SEAD : Unsupervised Ensemble of Streaming Anomaly Detectors

Saumya Gaurang Shah [1] **Abishek Sankararaman** [1] **Balakrishnan (Murali) Narayanaswamy** [1]
Vikramank Singh [1]

Abstract

Can we efficiently choose the best Anomaly Detection (AD) algorithm for a data-stream without requiring anomaly labels? Streaming anomaly detection is hard. SOTA AD algorithms are sensitive to their hyperparameters and no single method works well on all datasets. The best algorithm/hyper-parameter combination for a given data-stream can change over time with data drift. 'What is an anomaly?' is often application, context and dataset dependent. We propose SEAD (Streaming Ensemble of Anomaly Detectors), the first model selection algorithm for streaming, unsupervised AD. All prior AD model selection algorithms are either supervised, or only work in the offline setting when all data from the test set is available upfront. We show that SEAD is *(i)* unsupervised, i.e., requires no true anomaly labels, *(ii)* efficiently implementable in a streaming setting, *(iii)* agnostic to the choice of the base algorithms among which it chooses from, and *(iv)* adaptive to non-stationarity in the data-stream. Experiments on 14 non-trivial public datasets and an internal dataset corroborate our claims.

1. Introduction

Detecting anomalies on noisy, high-dimensional and non-stationary data-streams is becoming an important subroutine in several applications such as monitoring (AWS, 2025; Hagemann & Katsarou, 2020), fault-detection (Singh et al., 2024; Islam et al., 2021), intrusion detection (Lazarevic et al., 2003) and more. See the surveys of (Chandola et al., 2009; Iglesias Vázquez et al., 2023). Given this ubiquity, several anomaly detection (AD) settings and algorithms have been designed in the literature with various modeling assumptions (Ruff et al., 2021).

[1]Amazon Web Services, Santa Clara, CA, USA. Correspondence to: Saumya Gaurang Shah <saumyags@amazon.com>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

We study the *streaming unsupervised* setting, where an AD algorithm makes a decision as each data point streams in. Further, the algorithm is never given any feedback as to whether any of the inputs seen thus far were truly anomalous. This setting is increasingly common in modern cloud-computing and software monitoring systems where the volume and velocity of data is high and labeling any given event as anomalous or not is expensive in terms of human time and effort (Xu et al., 2018a;b; Sankararaman et al., 2022). Current unsupervised AD algorithms are effective at identifying anomalous events when the data satisfies their modeling assumptions such as distance preservation in low dimensional random projections (Manzoor et al., 2018), outliers increasing model complexity (Guha et al., 2016) or anomalies isolated from the center of benign data points (Ding & Fei, 2013).

However, no single model performs well on all datasets (Schmidl et al., 2022; Paparrizos et al., 2022; Han et al., 2022), and existing methods are sensitive to their hyperparameters (Zhao et al., 2021; Schmidl et al., 2022; Paparrizos et al., 2022; Han et al., 2022). For example, on the Pendigits dataset (Keller et al., 2012; Sathe & Aggarwal, 2016a), the IForestASD algorithm (Ding & Fei, 2013) obtains superior performance when compared to a simple rule based Statistical Process Control (SPC) (Shewhart, 1931) model, but the ordering is reversed on an internal Telemetry dataset (Table 7 in Sec 4). Similarly, on the Letter Recognition dataset (Slate, 1991; Micenková et al., 2014), changing the sliding window hyperparameter from 64 to 128 for the RRCF algorithm (Guha et al., 2016) increases the average precision score (APS) by $\sim 42\%$, but setting it to the default value of 256 brings down the APS by $\sim 13\%$. (cf. Table 7)

Table 1. Accuracy numbers on the 3 datasets shown in Figure 1. Each dataset has a perfect detector, but none of the 3 perfect detectors perform well on all 3 datasets. SEAD performs model selection online to yield perfect accuracy on all 3 datasets *without access to any labels*.

	DETECTOR1	DETECTOR2	DETECTOR3	SEAD
RDS_CPU_UTILIZATION_CC0C53	100%	0%	0%	100%
ELB_REQUEST_COUNT_8C0756	0%	100%	0%	100%
EC2_NETWORK_IN_257A54	0%	0%	100%	100%

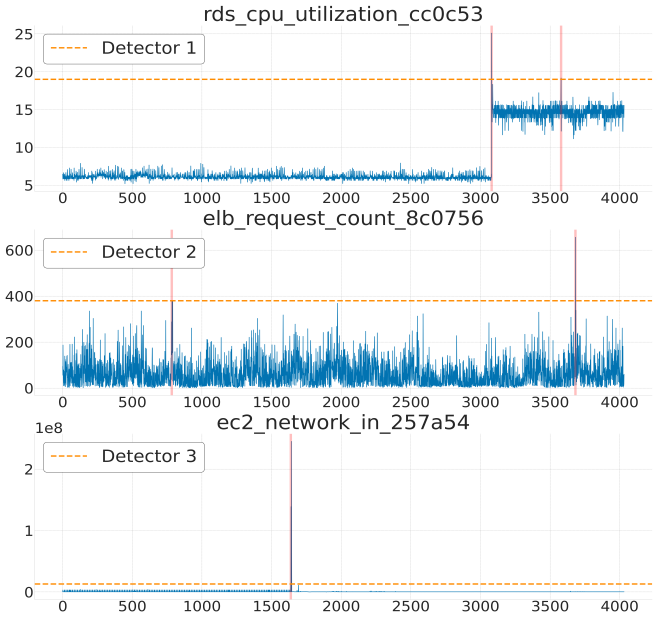

Figure 1. Three AD datasets from the Numenta Anomaly Detection Benchmark (Ahmad et al., 2017) (NAB), each of which has a perfect but different detector shown by the orange horizontal dotted line. However, each detector performs poorly on the other 2 datasets. SEAD is the perfect detector for all 3 datasets, as shown in Table 1

SEAD provides an algorithmic solution to a key challenge encountered in practice – the choice of which model to use when, especially when labels are rare or unavailable, and decision needs to be made in a streaming fashion. The vast literature on AD suggests that there are good models for most situations one encounters in practice. However, the assumption that one knows the distribution of data in advance is becoming an unrealistic one, especially in modern cloud-computing systems that serve a variety of workloads (Zhang et al., 2021; Singh et al., 2024).

1.1. Summarizing our contributions

We propose SEAD , the first online model selection algorithm for AD, that is both completely unsupervised, and works online as the data streams in. In particular, SEAD is *(i)* unsupervised and does not require anomaly labels, *(ii)* efficiently implementable online, i.e., the computational complexity to process each data-point on the stream does not depend on the stream length, *(iii)* agnostic to the choices of the base algorithms among which it chooses from, and *(iv)* adaptive to non-stationarity in the data-stream, i.e., the optimal choice of the algorithm changes as the data-stream's distribution changes.

1.2. Design principle behind SEAD

The key insight that SEAD leverages is that anomalies by definition are 'rare', which SEAD uses to work in a fully unsupervised fashion. Anomaly detection algorithms output an *anomaly score* for each input, with the convention that larger the score, more likely the input is anomalous (Schmidl et al., 2022). In SEAD , we exploit the observation that *a candidate AD algorithm is 'good' on a data-stream if it has consistently output small anomaly scores in the past, and similarly is 'poor' if it has output larger scores in the past.* This observation builds on the assumption that anomalies on any data-stream, by definition are rare. SEAD maintains a weight for each candidate AD algorithm with algorithms that have consistently output lower scores having higher weight and vice-versa. When a new data point comes in, SEAD outputs as its anomaly score the weighted combination of anomaly scores from the individual models.

The main technical contribution in SEAD is a methodology to set the weights for the individual models. SEAD chooses the weight using the classical multiplicative weights update (MWU) (Cesa-Bianchi et al., 1997) to predict with expert advice. We treat each candidate AD algorithm as an expert and treat both the advice and the loss incurred by an expert as its output anomaly score. This formulation ensures that a candidate AD algorithm that consistently outputs smaller anomaly scores, will have a higher weight. The complete pseudo code is in Algorithm 1.

We motivate our intuition of down weighing detectors with high scores in Figure 1, using results from 3 simple univariate datasets in the Numenta Anomaly Detection Benchmark (NAB) (Ahmad et al., 2017). Most NAB datasets are easily solvable by rules (Wu & Keogh, 2020), including the three datasets that we consider. Each dataset has a perfect anomaly detector of the form $y > a$ with a different value of the parameter a for each dataset, as shown in Table 1 and Figure 1. We show that SEAD is able to select the best detector in each case. Later, we compare with state-of-the-art streaming AD methods on more complex datasets, and show that SEAD obtains the best performance using both quantitative metrics and qualitative examples. SEAD is only as good as its individual components, so it needs at least one model to perform well at detecting anomalies on the given dataset i.e. the underlying data must satisfy assumptions of at least one base model.

An advantage of SEAD is its computational efficiency on a data-stream. Existing outlier ensembles cannot be used in the streaming setting because, they use computationally expensive $\mathcal{O}(n^2)$ operations like ranking the anomaly scores across time (Rayana & Akoglu, 2014; 2016) or using all data points to compute agreements between base detectors (Rayana et al., 2016), where n is the number of data points seen so far. Similarly, meta learning based model selection

methods (Ding et al., 2024; Zhao et al., 2021; 2022) also use expensive $\mathcal{O}(n^2)$ operations, looking at each historical data point. On the Ozone dataset (Zhang et al., 2008), MetaOD (Zhao et al., 2021) takes roughly 2 days to make predictions on a data stream of about 2,534 data points, often taking > 1 minute for a single data point. Ozone sensors should have low detection times of < 1 min because ozone exposure can have serious health consequences (Barreto et al., 2022). Our method SEAD takes only ~ 1746 seconds on the Ozone dataset, taking less than half a minute on average and processing each data point in $\mathcal{O}(1)$ time. SEAD ++ further improves runtime, taking only ~ 945 seconds with comparable detection performance.

Organization Section 2 describes the problem statement and the formal description of SEAD is in Section 3. Experimental results are in Section 4 followed by related work in Section 5 and conclusions in Section 6.

2. Problem Setting

Online data-stream. We consider the problem of online anomaly detection, where at each time $t = 1, 2, \cdots$, a data-point $X_t \in \mathcal{X}$ is presented to an algorithm which returns a non-negative score $S_t \in \mathbb{R}_+$ with the interpretation that larger the score, more anomalous the input. Throughout, we will assume that each data-point is an element of the set $\mathcal{X}$, which could be numerical, or a mix of numerical and categorical or even a mix of numerical and free-form text. The exact description of $\mathcal{X}$ is not crucial for our exposition. Additionally, associated with each data point X_t, is a binary label $Y_t \in \{0, 1\}$ with the interpretation that $Y_t = 1$ implies that X_t is anomalous. The AD algorithm does not have access to this label, nor can it get access to this over time.

Base AD algorithms. We assume that we have access to N base algorithms, denoted by $A_t^{(1)}(\cdot), \cdots, A_t^{(N)}(\cdot)$, where $S_t^{(i)} := A_t^{(i)}(X_t)$ is the anomaly score output by algorithm i on input X_t. Each base algorithm is also indexed by time t, allowing it to change over time, e.g. through incremental or online learning.

Online ensembling. The goal of our system is to predict an anomaly score $S_t \in \mathbb{R}_+$ at each time t for the input X_t, based only on the historical information seen so far. At each time t, the input X_t is scored by each of the N base algorithms to give the N individual scores $\{S_t^{(i)}\}_{i=1}^N$. The ensembling algorithm then outputs a single anomaly score $S_t \in \mathbb{R}_+$ by taking these N scores as input. The ensembling algorithm can be 'stateful', i.e., can depend on the past and present inputs $(X_s)_{s \le t}$, past and present base anomaly scores $\{S_s^{(i)}\}_{i \in [N], s \le t}$ and past outputs $\{S_s\}_{s<t}$. The algorithm however must be *(i)* online, i.e., produce the output S_t before observing the input X_{t+1}, and *(ii)* be unsupervised, i.e., Y_t is never revealed.

Performance Metric. The metric of interest for the algorithm $\mathcal{A}$ is the *Averaged-Precision* score (APS), which is informative of AD performance given the imbalanced nature of anomalies being only a small fraction of all datapoints (Ruff et al., 2021). We use this metric as it has been well documented that for most applications of AD, the APS is a meaningful metric of performance in practice compared to other binary classification metrics such as AUROC. However, note we cannot directly optimize the APS since the true anomaly labels are never revealed.

3. The SEAD Algorithm

The challenge in designing an ensembling algorithm is the fact that the true anomaly labels $Y_1, \cdots, Y_T$ are *never* revealed to the algorithm. This might lead one to conclude that the problem of online unsupervised aggregation cannot be solved. We show that if the data-stream is such that *anomalies are very rare for at least one of the base detectors*, then SEAD given in Algorithm 1 leverages this property to get an unsupervised anomaly aggregation algorithm with good performance.

Recall that the goal of an anomaly detection algorithm is to output smaller scores for non-anomalous inputs and larger scores for anomalous inputs. On a data-stream that has a very small number of anomalies, a good anomaly detection algorithm is one that usually outputs smaller scores. We leverage this observation and use the unsupervised loss of directly minimizing the anomaly score as a proxy for the supervised loss of APS in SEAD .

3.1. Unsupervised ensembling

Our proposed methodology is to output as anomaly score at time t, $S_t := \sum_{i=1}^N w_t^{(i)} S_t^{(i)}$, where $\sum_{i=1}^N w_t^{(i)} = 1$ and $w_t^{(i)} \ge 0$ for all i. Ideally, we want the weight $w_t^{(i)}$ to be large if algorithm i is 'good' and small when algorithm i is 'poor'. Since in our setting, the true binary labels are never revealed, we rely on the following unsupervised proxy loss to set the weights. An algorithm that has consistently output smaller anomaly scores in the past is likely to be superior and thus should be assigned a higher weight.

3.2. Normalizing the anomaly scores

A roadblock in this proxy is that the anomaly scores across different base algorithms may not be comparable. For example consider two different algorithms – the first one outputs the scores in the range $[1, 2]$, while the second one only outputs in the range $[0, 1]$. If we directly use the anomaly scores as the loss for each algorithm, the first algorithm will always get a lower weight compared to the second one, no matter the relative performance as measured by APS.

In SEAD , we circumvent this issue by normalizing each AD algorithm's output to be the quantile of the anomaly score according to the empirical distribution of the past scores, i.e., use Quantile $\left(S_t^{(i)}, \frac{1}{t}\sum_{j=1}^{t}\delta_{S_j^{(i)}}\right)$ (Line 8 in Algorithm 1). Henceforth, we refer to $S_t^{(i)} \in [0,1]$ to be the *normalized* anomaly score output at time t, by base algorithm i. The streaming quantiles can be efficiently implemented for example using t-digest data-structure (Dunning & Ertl, 2019). For larger data-streams, a windowed quantile that transform $S_t^{(i)}$ to be the quantile of the scores only in the past W time-steps, where W is a hyper-parameter can be efficient. The implementation in Algorithm 1 corresponds to $W = \infty$.

3.3. Prediction with expert advice

In order to set the weight $w_t^{(i)}$ to be large, if the scores $(S_1^{(i)}, \cdots, S_{t-1}^{(i)})$ are small and vice-versa, we use the classical Follow-the-regularized-leader (FTRL) algorithm to predict with expert advice (Cesa-Bianchi et al., 1997; Cesa-Bianchi & Lugosi, 2006). In our setting, the anomaly scores by each algorithm is both the prediction of that expert, as well as the loss incurred by following that expert. The ensembling algorithm observes the N expert predictions $(S_t^{(1)}, \cdots, S_t^{(N)})$ and outputs a score S_t as a convex combination of the N expert's predictions.

Formally, the FTRL primitive is to choose the weights $\mathbf{w}_t := [w_t^{(1)}, \cdots, w_t^{(n)}]$ at time t to be the minimizer of the sum of historical performance and regularizer as $\mathbf{w}_t^* := \arg\min_{w \in \Delta^{N-1}} \sum_{s=1}^{t-1} \langle \mathbf{w}, \mathbf{S}_s \rangle + \mathcal{R}_t(\mathbf{w})$, where Δ^{N-1} is the simplex on N dimensions and $\mathcal{R}_t(\cdot)$ is a non-negative regularization function.

3.4. Putting it together

SEAD employs the KL divergence as the regularizer, which penalizes deviation of the weights $\mathbf{w}$ from a fixed distribution π, which is a hyperparameter of SEAD . This distribution π encodes prior information over the base AD algorithms as to which ones are better than others. For example, if there is some offline data from the stream available, one can use the offline model-selection algorithm of (Zhao et al., 2021), to inform a prior distribution for the weights on the online stream. On the other hand, if there is no prior information, we set π as the uniform distribution, which reduces the KL divergence regularizer to the negative entropy of the weights $\mathbf{w}$, which corresponds to the classical multiplicative-weights update (MWU) algorithm. This is beneficial since it is well known in online learning literature that MWU is optimal (upto constant factors) in the absence of any prior information (Cesa-Bianchi et al., 1997).

Further to reduce complexity, *(i)* we set the weights at time t denoted by $\mathbf{w}_t$ by taking one gradient step from the previous weights $\mathbf{w}_{t-1}$, and *(ii)* re-parametrize $\mathbf{w}$ to be unconstrained using the softmax transform. In particular, we set the weights $\mathbf{w}_t \in \mathbb{R}^N$ and output the score (Line 10 of Algorithm 1)

$$S_t := \sum_{i=1}^{N} \frac{\exp(w_t^{(i)})}{\sum_{j=1}^{N} w_t^{(j)}} \cdot S_t^{(i)},$$

and update the weights using a single step of gradient descent (Line 13 of Algorithm 1)

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta \nabla_{\mathbf{w}} L_t(\mathbf{w})|_{\mathbf{w}=\mathbf{w}_t}, \tag{1}$$

where the loss function (Line 12 of Algorithm 1)

$$L_t(\mathbf{w}) \leftarrow \sum_{i=1}^{N} \left(\frac{\exp(w_i)}{\sum_{j=1}^{N} \exp(w_j)} S_t^{(i)} \right) + \lambda \mathrm{KL}\left(\frac{\exp(\mathbf{w})}{\sum_{j=1}^{N} \exp(w_j)} \middle\| \pi \right). \tag{2}$$

Observe that this loss function ensures that the final anomaly score output is minimized while ensuring that the weight distribution does not deviate too much from the prior information (if any) that is available.

Algorithm 1 SEAD Algorithm

1: **Input** Streaming data $\{\mathbf{x}_1, \mathbf{x}_2, \cdots\}$, learning rate $\eta > 0$, regularization $\lambda \geq 0$, prior distribution $\pi \in \Delta^{N-1}$
2: **Initialize** weights $\mathbf{w}_1 \leftarrow [1, \cdots, 1] \in \mathbb{R}^N$,
3: **Initialize** base AD detectors $A_1^{(i)}(\cdot)$, for all $1 \leq i \leq N$
4: **for** Time $t = 1, 2, \cdots$ **do**
5: Receive input $\mathbf{x}_t$
6: **for** each base detector $i \in \{1, \cdots, N\}$ **do**
7: $\widetilde{S}_t^{(i)} = A_t^{(i)}(\mathbf{x}_t)$
8: $S_t^{(i)} \leftarrow \text{Quantile}\left(\widetilde{S}_t^{(i)}; \frac{1}{t}\sum_{u=1}^{t}\delta_{\widetilde{S}_u^{(i)}}\right)$
9: **end for**
10: **Output** Score $S_t \leftarrow \sum_{i=1}^{N}\left(\frac{\exp(w_t^{(i)})}{\sum_{j=1}^{N}\exp(w_t^{(j)})} S_t^{(i)}\right)$
11: Update weights according to Equations (1) and (2).
12: $A_{t+1}^{(i)}(\cdot) \leftarrow \text{Update-Algo-}i(A_t^{(i)}(\cdot), \mathbf{x}_t), \forall i \in [N]$
13: **end for**

3.5. Online update to the base algorithms

In SEAD we allow for the N base-algorithms to also be incremented with each incoming sample $\mathbf{x}_t$: for example, a standard auto-encoder could be updated with a single gradient descent step as in (Mirsky et al., 2018), or an isolation forest can be incremented using a custom tree-update algorithm as in (Leveni et al., 2024). SEAD itself is agnostic to the update mechanism, except for the fact that the updates need to be unsupervised, i.e., the true anomaly label information is not known.

Algorithm 2 SEAD ++: Optimizing runtime by sampling

1: **Input:** Streaming data $\{\mathbf{x}_1, \mathbf{x}_2, \cdots\}$, learning rate $\eta > 0$, regularization strength $\lambda \geq 0$, prior distribution $\pi \in \Delta^{N-1}$, $K \in \{1, \cdots, N\}$
2: **Initialize** weights $\mathbf{w}_1 \leftarrow [1, \cdots, 1] \in \mathbb{R}^N$,
3: **Initialize** base AD detectors $A_1^{(i)}(\cdot)$, for all $1 \leq i \leq N$
4: **for** Time $t = 1, 2, \cdots$ **do**
5: Receive input $\mathbf{x}_t$
6: $E_t \leftarrow$ a set of K detectors sampled without replacement with detector $i \in [N]$ sampled with probability proportional to $\exp(w_t^{(i)})$.
7: **for** each base detector $i \in E_t$ **do**
8: $\widetilde{S}_t^{(i)} = A_t^{(i)}(\mathbf{x}_t)$
9: $\Lambda_t^{(i)} := \{1 \leq s \leq t \text{ s.t. } i \in E_t\}$
10: $S_t^{(i)} \leftarrow \text{Quantile}\left(\widetilde{S}_t^{(i)}; \frac{1}{|\Lambda_t^{(i)}|} \sum_{u \in \Lambda_t^{(i)}} \delta_{\widetilde{S}_u^{(i)}}\right)$
11: **end for**
12: **Output** Score $S_t \leftarrow \sum_{i \in E_t} \left(\frac{\exp(w_t^{(i)})}{\sum_{j \in E_t} \exp(w_t^{(j)})} S_t^{(i)}\right)$
13: Update weights according to Equations (3) and (4).
14: $A_{t+1}^{(i)}(\cdot) \leftarrow$ Update-Algo-$i(A_t^{(i)}(\cdot), \mathbf{x}_t), \forall i \in E_t$
15: **end for**

3.6. Theoretical Guarantees

Since SEAD is built on the FTRL abstraction, it also inherits several desirable properties of FTRL. In particular, SEAD enjoys small 'regret', namely the sum of all anomaly scores output by SEAD , is no more than $O(\sqrt{T \ln(N)})$ compared to the sum of anomaly scores output by *any* base-algorithm. When the parameters of learning rate and regularization strength are appropriately chosen, the MWU algorithm, and as a consequence SEAD , is adaptive to distribution shifts

Theorem 3.1 (Single best in hindsight, (Cesa-Bianchi & Lugosi, 2006))**.** *The difference between the expected cumulative sum of anomaly scores output by our algorithm and any base AD algorithm is no more than* $O\left(\sqrt{T \ln(N)}\right)$.

However, this result does not say that the performance of SEAD as measured by APS, which requires labels matches that of the best base-algorithm. Nevertheless, this theorem indicates why it is possible for SEAD to be as competitive as the best possible AD algorithm in hindsight, which our experiments corroborate.

3.7. SEAD ++: Runtime optimization by sampling base AD Algorithms

A drawback in SEAD as given in Algorithm 1 is that at every time t, all N base algorithms provide an anomaly-score and are incrementally updated. The computational cost can be prohibitive for large N, which can occur when different hyper-parameter configurations correspond to different base-detectors. We propose a modification to SEAD in Algorithm 2, which only considers the anomaly scores and incremental updates to K out of N algorithms at each time. The design of Algorithm 2 is motivated by the classical Thompson sampling algorithm (Russo et al., 2018), originally proposed to solve the special case when exactly one out of the N base detectors can be chosen at each time. Formally, Algorithm 2 takes $K \leq N$ as input and at each time t, samples a subset E_t consisting of K out of the N detectors. This subset is sampled without replacement from the set of N detectors with each detector i sampled with probability proportional to $\exp(w_t^{(i)})$ (Line 6 of Algorithm 2). The output S_t is given by a weighted combination of the scores by the detectors in E_t (line 11 of Algorithm 2). The weight vector $\mathbf{w}_{t+1}$ is obtained by a gradient step

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta \nabla_{\mathbf{w}} \widehat{L}_t(\mathbf{w}, E_t)|_{\mathbf{w}=\mathbf{w}_t}, \tag{3}$$

on a modified loss function $\widehat{L}_t(\mathbf{w}, E_t)$ that accounts for the sub-sampling of detectors

$$\widehat{L}_t(\mathbf{w}, E_t) \leftarrow \sum_{i \in E_t} \left(\frac{\exp(w_i)}{\sum_{j \in E_t} \exp(w_j)} S_t^{(i)}\right) + \lambda \text{KL}\left(\frac{\exp(\mathbf{w})}{\sum_{j=1}^N \exp(w_j)} \middle|\middle| \pi\right). \tag{4}$$

Observe that when $K = N$, Algorithm 2 is identical to Algorithm 1. The main savings in computation in Algorithm 2 comes in lines 8 and 14, where only K detectors are queried for an anomaly score and incrementally updated at each time. In contrast, Algorithm 1 queries anomaly scores and incrementally updates all N algorithms at each time.

The main reduction of runtime in SEAD ++ is in not needing to do forward-pass/inference and backward pass/gradient calculation on $K - N$ AD models. For many big models, these forward and backward passes are the main computational bottlenecks. While SEAD ++ does require maintaining weights for each base detector, the updates and storage for the weights are trivial and independent of the size of the base AD models.

4. Experiments

We empirically evaluate the following research questions.

1. Is SEAD *effective*, i.e., can match the performance of the best base AD Algorithm on any data-stream?
2. Is SEAD *efficient*, i.e., has a better detection performance and runtime in the streaming setting compared to offline methods?
3. Does SEAD ++ improve over the runtime of SEAD and, how does it affect detection performance?

In addition, we also show an anecdotal case study to further give intuition on how SEAD works.

4.1. Experimental Setup

We use 4 streaming anomaly detectors in our base models: IForestASD (IF) (Ding & Fei, 2013), xStream (XS) (Manzoor et al., 2018), Robust Random Cut Forest (RRCF) (Guha et al., 2016) and a simple rule based model based on Statistical Process Control (SPC) (Shewhart, 1931). For IForestASD, xStream and RRCF, we use open source implementations from PySAD (Yilmaz & Kozat, 2020). For the rule based model, we compute anomaly score for a D dimensional incoming data point $\mathbf{x} = [x_1, \cdots, x_D]$ as $\frac{1}{D}\sum_{i=1}^{D} \frac{x_i - \mu_i}{\sigma_i}$, where μ_i and σ_i are the mean and standard deviation for the i^{th} dimension. We initialize each of IForestASD, xStream and RRCF with 4 different hyperparameter configurations including the default configuration (please refer to Appendix C for more details). Since we focus on the streaming setting in this work, we only perturb parameters related to number of past data points stored in memory. We choose other parameters as recommended in their open source implementations. The rule based model does not have any parameters. We get a total of $N = 13$ base models. For our method SEAD , we choose hyperparameters $\eta = 1, \lambda = 10^{-6}$ and $\pi = Uniform$ distribution across all experiments. We use tdigest (Dunning & Ertl, 2019) for streaming quantile computation for normalizing anomaly scores. We set the first 100 data points for warm starting the base models and SEAD , but not for evaluation, i.e., is 'cold start'. We performed all experiments on a single c5.2xlarge AWS EC2 instance.

4.2. Datasets considered

We perform experiments on 15 datasets, of which 11 are from the Outlier Detection DataSets (ODDS) (Rayana, 2016), 3 are from the USP Data Stream Repository (Souza et al., 2020) and one is an internal telemetry dataset from a multiserver database cloud service. Our choice of datasets cover diverse applications and use-cases, including the INSECTS dataset (Souza et al., 2020) which has non-stationarity, a private telemetry dataset that has only 0.03% of its stream as anomalies and the IONOSPHERE dataset (Rayana, 2016; Keller et al., 2012) which has 36% of its stream as anomalies. Dataset characteristics are in Table 10 in Appendix B. The number of data points range from 214 to $567,000$ and the number of features vary from 1 to 71. We report the rankings of the base models and SEAD in Table 2 based on the AP metric.

4.3. Effectiveness of SEAD

We first observe that our datasets are non-trivial, i.e., no single algorithm performs well across all of these datasets (see Table 2 and Appendix A). RRCF outperforms other methods on the Letter, Glass and Telemetry datasets. xStream outperforms on the Ozone, Optdigits and INSECTS datasets. IForestASD is the best method on the Pendigits, Cardio and Ionosphere datasets. Surprisingly, the rule based method performs the best on multiple datasets: Pima, Mammography, HTTP and Telemetry. All base methods show sensitivity to their hyperparameters. Examples include RRCF on WBC, xStream on Pendigits and IForestASD on Optdigits datasets. The sensitivity to model selection and hyperparameter configurations underscores the need for an ensemble model that can appropriately aggregate base models depending on the dataset.

From Table 2, we see that SEAD obtains the best rank on average, outperforming about 60% of the base models and having the lowest variance. In the worst case, it ranks 9^{th} among all models. For all other base models, the rank in the worst case is $>= 10$, with most base models achieving the worst rank on at least one dataset. This demonstrates that even on the worst dataset, the performance of SEAD is good, while all other algorithms have at-least one dataset in which their performance is bad.

We also compare with simple ensemble baselines that aggregate base detectors using (a) mean, (b) max or (c) min of the anomaly scores in Tables 3 and in Appendix A. We only compare against simple baselines because SEAD is the first unsupervised online ensembling technique. SEAD outperforms simple ensemble baselines, achieving a higher rank on average with the lowest variance. We later show that SEAD ++ achieves detection performance comparable to these simple ensemble baselines, while reducing the runtime by half.

We evaluate the statistical significance of our results using Wilcoxon signed-rank test, concluding that our method is statistically different from all base models and ensemble baselines (in Appendix H). Using the same methodology for computing statistical significance, we show critical difference diagram for SEAD, the best performing variants of each base method and simple ensemble baselines in Figure 2. We report hyperparameter ablations in Appendix D.

4.4. Comparison with offline model selection method

Offline model selection methods cannot be used in the streaming setting, which requires processing each incoming data point in constant time. Nevertheless, for comparison, we relax the streaming requirement. We allow the offline meta learning method to look back at all of the data points seen so far, allowing $\mathcal{O}(n)$ computation instead of the $\mathcal{O}(1)$ computation time required in the streaming setting.

We compare with the offline model selection algorithms MetaOD (Zhao et al., 2021) and the method described in (Goswami et al., 2023). We evaluate on an open source dataset on which neither of the offline algorithms has been trained on: Ozone dataset (Zhang et al., 2008) from the USP

Table 2. Comparison with base models: Ranks on real world datasets

INDEX	SPC	RRCF_0	RRCF_1	RRCF_2	RRCF_3	XS_0	XS_1	XS_2	XS_3	IF_0	IF_1	IF_2	IF_3	SEAD
PIMA	1	13	12	8	10	11	9	4	5	3	6	7	5	2
PENDIGITS	13	10	11	9	11	12	6	5	8	4	2	3	1	7
LETTER	9	2	6	1	3	9	11	5	7	6	12	8	10	4
OPTDIGITS	10	8	7	9	13	6	3	1	2	9	12	11	4	5
IONOSPHERE	14	8	9	5	7	13	10	11	12	4	2	1	3	6
WBC	1	14	10	12	11	13	8	7	5	2	3	4	6	9
MAMMOGRAPHY	1	13	12	11	7	10	8	9	6	4	2	5	3	6
GLASS	11	9	1	2	3	13	4	12	10	6	8	7	9	5
VERTEBRAL	12	2	3	5	4	11	1	7	6	9	9	10	9	8
CARDIO	10	13	14	12	7	8	9	6	11	4	2	1	3	5
HTTP	1	11	11	11	12	10	7	3	2	8	9	6	4	5
INSECTS GRADUAL_IMBALANCED	11	7	9	5	8	10	2	2	1	7	4	6	6	3
INSECTS INCREMENTAL_IMBALANCED	11	9	4	5	7	12	6	2	3	10	9	8	7	1
OZONE	7	4	5	6	10	1	3	4	2	10	8	9	8	9
TELEMETRY	1	1	1	2	1	5	1	3	3	7	6	8	4	1
AVERAGE RANK	7.53	8.27	7.67	6.87	7.60	9.60	5.87	5.40	5.53	6.20	6.27	6.27	5.47	**5.07**
VARIANCE	25.41	18.78	17.38	13.84	13.11	11.54	11.12	10.83	12.27	6.89	13.07	9.07	7.12	**6.64**

Table 3. Comparison with ensemble baselines: Ranks on real world datasets

INDEX	MEAN	MAX	MIN	SEAD (OURS)
PIMA	2	2	2	2
PENDIGITS	5	5	12	7
LETTER	11	9	9	4
OPTDIGITS	8	6	13	5
IONOSPHERE	4	9	7	6
WBC	9	9	11	9
MAMMOGRAPHY	7	7	9	6
GLASS	9	9	9	5
VERTEBRAL	10	5	4	8
CARDIO	5	5	7	5
HTTP	9	4	12	5
INSECTS GRADUAL_IMBALANCED	1	3	1	3
INSECTS INCREMENTAL_IMBALANCED	1	1	1	1
OZONE	4	7	1	9
TELEMETRY	1	6	1	1
AVERAGE RANK	5.73	5.80	6.60	**5.07**
VARIANCE	12.35	6.74	20.69	**6.64**

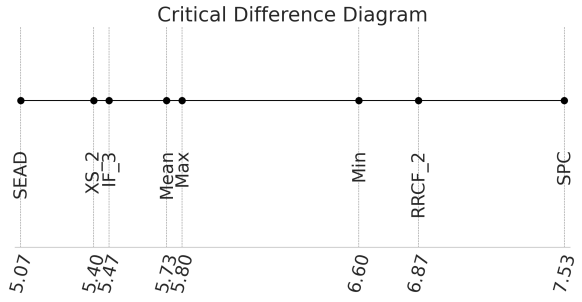

Figure 2. Critical difference diagram for SEAD, the best performing variants of each base method and simple ensemble baselines based on rank comparisons in Table 2 and Table 3. The values on the x axis represent the average ranks. All shown methods are statistically different from each other. Appendix C describes hyperparameter configurations for each base model.

dataset repository (Souza et al., 2020). We compare APS and runtime vs SEAD in Table 5. SEAD slightly improves over MetaOD and (Goswami et al., 2023) on detection performance, while providing 35x speedup in runtime. We describe the experimental details for adapting (Goswami et al., 2023)'s method to the online setting in Appendix G

4.5. Comparison with SEAD ++

We choose the number of sampled detectors $K = \lfloor N/2 \rfloor = 6$. We report ranks based on AP scores and runtime in seconds for SEAD ++ in Table 4. Note that SEAD and the ensemble baselines have approximately the same runtime because they differ in computationally trivial operations like taking mean, min and max or a linear sum across all N base anomaly detector scores. SEAD ++ obtains AP score comparable to the best ensemble baselines computed using

Table 4. Comparison with SEAD ++ ($K = 6$)

	RUNTIME (S)		AP RANK	
DATASET	SEAD	SEAD++	SEAD	SEAD++
PIMA	817.58	318.45	2	2
PENDIGITS	3680.16	1924.70	7	6
LETTER	1209.85	570.29	4	5
OPTDIGITS	3140.09	1554.65	5	6
IONOSPHERE	332.99	164.82	6	6
WBC	353.20	170.91	9	9
MAMMOGRAPHY	5567.67	2877.58	6	8
GLASS	199.60	91.06	5	6
VERTEBRAL	224.47	98.82	8	8
CARDIO	1334.02	648.94	5	10
HTTP	274102.52	116252.01	5	10
INSECTS GRADUAL_IMBALANCED	24000.80	12536.13	3	2
INSECTS INCREMENTAL_IMBALANCED	77740.12	38245.36	1	2
OZONE	1746.19	1002.05	9	6
TELEMETRY	2235.19	969.20	1	1
AVERAGE RANK	-	-	5.07	5.8
VARIANCE	-	-	6.64	8.74

Table 5. Comparison with offline model selection methods on the Ozone dataset

AVERAGE PRECISION			RUNTIME (S)		
SEAD	(ZHAO ET AL., 2021)	(GOSWAMI ET AL., 2023)	SEAD	(ZHAO ET AL., 2021)	(GOSWAMI ET AL., 2023)
0.059	0.052	0.070	1746	173684	61543

the mean and max of the anomaly scores, while providing a $\sim$ 2x speedup in runtime. Compared to SEAD , it trades off detection performance with reduced runtime. We report APS scores for SEAD ++ in Appendix A and comparisons with $K = \lfloor N/4 \rfloor$ and $K = \lfloor 3N/4 \rfloor$ in Appendix D.

4.6. Case study illustrating the evolution of weights over time of the base-algorithms in SEAD

Figure 3 shows an example where SEAD reduces the weight for a misfiring detector on the Ionosphere dataset (Sigillito et al., 1989; Keller et al., 2012). The red lines mark the ground truth labeled anomalies. The top plot shows the anomaly scores for the two models having the highest weight during this duration - IForestASD and xStream, and the bottom plot shows the weights assigned by SEAD to these two models. During the anomalous period till index 255, both IForestASD and xStream have similar scores. SEAD assigns a higher weight to xStream during this period. xStream outputs higher scores even after the anomalous period. The xStream scores on the benign data points are even higher than the xStream scores on the anomalous data points. Due to xStream misfiring on benign data points, SEAD downweights xStream and increases the weight for IForestASD. This behavior results in fewer false positives.

4.7. Robustness to uninformative base detectors

An advantage of SEAD is that its comparison against baselines like the mean aggregator improves with the addition of many random detectors. We perform an experiment where we double the number of base detectors to $N = 26$, adding 13 random detectors that each gives a random number as anomaly score to each input. We evaluate on the ODDS datasets, excluding the larger HTTP dataset because of computational reasons. As we can see in Table 6 the performance of SEAD does not diminish a lot while that of baselines like mean, max and min deteriorates significantly. The reason being that SEAD quickly identifies the bad detectors and down-weighs them, while baselines like mean cannot do this. When we add random detectors, the performance of SEAD is much more distinguishable than that of baselines. We report APS metrics in Appendix F.

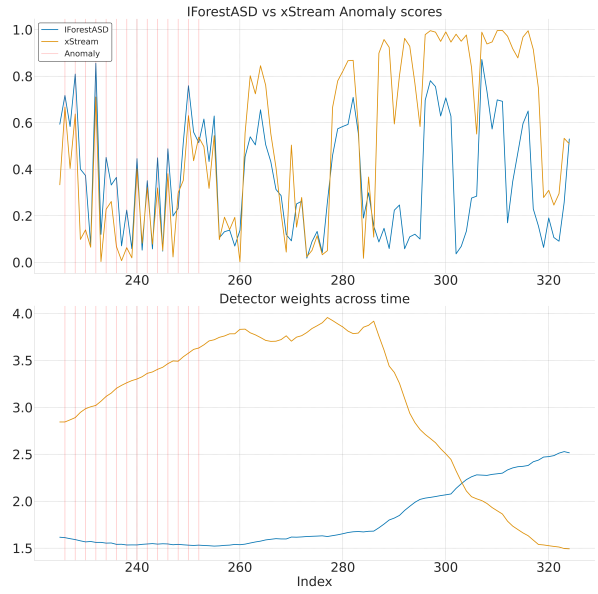

Figure 3. Evolution of weights across time for a subset of the Ionosphere (Sigillito et al., 1989; Keller et al., 2012) dataset. SEAD reduces false positives by down weighing xStream when it scores benign data points higher, and reassigning the weight to IForestASD.

Table 6. Average performance across ODDS datasets for SEAD and ensemble baselines after adding random detectors

MEAN	MAX	MIN	SEAD (OURS)
9.75%	22.31%	22.04%	0.88%

5. Related Work

5.1. Anomaly detection algorithms

AD is widely studied both in the machine learning literature as well as in applied data-science literature (see surveys of (Schmidl et al., 2022; Paparrizos et al., 2022; Iglesias Vázquez et al., 2023; Ruff et al., 2021)). Some AD algorithms are inherently ensemble based in order to be robust to dataset properties. These methods however only build ensembles of a specific class of models – an ensemble of histogram density estimators (Pevný, 2016), autoencoders (Mirsky et al., 2018), half-space chains (Manzoor et al., 2018), decision trees (Liu et al., 2008; Guha et al., 2016) and subspace outlier detectors (Sathe & Aggarwal, 2016b). On the contrary, SEAD is agnostic to the base-algorithms.

Selecting models based on new unsupervised metrics has been proposed in the literature – (Goix, 2016) proposes an extension of Excess-Mass and Mass-Volume based metrics to higher dimensions, while IREOS (Marques et al., 2020) measures how easily each outlier can be separated from other objects using maximum margin classifiers. However, it has been observed that selecting based on these unsupervised measures does not yield useful results (Ma et al., 2023).

5.2. Offline Model selection algorithms

The offline case when all data is available up-front has been studied in recent past. HYPER (Ding et al., 2024) uses a hypernetwork to predict model parameters for deep models, however it is limited only to deep models. MetaOD (Zhao et al., 2021) uses meta features to identify similar historical labeled datasets, while ELECT (Zhao et al., 2022) uses performance-driven similarity. All these algorithms need as input a collection of labeled datasets similar to the test dataset, while SEAD also works in the absence of offline labeled datasets. (Goswami et al., 2023) while proposing a state-of-the-art offline model selection method that does not require labeled offline data, nevertheless needs to rank the anomaly scores for all the test data across all models – making it inefficient to implement in the streaming setting. Ensembling algorithms (Gao et al., 2012) combine top-n outlier detection scores using outlier score normalization. (Rayana & Akoglu, 2014; 2016) use rank aggregation techniques for combining anomaly scores from base models. Other outlier ensembling methods use techniques like feature bagging (Lazarevic & Kumar, 2005), subsampling data (Zimek et al., 2013) and improved normalization (Gao & Tan, 2006; Kriegel et al., 2011).

However, all of these offline methods use the entire testing data to choose the best models, i.e., take $O(t)$ time to produce an anomaly score for the tth data point. Thus these are not computationally feasible in the streaming setting (see also Section 4).

6. Conclusions

We propose SEAD – the first unsupervised online model selection algorithm for anomaly detection. The main idea behind SEAD is to observe that the anomaly scores output by the candidate algorithms, can be used as an unsupervised proxy for its performance. SEAD ensembles anomaly scores using time-varying weights set by multiplicative weights update algorithm. We further optimized SEAD 's runtime by Thompson sampling that reduces run-time without significant accuracy degradation. We see through comprehensive experiments the value and applicability of SEAD .

Our work leaves open regret guarantees on SEAD that holds even under non-stationarity. Further optimizing SEAD to leverage offline labeled datasets to improve online selection under non-stationarity is interesting future work.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Aggarwal, C. C. and Sathe, S. Theoretical foundations and algorithms for outlier ensembles. *SIGKDD Explor. Newsl.*, 17(1):24–47, September 2015. ISSN 1931-0145. doi: 10.1145/2830544.2830549. URL https://doi.org/10.1145/2830544.2830549.

Ahmad, S., Lavin, A., Purdy, S., and Agha, Z. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2017.04.070. URL https://www.sciencedirect.com/science/article/pii/S0925231217309864. Online Real-Time Learning Strategies for Data Streams.

AWS. Using cloudwatch anomaly detection - amazon cloudwatch, 2025. URL https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch_Anomaly_Detection.html.

Barreto, D. N., Silva, W. R., Mizaikoff, B., and da Silveira Petruci, J. F. Monitoring ozone using portable substrate-integrated hollow waveguide-based absorbance sensors in the ultraviolet range. *ACS Measurement Science Au*, 2(1):39–45, 2022. doi: 10.1021/acsmeasuresciau.1c00028. URL https://doi.org/10.1021/acsmeasuresciau.1c00028.

Cesa-Bianchi, N. and Lugosi, G. *Prediction, learning, and games*. Cambridge university press, 2006.

Cesa-Bianchi, N., Freund, Y., Haussler, D., Helmbold, D. P., Schapire, R. E., and Warmuth, M. K. How to use expert advice. *Journal of the ACM (JACM)*, 44(3):427–485, 1997.

Chandola, V., Banerjee, A., and Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), jul 2009. ISSN 0360-0300. doi: 10.1145/1541880.1541882. URL https://doi.org/10.1145/1541880.1541882.

Ding, X., Zhao, Y., and Akoglu, L. Fast unsupervised deep outlier model selection with hypernetworks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, pp. 585–596, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3672003. URL https://doi.org/10.1145/3637528.3672003.

Ding, Z. and Fei, M. An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. *IFAC Proceedings Volumes*, 46(20):12–17, 2013. ISSN 1474-6670. doi: https://doi.org/10.3182/20130902-3-CN-3020.00044. URL https://www.sciencedirect.com/science/article/pii/S1474667016314999. 3rd IFAC Conference on Intelligent Control and Automation Science ICONS 2013.

Dunning, T. and Ertl, O. Computing extremely accurate quantiles using t-digests. *arXiv preprint arXiv:1902.04023*, 2019.

Gao, J. and Tan, P.-n. Converting output scores from outlier detection algorithms into probability estimates. In *Sixth International Conference on Data Mining (ICDM'06)*, pp. 212–221, 2006. doi: 10.1109/ICDM.2006.43.

Gao, J., Hu, W., Zhang, Z., and Wu, O. Unsupervised ensemble learning for mining top-n outliers. In Tan, P.-N., Chawla, S., Ho, C. K., and Bailey, J. (eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 418–430, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-30217-6.

Goix, N. How to evaluate the quality of unsupervised anomaly detection algorithms?, 2016. URL https://arxiv.org/abs/1607.01152.

Goswami, M., Challu, C. I., Callot, L., Minorics, L., and Kan, A. Unsupervised model selection for time series anomaly detection. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=gOZ_pKANaPW.

Guha, S., Mishra, N., Roy, G., and Schrijvers, O. Robust random cut forest based anomaly detection on streams. In Balcan, M. F. and Weinberger, K. Q. (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 2712–2721, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/guha16.html.

Hagemann, T. and Katsarou, K. A systematic review on anomaly detection for cloud computing environments. In *Proceedings of the 2020 3rd Artificial Intelligence and Cloud Computing Conference*, pp. 83–96, 2020.

Han, S., Hu, X., Huang, H., Jiang, M., and Zhao, Y. ADBench: Anomaly detection benchmark. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=foA_SFQ9zo0.

Iglesias Vázquez, F., Hartl, A., Zseby, T., and Zimek, A. Anomaly detection in streaming data: A comparison and evaluation study. *Expert Systems with Applications*, 233:120994, 2023. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2023.120994. URL https://www.sciencedirect.com/science/article/pii/S0957417423014963.

Islam, M. S., Pourmajidi, W., Zhang, L., Steinbacher, J., Erwin, T., and Miranskyy, A. Anomaly detection in a large-scale cloud platform. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pp. 150–159. IEEE, 2021.

Keller, F., Muller, E., and Bohm, K. Hics: High contrast subspaces for density-based outlier ranking. In *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*, ICDE '12, pp. 1037–1048, USA, 2012. IEEE Computer Society. ISBN 9780769547473. doi: 10.1109/ICDE.2012.88. URL https://doi.org/10.1109/ICDE.2012.88.

Kriegel, H.-P., Kröger, P., Schubert, E., and Zimek, A. Interpreting and unifying outlier scores. In *SDM*, 2011. URL https://api.semanticscholar.org/CorpusID:14262721.

Lazarevic, A. and Kumar, V. Feature bagging for outlier detection. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, pp. 157–166, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 159593135X. doi: 10.1145/1081870.1081891. URL https://doi.org/10.1145/1081870.1081891.

Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., and Srivastava, J. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of the 2003 SIAM international conference on data mining*, pp. 25–36. SIAM, 2003.

Leveni, F., Cassales, G. W., Pfahringer, B., Bifet, A., and Boracchi, G. Online isolation forest. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 413–422, 2008. doi: 10.1109/ICDM.2008.17.

Ma, M. Q., Zhao, Y., Zhang, X., and Akoglu, L. The need for unsupervised outlier model selection: A review and evaluation of internal evaluation strategies. *SIGKDD Explor. Newsl.*, 25(1):19–35, July 2023. ISSN 1931-0145. doi: 10.1145/3606274.3606277. URL https://doi.org/10.1145/3606274.3606277.

Manzoor, E., Lamba, H., and Akoglu, L. xstream: Outlier detection in feature-evolving data streams. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pp. 1963–1972, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355520. doi: 10.1145/3219819.3220107. URL https://doi.org/10.1145/3219819.3220107.

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), June 2020. ISSN 1556-4681. doi: 10.1145/3394053. URL https://doi.org/10.1145/3394053.

Micenková, B., McWilliams, B., and Assent, I. Learning outlier ensembles: The best of both worlds–supervised and unsupervised. 2014. ACM SIGKDD 2014 Workshop ODD2 ; Conference date: 24-08-2014 Through 24-08-2014.

Mirsky, Y., Doitshman, T., Elovici, Y., and Shabtai, A. Kitsune: An ensemble of autoencoders for online network intrusion detection. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society, 2018. URL https://www.ndss-symposium.org/wp-content/uploads/2018/02/ndss2018_03A-3_Mirsky_paper.pdf.

Paparrizos, J., Kang, Y., Boniol, P., Tsay, R. S., Palpanas, T., and Franklin, M. J. Tsb-uad: an end-to-end benchmark suite for univariate time-series anomaly detection. *Proceedings of the VLDB Endowment*, 15(8):1697–1711, 2022.

Pevný, T. Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102:275–304, 2016. URL https://api.semanticscholar.org/CorpusID:17445696.

Rayana, S. ODDS library, 2016. URL https://odds.cs.stonybrook.edu.

Rayana, S. and Akoglu, L. An ensemble approach for event detection and characterization in dynamic graphs. 2014. ACM SIGKDD ODD Workshop. 2014.

Rayana, S. and Akoglu, L. Less is more: Building selective anomaly ensembles. *ACM Trans. Knowl. Discov. Data*, 10 (4), May 2016. ISSN 1556-4681. doi: 10.1145/2890508. URL https://doi.org/10.1145/2890508.

Rayana, S., Zhong, W., and Akoglu, L. Sequential ensemble learning for outlier detection: A bias-variance perspective. *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1167–1172, 2016. URL https://api.semanticscholar.org/CorpusID:10078137.

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

Russo, D. J., Van Roy, B., Kazerouni, A., Osband, I., Wen, Z., et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

Sankararaman, A., Narayanaswamy, B., Singh, V. Y., and Song, Z. Fitness:(fine tune on new and similar samples) to detect anomalies in streams with drift and outliers. In *International Conference on Machine Learning*, pp. 19153–19177. PMLR, 2022.

Sathe, S. and Aggarwal, C. *LODES: Local Density Meets Spectral Outlier Detection*, pp. 171–179. 2016a. doi: 10.1137/1.9781611974348.20. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611974348.20.

Sathe, S. and Aggarwal, C. C. Subspace outlier detection in linear time with randomized hashing. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 459–468, 2016b. doi: 10.1109/ICDM.2016.0057.

Schmidl, S., Wenig, P., and Papenbrock, T. Anomaly detection in time series: a comprehensive evaluation. *Proc. VLDB Endow.*, 15(9):1779–1797, May 2022. ISSN 2150-8097. doi: 10.14778/3538598.3538602. URL https://doi.org/10.14778/3538598.3538602.

Shewhart, W. *Economic Control of Quality of Manufactured Product*. Books from Bell Telephone Laboratories. D. Van Nostrand Company, Incorporated, 1931. ISBN 9780598557667. URL https://books.google.com/books?id=JtVnAAAAMAAJ.

Sigillito, V., Wing, S., Hutton, L., and Baker, K. Ionosphere. UCI Machine Learning Repository, 1989. DOI: https://doi.org/10.24432/C5W01B.

Singh, V., Song, Z., Narayanaswamy, B. M., Vaidya, K. E., and Kraska, T. Vista: Machine learning based database performance troubleshooting framework in amazon rds. In *Proceedings of the 2024 ACM Symposium on Cloud Computing*, pp. 83–98, 2024.

Slate, D. Letter Recognition. UCI Machine Learning Repository, 1991. DOI: https://doi.org/10.24432/C5ZP40.

Souza, V. M. A., Reis, D. M., Maletzke, A. G., and Batista, G. E. A. P. A. Challenges in benchmarking stream learning algorithms with real-world data. *Data Mining and Knowledge Discovery*, 34:1805–1858, 2020. doi: 10.1007/s10618-020-00698-5.

Wu, R. and Keogh, E. J. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE Transactions on Knowledge and Data Engineering*, 35:2421–2429, 2020. URL https://api.semanticscholar.org/CorpusID:221995939.

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 2018 World Wide Web Conference*, WWW '18, pp. 187–196, Republic and Canton of Geneva, CHE, 2018a. International World Wide Web Conferences Steering Committee. ISBN 9781450356398. doi: 10.1145/3178876.3185996. URL https://doi.org/10.1145/3178876.3185996.

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 2018 World Wide Web Conference*, WWW '18, pp. 187–196, Republic and Canton of Geneva, CHE, 2018b. International World Wide Web Conferences Steering Committee. ISBN 9781450356398. doi: 10.1145/3178876.3185996. URL https://doi.org/10.1145/3178876.3185996.

Yilmaz, S. F. and Kozat, S. S. Pysad: A streaming anomaly detection framework in python. *arXiv preprint arXiv:2009.02572*, 2020.

Zhang, Kun, Fan, Wei, Yuan, and XiaoJing. Ozone Level Detection. UCI Machine Learning Repository, 2008. DOI: https://doi.org/10.24432/C5NG6W.

Zhang, Y., Guan, Z., Qian, H., Xu, L., Liu, H., Wen, Q., Sun, L., Jiang, J., Fan, L., and Ke, M. Cloudrca: A root cause analysis framework for cloud computing platforms. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 4373–4382, 2021.

Zhao, Y., Rossi, R., and Akoglu, L. Automatic unsupervised outlier model selection. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 4489–4502. Curran Associates, Inc., 2021. URL https://proceedings.neurips.

cc/paper_files/paper/2021/file/23c894276a2c5a16470e6a31f4618d73-Paper.pdf.

Zhao, Y., Zhang, S., and Akoglu, L. Toward Unsupervised Outlier Model Selection . In *2022 IEEE International Conference on Data Mining (ICDM)*, pp. 773–782, Los Alamitos, CA, USA, December 2022. IEEE Computer Society. doi: 10.1109/ICDM54844.2022.00088. URL https://doi.ieeecomputersociety.org/10.1109/ICDM54844.2022.00088.

Zimek, A., Gaudet, M., Campello, R. J., and Sander, J. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pp. 428–436, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450321747. doi: 10.1145/2487575.2487676. URL https://doi.org/10.1145/2487575.2487676.

A. Average Precision Metrics

We report Average Precision scores for the base models and SEAD in Table 7, ensemble baselines in Table 8 and SEAD ++ with $K = 6$ in Table 9.

Table 7. Comparison with base models: Average precision scores on real world datasets

DATASET	SPC	RRCF_0	RRCF_1	RRCF_2	RRCF_3	XSTREAM_0	XSTREAM_1	XSTREAM_2	XSTREAM_3	IFORESTASD_0	IFORESTASD_1	IFORESTASD_2	IFORESTASD_3	SEAD (OURS)
PIMA	0.616	0.400	0.430	0.461	0.445	0.442	0.455	0.499	0.492	0.501	0.491	0.488	0.492	0.519
PENDIGITS	0.016	0.070	0.069	0.076	0.069	0.039	0.092	0.115	0.085	0.244	0.262	0.254	0.292	0.091
LETTER	0.062	0.109	0.078	0.111	0.097	0.062	0.057	0.083	0.072	0.078	0.050	0.063	0.058	0.084
OPTDIGITS	0.038	0.057	0.068	0.039	0.023	0.082	0.218	0.272	0.225	0.039	0.028	0.036	0.086	0.084
IONOSPHERE	0.243	0.457	0.435	0.585	0.520	0.308	0.391	0.379	0.312	0.606	0.670	0.678	0.667	0.555
WBC	0.815	0.136	0.342	0.253	0.339	0.144	0.540	0.591	0.706	0.788	0.722	0.719	0.691	0.486
MAMMOGRAPHY	0.267	0.037	0.061	0.066	0.107	0.069	0.080	0.073	0.118	0.204	0.240	0.195	0.214	0.118
GLASS	0.062	0.100	0.188	0.179	0.145	0.057	0.116	0.058	0.068	0.113	0.101	0.104	0.100	0.115
VERTEBRAL	0.132	0.206	0.187	0.165	0.181	0.134	0.251	0.160	0.162	0.153	0.153	0.151	0.153	0.154
CARDIO	0.147	0.111	0.097	0.122	0.192	0.179	0.160	0.210	0.129	0.548	0.696	0.743	0.604	0.505
HTTP	0.425	0.004	0.004	0.004	0.003	0.005	0.019	0.071	0.099	0.013	0.007	0.024	0.051	0.035
INSECTS GRADUAL_IMBALANCED	0.111	0.152	0.141	0.156	0.151	0.140	0.165	0.165	0.178	0.152	0.157	0.154	0.154	0.161
INSECTS INCREMENTAL_IMBALANCED	0.133	0.147	0.163	0.162	0.156	0.128	0.161	0.184	0.170	0.146	0.147	0.152	0.156	0.189
OZONE	0.065	0.081	0.078	0.070	0.058	0.111	0.087	0.081	0.089	0.058	0.060	0.059	0.060	0.059
TELEMETRY	1.000	1.000	1.000	0.333	1.000	0.026	1.000	0.200	0.200	0.006	0.009	0.004	0.067	1.000

Table 8. Comparison with ensemble baselines: Average precision scores

DATASET	MEAN	MAX	MIN	SEAD (OURS)
PIMA	0.526	0.509	0.522	0.519
PENDIGITS	0.133	0.124	0.037	0.091
LETTER	0.055	0.060	0.058	0.084
OPTDIGITS	0.041	0.075	0.022	0.084
IONOSPHERE	0.624	0.423	0.497	0.555
WBC	0.464	0.362	0.319	0.486
MAMMOGRAPHY	0.117	0.108	0.076	0.118
GLASS	0.090	0.088	0.096	0.115
VERTEBRAL	0.149	0.174	0.186	0.154
CARDIO	0.237	0.219	0.180	0.505
HTTP	0.005	0.059	0.002	0.035
INSECTS GRADUAL_IMBALANCED	0.230	0.162	0.219	0.161
INSECTS INCREMENTAL_IMBALANCED	0.293	0.188	0.268	0.189
OZONE	0.086	0.067	0.112	0.059
TELEMETRY	1.000	0.022	1.000	1.000

B. Dataset descriptions

We report the number of data points, number of dimensions and the anomaly percentage for each dataset in Table 10. For the two INSECTS datasets, we use the most populous class 'Aedes aegypti (female)' as the benign class and the least populous class 'Culex quinquefasciatus (male)' as the anomalous class.

C. Hyperparameter configurations for base models

We report hyperparameter configurations for all base models in Table 11.

D. Ablations

We compare against the choices of $\eta = [1, 0.1, 0.01]$ and $\lambda = [10^{-2}, 10^{-4}, 10^{-6}]$ for SEAD in Table 13. We report ablations for SEAD ++ in Table 12, varying the number of sampled detectors K.

Table 9. Average Precision scores for SEAD++ ($K = 6$)

DATASET	SEAD++
PIMA	0.537
PENDIGITS	0.092
LETTER	0.078
OPTDIGITS	0.076
IONOSPHERE	0.535
WBC	0.536
MAMMOGRAPHY	0.091
GLASS	0.108
VERTEBRAL	0.155
CARDIO	0.131
HTTP	0.004
INSECTS GRADUAL_IMBALANCED	0.168
INSECTS INCREMENTAL_IMBALANCED	0.183
OZONE	0.076
TELEMETRY	1.000

Table 10. Dataset descriptions for the ODDS (Rayana, 2016), USP (Souza et al., 2020) and internal datasets

DATASET	#POINTS	#DIM.	#OUTLIERS (%)
PIMA (ODDS)	768	8	268 (35%)
PENDIGITS (ODDS)	6870	16	156 (2.27%)
LETTER (ODDS)	1600	32	100 (6.25%)
OPTDIGITS (ODDS)	5216	64	150 (3%)
IONOSPHERE (ODDS)	351	33	126 (36%)
WBC (ODDS)	278	30	21 (5.6%)
MAMMOGRAPHY (ODDS)	11183	6	260 (2.32%)
GLASS (ODDS)	214	9	9 (4.2%)
VERTEBRAL (ODDS)	240	6	30 (12.5%)
CARDIO (ODDS)	1831	21	176 (9.6%)
HTTP (ODDS)	567479	3	2211 (0.4%)
INSECTS GRADUAL_IMBALANCED (USP)	47251	33	4867 (10.3%)
INSECTS INCREMENTAL_IMBALANCED (USP)	148048	33	13331 (9%)
OZONE (USP)	2534	71	160 (12.5%)
TELEMETRY (PRIVATE)	3726	1	1 (0.03%)

Table 11. Hyperparameter configurations for the base models. Since we primarily focus on the streaming setting in this work, we only vary hyperparameters related to number of past data points stored in memory.

MODEL	HYPERPARAMETERS
RRCF	NUM_TREES = 4, SHINGLE_SIZE=4, TREE_SIZE=[32, 64, 128, 256]
IFORESTASD	INITIAL_WINDOW_X = NONE, WINDOW_SIZE=[256, 512, 1024, 2048]
XSTREAM	NUM_COMPONENTS = 100, N_CHAINS=100, DEPTH=25, WINDOW_SIZE=[8, 16, 25, 32]

Table 12. Ablations for SEAD ++ with different values for the number of sampled detectors K. The total number of base detectors is $N = 13$.

	RUNTIME (S)			AP SCORE		
DATASET	K=3	K=6	K=9	K=3	K=6	K=9
PIMA	143.347	318.449	589.054	0.455	0.537	0.526
PENDIGITS	1456.997	1924.700	2772.344	0.189	0.092	0.117
LETTER	440.250	570.286	916.786	0.073	0.078	0.133
OPTDIGITS	1552.878	1554.646	2299.575	0.043	0.076	0.142
IONOSPHERE	72.610	164.815	228.618	0.588	0.535	0.535
WBC	82.772	170.914	246.968	0.480	0.536	0.465
MAMMOGRAPHY	1955.149	2877.584	4006.977	0.086	0.091	0.098
GLASS	51.274	91.061	141.936	0.100	0.108	0.110
VERTEBRAL	65.503	98.824	151.341	0.195	0.155	0.182
CARDIO	351.500	648.944	941.799	0.252	0.131	0.190

Table 13. Performance across different hyperparameter combinations (η, λ)

	$\eta = 1$			$\eta = 0.1$			$\eta = 0.01$		
Dataset	$\lambda = 10^{-6}$	10^{-4}	10^{-2}	10^{-6}	10^{-4}	10^{-2}	10^{-6}	10^{-4}	10^{-2}
pima	0.519	0.519	0.520	0.531	0.531	0.531	0.527	0.527	0.527
pendigits	0.091	0.255	0.188	0.164	0.164	0.159	0.147	0.147	0.147
letter	0.084	0.083	0.076	0.066	0.066	0.064	0.056	0.056	0.056
optdigits	0.084	0.031	0.030	0.041	0.041	0.039	0.042	0.042	0.042
ionosphere	0.555	0.544	0.557	0.568	0.568	0.568	0.570	0.570	0.570
wbc	0.486	0.483	0.471	0.487	0.487	0.487	0.488	0.488	0.488
mammography	0.118	0.128	0.125	0.118	0.118	0.120	0.123	0.123	0.123
glass	0.115	0.129	0.124	0.093	0.093	0.093	0.094	0.094	0.094
vertebral	0.154	0.158	0.158	0.152	0.152	0.152	0.154	0.154	0.154
cardio	0.505	0.508	0.173	0.240	0.240	0.231	0.209	0.209	0.209

E. Example of evolution of weights

We show two examples where the best performing methods are different, and SEAD is able to select the best performing methods in each case. Figures 4 and 5 show the evolution of weights for two datasets. In Figure 4, on a variant of the INSECTS (Souza et al., 2020) dataset, SEAD gives highest weights to xStream models which obtain the best average precision. On the Cardio dataset (Aggarwal & Sathe, 2015; Sathe & Aggarwal, 2016a) in Figure 5, SEAD has the highest weights for the IForestASD models which obtain the best detection performance.

F. APS metrics after adding random detectors

We report Average Precision scores after adding random detectors for SEAD and ensemble baselines in Table 14. Adding random detectors demonstrates SEAD's robustness to uninformative base detectors - SEAD's performance drop is lower than the baselines as shown in Table 6.

G. Adapting (Goswami et al., 2023)'s method to the online setting for comparison in Table 5

The main issue with (Goswami et al., 2023)'s algorithm is that it is designed in the offline case where all the train data is available up-front, a single model is selected and applied as a batch to the entire inference dataset. In order to adapt it to the online setting, we ran inferences in batches of 50 data points, where we retrained and applied it to a contiguous batch of 50 data points. For each re-training, we use the entire data-stream seen thus far and re-train from scratch each time.

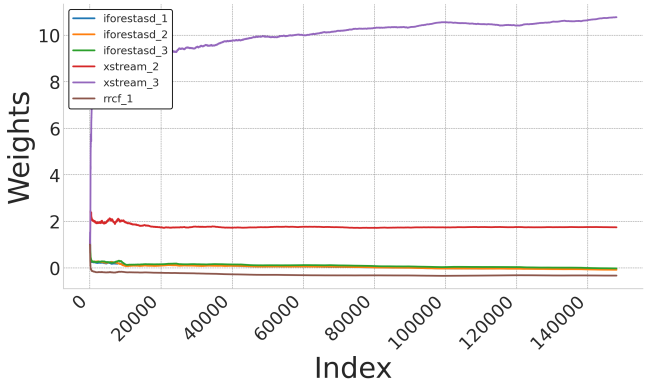

Figure 4. Evolution of weights across time for a variant of the INSECTS dataset, which is known to exhibit strong non-stationarity (Souza et al., 2020). SEAD 's performance is the best as indicated in Table 2. The evolution of the weights above shows SEAD having the highest weight to the second best performing base model.

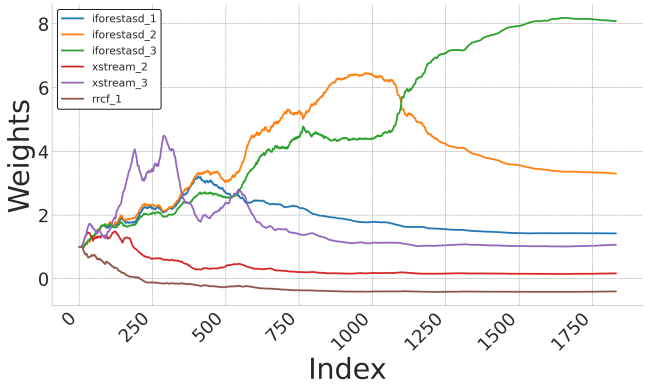

Figure 5. Evolution of weights across time for the Cardio dataset (Aggarwal & Sathe, 2015; Sathe & Aggarwal, 2016a). SEAD selects the top two best performing models.

Table 14. Comparison with ensemble baselines after adding 13 random detectors; the last row shows average performance drop across datasets after adding random detectors

DATASET	MEAN	MAX	MIN	SEAD (OURS)
PIMA	0.513	0.399	0.358	0.520
PENDIGITS	0.099	0.051	0.034	0.246
LETTER	0.056	0.055	0.073	0.088
OPTDIGITS	0.045	0.052	0.024	0.048
IONOSPHERE	0.493	0.323	0.385	0.491
WBC	0.436	0.176	0.121	0.292
MAMMOGRAPHY	0.081	0.053	0.029	0.060
GLASS	0.082	0.127	0.070	0.078
VERTEBRAL	0.154	0.193	0.192	0.137
CARDIO	0.195	0.147	0.098	0.518
AVG PERFORMANCE DROP	9.75%	22.31%	22.04%	0.88%

Operating this way significantly increased run-time due to repeated training on the stream. Repeated re-training is needed since we are operating on data-streams with non-stationarities. Adapting the algorithm presented in (Goswami et al., 2023) to be online where some computations can be re-used (for example the re-trainings need not be from scratch) is non-trivial and requires further research that is beyond the scope of this paper. The $35x$ increase in run-time was the fundamental reason we did not evaluate the algorithm from (Goswami et al., 2023) (and even MetaOD (Zhao et al., 2021)) on all the datasets.

H. Statistical significance tests to analyze performance difference between SEAD and base models

We perform Wilcoxon signed-rank test to evaluate the statistical significance of our results in Table 15. The datasets that we test on are diverse and no single method performs well on all datasets. Hence, each method has a high variance in Average Precision Score (APS) across the 15 datasets that we have tested on. The high variance makes statistical significance tests unreliable

To overcome this issue, we split each dataset into chunks of 50 contiguous data points. This is also relevant for the online learning paradigm where we want to evaluate the model continously over time. Since APS is not defined for chunks having all data points labeled as non anomalous, we only consider chunks which have at least one anomalous label. Using this splitting mechanism gives us 3,282 chunks across all datasets. We report p values for the Wilcoxon signed-rank test using the APS scores on the 3,282 chunks. Using a threshold of 0.01, we conclude that our method is statistically different from all base models and ensemble baselines.

Table 15. Statistical significance of SEAD performance compared to base detectors and baseline methods

MODEL	WILCOXON P-VALUE
SPC	6.93×10^{-258}
RRCF_0	4.26×10^{-68}
RRCF_1	1.47×10^{-43}
RRCF_2	3.16×10^{-14}
RRCF_3	7.43×10^{-29}
XSTREAM_0	2.09×10^{-95}
XSTREAM_1	3.53×10^{-12}
XSTREAM_2	1.80×10^{-4}
XSTREAM_3	1.60×10^{-3}
IFORESTASD_0	1.88×10^{-64}
IFORESTASD_1	8.08×10^{-63}
IFORESTASD_2	3.48×10^{-57}
IFORESTASD_3	2.94×10^{-51}
MEAN	6.41×10^{-171}
MAX	6.48×10^{-5}
MIN	6.17×10^{-44}