Unsupervised and Active Learning Using Maximin-Based Anomaly Detection

Zahra Ghafoori(✉), James C. Bezdek, Christopher Leckie, and Shanika Karunasekera

School of Computing and Information Systems, The University of Melbourne, Melbourne, VIC 3010, Australia
{ghafooriz,jbezdek,caleckie,karus}@unimelb.edu.au

Abstract. Unsupervised anomaly detection is commonly performed using a distance or density based technique, such as K-Nearest neighbours, Local Outlier Factor or One-class Support Vector Machines. One-class Support Vector Machines reduce the computational cost of testing new data by providing sparse solutions. However, all these techniques have relatively high computational requirements for training. Moreover, identifying anomalies based solely on density or distance is not sufficient when both point (isolated) and cluster anomalies exist in an unlabelled training set. Finally, these unsupervised anomaly detection techniques are not readily adapted for active learning, where the training algorithm should identify examples for which labelling would make a significant impact on the accuracy of the learned model. In this paper, we propose a novel technique called Maximin-based Anomaly Detection that addresses these challenges by selecting a representative subset of data in combination with a kernel-based model construction. We show that the proposed technique (a) provides a statistically significant improvement in the accuracy as well as the computation time required for training and testing compared to several benchmark unsupervised anomaly detection techniques, and (b) effectively uses active learning with a limited budget.

Keywords: Anomaly detection · Unsupervised learning · Active learning

1 Introduction

Anomaly detection is a key component of many monitoring applications, which aim to detect harmful rare events that can be subsequently controlled [8]. It has been used in a wide range of domains from cybersecurity [7,33] to health and safety applications such as fall detection for elderly people [27,35]. A key challenge for anomaly detection is the abundance of unlabelled data [23]. The high cost of labelling hinders the application of supervised anomaly detection techniques, which require labelled examples of anomalies and normal data [8]. Although one-class classification techniques mitigate this issue by building a

U. Brefeld et al. (Eds.): ECML PKDD 2019, LNAI 11906, pp. 90–106, 2020.
https://doi.org/10.1007/978-3-030-46150-8_6

normal profile given only normal data, they are not sufficiently robust to the presence of unknown anomalies in the training set [3,16,32]. Even if the training set only comprises normal data but is noisy, one-class classification can deliver unsatisfactory results [15,16]. Since one-class classification techniques such as *One-Class Support Vector Machine* (OCSVM) [30] and *Support Vector Data Description* (SVDD) [32] provide sparse solutions and are very fast during the testing phase, they have been enhanced to work in an unsupervised manner [3,15,16]. However, depending on the implementation and the characteristics of the dataset, training may require $O(n^2)$ to $O(n^3)$ operations, where n is the cardinality of the training data. Unsupervised anomaly detection techniques such as *K-Nearest Neighbours* (KNN) and *Local Outlier Factor* (LOF), have high computational requirements for processing new observations in a continuously monitored system. For scroring/labelling a new data point, anomaly scores of all or a subset of existing data points should be recomputed in a fairly large reference dataset. Therefore, these methods have limited scope for real-time anomaly detection. In other words, they do not learn an explicit model a priori, which can be later on used for timely evaluation of future observations [1]. iForest [25] is another unsupervised method that attempts to address these challenges by filtering anomalies through isolating trees that are trained on several subsets of the data. This way, iForest is not based on a density or distance measure and lowers the computational cost by sampling. However, the solution provided by iForest is not sparse like OCSVM and SVDD, and to score a test instance it must scan several trees.

In some applications, it might be possible to obtain expert feedback on whether an instance is normal or anomalous. Having that feedback on a small number of critical examples can make a substantial difference to the accuracy of the final model [23]. This process, known as *Active Learning* (AL), has been widely used in classification [34] and rare class discovery [17,20] using supervised or semi-supervised learning. Using AL in unsupervised anomaly detection is an emerging trend [1,19]. Sharma et al. [31] used active learning to train a two-class classifier for identifying operationally significant anomalies from insignificant ones in a flight trajectory dataset. They used the OCSVM algorithm first to identify top-ranked anomalies in an unsupervised manner. Given their scores, the top-ranked anomalies are then presented to an expert to generate a labelled set of operationally significant and insignificant anomalies. This labelled set is used to train a two-class SVM that distinguishes between interesting and unimportant anomalies. Pelleg and Moore [26] proposed a general framework for the same purpose that runs several loops for data modelling (using Gaussian mixtures) and labelling. The algorithm starts with an unlabelled dataset. After each modelling round, labels of 35 instances are asked from an expert. Three strategies are used to choose these instances: choosing instances with low likelihood under the current model, choosing instances that the model is not certain about them, and a combination of the two strategies. Our work is different as we aim to enhance the ability of the underlying unsupervised anomaly detection, which is used by these techniques to find interesting anomalies or discover rare classes. The ability of an anomaly detection technique to efficiently and

effectively use the AL budget is vital for adversarial tasks such as fraud detection, where anomalies can be similar to the normal data due to fraudsters mimicking normal behaviour [1].

In this paper, we present a novel approach called *Maximin-based Anomaly Detection* (MMAD). The contributions of this work are as follows. First, we use random sampling followed by the *Maximin* (MM) sampling technique [22] to select a *representative subset* of the input data, which achieves low constant computational cost for big datasets. Then, we use a *cluster validity index* (CVI) called the Silhouette index [29] on the representative samples that should contribute to defining a kernel-based model, which is subsequently used to score anomalies and normal data in nearly real-time. Second, we incorporate AL into MMAD. We show that with only a few labels, this enhancement of MMAD improves the accuracy of unsupervised anomaly detection. Our numerical experiments on benchmark anomaly detection datasets show that our proposed technique outperforms several state-of-the-art unsupervised anomaly detection techniques in terms of the time-complexity of training a model and testing the future data. Moreover, our technique provides statistically significant improvements in accuracy even when the AL budget is zero.

2 Definitions and Problem Specification

Let the normal data D^* be (an unknown) subset of a given unlabelled training set D, i.e., $D^* \subseteq \mathcal{D} = \{x_1, x_2, ..., x_n\} \subset \mathbb{R}^d$, drawn from an unknown probability distribution P on $\mathbb{R}^d$. The probability distribution P can be approximated by estimating the parameter values θ of P such that:

$$\theta = \arg\min_{\theta \in \Theta} \sum_{x} [(1 - P(x;\theta)) \times I(x \in D^*) + P(x;\theta) \times I(x \notin D^*)], \qquad (1)$$

where Θ represents the set of parameter values for the probability distribution, and $I(.)$ is the indicator function. The cardinality and mean value of a set D are respectively shown by $|D|$ and $\bar{D}$, and for the training set $|D| = n$. In the unsupervised case, estimating θ is done without ground truth or a priori knowledge about the data. The estimated probability distribution P can be used to score data instances such that anomalies get low scores.

We assume that a limited budget B for AL is available, i.e., labels of B instances can be queried from an oracle. *However, the training data might not be available after the query. Thus, the model should be updated after gaining a label from the oracle without having to build the model from scratch.*

3 Methodology

An accurate description of a parametric probability distribution $P(x;\theta)$ for real datasets is usually unavailable. Therefore, instead of solving (1), MMAD estimates a non-parametric probability density of the form:

$$P(x; \{w_{1..n}\}) = \sum_{i=1}^{n} w_i k(x, x_i), \ \textit{subject to}\ w_i \geq 0, \sum_{i=1}^{n} w_i = 1, \qquad (2)$$

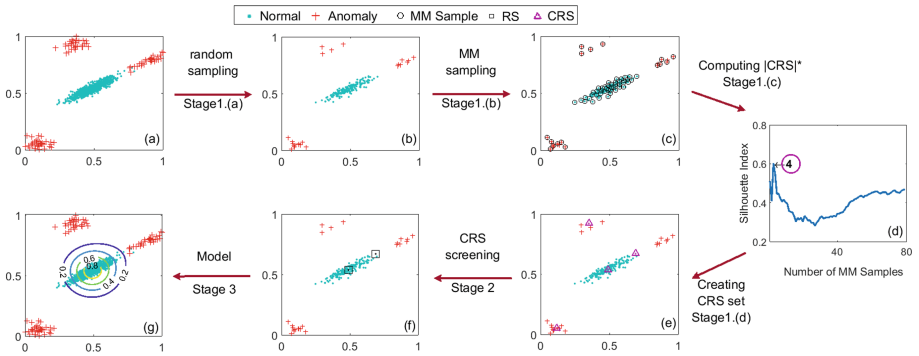

Fig. 1. Demonstration of the three stages of the MMAD Algorithm on the toy example X_{toy}. A candidate set of representative samples CRS is chosen in Stage 1. By removing anomalies from CRS, the final set of representative samples RS is generated in Stage 2. A kernel-based model is learned in Stage 3.

where k represents an arbitrary kernel function and w is the weight vector of the kernel. Our aim is to find a subset $\mathcal{X} \subseteq D$ of $\mathbb{R}^d$ that represents the normal data D^* and divides $\mathbb{R}^d$ into two subsets $\mathcal{X}$ and $\mathbb{R}^d - \mathcal{X}$. Substituting θ with $\{w_{1..n}\}$ in (1), $P(x; \{w_{1..n}\})$ is viewed as a scoring function that assigns high scores to normal data and low scores to anomalies. The computational cost of testing an instance using $P(x; \{w_{1..n}\})$ scales with the number of non-zero w_i $(i = 1..n)$, thus a sparse solution that minimises $\sum_{i=1}^{n} w_i$ is desirable.

To learn $P(x; \{w_{1..n}\})$, MMAD assigns the values of each w_i in three stages as shown in Fig. 1 for a toy example X_{toy}, and explained in Algorithm 1. Figure 1(a) draws X_{toy}. The first stage includes the following steps. A subset of X_{toy} is selected at random as set S in Fig. 1(b). MM sampling [22] is performed on S to generate a uniformly selected subsample from S shown in Fig. 1(c). Choosing a large value for $|S|$ and running this extra MM sampling step is necessary, because relying only on random sampling to secure a representative subset of the input data is not sufficient for two reasons. First, it does not guarantee the selection of representative samples from all clusters in the data, especially if there are very small clusters in the data. Second, random sampling does not provide a mechanism to eliminate anomalies from the model, especially when a high fraction of anomalies exists in the training data. After MM sampling, a set called *Candidate Representative Samples* (CRS) is built using representative MM samples. By evaluating each MM sample as a CRS object, in Fig. 1(d) an optimal cardinality $|CRS|^*$ for this set is defined. In this step the first two MM samples are designated as cluster centers, and a crisp 2-partition U_2 of S, i.e., a partition into two mutually exclusive clusters, is built using the *nearest neighbour prototype rule* (NPR). Then, the value of a CVI called the Silhouette index [29] is computed on U_2. After finding the third MM sample and designating it as a new cluster center, U_3 is built and the value of the Silhouette index is computed on it. This procedure stops when a predefined number of MM samples are assigned as

Algorithm 1. MMAD

Input: training set $D = \{x_1, x_2, ..., x_n\}$, sample size $|S|$
budget B, metric δ
Output: model M: RSs, W, and γ
Stage 1.(a):
S = randSample(D,$|S|$) ▷ random sampling from D
Stage 1.(b):
$[I_{CRS}, \Delta]$ = Maximin(S, $[0.4 * |S|]$, δ) ▷ call Algorithm 2
for $i = 2..|CRS|$ **do**
  compute $Silh_i$ using (3) given $C = \{s_{I^{1..i}_{CRS}}\}$
end for
$|CRS|^* = \underset{i}{argmax}\ Silh_i$ ▷ maximising Silhouette index
$I_{CRS} = I^{1..|CRS|^*}_{CRS}$
for $i = 1..|CRS|$ **do**
  ▷ Substitute CRSs with closet point to their cluster mean
  $c_i = \{s\}$, where $I^i_{CRS} == \arg\underset{j}{min}\ \delta(s, s_{I^j_{CRS}})$
  $I^i_{CRS} = \arg\underset{j}{min}\delta(\overline{c_i}, s_j \in c_i)$
end for
Stage 2:
identify *type* of the dataset
if *type* $== WS$ **then**
  I_{RS} = screen(I_{CRS}) ▷ remove anomalous CRSs
  set γ using (4)
else
  $I_{RS} = I_{CRS}$
  set γ using (5)
end if
if $B > 0$ **then**
  $I_{RS} = BS(I_{RS}, B, \Delta_{RS})$ ▷ call Algorithm 3
end if
Stage 3:
$cn_j = count(c_j), j = 1..|RS|$ ▷ count each cluster's members
$w_j = \frac{cn_j}{\sum cn_j}, j = 1..|RS|$
M = ($s_{I_{RS}}$, γ, W)

cluster centers and used to generate NPR partitions of S. Then, $|CRS|^*$ is chosen as the number of MM samples that maximises the computed Silhouette index values. The first $|CRS|^*$ MM samples form the set CRS (Fig. 1(e)). Removing anomalous instances from CRS is performed in Fig. 1(f) and the new points are called *Representative Samples* (RS). Optionally, if an AL budget is available for securing labels from an oracle, it will be spent. In this figure we assumed that no AL budget is available. Finally in Fig. 1(g), the model is built using RS data. The contour plot of the scores assigned to the data instances in this figure, shows that anomalies have scores close to zero. Next, we explain each of the steps in detail.

Algorithm 2. Maximin

Input: set S, number of objects $|CRS|$, metric δ
Output: Indexes I_{CRS}, pairwise dissimilarities Δ

1: $\Delta = [\delta(s_i, s_j)]$ for all $(i, j) \in 1..|S|$ ▷ Cost $O(|S|^2)$
2: $I^0_{CRS} = \underset{1 \le i \le |S|}{\arg\min}\ \delta(\overline{S}, s_i)$ ▷ Closet point to the mean of S
3: $\Gamma_0 = \Delta_{I^0_{CRS}}$
4: **for** $p = 1..|CRS|$ **do**
5: $\quad I^p_{CRS} = \underset{1 \le p \le |S|}{\arg\max}\ \Gamma_{p-1}$
6: $\quad \Gamma_p = [min(\Delta_{I^p_{CRS}}, \Gamma_{p-1})]$
7: **end for**

3.1 Stage 1: CRS Selection

To reduce the time-complexity for processing big data, MMAD first selects a subset S of the training set by shuffling it and randomly selecting $|S|$ instances. The sample S is used as an input to the MM algorithm [22], which is shown in Algorithm 2. Our implementation of MM sampling starts by finding the index of the closest point to the mean of S (line 2 of Algorithm 2). At each iteration of its main loop (lines $4-7$ of Algorithm 2), MM adds the index of the point that has maximum dissimilarity to the points already selected. We believe that this sampling technique guarantees that the indexes I_{CRS} of the representative object set CRS have at least one member from each of the unknown clusters in the data (see Proposition 1 in [18] for a theoretical guarantee in certain - but not all - cases). Towards this end, we need to define the optimal cardinality $|CRS|^*$. One way to achieve this is to evaluate the appropriateness of the cluster centers $C = \{c_{1..|CRS|}\}$ that are defined on the CRS objects. CVIs can be used for this purpose [4]. However, the presence of anomalies in a dataset can mislead CVIs. We experimented with several popular indexes including Dunn [12], C-Index [21], Calinski-Harabasz [6], Davies-Bouldin [11], and Silhouette [29], and concluded that the Silhouette index provides the best estimate of $|CRS|^*$ such that *most of the point anomalies are isolated as singleton cluster centers in* S.

Given cluster centers $C = \{c_{1..|C|}\}$, each sample $s_i \in S$ is assigned to the closest cluster center c_p with the NPR, i.e., $\|s_i - c_p\| < \|s_i - c_q\|\ \forall q \in 1..|C|, q \neq p$. The set of points accumulated this way for each cluster center are the $|C|$ crisp clusters in this partition of S. Let δ denote any metric on the input space. For each cluster center c_p, the Silhouette index uses the average within-cluster dissimilarity (cohesion) $In^\delta_i = \frac{1}{|c_p|}\sum_{s_i, s_j \in c_p} \delta(s_i, s_j)$ of s_i to the other members of c_p, which indicates how well s_i matches the other data in c_p. The smallest average between-cluster dissimilarity (separation) $Out^\delta_i = min\{\frac{1}{|c_q|}\sum_{\substack{s_i \in c_p, \\ s_j \in c_q, q \neq p}} \delta(s_i, s_j)\}$ of s_i measures how suitable would be the assignment of s_i to its closest neighbouring cluster c_q. The Silhouette index combines the cohesion (In^δ_i) and separation (Out^δ_i) measures to derive a Silhouette value for each s_i as:

$$Silh_{s_i} = \frac{Out_i^{\delta} - In_i^{\delta}}{max(In_i^{\delta}, Out_i^{\delta})}, \tag{3}$$

which is in the range $[-1, 1]$, where a high value indicates that s_i fits well within its own cluster. When $Silh_C = \frac{1}{|S|} \sum_{i=1}^{|S|} Silh_{s_i}$ is close to 1, the corresponding partition is preferred to those with lesser values.

Given that $|CRS|^*$ is not known a priori, we need to examine the value of the Silhouette index by increasing the number of CRS objects selected by MM, and choose $|CRS|^*$ such that the Silhouette index is maximised. Therefore, MMAD first initialises $|CRS| = [0.4 * |S|]$, and chooses $|CRS|$ objects in S by MM sampling (Fig. 1(c)). This value for $|CRS|$ is chosen because it is assumed that majority of the data is normal. Therefore, in the worst case, at least 60% of data is normal. MMAD starts with $Silh_1 = 0$. Then, it picks the first two CRS objects and computes $Silh_2$ considering these objects as cluster centers. From there, MMAD computes $Silh_3$ after adding the third CRS as a new cluster center, and keeps repeating this until the last selected object by MM is added to the cluster centers. Let the Silhouette index be maximised at $Silh_m$. Then, $|CRS|^* = m$, and the size of CRS is reduced such that it comprises the m first CRSs selected by MM sampling (Fig. 1(d)). After this, a further adjustment is performed on CRS as follows (lines 8–11 of Algorithm 1). A crisp m-partition U_m is built by considering each of the m samples in CRS as a cluster center. Each of these m points is then replaced by the closest point to the cluster mean in its partition (Fig. 1(e)). Since the representative samples are chosen using MM sampling, the cluster centers, i.e., CRSs, are expected to be close to the border of their partitions. Therefore, this further adjustment is required to locate them close to the center of the partition.

We identified two types of datasets when using the Silhouette index to estimate $|CRS|^*$: ***Not Well-Separated*** (NWS) and ***Well-Separated*** (WS). If the value of the Silhouette index in $[Silh_{1..|CRS|}]$ has an increasing trend peaking at $Silh_{|CRS|}$, then anomalies and normal samples are not well-separated, otherwise anomalies and normal samples are well-separated. Examples of these types of datasets are shown later in Fig. 3. This property is used in subsequent stages.

3.2 Stage 2: CRS Screening

This stage has two steps. First, an unsupervised heuristic approach is used to detect anomalies in the CRS set, which is generated in Stage 1. Removing potential anomalies from CRS results in creating the RS set, i.e., representative samples. Second, if a budget for AL is available, i.e., a small number of labels are allowed to be asked from an oracle, an optional active learning step is used to improve accuracy.

Given that in unsupervised learning anomalies are a minority, it is expected that anomalous clusters in data have fewer members compared to normal clusters. Based on this intuition, if a dataset is classified as type WS (i.e., normal data and anomalies are well-separated according to the Silhouette index values),

Algorithm 3. Budget Spending (BS)

Input: Representative Samples RS, budget B, dissimilarities Δ_{RS}
Output: I_{RS}
$\{y_i^{RS} = 1\}, i = 1..|RS|$
if $B \geqslant |RS|$ **then**
  $y_i^{RS} = l_i, 1 \leq i \leq |RS|$ ▷ get true label per RS_i
else
  $\Pi(I_{RS})$: same order as selected by MM sampling
  relocate indexes of clusters with one member to end of $\Pi(I_{RS})$
  for $i = 1..|B|$ **do**
    $y_{\Pi(i)}^{RS} = l_{\Pi(i)}, 1 \leq i \leq |RS|$ ▷ get true label of $RS_{\Pi(i)}$
    if $y_{\Pi(i)}^{RS} == -1$ **then** ▷ find indexes of other poor RSs
      $NNs = find(e^{-\gamma \times \Delta(\Pi(i),j)} > 0.5)$
      $y_{NNs}^{RS} = -1$
    end if
  end for
end if
$I_{RS} = \cup_i$ where $y_i^{RS} == 1$

the set CRS is used to generate RS as follows. Samples in CRS are sorted in the reverse order of the number of their cluster members. Let π^{CRS} denote the permuted indexes for the ordered set and n_π^i denotes the corresponding number of members in the ith largest cluster indexed by π_i^{CRS} where $i = 1...|CRS|$. The RS set is initially empty. Then, samples from CRS in the order given by π^{CRS} are added to RS until n_π^i is less than a predefined threshold. In this paper, we evaluate values of n_π^i as follows: when $\frac{n_\pi^{i-1}}{n_\pi^i} \geq 2$, we stop adding the remaining samples from CRS to RS. In contrast, if the dataset is classified as type NWS (i.e., differentiating anomalous and normal objects is difficult because they are not well separated by their distance nor by density), we choose $RS = CRS$ because there is insufficient information available to filter anomalies.

Active Learning Sub-stage. In some application contexts, it may be possible to apply AL to assign a label for one or more selected points as either normal or anomalous by asking an oracle for the labels. Given that the budget for asking an oracle for labels is usually restricted to a small number of points, a key challenge is how to select the points that are likely to be most informative in terms of improving the accuracy if a label is given. If a budget B is available via AL, we can ask for a maximum of B labels from the oracle. If $B \geq |RS|$, all the RSs can be validated by the oracle to remove any anomalies that were left from the screening stage. Otherwise, labels are asked for clusters that have more than one member in the order that they were selected by MM sampling. Anomalous RSs are removed accordingly. This stage of removing anomalous RSs are shown in Algorithm 3 and can take place here or at any time after building the model.

3.3 Stage 3: Model Construction

For the choice of the kernel function, in this paper we use the *Radial Basis kernel Function* (RBF), $k(x, y) = e^{-\gamma\|x-y\|^2}$ to localise the influence of RSs in the final model. The γ parameter is the kernel bandwidth and we set it based on the dataset type identified in Stage 1.

If the dataset is classified as type WS, anomalies and normal data are declared separable. The γ parameter in this case is estimated using the technique proposed by Evangelista et al. [13]. The reason is that this technique chooses a value for γ that maximises the dissimilarity of clusters in the data. We use a candidate set $\Gamma = \{2^{-6}, 2^{-5}, ..., 2^{6}\}$ for γ to extract the kernel matrix $K = [e^{-\gamma \times \Delta}]$, $\Delta = [\delta(s_i, s_j)]\ \forall (i, j) \in 1..|S|$, and select a γ^* such that:

$$\gamma^* = \underset{\gamma}{argmax} \frac{\sigma^2}{\overline{K}_{offDiag} + \varepsilon}, \tag{4}$$

where σ^2 and $\overline{K}_{offDiag}$ are the variance and the mean of the off-diagonal kernel matrix entries, and ε is a small value to avoid division by zero.

If the dataset is classified as type NWS, potential anomalies are close to normal data. Therefore, γ^* is chosen so that the similarities in the feature space created by the RBF kernel are approximately the same as those in the input space:

$$\frac{e^{-\gamma\Delta^2_{max}}}{e^{-\gamma\Delta^2_{min}}} = \frac{\Delta_{min}}{\Delta_{max}} \Longrightarrow \gamma^* = \frac{-ln(\frac{\Delta_{min}}{\Delta_{max}})}{\Delta^2_{max} - \Delta^2_{min}}. \tag{5}$$

where the values of Δ are an output of Algorithm 2. To build the final model, values of $\{w_{1..|RS|}\}$ are assigned for $RS = \{s_{I^{1..|RS|}_{RS}}\}$ and the rest of the data is deleted. For the jth RS, its weight is defined as $w_j = \frac{cn_j}{\sum_{t=1..|RS|} cn_t}$, where cn_j denotes the number of cluster members for the corresponding RS. The final model is $M = (RS, \{w_{1..|RS|}\}, \gamma)$ and a test point is evaluated using the following scoring function:

$$P(x; \{w_{1..|RS|}\}) = \sum_{i=1}^{|RS|} w_i \times k(x, RS_i). \tag{6}$$

If AL is used after constructing the model and deleting the training data, RSs labelled as anomalies by the oracle are deleted from the model, and the weight vector for the rest of the RSs is normalised to add up to one again.

4 Experimental Evaluation

We compare our MMAD method[1] to OCSVM [30] and its unsupervised extensions *Robust OCSVM* (ROCSVM) and ηOCSVM [3], and also to iForest [25]. The default settings appearing in the referenced papers were used for each of

[1] Implementation of MMAD is available at https://github.com/zghafoori/MMAD.

Table 1. Description of the datasets

Dataset	DSA	HAR	MNIST	Cancer	Credit	Shuttle	NKH	Pen	Satimage2	Forest
#Instances	1,117,521	10,299	70,000	675	283,726	36,752	221,902	6,870	5,801	286,048
#Features	45	561	784	10	30	9	3	16	36	10
#Normal	60,000	1,777	7,877	439	283,253	34,108	221,830	6,714	5,732	214,839
#Anomaly	1,057,521	8,522	62,123	236	473	2,644	72	156	69	2,747

these existing techniques. For iForest the default sample size is 256 and it trains 100 trees. For MMAD, the sample size $|S| = min(200, [|D|/2])$. We repeated our experiments by changing $|S|$ in the range $[100, 500]$, and observed that for $|S| > 200$, the result does not change significantly. Euclidean distance was used as the dissimilarity metric δ, but any other metric can be used when circumstances warrant a departure from this choice. To evaluate accuracy, we used the *Receiver Operating Characteristic* (ROC) curve and the corresponding *Area Under the Curve* (AUC). The reported AUC values were averaged over 100 runs. The experiments were conducted on a machine with an Intel Core i7CPU at 3.40 GHz and 16 GB RAM. The MATLAB LIBSVM toolbox [9] was used to implement OCSVM.

4.1 Datasets

We ran our experiments on four benchmark anomaly detection datasets from the *Outlier Detection DataSets* (ODDS) collection [28], namely *Forest Cover* (Forest), *Pendigits* (Pen), Satimage2, and Shuttle. From the *UCI Machine Learning Repository* [24], we selected the *Breast Cancer Wisconsin* (Cancer), MNIST, *Human Activity Recognition* (HAR), and *Daily and Sports Activities* (DSA) datasets. Table 1 shows that DSA, HAR and MNIST contain many more anomalies than normals. In the experiments we use a random subset of anomalies such that majority of data is normal.

For the Cancer dataset, the aim was to detect malignant breast cytology as anomalies. For the MNIST dataset, following Rayana [28], Bandaragoda et al. [5] and Amarbayasgalan et al. [2], digit zero was considered as the normal concept and instances of digit six were considered as anomalies. The HAR dataset included sensor signals of six different activities by a group of 30 volunteers within the age range $[19, 48]$. In this dataset, we used the sitting activity as the normal concept and walking in different ways, standing and laying as anomalies. The DSA dataset comprises sensor signals for 19 activities, each of which is performed by four females and four males within the age range $[20, 30]$. Again, the first activity (sitting) from all the 8 subjects in this dataset is considered as normal and the rest of activities from all subjects are considered as anomalies. This creates clusters of different shapes and cardinalities (Fig. 2(a)) in order to evaluate the effectiveness and robustness of anomaly detection methods. We removed duplicates from normal and anomalous instances, which resulted in generating a dataset with $60,000$ normal and $1,057,521$ anomalous instances for DSA. We also used the *Credit Card Fraud Detection* (Credit) [10] dataset that contains

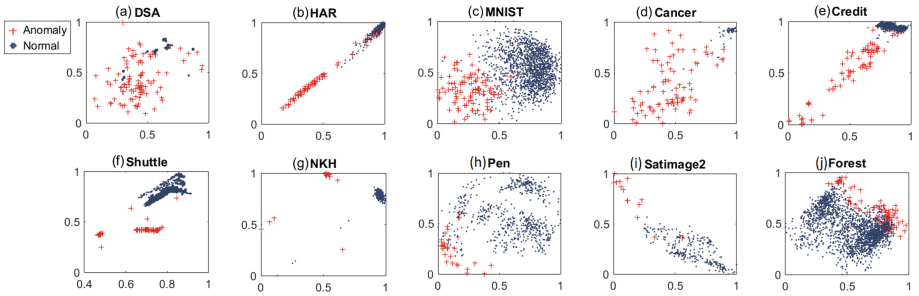

Fig. 2. Visualisation of the datasets

credit card transactions of European cardholders in September 2013. The goal was to detect fraudulent transactions as anomalies. Finally, from the NSL-KDD[2] dataset, we used HTTP data, which we refer to as *NSL-KDD-HTTP* (NKH). Attacks on HTTP services were regarded as anomalies in NKH.

The datasets' descriptions including the number of unique observations and features are summarised in Table 1. Down-sampling on anomalies and normal instances (if required) is used to generate training sets with anomaly fractions chosen from the set $\{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$. For each fraction of anomalies we randomly selected up to $10,000$ instances from each dataset, and repeated the experiment 100 times. All datasets were re-scaled in the range $[0, 1]$ based on maximum and minimum values observed in the training set. Test and training sets were randomly selected with a ratio of 1 to 4.

4.2 Results and Discussion

For each dataset, we took a subset of the data and used a variant of t-SNE called LN-SNE [14] to visualise it in Fig. 2. The labels were used to draw the plots and were not provided to any of the anomaly detection techniques. This figure shows that there are different types of anomalies in the datasets. DSA and MNIST mainly have point anomalies, while shuttle and NKH have clusters of anomalies. Some of the anomalies in HAR and Credit occur inside the normal region. The density and distance similarities for anomalies is similar to the normal samples in Satimage2 and Forest. In the Pen dataset, anomalies have higher density compared to the normal samples and appear very close to them.

To see how the Silhouette index (3) changes by increasing the number of selected representative objects using MM sampling, in Fig. 3, we plotted graphs of the Silhouette values for three datasets, namely DSA, HAR and Pen. The other datasets exhibit a similar trend so we do not present their corresponding plots to save space. The samples size was $|S| = 200$. The fraction of anomalies was increased in the sub-figures from (a) to (g) to test the robustness of the proposed technique to the anomaly ratio in different types of datasets. The fractions of

[2] http://nsl.cs.unb.ca/NSL-KDD/.

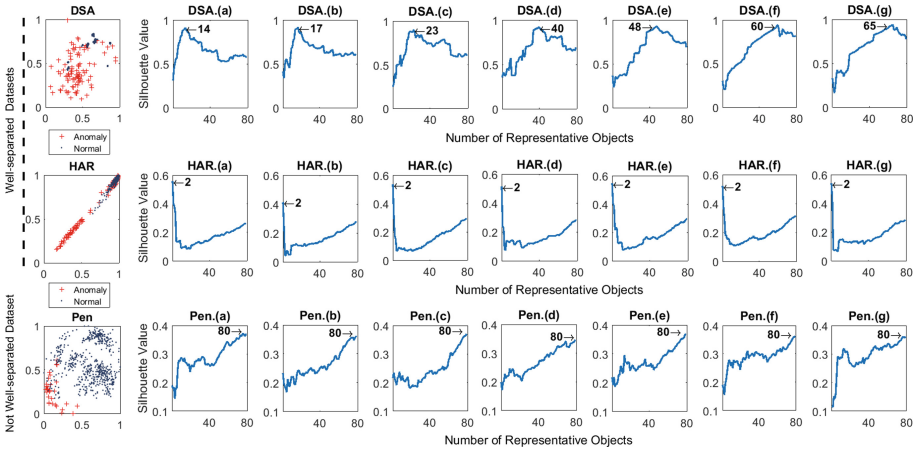

Fig. 3. Silhouette values obtained by increasing the number of objects selected by MM sampling for three representative datasets. For the well-separated datasets DSA and HAR, the the Silhouette value is maximised at an index less than 80, which indicates that separable clusters are identified in the data. However, for the Pen dataset, this index continues to increase to the maximum at 80, which means the data forms a single cluster.

anomalies are (a) 0.01, (b) 0.05, (c) 0.1, (d) 0.15, (e) 0.2, (f) 0.25, (g) 0.3. The index drops for DSA and HAR but reaches its maximum at $|CRS| = 80$ for Pen. Therefore, the DSA and HAR datasets are categorised as type WS (i.e., well-separated) based on the definitions in Sect. 3.1. The difference is that in DSA the normal data comprises several clusters and anomalies are point anomalies; each anomaly is treated as a separate cluster center to maximise the Silhouette index. However, in HAR there are two clusters in the data and one of them is anomalous. The Pen dataset is an example of a type NWS dataset (i.e., not well-separated) because the value of the Silhouette vector is maximised at the end. It can be confirmed via the visualisation given in Fig. 2 that anomalies and normal data in Pen are not well separated via distance or density dissimilarities when no ground truth is available.

Figure 4 depicts the average accuracy over 100 runs of each method for all the datasets for each anomaly fraction. This box-plot shows how the accuracy of the methods changes under different conditions of anomaly type and fraction. Each box-plot shows the variations of AUC for different fractions of anomalies including $\{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$. Increasing the fraction of anomalies can affect the accuracy of the corresponding technique in a negative manner. For MMAD, the results are reported for different AL budget limits $B = \{0, 1, 5, 10, 15, 20\}$. MMAD-B0 means that no budget was available for AL, while MMAD-B$*$ where $* \neq 0$ means that labels for $*$ number of samples could be asked from the AL oracle. MMAD-B0 and iForest have better results than the different versions of OCSVMs. On average, MMAD-B0 works better than iForest

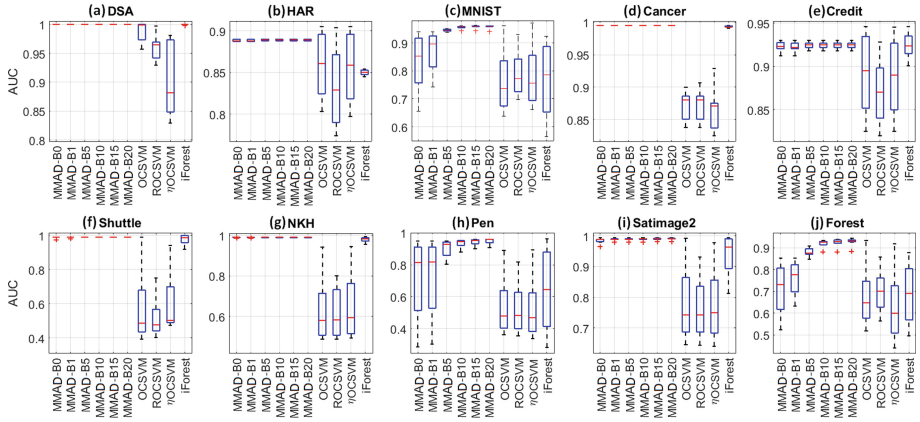

Fig. 4. The average and standard deviation of AUC for MMAD and other techniques.

Table 2. Results of Wilcoxon test for the accuracy of MMAD-B0 vs other methods.

MMAD*-B0 Vs	*-B1	*-B5	*-B10	*-B15	*-B20	OCSVM	ROCSVM	ηOCSVM	iForest
R^+	141	85	84	84	83	2296	2396	2391	1970
R^-	1344	1806	1807	1807	1808	189	89	94	515
p-value	2.23E-07	6.38E-10	6.09E-10	6.09E-10	5.82E-10	7.04E-10	1.47E-11	1.80E-11	2.07E-05

considering all scenarios of the anomaly fraction over all the datasets, especially in HAR, Shuttle and Satimage2. Using an AL budget in the first 7 datasets provides limited advantage because MMAD-B0 effectively screens anomalies in the training phase for these datsets. However, for Pen and Forest, which had the most difficult type of anomalies, access to a limited AL budget $B = 5$ or even $B = 1$ in Forest, increases the accuracy to a great extent.

To assess the statistical significance of the differences in the accuracy shown in Fig. 4 for each method, Table 2 lists the results of a Wilcoxon signed-rank test with a level of significance of $\alpha = 0.05$ on all the pairs of accuracy values per the anomaly fraction. In each comparison, the aim was to investigate to what extent the null hypothesis H_0, which indicates that there is no difference between the first and second methods in terms of their accuracy, can be rejected. For each comparison, the test returns the sum of positive ranks (R^+) and negative ranks (R^-) of the first method, and the p-value. The p-value represents the lowest level of significance of a hypothesis that results in a rejection and if it is less than α, the null hypothesis H_0 can be rejected and the improvement is significant at

Table 3. Training and testing CPU-times for MMAD compared to other methods.

Dataset	Train time (seconds) of					Test time (seconds per sample) of				
	MMAD	OCSVM	ROCSVM	ηOCSVM	iForest	**MMAD**	OCSVM	ROCSVM	ETAOCSVM	iFOREST
DSA	**0.093**	26.450	29.323	32.439	0.19	**3.15E-07**	1.27E-04	1.32E-05	4.85E-06	1.38E-03
HAR	**0.116**	38.047	39.759	38.141	0.41	**9.75E-06**	2.70E-04	1.27E-04	1.08E-04	1.37E-03
MNIST	**0.127**	269.453	281.008	289.487	0.67	**3.43E-05**	1.76E-03	4.76E-04	2.70E-04	1.47E-03
Cancer	**0.060**	0.054	0.060	0.114	0.10	**1.79E-06**	2.15E-05	1.70E-05	1.26E-05	1.37E-03
Credit	**0.097**	46.705	48.847	46.752	0.15	**3.29E-07**	9.26E-05	1.30E-05	8.61E-06	1.47E-03
Shuttle	**0.090**	18.605	19.171	34.718	0.16	**7.88E-07**	7.04E-05	1.71E-05	1.15E-05	1.36E-03
NKH	**0.098**	9.272	7.377	16.662	0.11	**<1E-12**	7.44E-05	1.80E-05	9.42E-06	1.50E-03
Pendig	**0.088**	11.042	11.911	24.427	0.25	**5.66E-06**	6.40E-05	2.37E-05	1.58E-05	1.53E-03
Satimage2	**0.091**	9.588	10.379	10.867	0.20	**1.21E-06**	5.83E-05	1.74E-05	1.08E-05	1.52E-03
Forest	**0.093**	28.479	29.537	39.239	0.20	**2.36E-06**	5.31E-05	9.30E-06	7.48E-06	1.47E-03
Average	**0.095**	45.770	47.737	53.285	0.24	**5.65E-06**	2.59E-04	7.32E-05	4.59E-05	1.44E-03

the level α. Table 2 shows that spending even a limited budget $B = 1$ provides a statistically significant improvement and by increasing the budget, a better result can be achieved. This finding shows the importance of using a technique that can incorporate AL into its training. The small p-values for MMAD-B0 against OCSVM, ROCSVM, ηOCSVM, and iForest, MMAD-B0 indicates that MMAD-B0 provides a statistically significant improvement compared to all of the comparison methods.

Table 3 reports the training and test CPU-times of the different techniques. MMAD and iForest have constant time given that they work on a fixed-size sample of the data. Given the size of the sample $|S|$, the training time-complexity for MMAD is $O(d \times |S|^2)$ and for iForest is $O(t \times |S| \times log|S|)$, where t is the number of trees. For testing, the time-complexity of MMAD is $O(d \times |RSs|)$ per instance, whereas it is $O(t \times log|S|)$ for iForest. The table shows that on average, the training time of MMAD is half of that for iForest, mainly because its default sample size is 200, whereas it is 256 for iForest. However, the testing time per sample for MMAD is more than 250 times less than iForest. For the HAR and Credit datasets, testing times per sample for MMAD are more than $4,000$ times faster than iForest. Both MMAD and iForest are considerably faster than the OCSVM-based methods. The improvements by MMAD in accuracy and its capability of being used with AL, demonstrates that MMAD outperforms iForest on the examined datasets in this paper. To compare the scalability of MMAD and iForest to the different types of OCSVMs, we chose the DSA dataset with over a million instances, and ran all the methods several times by changing the size of the training set from $10,000$ to $50,000$ with a step size equal to $10,000$. The results are shown in Fig. 5, confirming that MMAD and iForest have constant time regardless of the size of the training set, while the training time of the OCSVM variants depends on the size of the training set. This example shows the superiority of algorithms with constant time in processing big data.

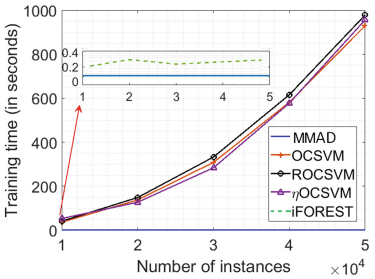

Fig. 5. Scalability of the five methods on DSA.

5 Conclusion

We have proposed a constant time unsupervised anomaly detection technique called MMAD that can be used effectively with active learning to enhance the accuracy of unsupervised anomaly detection when anomalies mimic the behaviour of normal data. MMAD combines a representative subset selection with a cluster validity index and kernel-based model construction in a novel way that results in statistically significant improvement of the accuracy and training time for unsupervised anomaly detection on the examined datasets. In our future work, we will study the use of kernels other than RBF, distance measures other than Euclidean, and CVIs other than the Silhouette index. The information gained via active learning can be used in new ways to further identify the dataset characteristics and improve the accuracy. Finally, this technique can be extended to perform constant time clustering for big datasets that have point or cluster anomalies.

References

1. Abe, N., Zadrozny, B., Langford, J.: Outlier detection by active learning. In: Proceedings ACM SIGKDD International Conference Data Mining Knowledge Discovery, pp. 504–509 (2006)
2. Amarbayasgalan, T., Jargalsaikhan, B., Ryu, K.: Unsupervised novelty detection using deep autoencoders with density based clustering. Appl. Sci. **8**(9), 1468 (2018)
3. Amer, M., Goldstein, M., Abdennadher, S.: Enhancing one-class support vector machines for unsupervised anomaly detection. In: Proceedings ACM SIGKDD Workshop, Outlier Detection Description, pp. 8–15 (2013)
4. Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J.M., Perona, I.: An extensive comparative study of cluster validity indices. Pattern Recognit. **46**(1), 243–256 (2013)
5. Bandaragoda, T.R., Ting, K.M., Albrecht, D., Liu, F.T., Wells, J.R.: Efficient anomaly detection by isolation using nearest neighbour ensemble. In: Proceedings of IEEE International Conference Data Mining Workshop, pp. 698–705 (2014)
6. Caliński, T., Harabasz, J.: A dendrite method for cluster analysis. Commun. Stat. Theory Methods **3**(1), 1–27 (1974)

7. Cao, Q., Yang, X., Yu, J., Palow, C.: Uncovering large groups of active malicious accounts in online social networks. In: Proceedings of ACM SIGSAC Conference Computer Communication Security, pp. 477–488 (2014)
8. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3), 15 (2009)
9. Chang, C.C., Lin, C.J.: Libsvm: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 1–27 (2011)
10. Dal Pozzolo, A., Caelen, O., Johnson, R.A., Bontempi, G.: Calibrating probability with undersampling for unbalanced classification. In: Proceedings IEEE Symposium Series Computer Intelligence, pp. 159–166 (2015)
11. Davies, D.L., Bouldin, D.W.: A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell. **2**, 224–227 (1979)
12. Dunn, J.C.: A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. J. Cybern. **3**(3), 32–57 (1973)
13. Evangelista, P.F., Embrechts, M.J., Szymanski, B.K.: Some properties of the Gaussian kernel for one class learning. In: Proceedings of International Conference Artificial Neural Network, pp. 269–278 (2007)
14. Ghafoori, Z., Erfani, S.M., Bezdek, J.C., Karunasekera, S., Leckie, C.A.: LN-SNE: Log-normal distributed stochastic neighbor embedding for anomaly detection. IEEE Trans. Knowl. Data Eng. **32**(4), 815–820 (2019)
15. Ghafoori, Z., Erfani, S.M., Rajasegarar, S., Bezdek, J.C., Karunasekera, S., Leckie, C.: Efficient unsupervised parameter estimation for one-class support vector machines. IEEE Trans. Neural Netw. Learn. Syst. **29**(10), 5557–5570 (2018)
16. Ghafoori, Z., Rajasegarar, S., Erfani, S.M., Karunasekera, S., Leckie, C.A.: Unsupervised parameter estimation for one-class support vector machines. In: Proceedings Pacific-Asia Conference Knowledge Discovery Data Mining, pp. 183–195 (2016)
17. Görnitz, N., Kloft, M., Brefeld, U.: Active and semi-supervised data domain description. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS (LNAI), vol. 5781, pp. 407–422. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04180-8_44
18. Hathaway, R.J., Bezdek, J.C., Huband, J.M.: Scalable visual assessment of cluster tendency for large data sets. Pattern Recogn. **39**(7), 1315–1324 (2006)
19. He, J., Carbonell, J.G.: Nearest-neighbor-based active learning for rare category detection. In: Proceedings of Advances Neural Information Processing System, pp. 633–640 (2008)
20. Hospedales, T.M., Gong, S., Xiang, T.: Finding rare classes: active learning with generative and discriminative models. IEEE Trans. Knowl. Data Eng. **25**(2), 374–386 (2013)
21. Hubert, L.J., Levin, J.R.: A general statistical framework for assessing categorical clustering in free recall. Psychol. Bull. **83**(6), 1072 (1976)
22. Kennard, R.W., Stone, L.A.: Computer-aided design experiments. Technometrics **11**(1), 137–148 (1969)
23. Krishnakumar, A.: Active learning literature survey. Technical report, University of California, Santa Cruz. 42 (2007)
24. Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml
25. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: Proceedings of IEEE International Conference Data Mining, pp. 413–422 (2008)

26. Pelleg, D., Moore, A.W.: Active learning for anomaly and rare-category detection. In: Proceedings of Advances Neural Information Processing System, pp. 1073–1080 (2005)
27. Quellec, G., Lamard, M., Cozic, M., Coatrieux, G., Cazuguel, G.: Multiple-instance learning for anomaly detection in digital mammography. IEEE Trans. Med. Imag. **35**(7), 1604–1614 (2016)
28. Rayana, S.: ODDS library. http://odds.cs.stonybrook.edu
29. Rousseeuw, P.J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Computat. Appl. Math. **20**, 53–65 (1987)
30. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
31. Sharma, M., Das, K., Bilgic, M., Matthews, B., Nielsen, D., Oza, N.: Active learning with rationales for identifying operationally significant anomalies in aviation. In: Berendt, B., et al. (eds.) ECML PKDD 2016. LNCS (LNAI), vol. 9853, pp. 209–225. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46131-1_25
32. Tax, D.M., Duin, R.P.: Support vector data description. Mach. Learn. **54**(1), 45–66 (2004)
33. Thottan, M., Ji, C.: Anomaly detection in ip networks. IEEE Trans. Signal Process. **51**(8), 2191–2204 (2003)
34. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J. Mach. Learn. Res. **2**(Nov), 45–66 (2001)
35. Wang, Y., Wu, K., Ni, L.M.: Wifall: Device-free fall detection by wireless networks. IEEE Trans. Mobile Comput. **16**(2), 581–594 (2017)