Detection of Thin Boundaries between Different Types of Anomalies in Outlier Detection Using Enhanced Neural Networks

Rasoul Kiani\textsuperscript{a}, Amin Keshavarzi \textsuperscript{a*}, and Mahdi Bohlouli\textsuperscript{b,c,d}

\textsuperscript{a}Department of Computer Engineering, Marvdasht Branch, Islamic Azad University, Marvdasht, Iran; \textsuperscript{b}Department of Computer Science and Information Technology, Institute for Advanced Studies in Basic Sciences, Zanjan, Iran; \textsuperscript{c}Research Center for Basic Sciences and Modern Technologies, Institute for Advanced Studies in Basic Sciences, Zanjan, Iran; \textsuperscript{d}Research and Innovation Department, Pentax GmbH, Bonn, Germany

ABSTRACT

Outlier detection has received special attention in various fields, mainly for those dealing with machine learning and artificial intelligence. As strong outliers, anomalies are divided into point, contextual and collective outliers. The most important challenges in outlier detection include the thin boundary between the remote points and natural area, the tendency of new data and noise to mimic the real data, unlabeled datasets and different definitions for outliers in different applications. Considering the stated challenges, we defined new types of anomalies called Collective Normal Anomaly and Collective Point Anomaly in order to improve a much better detection of the thin boundary between different types of anomalies. Basic domain-independent methods are introduced to detect these defined anomalies in both unsupervised and supervised datasets. The Multi-Layer Perceptron Neural Network is enhanced using the Genetic Algorithm to detect new defined anomalies with a higher precision so as to ensure a test error less than that be calculated for the conventional Multi-Layer Perceptron Neural Network. Experimental results on benchmark datasets indicated reduced error of anomaly detection process in comparison to baselines.

Introduction

Data extraction (Keshavarzi et al. 2008) and preprocessing operations lead to a refined explorable dataset in different machine learning applications such as cloud computing (Keshavarzi, Haghighat, and Bohlouli 2019, 2017), big data (Bohlouli et al. 2013), and sensor networks (Jafarizadeh, Keshavarzi, and Derikvand 2017). Preprocessing aims at identification and removing outliers to improve the quality of cleansing process (Agarwal 2013; Kiani, Mahdavi, and Keshavarzi 2015). Outliers show a higher deviation and are not in line with the behavior of general dataset, which could cause unexpected results in analytics.
Outliers probably created due to measurement error, the inherent variability of data or faulty sensors (Aggarwal 2015; Chandarana and Dhamecha 2015; Chandola, Banerjee, and Kumar 2009). Noises are weak outliers but anomalies are strong outliers. The boundary between the noises and anomalies is not clear but can be determined through different analytical methods (Aggarwal 2015). Anomalies are divided into three categories of point, contextual and collective anomalies (Aggarwal 2015; Chandola, Banerjee, and Kumar 2009; Malik, Sadawarti, and Kalra 2014; Song et al. 2007).

Point Anomalies (PA) are located at a considerable distance from normal data and diverged from the usual pattern of data. According to conditions, contextual anomalies can be (or not to be) outlier relative to normal data. Collective anomalies are a set of related outliers relative to normal data. Such anomalies may be free of deviations alone (Chandola, Banerjee, and Kumar 2009; Gupta et al. 2014; Malik, Sadawarti, and Kalra 2014). In fact, point and collective anomalies are two subsets of contextual anomalies (Aggarwal 2015; Chandola, Banerjee, and Kumar 2009). Based on the use of labeled data, outlier detection approaches are divided into supervised, semi-supervised and unsupervised methods (Bolton and Hand 2001; Fujimaki, Yairi, and Machida 2005; Steinwart, Hush, and Scovel 2005; Theiler and Cai 2003; Vijayarani and Nithya 2011). Also, outlier detection methods are divided into distribution-based, clustering-based, distance-based and density-based methods (Challagalla et al. 2010; Hodge and Austin 2004; Kou, Lu, and Dos Santos 2007; Zhang 2008). The key components of anomaly detection methods are research area, anomaly detection technique, problem characteristic and application domain (Chandola, Banerjee and Kumar 2009).

The most important challenges in outlier detection include the thin boundary between the remote points and natural area, the tendency of new data and noise to mimic the real data, unlabeled datasets and different definitions for outliers in different application areas (Chandola, Banerjee and Kumar 2009; Aggarwal 2015). In this paper, the thin boundary between normal data and various types of anomalies is examined. Furthermore, other types of anomalies called Collective Normal Anomaly (CNA) and Collective Point Anomaly (CPA) are investigated.

- **CNA**: There is a thin boundary between Normal Data (ND) and CNA. Due to the characteristics of ND, it is assumed that CNA can be clustered. CNA is a cluster that its standard deviation density is greater than or equal to the threshold for standard deviation of all clusters.
- **CPA**: CPA is a subset of Point Anomaly (PA) and there is a thin boundary between PA and CPA. Due to the characteristics of PA, it is assumed that CPA cannot be clustered.

Figure 1 shows ND, CNA, PA and CPA in a schematic plot and the thin boundary between various types of anomalies is visible.
For this purpose, unsupervised and supervised datasets are first studied. Using the proposed framework in this paper, the supervised dataset is divided into subsets based on the number of classes. The Multi-Layer Perceptron Neural Network (MLP-NN) is also improved using the Genetic Algorithm (GA) to detect the thin boundary between different types of anomalies. Because of the fact that neural network learning which is based on neurons’ weight and detection accuracy is variable in each epoch, using GA seems possible to solve this problem which is improved both better detection of new defined anomalies and reducing the test error.

The rest of this paper is organized as follows. The “Literature Review” section reviews related work in outlier detection. The proposed method is discussed in detail in the “The Proposed Method” section. The results are analyzed in the “Results and Discussion” section. Finally, conclusions are presented.

**Literature Review**

As a supervised or semi-supervised method, the neural networks have been used to detect outliers and anomalies in various fields such as host-based intrusion detection (Ghosh, Wanken, and Charron 1998), network intrusion detection (Ramadas, Ostermann, and Tjaden 2003; Smith et al. 2002; Zhang et al. 2001), credit card fraud detection (Aleskerov, Freisleben, and Rao 1997; Zhang et al. 2001), mobile fraud detection (Barson et al. 1996; Taniguchi et al. 1998), medical and public health domain (Campbell and Bennett 2001), fault detection in mechanical units (Diaz and Hollmen 2002; Li, Pont, and Jones 2002), structural damage detection (Sohn, Worden, and...
Farrar 2001), image processing (Augusteijn and Folkert 2002; Singh and Markou 2004) and anomalous topic detection in text data (Manevitz and Yousef 2001). Figure 2 shows the variation of efficiency with dimensions for all methods. Moreover, Figure 3 shows the variation of scalability with the dimensions respectively for all methods (Malik, Sadawarti, and Kalra 2014). As can be seen in Figures 2 and 3, both efficiency and scalability are considered based on dimensionality respectively for all the methods. It seems likely the methods based on NN and clustering benefit greatly

![Figure 2. Efficiency of various outlier detection methods in scale-up (Malik, Sadawarti, and Kalra 2014).](image1)

![Figure 3. Scalability of various outlier detection methods in scale-up (Malik, Sadawarti, and Kalra 2014).](image2)
from the best efficiency and scalability. For example, when dimensionality
is 80, the scalability of NN, clustering and density methods are approxi-
mately equal therefore NN seems a much better choice owing to the fact
that its efficiency is better.

A 3-step process has been proposed to detect false alarms and outliers
(Hachmi, Boujenfa, and Limam 2015). In the first step, preliminary alerts are
clustered to create a set of meta-alerts. In the second step, outliers are removed
from the meta-alerts. In the third step, a binary classification algorithm is used
to classify meta-alerts into attacks and false alarms. An extended statistical
unsupervised method has been used to detect outliers in object-relational data
(Riahi and Schulte 2015). For this purpose, a metric was introduced based on
the likelihood ratio of vectors of population association and individual asso-
ciation. To detect outliers in large matrices, a two-stage adaptive approach has
been suggested that its performance is guaranteed using an inference met (Li
and Haupt 2015). Song et al. (2007) proposed a general-purpose method called
conditional anomaly detection. They used three different learning algorithms
for their proposed model. A distributed outlier detector with a reasonable
speed and efficiency has been proposed to detect so-called global outliers in
a distributed database (Zhang, Cao, and Zhu 2012). Wang and Davidson
(2009) detected contextual outliers by using the random walks graph. The
most important feature of this approach is to consider scores for outliers.
A data-driven approach has been proposed to detect anomalies in the patient
management actions (Hauskrecht et al. 2010). This method is based on past
patient records in the electronic patient health record system. A semi-
supervised framework based on a fixed-background mixture has been pro-
posed to detect anomalies (Vatanen et al. 2012). This framework is robust
enough to detect patterns of anomaly model. To detect collective anomalies
and DoS attacks in network traffic analysis, a framework has been suggested
based on the X-means clustering algorithm (Ahmed and Mahmood 2014).
Noble and Cook (2003) used anomalous infrastructure detection and anom-
alous subgraph detection to provide a graph-based approach for anomaly
detection. Yang and Liu (2011) detected anomalies in collective moving
patterns using the hidden Markov model. Abnormal detection research pre-
processing the data and sets the normal sample set has been presented. This
method based on outlier mining calculated the outlier score of each sample in
the normal sample set (Zhang et al. 2018). Taylor et al. proposed the outlier
detection method using the super efficiency to remove the outlier little effect in
estimation since the neighbor outlier serves as a proxy benchmark. In other
words, they developed an alternative method based on the stochastic DEA
model of Banker (Boyd, Docken, and Ruggiero 2016). Ko et al. (2017) sug-
gested the model based on data integration and machine learning-based
anomaly detection so as to the overcome the conventional methods for esti-
mating the level of quality. Also, the method for segmentation and indexing
multi-dimensional time-series data is introduced. Maheshwari and Singh (2016) proposed an algorithm to output clusters and outliers in a divide and conquer manner. The method following outliers in each cluster identified core objects outliers. Guo et al. (2018) proposed a new distance-based method on which depends the data structure to detects such points. In the proposed method, first, a global binary tree is used and then the local distance score of point is calculated for evaluating to what degree the observations in an outlier. Zhao and Hryniewicki (2018) proposed an algorithm called XGBOD which was a new semi-supervised method. XCBOD described and demonstrated for enhanced detection of outliers from normal data. This framework combined the strengths of both supervised and unsupervised methods by a hybrid approach. Kutsuna and Yamamoto (2017) suggested a novel method for outlier detection using a binary decision diagram which is used as a new measure for detecting outliers. Lin et al. (2018) proposed a method that has employed a spatial-feature-temporal tensor model analyzed latent mobility patterns through unsupervised learning and LOF algorithm is used to localize anomaly in a given time interval. Macha and Akoglu (2018) proposed a new approach called x-PACS which are used reverse engineering to detect anomalies based on both the groups and characterizing subcase and features rules.

The Proposed Method

Assumptions

It is assumed that there are two types of datasets: (1) datasets containing labeled data and (2) those containing unlabeled data. The main assumption of our proposed method is that data has been previously labeled using common techniques such as clustering algorithm, decision tree, and hidden Markov Model. Equation (1) shows the relationship between different types of anomalies and normal data (Chandola, Banerjee, and Kumar 2009):

- PA: If an individual data instance can be considered as anomalous with respect to the rest of data, then the instance is termed as a PA.
- Collective Anomalies (CA): If a collection of related data instances is anomalous with respect to the entire data set, it is termed as a CA. The individual data instances in a collective anomaly may not be anomalies by themselves, but their occurrence together as a collection is anomalous.
- ND: ND instances occur in dense neighborhoods, while anomalies occur far from their closest neighbors.

The relationship between the PA and CPA is shown in Equation (2). Based on Equation (2) CPA is a subset of PA and there is a thin boundary between
PA and CPA. Equation (3) defines CPA that the neighborhood radius of CPA is less than average neighborhood radius of PA.

$$PA \cup CA \cup ND = \text{Dataset}$$  \hspace{1cm} (1)

$$CPA \subset PA$$  \hspace{1cm} (2)

$$CPA = \{ P_i \in CPA/ P_i \in PA \text{ and } \text{Out}_RadP_i < \text{Out}_RadPA \}$$  \hspace{1cm} (3)

where \(\text{Out}_RadP_i\) is the neighborhood radius of \(P_i\) as well as its average distance to PA and \(\text{Out}_RadPA\) is the neighborhood radius of PA. Equations (4)-(6) show the calculation of neighborhood radius.

$$\text{Out}_RadP_i = \frac{\sum_{i=1}^{k} MDist_{O_i}}{k}$$  \hspace{1cm} (4)

$$MDist_{O_i} = \left\{ \frac{\sum_{j=1}^{k} \text{Dist}(O_i, O_j)}{k-1} \right\}_{i \neq j, i = \{1, \ldots, k\}}$$  \hspace{1cm} (5)

$$\text{Dist}(O_i, O_j) = \left\{ \sqrt{(x_{O_i} - x_{O_j})^2 + (y_{O_i} - y_{O_j})^2} \right\}_{i, j = \{1, \ldots, k\}}$$  \hspace{1cm} (6)

where \(\text{Out}_RadP_i\) is the outlier radius of CPA, \(MDist\) is the mean distance table from point anomalies, \(O_i\) and \(O_j\) are PA, and \(k\) is the number of point anomalies.

Equation (7) shows the relationship between the ND and CNA. Based on Equation (7) CNA is a subset of ND and there is a thin boundary between ND and CNA. Due to the characteristics of ND means that their neighborhood radius is less than the mean distances from points, is assumed that CNA can be clustered in order to use in supervised data. The definition of CNA is given in Equation (8). CNA is a cluster that its standard deviation density is greater than or equal to the threshold for standard deviation of all clusters.

$$\text{CNA} \subset \text{ND}$$  \hspace{1cm} (7)

$$\text{CNA} = \{ C_i \in C \mid C_i \in \text{ND and } \sigma_{\text{Den}C_i} \geq \text{Th} \sigma_{\text{Den}C} \}$$  \hspace{1cm} (8)

where \(C_i\) is one of the detected clusters, \(C\) is the set of all clusters, \(\sigma_{\text{Den}C_i}\) is the standard deviation of cluster density \(C_i\), and \(\text{Th} \sigma_{\text{Den}C}\) is the threshold for standard deviation of all clusters.

In this paper, the research area is data mining, and the application range is independent of domain and the problem characteristic is the type of anomaly. The proposed method to detect different types of anomalies is described and a new framework is proposed for labeling supervised datasets in the proposed framework section. After that, MLP-NN is enhanced using the GA to increase the precision of anomaly detection.
The Proposed Framework

One of the important points considered in this paper is adaptability of the proposed algorithm with both supervised and unsupervised datasets. As previously mentioned, anomalies have been labeled using common techniques and are ready to be used in the neural network. In the first step, the supervised dataset is divided into sub-datasets to detect local anomalies.

**Step 1:** Various types of anomalies should be investigated in all classes in the supervised dataset. Thus, among \( k \) features in the reference dataset, \( l \) features with a higher separation capability should be analytically selected as the main features. In other words, based on three criteria including the type of dataset, functional domain and its features, the most distinguishing features should be selected. Although \( l \) features have been selected to suit all classes means that supervised dataset, they may be not the best if they are evaluated locally (in each sub-dataset). Thus, aggregation technique is used here. That is to say, aggregation is a type of data smoothing. Therefore, two aggregation techniques are used to reduce the number of features from \( k \) to \( l \). As a first technique, normalization is used to improve the accuracy of data mining algorithms. The second technique is calculating weight for all samples. The range of numbers has a direct effect on the weight obtained for each feature in the weighting process. If normalization techniques are not used, weights will be unbalanced leading to unsmoothed features. Accordingly, normalization technique is used to solve this problem to put all the numbers for all features in a constant range.

Equation (9) shows the weighting formula and Equation (10) shows the formula for constructing new features.

\[
W_i = \frac{X_i^1 + X_i^2 + \ldots + X_i^j}{j} \quad i = 1 \ldots n, \quad j = 1 \ldots m \text{ and } m = k - l \quad (9)
\]

where \( W_i \) is the weight of sample \( i \), \( i \) is the sample number, \( n \) is the total number of samples in the dataset, \( j \) is the feature number that its weight is calculated, \( k \) is the total number of features in the dataset. It should be noted that \( m = k-l \) since when \( l \) main features are considered as one of the main features globally. They do not have the best features locally in each sub-datasets, and \( X \) is the normalized sample.

\[
Att\_New^i_j = W_i + Att^i_j \quad i = 1 \ldots n, \quad j = 1 \ldots l \quad (10)
\]

where \( Att\_New \) is a new feature for each sample in the dataset, \( i \) is the number of samples and \( j \) is the number of features.

**Step 2:** Dividing datasets into sub-datasets based on the number of classes in supervised dataset, and apply clustering algorithm for unsupervised dataset.

**Step 3:** Labeling various types of anomalies.
Step 4: Normalization: since data in sub-datasets are affected by the weights used in Equation (9), normalization is applied at this stage.

Step 5: Integration of sub-datasets.

Figure 4 shows the proposed framework for labeling the supervised dataset.

**MLP Neural Network Optimization Using Genetic Algorithm**

MLP-NN and backpropagation learning algorithms are used to detect anomalies in supervised and unsupervised datasets. As a drawback, this type of neural network gives different precision, recall and detection error values each time it is applied to detect various types of anomalies with same inputs. By enhancing this type of neural network, its detection performance will be improved compared to MLP-NN with the conventional backpropagation learning algorithm.

![Diagram](image)

**Figure 4.** The proposed framework for labeling the supervised dataset.
Two MLP-NNs have been used in the overall scheme, one separately and the other as the fitness function in GA, but all the initial parameters are the same. Table 1 lists the MLP-NN parameters.

Different parameters used in the NN are selected according to the application range and the optimal adaptability.

(1) The goal is to improve the results of the neural network, so both networks have the same initial input weights and this is why this is done using the genetic algorithm.

(2) Selecting an appropriate matrix for weights and biases leads to rapid convergence of the neural network but improper selection leads to local optima. This is why they are considered in the range of 0 to 1.

(3) Since the data are shown in a two-dimensional space, two input neurons are considered, and the number of neurons increased.

(4) Since the data are divided into four groups, four neurons are considered in the output layer so that the output of each neuron can be 0 or 1. To show anomalies and ND, one of the output neurons is 1 and the other neurons are 0.

(5) Although an increase in the number of hidden layers increases the learning ability, calculations in the training and testing steps will increase. Most problems (models) that cannot be separated linearly (using a line in a two-dimensional space), can be solved by two to three layers in the MLP network (one output layer, one or two hidden layers). This is why the number of network layers is 2.

(6) In the MLP-NN, neurons activation function in the hidden layers must be of Sigmoid type. Otherwise, the MLP-NN becomes a single-layer perceptron neural network and cannot detect

---

Table 1. MLP neural network parameters.

| Row | Parameter                   | Value                  |
|-----|-----------------------------|------------------------|
| 1   | Input Weight and Bias       | Generated by GA        |
| 2   | Weight and Bias Range       | [0–1]                  |
| 3   | Input Neuron                | 2                      |
| 4   | Output Neuron               | 4                      |
| 5   | Layer Number                | 2 (1 Hidden Layer and 1 output Layer) |
| 6   | Hidden Layer Function       | Tansig                 |
| 7   | Hidden Layer Size           | 10                     |
| 8   | Output Layer Function       | Tansig                 |
| 9   | Initial Function            | Initlay                |
| 10  | Perform Function            | Mse                    |
| 11  | Train Function              | Trainscg               |
| 12  | Learning Algorithm          | Back Propagation       |
non-linear inseparable problems. There are two types of Sigmoid function including:

(a) Tansig: Hyperbolic tangent sigmoid transfer function. Tansig is a neural transfer function. Transfer functions calculate a layer’s output from its net input. This is mathematically equivalent to \( \tanh(N) \). It differs in that it runs faster than the MATLAB implementation of \( \tanh \), but the results can have very small numerical differences. This function is a good tradeoff for neural networks, where speed is important and the exact shape of the transfer function is not.

(b) Logsig: Log-sigmoid transfer function. Logsig is a transfer function. Therefore, Tansig is used.

(7) The small number of neurons in the hidden layer causes inadaptability while the large number of neurons in the hidden layer leads to overfitting. Therefore, 10 neurons were considered in the hidden layer by trial and error.

(8) The use of Sigmoid function in the output layer limits the network output to a small range. As previously stated, Tansig function is more appropriate for this purpose.

(9) Since the weights and biases are injected to the neural networks, Initlay (Layer-by-layer network initialization) function is used. Initlay is a network initialization function that initializes each layer \( i \) according to its own initialization function net and returns the network with each layer updated. Initlay does not have any initialization parameters.

(10) MSE (Mean squared normalized error performance function) is used as the performance function. MSE is a network performance function which measures the network’s performance according to the mean of squared errors and returns the mean squared error. Note that MSE can be called with only one argument because the other arguments are ignored. MSE supports those ignored arguments to conform to the standard performance function argument list.

(11) To select training algorithm for the MLP-NN, different parameters such as problem complexity, the number of data in the dataset, the number of weights and biases, the error and so on should be considered. According to the parameters listed above, Trainscg (Scaled conjugate gradient back propagation) function is used which is a network training function that updates weight and bias values according to the scaled conjugate gradient method. Trainscg can train any network as long as its weight, net input,
and transfer functions have derivative functions. Back propagation is used to calculate derivatives of performance with respect to the weight and bias variables. One of the main reasons for selecting trainscg function is to improve network generalization as well as this one of the ways to improve the network generalization is early stopping where the dataset is divided into training, evaluation and testing data and the trainscg function shows a better performance with early stopping.

This section outlines the GA steps to enhance the results of MLP-NN.

Step 1: the initial population is generated by the GA. The number of genes in individuals equals the number of weights and biases required for the MLP-NN. The purpose is to apply the same input weights to the MLP-NN outside the GA and the evaluation function (the MLP-NN inside the GA). The extracted weights are generated as an initial population in the form of a matrix where the number of rows equals the population in each generation and the number of columns is equal to the total number of weights and biases. In the future generations, GA will produce the next generation.

Step 2: Applying the crossover operator according to Equation (11).

\[
\text{Infant}_i = \begin{cases} 
    \{ G_i^j, \ldots, G_i^k \} = \{ X_i^j, \ldots, X_i^k \} & \text{where } i \in \{1, 2\} \\
    \{ G_i^k, \ldots, G_i^n \} = (X_i^j \times a) + (X_i^{j+1} \times (1 - a)) & \text{where } j \in \{k, \ldots, n\} \\
    & k \in \{2, \ldots, n - 1\} \\
    X_i^{j+1} = X_i^{j+1} & i = 1, X_i^{j+1} = X_i^{j-1} & i = 2
\end{cases}
\]  

where \( \text{Infant}_i \) is the \( i \)th infant, \( G_i^j \) is \( j \)th gene of the \( i \)th infant, \( k \) is a random number, \( X_i^j \) is \( j \)th gene of the \( i \)th parent, \( n \) the total number of genes equal to the number of weights and biases of the MLP-NN.

Step 3: The mutation operator (Equation (12)) is used to search in a larger space to avoid local optima.

\[
\text{Infant}_i^{\text{New}} = \begin{cases} 
    \text{Infant}_i^{\text{Old}} & \text{where } \text{Prob}_1 < \text{Mut. Rate} \\
    \{ X_1^{\text{Old}}, \ldots, X_n^{\text{New}}, \ldots, X_n^{\text{Old}} \} & \text{where } \text{Prob}_1 \geq \text{Mut. Rate}, \text{Prob}_2 = j \in \{1, \ldots, n\} \\
    X_j^{\text{New}} = X_j^{\text{Old}} + (k \times \text{Prob}_3) & \text{where } \text{Prob}_1 \geq \text{Mut. Rate}, \text{Prob}_2 = j \in \{1, \ldots, n\} \\
    0 < \text{Prob}_3 < 1, \{ k = -1 | \text{Prob}_4 < 0.5 \} \\
    & k = +1 | \text{Prob}_4 \geq 0.5
\end{cases}
\]
where \( Prob_1, Prob_2, Prob_3 \) are probabilistic values, \( Infant_i^{New} \) is the \( i^{th} \) mutated infant, \( Infant_i^{Old} \) the current un-mutated infant, \( Prob_i \) mutation probability of the \( i^{th} \) infant, \( Mut.Rate \) desired mutation rate, \( X_j^{Old} \) \( j^{th} \) un-mutated gene of the \( i^{th} \) infant, \( X_j^{New} \) \( j^{th} \) mutated gene of the \( i^{th} \) infant, \( Prob_2 \) is the number of gene to mutate the \( i^{th} \) infant, \( Prob_3 \) the change in the mutated gene and \( Prob_4 \) is increased or decreased change in the mutated gene.

**Step 4:** The evaluation function is used to make decisions for the next generation. For this purpose, the MLP-NN with the same parameters of the MLP-NN outside the genetic algorithm is used. The fitness function is defined as follows:

\[
Fitness = Final \ Test \ Error = Average \ of \ Target \ Test \ Error \quad (13)
\]

**Step 5:** The selection function is applied. Assuming a selection rate of 70%, 70% of the best errors are selected and the rest are selected randomly from the normal data. A ranking-based selection procedure is used, because a member with low adaptability may have appropriate and effective genes.

**Step 6:** One of the following conditions will end the algorithm.

- The implementation cycles of the GA.
- Reaching the minimal error shown by Goal.

**Step 7:** At the end, the test errors obtained from the enhanced and conventional MLP-NN are compared.

Table 2 lists the initial parameters of the GA.

The reasons for selecting the initial parameters of the GA are discussed.

1. The cycle is selected by trial and error.
2. Initial population is selected by trial and error.
3. The use of crossover operator generates members with adaptability higher than the average and this avoids dispersion. For this purpose, a single point is used. An increase in the number of points in the crossover operator will result in higher variation in the search space

| Row | Parameter         | Value |
|-----|-------------------|-------|
| 1   | Cycle             | 20    |
| 2   | Population Size   | 15    |
| 3   | Crossover \( \alpha \) | 0.3   |
| 4   | Mutation Rate     | 0.1   |
| 5   | Selection Rate    | 0.7   |
| 6   | Goal              | 0     |
| 7   | Fitness Function  | Test Error |

Table 2. The initial parameters of the genetic algorithm.
and a lower reliability (the answers will considerably change in different generations).

(4) Mutation leads to search in the space that has not been previously investigated. Mutation rate should not be high, because the GA becomes a completely random search algorithm and thus convergence is delayed.

(5) One of the problems with small population in the GA is local optima. To overcome this problem, Rank Scaling selection function is used. The default fitness scaling option, Rank, scales the raw scores based on the rank of each individual instead of its score. The rank of an individual is its position in the sorted scores: the rank of the fit individual is 1, the next most fit is 2, and so on. The rank scaling function assigns scaled values so that. Rank fitness scaling removes the effect of the spread of the raw scores.

(6) The target error of the fitting function is 0. When an error of 0 is achieved, the algorithm is stopped.

(7) The fitness function in the GA is defined using MLP-NN to minimize the test error. Figure 5 shows the proposed scheme to enhance MLP-NN using the GA.

Figure 5. The proposed scheme to enhance MLP-NN using the GA.
MLP-NN using the GA. It should be noted that the MLP-NN is enhanced using the GA to detect new defined anomalies with a higher precision so as to ensure a test error less than that calculated for the conventional MLP-NN.

**Results and Discussion**

The proposed techniques are applied to two parts (Part A and B). First, three datasets were selected based on an idea which showed the evaluation parameters include precession, recall, test error and ROC curve. Second, the ability of the proposed framework so as to detect the thin boundary challenge between new anomalies based on eight UCI datasets has considered. Additionally, we have used a few benchmark datasets based on the repository, which proposed in Campos et al. (2016) to calculate both true positive rate and false positive rate. It should be noted that datasets have picked in various fields.

**Part A**

The proposed techniques are applied to three datasets. The first dataset has been randomly generated and its data are labeled according to Table 3 after applying a clustering algorithm. Figure 6 shows the distribution of the first dataset in a two-dimensional space. Using the confusion matrix, the results of the enhanced and conventional MLP-NN are compared.

According to the matrix (Figures 7 and 8), the test error of the MLP-NN enhanced by the GA is 10% while the corresponding error for the conventional MLP-NN is 26.7%. Here, 1 represents the ND, 2 the CNA, 3 the CPA and 4 represents PA. The most important thing is high-precision detection of the thin boundary between various types of anomalies as is visible in the confusion matrix.

The second dataset is used to apply the proposed techniques in Rehm, Klawonn, and Kruse (2007). Table 4 shows the labeled data in the second

| Cluster | Cluster | Cluster | Cluster | Cluster | 5 #Cluster |
|---------|---------|---------|---------|---------|------------|
| 5 #CNA  | 4 #CNA  | 3 #CNA  | 2 #ND   | 1 #ND   | 73 #ND     |
| 3 12    | 12 25   | 48      |         |         | 27 #CNA    |
| 53      | 42      |         |         |         | 53 #CPA    |
|         |         |         |         |         | 42 #PA     |

**Table 3.** Labeling data in the first dataset.
dataset. Figure 9 shows the distribution of data in the second dataset in a two-dimensional space.

Figures 10 and 11 show the confusion matrix for the second dataset. The reasons why no data is not selected for Class 2 in the testing step are obvious: (1) insufficient data, (2) the procedure used to select and divide data in training, testing and evaluation steps. In the case of insufficient data, especially anomalies in a given dataset, training, evaluation and testing are performed using small number of data and even some classes may not be selected. To solve this problem, the following question should be answered: “Based on what criteria, the data are divided for testing, evaluation and training data?”

Random classification lowers the detection quality because different types of anomalies may be selected in the training or evaluation steps and this will decrease the quality of other stages and perhaps all samples of a type of anomaly may be selected in a stage. Our proposed method utilizes the same percentage to select different types of anomalies and data. For example, if
70% of the data is selected for training, a same ratio of a variety of data should be selected. For this purpose, DivideFcn function is defined in the program code to data. In order to select data at various stages of training, evaluation and testing on the basis of equal proportions this function is used. DivideFcn function is used to study the Iris dataset. It causes to increase the reliability of data selecting in each stage.

The third dataset used in this study is Iris dataset. Unlike previous datasets, Iris is a supervised dataset (Multi-Class). Therefore, the dataset is divided into three sub-datasets according to the proposed framework section (Setosa, Versicolor and Virginica, because there are three classes). The two main features in this dataset include petal length and petal width because they show the highest distinction between the data globally. Below, the data in each sub-dataset are labeled as shown in Table 5 to 7.

Figure 12 shows the distribution of three classes of Iris datasets in a two-dimensional space.

**Figure 7.** Neural network confusion matrix for the first dataset.
Figures 13 and 14 show the confusion matrix for the Iris dataset. The matrix indicates the quality of proposed techniques in a supervised dataset. The thin boundary between anomalies in Class 2 and 3 in the conventional MLP-NN is unacceptable while the MLP-NN enhanced by the GA provides acceptable results.

Figure 15 shows the comparison of test error parameter based on enhanced and conventional MLP-NN.

Table 4. Labeling data in the second dataset.

| Cluster | Cluster | Cluster | #Point | #Cluster |
|---------|---------|---------|--------|----------|
| 3       | 2       | 1       | 45     |          |
| #CNA    | #ND     | #ND     | 22     | #ND      |
| 4       | 9       | 13      | 4      | #CNA     |
| 13      |         |         | 13     | #CPA     |
| 6       |         |         | 6      | #PA      |

Figures 13 and 14 show the confusion matrix for the Iris dataset. The matrix indicates the quality of proposed techniques in a supervised dataset. The thin boundary between anomalies in Class 2 and 3 in the conventional MLP-NN is unacceptable while the MLP-NN enhanced by the GA provides acceptable results.

Figure 15 shows the comparison of test error parameter based on enhanced and conventional MLP-NN.
Figure 9. Distribution of data in the second dataset in a two-dimensional space.

Figure 10. Neural network confusion matrix for the second dataset.
At the end of this part, the other standard evaluation metrics, called True Positive Rate (TPR/sensitivity) and False Positive Rate (FPR), are calculated based on Equations (14)–(16) and results are presented in Table 8. Table 9 shows the specification of UCI dataset and Table 10 shows specification of dataset which was used by Campos et al. (Campos et al., 2016). Additionally, Figures 16–18 show that the proposed method which enhanced by GA outperforms the conventional MLP-NN in terms of ROC curve. Also, Figure 19 shows the number of normal and anomaly data of selected data.

![Test GA Confusion Matrix](image)

**Figure 11.** Genetic algorithm confusion matrix for the second dataset.

| Cluster | Cluster #Point | #Cluster |
|---------|---------------|----------|
| Cluster 2 | 15 | #ND |
| Cluster 1 | 19 | #CNA |
| Cluster 3 | 9 | #CPA |
| #CNA | 7 | #PA |

**Table 5.** Labeling data in the third dataset, the first sub-dataset.
Table 6. Labeling data in the third dataset, the second sub-dataset.

| Cluster | Cluster | Cluster | #Point |
|---------|---------|---------|--------|
| 3       | 2       | 1       | 50     |
| #CNA    | #CNA    | #ND     |        |
| 5       | 8       | 20      |        |

Table 7. Labeling data in the third dataset, the third sub-dataset.

| Cluster | Cluster | Cluster | Cluster | #Point |
|---------|---------|---------|---------|--------|
| 4       | 3       | 2       | 1       | 50     |
| #CNA    | #CNA    | #CNA    | #ND     |        |
| 4       | 8       | 6       | 17      |        |

Figure 12. Iris dataset distribution in a two-dimensional space, green: Setosa, red: Versicolor and blue: Virginica.
Figure 13. Neural network confusion matrix for Iris dataset.

Figure 14. Genetic algorithm confusion matrix for Iris dataset.

set before the definition of new type of anomalies in this paper. Figure 20 shows these type of data after definition.

\[ TPR(\text{Sensitivity}) = \frac{TP}{TP + FN} \]  

(14)
Table 8. The comparison of TPR and FPR metrics in three datasets based on confusion matrix.

| Classes | 1(ND) | 2(CNA) | 3(CPA) | 4(PA) |
|---------|-------|--------|--------|-------|
|         | TPR   | FPR    | TPR    | FPR   |
| D1-without GA | 1 0.166667 | 0   0 | 0.857143 | 0.2 0.666667 |
| D1-using GA   | 1 0.0625 | 0.6 0 | 0.857143 | 0 1 0.086957 |
| D2-without GA | 1 0.5 0 | 0 | 1 0.2 0 | 0 |
| D2-using GA   | 1 0 0 | 0 | 1 0 | 1 0 |
| D3-without GA | 1 0.363636 | 0.571429 0.2 | 0 0 | 1 0 |
| D3-using GA   | 1 0.166667 | 0.714286 0.125 | 0.5 0 | 1 0 |

Table 9. Number of data based on overall eight UCI datasets.

| Dataset name | #Cases | #Classes | #Attributes | #Cluster | #ND | #CNA | #CPA | #PA |
|--------------|--------|----------|-------------|----------|-----|------|------|-----|
| Splice       | 3190   | 3        | 62          | 12       | 2422 | 477  | 215  | 76  |
| Anneal       | 898    | 6        | 39          | 9        | 452  | 346  | 68   | 32  |
| Breast-w     | 699    | 2        | 9           | 7        | 412  | 199  | 63   | 25  |
| Credit-a     | 690    | 2        | 15          | 7        | 418  | 174  | 71   | 27  |
| Soybean      | 683    | 19       | 35          | 6        | 454  | 148  | 58   | 23  |
| Monk         | 432    | 2        | 6           | 5        | 302  | 62   | 49   | 19  |
| Colic        | 368    | 2        | 22          | 5        | 157  | 129  | 54   | 28  |
| Heart-c      | 303    | 2        | 13          | 4        | 146  | 95   | 41   | 21  |

Table 10. Datasets available on Campos et al. (2016) repository.

| Dataset name | Instances | Inliers | Outliers | Attributes |
|--------------|-----------|---------|----------|------------|
| WPBC         | 198       | 151     | 47       | 33         |
| Ionosphere   | 351       | 225     | 126      | 32         |
| Waveform     | 3443      | 3343    | 100      | 21         |
| Ananthroid   | 7200      | 6666    | 534      | 21         |
| Pima         | 768       | 500     | 268      | 8          |
| Parkinson    | 195       | 18      | 147      | 22         |
True Negative Rate (TPR or Specificity) = \frac{TN}{TN + FP} \quad (15)

FPR = 1 - TNR \quad (16)
Figure 18. The ROC curve based on D3 dataset.

Figure 19. The thin boundary challenge between normal data and anomalies before new definition of anomalies overall eight UCI datasets.
Part B

In this part, the proposed techniques have applied on eight UCI datasets which presented the number of ND, CNA, CPA, PA and the thin boundary challenge between new anomalies including both CNA and CPA. Additionally, we have used a few benchmark datasets based on Campos et al. (2016) repository, which are more appropriate to consider ROC curve and results are presented in Figures 21–26.

Conclusion

The thin boundary between various types of anomalies was studied. For this purpose, a new framework was introduced to adapt the proposed approach to both supervised and unsupervised datasets. Then, the MLP-NN was enhanced using the GA to ensure a test error less than that calculated for the conventional MLP-NN. Moreover, new types of anomalies were investigated by applying the proposed method on benchmark datasets. The most important features of these methods include adaptability to both supervised and unsupervised datasets, improved detection

Figure 20. The thin boundary challenge between normal data and anomalies after new definition of anomalies overall eight UCI datasets.
Figure 21. The ROC curve based on WPBC dataset.

Figure 22. The ROC curve based on Ionosphere dataset.
Figure 23. The ROC curve based on Waveform dataset.

Figure 24. The ROC curve based on Annthyroid dataset.
Figure 25. The ROC curve based on Pima dataset.

Figure 26. The ROC curve based on Parkinson dataset.
of various types of anomalies, increased reliability and enhancement of MLP-NN by the GA.

In the future work, for comparing both the efficiency and effectiveness of the proposed algorithm large datasets in a special field (e.g., intrusion detection, credit card fraud detection) will be used. A suitable technique will be provided to inject outliers in a dataset with insufficient outliers. This results in high-quality division of data into training, evaluation and testing data and thus reduces the test error.

**ORCID**

Amin Keshavarzi [http://orcid.org/0000-0002-1117-4052](http://orcid.org/0000-0002-1117-4052)

**References**

Agarwal, S. 2013. Data mining: Data mining concepts and techniques. Machine Intelligence and Research Advancement (ICMIRA), 2013 International Conference on, Katra, India, IEE, 203–07.

Aggarwal, C. C. 2015. Outlier analysis. *Data mining*, 237–63. Springer, Springer, Berlin, Heidelberg.

Ahmed, M., and A. N. Mahmood 2014. Network traffic analysis based on collective anomaly detection. Industrial electronics and applications (ICIEA), 2014 IEEE 9th Conference on, Hangzhou, China, IEEE, 1141–46.

Aleskerov, E., B. Freisleben, and B. Rao 1997. CARDWATCH: A neural network based database mining system for credit card fraud detection. Computational Intelligence for Financial Engineering (CIFEr), New York City, USA, 1997., Proceedings of the IEEE/IAFE 1997, IEEE, 220–26.

Augusteijn, M., and B. A. Folkert. 2002. Neural network classification and novelty detection. *International Journal of Remote Sensing* 23:2891–902.

Barson, P., S. Field, N. Davey, G. McAskie, and R. Frank. 1996. The detection of fraud in mobile phone networks. *Neural Network World* 6:477–84.

Bohlouli, M., F. Schulz, L. Angelis, D. Pahor, I. Brandic, D. Atlan, and R. Tate. 2013. Towards an integrated platform for big data analysis. In: Fathi M. (eds) *Integration of practice-oriented knowledge technology: Trends and prospectives*, 47–56. Berlin, Heidelberg: Springer.

Bolton, R. J., and D. J. Hand. 2001. Unsupervised profiling methods for fraud detection. *Credit Scoring and Credit Control* VII:235–55.

Boyd, T., G. Docken, and J. Ruggiero. 2016. Outliers in data envelopment analysis. *Journal of Centrum Cathedra*, Vol. 9 No. 2, pp. 168-183.

Campbell, C., and Bennett, K. 2000. A Linear Programming Approach to Novelty Detection. In Proceedings of the 13th International Conference on Neural Information Processing Systems, Denver, Colorado, USA, Wiki Loves LovePhotograph your local culture, help Wikipedia and win!Colorado (pp. 374–380). MIT Press, USA.

Campos, G. O., A. Zimek, J. Sander, R. J. Campello, B. Micenková, E. Schubert, I. Assent, and M. E. Houle. 2016. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery* 30 (4):891–927. doi:10.1007/s10618-015-0444-8.
Challagalla, A., S. S. Dhiraj, D. V. Somayajulu, T. S. Mathew, S. Tiwari, and S. S. Ahmad 2010. Privacy preserving outlier detection using hierarchical clustering methods. Computer Software and Applications Conference Workshops (COMPSACW), Seoul, Korea, 2010 IEEE 34th Annual, IEEE, 152–57.

Chandarana, D. R., and M. V. Dhamecha 2015. A survey for different approaches of Outlier Detection in data mining. International Conference on Electrical, Electronics, Signals, Communication and Optimization (EESCO). Visakhapatnam, India

Chandola, V., A. Banerjee, and V. Kumar. 2009. Anomaly detection: A survey. ACM computing surveys (CSUR) 41:15.

Diaz, I., and J. Hollmen 2002. Residual generation and visualization for understanding novel process conditions. Neural Networks, 2002. IJCNN’02. Honolulu, HI, USA, Proceedings of the 2002 International Joint Conference on, IEEE, 2070–75.

Fujimaki, R., T. Yairi, and K. Machida 2005. An approach to spacecraft anomaly detection problem using kernel feature space. Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, Chicago, USA, ACM, 401–10.

Ghosh, A. K., J. Wanken, and F. Charron 1998. Detecting anomalous and unknown intrusions against programs. Computer Security Applications Conference, 1998. Proceedings. 14th Annual, Phoenix, AZ, USA, IEEE, 259–67.

Guo, F., C. Shi, X. Li, J. He, and W. Xi 2018. Outlier detection based on the data structure. 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, IEEE, 1–6.

Gupta, M., J. Gao, C. Aggarwal, and J. Han. 2014. Outlier detection for temporal data. Transactions on Knowledge and Data Engineering 5:1–129.

Hachmi, F., K. Boujenfa, and M. Limam 2015. A three-stage process to detect outliers and false positives generated by intrusion detection systems. Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on, Liverpool, UK, IEEE, 1749–55.

Hauskrecht, M., M. Valko, I. Batal, G. Clermont, S. Visweswaran, and G. F. Cooper 2010. Conditional outlier detection for clinical alerting. AMIA annual symposium proceedings, American Medical Informatics Association, Washington, DC, USA, 286.

Hodge, V., and J. Austin. 2004. A survey of outlier detection methodologies. Artificial intelligence review 22:85–126.

Jafarizadeh, V., A. Keshavarzi, and T. Derikvand. 2017. Efficient cluster head selection using Naïve Bayes classifier for wireless sensor networks. Wireless Networks 23 (3):779–85. doi:10.1007/s11276-015-1169-8.

Keshavarzi, A., A. T. Haghighat, and M. Bohlouli. 2017. Adaptive resource management and provisioning in the cloud computing: A survey of definitions, standards and research roadmaps. KSII Transactions on Internet & Information Systems 11 (9), pp. 4280-4300.

Keshavarzi, A., A. T. Haghighat, and M. Bohlouli. 2019. Enhanced time-aware QoS prediction in multi-cloud: A hybrid k-medoids and lazy learning approach (QoPC). Computing, 1–27.

Keshavarzi, A., A. M. Rahmani, M. Mohsenzadeh, and R. Keshavarzi. 2008. Recognition of data records in semi-structured web-pages using ontology and χ² statistical distribution. In Advanced data mining and applications. ADMA 2008. Lecture notes in computer science, ed. C. Tang, C. X. Ling, X. Zhou, N. J. Cercone, and X. Li, vol. 5139. (pp 675-682) Berlin, Heidelberg: Springer.

Kiani, R., S. Mahdavi, and A. Keshavarzi. 2015. Analysis and prediction of crimes by clustering and classification. International Journal of Advanced Research in Artificial Intelligence 4 (8):11–17. doi:10.14569/issn.2165-4069.
Ko, T., J. H. Lee, H. Cho, S. Cho, W. Lee, and M. Lee. 2017. Machine learning-based anomaly detection via integration of manufacturing, inspection and after-sales service data. *Industrial Management & Data Systems* 117:927–45. doi:10.1007/s13760-017-0817-4.

Kou, Y., C.-T. Lu, and R. F. Dos Santos 2007. Spatial outlier detection: A graph-based approach. *ictai, Patras, Greece, IEEE*, 281–88.

Kutsuna, T., and A. Yamamoto. 2017. Outlier detection using binary decision diagrams. *Data Mining and Knowledge Discovery* 31 (2):548–72. doi:10.1007/s10618-016-0486-6.

Li, X., and J. Haupt. 2015. Identifying outliers in large matrices via randomized adaptive compressive sampling. *IEEE Transactions on Signal Processing* 63:1792–807.

Li, Y., M. J. Pont, and N. B. Jones. 2002. Improving the performance of radial basis function classifiers in condition monitoring and fault diagnosis applications where unknown faults may occur. *Pattern Recognition Letters* 23:569–77.

Lin, C., Q. Zhu, S. Guo, Z. Jin, Y. R. Lin, and N. Cao. 2018. Anomaly detection in spatiotemporal data via regularized non-negative tensor analysis. *Data Mining and Knowledge Discovery* 32 (4):1056–73. doi:10.1007/s10618-018-0560-3.

Macha, M., and L. Akoglu. 2018. Explaining anomalies in groups with characterizing subspace rules. *Data Mining and Knowledge Discovery* 32 (5):1444–80. doi:10.1007/s10618-018-0585-7.

Maheshwari, K., and M. Singh 2016. Outlier detection using divide-and-conquer strategy in density based clustering. Recent Advances and Innovations in Engineering (ICRAIE), 2016 International Conference on, Jaipur, India, IEEE, 1–5.

Malik, K., H. Sadawarti, and G. S. Kalra. 2014. Comparative analysis of outlier detection techniques. *International Journal of Computer Applications* 97:12–21.

Manevitz, L. M., and M. Yousef. 2001. One-class SVMs for document classification. *Journal of machine Learning research* 2:139–54.

Noble, C. C., and D. J. Cook 2003. Graph-based anomaly detection. Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, Washington, D.C., USA, ACM, 631–36.

Ramadas, M., S. Ostermann, and B. Tjaden 2003. Detecting anomalous network traffic with self-organizing maps. International Workshop on Recent Advances in Intrusion Detection, Pittsburgh, PA, USA, Springer, 36–54.

Rehm, F., F. Klawonn, and R. Kruse. 2007. A novel approach to noise clustering for outlier detection. *Soft Computing* 11:489–94.

Riahi, F., and O. Schulte 2015. Model-based outlier detection for object-relational data. Computational Intelligence, 2015 IEEE Symposium Series on, Cape Town, South Africa, IEEE, 1590–98.

Singh, S., and M. Markou. 2004. An approach to novelty detection applied to the classification of image regions. *IEEE Transactions on Knowledge and Data Engineering* 16:396–407.

Smith, R., A. Bivens, M. Embrechts, C. Palagiri, and B. J. Szymanski 2002. Clustering approaches for anomaly based intrusion detection. Proceedings of intelligent engineering systems through artificial neural networks, St. Louis, Missouri, USA, 579–84.

Sohn, H., K. Worden, and C. R. Farrar 2001. Novelty detection under changing environmental conditions. Smart Structures and Materials 2001: Smart Systems for Bridges, Structures, and Highways, International Society for Optics and Photonics, 108–19.

Song, X., M. Wu, C. Jermaine, and S. Ranka. 2007. Conditional anomaly detection. *IEEE Transactions on Knowledge and Data Engineering* 19:631–45.

Steinwart, I., D. Hush, and C. Scovel. 2005. A classification framework for anomaly detection. *Journal of Machine Learning Research* 6:211–32.

Taniguchi, M., M. Haft, J. Hollmen, and V. Tresp 1998. Fraud detection in communication networks using neural and probabilistic methods. *Acoustics, Speech and Signal Processing,*
1998. Seattle, WA, USA, Proceedings of the 1998 IEEE International Conference on, IEEE, 1241–44.

Theiler, J. P., and D. M. Cai 2003. Resampling approach for anomaly detection in multispectral images. Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery IX, International Society for Optics and Photonics, Orlando, Florida, USA, 230–41.

Vatanen, T., M. Kuusela, E. Malmi, T. Raiko, T. Aaltonen, and Y. Nagai 2012. Semi-supervised detection of collective anomalies with an application in high energy particle physics. IJCNN, Brisbane, QLD, Australia, 1–8.

Vijayarani, S., and S. Nithya. 2011. An efficient clustering algorithm for outlier detection. International Journal of Computer Applications 32:22–27.

Wang, X., and I. Davidson 2009. Discovering contexts and contextual outliers using random walks in graphs. Data Mining, 2009. ICDM’09. Ninth IEEE International Conference on, Miami, FL, USA, IEEE, 1034–39.

Yang, S., and W. Liu 2011. Anomaly detection on collective moving patterns: A hidden Markov model based solution. 2011 IEEE International Conferences on Internet of Things, and Cyber, Physical and Social Computing, Dalian, China, IEEE, 291–96.

Zhang, J. 2008. Towards outlier detection for high-dimensional data streams using projected outlier analysis strategy. Canada: Dalhousie University.

Zhang, J., J. Cao, and X. Zhu 2012. Detecting global outliers from large distributed databases. Proceedings of the 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2012), Sichuan, China, IEEE, 1632–36.

Zhang, L., C. Liu, Y. Chen, and S. Lao 2018. Abnormal Detection Research Based on Outlier Mining. 2018 11th International Conference on Intelligent Computation Technology and Automation (ICICTA), Changsha, China, IEEE, 5–7.

Zhang, Z., J. Li, C. Manikopoulos, J. Jorgenson, and J. Ucles 2001. HIDE: A hierarchical network intrusion detection system using statistical preprocessing and neural network classification. Proc. IEEE Workshop on Information Assurance and Security, 85–90.

Zhao, Y., and M. K. Hryniewicki 2018. Xgbod: Improving supervised outlier detection with unsupervised representation learning. 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, IEEE, 1–8.