ECOD: Unsupervised Outlier Detection Using Empirical Cumulative Distribution Functions

Zheng Li*, Yue Zhao*, *Student Member* Xiyang Hu, Nicola Botta, Cezar Ionescu, and George H. Chen

Abstract—Outlier detection refers to the identification of data points that deviate from a general data distribution. Existing unsupervised approaches often suffer from high computational cost, complex hyperparameter tuning, and limited interpretability, especially when working with large, high-dimensional datasets. To address these issues, we present a simple yet effective algorithm called ECOD (Empirical-Cumulative-distribution-based Outlier Detection), which is inspired by the fact that outliers are often the "rare events" that appear in the tails of a distribution. In a nutshell, ECOD first estimates the underlying distribution of the input data in a nonparametric fashion by computing the empirical cumulative distribution per dimension of the data. ECOD then uses these empirical distributions to estimate tail probabilities per dimension for each data point. Finally, ECOD computes an outlier score of each data point by aggregating estimated tail probabilities across dimensions. Our contributions are as follows: (1) we propose a novel outlier detection method called ECOD, which is both parameter-free and easy to interpret; (2) we perform extensive experiments on 30 benchmark datasets, where we find that ECOD outperforms 11 state-of-the-art baselines in terms of accuracy, efficiency, and scalability; and (3) we release an easy-to-use and scalable (with distributed support) Python implementation for accessibility and reproducibility.

Index Terms—outlier detection, anomaly detection, distributed learning, scalability, empirical cumulative distribution function.

1 Introduction

Outliers, also sometimes referred to as anomalies, are data points that have different data characteristics from "normal" observations. Since the presence of outliers can markedly impact the results of statistical analyses, removing outliers is often a crucial preprocessing step in data analysis models ([1], [2]). However, to removing outliers, we need to identify them first, which is the goal of outlier detection (OD). OD has many applications, such as fraud detection ([3], [4], [5]), network intrusion detection ([6], [7]), social media analysis ([8], [9]), video analysis [10], intelligent transportation [11], [12], [13], and data generation [14]. These applications often demand OD algorithms with high detection accuracy and fast execution while being easy to interpret.

Numerous unsupervised OD algorithms have been proposed over the years (e.g., [15], [16], [17], [18], [19], [20]; we provide a more detailed overview in Section 2). These existing approaches have a few limitations. First of all, many of these methods, especially the ones requiring density estimation and pairwise distance calculation, suffer from the curse of dimensionality—both detection accuracy and runtime efficiency worsen rapidly as the number of data points and their dimensionality increase ([21], [22]). Second, most methods require hyperparameter tuning, which is difficult in the unsupervised setting ([23], [24]).

To address these limitations, we propose a simple yet effective method using **e**mpirical **c**umulative distribution functions for **o**utlier **d**etection, abbreviated as ECOD. ECOD is motivated by the definition of outliers, which may be viewed as the *rare events* in the data ([25], [26]). Rare events are often the ones that appear in one of the tails of a distribution. As a concrete example, if the data are sampled from a one-dimensional Gaussian, then points in the left or right tails are often considered to be the rare events or outliers. This has motivated commonly used heuristic OD approaches such as the "three-sigma" rule that declares points more than three standard deviations from the mean to be outliers ([27], [28]; there's also a robust variant called the "1.5 IQR" rule [29]). However, the three-sigma rule only uses the mean and standard deviation of a distribution. Instead, one could capture more information of the distribution by building a histogram of the data and use bins with low counts to determine where outliers are [30]. However, such an approach requires tuning over different ways to bin the data for histogram construction. Our approach ECOD avoids this problem of tuning altogether by estimating the empirical cumulative distribution function (ECDF) of the data, which has no parameters to tune and approximates the entire distribution without making any parametric assumptions.

The technical difficulty in using ECDFs arises when working with high-dimensional data: the joint ECDF over all variables converges more slowly to the true joint CDF as the number of dimensions increases [31]. Our approach ECOD sidesteps this issue in a straightforward manner: we compute a univariate ECDF for each dimension separately. Then to measure the outlyingness of a data point, we compute its tail probability across all dimensions via an

Y. Zhao is the corresponding author.

- *Z. Li is with Northeastern University-Toronto and Arima Inc., Toronto, Canada, M5G. E-mail: winston@arimadata.com*
- *Y. Zhao, X. Hu, and G. H. Chen are with the Heinz College of Information Systems and Public Policy, Carnegie Mellon University, Pittsburgh, PA, 15213. E-mail: {zhaoy,xiyanghu,georgechen}@cmu.edu*
- *N. Botta is with Potsdam Institute for Climate Impact Research, Potsdam, Germany, 14473. E-mail: botta@pik-potsdam.de*
- *C. Ionescu is with Deggendorf Institute of Technology, Deggendorf, Germany, 94469. E-mail: cezar.ionescu@th-deg.de*
- **Z. Li and Y. Zhao contributed equally with alphabetical ordering*

Received April 30, 2021; revised December 31, 2021; accepted March 4, 2022

independence assumption, which amounts to multiplying all the estimated tail probabilities from the different univariate ECDFs. We do this calculation in log space and in a manner that accounts for both left and right tails of each of the dimensions. Despite this independence assumption appearing to be quite strong, ECOD turns out to work very well in practice.

In summary, we propose a novel OD approach ECOD that has the following key advantages:

- **Effectiveness**: Through extensive evaluation, we show that ECOD outperforms 11 popular baseline OD methods on 30 benchmarks. Specifically, ECOD ranks the highest, and scores 2% higher in the area under the receiver operating characteristic curve and 5% higher in average precision than the second best detector.
- **Efficiency and scalability**: ECOD has time complexity $\mathcal{O}(nd)$, where n is the number of data points and d is the number of dimensions, and can trivially be parallelized across dimensions. Moreover, since ECOD has no hyperparameters, there is no time spent on hyperparameter tuning. With a single thread, ECOD can handle datasets with 1,000,000 observations and 10,000 features on a standard personal laptop in 2 hours.
- **Interpretability**: ECOD is easy to interpret. For any data point, we can look at its left or right estimated tail probability per dimension. This tells us how each dimension contributes to the overall outlier score we use. This information provides guidance to practitioners regarding which dimensions to focus on for improving data quality.

The rest of this paper is organised as follows: we review some existing OD techniques along with their strengths and limitations in Section 2. We describe the algorithmic design of ECOD and its properties in Section 3. We carry out experiments to compare ECOD with state-of-the-art OD methods and demonstrate that it is one of the most accurate, efficient, and scalable methods in Section 4. Finally, we conclude the paper in Section 5 with future directions, including a discussion on how to remove the independence assumption in how we aggregate tail probability information across dimensions. To facilitate reproducibility and accessibility, we open-source ECOD with distributed learning support as part of the popular PyOD library[1].

2 Related Work

Over the years, different types of unsupervised outlier detection algorithms have been proposed ([32], [33]). In this section, we give an overview of key methods for tabular data. Among them, eleven state-of-the-art algorithms are selected as baselines for our experiments in Section 4.

2.1 Proximity-based Algorithms

First, proximity-based algorithms, as their name suggests, are based on local neighborhood information around each point. These algorithms can be categorized into density- and distance-based algorithms [32].

[1]Implementation in PyOD library: https://github.com/yzhao062/pyod/blob/master/pyod/models/ecod.py

The core principle behind density-based outlier detection methods is that an outlier can be found in a low-density region, whereas non-outliers (also called *inliers*) are assumed to appear in dense neighborhoods. For example, the local outlier factor, proposed by Breunig *et al.* [34], is based on the concept of local density, where locality is given by the k nearest neighbors. Distances of these nearest neighbors are then used to help estimate local densities. We can declare a data point with substantially lower local density than its neighbors to be an outlier. A few improvements were later introduced, including the Connective-based Outlier Factor (COF) by Tang *et al.* [35], and LOcal Correlation Integral (LOCI) by Papadimitriou *et al.* [36].

Distance-based methods compare objects to their neighbors, and those that are considerably far from their neighbors are deemed outliers. This procedure does not estimate local densities. In terms of determining which points are considered neighbors, commonly the k nearest neighbors are used [15].

Proximity-based methods are mostly nonparametric as they do not assume any parametric distribution for the data. This can be a key advantage in outlier detection as minimal prior knowledge is required on the probability distribution of the dataset. This makes them intuitive and easy to understand. However, this family of methods are typically computationally expensive, sensitive to hyperparameters such as how to define the neighbors (e.g., which distance/similarity function to use, how many neighbors to consider), and vulnerable to the curse of dimensionality. Our proposed algorithm ECOD address both drawbacks—it has no hyperparameters to tune, and has time complexity that scales linearly in the dataset size and the number of dimensions.

2.2 Statistical Models

In the context of OD, approaches based on statistical models first fit probability distributions to data points. They then determine whether points are outliers based on the fitted models. These approaches are usually categorized into two main groups—parametric and nonparametric methods. The key difference is that parametric methods assume that the data come from a parametric distribution, so that fitting such a distribution amounts to learning the parameters of the assumed parametric distribution. Common parametric methods for OD include using Gaussian mixture models (GMM) [37] and linear regression [38]. In contrast, nonparametric methods do not assume a parametric model for the data. Some examples include Kernel Density Estimation (KDE) [39], histogram-based methods (HBOS) [30], and a few other variants. Typically, parametric models for OD are fast to use after the model fitting step, whereas nonparametric models for OD can be more expensive to work (for example, depending on the kernel used for KDE, the fitted model could—for an exact solution—require comparing a candidate test data point with all training data in deciding whether the test point is an outlier or not).

Note that our proposed algorithm ECOD uses a statistical model that is nonparametric although this model cannot represent all possible multivariate distributions. In particular, we model each dimension of the data in a fully nonparametric fashion (with a univariate ECDF) but to aggregate information across dimensions we assume that the different dimensions are independent. Thus, even though our approach has no parameters, it does make a strong structural assumption on the underlying joint probability distribution over the different dimensions/features.

2.3 Learning-based Models

Learning-based approaches involve training machine learning models to predict which points are outliers. Based on a training set, a classification model is trained to classify outliers and inliers. The same model is used to predict which points in the test set are most likely outliers. A classical method in this category is the one-class SVM (OCSVM) [16]. Another popular method leverages clustering to model the behavior of data points [40], and identifies outliers based on cluster assignments.

Using neural networks for detecting outliers has gained more attention in the last decade. Some representative examples from this line of work include the use of generative adversarial networks (GANs) [20], autoencoders [41] (including variational autoencoders), and reinforcement learning [42]. Recent developments in this direction can be found in the survey by Pang *et al.* [33].

In general, learning-based methods often work well in practice when working with large datasets. However, most learning-based methods are computationally expensive. Moreover, they tend to involve nontrivial hyperparameter tuning especially in the unsupervised setting. Lastly, state-of-the-art learning-based methods that use deep nets are often difficult to interpret.

2.4 Ensemble-based models

Ensemble-based approaches in OD combine results from various base outlier detectors to produce more robust OD results. The intuition is similar to how ensembling works in standard classification/regression tasks (e.g., bagging, boosting, random forests). Notable works of ensemble-based OD include feature bagging [25] that use various sub-feature spaces, isolation forests [17] that aggregate the information from multiple base trees, LSCP [18] that dynamically picks the best base estimator for each data point, and SUOD [21] that uses a large number of heterogeneous estimators.

In general, ensemble-based methods for OD often work well in practice even for high-dimensional datasets. However, these methods also can involve nontrivial tuning, such as in selecting the right meta-detectors [23]. Additionally, ensemble-based methods are often less interpretable.

Our proposed algorithm ECOD could be thought of as an ensemble model in that we are learning a nonparametric statistical model per dimension of the data, and then we aggregate/ensemble the models across dimensions to detect outliers. Because each "base" model uses only a single dimension of the data, and we aggregate the base models in a straightforward manner, our resulting learned model turns out to be easy to interpret.

3 Proposed Algorithm: ECOD

We now present the details of ECOD in three subsections. First, we provide a problem statement of unsupervised outlier detection and the associated challenges in Section 3.1. We then provide the motivation and technical details of ECOD in Section 3.2. We discuss properties of ECOD in Section 3.3, including interpretability and scalability.

3.1 Problem Formulation and Challenges

We consider the following standard unsupervised outlier detection (UOD) setup. We assume that we have n data points $X_1, X_2, \ldots, X_n \in \mathbb{R}^d$ that are sampled i.i.d. We collectively refer to this entire dataset by the matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, which is formed by stacking the different data points' vectors as rows. Given $\mathbf{X}$, an OD model M assigns, for each data point X_i, an outlier score $O_i \in \mathbb{R}$ (higher means more likely an outlier).

There are a few notable challenges for UOD:

- *Curse of dimensionality*: many OD algorithms described in Section 2 become less accurate or have computational scalability issues when the input dataset $\mathbf{X}$ is high-dimensional (d is either larger than or of similar order of magnitude as n), or the number of data points n is large. Specifically for proximity-based methods, density estimation and (pairwise) distance calculation become more computationally expensive [21]. Moreover, in general, density estimation in the high-dimensional setting can require the number of data points to scale exponentially in the number of dimensions [43].
- *Limited interpretability*: OD applications often require interpretability/explainability to a certain extent. For instance, in detecting whether a financial transaction is an outlier and considered "fraudulent", it could be helpful providing some sort of evidence as to why a transaction is considered fraudulent or not. This capacity is often absent in most existing OD algorithms we are aware of.
- *Complexity in hyperparameter tuning*: without access to any ground truth labels for which data points are outliers, model selection and hyperparameter tuning are challenging in existing OD algorithms [23].

3.2 The Proposed ECOD

3.2.1 Motivation and High-level Idea

A natural way to characterize outliers is to take them to correspond to rare events that occur in low density parts of probability distribution ([25], [26]). If the distribution is unimodal, then these rare events occur in the tails of the distribution. With this motivation, for each observation X_i, our method ECOD is based on computing the probability of observing a point at least as "extreme" as X_i in terms of tail probabilities.

Specifically, let $F : \mathbb{R}^d \to [0,1]$ denote the joint cumulative distribution function (CDF) across all d dimensions/features. In particular, $X_1, X_2, \ldots, X_n$ are assumed to be sampled i.i.d. from a distribution with joint CDF F. For a vector $z \in \mathbb{R}^d$, we denote its j-th entry as $z^{(j)}$, e.g., we write the j-th entry of X_i as $X_i^{(j)}$. We use the random variable X to denote a generic random variable with the same distribution as each X_i. Then by the definition of a joint CDF, for any $x \in \mathbb{R}^d$,

$$F(x) = \mathbb{P}(\underbrace{X^{(1)} \le x^{(1)}, X^{(2)} \le x^{(2)}, \ldots, X^{(d)} \le x^{(d)}}_{\text{abbreviated as } “X \le x”})$$

Algorithm 1 Unsupervised OD Using ECDF (ECOD)

```
Inputs: input data $\mathbf{X} = \{X_i\}_{i=1}^n \in \mathbb{R}^{n \times d}$ with $n$ samples
    and $d$ features; $X_i^{(j)}$ refers to the value of $j$-th feature of
    the $i$-th sample
Outputs: outlier scores $\mathbf{O} := \text{ECOD}(\mathbf{X}) \in \mathbb{R}^n$
1: for each dimension $j$ in $1, \dots, d$ do
2:     Estimate left and right tail ECDFs (using equations (1) and 2, which we reproduce below):
           left tail ECDF: $\widehat{F}_{\text{left}}^{(j)}(z) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{X_i^{(j)} \le z\}$ for $z \in \mathbb{R}$,
           right tail ECDF: $\widehat{F}_{\text{right}}^{(j)}(z) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{X_i^{(j)} \ge z\}$ for $z \in \mathbb{R}$.
3:     Compute the sample skewness coefficient for the $j$-th feature's distribution:
           $\gamma_j = \frac{\frac{1}{n}\sum_{i=1}^{n}(X_i^{(j)} - \overline{X^{(j)}})^3}{\left[\frac{1}{n-1}\sum_{i=1}^{n}(X_i^{(j)} - \overline{X^{(j)}})^2\right]^{3/2}}$,
       where $\overline{X^{(j)}} = \frac{1}{n}\sum_{i=1}^{n} X_i^{(j)}$ is the sample mean of the $j$-th feature.
4: end for
5: for each sample $i$ in $1, \dots, n$ do
6:     Aggregate tail probabilities of $X_i$ to obtain outlier score $O_i$:    // §3.2.2
           $O_{\text{left-only}}(X_i) = -\sum_{j=1}^{d} \log(\widehat{F}_{\text{left}}^{(j)}(X_i^{(j)}))$,
           $O_{\text{right-only}}(X_i) = -\sum_{j=1}^{d} \log(\widehat{F}_{\text{right}}^{(j)}(X_i^{(j)}))$,
           $O_{\text{auto}}(X_i) = -\sum_{j=1}^{d} [\mathbb{1}\{\gamma_j < 0\} \log(\widehat{F}_{\text{left}}^{(j)}(X_i^{(j)}))$
               $+ \mathbb{1}\{\gamma_j \ge 0\} \log(\widehat{F}_{\text{right}}^{(j)}(X_i^{(j)}))]$.
7:     Set the final outlier score for point $X_i$ to be
           $O_i = \max\{O_{\text{left-only}}(X_i), O_{\text{right-only}}(X_i), O_{\text{auto}}(X_i)\}$.
8: end for
9: Return outlier scores $\mathbf{O} = (O_1, \dots, O_n)$
```

This probability is a measure of how "extreme" X_i is in terms of left tails: the smaller $F(X_i)$ is, then the less likely a point X sampled from the same distribution as X_i will satisfy the inequality $X \le X_i$ (again, this inequality needs to hold across all d dimensions). Similarly, $1 - F(X_i)$ is also a measure of how "extreme" X_i is, however, looking at the right tails of every dimension instead of the left tails. Therefore, if either $F_{\mathbf{X}}(X_i)$ or $1-F_{\mathbf{X}}(X_i)$ is extremely small, then this suggests that X_i corresponds to a rare event and is, therefore, likely to be an outlier.

The challenge is that in practice, we do not know the true joint CDF and have to estimate it from data. The rate of convergence in estimating a joint CDF using a joint ECDF slows down as the number of dimensions increases [31]. As a simplifying assumption, we assume that the different dimensions/features are independent so that joint CDF has the factorization

$$F(x) = \prod_{j=1}^{d} F^{(j)}(x^{(j)}) \qquad \text{for } x \in \mathbb{R}^d,$$

where $F^{(j)} : \mathbb{R} \to [0,1]$ denotes the univariate CDF of the j-th dimension: $F^{(j)}(z) = \mathbb{P}(X^{(j)} \le z)$ for $z \in \mathbb{R}$.

Now it suffices to note that univariate CDF's can be accurately estimated simply by using the empirical CDF (ECDF), namely:

$$\widehat{F}_{\text{left}}^{(j)}(z) := \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{X_i^{(j)} \le z\} \qquad \text{for } z \in \mathbb{R}, \tag{1}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function that is 1 when its argument is true and is 0 otherwise.

Previously, we mentioned that the right tail could be obtained by looking at 1 minus a CDF. However, we remark that there is a slight asymmetry in doing this since

$$1 - F^{(j)}(z) = 1 - \mathbb{P}(X^{(j)} \le z) = \mathbb{P}(\underbrace{X^{(j)} > z}_{\text{note the strict inequality}}).$$

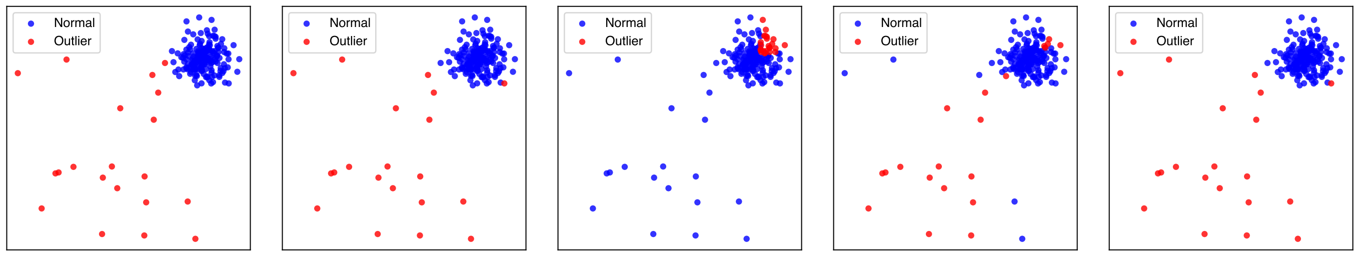
Basically $F^{(j)}(z)$ does not use a strict inequality whereas $1 - F^{(j)}(z)$ does. In how we aggregate tail probabilities, we will combine information from both left and right tails, and for symmetry, we actually separately also compute the "right-tail" ECDF:

$$\widehat{F}_{\text{right}}^{(j)}(z) := \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{X_i^{(j)} \ge z\} \qquad \text{for } z \in \mathbb{R}. \tag{2}$$

Thus, we can estimate the joint left and right-tail ECDFs across all d dimensions under an independence assumption via the estimates

$$\widehat{F}_{\text{left}}(x) = \prod_{j=1}^{d} \widehat{F}_{\text{left}}^{(j)}(x^{(j)}) \text{ and } \widehat{F}_{\text{right}}(x) = \prod_{j=1}^{d} \widehat{F}_{\text{right}}^{(j)}(x^{(j)}) \quad \text{for } x \in \mathbb{R}^d. \tag{3}$$

Our proposed method ECOD builds on the ideas we just presented and consists of two main steps: first, we compute each dimension's left- and right-tail ECDFs as given in equations (1) and (2). Next, for every point X_i, we aggregate its tail probabilities $\widehat{F}_{\text{left}}^{(j)}(X_i^{(j)})$ and $\widehat{F}_{\text{right}}^{(j)}(X_i^{(j)})$ to come up with a final outlier score $O_i \in [0, \infty)$; higher means more likely to be an outlier. Note that these outlier scores are not probabilities and are meant to be used to comparatively between the data points. We explain in more detail how we aggregate tail probabilities in Section 3.2.2. An important idea we use in aggregating tail probabilities is that it does not always make sense to only consider the left tail probability for every dimension and then separately only consider the right tail probability for every dimension. Meanwhile, especially when d is large, it is impractical to consider all 2^d possible combinations of whether to use the left or right tail probability per dimension. Our aggregation step uses the skewness of a dimension's distribution to automatically select whether we use the left or the right tail probability for a dimension. The pseudocode of ECOD is given in Algorithm 1.



(a) Ground truth (b) Left tail (c) Right tail (d) Avg of two tails (e) Skewness corrected

Fig. 1: An illustration of how using different tail probabilities affect the results. The leftmost column are plots of the ground truth, the second column are the result if left tail probabilities are used. The middle column corresponds to outlier detection if right tail probabilities are used, followed by using the average of both tail probabilities in the fourth column and skewness corrected tail probabilities (SC) in the rightmost column (which shows the best result).

3.2.2 *Aggregating Outlier Scores*

Using skewness to decide on whether to use the left or the right tail probability for a specific dimension. Using only the left tail probability for every dimension or only the right tail probability for every dimension can be limiting in finding outliers; it could be that in a specific dimension, being in the left tail could be considered more of an outlier whereas in another dimension, we should instead consider the right tail. To automatically decide on which tail we should consider for a particular dimension, we consider the skewness of that dimension's distribution.

Consider Fig. 1, where we demonstrate the effect of using different tail probabilities in catching outliers. The subfigure on the left most column is the ground truth of synthetic dataset, where outliers are labelled in red and the normal points are labeled in blue. The dataset is generated by taking 180 samples from a normal distribution centered at the top right corner of a unit square as inliers, combined with 20 samples from a uniform distribution on the square as outliers. In the next three columns, we show the results of using left tail probabilities, right tail probabilities, and averages of both tail probabilities to denote outlyingness, respectively. For each of the situations, we consider the same number of outliers (i.e., 20).

Because of the nature of this dataset (i.e., all outliers fall on the left end of the 2-dimensional distribution), using the left tail probabilities works surprising well (Fig. 1, column 2), because all outliers, by the construction of the dataset, are "smaller" than inliers. On the other hand, using the right tail worked extremely poorly (Fig. 1, column 3), because there are no extremely large outliers. Thus, it would have wrongfully captured the relatively large points as outliers. Taking an average of the two tail probabilities (Fig. 1, column 4) compromises the situation, and identifies half of the left tail outliers and misidentifies half of the right tail. We should point out that if the dataset is flipped, and all outliers appear on the top right corner while the inliers appear on the bottom left, then using the right tail probabilities for identifying outliers would have worked perfectly, while left tail probabilities would perform poorly. Averaging, on the other hand, works if both large and small outliers occur.

To overcome this, we note that the skewness of the dataset plays a major role in whether left tail probabilities or right tail probabilities should be used. In this example, both marginals of X_1 and X_2 skew negatively (i.e., the left tail is longer and the mass of the distribution is concentrated on the right). In this case, it makes sense to use the left tail probabilities. For dimensions that skew positively, using the right tail probabilities would become a better choice. The right most column of Fig. 1 demonstrates the outcome of ECOD of tail selection is based on skewness of each dimension, and we can see that it is able to capture the correct outliers. We called this the skewness corrected (SC) version of ECOD and in Section 4.2, we compare the performance of different variants of ECOD using different tail probabilities.

We therefore propose to decide which tail to use in ECOD based on the skewness of the underlying distribution, where the sample skewness coefficient of dimension j can be calculated as below [44]:

$$\gamma_j = \frac{\frac{1}{n}\sum_{i=1}^{n}(X_i^{(j)} - \overline{X^{(j)}})^3}{\left[\frac{1}{n-1}\sum_{i=1}^{n}(X_i^{(j)} - \overline{X^{(j)}})^2\right]^{3/2}},$$

where $\overline{X^{(j)}} = \frac{1}{n}\sum_{i=1}^{n} X_i^{(j)}$. When $\gamma_j < 0$, we can thus consider points in the left tail to be more outlying. When $\gamma > 0$, we instead consider points in the right tail to be more outlying.

Final aggregation step. To compute the final outlier score per data point, we simply work in negative log probability space. In particular, we compute equation (3) as:

$$O_{\text{left-only}}(X_i) := -\log \widehat{F}_{\text{left}}(X_i) = -\sum_{j=1}^{d}\log(\widehat{F}_{\text{left}}^{(j)}(X_i^{(j)})), \tag{4}$$

$$O_{\text{right-only}}(X_i) := -\log \widehat{F}_{\text{right}}(X_i) = -\sum_{j=1}^{d}\log(\widehat{F}_{\text{right}}^{(j)}(X_i^{(j)})). \tag{5}$$

Meanwhile, we also compute the "automatic" version that decides whether to use the left or the right tail of the j-th dimension based on whether $\gamma_j < 0$ or $\gamma_j > 0$. Note that if the data are sampled from a continuous random variable, then $\gamma = 0$ with probability 0, so for simplicity, we just break the tie in favor of one of the two directions. We specifically use the automatic outlier score:

$$O_{\text{auto}}(X_i) = -\sum_{j=1}^{d}[\mathbb{1}\{\gamma_j < 0\}\log(\widehat{F}_{\text{left}}^{(j)}(X_i^{(j)})) + \mathbb{1}\{\gamma_j \geq 0\}\log(\widehat{F}_{\text{right}}^{(j)}(X_i^{(j)})).$$

Since we are operating in negative log probability space, lower probability (which suggests a more rare occurrence) corresponds to higher negative log probability. We use whichever negative log probability score is highest as the final outlier score O_i for point X_i, i.e.,

$$O_i = \max\{O_{\text{left-only}}(X_i), O_{\text{right-only}}(X_i), O_{\text{auto}}(X_i)\}. \quad (6)$$

Essentially we are using the most extreme of the three scores computed.

3.3 Properties of ECOD

3.3.1 ECOD *as an Interpretable Outlier Detector*

Interpretability in machine learning is an important concept, as it provides domain experts some insights into how algorithms make their decisions [45], [46]. Interpretable algorithms provide both transparency and reliability. Having a transparent model means that humans can learn from the thought process of models, and try to discover the "whys" behind why a particular data point is classified. Clearly, Interpretability is also critical in outlier detection applications [47]. For instance, explaining why a transaction is fraudulent is equally important to identifying it.

As ECOD evaluates the outlying behaviors on a dimensional basis, we could use ECOD as an explainable detector for dimensional contribution. From equation 6, we know that if a sample has a large outlier score, then at least one of the three tail probabilities is large. Thus, let $O_i^{(j)}$ be the dimensional outlier score for dimension j of X_i, and using fact that the $\log$ function is monotonic, we can see that it represents the degree of outlyingness of dimension j. This can be compared against some preset thresholds, such as $-\log(0.01) = 4.61$, or top α percent of $O(X_i^{(j)})$ $\forall i$ to give practitioners some indications why particular points are considered outliers. By plotting its corresponding graph (i.e., *Dimensional Outlier Graph*), a more direct understanding of features' contribution can be acquired.

Illustration with Breast Cancer Wisconsin Dataset. We demonstrate the interpretability aspect of ECOD with the Breast Cancer Wisconsin (Diagnostic) Data Set (*Breastw*, Table 1) as an example. Cell samples are provided by Dr. William H. Wolberg from the University of Wisconsin as part of his reports on clinical cases [48]. The data used has 369 samples, with 9 features listed below, and two outcome classes (benign and malignant). Features are scored on a scale of 1-10, and are (1) Clump Thickness, (2) Uniformity of Cell Size, (3) Uniformity of Cell Shape, (4) Marginal Adhesion, (5) Single Epithelial Cell Size, (6) Bare Nuclei, (7) Bland Chromatin, (8) Normal Nucleoli and (9) Mitoses.

We are particularly interested in providing an explanation, in addition to giving the right classification of outliers (malignant samples), as this can provide useful guidance

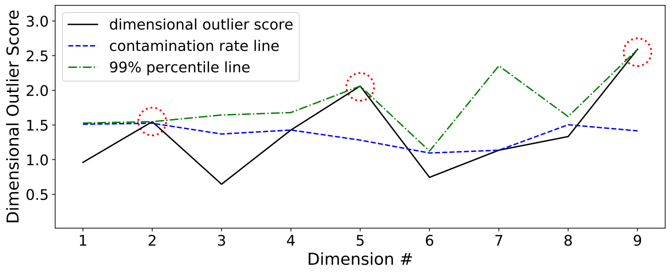


Fig. 2: *Dimensional Outlier Graph* for sample 70 (classified as an outlier) of BreastW dataset; dimension 2, 5, 9 are identified with most contribution to its outlyingness.

for physicians to further investigate why certain cells are potentially harmful. We illustrate how ECOD attempts to explain its outlier detection process. For each of dimension j (9 dimensions in total) of the i-th data point, we plot $O(X_i^{(j)})$, the dimensional outlier score in black. Clearly, different dimensions show different contribution. We also plot the 99^{th} percentile band in green as a reference line (which corresponds to $1-\alpha$ percentage of outliers). In the analysis below, we plot the Dimensional Outlier Graph for 70-th sample ($i = 70$), which is malignant (outlier) and has been successfully classified by ECOD.

We see that for dimension/feature 2, 5 and 9, the dimensional outlier scores have "touched" the 99^{th} percentile band, while all other features remain below the contamination rate band. This suggests that this point is likely an outlier because it is extremely non-uniform in size (dim. 2), has a large single epithelial cell size (dim. 5), and is more likely to reproduce via mitosis rather than meiosis (dim. 9).

3.3.2 *Complexity Analysis*

ECOD has $\mathcal{O}(nd)$ time and space complexity, and we break down the analysis by steps. In the estimation steps (Section 3.2.1), computing ECDF for all d dimensions using n samples leads to $\mathcal{O}(nd)$ time and space complexity. In the aggregation steps (Section 3.2.2), tail probability calculation and aggregation also lead to $\mathcal{O}(nd)$ time and space complexity.

3.3.3 *Acceleration by Distributed Learning*

Since ECOD estimates each dimension independently, it is suited for distributed learning acceleration with multiple workers. Given there are t available workers for distributed computing, e.g., t cores on a single machine. This constructs the worker pool as $\mathcal{W} = \{W_1, ..., W_t\}$. Without distributed learning, ECOD will estimate each dimension $j \in \{1, \ldots, d\}$ of $\mathbf{X}$ iteratively, e.g., with a for loop. With multiple workers available ($t > 1$), a generic scheduling system equally splits d ECDF estimation and tail probability computation tasks into t groups, so each available worker in $\mathcal{W}$ will process roughly $\lceil \frac{d}{t} \rceil$ estimation and computation task. Similar to the non-distributed setting, the results from each dimension will then be aggregated to generate the final outlier scores.

4 Empirical Evaluation

TABLE 1: 30 real-world benchmark datasets used in this study for evaluation. The datasets from the `ODDS` repo. are appended with "(mat)", and the datasets from the `DAMI` repo. are appended with "(arff)". For the *Shuttle* dataset in both databases, we randomly subsample 10,000 samples.

Dataset	# Samp. (n)	# Dims. (d)	% Outlier
Arrhythmia (mat)	452	274	14.601
Breastw (mat)	683	9	34.992
Cardio (mat)	1831	21	9.612
Ionosphere (mat)	351	33	35.897
Lympho (mat)	148	18	4.054
Mammography (mat)	11183	6	2.325
Optdigits (mat)	5216	64	2.875
Pima (mat)	768	8	34.895
Satellite (mat)	6435	36	31.639
Satimage-2 (mat)	5803	36	1.223
Shuttle (mat) (*)	10000	9	7.120
Speech (mat)	3686	400	1.654
WBC (mat)	378	30	5.555
Wine (mat)	129	13	7.751
Arrhythmia (arff)	450	259	45.777
Cardiotocography (arff)	2114	21	22.043
HeartDisease (arff)	270	13	44.444
Hepatitis (arff)	80	19	16.250
InternetAds (arff)	1966	1555	18.718
Ionosphere (arff)	768	8	34.895
KDDCup99 (arff)	4207	57	39.909
Lymphography (arff)	340	9	9.116
Pima (arff)	351	32	35.897
Shuttle (arff) (*)	10000	41	0.156
SpamBase (arff)	148	18	4.054
Stamps (arff)	1013	9	1.283
Waveform (arff)	3443	21	2.904
WBC (arff)	223	9	4.484
WDBC (arff)	367	30	2.724
WPBC (arff)	198	33	23.737

Our experiments answer the following questions:

1) Among the variants of ECOD using different tail probabilities, which one yields the best performance? (§4.2)
2) How effective is ECOD, in comparison to state-of-the-art (SOTA) outlier detectors? (§4.3.1)
3) Under which conditions, the performance of ECOD may degrade? (§4.3.2)
4) How efficient and scalable is ECOD regarding high-dimensional, large datasets? (§4.4)

4.1 Experiment Setup

Datasets. Table 1 summarises 30 public outlier detection benchmark datasets used in this study from `ODDS`[1] [49] and `DAMI`[2] [50] data repositories. Two special notes should be made about the datasets. First, multiple versions of the same dataset exist in different literature. For example, two versions of the *Pima* dataset exist: the *.mat* version has 8 dimensions and 768 observations, while the *.arff* version has 32 dimensions and 351 observations. This is because different subsets of the original data were used by different authors, and they keep different dimensions and/or observations. For clarity, we specify the source of a dataset by appending "(mat)" (from `ODDS`) or "(arff)" (from `DAMI`) to its name. Second, because some baselines fail to converge for a large number of observations, we truncate both versions of *Shuttle* down to 10,000 observations. The truncation is done by taking 10,000 random rows from the original datasets (which have slightly over 60,000 observations each).

Evaluation metrics. In each experiment, 60% of the data is used for training and the remaining 40% is set aside for testing. Performance is evaluated by taking the average score of 10 independent trials using area under the receiver operating characteristic (ROC) and average precision (AP). Both metrics are widely used in outlier mining research [51], [52], [53], [54]. We report both the raw comparison results and the critical difference (CD) plots to show statistical difference [55], [56], where it visualize the statistical comparison by Wilcoxon signed-rank test with Holm's correction.

Baselines, Implementation, and Environment. We compare the performance of ECOD with 11 leading outlier detectors. We aim to include a variety of detectors to make the comparison robust. Specifically, the 11 competitors are Angle-Based Outlier Detection (ABOD) [57], Clustering-Based Local Outlier Factor (CBLOF) [40], Histogram-based Outlier Score (HBOS) [30], Isolation Forest (IForest) [17], k Nearest Neighbors (KNN) [15], Lightweight On-line Detector of Anomalies (LODA) [58], Local Outlier Factor(LOF) [34], Locally Selective Combination in Parallel Outlier Ensembles (LSCP) [18], One-Class Support Vector Machines (OCSVM) [16], PCA-based outlier detector (PCA) [59], and Scalable Unsupervised Outlier Detection (SUOD) [21]. Their technical strength and limitations are discussed in Section 2.

Notably, PyOD is a popular open-source Python toolbox for performing scalable outlier detection on multivariate data [60]. A wide range of outlier detection algorithms are included under a single, well-documented API. This allows us to easily compare ECOD with other baseline detectors. Our implementation of ECOD is also under the framework of PyOD , and both single-process and distributed versions are readily available in PyOD[3]. For fair comparison, we use the single-process version in this study. In the subsequent experiments, a Windows laptop with Intel i5-8265U @ 1.60 GHz quad core CPU and 8GB of memory is used.

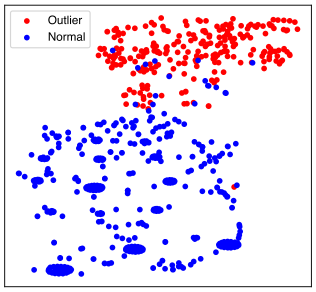
4.2 The Effect of Using Different Tails with ECOD

How does the usage of different tail probabilities affect the performance of ECOD? In the first experiment, we compare the performances of ECOD while using different tail probabilities for measuring sample outlyingness. Specifically, the three variants and ECOD are compared: (1) only using left tail probability (ECOD-L) (2) only using right tail probability (ECOD-R) (3) using the average of both tail probabilities (ECOD-B) and (4) using automatically selected tail probability (ECOD) as outlined in Section 3.2.2. Note that with the three variants, the outlier score of the i-th sample is equal to $O_{\text{left-only}}(X_i)$, $O_{\text{right}}(X_i)$, and $\frac{1}{2}(O_{\text{left-only}}(X_i) + O_{\text{right-only}}(X_i))$ respectively, where $O_{\text{left-only}}(X_i)$ and $O_{\text{right-only}}(X_i)$ are the sum of negative log probabilities defined in Eq. (5). Differently, ECOD uses both tails automatically as described in

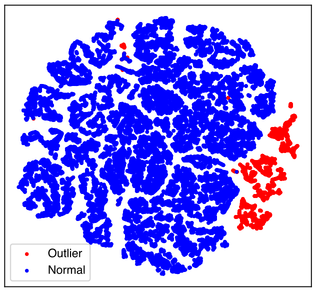
[1]ODDS Library: http://odds.cs.stonybrook.edu

[2]DAMI Datasets: http://www.dbs.ifi.lmu.de/research/outlier-evaluation/DAMI

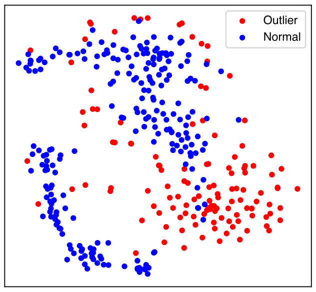
[3]ECOD implementation in PyOD library: https://github.com/yzhao062/pyod/blob/master/pyod/models/ecod.py



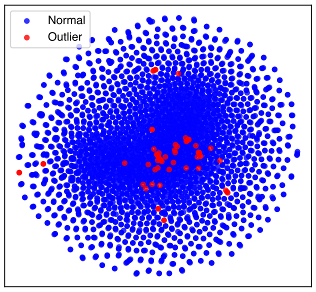
(a) *Breastw (mat)*



(b) *Shuttle (mat)*



(c) *Ionosphere (mat)*



(d) *Speech (mat)*

Fig. 3: 2-D embedding of selected datasets. In the first two datasets (*Breastw* and *Shuttle*), the outliers locate at the tail for at least one dimension, and ECOD is the best algorithm in comparison to the baselines. In the last two datasets (*Ionosphere* and *Speech*), where outliers do not locate at the tails in any dimension but mingle with inliers, ECOD's performance degrades.

TABLE 2: ROC score of 3 ECOD variants and ECOD (average of 10 independent trials, highest score highlighted in bold). Clearly, ECOD (the rightmost column) outperforms (ranking 1st in 18 out of 30 dataset).

Data	ECOD-L	ECOD-R	ECOD-B	ECOD
Arrhythmia (mat)	0.716	0.659	0.760	**0.802**
Breastw (mat)	0.665	0.641	0.774	**0.993**
Cardio (mat)	0.286	0.725	0.589	**0.897**
Ionosphere (mat)	0.765	0.523	0.810	**0.830**
Lympho (mat)	0.406	0.677	0.677	**0.993**
Mammography (mat)	0.718	0.384	0.706	**0.894**
Optdigits (mat)	0.993	0.997	**0.997**	0.732
Pima (mat)	0.550	0.990	**1.000**	0.663
Satellite (mat)	0.279	**0.736**	0.604	0.661
Satimage-2 (mat)	0.490	0.819	0.792	**0.985**
Shuttle (mat)	0.151	0.707	0.664	**0.990**
Speech (mat)	0.688	0.746	**0.873**	0.484
WBC (mat)	0.578	0.532	0.588	**0.974**
Wine (mat)	0.025	**0.990**	0.988	0.949
Arrhythmia (arff)	0.012	**0.984**	0.924	0.761
Cardiotocography (arff)	0.432	0.576	0.493	**0.637**
HeartDisease (arff)	0.805	0.687	0.821	**0.672**
Hepatitis (arff)	0.008	**0.993**	0.991	0.843
InternetAds (arff)	0.887	0.687	**0.936**	0.676
Ionosphere (arff)	0.970	0.316	**0.933**	0.821
KDDCup99 (arff)	0.730	0.383	0.715	**0.997**
Lymphography (arff)	0.884	0.732	0.995	**0.998**
Pima (arff)	0.241	0.892	**0.898**	0.646
Shuttle (arff)	0.460	0.666	0.693	**0.879**
SpamBase (arff)	0.225	**0.755**	0.584	0.700
Stamps (arff)	0.749	0.414	0.581	**0.938**
Waveform (arff)	0.926	0.813	**0.973**	0.786
WBC (arff)	0.461	0.458	0.450	**0.990**
WDBC (arff)	0.031	0.975	0.884	**0.982**
WPBC (arff)	0.040	**0.965**	0.748	0.546
AVG	0.560	0.714	0.781	**0.824**

TABLE 3: Average precision (AP) of 3 ECOD variants and ECOD (average of 10 independent trials, highest score highlighted in bold). Clearly, ECOD (the rightmost column) outperforms (ranking 1st in 18 out of 30 dataset).

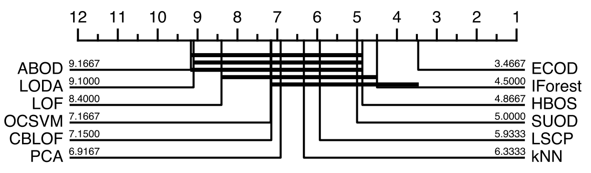
Data	ECOD-L	ECOD-R	ECOD-B	ECOD
Arrhythmia (mat)	**0.503**	0.318	0.497	0.472
Breastw (mat)	0.209	0.987	0.984	**0.987**
Cardio (mat)	0.450	0.163	**0.587**	0.579
Ionosphere (mat)	0.363	0.006	0.151	**0.719**
Lympho (mat)	0.592	0.334	0.622	**0.893**
Mammography (mat)	0.528	0.431	**0.912**	0.429
Optdigits (mat)	0.013	0.392	**0.426**	0.053
Pima (mat)	0.026	0.049	0.045	**0.540**
Satellite (mat)	0.229	**0.611**	0.460	0.585
Satimage-2 (mat)	0.550	0.259	0.523	**0.859**
Shuttle (mat)	0.113	0.044	0.670	**0.980**
Speech (mat)	0.015	0.022	**0.022**	0.019
WBC (mat)	0.035	0.772	0.464	**0.782**
Wine (mat)	0.044	**0.666**	0.218	0.608
Arrhythmia (arff)	0.710	0.636	0.751	**0.751**
Cardiotocography (arff)	0.370	0.294	**0.493**	0.377
HeartDisease (arff)	0.319	**0.663**	0.508	0.640
Hepatitis (arff)	0.444	0.221	0.434	**0.584**
InternetAds (arff)	0.156	0.511	**0.512**	0.510
Ionosphere (arff)	0.572	0.334	**0.607**	0.530
KDDCup99 (arff)	0.166	0.219	0.278	**0.566**
Lymphography (arff)	0.120	0.986	**1.000**	0.463
Pima (arff)	0.244	0.590	0.477	**0.703**
Shuttle (arff)	0.014	0.068	0.073	**0.238**
SpamBase (arff)	0.254	0.568	0.527	**0.981**
Stamps (arff)	0.199	0.220	**0.352**	0.085
Waveform (arff)	0.049	0.031	0.042	**0.078**
WBC (arff)	0.028	0.838	0.820	**0.838**
WDBC (arff)	0.015	**0.842**	0.622	0.840
WPBC (arff)	0.199	**0.266**	0.211	0.243
AVG	0.251	0.411	0.476	**0.564**

Eq. (6) Table 2 and 3 show the results comparison regarding both ROC and AP on the benchmark datasets.

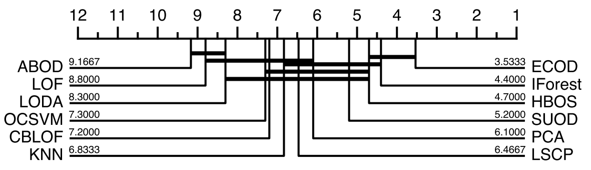
The proposed ECOD **achieves the best performance**, with an average ROC of 0.824 and average precision of 0.564. It is better than ECOD-B using the average of two tails (avg. ROC: 0.781, avg. AP: 0.476), ECOD-R using right tails (avg. ROC: 0.714, avg. AP: 0.411), and ECOD-L using left tails (avg. ROC: 0.506, avg. AP: 0.251). Its superiority can be credited to the automatic selection of tail probabilities by the skewness of the underlying datasets. Clearly, the 30 benchmark datasets would have outliers on the different tails of the distribution, and none of the variants can capture this, although taking the average in ECOD-B alleviates the problem and therefore ranks at the second position. Consequently, we use the automatic version in ECOD by default, which is more carefully analyzed through the rest of the paper.

TABLE 4: ROC scores of detector performance (average of 10 independent trials, highest score highlighted in bold); rank is shown in parenthesis (lower is better). ECOD outperforms all baselines with the highest avg. score and ranks the highest in 13 out of 30 datasets.

Data	ABOD	CBLOF	HBOS	IForest	KNN	LODA	LOF	LSCP	OCSVM	PCA	SUOD	ECOD
Arrhythmia (mat)	0.769 (11)	0.784 (7)	**0.822 (1)**	0.802 (3)	0.786 (6)	0.75 (12)	0.779 (10)	0.793 (5)	0.781 (9)	0.782 (8)	0.811 (2)	0.802 (3)
Breastw (mat)	0.396 (12)	0.965 (6)	0.983 (4)	0.987 (2)	0.976 (5)	0.987 (2)	0.47 (11)	0.933 (9)	0.961 (7)	0.959 (8)	0.929 (10)	**0.994 (1)**
Cardio (mat)	0.569 (12)	0.81 (9)	0.835 (8)	0.923 (3)	0.724 (10)	0.856 (7)	0.574 (11)	0.88 (6)	0.935 (2)	**0.95 (1)**	0.9 (4)	0.897 (5)
Ionosphere (mat)	0.925 (2)	0.897 (3)	0.561 (12)	0.847 (7)	**0.927 (1)**	0.798 (10)	0.875 (4)	0.857 (6)	0.842 (8)	0.796 (11)	0.862 (5)	0.831 (9)
Lympho (mat)	0.911 (11)	0.967 (10)	**0.996 (1)**	0.993 (3)	0.975 (9)	0.73 (12)	0.977 (7)	0.984 (5)	0.976 (8)	0.985 (4)	0.979 (6)	0.994 (2)
Mammography (mat)	0.549 (12)	0.821 (10)	0.834 (9)	0.86 (5)	0.841 (8)	0.881 (3)	0.729 (11)	0.855 (6)	0.872 (4)	0.882 (2)	0.851 (7)	**0.894 (1)**
Optdigits (mat)	0.467 (9)	0.769 (2)	**0.873 (1)**	0.721 (4)	0.371 (12)	0.398 (11)	0.45 (10)	0.658 (6)	0.5 (8)	0.509 (7)	0.68 (5)	0.733 (3)
Pima (mat)	0.679 (3)	0.658 (8)	0.7 (2)	0.678 (4)	**0.708 (1)**	0.592 (12)	0.627 (10)	0.664 (6)	0.622 (11)	0.648 (9)	0.666 (5)	0.664 (6)
Satellite (mat)	0.571 (11)	0.749 (2)	**0.758 (1)**	0.702 (4)	0.684 (5)	0.597 (10)	0.557 (12)	0.665 (6)	0.662 (7)	0.599 (9)	0.722 (3)	0.661 (8)
Satimage-2 (mat)	0.819 (11)	**0.999 (1)**	0.98 (9)	0.995 (3)	0.954 (10)	0.987 (5)	0.458 (12)	0.981 (8)	0.998 (2)	0.982 (7)	0.993 (4)	0.985 (6)
Shuttle (mat)	0.659 (9)	0.608 (10)	0.995 (4)	**0.998 (1)**	0.738 (8)	0.589 (11)	0.524 (12)	0.939 (7)	0.995 (4)	0.994 (6)	0.996 (3)	**0.998 (1)**
Speech (mat)	**0.627 (1)**	0.452 (9)	0.454 (8)	0.447 (11)	0.458 (6)	0.455 (7)	0.471 (3)	0.465 (5)	0.447 (11)	0.45 (10)	0.469 (4)	0.485 (2)
Wbc (mat)	0.905 (12)	0.92 (9)	0.952 (2)	0.931 (7)	0.937 (3)	0.913 (11)	0.935 (4)	0.931 (7)	0.932 (6)	0.916 (10)	0.935 (4)	**0.975 (1)**
Wine (mat)	0.431 (11)	0.284 (12)	0.9 (3)	0.816 (6)	0.518 (10)	0.787 (8)	0.905 (2)	0.862 (5)	0.636 (9)	0.801 (7)	0.868 (4)	**0.949 (1)**
Arrhythmia (arff)	0.74 (11)	0.751 (6)	0.749 (9)	0.757 (3)	0.751 (6)	0.703 (12)	0.749 (9)	0.757 (3)	0.756 (5)	0.751 (6)	**0.773 (1)**	0.762 (2)
Cardiotocography (arff)	0.459 (12)	0.567 (9)	0.594 (8)	0.683 (3)	0.49 (11)	0.674 (4)	0.505 (10)	0.645 (6)	0.698 (2)	**0.755 (1)**	0.66 (5)	0.637 (7)
HeartDisease (arff)	0.597 (6)	0.593 (7)	**0.689 (1)**	0.602 (4)	0.614 (3)	0.482 (12)	0.563 (10)	0.599 (5)	0.555 (11)	0.576 (9)	0.587 (8)	0.673 (2)
Hepatitis (arff)	0.716 (11)	0.759 (10)	0.796 (6)	0.794 (7)	0.811 (4)	0.608 (12)	0.831 (2)	0.784 (8)	0.764 (9)	0.804 (5)	0.814 (3)	**0.843 (1)**
InternetAds (arff)	0.643 (6)	0.617 (10)	**0.699 (1)**	0.685 (2)	0.622 (8)	0.528 (12)	0.606 (11)	0.675 (4)	0.623 (7)	0.622 (8)	0.673 (5)	0.676 (3)
Ionosphere (arff)	**0.925 (1)**	0.886 (3)	0.552 (12)	0.836 (8)	0.919 (2)	0.822 (9)	0.86 (5)	0.855 (6)	0.838 (7)	0.787 (11)	0.862 (4)	0.822 (9)
KDDCup99 (arff)	0.693 (11)	**0.997 (1)**	0.995 (5)	**0.997 (1)**	0.761 (9)	0.728 (10)	0.577 (12)	0.96 (8)	0.995 (5)	**0.997 (1)**	0.995 (5)	**0.997 (1)**
Lymphography (arff)	0.986 (11)	0.996 (7)	0.998 (3)	0.999 (1)	0.997 (4)	0.839 (12)	0.995 (8)	0.995 (8)	0.992 (10)	0.997 (4)	0.997 (4)	**0.999 (1)**
Pima (arff)	0.667 (3)	0.666 (4)	0.659 (6)	0.676 (2)	**0.706 (1)**	0.597 (12)	0.658 (7)	0.661 (5)	0.64 (11)	0.655 (8)	0.65 (9)	0.647 (10)
Shuttle (arff)	0.834 (10)	0.98 (2)	0.792 (11)	0.866 (9)	0.961 (4)	0.707 (12)	**0.986 (1)**	0.941 (6)	0.966 (3)	0.929 (7)	0.955 (5)	0.88 (8)
SpamBase (arff)	0.435 (11)	0.548 (7)	0.67 (2)	0.633 (3)	0.54 (8)	0.445 (10)	0.426 (12)	0.588 (4)	0.539 (9)	0.555 (6)	0.562 (5)	**0.701 (1)**
Stamps (arff)	0.762 (10)	0.73 (12)	0.918 (2)	0.913 (4)	0.874 (9)	0.901 (6)	0.739 (11)	0.906 (5)	0.88 (8)	0.917 (3)	0.898 (7)	**0.939 (1)**
Waveform (arff)	0.652 (10)	0.714 (5)	0.683 (7)	0.681 (8)	0.74 (3)	0.622 (12)	0.735 (4)	0.76 (2)	0.665 (9)	0.634 (11)	0.706 (6)	**0.787 (1)**
WBC (arff)	0.956 (11)	0.966 (10)	0.982 (3)	**0.991 (1)**	0.972 (8)	0.981 (4)	0.932 (12)	0.967 (9)	0.975 (6)	0.974 (7)	0.981 (4)	0.99 (2)
WDBC (arff)	0.903 (12)	0.909 (11)	0.967 (2)	0.946 (3)	0.927 (9)	0.946 (3)	0.92 (10)	0.937 (6)	0.938 (5)	0.929 (8)	0.932 (7)	**0.983 (1)**
WPBC (arff)	0.457 (12)	0.492 (11)	0.536 (2)	0.516 (6)	0.526 (3)	0.503 (8)	0.511 (7)	0.524 (5)	0.498 (9)	0.496 (10)	0.526 (3)	**0.547 (1)**
AVG	0.69 (12)	0.762 (8)	0.797 (5)	0.809 (2)	0.76 (9)	0.713 (10)	0.697 (11)	0.801 (4)	0.783 (7)	0.788 (6)	0.808 (3)	**0.825 (1)**



(a) Critical difference diagram of ROC



(b) Critical difference diagram of AP

Fig. 4: Critical difference diagram showing pairwise statistical difference comparison of ECOD and the baselines regarding both ROC and AP. ECOD outperforms all baselines, and the result is statistically significant regarding ROC.

4.3 Performance Comparison with Baselines

4.3.1 Overall Results

ECOD **consistently outperforms regarding both ROC and AP**. Of the 30 datasets in Table 1, ECOD scores the highest in terms of ROC (see Table 4). It achieves an average ROC score of 0.825, which is 2% higher than the second best alternative—iForest. Notably, iForest has been the SOTA method in many large-scale outlier analysis [54], [61]. Moreover, ECOD ranks first in 13 out of 30 occasions and ranks in the top three places in 21 out of 30 occasions. The critical difference plot in Fig. 4 corroborates the findings—it shows that ECOD outperforms with a statistical significance.

Similarly, Table 5 shows that ECOD is also superior to the baselines regarding AP. Of the 12 OD methods, ECOD achieves an average AP score of 0.565, which bring 5% improvement over the second place—iForest. Additionally, ECOD ranks first in 12 out of 30 datasets, and ranks in the top three places on 20 datasets. The further analysis with the critical difference plot in Fig. 4 corroborates that ECOD outperforms the baselines.

Additionally, it is worthy to note that ECOD (avg. ROC: 0.825, avg. AP: 0.565) is better than HBOS (avg. ROC: 0.795, avg. AP: 0.509) with a similar mechanism but estimate histogram per dimension instead. We credit the edge to: (1) ECOD uses more information than HBOS—the latter does not capture the difference for the samples in the same bin and (2) HBOS needs deciding the number of bins, which is hard to tune under unsupervised settings.

4.3.2 Case Study

Although ECOD outperforms on most of the datasets, we notice that its performance degrades markedly on some of the datasets, as shown in Table 4 and 5. In the case study, we further investigate by visualizing the selected datasets in 2-D by t-SNE [62].

In Fig. 3, we first present two datasets (*Breastw (mat)* and *Shuttle (mat)*) where ECOD outperforms all baselines, and then two datasets (*Ionosphere (mat)* and *Speech (mat)*) where it does not achieve top performance. The visualization suggests that when outliers locates at the tails in at least some dimensions, ECOD could accurate capture them. However, its performance degrades when outliers are well mingled

TABLE 5: Average precision (AP) of detector performance (average of 10 independent trials, highest score highlighted in bold); rank is shown in parenthesis (lower is better). ECOD outperforms all baselines with the highest avg. performance, and ranks highest in 11 out of 30 datasets.

Data	ABOD	CBLOF	HBOS	IForest	KNN	LODA	LOF	LSCP	OCSVM	PCA	SUOD	ECOD
Arrhythmia (mat)	0.359 (12)	0.399 (9)	**0.493 (3)**	**0.506 (1)**	0.397 (10)	0.436 (5)	0.374 (11)	0.411 (6)	0.405 (7)	0.402 (8)	0.495 (2)	0.473 (4)
Breastw (mat)	0.295 (12)	0.914 (8)	0.953 (5)	0.972 (3)	0.927 (7)	0.978 (2)	0.322 (11)	0.826 (9)	0.934 (6)	0.96 (4)	0.791 (10)	**0.988 (1)**
Cardio (mat)	0.194 (11)	0.414 (8)	0.46 (6)	0.576 (3)	0.345 (10)	0.424 (7)	0.163 (12)	0.396 (9)	0.532 (4)	**0.611 (1)**	0.462 (5)	0.579 (2)
Ionosphere (mat)	0.914 (2)	0.871 (3)	0.366 (12)	0.789 (8)	**0.924 (1)**	0.716 (11)	0.821 (6)	0.818 (7)	0.823 (5)	0.736 (9)	0.824 (4)	0.719 (10)
Lympho (mat)	0.517 (11)	0.808 (9)	**0.925 (2)**	**0.952 (1)**	0.82 (7)	0.417 (12)	0.825 (6)	0.861 (4)	0.814 (8)	0.852 (5)	0.799 (10)	0.894 (3)
Mammography (mat)	0.023 (12)	0.137 (9)	0.122 (10)	0.233 (3)	0.17 (6)	0.281 (2)	0.118 (11)	0.169 (7)	0.188 (5)	0.199 (4)	0.164 (8)	**0.429 (1)**
Optdigits (mat)	0.028 (8)	0.062 (2)	**0.196 (1)**	0.055 (3)	0.022 (12)	0.024 (11)	0.029 (7)	0.043 (6)	0.028 (8)	0.028 (8)	0.046 (5)	0.053 (4)
Pima (mat)	0.511 (4)	0.477 (7)	**0.569 (1)**	0.503 (5)	**0.515 (3)**	0.408 (12)	0.43 (11)	0.47 (9)	0.461 (10)	0.478 (6)	0.473 (8)	0.541 (2)
Satellite (mat)	0.397 (11)	**0.69 (1)**	**0.687 (2)**	0.654 (3)	0.543 (9)	0.581 (8)	0.39 (12)	0.51 (10)	0.653 (4)	0.603 (6)	0.608 (5)	0.585 (7)
Satimage-2 (mat)	0.187 (11)	**0.978 (1)**	0.758 (7)	0.929 (3)	0.419 (9)	0.87 (5)	0.027 (12)	0.333 (10)	0.975 (2)	0.874 (4)	0.623 (8)	0.86 (6)
Shuttle (mat)	0.171 (11)	0.195 (10)	0.98 (3)	**0.986 (1)**	0.204 (9)	0.379 (8)	0.142 (12)	0.715 (7)	0.902 (6)	0.926 (4)	0.913 (5)	**0.981 (2)**
Speech (mat)	**0.04 (1)**	0.022 (6)	0.027 (3)	0.018 (11)	0.022 (6)	0.017 (12)	0.024 (5)	0.027 (3)	0.021 (9)	0.022 (6)	0.029 (2)	0.02 (10)
Wbc (mat)	0.355 (12)	0.5 (11)	0.663 (2)	0.59 (4)	0.529 (9)	0.564 (5)	0.558 (6)	0.554 (7)	0.514 (10)	0.534 (8)	0.602 (3)	**0.783 (1)**
Wine (mat)	0.084 (11)	0.06 (12)	0.405 (2)	0.279 (6)	0.095 (10)	0.278 (7)	0.361 (4)	0.299 (5)	0.141 (9)	0.254 (8)	0.364 (3)	**0.608 (1)**
Arrhythmia (arff)	0.699 (11)	0.712 (8)	0.75 (3)	0.746 (4)	0.712 (8)	0.697 (12)	0.704 (10)	0.718 (5)	0.716 (6)	0.714 (7)	**0.751 (2)**	**0.752 (1)**
Cardiotocography (arff)	0.247 (12)	0.363 (9)	0.366 (8)	0.434 (2)	0.311 (10)	0.432 (3)	0.258 (11)	0.374 (7)	0.419 (4)	**0.478 (1)**	0.398 (5)	0.378 (6)
HeartDisease (arff)	0.534 (4)	0.521 (9)	**0.625 (2)**	0.534 (4)	0.538 (3)	0.445 (12)	0.478 (11)	0.525 (8)	0.513 (10)	0.53 (6)	0.528 (7)	**0.64 (1)**
Hepatitis (arff)	0.33 (12)	0.374 (11)	0.473 (6)	0.442 (8)	0.475 (5)	0.396 (10)	0.499 (4)	0.472 (7)	0.427 (9)	0.543 (3)	0.557 (2)	**0.585 (1)**
InternetAds (arff)	0.276 (10)	0.315 (8)	**0.535 (1)**	0.49 (3)	0.281 (9)	0.242 (12)	0.262 (11)	0.387 (5)	0.316 (7)	0.32 (6)	0.475 (4)	0.51 (2)
Ionosphere (arff)	**0.915 (1)**	0.862 (3)	0.364 (12)	0.777 (8)	**0.915 (1)**	0.763 (9)	0.811 (7)	0.819 (6)	0.822 (5)	0.723 (10)	0.835 (4)	0.703 (11)
KDDCup99 (arff)	0.018 (12)	**0.198 (5)**	**0.278 (1)**	**0.273 (2)**	0.046 (10)	0.135 (7)	0.028 (11)	0.127 (8)	0.125 (9)	**0.199 (4)**	0.157 (6)	**0.239 (3)**
Lymphography (arff)	0.801 (11)	0.925 (7)	0.978 (3)	**0.992 (1)**	0.942 (6)	0.491 (12)	0.925 (7)	0.922 (9)	0.837 (10)	0.953 (4)	0.944 (5)	**0.982 (2)**
Pima (arff)	0.506 (5)	0.484 (7)	0.528 (2)	0.515 (4)	**0.525 (3)**	0.42 (12)	0.458 (11)	0.48 (9)	0.478 (10)	0.484 (7)	0.487 (6)	**0.53 (1)**
Shuttle (arff)	0.263 (5)	0.358 (3)	0.08 (12)	0.13 (10)	0.36 (2)	0.132 (9)	**0.395 (1)**	0.184 (7)	0.304 (4)	0.232 (6)	0.171 (8)	0.085 (11)
SpamBase (arff)	0.38 (10)	0.414 (8)	0.525 (2)	0.492 (3)	0.422 (6)	0.375 (11)	0.349 (12)	0.45 (4)	0.412 (9)	0.42 (7)	0.431 (5)	**0.567 (1)**
Stamps (arff)	0.247 (10)	0.241 (11)	0.398 (5)	0.394 (6)	0.341 (9)	0.404 (4)	0.229 (12)	0.425 (2)	0.346 (8)	0.411 (3)	0.392 (7)	**0.464 (1)**
Waveform (arff)	0.066 (8)	0.129 (2)	0.055 (10)	0.056 (9)	**0.134 (1)**	0.052 (12)	0.108 (4)	0.114 (3)	0.07 (6)	0.054 (11)	0.07 (6)	**0.079 (5)**
WBC (arff)	0.542 (11)	0.556 (9)	0.694 (6)	**0.863 (1)**	0.581 (8)	0.724 (4)	0.33 (12)	0.543 (10)	0.702 (5)	0.638 (7)	0.728 (3)	0.838 (2)
WDBC (arff)	0.43 (12)	0.667 (9)	0.795 (2)	0.718 (6)	0.654 (10)	0.794 (3)	0.704 (7)	0.773 (4)	0.612 (11)	0.688 (8)	0.729 (5)	**0.841 (1)**
WPBC (arff)	0.21 (12)	0.224 (8)	0.23 (6)	0.231 (5)	0.233 (4)	0.223 (9)	0.225 (7)	**0.248 (1)**	0.223 (9)	0.223 (9)	0.245 (2)	**0.244 (3)**
AVG	0.351 (12)	0.462 (8)	0.509 (3)	0.538 (2)	0.447 (9)	0.436 (10)	0.378 (11)	0.466 (7)	0.49 (6)	0.503 (4)	0.503 (4)	**0.565 (1)**

with normal points (Fig. 3, subfigure c) or hidden in the middle of normal points regarding all dimensions (Fig. 3, subfigure d). We want to point out that it is unlikely that an outlier resembles normal points in all dimensions, and this explains why ECOD could consistently work for most of the datasets.

TABLE 6: ECOD's runtime (in seconds) on a moderate laptop under different sample size (n) and dimensions (d). It scales well to handle high-dimensional, large datasets.

	d=10	d=100	d=1,000	d=10,000
n=1,000	0.068	0.173	1.163	11.460
n=10,000	0.171	0.468	5.244	55.190
n=100,000	0.640	7.185	70.541	567.105
n=1,000,000	11.403	130.974	694.405	5376.593

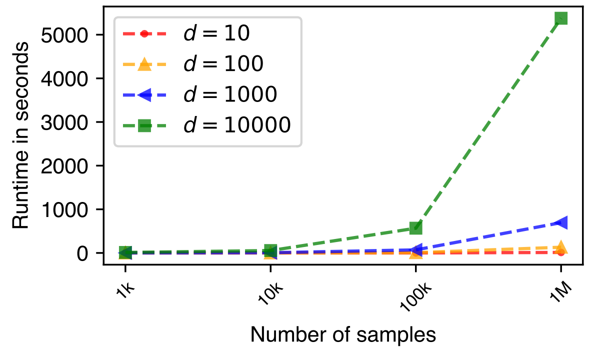


Fig. 5: Runtime of ECOD on the synthetic datasets with varying number of samples and dimensions. It is efficient and scalable to handle a million samples with 10,000 dimensions in less than 2 hours.

4.4 Runtime Efficiency and Scalability

ECOD **is an efficient algorithm with fast computation**. Table 7 shows that of the 12 algorithms tested, ECOD ranks third in runtime—an average of 0.228 seconds to process each dataset. The fast two algorithms are HBOS and LODA, which are known to be efficient by building uni-dimensional histograms. Intuitively, ECOD's efficiency is attributed to the feature independence assumption.

ECOD **is also a scalable algorithm that suits high dimensional settings**. Unlike proximity-based models that require extensive distance calculation or density estimation, ECOD incurs very little computational overhead. We carry out the experiment by generating synthetic datasets of dimensions 10, 100, 1,000, and 10,000 and the number of observations are 1,000, 10,000, 100,000, and 1,000,000. All datasets are randomly generated and used strictly for evaluating ECOD's computation time. As such, we ignore the performance metrics and only keep performance time. Fig. 5 and Table 6 illustrate ECOD's performance and its scalability across varying the number of samples (n) and dimensions (d). Even on the moderately equipped personal computer, ECOD can easily handle datasets with 10,000 dimensions and 1,000,000 data points in 2 hours. The result is consistent with our complexity analysis in Section 3.3.2—ECOD has $\mathcal{O}(nd)$ time complexity that scales linearly in both the number of samples and dimensions.

TABLE 7: Run time (in seconds) of detectors (average of 10 independent trials, the fastest is highlighted in bold); rank is shown in parenthesis (lower is better). ECOD is one of the fastest algorithms.

Data	**ABOD**	**CBLOF**	**HBOS**	**IForest**	**KNN**	**LODA**	**LOF**	**LSCP**	**OCSVM**	**PCA**	**SUOD**	**ECOD**
Arrhythmia (mat)	0.236 (7)	0.258 (8)	**0.201 (6)**	**0.306 (10)**	0.077 (5)	0.05 (2)	0.066 (4)	0.984 (11)	**0.042 (1)**	0.058 (3)	2.004 (12)	0.266 (9)
Breastw (mat)	0.17 (9)	0.076 (8)	0.004 (2)	0.384 (10)	0.038 (7)	0.037 (6)	0.007 (3)	0.524 (11)	0.01 (4)	**0.002 (1)**	1.431 (12)	**0.024 (5)**
Cardio (mat)	0.546 (10)	0.175 (8)	0.011 (2)	0.426 (9)	0.158 (7)	0.06 (4)	0.116 (6)	1.512 (11)	0.086 (5)	**0.005 (1)**	1.811 (12)	0.053 (3)
Ionosphere (mat)	0.083 (9)	0.053 (8)	0.012 (4)	0.271 (10)	**0.015 (5)**	0.025 (6)	0.006 (3)	0.45 (11)	0.004 (2)	**0.003 (1)**	1.458 (12)	0.047 (7)
Lympho (mat)	0.036 (8)	0.051 (9)	**0.009 (5)**	**0.253 (10)**	0.008 (4)	0.023 (6)	0.003 (3)	0.32 (11)	**0.001 (1)**	0.002 (2)	1.399 (12)	0.036 (8)
Mammography (mat)	1.701 (10)	0.268 (6)	**0.006 (1)**	0.689 (8)	0.473 (7)	0.091 (4)	0.255 (5)	4.6 (12)	1.364 (9)	0.007 (2)	2.517 (11)	**0.044 (3)**
Optdigits (mat)	2.154 (10)	0.433 (5)	**0.032 (1)**	0.725 (6)	1.457 (9)	0.063 (3)	1.371 (8)	14.566 (12)	1.169 (7)	0.046 (2)	7.114 (11)	0.229 (4)
Pima (mat)	0.176 (9)	0.072 (8)	**0.001 (1)**	0.267 (10)	**0.03 (7)**	0.023 (6)	0.009 (4)	0.577 (11)	0.007 (3)	0.003 (2)	1.452 (12)	0.023 (6)
Satellite (mat)	1.602 (10)	**0.399 (5)**	**0.018 (1)**	0.669 (6)	0.867 (8)	0.071 (3)	0.835 (7)	8.549 (12)	1.075 (9)	0.024 (2)	5.173 (11)	0.158 (4)
Satimage-2 (mat)	1.43 (10)	**0.354 (5)**	0.017 (2)	0.546 (6)	0.667 (8)	0.063 (3)	0.64 (7)	7.699 (12)	0.91 (9)	0.017 (2)	4.742 (11)	0.146 (4)
Shuttle (mat)	2.01 (10)	0.175 (4)	0.006 (2)	**0.634 (6)**	0.788 (8)	0.076 (3)	0.728 (7)	7.973 (12)	1.33 (9)	**0.003 (1)**	3.566 (11)	**0.189 (5)**
Speech (mat)	**7.85 (10)**	2.036 (5)	0.167 (2)	3.589 (6)	7.842 (9)	**0.154 (1)**	7.274 (8)	49.828 (12)	4.272 (7)	1.052 (3)	14.31 (11)	1.148 (4)
Wbc (mat)	0.08 (9)	0.06 (8)	0.008 (4)	0.239 (10)	0.015 (5)	0.021 (6)	0.007 (3)	0.47 (11)	0.004 (2)	**0.002 (1)**	1.495 (12)	**0.055 (7)**
Wine (mat)	0.026 (8)	0.046 (9)	0.005 (5)	0.243 (10)	0.005 (5)	0.02 (7)	0.002 (3)	0.313 (11)	**0.001 (1)**	0.002 (3)	1.421 (12)	**0.019 (6)**
Arrhythmia (arff)	0.226 (8)	0.23 (9)	0.169 (6)	0.249 (10)	0.065 (4)	0.043 (2)	0.057 (3)	0.947 (11)	**0.039 (1)**	0.095 (5)	**2.025 (12)**	**0.226 (8)**
Cardiotocography (arff)	0.368 (10)	0.124 (7)	0.007 (2)	0.286 (9)	0.138 (8)	0.049 (4)	0.099 (6)	1.769 (11)	0.084 (5)	**0.007 (2)**	1.876 (12)	0.037 (3)
HeartDisease (arff)	0.048 (9)	0.044 (8)	**0.005 (4)**	0.21 (10)	0.008 (5)	0.018 (7)	0.004 (3)	0.366 (11)	**0.001 (1)**	0.002 (2)	1.38 (12)	**0.016 (6)**
Hepatitis (arff)	0.013 (6)	0.028 (9)	0.004 (5)	0.187 (10)	0.003 (4)	0.016 (7)	0.002 (3)	0.292 (11)	0.001 (2)	0.001 (2)	1.378 (12)	**0.021 (8)**
InternetAds (arff)	5.648 (9)	1.36 (3)	**0.493 (2)**	5.605 (8)	5.252 (6)	**0.227 (1)**	6.11 (10)	58.435 (12)	4.159 (5)	5.496 (7)	21.974 (11)	1.855 (4)
Ionosphere (arff)	**0.075 (9)**	0.049 (8)	0.012 (4)	0.236 (10)	**0.014 (5)**	0.02 (6)	0.007 (3)	0.447 (11)	0.004 (2)	0.004 (2)	1.498 (12)	0.036 (7)
KDDCup99 (arff)	136.469 (9)	**2.1 (5)**	**0.135 (1)**	**7.714 (6)**	133.6 (8)	0.409 (3)	141.9 (10)	839.8 (12)	175.1 (11)	**0.301 (2)**	68.726 (7)	**0.965 (4)**
Lymphography (arff)	0.039 (8)	0.053 (9)	0.007 (5)	**0.255 (10)**	0.007 (5)	0.023 (7)	0.003 (3)	0.31 (11)	**0.001 (1)**	0.002 (2)	1.503 (12)	**0.02 (6)**
Pima (arff)	0.201 (9)	0.087 (8)	0.004 (2)	0.319 (10)	**0.034 (6)**	0.029 (5)	0.014 (4)	0.573 (11)	0.013 (3)	**0.002 (1)**	1.43 (12)	**0.043 (7)**
Shuttle (arff)	0.248 (8)	0.079 (7)	0.004 (2)	0.366 (9)	0.05 (6)	0.033 (5)	**0.023 (4)**	0.712 (10)	0.018 (3)	**0.003 (1)**	1.481 (12)	0.912 (11)
SpamBase (arff)	1.329 (10)	0.303 (5)	**0.017 (1)**	0.514 (6)	0.904 (9)	0.061 (3)	0.888 (8)	8.349 (12)	0.666 (7)	0.032 (2)	3.836 (11)	**0.109 (4)**
Stamps (arff)	0.071 (9)	0.055 (8)	0.004 (4)	0.233 (10)	0.013 (5)	0.018 (7)	0.003 (3)	0.384 (11)	0.002 (2)	0.002 (2)	1.382 (12)	**0.014 (6)**
Waveform (arff)	1.018 (10)	0.244 (5)	0.01 (2)	0.478 (8)	**0.521 (9)**	0.072 (4)	0.429 (7)	3.479 (12)	0.261 (6)	**0.008 (1)**	2.357 (11)	**0.047 (3)**
WBC (arff)	0.048 (8)	0.05 (9)	0.004 (4)	**0.243 (10)**	0.009 (5)	0.021 (7)	0.003 (3)	0.335 (11)	0.001 (2)	0.001 (2)	1.367 (12)	0.013 (6)
WDBC (arff)	0.092 (9)	0.062 (8)	0.009 (4)	0.245 (10)	0.016 (5)	0.022 (6)	0.007 (3)	0.466 (11)	0.005 (2)	**0.004 (1)**	1.5 (12)	**0.032 (7)**
WPBC (arff)	0.046 (7)	0.052 (8)	0.01 (5)	0.237 (10)	0.008 (4)	0.021 (6)	0.004 (3)	**0.353 (11)**	**0.002 (1)**	0.003 (2)	1.383 (12)	**0.062 (9)**
AVG	5.468 (9)	0.312 (5)	**0.046 (1)**	0.887 (6)	5.105 (7)	0.062 (2)	5.362 (8)	33.847 (12)	6.355 (11)	0.24 (4)	5.5 (10)	**0.228 (3)**

Additionally, ECOD requires no re-training fit new data points with relatively large data samples if no data shift is assumed, and thus is highly desirable for applications where real time predictions are needed. Within PyOD, we also provide the distributed implementation with even more efficiency and scalability.

5 Conclusion and Future Works

In this paper, we present a novel outlier detection method called ECOD that uses empirical cumulative distribution functions in measuring outlyingness of data points. Specifically, it computes left- and right-tail univariate ECDFs per dimension. For every data point, ECOD uses the univariate ECDFs to estimate tail probabilities for the data point and aggregates these tail probabilities to come up with a final outlier score. The intuition of ECOD somewhat resembles how p-value works: for a specific data point, we are looking at how low its tail probability is. The proposed ECOD is parameter-free without the need of hyperparameter tuning. Our extensive evaluation on 30 benchmark datasets show that ECOD outperforms SOTA baselines, while being a fast and scalable outlier detection algorithm. We provide an easy-to-use Python implementation of ECOD for both single- and multi-thread support. We leave accounting for how features are related for future work. For instance, we can learn vine copulas [63] or probabilistic graphical models [64] to identify blocks of features that are highly-correlated, and only for these blocks (rather than across the entire dataset), we could compute a joint ECDF. We suspect that these strategies would require some amount of hyperparameter tuning though. Separately, our approach inherently is not designed to handle multimodal distributions for which an outlier could be in neither left nor right tails. Figuring out a way to extend ECOD to such settings while still being fast and scalable is also a possible direction worth exploring.

References

[1] X. Yang, L. Tan, and L. He, "A robust least squares support vector machine for regression and classification with noise," *Neurocomputing*, vol. 140, pp. 41–52, 2014.

[2] M. Najafi, L. He, and S. Y. Philip, "Outlier-robust multi-aspect streaming tensor completion and factorization." in *IJCAI*, 2019, pp. 3187–3194.

[3] B. Cao, M. Mao, S. Viidu, and P. Yu, "Collective fraud detection capturing inter-transaction dependency," in *KDD 2017 Workshop on Anomaly Detection in Finance*. PMLR, 2018, pp. 66–75.

[4] J. Tao, J. Lin, S. Zhang, S. Zhao, R. Wu, C. Fan, and P. Cui, "Mvan: Multi-view attention networks for real money trading detection in online games," in *KDD*, 2019, pp. 2536–2546.

[5] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 315–324.

[6] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *CSUR*, vol. 41, no. 3, p. 15, 2009.

[7] S. Zhang, B. Li, J. Li, M. Zhang, and Y. Chen, "A novel anomaly detection approach for mitigating web-based attacks against clouds," in *2015 IEEE 2nd International Conference on Cyber Security and Cloud Computing*. IEEE, 2015, pp. 289–294.

[8] W. Yu, J. Li, M. Z. A. Bhuiyan, R. Zhang, and J. Huai, "Ring: Real-time emerging anomaly monitoring system over text streams," *IEEE Transactions on Big Data*, vol. 5, no. 4, pp. 506–519, 2017.

[9] J. Zhao, X. Liu, Q. Yan, B. Li, M. Shao, and H. Peng, "Multi-attributed heterogeneous graph convolutional network for bot detection," *Information Sciences*, vol. 537, pp. 380–393, 2020.

[10] P. Cui, L.-F. Sun, Z.-Q. Liu, and S.-Q. Yang, "A sequential monte carlo approach to anomaly detection in tracking visual events," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[11] S. Chen, W. Wang, and H. van Zuylen, "A comparison of outlier detection algorithms for its data," *Expert Systems with Applications*, vol. 37, no. 2, pp. 1169–1178, 2010.

[12] R. Xu, Y. Guo, X. Han, X. Xia, H. Xiang, and J. Ma, "Opencda: an open cooperative driving automation framework integrated with co-simulation," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2021, pp. 1155–1162.

[13] R. Xu, H. Xiang, X. Xia, X. Han, J. Liu, and J. Ma, "Opv2v: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication," *arXiv preprint arXiv:2109.07644*, 2021.

[14] Z. Li, Y. Zhao, and J. Fu, "SynC: A copula based framework for generating synthetic data from aggregated sources," in *IEEE International Conference on Data Mining Workshops (ICDMW)*, 2020.

[15] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *ACM SIGMOD Record*, vol. 29, 2000, pp. 427–438.

[16] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[17] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*. IEEE, 2008, pp. 413–422.

[18] Y. Zhao, Z. Nasrullah, M. K. Hryniewicki, and Z. Li, "LSCP: locally selective combination in parallel outlier ensembles," in *SDM*. SIAM, May 2019, pp. 585–593.

[19] L. Ruff, N. Görnitz, L. Deecke, S. A. Siddiqui, R. A. Vandermeulen, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, ser. Proceedings of Machine Learning Research, J. G. Dy and A. Krause, Eds., vol. 80. PMLR, 2018, pp. 4390–4399. [Online]. Available: http://proceedings.mlr.press/v80/ruff18a.html

[20] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, and X. He, "Generative adversarial active learning for unsupervised outlier detection," *TKDE*, 2019.

[21] Y. Zhao, X. Hu, C. Cheng, C. Wang, C. Wan, W. Wang, J. Yang, H. Bai, Z. Li, C. Xiao, Y. Wang, Z. Qiao, J. Sun, and L. Akoglu, "SUOD: Accelerating large-scale unsupervised heterogeneous outlier detection," *Proceedings of Machine Learning and Systems*, 2021.

[22] Y. Zhao, G. H. Chen, and Z. Jia, "Tod: Tensor-based outlier detection," *arXiv preprint arXiv:2110.14007*, 2021.

[23] Y. Zhao, R. Rossi, and L. Akoglu, "Automatic unsupervised outlier model selection," in *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[24] L. Akoglu, "Anomaly mining - past, present and future," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, Z. Zhou, Ed. ijcai.org, 2021, pp. 4932–4936. [Online]. Available: https://doi.org/10.24963/ijcai.2021/697

[25] A. Lazarevic and V. Kumar, "Feature bagging for outlier detection," in *KDD*. ACM, 2005, pp. 157–166.

[26] D. Pokrajac, A. Lazarevic, and L. J. Latecki, "Incremental local outlier detection for data streams," in *2007 IEEE symposium on computational intelligence and data mining*. IEEE, 2007, pp. 504–515.

[27] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of experimental social psychology*, vol. 49, no. 4, pp. 764–766, 2013.

[28] Z. A. Bakar, R. Mohemad, A. Ahmad, and M. M. Deris, "A comparative study for outlier detection techniques in data mining," in *2006 IEEE conference on cybernetics and intelligent systems*. IEEE, 2006, pp. 1–6.

[29] F. M. Dekking, C. Kraaikamp, H. P. Lopuhaä, and L. E. Meester, *A Modern Introduction to Probability and Statistics: Understanding why and how*. Springer Science & Business Media, 2005.

[30] M. Goldstein and A. Dengel, "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm," *KI-2012: Poster and Demo Track*, pp. 59–63, 2012.

[31] M. Naaman, "On the tight constant in the multivariate Dvoretzky-Kiefer-Wolfowitz inequality," *Statistics & Probability Letters*, vol. 173, p. 109088, 2021.

[32] C. C. Aggarwal, "Outlier analysis," in *Data mining*. Springer, 2015, pp. 237–263.

[33] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021.

[34] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: identifying density-based local outliers," in *ACM SIGMOD Record*, vol. 29, 2000, pp. 93–104.

[35] J. Tang, Z. Chen, A. Fu, and D. Cheung, *LOF: Enhancing effectiveness of outlier detections for low density patterns*. Springer, 2002.

[36] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos, "Loci: Fast outlier detection using the local correlation integral," in *ICDE*. IEEE, 2003, pp. 315–326.

[37] X. Yang, L. Latecki, and D. Pokrajac, "Outlier detection with globally optimal exemplar-based gmm," in *SDM*. SIAM, April 2009, pp. 145–154.

[38] M. H. Satman, "A new algorithm for detecting outliers in linear regression," *Int. J. Statist. Probab.*, vol. 2, no. 3, pp. 101–109, 2013.

[39] M. Pavlidou and G. Zioutas, *Kernel density outlier detector*. Springer, 2014.

[40] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1641–1650, 2003.

[41] J. Chen, S. Sathe, C. Aggarwal, and D. Turaga, "Outlier detection with autoencoder ensembles," in *Proceedings of the 2017 SIAM international conference on data mining*. SIAM, 2017, pp. 90–98.

[42] G. Pang, A. v. d. Hengel, C. Shen, and L. Cao, "Toward deep supervised anomaly detection: Reinforcement learning from partially labeled anomaly data," *arXiv preprint arXiv:2009.06847*, 2020.

[43] C. J. Stone, "Optimal rates of convergence for nonparametric estimators," *The Annals of Statistics*, pp. 1348–1360, 1980.

[44] R. A. Groeneveld and G. Meeden, "Measuring skewness and kurtosis," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 33, no. 4, pp. 391–399, 1984.

[45] X. Hu, C. Rudin, and M. Seltzer, "Optimal sparse decision trees," *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[46] X. Hu, Y. Huang, B. Li, and T. Lu, "Uncovering the source of machine bias," *ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Machine Learning for Consumers and Markets Workshop*, 2021.

[47] N. Gupta, D. Eswaran, N. Shah, L. Akoglu, and C. Faloutsos, "Beyond outlier detection: Lookout for pictorial explanation," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2018, pp. 122–138.

[48] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, "Breast cancer diagnosis and prognosis via linear programming," *Operations Research*, vol. 43, no. 4, pp. 570–577, 1995.

[49] S. Rayana, "Odds library," 2016. [Online]. Available: http://odds.cs.stonybrook.edu

[50] G. O. Campos, A. Zimek, J. Sander, R. J. G. B. Campello, B. Micenková, E. Schubert, I. Assent, and M. E. Houle, "On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study," *Data Mining Knowledge Discovery*, vol. 30, no. 9-10, pp. 891–927, 2016.

[51] C. C. Aggarwal and S. Sathe, *Outlier ensembles: An introduction*, 1st ed. Springer, 2017.

[52] L. Akoglu, H. Tong, J. Vreeken, and C. Faloutsos, "Fast and Reliable Anomaly Detection in Categorical Data," in *CIKM*, 2012.

[53] Y. Zhao and M. K. Hryniewicki, "Xgbod: improving supervised outlier detection with unsupervised representation learning," in *IJCNN*. IEEE, 2018.

[54] A. Emmott, S. Das, T. Dietterich, A. Fern, and W.-k. Wong, "A Meta-Analysis of the Anomaly Detection Problem," *arXiv preprint*, no. 2015, 2015. [Online]. Available: http://arxiv.org/abs/1503.01158

[55] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[56] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data mining and knowledge discovery*, vol. 33, no. 4, pp. 917–963, 2019.

[57] H.-P. Kriegel, M. Schubert, and A. Zimek, "Angle-based outlier detection in high-dimensional data," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 444–452.

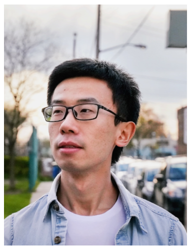
[58] T. Pevný, "LODA: Lightweight on-line detector of anomalies," *Machine Learning*, vol. 102, no. 2, pp. 275–304, 2016.

[59] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," MIAMI UNIV CORAL GABLES FL DEPT OF ELECTRICAL AND COMPUTER ENGINEERING, Tech. Rep., 2003.

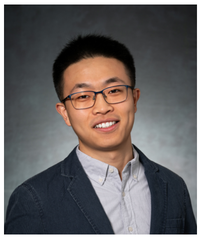
[60] Y. Zhao, Z. Nasrullah, and Z. Li, "Pyod: A python toolbox for scalable outlier detection," *Journal of Machine Learning Research*, vol. 20, no. 96, pp. 1–7, 2019. [Online]. Available: http://jmlr.org/papers/v20/19-011.html

[61] M. Q. Ma, Y. Zhao, X. Zhang, and L. Akoglu, "A large-scale study on unsupervised outlier model selection: Do internal strategies suffice?" *arXiv preprint arXiv:2104.01422*, 2021.
[62] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.
[63] H. Joe and D. Kurowicka, *Dependence modeling: vine copula handbook*. World Scientific, 2011.
[64] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC, 2019.

Zheng Li is the founder of Arima, a Toronto based data mining startup and an adjunct lecturer at Northeastern University Toronto.



Yue Zhao is a Ph.D. student at Carnegie Mellon University (CMU), focusing on designing and building automated, scalable, and accelerated machine learning systems (MLSys), with realization and applications in outlier detection, healthcare, graph neural networks, and ensemble learning. At CMU, he has collaborated and been mentored by Prof. Leman Akoglu, Prof. George H. Chen, and Prof. Zhihao Jia. He has coauthored more than 20 papers on anomaly detection, AutoML, and machine learning systems in leading venues including JMLR, TKDE, NeurIPS, MLsys, and AAAI. He is a recipient of the prestigious 2022 Norton Labs Graduate Fellowship (formerly known as the Symantec Research Labs Fellowship).



Xiyang Hu is a Ph.D. student at Carnegie Mellon University. He got his M.Sc. in Statistical Science from Duke University, and B.Arch. in Architecture with a minor in Computer Science from Tsinghua University. His research interests include machine learning, deep learning, data mining, and social impacts of AI.

Nicola Botta has worked on numerical methods for conservation laws, agent-based modelling and verified decision making. He is a senior scientist at the Potsdam Institute for Climate Impact Research, Potsdam, Germany and adjunct professor in the Functional Programming division at the Computer Science and Engineering Department, Chalmers University of Technology, Sweden.

Cezar Ionescu is Professor of Computer Science at Technische Hochschule Deggendorf, Deggendorf, Germany.

George H. Chen is an assistant professor of information systems at Carnegie Mellon University. He primarily works on building trustworthy machine learning models for time-to-event prediction (survival analysis) and for time series analysis. He often uses nonparametric prediction models that work well under very few assumptions on the data. His main application area is in healthcare.