter k, we report the best-performing k (selected to maximize average AP) to ensure fair comparisons.

The baselines include standard methods such as random selection (Singleton), classical detectors (e.g., LOF); naïve and random ensemble methods (e.g., IForest, Random Ensemble, Mega Ensemble, RandNet (Chen et al., 2017)); deep learning approaches (e.g., RDA (Zhou & Paffenroth, 2017), DAGMM (Zong et al., 2018), DeepSVDD) (Ruff et al., 2018); and recent meta-learning or ensemble-based selectors, including MetaOD (Zhao et al., 2020), LSCP (Zhao et al., 2019a), and the state-of-the-art ELECT (Zhao et al., 2022). We further consider ensemble variants built on ELECT (top-k aggregation and random expansion) to assess whether simple rank-based or hybrid strategies suffice. Finally, we report a supervised greedy oracle, which iteratively selects models using ground-truth labels as an upper bound. Detailed descriptions are provided in App. A.6.

Hyperparameter & Implementation Details. Detailed hyperparameter settings are provided in App. A.7, and implementation details are included in App. A.8.

Metrics. We evaluate all methods using five metrics: Average Precision (AP), Average Rank (AR), ROC-AUC, Precision@n, and Max F1-score. Details of metrics are provided in App. A.9.

4.2. Main Results

Table 1 compares MetaEns with unsupervised baselines on 39 benchmark datasets. The per-dataset results are in App. A.10. Overall, MetaEns achieves the best performance across metrics. Notably, it consistently outperforms the strongest meta-learning baseline ELECT, despite both methods sharing the same primary detector, demonstrating the effectiveness of our sequential partner selection strategy.

Naïve aggregation strategies perform substantially worse. Methods that average many detectors (Mega Ensemble) or randomly form ensembles (Random Ensemble) fail to filter weak or redundant models, highlighting the necessity of informed selection. Similarly, single detectors and static meta-learners such as Global Best fall behind adaptive model selection methods. Deep learning baselines show limited effectiveness on these tabular anomaly detection tasks, reflecting the difficulty of learning robust representations from tabular data.

We also compare with ensemble variants built on ELECT. Simply aggregating top-ranked models provides only marginal improvements over the single-model selector, whereas MetaEns's context-aware expansion yields clear gains. Across additional evaluation metrics, MetaEns remains among the top performers while using compact, adaptive ensembles, demonstrating both effectiveness and efficiency.

4.3. Ablation Study

To understand the contribution of each component in MetaEns, we perform ablations by selectively disabling key mechanisms. Table 2 shows the ablation study. The full model performs best overall.

[1] https://giftpathao.com

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Table 1. Performance comparison on the 39 datasets. Best results are in **bold**; second-best are underlined.

Method	AP ↑	Rank ↓	ROC-AUC ↑	Prec@n ↑	Prec@k* ↑	Max-F1 ↑	Ens Size
Theoretical Upper Bound							
Greedy Oracle	0.6877	1.0	0.8968	0.8333	0.6504	0.6906	10
Single Model Baselines							
`Singleton`	0.3495 ± 0.0255	147.9 ± 22.0578	0.7081 ± 0.0248	0.4287 ± 0.0426	0.3304 ± 0.0283	0.4075 ± 0.0190	1
`LOF`	0.3513 ± 0.0038	120.1 ± 2.2336	0.7439 ± 0.0063	0.4487 ± 0.0075	0.3297 ± 0.0037	0.4169 ± 0.0051	1
`Global Best`	0.3787 ± 0.0074	122.7 ± 9.0314	0.7583 ± 0.0049	0.4513 ± 0.0162	0.3593 ± 0.0093	0.4266 ± 0.0075	1
Naïve & Random Ensembles							
`IForest`	0.3858 ± 0.0016	117.0 ± 5.8119	0.7699 ± 0.0018	0.4733 ± 0.0092	0.3619 ± 0.0033	0.4337 ± 0.0022	200
`RandNet`	0.3460 ± 0.0018	170.6 ± 2.9136	0.6865 ± 0.0029	0.4574 ± 0.0035	0.3189 ± 0.0034	0.3927 ± 0.0027	20
`Mega Ensemble`	0.3970 ± 0.0000	100.0 ± 0.0000	0.7737 ± 0.0000	0.4872 ± 0.0000	0.3782 ± 0.0000	0.4443 ± 0.0000	297
`Random Ensemble`	0.3759 ± 0.0175	124.3 ± 13.6874	0.7477 ± 0.0106	0.4597 ± 0.0281	0.3608 ± 0.0172	0.4290 ± 0.0140	3
Deep Learning Baselines							
`RDA`	0.2742 ± 0.0065	211.9 ± 11.4741	0.7063 ± 0.0064	0.3338 ± 0.0182	0.2721 ± 0.0090	0.3515 ± 0.0068	1
`DAGMM`	0.2958 ± 0.0124	221.6 ± 6.8508	0.6676 ± 0.0134	0.3872 ± 0.0193	0.3013 ± 0.0135	0.3681 ± 0.0129	1
`DeepSVDD`	0.2073 ± 0.0115	247.5 ± 6.8516	0.5905 ± 0.0164	0.3254 ± 0.0225	0.2116 ± 0.0147	0.2968 ± 0.0073	1
Meta-Learning & Ensemble Methods							
`LSCP`	0.3484 ± 0.0173	124.3 ± 11.1161	0.7560 ± 0.0096	0.4638 ± 0.0136	0.3441 ± 0.0208	0.4251 ± 0.0123	1
`MetaOD`	0.3989 ± 0.0024	101.0 ± 7.6594	0.7547 ± 0.0014	0.4792 ± 0.0050	0.3746 ± 0.0032	0.4392 ± 0.0022	1
`ELECT`+Random	0.3981 ± 0.0060	102.2 ± 9.2232	0.7719 ± 0.0068	0.4864 ± 0.0192	0.3778 ± 0.0103	0.4449 ± 0.0067	10
`ELECT` (Top-1)	0.4069 ± 0.0063	85.8 ± 7.8712	0.7734 ± 0.0046	0.5167 ± 0.0085	0.3861 ± 0.0059	0.4519 ± 0.0051	1
`ELECT` (Top-10)	0.4117 ± 0.0050	83.2 ± 6.8118	0.7785 ± 0.0038	0.5213 ± 0.0088	0.3856 ± 0.0055	0.4546 ± 0.0043	10
`MetaEns` (**Ours**)	**0.4308** ± 0.0064	**59.3** ± 6.9610	**0.7867** ± 0.0045	**0.5479** ± 0.0077	**0.4042** ± 0.0063	**0.4681** ± 0.0069	2.2

Table 2. Ablation study on the 39 datasets. ΔAP measures the performance degradation relative to the full model.

Variant	AP ↑	Rank ↓	ΔAP
w/o diversity (β=0)	0.4185	77	−0.0169
w/o family-risk (λ_{fam}=0)	0.3995	72	−0.0359
single-part gain model	0.4133	87	−0.0221
`MetaEns`	**0.4354**	**56**	—

Removing β shows a performance decrease, as β controls intra-family similarity while λ_{fam} controls family-level risk. Removing family-risk regularization ($\lambda_{\text{fam}} = 0$) causes the largest performance drop, showing that family-aware risk control is the most critical component for robust ensemble construction.

Replacing the two-part meta-model with a single gain predictor leads to a consistent but moderate degradation, supporting our design choice of separating improvement probability and magnitude. Notably, even this simplified variant remains competitive with strong single-model selectors, indicating that the sequential selection framework itself is effective, with the two-part architecture providing further gains.

4.4. Statistical Significance

To validate the performance improvements of `MetaEns`, we conduct paired Wilcoxon signed-rank tests comparing our method against all baselines across the 39 datasets. `MetaEns` achieves statistically significant improvements over all baselines. The details of statistical significance analysis are provided in App. A.11.

4.5. Robustness to Initialization

A central question is whether `MetaEns`'s gains depend on a specific primary selector or on strong initial models. To investigate this, we evaluate `MetaEns` under diverse initialization strategies and analyze its behavior when the starting model performs poorly. We pair `MetaEns` with heterogeneous primary selectors, including the meta-learning method `ELECT`, density-based method `LOF`, random ensemble method `IForest`, and a random singleton baseline. For each configuration, we compare the primary selector's AP with the final AP achieved by `MetaEns`. Detailed scatter plots and per-selector analyses are shown in Fig. 2. Across all selectors, `MetaEns` consistently improves over the starting model, demonstrating that its partner selection mechanism is selector-agnostic and not tied to a particular initialization strategy. Improvements are especially pronounced in challenging cases where the primary model underperforms. Rather than propagating initial errors, `MetaEns` effectively recovers performance by selecting complementary detectors from diverse algorithmic families. Even when initialized with weak or biased selectors, the method constructs strong ensembles, indicating that gains stem from diversity-aware partner selection rather than reliance on the primary model quality.

4.6. Model Diversity Analysis

To illustrate the difference between `MetaEns` and baseline selection strategies, we project the 297-model candidate pool into a two-dimensional space using `t-SNE` (van der Maaten & Hinton, 2008) based on prediction-correlation distance. This embedding reveals that models naturally cluster

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

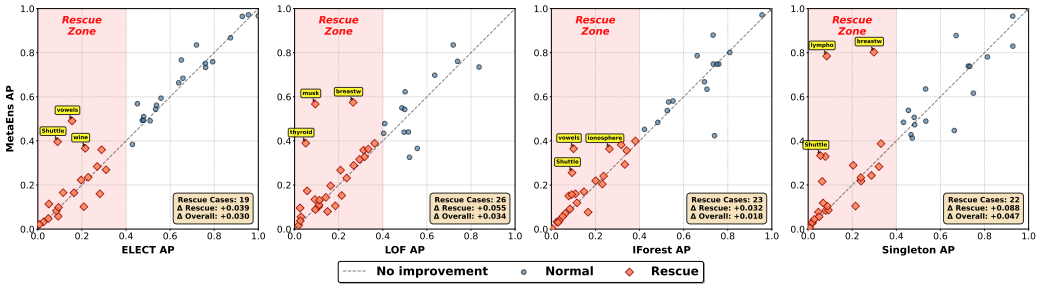

Figure 2. Robustness analysis across four different primary selectors: `ELECT`, `LOF`, `HBOS`, and `Random Selection`. Each panel compares the primary model's performance (x-axis) against the final `MetaEns` ensemble (y-axis). Points above the diagonal indicate improvement. The shaded red "Rescue Zone" highlights where the primary model fails (AP < 0.4). `MetaEns` consistently rescues performance in these failure modes across all primary selectors, demonstrating that its diverse partner selection logic is robust and selector-agnostic.

by algorithmic family, as detectors within the same family tend to produce similar outlier rankings. Figure 3 shows the resulting visualization on four representative datasets (**Speech**, **WBC**, **Waveform**, and **Shuttle**), overlaid with the models selected in our experiments. Background points correspond to all 297 detectors, colored by family, and exhibit clear family-level clustering. In all cases, `ELECT-10` selects models concentrated within a single family cluster (e.g., `HBOS` for **Speech** and `IForest` for **WBC**), indicating limited diversity. In contrast, `MetaEns` selects models distributed across multiple distinct family clusters. For example, on **Speech**, selections span four families (`HBOS`, `IForest`, `OCSVM`, `ABOD`), while on **WBC**, selections cover three (`IForest`, `OCSVM`, `HBOS`). This visualization supports the observation that `MetaEns` does not rely solely on the initial model chosen by `ELECT`. Although both methods share the same primary detector, `MetaEns`'s family-risk regularization encourages selection of complementary models from different algorithm families. The dispersion of selected detectors suggests that the framework captures diverse decision patterns, which is particularly important in unsupervised settings where validation labels are unavailable.

4.7. Pool Size Analysis

A detailed analysis of the effect of candidate pool size is provided in App. A.12. The study shows how `MetaEns`'s performance evolves as the number of available detectors increases, demonstrating that the method scales well and remains stable while benefiting from greater model diversity.

4.8. Effects of Meta-model Architectures

We study the choice of meta-model architectures in App. A.13. The comparison evaluates several alternative learning families for marginal gain prediction and candidate filtering, showing that the default `ExtraTrees` configuration achieves the best balance between predictive performance and computational efficiency.

4.9. Effect of Expanded Model Pools

To evaluate `MetaEns`' scalability and robustness to model pool composition, we conduct additional experiments with an expanded pool of 310 models that incorporates neural networks. A detailed analysis of the effect of expanded model pool is provided in App. A.14.

4.10. Effect of Ensemble Size

To understand how ensemble size affects performance, we analyze the relationship between the number of models and detection quality. A detailed analysis of the effect of ensemble size is provided in App. A.15.

5. Related Work

5.1. Unsupervised Outlier Detection

Unsupervised outlier detection has been studied across diverse methodological paradigms. Statistical methods such as `HBOS` (Aryal et al., 2021) identify anomalies in low-density regions under feature independence assumptions. Density- and distribution-based methods, including `LOF` (Breunig et al., 2000), `COPOD` (Li et al., 2020), and `ECOD` (Li et al., 2023), quantify deviations using local density or tail probabilities of empirical distributions. Tree-based methods such as `IForest` (Liu et al., 2008) exploit random partitioning to isolate anomalies, while distance-based methods including `kNN` (Chehreghani, 2016) and `ODIN` (Hautamäki et al., 2004) detect outliers via spatial isolation. One-class classifiers, such as `OCSVM` (Schölkopf et al., 1999) and `SVDD` (Tax & Duin, 2004), learn decision boundaries enclosing normal data, whereas subspace-based methods like `PCA` (Chapel & Friguet, 2014) rely on reconstruction errors. More recently, deep learning–based methods learn expressive representations of normality. These include autoencoders (`AE`s) (Goodge et al., 2020), variational autoencoders (`VAE`s) (Xu et al., 2018), and generative adversarial networks (`GAN`s) (Lim et al., 2018). Extensions such as `Deep SVDD` (Ruff et al., 2018), `DAGMM` (Zong et al., 2018), and `RDA` (Zhou & Paffenroth, 2017) integrate rep-

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

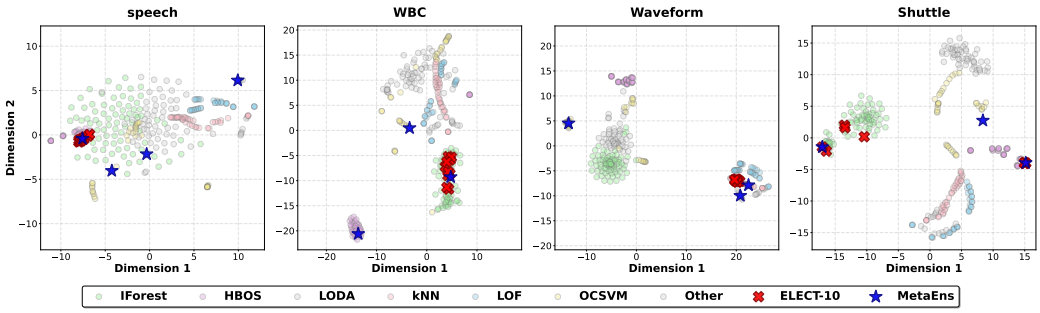

Figure 3. Model diversity visualization using `t-SNE` projection across four datasets. `ELECT-10` selections tend to cluster within a single family, whereas `MetaEns` selects models spanning multiple families, indicating greater ensemble diversity.

resentation learning with one-class objectives or density estimation to improve robustness. Transformer-based models (Kim et al., 2024) have also been employed to capture complex dependencies. Despite their effectiveness, these methods are often sensitive to architectural choices and hyperparameters, motivating ensemble-based strategies to improve generalization.

5.2. Ensembles for Outlier Detection

Ensemble methods aim to improve robustness and stability by combining multiple detectors (Aggarwal & Sathe, 2015). Common aggregation strategies include score averaging, ranking, and voting; however, naively combining all detectors often incurs high computational cost, performance degradation due to weak models, and limited adaptivity. Several ensemble designs address these issues implicitly. Feature-bagging ensembles promote diversity through random subspaces (Noto et al., 2010; Lazarevic & Kumar, 2005), while tree-based ensembles such as `IForest` (Liu et al., 2008) embed diversity via randomized construction. Sequential ensembles refine performance by reweighting instances across rounds, as exemplified by `XGBOD` (Zhao & Hryniewicki, 2018). Stacking-based approaches combine heterogeneous detectors, including autoencoder (Chen et al., 2017) and `GAN` ensembles (Han et al., 2021) , to preserve complementary behaviors. Yet, most existing methods rely on fixed aggregation schemes or require constructing the full ensemble upfront, limiting adaptivity in unsupervised settings. Unlike existing ensembles that require training data, `MetaEns` operates fully unsupervised and focuses on model selection rather than score reweighting.

5.3. Model Selection

Model selection is well established in supervised learning via information-theoretic criteria such as AIC and BIC (Schwarz, 1978), cross-validation (Stone, 1974), structural risk minimization (Vapnik, 1998), as well as modern hyperparameter optimization techniques including Bayesian optimization (Snoek et al., 2012) and automated machine learning frameworks (Feurer et al., 2015). In contrast, unsupervised outlier model selection is considerably more challenging due to the lack of labels, extreme class imbalance, and heterogeneous anomaly patterns. Early approaches rely on internal validation heuristics derived from score distributions or stability under perturbations (Goix, 2016; Marques et al., 2020), which often correlate weakly with true detection performance. Recent studies explore meta-learning approaches (Vilalta & Drissi, 2002; Hospedales et al., 2022) that leverage labeled meta-datasets to generalize model selection across tasks. Representative methods include `MetaOD` (Zhao et al., 2020) and `ELECT` (Zhao et al., 2022), which focus primarily on selecting individual detectors. Ensemble-based model selection remains underexplored: `LSCP` (Zhao et al., 2019a) performs instance-wise detector selection but assumes local data consistency and requires constructing the full ensemble in advance. In contrast, the proposed `MetaEns` framework performs sequential ensemble model selection without assuming local consistency and supports adaptive early stopping, enabling efficient construction of compact, high-performing ensembles in fully unsupervised settings.

6. Conclusion

We address the problem of unsupervised ensemble outlier model selection, where ensembles must be constructed without access to labels. We propose `MetaEns`, a meta-learned framework that predicts marginal ensemble gains and combines these with a submodular-inspired proxy objective to guide adaptive ensemble construction with early stopping. Extensive experiments on 39 real-world datasets show that `MetaEns` can consistently outperform strong unsupervised baselines while selecting substantially smaller ensembles, highlighting the importance of family-level diversification and risk control in unsupervised settings.

Future work includes exploring richer meta-representations to improve gain prediction under distribution shift, extending the framework to streaming or non-stationary data, and developing theoretical or uncertainty-aware variants of the proxy objective to better understand unsupervised ensemble selection.

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Broader Impact

Unsupervised outlier detection is used widely in high-impact domains such as fraud detection, cybersecurity, healthcare monitoring, and scientific data analysis. By enabling adaptive and data-driven ensemble construction without requiring labeled data, `MetaEns` has the potential to improve the robustness and reliability of outlier detection systems deployed in practice. In particular, the ability to construct compact ensembles can reduce computational cost and energy consumption, which is beneficial for large-scale or resource-constrained settings.

Limitations

This work has several limitations. First, `MetaEns` relies on labeled meta-datasets to learn transferable patterns of marginal ensemble gain. While our experiments suggest strong generalization across diverse datasets, performance may degrade if test tasks differ substantially from the meta-training distribution. Second, the family-risk regularization depends on a predefined mapping of detectors to algorithmic families, which may be coarse and require domain knowledge to define for new detectors. Finally, we focus on batch settings and do not address streaming or non-stationary data, which we leave for future work.

References

Aggarwal, C. C. and Sathe, S. Theoretical foundations and algorithms for outlier ensembles. *SIGKDD Explor.*, 17: 24–47, 2015.

Aryal, S., Baniya, A. A., Razzak, I., and Santosh, K. SPAD+: an improved probabilistic anomaly detector based on one-dimensional histograms. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, 2021.

Breiman, L. Random forests. *Mach. Learn.*, 45(1):5–32, 2001.

Breunig, M. M., Kriegel, H., Ng, R. T., and Sander, J. LOF: identifying density-based local outliers. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pp. 93–104, 2000.

Campos, G. O., Zimek, A., Sander, J., Campello, R. J. G. B., Micenková, B., Schubert, E., Assent, I., and Houle, M. E. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Min. Knowl. Discov.*, 30:891–927, 2016.

Chandola, V., Banerjee, A., and Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, 2009.

Chapel, L. and Friguet, C. Anomaly detection with score functions based on the reconstruction error of the kernel PCA. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, pp. 227–241, 2014.

Chehreghani, M. H. K-nearest neighbor search and outlier detection via minimax distances. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pp. 405–413, 2016.

Chen, J., Sathe, S., Aggarwal, C. C., and Turaga, D. S. Outlier detection with autoencoder ensembles. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pp. 90–98, 2017.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pp. 785–794, 2016.

Cheng, L., Wang, Y., Liu, X., and Li, B. Outlier detection ensemble with embedded feature selection. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 3503–3512, 2020.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J. T., Blum, M., and Hutter, F. Efficient and robust automated machine learning. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2962–2970, 2015.

Geurts, P., Ernst, D., and Wehenkel, L. Extremely randomized trees. *Mach. Learn.*, 63(1):3–42, 2006.

Goix, N. How to evaluate the quality of unsupervised anomaly detection algorithms? *CoRR*, abs/1607.01152, 2016.

Goldstein, M. and Dengel, A. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012: poster and demo track*, 1:59–63, 2012.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. ISBN 978-0-2620-3561-3.

Goodge, A., Hooi, B., Ng, S., and Ng, W. S. Robustness of autoencoders for anomaly detection under adversarial impact. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1244–1250, 2020.

Han, S., Hu, X., Huang, H., Jiang, M., and Zhao, Y. Adbench: Anomaly detection benchmark. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Han, X., Chen, X., and Liu, L. GAN ensemble for anomaly detection. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 4090–4097, 2021.

Hautamäki, V., Kärkkäinen, I., and Fränti, P. Outlier detection using k-nearest neighbour graph. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pp. 430–433, 2004.

Hospedales, T. M., Antoniou, A., Micaelli, P., and Storkey, A. Meta-learning in neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(9):5149–5169, 2022.

James, G., Witten, D., Hastie, T., and Tibshirani, R. *An Introduction to Statistical Learning–with Applications in R*. Springer, 2013. ISBN 978-1-4614-7137-0.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T. Lightgbm: A highly efficient gradient boosting decision tree. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3146–3154, 2017.

Kim, H., Lee, C. H., and Hong, C. Transformer for point anomaly detection. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 1080–1088, 2024.

Kriegel, H., Schubert, M., and Zimek, A. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, pp. 444–452, 2008.

Kulesza, A. and Taskar, B. Determinantal point processes for machine learning. *Found. Trends Mach. Learn.*, 5 (2-3):123–286, 2012.

Lazarevic, A. and Kumar, V. Feature bagging for outlier detection. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pp. 157–166, 2005.

Li, Z., Zhao, Y., Botta, N., Ionescu, C., and Hu, X. COPOD: copula-based outlier detection. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pp. 1118–1123, 2020.

Li, Z., Zhao, Y., Hu, X., Botta, N., Ionescu, C., and Chen, G. H. ECOD: unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Trans. Knowl. Data Eng.*, 35(12):12181–12193, 2023.

Lim, S. K., Loo, Y., Tran, N., Cheung, N., Roig, G., and Elovici, Y. DOPING: generative data augmentation for unsupervised anomaly detection with GAN. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pp. 1122–1127, 2018.

Liu, F. T., Ting, K. M., and Zhou, Z. Isolation forest. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pp. 413–422, 2008.

Liu, Y., Li, Z., Zhou, C., Jiang, Y., Sun, J., Wang, M., and He, X. Generative adversarial active learning for unsupervised outlier detection. *IEEE Trans. Knowl. Data Eng.*, 32:1517–1528, 2020.

Marques, H. O., Campello, R. J. G. B., Sander, J., and Zimek, A. Internal evaluation of unsupervised outlier detection. *ACM Trans. Knowl. Discov. Data*, 14(4):47:1–47:42, 2020.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. An analysis of approximations for maximizing submodular set functions-I. *Math. Program.*, 14(1):265–294, 1978.

Noto, K., Brodley, C. E., and Slonim, D. K. Anomaly detection using an ensemble of feature models. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pp. 953–958, 2010.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E. Z., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8024–8035, 2019.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., VanderPlas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830, 2011.

Pevný, T. Loda: Lightweight on-line detector of anomalies. *Mach. Learn.*, 102(2):275–304, 2016.

Rayana, S., Zhong, W., and Akoglu, L. Sequential ensemble learning for outlier detection: A bias-variance perspective. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pp. 1167–1172, 2016.

Ruff, L., Görnitz, N., Deecke, L., Siddiqui, S. A., Vandermeulen, R. A., Binder, A., Müller, E., and Kloft, M. Deep one-class classification. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 4390–4399, 2018.

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., and Müller, K. A unifying review of deep and shallow anomaly detection. *Proc. IEEE*, 109(5):756–795, 2021.

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., and Platt, J. C. Support vector method for novelty detection. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 582–588, 1999.

Schwarz, G. Estimating the dimension of a model. *Ann. Stat.*, 6(2):461–464, 1978.

Snoek, J., Larochelle, H., and Adams, R. P. Practical bayesian optimization of machine learning algorithms. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2960–2968, 2012.

Stone, M. Cross-validatory choice and assessment of statistical predictions. *J. R. Stat. Soc., B: Stat. Methodol.*, 36 (1):111–147, 1974.

Tang, J., Chen, Z., Fu, A. W., and Cheung, D. W. Enhancing effectiveness of outlier detections for low density patterns. In *Advances in Knowledge Discovery and Data Mining, 6th Pacific-Asia Conference, PAKDD 2002, Taipei, Taiwan, May 6-8, 2002, Proceedings*, pp. 535–548, 2002.

Tax, D. M. J. and Duin, R. P. W. Support vector data description. *Mach. Learn.*, 54(1):45–66, 2004.

van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *J. Mach. Learn. Res.*, 9(86):2579–2605, 2008.

Vapnik, V. *Statistical Learning Theory*. Wiley, 1998. ISBN 978-0-471-03003-4.

Vilalta, R. and Drissi, Y. A perspective view and survey of meta-learning. *Artif. Intell. Rev.*, 18(2):77–95, 2002.

Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., Liu, Y., Zhao, Y., Pei, D., Feng, Y., Chen, J., Wang, Z., and Qiao, H. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the World Wide Web Conference on World Wide Web (WWW)*, pp. 187–196, 2018.

Zhao, Y. and Hryniewicki, M. K. XGBOD: improving supervised outlier detection with unsupervised representation learning. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2018.

Zhao, Y., Nasrullah, Z., Hryniewicki, M. K., and Li, Z. LSCP: locally selective combination in parallel outlier ensembles. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pp. 585–593, 2019a.

Zhao, Y., Nasrullah, Z., and Li, Z. Pyod: A python toolbox for scalable outlier detection. *J. Mach. Learn. Res.*, 20: 96:1–96:7, 2019b.

Zhao, Y., Rossi, R. A., and Akoglu, L. Automating outlier detection via meta-learning. *CoRR*, abs/2009.10606, 2020.

Zhao, Y., Rossi, R. A., and Akoglu, L. Automatic unsupervised outlier model selection. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4489–4502, 2021.

Zhao, Y., Zhang, S., and Akoglu, L. Toward unsupervised outlier model selection. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pp. 773–782, 2022.

Zhou, C. and Paffenroth, R. C. Anomaly detection with robust deep autoencoders. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pp. 665–674, 2017.

Zimek, A., Campello, R. J. G. B., and Sander, J. Ensembles for unsupervised outlier detection: challenges and research questions a position paper. *SIGKDD Explor.*, 15 (1):11–22, 2013.

Zong, B., Song, Q., Min, M. R., Cheng, W., Lumezanu, C., Cho, D., and Chen, H. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

A. Appendix

A.1. State Representation Features

State Representation. The state representation $\phi(f_i, f^*_{i-1}, P)$ is a fixed 61-dimensional feature vector that encodes the interaction between a candidate model f_i, the most recently selected detector f^*_{i-1}, and the current ensemble P. It consists of three 20-dimensional feature blocks and one scalar feature, namely: (i) $\phi_{f^*_{i-1},f_i}$, capturing pairwise interactions between the last selected model and the candidate; (ii) $\phi_{f^*_{i-1},P}$, summarizing interactions between the last selected model and the existing ensemble; (iii) $\phi_{f_i,P}$, summarizing interactions between the ensemble and the candidate; and (iv) $|P|$, the current ensemble cardinality.

Each 20-dimensional block is composed of 15 base pairwise features and 5 context features, computed from normalized outlier score vectors $\mathbf{o}_f \in [0,1]^N$, where scores are Min–Max normalized independently for each detector and dataset. Table A.1 provides the complete specification of all feature dimensions.

Feature Aggregation for Ensemble Context. For ensembles with $|P| > 1$, set-level context features are computed

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Table A.1. Feature specification for pairwise model comparison. All features are computed from normalized outlier score vectors $\mathbf{o}_p, \mathbf{o}_q \in [0,1]^N$.

Feature	Type	Mathematical Definition
Base Pairwise Features (15 dimensions)		
`pearson`	Correlation	Pearson correlation: $\rho(\mathbf{o}_p, \mathbf{o}_q) = \frac{\text{cov}(\mathbf{o}_p, \mathbf{o}_q)}{\sigma_p \sigma_q}$
`tail_pearson`	Correlation	Pearson correlation on $\mathcal{U} = \text{top-}k_p \cup \text{top-}k_q$ where $k = \lceil 0.1N \rceil$
`jaccard`	Set similarity	Jaccard index: $J = \lvert\text{top-}k_p \cap \text{top-}k_q\rvert / \lvert\text{top-}k_p \cup \text{top-}k_q\rvert$
`rel_kurtosis`	Moment ratio	Relative kurtosis: $\text{kurt}(\mathbf{o}_q) / \max(\text{kurt}(\mathbf{o}_p), 10^{-3})$
`tail_pos_disagreement`	Distributional	Mean positive difference: $\mathbb{E}_{i \in \mathcal{U}}[\max(0, \mathbf{o}_{q,i} - \mathbf{o}_{p,i})]$
`centrality`	Distributional	Correlation with pool mean: $\rho(\mathbf{o}_q, \bar{\mathbf{o}})$ where $\bar{\mathbf{o}} = \frac{1}{K}\sum_{j=1}^{K} \mathbf{o}_j$
`cand_std`	Distributional	Standard deviation: $\sigma(\mathbf{o}_q)$
`pseudo_ap`	Pseudo-label	Average Precision treating top-k_p as positive class, $\mathbf{o}_q$ as predictions
`pseudo_roc`	Pseudo-label	ROC AUC treating top-k_p as positive class, $\mathbf{o}_q$ as predictions
`tail_entropy`	Distributional	Shannon entropy: $H(\mathbf{o}_q) = -\sum_{b=1}^{10} p_b \log p_b$ where p_b is 10-bin histogram
`score_dist_l2`	Distance	L_2 distance of sorted vectors: $\lVert\text{sort}(\mathbf{o}_p) - \text{sort}(\mathbf{o}_q)\rVert_2$
`cosine_dist`	Distance	Cosine distance: $1 - \frac{\mathbf{o}_p \cdot \mathbf{o}_q}{\lVert\mathbf{o}_p\rVert_2 \lVert\mathbf{o}_q\rVert_2}$
`tail_divergence`	Distance	Mean absolute difference: $\mathbb{E}_{i \in \mathcal{U}}[\lvert\mathbf{o}_{q,i} - \mathbf{o}_{p,i}\rvert]$
`same_family`	Categorical	Binary indicator: $\mathbb{I}[\mathcal{F}(p) = \mathcal{F}(q)]$ where $\mathcal{F}$ maps models to families
`same_as_ensemble_count`	Set overlap	Count of partners $s \in S$ with $J(q,s) > 0.5$; equals 0 when $S = \emptyset$
Primary-Context Features (5 dimensions)		
`prim_std`	Distributional	Standard deviation of primary: $\sigma(\mathbf{o}_p)$
`prim_entropy`	Distributional	Shannon entropy of primary score distribution: $H(\mathbf{o}_p)$
`prim_centrality`	Distributional	Centrality of primary: $\rho(\mathbf{o}_p, \bar{\mathbf{o}})$
`prim_kurtosis`	Moment	Excess kurtosis of primary: $\text{kurt}(\mathbf{o}_p)$
`prim_skewness`	Moment	Skewness of primary: $\text{skew}(\mathbf{o}_p)$
Set-Level Scalar Features (1 dimension)		
$\lvert P\rvert$	Cardinality	Number of partners: $\lvert S\rvert \in \{0, 1, 2, \ldots\}$

via mean pooling to ensure permutation invariance:

$$\phi_{f_{i-1}^*, P} = \frac{1}{|P| - 1} \sum_{f \in P \setminus \{f_{i-1}^*\}} \phi_{f_{i-1}^*, f}$$

$$\phi_{f_i, P} = \frac{1}{|P|} \sum_{f \in P} \phi_{f, f_i}$$

Here, the first equation measures the variance between the last selected model and the current ensemble. The second equation measures the variance between the candidate model and the current ensemble. When $|P| = 1$ (i.e., only the primary detector has been selected), $\phi_{f_{i-1}^*, P}$ is zero-padded, while $\phi_{f_i, P}$ is computed directly from the primary detector. This design maintains a fixed 61-dimensional representation across all selection stages and allows the meta-model to distinguish early-stage expansion from later iterative selection.

A.2. Framework Algorithm

Offline Meta-training Procedure. Algorithm 1 summarizes the offline meta-training phase of `MetaEns`. Given labeled meta-datasets $\mathcal{M} = \{(\mathbf{M}_\ell, \mathbf{y}_\ell)\}$ and a detector pool Ω, we simulate oracle greedy ensemble construction by iteratively expanding an ensemble $P \subseteq \Omega$. Starting from an oracle-selected primary detector f_1^*, at each step we evaluate every candidate $f_i \in \Omega \setminus P$ using its state representation $\phi(f_i, f_{i-1}^*, P)$ and compute the true marginal gain $G(f_i \mid P)$ in Average Precision. These state–gain pairs are used to train a two-part meta-model that estimates the probability and magnitude of marginal improvement, yielding a predictor $\hat{G}(f_i \mid P)$ that is later used to guide label-free online ensemble selection.

Online Model Selection Procedure. Algorithm 2 describes the online ensemble selection procedure of `MetaEns` for a new unlabeled dataset. Starting from a primary detector f_1^* selected by an unsupervised criterion, we iteratively construct an ensemble P by adding candidates $f_i \in \Omega \setminus P$ that maximize the proxy marginal utility $\Delta U(f_i \mid P)$. The proxy utility combines the predicted marginal gain $\hat{G}(f_i \mid P)$ from the meta-model with redundancy discounting and family-risk regularization, thereby enforcing diminishing returns as the ensemble grows. The selection process terminates automatically when $\max_{f_i} \Delta U(f_i \mid P) \leq 0$ or when the budget $|P| = \eta$ is reached, yielding compact, dataset-adaptive ensembles without access to labels.

A.3. Complexity Analysis

We conduct a computational complexity analysis to clarify how the proposed framework scales with the size of the candidate model pool and dataset size.

`MetaEns` operates in $O(|\Omega|^2 \cdot N + \eta|\Omega| \cdot C_{\text{inf}})$ time, where $|\Omega|$ is the size of the candidate model pool, N is the number of samples in the target dataset, η is the ensemble budget, and C_{inf} is the meta-model inference cost. The $O(|\Omega|^2 \cdot$

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Algorithm 1 `MetaEns` Offline Meta-training: Oracle Rollouts for Marginal-Gain Supervision

Require: Labeled meta-datasets $\mathcal{M} = \{(\mathbf{M}_\ell, \mathbf{y}_\ell)\}_{\ell=1}^{L}$; detector pool Ω; budget η; rollout length $T \leq \eta$; state function $\phi(f_i, f_{i-1}^*, P)$
Ensure: Trained meta-models $(f_{\text{cls}}, f_{\text{reg}})$

Initialize $\mathcal{D}_{\text{cls}} \leftarrow \emptyset$, $\mathcal{D}_{\text{reg}} \leftarrow \emptyset$
for all $(\mathbf{M}, \mathbf{y}) \in \mathcal{M}$ **do**
  (Oracle primary)
  $f_1^* \leftarrow \arg\max_{f \in \Omega}\ \text{AP}(f(\mathbf{M}), \mathbf{y})$
  $P \leftarrow \{f_1^*\}$
  $f_{\text{last}} \leftarrow f_1^*$
  for $t = 2$ to T **do**
    for all $f_i \in \Omega \setminus P$ **do**
      $\boldsymbol{\phi}_i \leftarrow \phi(f_i, f_{\text{last}}, P)$
      $G(f_i \mid P) \leftarrow \text{AP}(\mathbf{o}_{P \cup \{f_i\}}, \mathbf{y}) - \text{AP}(\mathbf{o}_P, \mathbf{y})$
      $y_{\text{cls}} \leftarrow \mathbb{I}(G(f_i \mid P) > 0)$
      $\mathcal{D}_{\text{cls}} \leftarrow \mathcal{D}_{\text{cls}} \cup \{(\boldsymbol{\phi}_i, y_{\text{cls}})\}$
      if $y_{\text{cls}} = 1$ **then**
        $y_{\text{reg}} \leftarrow G(f_i \mid P)$
        $\mathcal{D}_{\text{reg}} \leftarrow \mathcal{D}_{\text{reg}} \cup \{(\boldsymbol{\phi}_i, y_{\text{reg}})\}$
      end if
    end for
    $f^* \leftarrow \arg\max_{f_i \in \Omega \setminus P}\ G(f_i \mid P)$
    if $G(f^* \mid P) \leq 0$ **then**
      break
    end if
    $P \leftarrow P \cup \{f^*\}$; $f_{\text{last}} \leftarrow f^*$
    if $|P| \geq \eta$ **then**
      break
    end if
  end for
end for
Train classifier f_{cls} on $\mathcal{D}_{\text{cls}}$ (balanced weights)
Train regressor f_{reg} on $\mathcal{D}_{\text{reg}}$ (positives only)
return $(f_{\text{cls}}, f_{\text{reg}})$

Algorithm 2 `MetaEns`: Online Selection via Submodular-Inspired Proxy Maximization

Require: dataset $\mathbf{X}$, candidate pool Ω, meta-model $M(\cdot)$ producing $\hat{G}$, state representation $\phi(\cdot)$, family mapping $\mathcal{F}$, thresholds τ_1, τ_2, budget $\eta \geq 1$, diversity β, risk λ_{fam}, k_{top}
Ensure: Ensemble P

Primary model selection using `ELECT` *as* $\mathcal{S}(\cdot)$
$f_1^* \leftarrow \mathcal{S}(\mathbf{X}, \Omega)$
$P \leftarrow \{f_1^*\}$
$\Omega_{\text{pool}} \leftarrow \Omega \setminus \{f_1^*\}$
Stage 1: Select first expansion model
for all $f_i \in \Omega_{\text{pool}}$ **do**
  $\hat{G}(f_i \mid P) \leftarrow M.\text{predict}(\phi(f_i, f_1^*, P))$
end for
$f_2^* \leftarrow \arg\max_{f_i \in \Omega_{\text{pool}}}\ \hat{G}(f_i \mid P)$
if $\hat{G}(f_2^* \mid P) < \tau_1$ **then**
  return P
end if
$P \leftarrow P \cup \{f_2^*\}$
Stage 2: Iterative expansion
while $|P| < \eta$ **do**
  $u_{\text{best}} \leftarrow 0$
  for all $f_i \in \Omega_{\text{pool}} \setminus P$ **do**
    $\hat{G}(f_i \mid P) \leftarrow M.\text{predict}\big(\phi(f_i, f_{|P|-1}^*, P)\big)$
    if $\hat{G}(f_i \mid P) < \tau_2$ **then**
      continue
    end if
    Compute $\Delta U(f_i \mid P)$
    if $\Delta U(f_i \mid P) > u_{\text{best}}$ **then**
      $f^* \leftarrow f_i$
      $u_{\text{best}} \leftarrow \Delta U(f_i \mid P)$
    end if
  end for
  if $u_{\text{best}} \leq 0$ **then**
    break
  end if
  $P \leftarrow P \cup \{f^*\}$
end while
return P

N) term corresponds to one-time feature pre-computation of similarity statistics for the target dataset: correlation-based features require $O(|\Omega|^2 \cdot N)$ operations, while Jaccard similarity over top-k_{top} instances requires $O(|\Omega| \cdot N)$ time for extracting top-k_{top} indices and $O(|\Omega|^2 \cdot k_{\text{top}})$ time for set intersections, becoming independent of N after ranking. The interactive selection phase has complexity $O(\eta|\Omega| \cdot C_{\text{inf}})$ and is independent of N. For tree-based meta-models, $C_{\text{inf}} = O(n_{\text{trees}} \cdot d_\phi)$, where n_{trees} is the number of trees and d_ϕ is the feature dimensionality.

A.4. Datasets

We conduct experiments on the 39-dataset benchmark introduced in `ELECT` (Zhao et al., 2022), which comprises independent, real-world tabular anomaly detection tasks sourced from the Outlier Detection Data Sets (ODDS)[2] and the UCI Machine Learning Repository (Campos et al., 2016). Detailed characteristics of all datasets used in our experiments are summarized in Table A.2.

[2]https://giftpathao.com

A.5. Candidate Model Pool

We construct a diverse candidate pool of $M = 297$ unsupervised outlier detection models by systematically varying hyperparameters within 8 widely-used algorithmic families. For each dataset, all 297 models are pre-fitted and their anomaly scores cached for efficient meta-learning experiments. Table A.3 summarizes the configuration.

A.6. Baselines

We compare the proposed `MetaEns` framework against 16 unsupervised baselines categorized into four groups. For ensemble baselines dependent on a size parameter k, we report the best k performance—the fixed ensemble size that yields the highest average AP across the benchmark—to ensure a strong competitive baseline.

Single Baselines: (1) `Singleton`: Selects a single model

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Table A.2. Summary of benchmark datasets.

Dataset	#Samples	Dimensionality	Contamination (%)	Type
ALOI	49,534	27	3.04	Numeric
Annthyroid	7,129	21	7.49	Mixed
Arrhythmia	450	259	45.78	Mixed
Cardiotocography	2,114	21	22.04	Numeric
Glass	214	7	4.21	Numeric
HeartDisease	270	13	44.44	Mixed
InternetAds	1,966	1,555	18.72	Mixed
PageBlocks	5,393	10	9.46	Numeric
PenDigits	9,868	16	0.20	Numeric
Pima	768	8	34.90	Numeric
Shuttle	1,013	9	1.28	Numeric
SpamBase	4,207	57	39.91	Numeric
Stamps	340	9	9.12	Numeric
WBC	223	9	4.48	Numeric
WDBC	367	30	2.72	Categorical
WPBC	198	33	23.74	Numeric
Waveform	3,443	21	2.90	Numeric
Wilt	4,819	5	5.33	Numeric
annthyroid	7,200	6	7.42	Mixed
arrhythmia	452	274	14.60	Mixed
breastw	683	9	34.99	Numeric
glass	214	9	4.21	Numeric
ionosphere	351	33	35.90	Numeric
letter	1,600	32	6.25	Numeric
lympho	148	18	4.05	Categorical
mammography	11,183	6	2.32	Numeric
mnist	7,603	100	9.21	Numeric
musk	3,062	166	3.17	Numeric
optdigits	5,216	64	2.88	Numeric
pendigits	6,870	16	2.27	Numeric
pima	768	8	34.90	Numeric
satellite	6,435	36	31.64	Numeric
satimage-2	5,803	36	1.22	Numeric
speech	3,686	400	1.65	Numeric
thyroid	3,772	6	2.47	Mixed
vertebral	240	6	12.50	Numeric
vowels	1,456	12	3.43	Numeric
wbc	378	30	5.56	Categorical
wine	129	13	7.75	Numeric

uniformly at random from the candidate pool; (2) `LOF` (Breunig et al., 2000); (3) `Global Best`: A static meta-learner selecting the single model with the highest average performance across all training datasets;

Naïve & Random Ensemble Baselines: (4) `IForest` (Liu et al., 2008): A tree-based ensemble that isolates anomalies using random partitions; (5) `RandNet` (Chen et al., 2017): An ensemble of autoencoders with randomized connectivity to mitigate overfitting and enhance diversity; (6) `Random Ensemble`: An ensemble of k randomly selected models; (7) `Mega Ensemble`: A naïve ensemble that averages outlier scores from all 297 models in the pool.

Deep Learning Baselines: (8) `RDA` (Zhou & Paffenroth, 2017): A robust deep autoencoder that decomposes input into low-dimensional manifold and sparse noise components; (9) `DAGMM` (Zong et al., 2018): An end-to-end framework jointly optimizing a compression network and a Gaussian Mixture Model for density estimation; (10) `DeepSVDD` (Ruff et al., 2018): Maps data into a minimum volume hypersphere to extract common factors of variation.

Meta-Learning and Ensemble Baselines: (11) `MetaOD` (Zhao et al., 2020): A model selection as a cold-start recommendation problem using matrix factorization on historical meta-features; (12) `LSCP` (Zhao et al., 2019a): An ensemble framework that selects competent base detectors for local regions of test instances; (13) `ELECT` (Zhao et al., 2022): The current state-of-the-art model selector, which identifies the single best model based on performance-driven task similarity. (14) `ELECT+k`: An ensemble baseline that aggregates the top-k models ranked by `ELECT`. This tests whether simple rank-based selection is sufficient compared to our context-aware approach. (15) `ELECT+r`: A hybrid baseline that initializes with the high-quality primary model selected by `ELECT` but expands the ensemble with random partners.

Theoretical Upper Bound: (16) `Greedy Oracle`: A fully supervised upper bound that utilizes ground-truth labels to iteratively select the candidate maximizing marginal AP at each step. This baseline quantifies the maximum potential performance achievable by a greedy sequential selection strategy.

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Table A.3. Candidate model pool specification ($M = 297$ total models).

Family	Count	Hyperparameter Grid
`kNN` (Chehreghani, 2016)	36	method $\in$ {largest, mean, median}; $k \in \{1, 5, 10, 15, 20, 25, 50, 60, 70, 80, 90, 100\}$
`LOF` (Breunig et al., 2000)	36	metric $\in$ {euclidean, manhattan, minkowski}; $k \in \{1, 5, 10, 15, 20, 25, 50, 60, 70, 80, 90, 100\}$
`IForest` (Liu et al., 2008)	81	$n_{\text{estimators}} \in \{10, 20, 30, 40, 50, 75, 100, 150, 200\}$; max_samples $\in \{0.1, 0.2, \ldots, 0.9\}$
`HBOS` (Goldstein & Dengel, 2012)	40	$n_{\text{bins}} \in \{5, 10, 20, 30, 40, 50, 75, 100\}$; tolerance $\in \{0.1, 0.2, 0.3, 0.4, 0.5\}$
`OCSVM` (Schölkopf et al., 1999)	36	kernel $\in$ {linear, poly, rbf, sigmoid}; $\nu \in \{0.1, 0.2, \ldots, 0.9\}$
`LODA` (Pevný, 2016)	54	$n_{\text{bins}} \in \{5, 10, 15, 20, 25, 30\}$; $n_{\text{cuts}} \in \{10, 20, 30, 40, 50, 75, 100, 150, 200\}$
`ABOD` (Kriegel et al., 2008)	7	$n_{\text{neighbors}} \in \{3, 5, 10, 15, 20, 25, 50\}$
`COF` (Tang et al., 2002)	7	$n_{\text{neighbors}} \in \{3, 5, 10, 15, 20, 25, 50\}$
Total	**297**	

A.7. Hyperparameter Settings

We configure the following hyperparameters, which remain fixed across all 39 test datasets to ensure fair comparison. For primary model selection via `ELECT` (Zhao et al., 2022), we adopt the default configuration. For marginal gain prediction, we train a two-part meta-model using `ExtraTrees`: (i) a classifier component for predicting improvement probability ($y_p = \mathbb{I}(g > 0)$) with balanced class weights, and (ii) a regression component for predicting improvement magnitude ($y_m = \max(0, g)$) optimized for MAE loss. For sequential selection, we set acceptance thresholds τ_1 and τ_2 for first-partner selection and iterative expansion, respectively. For diversity modulation, we configure discount strength β, family-risk weight λ_{fam}, and Jaccard similarity top-k threshold k_{top}. The ensemble size budget is η with adaptive early stopping. All hyperparameters are tuned via cross-validation on meta-training data. Detailed hyperparameter specifications are provided in the Appendix.

A.8. Implementation Details

We implement the proposed framework using PyTorch 1.13 (Paszke et al., 2019) for neural components and Scikit-learn 1.2 (Pedregosa et al., 2011) and PyOD 1.0 (Zhao et al., 2019b) for base outlier detectors in Python 3.9. All experiments were executed on a high-performance computing cluster equipped with dual NVIDIA RTX 3090 GPUs (24GB VRAM), AMD EPYC 7742 64-Core Processors, and 512GB RAM. The source code is available at `https://anonymous.4open.science/r/MetaEns`. For preprocessing, numeric features are scaled using RobustScaler (median removal and scaling according to the interquartile range) to preserve the integrity of global outliers. Categorical features are transformed via target encoding with leave-one-out regularization ($\alpha = 10$) to prevent data leakage. We add binary missingness indicators and impute missing values with median/mode. Since raw anomaly scores from different algorithms operate on vastly different scales, we strictly apply Min-Max Normalization per model per dataset to all model outputs before aggregation. To ensure reproducibility, we fix random seeds to 42 for all stochastic components.

A.9. Evaluation Metrics

Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ denotes an evaluation dataset, where $y_i \in \{0, 1\}$ indicates whether instance $\mathbf{x}_i$ is anomalous ($y_i = 1$) or normal ($y_i = 0$). Let $\mathbf{o} = (o_1, \ldots, o_N)$ denote the outlier score vector produced by a method, where larger values indicate higher anomaly likelihood. Let $\pi = \sum_{i=1}^N y_i$ denote the number of anomalies in the dataset. All ranking-based metrics are computed by sorting instances in descending order of o_i.

Average Precision (AP). Average Precision summarizes the area under the precision–recall curve:

$$\text{AP} = \frac{1}{\pi} \sum_{k=1}^{N} \mathbb{I}(y_i = 1)\ P(k), \tag{10}$$

where y_k is the label of the instance ranked at position k, and $P(k)$ is the precision at cutoff k:

$$P(k) = \frac{1}{k} \sum_{j=1}^{k} \mathbb{I}\big(y_j) = 1\big) \tag{11}$$

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Average Rank (AR). Let $r_{f_i,\mathbf{X}_j}$ denote the rank of method f_i on dataset $\mathbf{X}_j$ according to Average Precision, where rank 1 indicates the best-performing method. The Average Rank of method f_i is defined as follows.

$$\mathrm{AR}_{f_i} = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{X}_j \in \mathcal{D}} r_{f_i,\mathbf{X}_j}, \tag{12}$$

where $\mathcal{D}$ denotes the set of benchmark datasets. Lower values indicate better overall performance.

ROC-AUC. The ROC-AUC measures the probability that an anomalous instance receives a higher outlier score than a normal instance:

$$\mathrm{AUC} = \mathbb{P}(o^+ > o^-) = \frac{1}{\pi(N-\pi)} \sum_{i:y_i=1} \sum_{j:y_j=0} \mathbb{I}(o_i > o_j), \tag{13}$$

where o^+ and o^- denote outlier scores of anomalous and normal instances, respectively.

Precision@n. Precision@n evaluates the accuracy of the top-ranked predictions and is defined as

$$\mathrm{Precision@n} = \frac{1}{n} \sum_{k=1}^{n} \mathbb{I}\big(y_{(k)} = 1\big), \tag{14}$$

where $n = \pi$ is set to the number of ground-truth anomalies in the dataset. This metric reflects practical inspection scenarios where only the top-ranked instances are examined.

Max F1-score. For a decision threshold τ applied to outlier scores, predicted labels are defined as follows.

$$\hat{y}_i(\tau) = \mathbb{I}(o_i \geq \tau) \tag{15}$$

Precision and recall at threshold τ are given as follows.

$$\begin{aligned} \mathrm{Prec}(\tau) &= \frac{\sum_i \mathbb{I}(\hat{y}_i(\tau) = 1 \wedge y_i = 1)}{\sum_i \mathbb{I}(\hat{y}_i(\tau) = 1)} \\ \mathrm{Rec}(\tau) &= \frac{\sum_i \mathbb{I}(\hat{y}_i(\tau) = 1 \wedge y_i = 1)}{\sum_i \mathbb{I}(y_i = 1)} \end{aligned} \tag{16}$$

The F1-score at threshold τ is defined as follows.

$$\mathrm{F1}(\tau) = \frac{2 \cdot \mathrm{Prec}(\tau) \cdot \mathrm{Rec}(\tau)}{\mathrm{Prec}(\tau) + \mathrm{Rec}(\tau)} \tag{17}$$

The Max F1-score is defined as follows.

$$\mathrm{MaxF1} = \max_{\tau} \mathrm{F1}(\tau) \tag{18}$$

A.10. Detailed Experimental Results

Table A.4 reports the detailed performance of `MetaEns` and all baselines on each of the 39 benchmark datasets. For each dataset, we report Average Precision (AP) scores computed under the same evaluation protocol described in Section 4. This table complements the aggregated results in the main paper by providing a per-dataset view of method behavior, enabling fine-grained comparison and reproducibility analysis.

A.11. Statistical Significance Analysis

To rigorously validate the performance improvements of `MetaEns`, we conduct paired Wilcoxon signed-rank tests comparing our method against all baselines across the 39-dataset benchmark. The Wilcoxon signed-rank test is a non-parametric statistical test appropriate for paired comparisons when distributions may not be normal. We perform one-sided tests with the alternative hypothesis that `MetaEns` achieves higher Average Precision than each baseline, using a significance level of $\alpha = 0.05$.

Table A.5 presents the statistical significance results. The ΔAP column shows the mean difference in AP between `MetaEns` and each baseline (positive values indicate `MetaEns` outperforms the baseline). The Win% represents the proportion of datasets where `MetaEns` achieves AP greater than or equal to the baseline.

`MetaEns` demonstrates statistically significant improvements over all baselines, including all deep learning methods, naïve and random ensembles, meta-learning approaches, and single baselines. Further, the consistent positive ΔAP values and win rates above 60% across all comparisons underscore the robustness of `MetaEns`.

A.12. Pool Size Analysis

We study the performance and robustness of `MetaEns` by varying the size of the candidate model pool from 10 to 297 detectors. For each pool size, we randomly sample the specified number of models and apply the full `MetaEns` selection procedure. To account for variability introduced by sub-pool sampling, experiments are repeated with 10 random seeds and results are averaged.

Figure A.1 shows the relationship between pool size and final ensemble performance measured by AP. The blue curve denotes the average AP across the 10 runs, and the shaded region indicates ± 1 standard deviation.

Performance improves steadily as the pool size increases, indicating that `MetaEns` effectively leverages a richer set of candidate models. Gains begin to plateau around 250 models, suggesting diminishing returns beyond this point. This trend indicates that the full pool of 297 models is sufficient to capture the diversity needed for robust anomaly detection across the benchmark datasets.

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Table A.4. Detailed Performance Comparison Across 39 Benchmark Datasets. For each method, we report Average Precision (AP) and model rank in parentheses (lower rank is better, 1–297). Best results in **bold**. Abbreviations: `RS` = `Random Selection`, `RE` = `Random Ensemble` (k=3), `ELECT` = `ELECT` (Top-1), `ELECT-10` = `ELECT` (Top-10).

Dataset	RS	IForest	LOF	GB	ME	RE	RDA	DAGMM	DeepSVDD	RandNet	LSCP	MetaOD	ELECT-1	ELECT-10	MetaEns
ALOI	0.039 (9)	0.034 (13)	0.074 (5)	0.032 (15)	0.036 (11)	0.035 (12)	0.039 (8)	0.038 (10)	0.049 (6)	0.040 (7)	0.077 (4)	0.033 (14)	0.164 (2)	0.158 (3)	**0.164 (1)**
Annthyroid	0.104 (11)	0.113 (9)	0.129 (7)	0.139 (5)	0.135 (6)	0.129 (8)	0.083 (14)	0.090 (12)	0.109 (10)	0.066 (15)	0.083 (13)	0.155 (4)	0.197 (3)	0.218 (2)	**0.223 (1)**
Arrhythmia	0.675 (12)	0.765 (2)	0.755 (6)	0.751 (7)	0.748 (8)	0.739 (11)	0.669 (13)	0.637 (14)	0.501 (15)	0.747 (9)	0.746 (10)	**0.768 (1)**	0.761 (4)	0.764 (3)	0.757 (5)
Cardiotocography	0.412 (7)	0.437 (5)	0.302 (14)	0.473 (2)	0.408 (8)	0.398 (9)	0.264 (15)	0.362 (12)	0.308 (13)	**0.531 (1)**	0.392 (10)	0.442 (4)	0.429 (6)	0.444 (3)	0.384 (11)
Glass	0.134 (9)	0.153 (7)	0.092 (15)	0.197 (5)	0.120 (11)	0.136 (8)	0.192 (6)	0.121 (10)	0.107 (13)	0.117 (12)	0.213 (2)	**0.253 (1)**	0.209 (4)	0.212 (3)	0.102 (14)
HeartDisease	**0.598 (1)**	0.541 (7)	0.574 (2)	0.525 (11)	0.571 (3)	0.566 (4)	0.458 (14)	0.441 (15)	0.546 (6)	0.536 (9)	0.480 (13)	0.523 (12)	0.538 (8)	0.532 (10)	0.562 (5)
InternetAds	0.331 (12)	0.527 (5)	0.366 (11)	0.455 (9)	0.482 (8)	0.405 (10)	0.296 (13)	0.277 (14)	0.202 (15)	0.525 (6)	0.576 (2)	0.525 (7)	0.534 (4)	0.536 (3)	**0.579 (1)**
PageBlocks	0.367 (13)	0.465 (8)	0.531 (3)	0.409 (10)	0.385 (12)	0.390 (11)	0.436 (9)	0.262 (14)	0.227 (15)	0.551 (2)	**0.564 (1)**	0.467 (6)	0.480 (5)	0.466 (7)	0.494 (4)
PenDigits	0.006 (10)	0.005 (12)	0.019 (4)	0.006 (11)	0.007 (8)	0.007 (7)	0.020 (3)	0.014 (6)	**0.068 (1)**	0.002 (15)	0.015 (5)	0.005 (14)	0.007 (9)	0.005 (13)	0.021 (2)
Pima	0.466 (10)	**0.516 (1)**	0.514 (2)	0.436 (13)	0.500 (5)	0.488 (8)	0.451 (12)	0.409 (15)	0.451 (11)	0.410 (14)	0.474 (9)	0.499 (6)	0.509 (3)	0.507 (4)	0.491 (7)
Shuttle	0.129 (5)	0.069 (13)	0.355 (3)	0.090 (9)	0.117 (6)	0.114 (7)	0.040 (14)	0.175 (4)	0.388 (2)	0.022 (15)	0.084 (10)	0.071 (12)	0.090 (8)	0.079 (11)	**0.396 (1)**
SpamBase	0.476 (9)	0.480 (7)	0.355 (14)	0.533 (4)	0.483 (6)	0.439 (10)	0.399 (12)	0.349 (15)	0.367 (13)	0.404 (11)	0.494 (5)	0.479 (8)	0.559 (2)	0.557 (3)	**0.594 (1)**
Stamps	0.249 (11)	0.307 (8)	0.333 (5)	0.333 (4)	0.333 (3)	0.256 (10)	0.196 (12)	0.113 (15)	0.164 (13)	0.123 (14)	0.334 (2)	**0.345 (1)**	0.309 (7)	0.327 (6)	0.269 (9)
WBC	0.556 (11)	0.882 (3)	0.875 (5)	0.827 (9)	0.864 (8)	0.604 (10)	0.274 (15)	0.537 (12)	0.368 (14)	0.394 (13)	**0.895 (1)**	0.877 (4)	0.874 (6)	0.885 (2)	0.865 (7)
WDBC	0.647 (11)	0.647 (10)	0.691 (7)	0.716 (4)	0.697 (5)	0.684 (8)	0.092 (15)	0.551 (13)	0.316 (14)	0.630 (12)	**0.800 (1)**	0.678 (9)	0.760 (3)	0.694 (6)	0.768 (2)
WPBC	0.233 (6)	0.231 (8)	0.232 (7)	0.239 (4)	0.231 (9)	0.230 (10)	0.243 (3)	0.206 (15)	**0.266 (1)**	0.229 (11)	0.265 (2)	0.227 (13)	0.229 (12)	0.225 (14)	0.235 (5)
Waveform	0.078 (7)	0.061 (9)	0.131 (3)	0.055 (13)	0.066 (8)	0.081 (6)	0.029 (15)	0.033 (14)	0.056 (12)	0.058 (10)	0.168 (2)	0.056 (11)	0.115 (5)	0.118 (4)	**0.190 (1)**
Wilt	0.048 (9)	0.045 (13)	0.053 (6)	0.041 (15)	0.043 (14)	0.055 (4)	**0.283 (1)**	0.089 (2)	0.046 (10)	0.054 (5)	0.079 (3)	0.045 (12)	0.048 (8)	0.046 (11)	0.048 (7)
annthyroid	0.195 (11)	0.314 (6)	0.204 (10)	0.366 (4)	0.285 (7)	0.247 (9)	0.161 (12)	0.128 (14)	0.120 (15)	0.135 (13)	0.257 (8)	0.336 (5)	0.452 (2)	0.395 (3)	**0.531 (1)**
arrhythmia	0.424 (12)	0.479 (4)	0.464 (7)	**0.502 (1)**	0.445 (10)	0.431 (11)	0.313 (13)	0.257 (15)	0.309 (14)	0.460 (8)	0.459 (9)	0.482 (2)	0.474 (6)	0.480 (3)	0.475 (5)
breastw	0.895 (9)	0.969 (4)	0.392 (14)	0.967 (5)	0.979 (2)	0.941 (8)	0.883 (10)	0.782 (11)	0.320 (15)	0.686 (12)	0.588 (13)	**0.979 (1)**	0.955 (6)	0.955 (7)	0.972 (3)
glass	0.076 (12)	0.093 (7)	0.083 (10)	0.116 (4)	0.085 (9)	0.080 (11)	0.195 (2)	0.052 (14)	0.041 (15)	**0.205 (1)**	0.125 (3)	0.116 (5)	0.092 (8)	0.104 (6)	0.058 (13)
ionosphere	0.636 (13)	0.809 (4)	0.799 (6)	0.758 (10)	0.803 (5)	0.745 (11)	**0.949 (1)**	0.599 (14)	0.565 (15)	0.735 (12)	0.790 (8)	0.910 (2)	0.793 (7)	0.813 (3)	0.759 (9)
letter	0.140 (5)	0.087 (12)	0.244 (2)	0.088 (10)	0.130 (6)	0.220 (3)	**0.311 (1)**	0.094 (8)	0.081 (15)	0.100 (7)	0.209 (4)	0.091 (9)	0.085 (13)	0.088 (11)	0.082 (14)
lympho	0.782 (9)	0.944 (3)	0.857 (6)	0.976 (2)	0.877 (5)	0.822 (8)	0.151 (14)	0.519 (12)	0.056 (15)	0.897 (4)	0.173 (13)	**1.000 (1)**	0.720 (10)	0.852 (7)	0.556 (11)
mammography	0.184 (9)	0.221 (6)	0.121 (11)	0.221 (7)	0.247 (3)	0.198 (8)	0.171 (10)	0.100 (12)	0.070 (15)	0.073 (14)	0.082 (13)	0.238 (4)	0.269 (2)	0.235 (5)	**0.284 (1)**
mnist	0.246 (12)	0.265 (9)	0.379 (3)	0.262 (10)	0.278 (8)	0.290 (5)	**0.393 (1)**	0.225 (13)	0.207 (14)	0.392 (2)	0.371 (4)	0.246 (11)	0.282 (7)	0.282 (6)	0.160 (15)
musk	0.548 (12)	1.000 (5)	0.090 (14)	1.000 (6)	1.000 (7)	0.882 (10)	0.450 (13)	0.856 (11)	0.043 (15)	0.998 (8)	0.994 (9)	1.000 (4)	1.000 (3)	1.000 (2)	**1.000 (1)**
optdigits	0.038 (11)	0.051 (6)	0.021 (14)	0.043 (10)	0.054 (5)	0.050 (7)	0.023 (13)	0.018 (15)	0.032 (12)	0.070 (3)	0.072 (2)	0.055 (4)	0.050 (8)	0.048 (9)	**0.114 (1)**
pendigits	0.195 (9)	0.279 (3)	0.044 (13)	0.255 (7)	0.260 (6)	0.214 (8)	0.035 (15)	0.083 (11)	0.036 (14)	0.182 (10)	0.066 (12)	0.266 (5)	0.289 (2)	0.272 (4)	**0.359 (1)**
pima	0.477 (10)	0.500 (5)	0.493 (6)	0.465 (11)	0.503 (4)	**0.521 (1)**	0.405 (13)	0.431 (12)	0.398 (14)	0.368 (15)	0.490 (7)	0.505 (3)	0.480 (9)	0.487 (8)	0.516 (2)
satellite	0.604 (9)	0.660 (4)	0.397 (15)	0.660 (5)	0.680 (2)	0.613 (8)	0.508 (12)	0.574 (10)	0.479 (13)	0.527 (11)	0.415 (14)	0.658 (6)	0.658 (7)	0.671 (3)	**0.698 (1)**
satimage-2	0.709 (10)	0.926 (4)	0.142 (13)	0.915 (7)	0.944 (2)	0.754 (9)	0.100 (14)	0.394 (11)	0.063 (15)	**0.961 (1)**	0.393 (12)	0.921 (6)	0.927 (3)	0.924 (5)	0.895 (8)
speech	0.027 (2)	0.018 (14)	0.020 (9)	0.027 (4)	0.021 (8)	**0.028 (1)**	0.019 (10)	0.023 (7)	0.027 (3)	0.019 (11)	0.019 (12)	0.018 (13)	0.026 (5)	0.026 (6)	0.017 (15)
thyroid	0.464 (7)	0.557 (5)	0.335 (11)	0.380 (10)	0.497 (6)	0.434 (8)	0.309 (12)	0.146 (14)	0.049 (15)	0.176 (13)	0.410 (9)	0.612 (3)	0.651 (2)	0.604 (4)	**0.746 (1)**
vertebral	0.095 (8)	0.096 (7)	0.088 (13)	0.092 (11)	0.087 (14)	0.090 (12)	0.116 (3)	0.144 (2)	0.103 (4)	**0.156 (1)**	0.000 (15)	0.098 (6)	0.094 (9)	0.093 (10)	0.099 (5)
vowels	0.178 (7)	0.138 (11)	0.385 (3)	0.106 (13)	0.233 (5)	0.195 (6)	0.458 (2)	0.067 (14)	0.057 (15)	0.140 (10)	0.369 (4)	0.127 (12)	0.155 (9)	0.165 (8)	**0.491 (1)**
wbc	0.612 (6)	0.608 (7)	0.650 (3)	0.582 (11)	0.597 (8)	0.616 (5)	0.241 (15)	0.415 (14)	0.447 (13)	0.596 (9)	0.542 (12)	0.591 (10)	0.639 (4)	**0.668 (1)**	0.663 (2)
wine	0.246 (5)	0.213 (10)	0.290 (2)	0.234 (6)	0.255 (4)	0.274 (3)	0.093 (13)	0.112 (12)	0.081 (14)	0.220 (7)	0.000 (15)	0.211 (11)	0.215 (8)	0.215 (9)	**0.366 (1)**
Average	0.342 (9.00)	0.398 (7.08)	0.330 (7.79)	0.392 (7.79)	0.397 (6.79)	0.369 (7.82)	0.276 (10.08)	0.275 (11.74)	0.208 (11.90)	0.347 (9.05)	0.348 (7.51)	0.408 (6.72)	0.413 (5.87)	0.414 (5.85)	**0.435 (5.00)**
Std Dev	0.246	0.308	0.248	0.302	0.301	0.268	0.223	0.227	0.171	0.285	0.267	0.313	0.294	0.297	0.285

Table A.5. Statistical significance testing: Paired Wilcoxon signed-rank test comparing `MetaEns` against baselines across 39 datasets (one-sided test, $\alpha = 0.05$).

Ours	Baseline	p-value	ΔAP	Win%
`MetaEns`	`Random Selection`	0.0000	+0.0939	79.5%
`MetaEns`	`IForest`	0.0151	+0.0378	64.1%
`MetaEns`	`LOF`	0.0022	+0.1052	69.2%
`MetaEns`	`Global Best`	0.0050	+0.0438	71.8%
`MetaEns`	`Mega Ensemble`	0.0096	+0.0384	66.7%
`MetaEns`	`Random Ensemble`	0.0002	+0.0662	74.4%
`MetaEns`	`RDA`	0.0001	+0.1599	76.9%
`MetaEns`	`DAGMM`	0.0000	+0.1605	84.6%
`MetaEns`	`DeepSVDD`	0.0000	+0.2272	84.6%
`MetaEns`	`RandNet`	0.0013	+0.0884	71.8%
`MetaEns`	`LSCP`	0.0047	+0.0876	66.7%
`MetaEns`	`MetaOD`	0.0390	+0.0275	61.5%
`MetaEns`	`ELECT-1`	0.0314	+0.0222	66.7%
`MetaEns`	`ELECT-10`	0.0386	+0.0213	61.5%

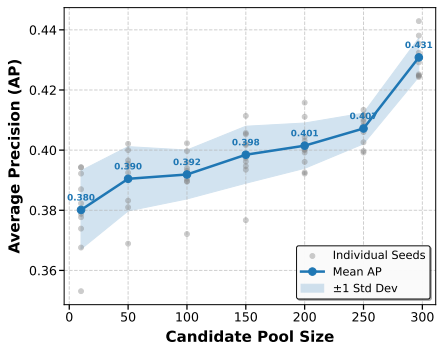

Figure A.1. Impact of candidate pool size on `MetaEns` performance. Results are averaged over 10 random seeds for each pool size. The shaded area represents ± 1 standard deviation.

A.13. Effects of Meta-Model Architectures

The effectiveness of `MetaEns` depends on the underlying models used to predict marginal gains (f_{reg}) and filter candidates (f_{cls}). In this section, we justify our default choice of `ExtraTrees` by comparing it with five alternative learning families: `Random Forest` (Breiman, 2001), `XGBoost` (Chen & Guestrin, 2016), `LightGBM` (Ke et al., 2017), `Multi-Layer Perceptrons` (Goodfellow et al., 2016), and `Linear Models` (James et al., 2013). All methods are evaluated using the same meta-features and training protocol.

To ensure a fair comparison, we employ strong, standardized configurations across baselines. Tree ensembles (`Random Forest`, `ExtraTrees`, `XGBoost`, `LightGBM`) use 500 estimators for classification tasks and 800 for regression tasks. Gradient boosting models (`XGBoost`, `LightGBM`) adopt a subsample ratio of 0.8 to improve generalization. The `Multi-Layer Perceptrons` model consists of two hidden layers with sizes $(100, 50)$, trained for 500 epochs. `Linear Models` use Logistic Regression and Ridge Regression with default regularization.

Table A.6 summarizes the results. *Train (s)* denotes the average wall-clock time to train the meta-model on the meta-

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Table A.6. Comparison of meta-model architectures for `MetaEns` selection quality.

Method	AP ↑	Rank ↓	ROC-AUC ↑	Train (s) ↓	Infer (s) ↓
`ExtraTrees` (**Ours**)	**0.4308** ± 0.0064	**59.3** ± 6.9610	**0.7867** ± 0.0045	4.1518	1.6839
`Linear Models`	0.4056 ± 0.0015	105.2 ± 2.5000	0.7737 ± 0.0010	6.9150	**1.0327**
`XGBoost`	0.4110 ± 0.0055	108.0 ± 8.2000	0.7664 ± 0.0040	**1.7408**	1.4910
`Random Forest`	0.4043 ± 0.0062	77.0 ± 7.1000	0.7617 ± 0.0050	15.5561	1.5200
`LightGBM`	0.3899 ± 0.0058	99.0 ± 8.5000	0.7600 ± 0.0042	1.9412	1.3405
`Multi-Layer Perceptrons`	0.3711 ± 0.0115	110.0 ± 12.5000	0.7607 ± 0.0090	35.6667	10.7667

training set (leave-one-out protocol), while *Infer (s)* measures the time required to select models for a new dataset. `ExtraTrees` consistently achieves the best predictive performance, with the highest AP (0.4308) and best average rank (59.3), while maintaining efficient training (1.37s) and inference (1.57s). `XGBoost` provides competitive training speed but trails in AP (0.4110). `Linear Models` offer the fastest inference (1.03s) but cannot capture the non-linear relationships required for accurate selection, resulting in substantially worse ranking performance. `Multi-Layer Perceptrons` perform poorly both in accuracy (lowest AP 0.3711) and computational efficiency, indicating that neural architectures are less suitable for this structured meta-learning task.

A.14. Effect of Expanded Model Pools

To evaluate `MetaEns`' scalability and robustness to model pool composition, we conduct additional experiments with an expanded pool of 310 models that incorporates neural network variants alongside the original 297 classical detectors.

Expanded Pool Composition. The expanded pool adds 13 neural network variants spanning 5 deep learning families: `AutoEncoder` (Chen et al., 2017) (4 architectural variants), `Variational AutoEncoder` (Xu et al., 2018) (3 variants), `SO_GAAL` (Liu et al., 2020) (2 configurations), `MO_GAAL` (Liu et al., 2020) (2 configurations), and `DeepSVDD` (Ruff et al., 2018) (2 variants). These models represent diverse deep learning paradigms, including reconstruction-based methods, adversarial approaches, and deep one-class classification. Table A.7 provides the complete specification of the 310-model expanded pool.

Overall Performance Comparison. Table A.8 presents the scalability analysis results comparing `MetaEns` performance on the original (297 models) versus expanded (310 models) pools. `MetaEns` demonstrates robust performance across both pool configurations, maintaining competitive efficacy with only a modest decrease in Average Precision (0.0186) and an increase in median rank (13.0).

Detailed Dataset-by-Dataset Results. Table A.9 presents the complete performance comparison of all methods on the expanded 310-model pool. The table shows Average Precision (AP) values and model ranks (1–310, where lower ranks indicate better performance) for each method across all 39 benchmark datasets, with best results for each dataset highlighted in bold.

Results Analysis. The combined results demonstrate several key findings:

- **Scalability:** `MetaEns` successfully handles the expanded 310-model pool with minimal performance impact, validating its applicability to larger model collections.
- **Robustness:** Performance remains stable across different pool compositions, indicating robust meta-learning that generalizes across model types.
- **Discriminative Selection:** Analysis reveals that despite several neural network models achieving competitive individual performance, `MetaEns` predominantly selects classical models, demonstrating intelligent cross-dataset pattern recognition.
- **Competitive Performance:** `MetaEns` achieves the best performance on 10 out of 39 datasets (25.6%) on the expanded pool, demonstrating consistent competitive behavior across diverse anomaly detection tasks.

The stable performance across pool configurations validates `MetaEns`' design principle of leveraging comprehensive cross-dataset information for intelligent model selection, supporting its applicability as new model architectures emerge in the anomaly detection landscape.

A.15. Ensemble Size Impact Analysis

To understand how ensemble size affects performance, we analyze the relationship between the number of models and detection quality. Figure A.2 presents the Average Precision (AP) and ROC AUC as a function of ensemble size for several representative methods.

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Table A.7. Expanded candidate model pool specification with neural network variants ($M = 310$ total models).

Family	Count	Hyperparameter Grid
`kNN` (Chehreghani, 2016)	36	method $\in$ {largest, mean, median}; $k \in \{1, 5, 10, 15, 20, 25, 50, 60, 70, 80, 90, 100\}$
`LOF` (Breunig et al., 2000)	36	metric $\in$ {euclidean, manhattan, minkowski}; $k \in \{1, 5, 10, 15, 20, 25, 50, 60, 70, 80, 90, 100\}$
`IForest` (Liu et al., 2008)	81	$n_{\text{estimators}} \in \{10, 20, 30, 40, 50, 75, 100, 150, 200\}$; max_samples $\in \{0.1, 0.2, \ldots, 0.9\}$
`HBOS` (Goldstein & Dengel, 2012)	40	$n_{\text{bins}} \in \{5, 10, 20, 30, 40, 50, 75, 100\}$; tolerance $\in \{0.1, 0.2, 0.3, 0.4, 0.5\}$
`OCSVM` (Schölkopf et al., 1999)	36	kernel $\in$ {linear, poly, rbf, sigmoid}; $\nu \in \{0.1, 0.2, \ldots, 0.9\}$
`LODA` (Pevný, 2016)	54	$n_{\text{bins}} \in \{5, 10, 15, 20, 25, 30\}$; $n_{\text{cuts}} \in \{10, 20, 30, 40, 50, 75, 100, 150, 200\}$
`ABOD` (Kriegel et al., 2008)	7	$n_{\text{neighbors}} \in \{3, 5, 10, 15, 20, 25, 50\}$
`COF` (Tang et al., 2002)	7	$n_{\text{neighbors}} \in \{3, 5, 10, 15, 20, 25, 50\}$
Neural Network Extensions		
`Autoencoder` (Chen et al., 2017)	4	Architecture variants: `[64,32,32,64]`, `[128,64,64,128]`, `[32,16,16,32]`, `[64,32,16,16,32,64]`
`Variational Autoencoder` (Xu et al., 2018)	3	Encoder-decoder pairs: `([64,32],[32,64])`, `([128,64],[64,128])`, `([32,16],[16,32])`
`SO_GAAL` (Liu et al., 2020)	2	Single-objective GAN: `stop_epochs`=20; lr $\in \{0.0005, 0.001\}$
`MO_GAAL` (Liu et al., 2020)	2	Multi-objective GAN: `stop_epochs`=20; $k \in \{5, 20\}$
`DeepSVDD` (Ruff et al., 2018)	2	Deep SVDD: pre-training $\in$ {True, False}
Classical Total	**297**	Traditional anomaly detection methods
Neural Network Total	**13**	Deep learning-based detection methods
Grand Total	**310**	Complete expanded model pool

A key finding from our experiments is that `MetaEns` employs adaptive ensemble sizing, selecting an average of 2.2 models per dataset across the 39-dataset benchmark. This adaptive behavior contrasts sharply with fixed-size baselines: `ELECT` Top-10 always uses 10 models, while Random Ensemble uses a fixed best $k = 3$. The ability to adjust ensemble size based on dataset characteristics is a crucial advantage of our approach.

The performance comparison in Table 1 reveals important insights about ensemble construction strategies. `ELECT` Top-10, despite using 10 models, achieves only 0.414 AP—a marginal 0.001 improvement over `ELECT` (Top-1) with 0.413 AP. This negligible gain from expanding to 10 models suggests that simple rank-based aggregation without diversity consideration leads to redundant model selection. In contrast, `MetaEns` achieves 0.435 AP with an average of only 2.2 models, demonstrating that carefully selected small ensembles can substantially outperform large ensembles of top-ranked models.

The Random Ensemble baseline with best $k = 3$ achieves 0.369 AP, performing worse than both the single `ELECT` selector (0.413 AP) and `MetaEns` (0.435 AP). This indicates that expanding ensembles with randomly chosen partners actively degrades performance by introducing low-quality models that add noise to the ensemble prediction. This finding validates our hypothesis that partner selection must be guided by both quality and diversity considerations.

Our adaptive stopping mechanism enables dataset-specific ensemble construction: simple datasets may benefit from small ensembles to avoid overfitting, while more complex datasets with diverse outlier patterns may require additional models to capture complementary perspectives. Fixed ensemble size baselines cannot adapt to this heterogeneity, leading to suboptimal performance across the benchmark.

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.

Table A.8. `MetaEns` scalability analysis: Performance on original (297) vs. expanded (310) model pools. Results demonstrate method robustness across different pool compositions.

Pool Configuration	Size	AP ↑	Rank ↓	ROC-AUC ↑	Ens Size
Classical Models Only	297	0.4308 ± 0.0064	59.3 ± 6.9610	0.7867 ± 0.0045	2.2
Classical + Neural Networks	310	0.4122 ± 0.0068	72.3 ± 11.4018	0.7823 ± 0.0044	2.3

Table A.9. Detailed Performance Comparison Across 39 Benchmark Datasets on Expanded Neural Network Pool (310 models). For each method, we report Average Precision (AP) and model rank in parentheses (lower rank is better, 1–310). Best results in **bold**. Abbreviations: RS = Random Selection, RE = Random Ensemble (k=3), ELECT-1 = ELECT (Top-1), ELECT-10 = ELECT (Top-10).

Dataset	RS	IForest	LOF	GB	ME	RE	RDA	DAGMM	DeepSVDD	RandNet	LSCP	MetaOD	ELECT-1	ELECT-10	MetaEns
ALOI	0.039 (9)	0.034 (13)	0.074 (5)	0.032 (15)	0.036 (11)	0.035 (12)	0.039 (8)	0.038 (10)	0.049 (6)	0.040 (7)	0.077 (4)	0.033 (14)	0.164 (2)	0.158 (3)	**0.164 (1)**
Annthyroid	0.104 (11)	0.113 (9)	0.129 (7)	0.139 (5)	0.136 (6)	0.129 (8)	0.083 (14)	0.090 (12)	0.109 (10)	0.066 (15)	0.083 (13)	0.155 (4)	0.197 (3)	**0.218 (1)**	0.197 (2)
Arrhythmia	0.675 (12)	0.765 (2)	0.755 (5)	0.751 (6)	0.746 (9)	0.737 (10)	0.669 (13)	0.637 (14)	0.501 (15)	0.747 (7)	0.746 (8)	**0.768 (1)**	0.761 (4)	0.764 (3)	0.726 (11)
Cardiotocography	0.412 (7)	0.437 (5)	0.302 (14)	0.473 (2)	0.410 (8)	0.400 (9)	0.264 (15)	0.362 (12)	0.308 (13)	**0.531 (1)**	0.392 (10)	0.442 (4)	0.429 (6)	0.444 (3)	0.384 (11)
Glass	0.134 (8)	0.153 (7)	0.092 (15)	0.197 (5)	0.119 (11)	0.131 (9)	0.192 (6)	0.121 (10)	0.107 (13)	0.117 (12)	0.213 (2)	**0.253 (1)**	0.209 (4)	0.212 (3)	0.102 (14)
HeartDisease	**0.598 (1)**	0.541 (7)	0.574 (2)	0.525 (11)	0.567 (3)	0.566 (4)	0.458 (14)	0.441 (15)	0.546 (6)	0.536 (9)	0.480 (13)	0.523 (12)	0.538 (8)	0.532 (10)	0.562 (5)
InternetAds	0.339 (12)	0.527 (5)	0.366 (11)	0.455 (9)	0.476 (8)	0.390 (10)	0.296 (13)	0.277 (14)	0.202 (15)	0.525 (6)	0.576 (2)	0.525 (7)	0.534 (4)	0.536 (3)	**0.579 (1)**
PageBlocks	0.325 (13)	0.465 (8)	0.531 (3)	0.409 (10)	0.393 (11)	0.368 (12)	0.436 (9)	0.262 (14)	0.227 (15)	0.551 (2)	**0.564 (1)**	0.467 (6)	0.480 (5)	0.466 (7)	0.494 (4)
PenDigits	0.006 (10)	0.005 (12)	0.019 (4)	0.006 (11)	0.007 (8)	0.007 (7)	0.020 (3)	0.014 (6)	**0.068 (1)**	0.002 (15)	0.015 (5)	0.005 (14)	0.007 (9)	0.005 (13)	0.021 (2)
Pima	0.466 (10)	**0.516 (1)**	0.514 (2)	0.436 (13)	0.497 (7)	0.488 (8)	0.451 (12)	0.409 (15)	0.451 (11)	0.410 (14)	0.474 (9)	0.499 (6)	0.509 (4)	0.507 (5)	0.512 (3)
Shuttle	0.129 (5)	0.069 (13)	0.355 (3)	0.090 (9)	0.119 (6)	0.114 (7)	0.040 (14)	0.175 (4)	0.388 (2)	0.022 (15)	0.084 (10)	0.071 (12)	0.090 (8)	0.079 (11)	**0.436 (1)**
SpamBase	0.476 (9)	0.480 (6)	0.355 (14)	0.533 (3)	0.479 (8)	0.439 (10)	0.399 (12)	0.349 (15)	0.367 (13)	0.404 (11)	0.494 (5)	0.479 (7)	**0.559 (1)**	0.557 (2)	0.530 (4)
Stamps	0.267 (10)	0.307 (8)	0.333 (3)	0.333 (4)	0.332 (5)	0.265 (11)	0.196 (12)	0.113 (15)	0.164 (13)	0.123 (14)	0.334 (2)	**0.345 (1)**	0.309 (7)	0.327 (6)	0.269 (9)
WBC	0.483 (12)	0.882 (3)	0.875 (6)	0.827 (9)	0.858 (8)	0.605 (10)	0.274 (15)	0.537 (11)	0.368 (14)	0.394 (13)	**0.895 (1)**	0.877 (4)	0.874 (7)	0.885 (2)	0.875 (5)
WDBC	0.647 (11)	0.647 (10)	0.691 (6)	0.716 (4)	0.686 (7)	0.684 (8)	0.092 (15)	0.551 (13)	0.316 (14)	0.630 (12)	**0.800 (1)**	0.678 (9)	0.760 (3)	0.694 (5)	0.768 (2)
WPBC	0.233 (6)	0.231 (8)	0.232 (7)	0.239 (4)	0.230 (9)	0.229 (10)	0.243 (3)	0.206 (15)	**0.266 (1)**	0.229 (11)	0.265 (2)	0.227 (13)	0.229 (12)	0.225 (14)	0.235 (5)
Waveform	0.078 (7)	0.061 (9)	0.131 (3)	0.055 (13)	0.067 (8)	0.080 (6)	0.029 (15)	0.033 (14)	0.056 (12)	0.058 (10)	**0.168 (1)**	0.056 (11)	0.115 (5)	0.118 (4)	0.166 (2)
Wilt	0.048 (9)	0.045 (13)	0.053 (6)	0.041 (15)	0.044 (14)	0.055 (4)	**0.283 (1)**	0.089 (2)	0.046 (10)	0.054 (5)	0.079 (3)	0.045 (12)	0.048 (8)	0.046 (11)	0.048 (7)
annthyroid	0.195 (11)	0.314 (6)	0.204 (10)	0.366 (4)	0.290 (7)	0.247 (9)	0.161 (12)	0.128 (14)	0.120 (15)	0.135 (13)	0.257 (8)	0.336 (5)	0.452 (2)	0.395 (3)	**0.568 (1)**
arrhythmia	0.424 (12)	0.479 (4)	0.464 (7)	**0.502 (1)**	0.444 (10)	0.431 (11)	0.313 (13)	0.257 (15)	0.309 (14)	0.460 (8)	0.459 (9)	0.482 (2)	0.474 (6)	0.480 (3)	0.475 (5)
breastw	0.895 (9)	0.969 (4)	0.392 (14)	0.967 (5)	**0.979 (1)**	0.940 (8)	0.883 (10)	0.782 (11)	0.320 (15)	0.686 (12)	0.588 (13)	0.979 (2)	0.955 (6)	0.955 (7)	0.977 (3)
glass	0.076 (12)	0.093 (8)	0.083 (11)	0.116 (4)	0.085 (10)	0.095 (7)	0.195 (2)	0.052 (14)	0.041 (15)	**0.205 (1)**	0.125 (3)	0.116 (5)	0.092 (9)	0.104 (6)	0.058 (13)
ionosphere	0.636 (13)	0.809 (4)	0.799 (6)	0.758 (10)	0.806 (5)	0.752 (11)	**0.949 (1)**	0.599 (14)	0.565 (15)	0.735 (12)	0.790 (8)	0.910 (2)	0.793 (7)	0.813 (3)	0.771 (9)
letter	0.140 (5)	0.087 (12)	0.244 (2)	0.088 (10)	0.134 (6)	0.225 (3)	**0.311 (1)**	0.094 (8)	0.081 (15)	0.100 (7)	0.209 (4)	0.091 (9)	0.085 (13)	0.088 (11)	0.082 (14)
lympho	0.782 (9)	0.944 (3)	0.857 (6)	0.976 (2)	0.877 (5)	0.827 (8)	0.151 (14)	0.519 (12)	0.056 (15)	0.897 (4)	0.173 (13)	**1.000 (1)**	0.720 (10)	0.852 (7)	0.556 (11)
mammography	0.184 (9)	0.221 (6)	0.121 (11)	0.221 (7)	0.252 (3)	0.198 (8)	0.171 (10)	0.100 (12)	0.070 (15)	0.073 (14)	0.082 (13)	0.238 (4)	0.269 (2)	0.235 (5)	**0.284 (1)**
mnist	0.246 (12)	0.265 (9)	0.379 (3)	0.262 (10)	0.274 (8)	0.295 (5)	**0.393 (1)**	0.225 (13)	0.207 (14)	0.392 (2)	0.371 (4)	0.246 (11)	0.282 (7)	0.282 (6)	0.160 (15)
musk	0.506 (12)	**1.000 (5)**	0.090 (14)	1.000 (6)	1.000 (7)	0.818 (11)	0.450 (13)	0.856 (10)	0.043 (15)	0.998 (8)	0.994 (9)	1.000 (4)	1.000 (3)	1.000 (2)	1.000 (1)
optdigits	0.037 (11)	0.051 (6)	0.021 (14)	0.043 (10)	0.053 (5)	0.051 (7)	0.023 (13)	0.018 (15)	0.032 (12)	0.070 (3)	0.072 (2)	0.055 (4)	0.050 (8)	0.048 (9)	**0.114 (1)**
pendigits	0.195 (9)	0.279 (3)	0.044 (13)	0.255 (7)	0.258 (6)	0.211 (8)	0.035 (15)	0.083 (11)	0.036 (14)	0.182 (10)	0.066 (12)	0.266 (5)	**0.289 (1)**	0.272 (4)	0.285 (2)
pima	0.474 (10)	0.500 (5)	0.493 (6)	0.465 (11)	0.501 (4)	0.520 (2)	0.405 (13)	0.431 (12)	0.398 (14)	0.368 (15)	0.490 (7)	0.505 (3)	0.480 (9)	0.487 (8)	**0.522 (1)**
satellite	0.571 (9)	0.660 (3)	0.397 (15)	0.660 (4)	**0.682 (1)**	0.609 (7)	0.508 (12)	0.574 (8)	0.479 (13)	0.527 (11)	0.415 (14)	0.658 (5)	0.658 (6)	0.671 (2)	0.535 (10)
satimage-2	0.709 (10)	0.926 (5)	0.142 (13)	0.915 (8)	0.944 (3)	0.729 (9)	0.100 (14)	0.394 (11)	0.063 (15)	0.961 (2)	0.393 (12)	0.921 (7)	0.927 (4)	0.924 (6)	**0.965 (1)**
speech	0.026 (5)	0.018 (14)	0.020 (9)	0.027 (2)	0.021 (8)	0.026 (4)	0.019 (10)	0.023 (7)	**0.027 (1)**	0.019 (11)	0.019 (12)	0.018 (13)	0.026 (3)	0.026 (6)	0.017 (15)
thyroid	0.464 (6)	0.557 (4)	0.335 (11)	0.380 (10)	0.486 (5)	0.434 (7)	0.309 (12)	0.146 (14)	0.049 (15)	0.176 (13)	0.410 (8)	0.612 (2)	**0.651 (1)**	0.604 (3)	0.406 (9)
vertebral	0.095 (8)	0.096 (7)	0.088 (13)	0.092 (11)	0.087 (14)	0.090 (12)	0.116 (3)	0.144 (2)	0.103 (4)	**0.156 (1)**	0.000 (15)	0.098 (6)	0.094 (9)	0.093 (10)	0.099 (5)
vowels	0.169 (7)	0.138 (11)	0.385 (3)	0.106 (13)	0.228 (5)	0.192 (6)	0.458 (2)	0.067 (14)	0.057 (15)	0.140 (10)	0.369 (4)	0.127 (12)	0.155 (9)	0.165 (8)	**0.491 (1)**
wbc	0.612 (6)	0.608 (7)	0.650 (3)	0.582 (11)	0.597 (8)	0.616 (5)	0.241 (15)	0.415 (14)	0.447 (13)	0.596 (9)	0.542 (12)	0.591 (10)	0.639 (4)	**0.668 (1)**	0.665 (2)
wine	0.246 (5)	0.213 (10)	0.290 (2)	0.234 (6)	0.253 (4)	0.272 (3)	0.093 (13)	0.112 (12)	0.081 (14)	0.220 (7)	0.000 (15)	0.211 (11)	0.215 (8)	0.215 (9)	**0.364 (1)**
Average	**0.337**	**0.398**	**0.330**	**0.392**	**0.396**	**0.367**	**0.276**	**0.275**	**0.208**	**0.347**	**0.348**	**0.408**	**0.413**	**0.414**	**0.422**
Std Dev	**0.243**	**0.308**	**0.248**	**0.302**	**0.300**	**0.264**	**0.223**	**0.227**	**0.171**	**0.285**	**0.267**	**0.313**	**0.294**	**0.297**	**0.283**

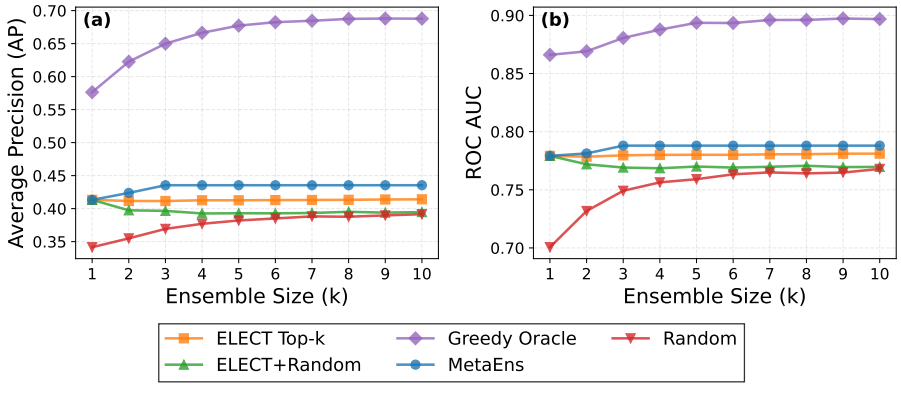

Figure A.2. Ensemble size impact on performance (AP) and robustness (ROC AUC). **(Left)** Average Precision vs ensemble size shows that `MetaEns` achieves rapid performance gains with small ensembles ($k \leq 3$), then plateaus, confirming the effectiveness of the family-risk regularizer in adaptive ensemble stopping. **(Right)** ROC AUC demonstrates consistent robustness across different ensemble sizes. Greedy Oracle (with oracle access to test labels) provides an upper bound, while Random selection serves as the lower bound. ELECT performs comparably to single models ($k = 1$) across ensemble sizes due to its ranking-based selection strategy.

Confidential reviewer copy. This manuscript is under double-blind review by ICML 2026. Unauthorized sharing, redistribution, or disclosure is strictly prohibited.