DIP-ECOD: Improving Anomaly Detection in Multimodal Distributions

Kaixi Yang[1,*,†], Paul Miller[1,*,†] and Jesús Martinez-del-Rincon[1,*,†]

[1]Centre for Secure Information Technologies(CSIT), Queen's University Belfast, Belfast, United Kingdom

Abstract

Anomaly detection algorithms identify unusual events and outliers in large datasets where manual approaches are highly impractical. Most prior anomaly detection methods assume simple unimodal Gaussian data distributions; however, they produce suboptimal results on complex multimodal distributions. To address this problem, we propose Dip-ECOD, a novel anomaly detection algorithm leveraging unsupervised machine learning that generalises to both multimodal and unimodal distributions. Dip-ECOD integrates a dip test within the ECOD framework, using SkinnyDip to split a probability distribution into separate modes, after which ECOD is applied. In this way, difficult-to-find outliers between modes and hidden in the distribution tails of each mode are also detected. Experiments using nine benchmark datasets across a range of domains such as healthcare and imagery demonstrate Dip-ECOD's improved performance over ECOD in detecting outliers in both multimodal and unimodal distributions, with Dip-ECOD achieving an average AUC score of 0.791 compared to ECOD's 0.761. Further, using a proprietary enterprise dataset, we show Dip-ECOD effectively identifies anomalous GitHub commits, indicating its applicability to information security and software vulnerability, where multimodal distributions are expected.

Keywords

anomaly detection, unsupervised learning, modality testing, distributed learning

1. Instructions

Outliers refer to rare events, data points, and unusual behaviors that do not follow the same trends, distributions, or patterns as the majority of a dataset. Outlier detection (OD) algorithms have a range of important applications, for example, intrusion detection [1], log anomaly detection [2], malware detection [3], fraud detection [4], and medical diagnosis [5]. Proximity-based approaches and machine-learning algorithms previously obtained promising results, although often led to high computation costs and suffered from the curse of dimensionality [6], [7]. Recent work [8], [9], [10], [11] developed distribution-based approaches by fitting probability distributions to data points. However, they assumed unimodal distributions [8], [12], meaning a performance drop on more complex data distributions. While unimodal Gaussian distributions are a common assumption that fits many tasks, it does not hold for more nuanced problems where multiple phenomena or data sources can cause abnormality. In this regard, there is little existing work focused on multimodal datasets [13],[14], [15] or detecting anomalies within multimodal distributions. With the majority of real-world datasets [1], [4], [5] being multimodal, we are thus motivated to investigate this under-researched area of OD.

This paper presents Dip-ECOD, a novel unsupervised learning method for effective anomaly detection within multimodal distributions. Our proposed approach generalises Empirical Cumulative-distribution-based Outlier Detection (ECOD) [8] from simple unimodal Gaussian distributions to more widely applicable multimodal cases. This is achieved using the statistical dip test [16] of unimodality and integrating it within the ECOD framework. We assume each dataset dimension is a multimodal Gaussian, a generalisation of unimodal Gaussian. Inspired by SkinnyDip [17], we apply the dip test [16] recursively to detect all the areas that cover the least frequent value between the modes, which we call anti-mode intervals [18], and use midpoints of each anti-mode interval as the split points for separating multimodal

CAMLIS'24: Conference on Applied Machine Learning for Information Security, October 24–25, 2024, Arlington, VA

*Corresponding author.

†These authors contributed equally.

kyang03@qub.ac.uk (K. Yang); p.miller@qub.ac.uk (P. Miller); j.martinez-del-rincon@qub.ac.uk (J. Martinez-del-Rincon)

distributions into anti-mode distributions. After dividing the data distribution into separate unimodal distributions, ECOD is applied to each anti-mode distribution to detect outliers at the tail of each mode's distribution.

Dip-ECOD's performance is first demonstrated on multiple benchmark datasets from varying real-world domains such as healthcare and imagery. Afterward, we further show its applicability to information security by evaluating Dip-ECOD against a proprietary enterprise cybersecurity dataset of GitHub commits. Identifying unusual and suspicious GitHub commits can prevent malicious actors from executing software attacks, though in reality, it is impossible to check all codebase commits by hand so OD algorithms are required.

Our contributions are:

- A novel unsupervised anomaly detection method, Dip-ECOD, performs on both multimodal and unimodal distributions, including the first use of the statistical dip test within OD.
- An extensive evaluation on nine different real-world benchmark datasets showing Dip-ECOD's wider general-purpose applicability. Dip-ECOD outperforms eight other state-of-the-art methods to achieve an average AUC score of 0.791.
- A further evaluation using a proprietary enterprise dataset of GitHub commits showing its suitability to information security.

The rest of the paper is organised as follows. In Section 2, related work on OD is reviewed. Our approach, Dip-ECOD, is discussed in Section 3. The proposed model is evaluated in Section 4 while Section 5 contains experimental results and analysis. Section 6 presents our conclusions and discusses future work.

2. Related Work

In this section, relevant anomaly detection approaches and software vulnerability detection in the literature are reviewed.

2.1. Distribution-based approaches

Anomaly detection methods based on out-of-distribution assumptions are implemented by fitting probability distributions to data points, where the anomalies are those that have a very low probability. Common distribution-based approaches for anomaly detection include the Gaussian Mixture Model (GMM) [10], linear regression [11] or Kernel Density Estimation (KDE) [9]. However, those methods either require tuning hyperparameters or have high computational costs [8]. To address the sensitivity to key parameters, a parameter-free, interpretable, and fast algorithm was introduced by Li. et al [8], called ECOD. ECOD is an unsupervised learning anomaly detection technique based on empirical cumulative distribution function (ECDF) [19]. ECOD is inspired by the fact that outliers are often the rare events that appear in the tails of a distribution. For each dimension, a univariate ECDF was computed; then outlier scores were calculated by multiplying all of the estimated tail probabilities from each of the univariate ECDFs. However, ECOD is built on the assumption of unimodal distributions, and so may not perform well on multimodal distributions where for instance the outliers lie between modes not detectable by ECOD. It has been stated the majority of datasets are multimodal [20], meaning it is worthwhile for us to investigate a method to handle the multimodal scenario. Therefore, taking inspiration from [8] and [17], we propose a novel anomaly detection method called Dip-ECOD to solve this problem of anomaly detection from data with multimodal distributions, with our motivation being that rare events are also located in the shallows between frequent events rather than solely in the tails.

2.2. GitHub repository analysis for software vulnerability

There are many studies on detecting vulnerabilities in GitHub repositories. The existing methods analyse GitHub repositories from three aspects: source code patches [21], [22], [23], [24], commit messages [25], and commits logs [26], [27], [28].

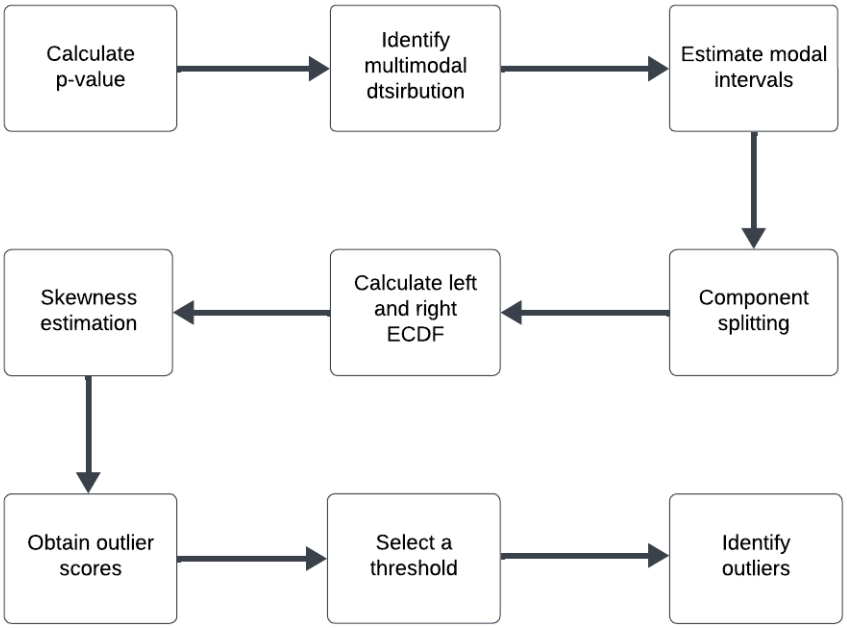


Figure 1: Overview of DIP-ECOD approach

Among the most recent approaches, deep Learning architectures and neural networks, such as RNN [21], [25] and hierarchical attention networks [22] have been proposed reporting good results using either code change features or commit messages. However, those methods need the complete code that is built into the packages or needs to access the whole source code files, which means that the code snippets are not suitable for the methods and the methods are slow and cumbersome to be used in real-time at commit upload. They also required a large amount of annotated training data given their supervised approach, so they rely on either synthetic source code [21], such as the Juliet Test Suite [29] or privately annotated scrapped datasets [25].

Simpler approaches, less reliant on annotated data have also been proposed. In this respect, commit logs are also analysed to detect anomalous commits. Gonzalez. et al build a rule-based model to detect malicious commits from the metadata of commit logs [26]. The metadata features they considered include lines of code added, removed, and modified, the number of files, and the number of unique file types modified. Although they successfully detect more than 50% of the malicious commits, there are still high false positives. It shows that there is still space to improve. Therefore, we proposed detecting software vulnerabilities by analysing commit logs using unsupervised approaches.

3. Methodology

In this section, we propose Dip-ECOD, our novel anomaly detection method. This approach is based on the assumption that a dataset distribution is an equally weighted multimodal mixture Gaussian, with rare events lying both in the tails and between the mixture components. Our aim is to generalise the anomaly detection method ECOD [8] to multimodal datasets. We also assume the outliers are rare events whose data points occur in low-density regions of the probability distribution [30, 31].

Figure. 1 presents the architecture of our proposed approach, Dip-ECOD. Our approach involves first estimating the modality of each dimension from data by using a dip test, a p-value is calculated and modal intervals are returned. If it is unimodal, the ECDF is estimated straightaway; otherwise component splitting is applied to locate each component's modal intervals recursively, followed by taking the middle points between the modal intervals to split a multimodal distribution into individual components. Then ECDF of each component is estimated and the skewness is calculated. The outlier

score for each data point is obtained by concatenating the outlier scores for each dimension. More details are demonstrated in the following section.

3.1. Modality estimation

We estimate the modality of a dataset first. In the case of multimodality, as most real datasets are expected [20], DIP-ECOD's component splitting will detect and isolate each modality. Otherwise, if it happens to be unimodal, our approach becomes equivalent to ECOD. More details can be found in Section 3.2.

Hartigan's dip test [16] is a common approach to test the modality of distributions. In the dip test, the null hypothesis assumes that the observed dataset is drawn from an unimodal distribution, which means that there is only one mode in the distribution. The alternative hypothesis suggests that the dataset is derived from a multimodal distribution, meaning that there is more than one mode.

Dip statistic dip calculates the maximum difference between the empirical distribution function of the observed dataset, denoted $F(x)$, and the best-fitting unimodal function $G(x)$ that minimises that maximum difference, which represents the departure of the samples from unimodal distribution and it can be calculated as

$$dip = max_x|F(x) - G(x)| \tag{1}$$

with $dip \in (0, 0.25]$, where if a dip value closes to zero, the distribution is unimodal, while the distribution is deemed multimodal if dip is close to 0.25. $G(x)$ can be obtained by estimating the greatest convex minorant (g.c.m) and least concave majorant (l.c.m) of $F(x)$ [16], [32]. The modal interval (X_L, X_U), which indicates the region of the steepest slope in the $F(x)$, i.e. the g.c.m of $F(x)$ is in $(-\infty, X_L]$ and the l.c.m of $F(x)$ is in $[X_U, \infty)$. The g.c.m of $F(x)$ in $(-\infty, X_L]$ means that function $G(x)$ is convex in $(-\infty, X_L]$ and nothing is greater than $F(x)$. In contrast, l.c.m of $F(x)$ in $[X_U, \infty)$ means that function $G(x)$ is concave in $[X_U, \infty)$ and nothing is less than $F(x)$ [16]. In other words, $F(x)$ increases in $(-\infty, X_L]$ and decreases in $[X_U, \infty)$. The modal interval will be used in the following stages of DIP-ECOD.

To assess the significance of the dip statistic, given the sample size n and the computed dip statistic dip, the p-value can be calculated using the formula

$$p - value = \sqrt{n} * dip \tag{2}$$

A p-value represents the probability of occurrence of the given events. In other words, it measures the difference between the data and the null hypothesis using an estimate of the parameter of interest. The smaller the p-value, the stronger the evidence against the null hypothesis. Note the p-value is affected by the sample size. The larger the sample size, the smaller the p-value [33]. As the sample size increases, the uncertainty about where the population mean lies may decrease. With a very large sample, the standard error becomes extremely small, so that even minuscule distances between the estimate and the null hypothesis become statistically significant [34].

We then compare the p-value with a threshold α to determine whether our p-value is low enough to reject the null hypothesis, i.e. where the dataset is unimodal. The higher the α, the higher the probability of rejecting the null hypothesis and therefore identifying the data as multimodal. On the other hand, the lower the α, the lower the probability of identifying the data as multimodal. After careful consideration, we choose α = 0.001 as our default setting for all the experiments.

3.2. Cumulative distribution function (CDF) and empirical cumulative distribution function (ECDF)

A cumulative distribution function (CDF) tells us the probability that a random variable X takes on a value less than or equal to x, which is usually denoted as $F_x(x)$. The CDF of a random variable X is formally defined by:

$$F_x(x) = P(X \leq x) \tag{3}$$

where $P(X \leq x)$ is the probability that the random variable X takes on a value less than or equal to x. A CDF has four properties: it is non-decreasing and right-continuous [35], with $\lim_{x \to -\infty} F_x(x) = 0$ and $\lim_{x \to +\infty} F_x(x) = 1$, illustrating the CDF approaches 1 as x becomes large and vice versa.

To estimate a CDF, an empirical cumulative distribution function (ECDF) [19] is used. It assigns a probability density of $\frac{1}{n}$ to each datum, orders the data from smallest to largest by data value, and calculates the sum of the assigned probability densities up to and including each datum. The result is a step function that increases by $\frac{1}{n}$ at each datum. The ECDF, $\bar{F_x}(x)$, is formally defined as:

$$\bar{F_x}(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}(X \leq x) \tag{4}$$

where $\mathbb{1}(X \leq x)$ is an indicator of the event $X \leq x$.

3.3. Mixture distribution

A mixture distribution, also called a multimodal distribution, is a collection of probability distributions. Each distribution that forms the mixture distribution is called a mixture component and can be regarded as an unimodal distribution. A weight is associated with each component to model its influence on the overall distribution. The number of components in a mixture distribution can be finite or infinite. However, in practical terms, a finite number of components should suffice to model the mixture distribution within a defined error ϵ. In our case, we only assume that the mixture distributions we investigate have a finite number of components.

Given a finite set of CDF, $F_1(x)$, $F_2(x)$,..., $F_n(x)$ and weights w_1, w_2,..., w_n such that $w_i \leq 0$ and, the CDF of the mixture function can be represented by $\tilde{F(x)}$ as a sum, formally:

$$\tilde{F(x)} = \sum_{i=1}^{n} w_i F_i(x) \tag{5}$$

where $w_i F_i(x)$ can relate to the combination of individual distributions $F_i(x)$ (or components) and their associated weight w_i.

3.4. Component splitting

The main idea of our method is to generalise ECOD to multimodal distribution. To do so, a process called component splitting is introduced to split the multimodal distribution into individual unimodal distributions or components. In this section, we will focus on explaining how we split multimodal distributions.

Once a multimodal distribution is identified, (see sec.3.1), the intervals of each mode from each component in each multimodal distribution, called modal intervals, are also identified by using [17]. Next, the intervals outside the modal intervals are the areas where the modes do not local, i.e. anti-modal interval, can also be identified. Then, the middle points are obtained by taking the middle points between two anti-modal intervals for component purposes. Consequently, the multimodal distribution is split into multiple unimodal distributions or components according to the middle points. Finally, ECOD can be applied to each component to calculate the outlier scores.

We assume that we have a multimodal dataset with n data points $X = \{X_i\}_{i=1}^{n} \in \mathrm{R}^{n \times d}$. There are m components or modes or modal intervals in the dataset and the index of each component is represented as p. The location of each modal interval p, i.e. $[X_{L_p}^{(j)}, X_{U_p}^{(j)}]$ of each component $\{C_p^{(j)}\}_{p=1}^{m}$ at dimension j are returned by using SkinnyDip from [17], where $X_{L_p}^{(j)}$ and $X_{U_p}^{(j)}$ present lower bound and upper bound of the modal interval. As a result, a set of the modal intervals of m components is obtained, denoted as: $\{[X_{L1}^{(j)}, X_{U1}^{(j)}], ..., [X_{Lm}^{(j)}, X_{Um}^{(j)}]\}$.

The split point can be located according to the modal intervals obtained from the previous step. Knowing that there are m components in total, the middle point between each of two modal intervals

is taken as the split point $s = \{s_k\}_{k=1}^{m-1} \in R$ where represents the k^{th} split point of the j^{th} dimension and there are up to $m-1$ split points by using the function below.

$$s_k^{(j)} = X_{U(p-1)}^{(j)} + \frac{X_{Lp}^{(j)} - X_{U(p-1)}^{(j)}}{2} \tag{6}$$

where $X_{U(p-1)}^{(j)}$ is the upper bound from the first component and $X_{Lp}^{(j)}$ is the lower bound from the next component which beside the first component. As a result, the multimodal distribution is split into multiple unimodal components, where each component $C_p^{(j)}$ consists of all the data points with the range of two split points, i.e. $C_1^{(j)} = \{[X_1^{(j)}, s_1^{(j)}]\}$. Each component consists of q data points where $q = \{q_w\}_{w=1}^{m} = |\{\{C_p^{(j)}\}_{p=1}^{m}\}| = |s_p^{(j)} - s_{p-1}^{(j)}| \in R$. For instance, q_2 can be formulated as $q_2 = |\{C_2^{(j)}\}| = |s_2^{(j)} - s_1^{(j)}|$.

Hence all the components are separated as individual unimodal distribution. A similar procedure as ECOD is applied to measure how anomalous each X_i is inside each component, the left and right tail ECDFs of each component are estimated using the following equations:

$$\hat{F}^{(j)}_{l(C_p^{(j)})}(z_w) = \frac{1}{q_w} \sum_{i=1}^{q_w} \mathbb{1}\{X_i^{(j)} \leq z_w\} \tag{7}$$

$for\ z_w \in C_p^{(j)}$

$$\hat{F}^{(j)}_{r(C_p^{(j)})}(z_w) = \frac{1}{q_w} \sum_{i=1}^{q_w} \mathbb{1}\{X_i^{(j)} \geq z_w\} \tag{8}$$

$for\ z_w \in C_p^{(j)}$

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

Therefore, across all the dimensions d, based on the assumption that different dimensions are independent of each other, we estimate the joint left and right tail ECDFs for each data point as:

$$\begin{aligned} \hat{F}_l(x) &= \prod_{j=1}^{d} \hat{F}_l^{(j)}(x^{(j)}) \\ \hat{F}_r(x) &= \prod_{j=1}^{d} \hat{F}_r^{(j)}(x^{(j)}) \end{aligned} \tag{9}$$

$for\ x \in R^d$

After estimating the joint left and right ECDF from the previous step, as a part of the computation of the outlier score for each data point, skewness is also needed. Skewness is the degree of asymmetry of a distribution. A distribution can have right (or positive), left (or negative), or zero skewness. To decide which tail should be kept, the skewness of each component is computed by using the function:

$$\gamma_{j_w} = \frac{\frac{1}{q_w} \sum_{i=1}^{q_w} (X_i^{(j)} - \tilde{X_w}^{(j)})^3}{[\frac{1}{q_w - 1} \sum_{i=1}^{q_w} X_j^{(j)} - \tilde{X_w}^{(j)}]^{\frac{3}{2}}} \tag{10}$$

where $\tilde{X_w}^{(j)} = \frac{1}{q_w} X_i^{(j)}$ is the data point mean of the $j^{(th)}$ feature in each component.

Finally, we move to the stage to calculate the outlier score for each data point. Since the values of ECDFs are in the range of $[0, 1]$ and the potential outliers have lower ECDFs. The negative log probability is applied to aggregate the tail probabilities. As a result, lower tail probability corresponds

to higher outlier scores and vice versa. The left, right, and auto outlier score O_i for the point X_i of the j^{th} dimension are calculated as the equations shown below:

$$O_l(X_i) = -\sum_{j=1}^{d} log(\hat{F}_l^{(j)}(X_i^{(j)})) \tag{11}$$

$$O_r(X_i) = -\sum_{j=1}^{d} log(\hat{F}_r^{(j)}(X_i^{(j)}))] \tag{12}$$

$$\begin{aligned} O_{auto}(X_i) = -\sum_{j=1}^{d} [\mathbb{1}\{\gamma_j < 0\} log(\hat{F}_l^{(j)}(X_i^{(j)})) \\ +\mathbb{1}\{\gamma_j \geq 0\} log(\hat{F}_r^{(j)}(X_i^{(j)}))] \end{aligned} \tag{13}$$

and then the highest tail probability among $O_l(X_i)$, $O_r(X_i)$ and $O_{auto}(X_i)$ is taken as the outlier score O_i for X_i in each dimension shown as:

$$O_i = max(O_l(X_i), O_r(X_i), O_{auto}(X_i)) \tag{14}$$

Lastly, the outlier score $O_i \in [0, \infty)$ for every point X_i is obtained. The higher the outlier score, the more likely it is an outlier. Note that the outlier scores cannot be interpreted as the probabilities, they are only used for relative comparison across data points. The pseudocode of Dip-ECOD is displayed in Algorithm 1.

Algorithm 1 Unsupervised Outlier Detection using Dip-ECOD

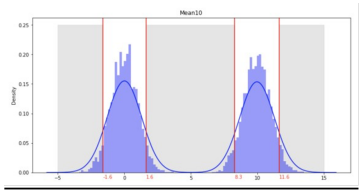
Input:input data $X = \{X_i\}_{i=1}^{n} \in \mathrm{R}^{n \times d}$ with n data point and d features; $X_i^{(j)}$ refers to the value of the j^{th} feature of the i^{th} data point.

Output: outlier score $O := DipECOD(X) \in \mathrm{R}^n$

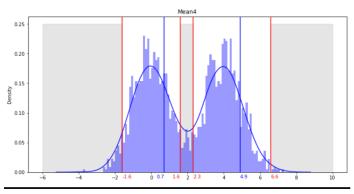
for each dimension j in $1, ..., d$ **do**
  Estimate modal intervals $[X_{L_p}^{(j)}, X_{U_p}^{(j)}]$ of each component $C_p^{(j)}$
  Obtain split points $s = \{s_k\}_{k=1}^{m-1} \in$R,s$_k^{(j)}$ refers to the k^{th} split point of the j^{th} feature
  Split multimodal distribution into m components $\{C_p^{(j)}\}_{p=1}^{m}$ according to split points $s_k^{(j)}$, where each components consist of $q = \{q_w\}_{w=1}^{m} \in R$ data points
  for each component p in $1, ..., m$ **do**
    Estimate left and right tail ECDFs for each componen
    Compute the data point skewness coefficient γ_{j_w}
  end for
end for
for each data point i in 1,..., n **do**
  Aggregate tail probability X_i to obtain left, right and auto outlier scores $O_l^{(j)}(X_i)$, $O_r^{(j)}(X_i)$ and $O_{auto}^{(j)}(X_i)$
  Select the highest outlier score for X_i as $O^{(j)}$
end for
Obtain the final outlier score O_i
return $O = (O_1, ..., O_n)$

4. Experimental Settings

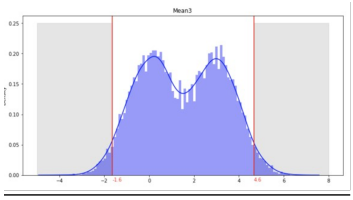
In this section, we present in detail the datasets used to evaluate the performance of Dip-ECOD, the performance measure, and the baseline methods we compare with.



(a) μ_1 = 0 and μ_2 = 10.The modal interval is [-1.6, 1.6] for the first component and [8.3, 11.6] for the second component and the split point is 5.



(b) μ_1 = 0 and μ_2 = 4.The modal interval is [-1.6, 1.6] for the first component and [2.3, 6.6] for the second component and the split point is 1.95.



(c) μ_1 = 0 and μ_2 = 3.The modal interval is [-1.6, 1.6] for the first component and [1.4, 4.6] for the second component and the split point is 1.5.

Figure 2: A PDF of a 1-dimensional bimodal distribution dataset with different values of μ_1 and μ_2 for the first component and the second component respectively. The outliers are in grey areas. The red lines are split points.

4.1. Dataset

We perform three types of evaluations in our validation, including synthetic datasets, nine public benchmark datasets commonly used in previously published work [8], and a proprietary enterprise dataset in software vulnerabilities. More details on each type of dataset are discussed in the following subsections.

4.1.1. Synthetic dataset

We first create a synthetic dataset to evaluate the capability of the proposed method in a multimodal scenario. A bi-dimensional bimodal synthetic dataset of 1,000 data points is built, where the distribution of each dimension is multimodal or consists of more than one Gaussian.

Moreover, for evaluating our model on different shapes of the multimodal dataset, we have one fixed standard Gaussian, μ = 0 and σ^2 = 1; plus a second moving Gaussian which also has a σ^2 = 1 but its mean moves from 10 to 2 on each iteration, in increments of 1. In this way, the distance between two Gaussian changes. The larger the mean of the second moving Gaussian, the fewer the overlapping points, and the more distinguishable the outliers. The data points present below the 5^{th} percentile and above the 95^{th} percentile for each Gaussian have been labeled as outliers for Gaussian centered between 6 and 10 since there are no overlapping points between two Gaussian. Figure 2a shows an example of how we label the data points when μ = 10 for each dimension. In Figure 2a, the grey area represents outliers, with the red lines being the 5^{th} percentile and the 95^{th} percentile of each Gaussian.

For Gaussian centered at 4 and 5, the data points below the 5^{th} percentile for the fixed Gaussian, and above the 95^{th} percentile for the moving Gaussian are labeled as outliers. For the overlapping points on both fixed and moving Gaussian, we label as outliers the data points that are above the 95^{th} percentile of fixed Gaussian and below the 5^{th} percentile of the moving Gaussian, per Figure 2b. For Gaussian centered $\leq$ 3, we only label as outliers data points which are below the 5^{th} percentile for the fixed Gaussian, and above the 95^{th} percentile for the moving Gaussian, per Figure 2. The data points in the middle are not included in the outliers.

4.1.2. Public benchmark datasets

Table 1 summarises the nine public anomaly detection benchmark datasets selected for this study, including their dimensionality d and proportion of outliers. These benchmark datasets are part of ODDS database [20] which also provides ground truth. The ODDS database was developed in 2016 to provide real-world anomaly detection datasets with labels. Furthermore, the datasets cover different application areas and domains, for instance, medicine (arrhythmia, wbc, pima, and lympo) and aerospace imagery (satellite and satimage-2). The selected datasets are all multidimensional, which means there is more than one feature in each dataset.

Table 1
9 real-world benchmark datasets used in this study for evaluation.

Dataset	# samples	dimension	% outliers
arrhythmia	452	274	14.601
optdigits	5216	64	2.875
pima	768	8	34.895
satimage2	5803	36	1.223
satellite	6435	36	31.639
wine	129	13	7.751
wbc	278	30	5.555
speech	3686	400	1.654
lympo	148	18	4.054

4.1.3. Proprietary enterprise dataset

To highlight the applicability of Dip-ECOD for software vulnerability specifically, and for information security in general, we also evaluate against the task of detecting anomalous outliers from real-world GitHub commits in a software repository. Although this dataset is proprietary and non-public, we feel it is worthwhile to provide the insights gained from using Dip-ECOD on this dataset.

After selecting an up-to-date commercial repository, we extracted commits over the last five years (2018 January-2023 January) from the author who made the most commits, 743 commits. An example of a GitHub commit is shown in Figure 3. Only contextual features such as the environmental data around the changed code are extracted. We do this because we would like to detect the anomalous behaviors from a high-level aspect instead of from the content data i.e. changed code itself. This approach is faster and more privacy-preserving than analysing code. The contextual features for our experiments include changed code, commit ID, number of changed files, number of added files, number of deleted files, and commit time by using git and then saving as a CSV file for analysis. Expected outliers could be anomalous working patterns of the author, unusual code commits or deletions, etc.

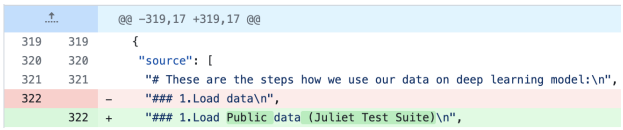


Figure 3: An example of contextual features from a GitHub commit.

4.2. Experimental setup and evaluation metrics

We split each dataset, using 60% for the training set and 40% for the testing set, with a 10-fold stratification used to help avoid overfitting and artificially high results. Our main focus is to see how much the model is capable of distinguishing between classes. Area Under the Curve (AUC) score is calculated from the area under the Receiver Operating Characteristic curve (ROC). The positive case is an outlier, and the negative case is a non-outlier, or normal, data point. The ROC curve is created by plotting the true positive rate (TPR) or sensitivity against the false positive rate (FPR) at various threshold settings. The calculation of TPR and FPR are formally shown in Equation 15 and Equation 16:

$$TPR = \frac{TP}{TP + FN} \tag{15}$$

$$FPR = \frac{FP}{FP + TN} \tag{16}$$

AUC represents the degree or measure of class separability by varying TPR and FPR. It tells us to what extent the model can distinguish between classes. The higher the AUC, the better the model predicts

each class correctly. In our case, this is how well the model distinguishes between outliers and normal data points. An excellent model has an AUC near 1, meaning it has a good measure of separability. A poor model has an AUC near 0, meaning it has a worsening measure of separability, predicting the negative class as positive and the positive class as negative. When AUC is 0.5, it means the model has no class separation capacity.

4.3. State-of-the-art comparison setup

We compare the performance of Dip-ECOD with ECOD [8] and seven other state-of-the-art outlier detection algorithms. This variety of algorithms ensures the state-of-the-art comparison is more robust. The seven other outlier detection algorithms are Clustering-Based Local Outlier Factor (CBLOF)[36], Histogram-based Outlier Score (HBOS)[37], Isolation Forest (IForest) [38], k Nearest Neighbors (KNN) [39], Lightweight On-line Detector of Anomalies (LODA) [40], Local Outlier Factor(LOF) [41], and PCA-based outlier detector (PCA) [42].

5. Results

To evaluate the performance of Dip-ECOD we conduct experiments on a variety of synthetic data and real-world data sets, comparing Dip-ECOD's results with those of the state-of-art algorithms. AUC is used to measure the overall performance of each algorithm.

5.1. Synthetic dataset evaluations

First, to demonstrate Dip-ECOD is suitable for multi-dimensional multimodal datasets, we tested on the synthetic datasets described in Section 4.1.1. As illustrated in Figure 4, the Gaussian lies along the diagonal, i.e. the two dimensions are independent of each other. Three experiments are performed; varying the means of the moving Gaussian, varying α of the Dip Test, and varying the number of modes.

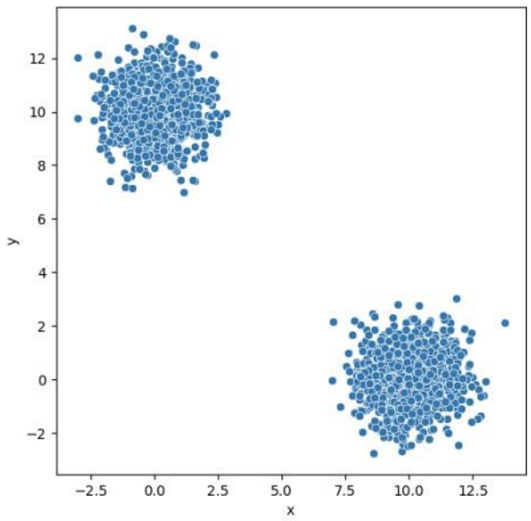


Figure 4: 2-D synthetic bimodal data.

5.1.1. Varying the means μ of the moving Gaussian

The first experiment focuses on illustrating the main contribution of the proposed algorithm, which is performance against multimodal distributions. By changing the mean of the moving Gaussian, the modality and how obvious that modality is also changes. The larger the value of the mean, the more

likely the multimodality. The AUC of varying means of the moving Gaussian is displayed in Figure 5. It can be seen that DIP-ECOD's AUC is significantly better than that of ECOD.

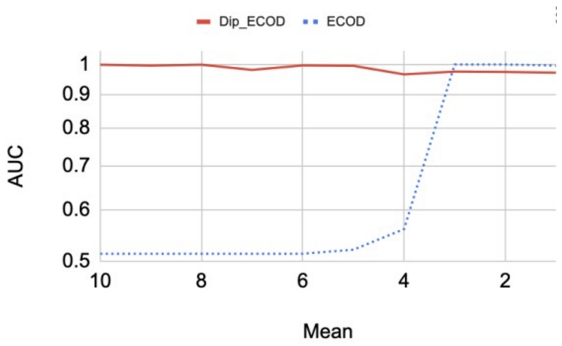


Figure 5: Average AUC of DIP-ECOD and ECOD across varying means of the moving Gaussian.

However, when the mean of the moving Gaussian is less than 3, i.e. the two Gaussian are closer together, DIP-ECOD performance is similar but slightly lower than ECOD. This is likely due to distribution approaching unimodality where ECOD performs slightly better. This result proves two things: firstly, DIP-ECOD performs well on multimodal data, and secondly, ECOD indeed works well, as expected, on unimodal data.

5.1.2. Varying α of the Dip Test

As section 3.1 discussed, the threshold α determines whether our p-value is low enough to reject the null hypothesis, i.e. where the dataset is unimodal. An optimal α could balance Type I and Type II errors. In this case, α = 0.001 is the default setting for our remaining experiments. Note that the larger the sample size, the smaller the p-value. This is because the larger the data size, the more accurate the statistical test, and therefore, the stronger the evidence rejecting or supporting the null hypothesis [33].

5.1.3. Varying the number of components

This third evaluation focuses on the impact of varying the number of components or modes. We created additional synthetic datasets with distributions beyond the bimodal dataset but still allocated 1,000 data points for each Gaussian. We have one fixed standard Gaussian, μ = 0 and σ^2 =1, and we have multiple Gaussian which also have a σ^2 =1, with their means being multiples of 1 to 10. By using α = 0.001, the number of components from 2 to 10 is explored.

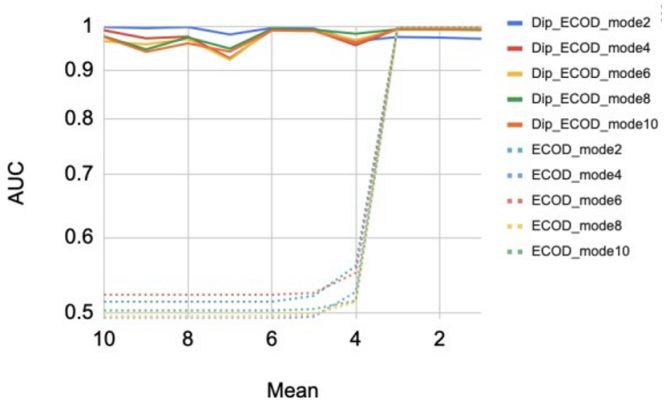


Figure 6: Average AUC of DIP-ECOD and ECOD across varying number of components.

Figure 6 shows there is no significant change in AUC as the number of components varies. We can also see that our model outperforms ECOD in multimodal cases. In summary, the number of

components does not have a significant effect on the performance of our model, and it strongly proves that DIP-ECOD outperforms ECOD on multimodal distributions.

5.2. Real-world benchmark dataset evaluations

To demonstrate the generalisation potential of our approach to real-world cases, we evaluate DIP-ECOD on the previously discussed nine selected publicly available benchmark datasets against eight other state-of-the-art methods. The datasets are of varying modality types to test the performance of the DIP-ECOD model in both multimodal and unimodal scenarios.

Per Table 2, results show DIP-ECOD consistently outperforms all other models, including ECOD. Overall, DIP-ECOD achieves an average AUC score of 0.791, compared to the ECOD AUC of only 0.761 and the lowest AUC average from LOF of 0.617. This demonstrates DIP-ECOD performs better than the other models when it comes to classifying outliers and normal samples. Moreover, DIP-ECOD reaches a higher AUC than ECOD on all the Gaussian multimodal datasets and matches ECOD on all the strongly unimodal datasets, especially the real-world datasets that have a mix of unimodal features and multimodal features. In addition, DIP-ECOD generalises better than ECOD in multimodal mixture Gaussian cases while maintaining strong performance on the unimodal datasets. Although it is only a 0.03 improvement on AUC, it could reduce False Negative rates dramatically on large datasets, such as a dataset with 100,000 samples.

Table 2
Average AUC on benchmark datasets.

	Dataset	ECOD	DIP-ECOD	LOF	IForest	HBOS	LODA	KNN	CBLOF	PCA
Multimodal	arrhythmia	0.852	**0.855**	0.678	0.641	0.700	0.598	0.692	0.654	0.675
	optdigits	0.584	**0.734**	0.479	0.472	0.483	0.468	0.487	0.479	0.486
	pima	0.602	**0.626**	0.588	0.553	0.532	0.567	0.558	0.543	0.544
	satimage-2	0.965	**0.988**	0.528	0.925	0.849	0.917	0.935	0.921	0.920
	satellite	0.564	**0.632**	0.553	0.695	0.686	0.625	0.602	0.597	0.590
Unimodal	wine	**0.864**	**0.864**	0.624	0.592	0.821	0.602	0.449	0.647	0.48
	wbc	**0.895**	**0.895**	0.772	0.790	0.820	0.778	0.828	0.726	0.699
	speech	**0.522**	**0.522**	0.510	0.506	0.510	0.502	0.529	0.525	0.507
	lympo	**1.0**	**1.0**	0.823	0.893	0.923	0.642	0.948	0.992	0.964
Average AUC		0.761	**0.791**	0.617	0.674	0.703	0.633	0.670	0.676	0.652

5.3. Proprietary enterprise dataset evaluations

In this last experiment, we evaluate our approach using the enterprise dataset described in Section 4.1.3 by applying DIP-ECOD to identify potentially unusual and anomalous GitHub commits. Five features are extracted from the GitHub commits, per Table 3 - the name of the author, the commit data, the number of additions and deletions, plus the number of changed files. Implicitly these are key contextual features that may contain anomalous signals. We analyse all the commits from the most active author in the selected repository, numbering 738 commits over five years.

Feature engineering transforms each commit date's year, month, and day into individual features. Days are converted to integers between 1 and 7, where 1 represents Monday, 2 represents Tuesday, and so on, with the same for month integers running from 1 to 12. The 24-hour time of the commit date is represented as a float, with the number of additions, deletions, and changed files remaining as integers. To illustrate, a commit made at 3:11 am Thursday, August 2017 with 29 additions, and 1 deletion across 1 file, is converted to the feature vector [3.11, 4, 8, 2017, 29, 1, 1].

DIP-ECOD identifies potentially unusual and suspicious commits from the enterprise data. We set the top 1% as anomalies in this case. Therefore there are 7 out of 743 commits identified as anomalous. The results are reviewed by an expert engineer and 3 out of 7 are confirmed as TPs that were never detected by the company. For instance, the commit made on Thursday, August 3^{rd}, 2017, at 3:11 am

Table 3
Example of the five extracted features from the enterprise GitHub repository for each commit

Author	Max Williams
Commit date	2017-03-24 3:11am
Number of addition	29
Number of deletion	1
Number of changed file	1

with 29 additions, 1 deletion, and 1 changed file is identified as an outlier with the highest outlier score of 10.94, which indeed seems unusual given the time is in the middle of the night and does not fit the normal working patterns of the developer, i.e. 9.00 a.m. to 5.00 p.m. This demonstrates the potential of our model to be used to identify unusual events on information security from a range of unlabelled real-world datasets with multimodal distributions.

6. Conclusion and future work

A novel anomaly detection algorithm, Dip-ECOD, has been presented, offering superior outlier detection performance on multimodal datasets without loss of performance in the unimodal case. Results show Dip-ECOD outperforms eight other state-of-art techniques using five datasets of multimodal mixture Gaussian and four unimodal datasets, achieving an AUC of 0.791 on average, compared with 0.761 achieved by the next best, ECOD, and surpassing other 7 techniques. Dip-ECOD has a set of unique properties that makes it interpretable while not requiring distance calculation, making it faster to compute. Dip-ECOD exploits the Dip Test to identify modal and anti-modal intervals, enabling the modeling of a multimodal Gaussian mixture dataset as separate unimodal Gaussians, enabling outliers hidden between modes to be identified.

Like other anomaly detection techniques, Dip-ECOD may have a limitation in that the dip test and ECOD both treat each dimension independently as univariate data, meaning information occasionally may be lost as a consequence, while Dip-ECOD is also generally restricted to continuous data. Future work could consider including the likelihood of each data point belonging to a given mode to better separate multimodal data. Lastly, the dependencies of each dimension and the computational complexity of our model may be worth investigating.

Acknowledgment

This project has received funding from eBay Ltd. and filed a patent application in the US. We are grateful to our colleagues Stuart Millar, Jake Sloan, Marc Patterson, Alok Lal, and Xin Hong for their insightful comments on the early draft of this work.

References

[1] X. Hong, H. Li, P. Miller, J. Zhou, L. Li, D. Crookes, Y. Lu, X. Li, H. Zhou, Component-based feature saliency for clustering, IEEE transactions on knowledge and data engineering 33 (2019) 882–896.

[2] Z. Liu, J. Buford, Anomaly detection of command shell sessions based on distilbert: Unsupervised and supervised approaches, arXiv preprint arXiv:2310.13247 (2023).

[3] R. Harang, E. M. Rudd, Sorel-20m: A large scale benchmark dataset for malicious pe detection, arXiv preprint arXiv:2012.07634 (2020).

[4] A. H. Nadim, I. M. Sayem, A. Mutsuddy, M. S. Chowdhury, Analysis of machine learning techniques for credit card fraud detection, in: 2019 International Conference on Machine Learning and Data Engineering (iCMLDE), IEEE, 2019, pp. 42–47.

[5] W.-K. Wong, A. W. Moore, G. F. Cooper, M. M. Wagner, Bayesian network anomaly pattern detection for disease outbreaks, in: Proceedings of the 20th International Conference on Machine Learning (ICML-03), 2003, pp. 808–815.
[6] Y. Zhao, X. Ding, J. Yang, H. Bai, Suod: toward scalable unsupervised outlier detection, arXiv preprint arXiv:2002.03222 (2020).
[7] Y. Zhao, G. H. Chen, Z. Jia, Tod: Gpu-accelerated outlier detection via tensor operations, arXiv preprint arXiv:2110.14007 (2021).
[8] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, G. Chen, Ecod: Unsupervised outlier detection using empirical cumulative distribution functions, IEEE Transactions on Knowledge and Data Engineering (2022).
[9] M. Pavlidou, G. Zioutas, Kernel density outlier detector, in: Topics in Nonparametric Statistics: Proceedings of the First Conference of the International Society for Nonparametric Statistics, Springer, 2014, pp. 241–250.
[10] X. Yang, L. J. Latecki, D. Pokrajac, Outlier detection with globally optimal exemplar-based gmm, in: Proceedings of the 2009 SIAM international conference on data mining, SIAM, 2009, pp. 145–154.
[11] M. H. Satman, A new algorithm for detecting outliers in linear regression, International Journal of statistics and Probability 2 (2013) 101.
[12] H.-P. Kriegel, P. Kröger, E. Schubert, A. Zimek, Loop: local outlier probabilities, in: Proceedings of the 18th ACM conference on Information and knowledge management, 2009, pp. 1649–1652.
[13] K. Highnam, K. Arulkumaran, Z. Hanif, N. R. Jennings, Beth dataset: Real cybersecurity data for unsupervised anomaly detection research, in: CEUR Workshop Proc, volume 3095, 2021, pp. 1–12.
[14] Y.-J. Kang, Y. Noh, Development of hartigan's dip statistic with bimodality coefficient to assess multimodality of distributions, Mathematical Problems in Engineering 2019 (2019) 1–17.
[15] C. Zhang, B. E. Mapes, B. J. Soden, Bimodality in tropical water vapour, Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography 129 (2003) 2847–2866.
[16] P. Hartigan, Algorithm as 217: Computation of the dip statistic to test for unimodality, Journal of the Royal Statistical Society. Series C (Applied Statistics) 34 (1985) 320–325.
[17] S. Maurus, C. Plant, Skinny-dip: clustering in a sea of noise, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 1055–1064.
[18] Multimodal distribution - wikipedia, https://en.wikipedia.org/wiki/Multimodal_distribution, 2023. Accessed: 2023-09-11.
[19] Empirical distribution function, https://en.wikipedia.org/wiki/Empirical_distribution_function, 2023. Accessed: 2023-09-11.
[20] Outlier detection dataset, http://odds.cs.stonybrook.edu/, 2023. Accessed: 2023-09-11.
[21] N. Saccente, J. Dehlinger, L. Deng, S. Chakraborty, Y. Xiong, Project achilles: A prototype tool for static method-level vulnerability detection of java source code using a recurrent neural network, in: 2019 34th IEEE/ACM International Conference on Automated Software Engineering Workshop (ASEW), IEEE, 2019, pp. 114–121.
[22] T. Hoang, H. J. Kang, D. Lo, J. Lawall, Cc2vec: Distributed representations of code changes, in: Proceedings of the ACM/IEEE 42nd international conference on software engineering, 2020, pp. 518–529.
[23] T. H. M. Le, D. Hin, R. Croft, M. A. Babar, Deepcva: Automated commit-level vulnerability assessment with deep multi-task learning, in: 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), IEEE, 2021, pp. 717–729.
[24] H. Hanif, S. Maffeis, Vulberta: Simplified source code pre-training for vulnerability detection, in: 2022 International joint conference on neural networks (IJCNN), IEEE, 2022, pp. 1–8.
[25] Y. Zhou, A. Sharma, Automated identification of security issues from commit messages and bug reports, in: Proceedings of the 2017 11th joint meeting on foundations of software engineering, 2017, pp. 914–919.
[26] D. Gonzalez, T. Zimmermann, P. Godefroid, M. Schäfer, Anomalicious: Automated detection of anomalous and potentially malicious commits on github, in: 2021 IEEE/ACM 43rd International

Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), IEEE, 2021, pp. 258–267.
[27] G. Bhandari, A. Naseer, L. Moonen, Cvefixes: automated collection of vulnerabilities and their fixes from open-source software, in: Proceedings of the 17th International Conference on Predictive Models and Data Analytics in Software Engineering, 2021, pp. 30–39.
[28] F. Lomio, E. Iannone, A. De Lucia, F. Palomba, V. Lenarduzzi, Just-in-time software vulnerability detection: Are we there yet?, Journal of Systems and Software 188 (2022) 111283.
[29] Software assurance reference dataset, https://samate.nist.gov/SARD/testsuite.php, 2022. Accessed: 2022-04-03.
[30] A. Lazarevic, V. Kumar, Feature bagging for outlier detection, in: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, 2005, pp. 157–166.
[31] D. Pokrajac, A. Lazarevic, 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.
[32] P. Hartigan, Computation of the dip statistic to test for unimodality: Algorithm as 217, Applied Statistics 34 (1985) 320–5.
[33] E. Gómez-de Mariscal, V. Guerrero, A. Sneider, H. Jayatilaka, J. M. Phillip, D. Wirtz, A. Muñoz-Barrutia, Use of the p-values as a size-dependent function to address practical differences when analyzing large datasets, Scientific reports 11 (2021) 20942.
[34] M. Lin, H. Lucas, G. Shmueli, Too big to fail: large samples and the p-value problem. 2 information systems research, 2013.
[35] K. I. Park, M. Park, James, Fundamentals of probability and stochastic processes with applications to communications, Springer, 2018.
[36] Z. He, X. Xu, S. Deng, Discovering cluster-based local outliers, Pattern recognition letters 24 (2003) 1641–1650.
[37] M. Goldstein, A. Dengel, Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm, KI-2012: poster and demo track 1 (2012) 59–63.
[38] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation forest, in: 2008 eighth ieee international conference on data mining, IEEE, 2008, pp. 413–422.
[39] G. Guo, H. Wang, D. Bell, Y. Bi, K. Greer, Knn model-based approach in classification in otm confederated international conferences" on the move to meaningful internet systems", Sicily, Italy (2003) 986–996.
[40] T. Pevnỳ, Loda: Lightweight on-line detector of anomalies, Machine Learning 102 (2016) 275–304.
[41] M. M. Breunig, H.-P. Kriegel, R. T. Ng, J. Sander, Lof: identifying density-based local outliers, in: Proceedings of the 2000 ACM SIGMOD international conference on Management of data, 2000, pp. 93–104.
[42] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, L. Chang, A novel anomaly detection scheme based on principal component classifier, in: Proceedings of the IEEE foundations and new directions of data mining workshop, IEEE Press, 2003, pp. 172–179.