Wireless Personal Communications (2022) 124:2637–2659
https://doi.org/10.1007/s11277-022-09482-8

Enhancing Detection of R2L Attacks by Multistage Clustering Based Outlier Detection

J. Rene Beulah[1] · M. Nalini[2] · D. Shiny Irene[1] · D. Shalini Punithavathani[3]

Accepted: 4 January 2022 / Published online: 29 January 2022

Abstract
The modern society is greatly benefited by the advancement of Internet. The contemporary humanity is significantly profited by the Internet. The ease of access to the Internet have given rise to tremendous security threats. With the emergence of new varieties of attacks, the attack prevention techniques like firewall, data encryption and user authentication are not adequate in making a system completely secure because guaranteed prevention of all kinds of security breaches is impractical. Intrusions pose a serious threat to individuals and organizations in this digital era. An Intrusion Detection System operates as part of a set of system security tools to achieve a defined level of assurance for the protection of information systems. In this work, a novel multistage clustering-based approach is proposed and implemented which addresses the challenge of increasing DR while maintaining a low FAR. The novelty of this work lies in the way of clustering which works in a reverse manner and forms clusters in a more meaningful way and which is applicable for mixed attribute types. In addition, the multiple stages of clustering help in identifying most of the Remote to Local (R2L) attacks. The performance of the proposed method is evaluated on the standard NSL-KDD benchmark dataset and the experimental results yielded 99.52% detection rate (DR), 1.15% false alarm rate and 99.22% classification accuracy. In specific, it deliberates on detecting R2L attacks and has detected 98.73% of such attacks.

Keywords Classification · Feature extraction · NSL-KDD · Semi-supervised · Intrusion detection · Anomaly

1 Introduction

The developing accessibility of Internet has expanded the assortment of online services offered to business and private clients. However, these opportunities are exploited which have led to the alarming increase in the number and sophistication of attacks on online users. Specialized security mechanisms have made it harder to intrude secured networks, but still intrusion is possible. Configuration errors, technology weaknesses or security

✉ J. Rene Beulah
renebeulah@gmail.com

Extended author information available on the last page of the article

policy limitations may sometimes lead to intrusion of networked systems and to information stealing. This gives rise to the indispensable need for intrusion detection systems.

Intrusion Detection System (IDS) has been a subject of substantial research for the past four decades. Many research works are available which aim in improving accuracy by applying various techniques. Still network intrusion detection is a dynamic research area as intruders or attackers have increased attacks on all kinds of networking set-ups. Currently known intrusion detection techniques can be categorized into two generic models namely misuse detection and anomaly detection. In misuse detection systems, signatures of known attacks are stored in a database which is then used to examine the incoming traffic data. In anomaly-based systems, every activity in the network is monitored by which the normal network behavior is learnt. Every deviation from this normal behaviour is used to signify intrusion. Despite being efficient in identifying new attacks, anomaly-based systems suffer from high false alarm rate. The typical network intrusion detection research focuses on the proposal and implementation of new detection techniques with increased detection capability and reduced false alarms.

The research work presented in this paper aims to improve anomaly-based intrusion detection method by applying a data mining technique called outlier analysis. Outlier analysis helps in determining objects that are distinctly different from the rest of the objects and hence is suitable for anomaly-based intrusion detection. There are many approaches for detecting outliers among which clustering-based methods are more common. Since clustering is a natural way for identifying outliers, this research employs a semi-supervised clustering-based approach to learn normal behavior and to identify attack sequences.

Network attacks are generally classified into four types Denial of Service (DoS) attacks, Probing attacks, User to Root (U2R) attacks and Remote to Local (R2L) attacks. Most of the research works in the literature are efficient in detecting DoS, Probe and U2R attacks, but fail to detect R2L attacks. Usually, R2L attacks are inserted in the data segments of a packet and hardly require a few connections. R2L attacks are rare and involve only a few network connection which make it very hard to detect it [1]. However, failing to identify R2L attacks is more serious than that of probe attacks or Denial of Service attacks. R2L attacks grant local network access to the attacker [2, 3]. This work focuses on improving the detection rate of R2L attacks, thereby improving the overall accuracy of the IDS.

The significant contribution of this research is a new method for detecting the presence of outliers in a dataset. The method proposed is a clustering-based one. Even though there are some techniques in the present research with respect to outlier detection using clustering, the work presented in this paper differs significantly as it applies clustering in reverse. It is also claimed in this work that the clusters formed are meaningful. In addition, the proposed work can be applied for datasets with different types of attributes like nominal, ordinal and real values, which is a real challenge. In this paper, the proposed outlier detection method is applied for finding network intrusions. But it can be applied for problems in any application domain. Another noteworthy claim is that, compared to the other similar works in literature, the proposed work is able to enhance the detection of R2L attacks.

The rest of this paper is organized as follows: Section 2 presents a review of literature. Section 3 describes the framework of outlier based intrusion detection in a semi-supervised fashion. Section 4 gives a description of the dataset used and discusses Improved Hybrid Feature Selection (IHFS), the method applied for feature selection. Section 5 gives a brief account on Clustering Based Outlier Detection (CBOD), our previous work, analyzes its performance and pinpoints its inability to detect some R2L attacks. Section 6 gives a comprehensive description of the proposed method, Multi-stage Clustering Based Outlier Detection (MCBOD), proves its ability to detect most of the R2L attacks and analyzes its

performance with reference to the existing methods in the literature. The conclusions and future scope are presented in sect. 7.

2 Related Work

Denning was the first researcher to propose Anomaly-based IDS in 1987 [4]. Much has been published attempting to improve the performance of anomaly-based techniques for the past few decades and it is not possible to cover all aspects in a short review. Several promising anomaly-based approaches have been examined in the literature trying to provide a safe and secure environment for organizations. A summary of such anomaly-based methods, its classification, advantages and disadvantages, a detailed review of such schemes can be found in [5–12]. This section presents a comprehensive list of works with specific reference to data mining.

Data mining methods were first applied for IDS by Lee and Stolfo [13]. Since then, data mining is one of the most known methods suitable for anomaly-based intrusion detection [14]. Even though there are many types of attacks and many ways of deploying IDS, the attack behavior can be learnt by applying Machine learning and deep learning techniques [15]. An elaborate survey of research works involving data mining and machine learning methods is given in [2]. A review on outlier/anomaly detection in time series data is presented in [16] which provides a structured and comprehensive state-of-the-art on outlier detection techniques in the context of time series.

De la Hoz et al. [17] have applied an amalgamation of statistical methods and Self-Organizing Maps (SOM) to detect intrusions. Principal Component Analysis (PCA) and Fisher Discriminant Ratio (FDR) were used for feature selection and Bayesian SOM is applied for identifying attacks. It reports a detection rate of 97% and false alarm rate of 7%.

Tahir et al. [18] have combined K-means clustering and Support Vector Machine (SVM) to identify intrusions and the results show a DR of 96.26% and a FAR of 3.7%.

Singh et al. [19] have proposed an intrusion detection system using network traffic profiling and online sequential extreme learning machine (OS-ELM). An ensemble feature selection method is used and 21 features were selected. The features ‘protocol’ and ‘service’ are combined in Alpha Profiling. A clustering algorithm is used to group redundant connections in Beta Profiling. OS-ELM is used as a classifier to detect intrusions. The experimental results show a DR of 99.01% and a FAR of 1.74%.

Bhuyan et al. [20] have presented a multi-step outlier-based approach for anomaly detection (MOAD). An attribute selection method applying mutual information and generalized entropy is attempted to pick the important attributes. A tree-based clustering technique is used to generate reference points and an outlier score function ranks the incoming network technique to identify anomalies. The method produced 97.23% DR and 0.98% FAR.

Bamakan et al. [21] have proposed a framework using a new optimization method time-varying chaos particle swarm optimization (TVCPSO) to do parameter setting and feature selection for multiple criteria linear programming (MCLP) and SVM. The empirical results show 97.03% DR and 0.87% FAR.

Enache et al. [22] have suggested an IDS model IG-BAL-SVM, that uses information gain (IG) to select 26 features and SVM classifier. Bat Algorithm with Levy flights (BAL) is used to enhance the randomization of the input parameters of SVM. It is claimed that a DR of 99.15% and a FAR of 1.9% can be achieved.

Hassan [23] has built a cost sensitive learning model 'Metacost' for detecting R2L and U2R attacks. This technique has two phases: bagging for relabeling each training example with the cost, and retraining the classifier with the cost. KDD-cup 99 dataset is used for testing the results.

Paliwal and Gupta [24] have proposed a methodology based on Genetic Algorithm for detecting R2L attacks. KDD-cup 99 dataset is used for experiments.

Revathi and Malathi [25] have proposed a new concept for examining every R2L attack using random forest algorithm. This work is carried out on NSL-KDD dataset using Weka and the results are analyzed in terms of Precision, Recall and F-value.

Jeya et al. [26] have proposed a model using Fisher Discriminant Analysis, applied it on KDD-Cup 99 dataset and have improved the detection rate of U2R and R2L attacks.

Nguyen et al. [27] have proposed a novel hybrid approach between clustering methods and autoencoders for detecting network anomalies in a semi-supervised manner. The distance to the closest cluster is used as an anomaly score and the method is applied on CTU13 dataset.

Pu et al. [28] have suggested an unsupervised anomaly detection method which combines Sub-Space Clustering and One Class Support Vector Machine to detect attacks without any prior knowledge and have evaluated the method using the NSL-KDD dataset.

A Multi-level outlier detection algorithm is proposed by Li et al. [29] that uses multi-level unsupervised learning to cluster the data and discover outliers and is tested on datasets in different fields with different sizes and dimensions.

Elmogy et al. [30] have presented a clustering based outlier detection framework which enables analysts to effectively find out outliers on time with request even within huge datasets.

Aljawarneh et al. [31] have proposed a model which merges 7 classifiers in WEKA. Initially, Vote Algorithm and Information Gain are applied to filter data. The hybrid classifier is applied next. Experiments are carried out on NSL-KDD dataset.

Tama et al. [32] have developed a two-level classifier collective model based on rotation forest and bagging, where an ensemble feature selection method which applies particle swarm optimization, ant colony algorithm and genetic algorithm is used to extract important attributes.

Mohammed et al. [33] have implemented a deep neural network (DNN) for classifying intrusions from normal network traffic. The model has a fully-connected network structure with three layers with rectified linear unit (ReLU) activation function on the first two layers and the Sigmoid function on the output layer and the method is applied on NSL-KDD dataset.

Manimurugan et al. [34] have used Crow Search Optimization algorithm with Adaptive Neuro-Fuzzy Inference System (ANFIS) and applied it on NSL-KDD dataset.

On reviewing the literature, it is inferred that very few methods were designed to handle attributes with mixed types. In addition, it is noted that some works were successful in improving the DR and some others in reducing FAR, while it is always a tough task to achieve a fair balance between DR and FAR which is indeed a difficult task. Another common problem is the reduced detection rate of R2L attacks.

To fill the research gap in the earlier research, we have proposed a well-defined anomaly detection scheme that can handle data with mixed attribute types efficiently, that maintains a good balance between DR and FAR and that can detect most of the R2L attacks. The proposed MCBOD method was successful with respect to (i) The overall increase in DR to 99.52% (ii) The reduction of FAR to 1.15% (iii) The improved overall classification accuracy of 99.23% (iv) Can detect 98.73% of R2L attacks.

3 Semi-Supervised Outlier-Based Network Anomaly Identification

Approaches for network anomaly identification can be broadly classified into three: *supervised*, *unsupervised* and *semi-supervised*. If an IDS is trained with labeled normal as well as anomalous instances, then the approach is supervised. If no training data is used, then it an unsupervised approach. Usually, anomaly detection schemes make use of the normal instances alone for training purposes and these approaches are considered as semi-supervised.

Figure 1 shows an outline of a semi-supervised anomaly based IDS. The three phases are: Attribute/Feature Selection, Training and Testing. Intrusion datasets come with separate training and testing sets. The aim of attribute selection is to select the best attributes that help in accurate classification. Anomaly based methods have to learn the normal traffic pattern and hence only the instances that represent normal traffic in the training data are given as input to the training phase which creates a classifier model as output. All the instances in the testing data are given as input to this classifier model which predicts whether each traffic connection in the testing data is normal or an attack. The prediction of the classifier model is compared to that of the actual class to come up with the confusion matrix which helps in analyzing the results with respect to various parameters.

Outlier detection is a data mining technique that has become an exceedingly important area of research as it finds application in diverse fields. Gogoi et al. have presented a detailed survey of outlier analysis methods with special reference to that used for intrusion detection [35].

For anomaly-based IDS, outlier detection is more suitable as outliers indicate activities that deviate from the normal behavior. In some applications, outliers are treated as noise

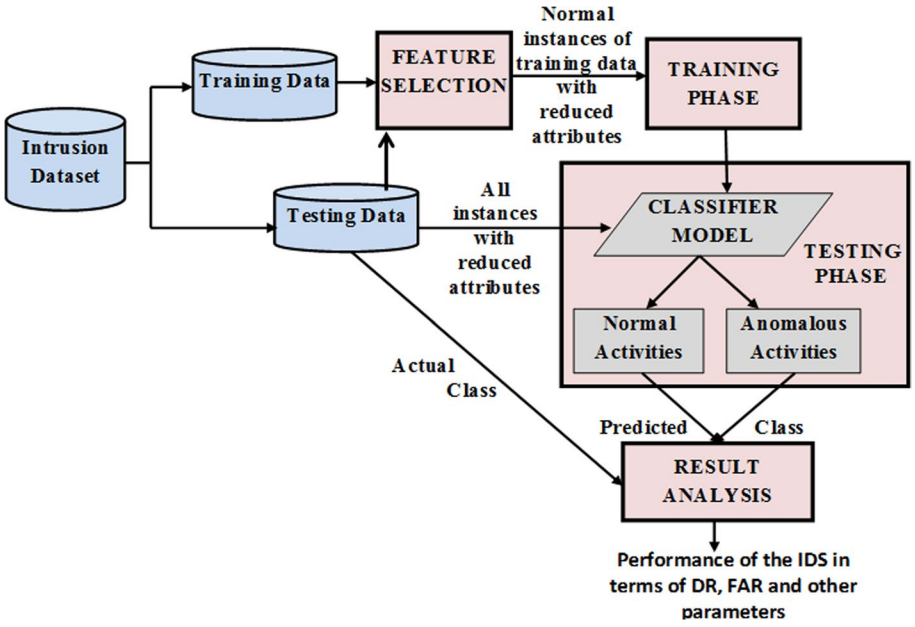

Fig. 1 Framework of a semi-supervised anomaly-based IDS

and are removed in the pre-processing step whereas in some other application, outliers are the elements carrying significant information. IDS The concept of outlier detection is much suitable for anomaly-based intrusion detection systems as intrusive behaviors differ from that of normal user behavior. An outlier may denote an intruder in a network with malicious intentions. Wherever the applications of outlier detection are listed, we can find network intrusion detection listed in it and outlier detection is finding its prominent place in the literature of IDS. There are different approaches for outlier detection and an up-to-date survey of outlier detection techniques applied for IDS are detailed in [36].

Among the varied approaches for IDS using outlier detection, clustering helps in identifying outliers more naturally [37]. Clustering-based approaches group the dataset using some clustering algorithm to divide the dataset into clusters. It is inferred that further research in clustering-based outlier detection may yield promising results and hence the work proposed in this paper follows this approach. The network traffic data is clustered based on the properties of the connections and the behavior of each cluster is analyzed to learn the normal network behavior.

4 Dataset Description & Feature Selection

4.1 Dataset Description

KDDCup99 has served as the benchmark dataset for evaluating the performance of intrusion detection techniques from its inception, i.e., from the year 2000 to 2014. Recently, researchers have started using NSL-KDD, a smaller version of KDDCup99 [38]. Though a variety of outlier detection approaches have been proposed for intrusion detection in the literature, the efficiency and performance cannot be exactly compared because KDDCup99 dataset is much bulky and researchers have presented the results obtained on their own sub samples and not on the entire dataset. The introduction of NSL-KDD solves this problem as it has reasonable number of records which makes it possible for the researchers to work and publish their results on the entire dataset. This has paved the way for researchers to perform an effective comparison of different approaches.

The NSL-KDD dataset has separate training and testing datasets with 1,25,973 instances in the training set and 22,544 instances in the testing set. Each instance represents a connection, which is a sequence of TCP packets starting and ending at some well-defined time, between which data flows to and from a source IP address to a target IP address under some well-defined protocol. Each instance is characterized by 41 attributes/features where the last attribute is the class attribute which tells whether the instance represents a normal connection or an attack. The attacks given in the dataset correspond to one of the four types – DoS, Probe, U2R and R2L. DoS and probe attacks are network-based attacks that leave traces in network packet data. R2L attacks also involve network-based attacks but also include some attacks that attempt to misuse host-based programs. U2R attacks attempt to gain super user privileges on the host machine either by misusing programs or by running malicious software.

The training set contains 22 attack types and the testing set contains 38 attack types. The testing set is specifically designed to contain 17 additional attack types that are not present in the training set so as to check the capability of the intrusion detection techniques to detect new unseen attacks. Also the probability distribution of the training and testing

sets are different which makes the intrusion task more realistic. A detailed description of the dataset is given in [39].

4.2 Improved Hybrid Feature Selection (IHFS)

Preprocessing is an essential step as the intrusion dataset is high dimensional and the attributes are of varied types. Feature selection plays an important role in enriching the accuracy of the classifier by removing the irrelevant and redundant attributes that distract the classification process. A hybrid feature selection method, IHFS [40] is used for this purpose. A hybrid, in this context can be defined as a combination or composition of two or more computational techniques which provide greater advantages than the individual techniques and hence improve data analysis. There are two main steps in IHFS: Generating Candidate Feature Sets and Finding the Optimal Feature Set. Four built-in feature selection methods, CfsSubsetEval (CFS) [41], GainRatioAttributeEval (GR), OneRAttributeEval (OneR) and SymmetricalUncertAttributeEval (SU) were applied and the top N features are extracted from each method. These top N features are combined to generate a candidate feature set as shown in Fig. 2. This is repeated for different values of N to get different candidate feature sets. Then the best candidate feature set is selected by evaluating the performance of different classifiers on each candidate set. The classifiers chosen for this were BayesNet, Logistic [42], IB1 [43], NBTree [44] and SVM. This feature selection work is carried out using WEKA, Weikato Environment for Knowledge Analysis, developed by the University of Weikato. Table 1 lists the selected attributes with the notation used to denote each feature throughout this paper and also gives a description of each. All these 6 features have significant role in improving the classification accuracy.

5 Clustering Based Outlier Detection (CBOD)

This section gives a brief account on Clustering Based Outlier Detection (CBOD) which is detailed in our previous work [45] with a generalized framework which enables the ease of employment of this approach for detecting outliers in any application domain. It is specially designed to handle datasets having mixed attributes easily and it can be used to efficiently distinguish normal and anomalous events.

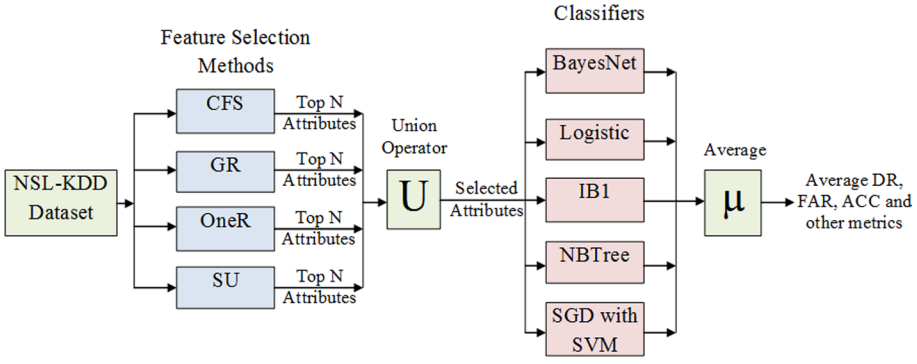

Fig. 2 Block diagram of IHFS

Table 1 Description of selected attributes with notation

Notation	Attribute name	Destription
A1	service	Network service on the destination
A2	flag	Status of the connection
A3	logged_in	Indicates successful login or not
A4	src_bytes	Data bytes from source to destination
A5	dst_bytes	Data bytes from destination to source
A6	srv_serror_rate	% of "SYN" errors (same host connections)

5.1 CBOD Training Phase

Let *NormalTrainSet* denote the set of normal instances taken for training purpose. Among the 1,25,973 records in the training set, 67,343 records were found to indicate normal connections and these records were given as input for the training phase. This phase will form clusters with the help of the categorical attributes and learns the boundary of each cluster with respect to the numerical attributes. The clusters are finally represented with three parameters *cluster_center*, *max_diff* and *max_dist.* The steps involved in CBOD training is depicted in Fig. 3.

The *cluster_center* is the representative point of the cluster and is chosen as the more frequent data point. The parameter *max_dist* is the maximum allowable distance between a data point $P\left(x_1, y_1\right)$ and the *cluster_center* C (x, y). The distance between every data point P and the cluster_center C is calculated as $D = abs\left(x - x_1\right) + abs\left(y - y_1\right)$. The value of max_dist is the maximum allowable difference between src_bytes and dst_bytes and is fixed based on range analysis on the values of D. The training gets over by computing the parameters and saving them in a 114×3 cluster parameter matrix, listing the values of the three parameters for every cluster.

5.2 CBOD Testing Phase

The instances in the testing dataset are read sequentially. Find the matching profile based on the values of the attributes A_1, A_2 and A_3. If the profile is found in the list of trained profile, then go to the next step, else mark the record as an attack. If the profile is found, then check the values of the parameters *cluster_center*, *max_diff* and *max_dist* for that cluster. If cluster_center is (0,0), then it indicates a smaller cluster and hence an outlier. If the cluster_center is non-zero, then check the value of A_6. If it is non-zero, then it indicates there is an error and hence an outlier. Otherwise, compute *diff*, the difference between A_4 and A_5. Also compute *dist*, the distance between the particular instance and the corresponding *cluster_center*. If $diff < max\_diff$ and if $dist < max\_dist$, then it denotes an usual traffic connection, else mark it as an attack. These steps are shown in Fig. 4.

5.3 Performance Analysis of CBOD

The parameters used to assess the efficiency of the proposed method are Detection Rate (DR), False Alarm Rate (FAR) and classification accuracy (ACC). Formulae for all these metrics are given in [25]. The CBOD method was able to yield 97.84% DR, 1.88% FAR

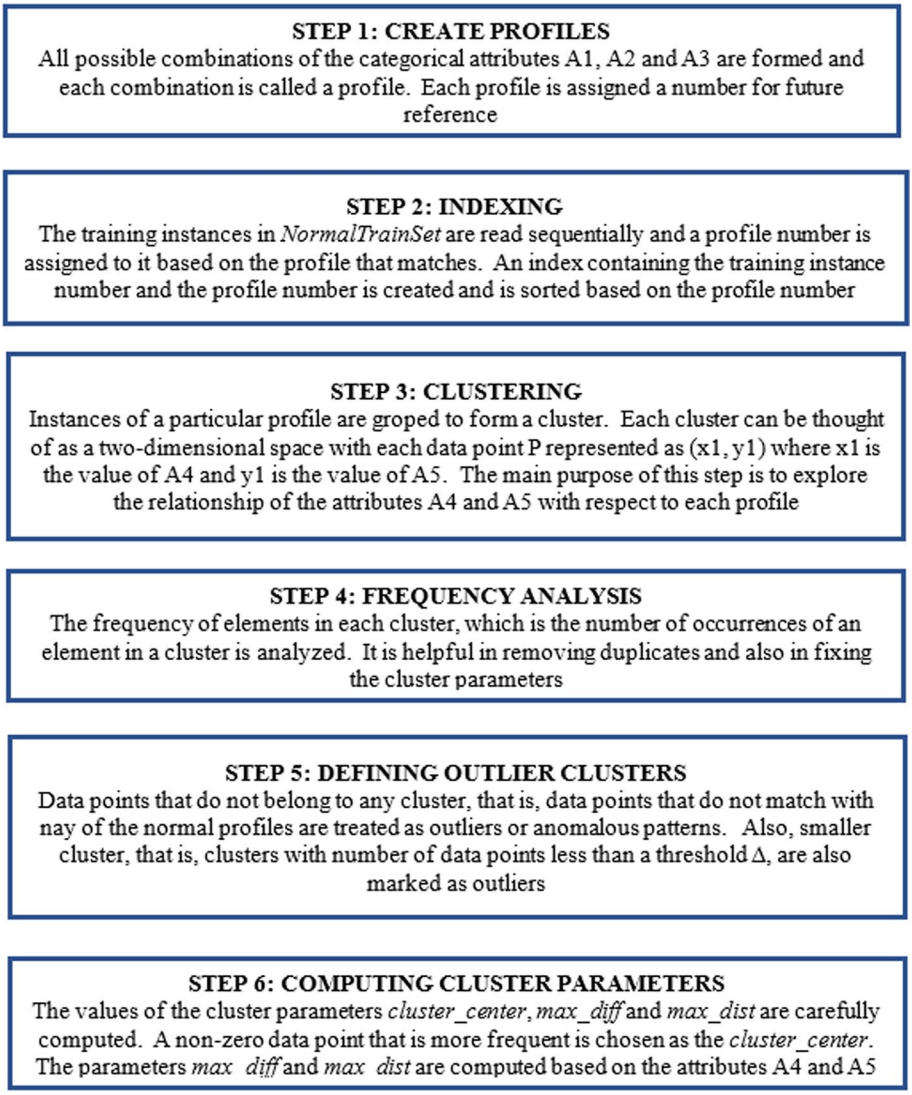

Fig. 3 CBOD Training Phase

and 97.96% ACC. Analysis revealed that CBOD was able to detect 99.81% of DoS attacks, 99.71% of Probing attacks, 90.95% of R2L attacks and 96.50% of U2R attacks. As R2L attacks have a low detection rate, the detection rate of individual R2L attack is analyzed in detail and is presented in Table 2.

From the table, it is very clear that the detection rates of two of the R2L attacks namely *snmpguess* and *snmpgetattack* are 80% and 0% respectively, while all the other R2L attacks got detected almost completely. It is alarming to note that all the *snmpgetattacks* went

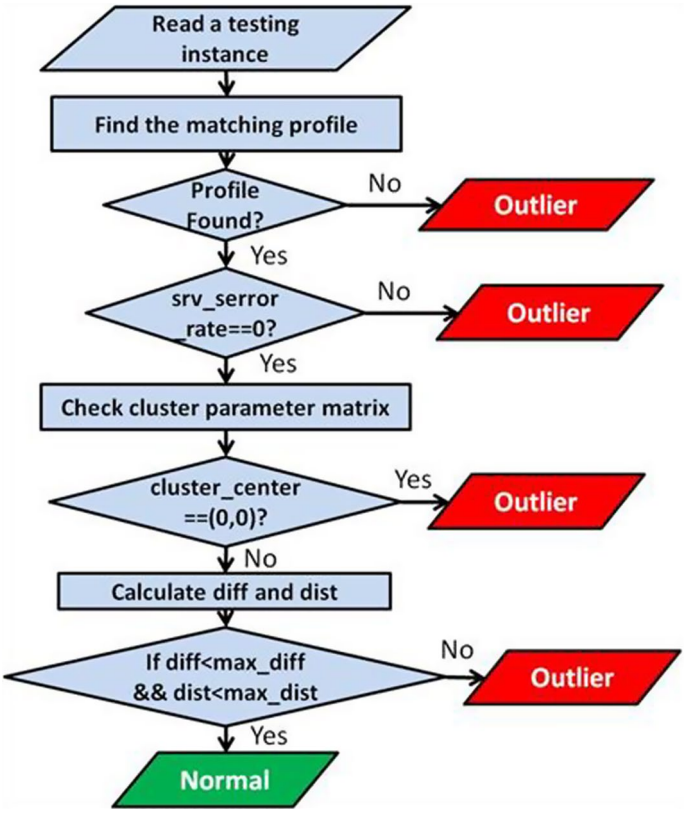

Fig. 4 Steps in CBOD Testing

Table 2 Detection rate of R2l attacks—Cbod

Attack name	Actual present	Detected	Missed	Detection rate %
guess_passwd	1231	1231	0	100
ftp_write	3	3	0	100
Multihop	18	15	3	83.33
imap	1	1	0	100
warezmaster	944	942	2	99.79
phf	2	1	1	50
snmpgetattack	178	0	178	0
snmpguess	331	268	63	80.97
named	17	15	2	88.24
sendmail	14	14	0	100
xlock	9	9	0	100
xsnoop	4	4	0	100

undetected. The reason behind the low DR of R2L attacks is the similarity between *snmpgetattack* and *snmpguess* and normal behavior [46].

5.4 Why Snmpgetattacks went Undetected by CBOD?

Simple Network Management Protocol (SNMP) is used to monitor and manage network devices remotely. Attackers are progressively mishandling network devices which are designed to give public response to SNMP requests over the Internet. Intruders could design requests which seem to have originated from the IP address of the proposed victim. The snmpget command is given to accept data from a host in some remote location by giving the name of the host, authentication information and an object identifier OID. On gaining read-only access all data in Management Information Base (MIB) can be read. MIB is a database used for managing the entities in a communication network managed using the Simple Network Management Protocol (SNMP). After gaining read–write access, many attributes can be modified. Devices that support SNMP can be exploited if the SNMP service is directly revealed to the Internet.

The SNMP community password is like a user id or password that allows access to a router's or device's statistics. By convention, most of the devices get shipped from the factory with the community password set to 'public'. The network managers change the community passwords to customized values in the device setup.

In *snmpguess* attack, the attacker keeps on sending SNMP requests to a router using different passwords until receiving a response from that router indicating that the password is correct. An attacker who has guessed the SNMP community password of a router will then be able to monitor the traffic levels on that router. The attacker monitoring traffic corresponds to the *snmpgetattack*.

The CBOD method was able to detect 80% of *snmpguess* attacks as they were similar to *guess_passwd* attacks. Once *snmpguess* attack is successful, then *snmpget* command can be used to monitor the traffic on the victim without being detected. As *snmpgetattackks* are similar to the normal SNMP connections, all the *snmpgetattacks* were not detected by CBOD.

5.5 Attributes Needed to Detect Snmpgetattacks

On analyzing the *snmpgetattack* and *snmpguess* attack connections it was found that they were of profile 70 with *service* = 'private', *flag* = 'SF' and *logged_in* = 0. In the testing set, there are 1544 instances of profile 70, out of which 840 are normal and 704 are anomalous. The analysis on normal and attack connections of profile 70 revealed that they are almost similar especially with respect to the six attributes selected by IHFS. In many cases the normal and attack connections are exactly the same. Hence it is deduced that the six attributes are not adequate to recognize *snmpgetattack* and *snmpguess* attacks.

In order to exactly identify the attributes that really influence the identification of these attacks, normal and attack instances of profile 70 with all the 41 attributes are analyzed. The normal and attack instances are saved separately as two datasets. For each attribute, the mean and standard deviation are computed for the normal dataset and for the attack dataset. This revealed that most of the attributes are found to be zero-valued and so these attributes are ignored. In addition, for some attributes, the mean and standard deviation are similar for normal and attack connections and so these attributes are ignored. Finally, the attributes *count*, *srv_count*, *dst_host_count* and *dst_host_srv_count* are found to have some difference in normal and attack connections. Hence these four features are added to the list of selected features and the notation for these features is shown is Table 3.

Table 3 Description of additional attributes with notation

Notation	Attribute name	Destription
A7	count	No. of connections to the same host
A8	srv_count	No. of connections to the same host using the same service
A9	dst_host_count	No. of connections having the same destination host
A10	dst_host_srv_count	No. of connections having the same destination host using the same service

6 Proposed Method: Multistage Clustering Based Outlier Detection (MCBOD)

The CBOD method is extended to a Multistage Clustering-based Outlier Detection (MCBOD) approach which is designed specifically to detect the R2L attacks missed by CBOD approach. Even though the normal and attack connections have some variations with respect to the attributes $A_7, A_8, A_9 \& A_{10}$, these variations are not much significant and so it is not straightforward to identify and learn a boundary between normal and attack traffic. This needs an exhaustive and thorough analysis of the dependence of the attributes on one another which is done with the help of MCBOD.

The proposed MCBOD approach groups the dataset into clusters in different stages based on different sets of attributes and has separate training and testing phases. The testing phase closely monitors the normal network traffic by grouping the traffic patterns into multiple stages of clusters and the properties of clusters at different stages are learned. The testing phase checks whether the incoming traffic pattern matches with the properties of the saved clusters. If a match is found, then it is a normal activity; otherwise an intrusion.

6.1 MCBOD Training Phase

The main work in MCBOD training phase is to create three stages of clusters. The key idea behind dividing a cluster into sub-clusters is to analyze the frequency of each data point in a cluster and to create sub-clusters to represent each frequent data point and one sub-cluster to represent all the infrequent data points. Thus, if there are f frequent data points in a cluster, then $f + 1$ sub-clusters will be formed. The procedure for creating clusters of multiple stages is explained in detail in the following sub sections.

i. Stage I Clustering

The *NormalTrainSet* containing the normal instances of the NSL-KDD training set is given as input to the MCBOD training phase. Like CBOD method, every instance in the NormalTrainSet is associated with a profile number based on the attributes A_1, A_2 and A_3 and Stage I clusters are formed based on the corresponding profiles. The 114 possible stage I clusters are denoted from $C1_1$ to $C1_{114}$ and the data points in these clusters represent the values of A_4 and A_5. Except $C1_{70}$, all the other clusters of stage I need not be divided into sub-clusters and the training phase is the same as that of CBOD to learn the relationship between A_4 and A_5 for each profile. As *snmpgetattack* and *snmpguess* attacks occur only in profile 70, $C1_{70}$ needs more analysis to be done with respect to the additional attributes A_7, A_8, A_9 and A_{10} and hence it is sub-clustered into two more levels.

ii. Stage II Clustering

$C1_{70}$, the stage I cluster of profile 70, will be having data points in the form (x_1, x_2) with x_1 and x_2 representing the values of the attributes A_4 and A_5 respectively. The cluster $C1_{70}$ has 817 data points. A frequency analysis is done on this cluster which tabulates every distinct data point and its number of occurrences. Surprisingly, the frequency analysis revealed that there are many overlapping data points, that is, the same data points occurred many times. The most frequent data points are (105,0), (105,105), (105,145), (105,146), (105,147), (28,0), (1,0) and (1,1). There were few data points with other values too.

As there are 8 frequent data points, $C1_{70}$ is divided into 9 sub-clusters. Eight sub-clusters correspond to the 8 frequent data points and the 9^{th} sub-cluster corresponds to all the infrequent data points. Thus, nine clusters of stage II are formed and are denoted as $C2_1$ to $C2_9$ as given in Table 4.

The reason behind forming sub-clusters is that each sub-cluster may represent a specific behavior with respect to the additional attributes A_7, A_8, A_9 and A_{10}. Based on the description of these attributes given in Table 3, it is clear that the attributes A_7 and A_8 are related to each other and the attributes A_9 and A_{10} are related to each other. The attributes A_7 and A 8 were chosen to be analyzed in this step. Hence, each cluster of stage II will be having data points in the form (y_1, y_2) where y_1 and y_2 represent the values of the attributes A_7 and A_8 respectively.

Thus, each instance of $C1_{70}$ is assigned to one of the clusters $C2_1$ to $C2_9$ based on the values of the attributes A_4 and A_5 and the values of the attributes A_7 and A_8 are stored in them. In order to avoid repetitions it is ensured that each data point is stored only once.

iii. Stage III Clustering

Each stage II cluster is divided into sub-clusters with the same kind of frequency analysis done in Stage II clustering to form clusters of stage III as shown in Table 7. Each cluster of stage III will be having data points in the form (z_1, z_2) where z_1 and z_2 represent the values of the attributes A_9 and A_{10} respectively.

From Table 7 it can be seen that (1,1), (2,2), (3,3) and (4,4) are the four most frequent data points. The clusters $C2_1$, $C2_4$ and $C2_5$ are divided into five sub-clusters each, with four clusters representing the four frequent data points and one cluster representing all the infrequent data points. The cluster $C2_2$ had only three frequent data points (1, 1), (2, 2), (3, 3) and so it was divided into four sub-clusters. All the elements in cluster $C2_3$ represented any one of the data points (1, 1), (2, 2), (3, 3), (4, 4) and so it was divided into four sub-clusters. The cluster $C2_6$ had (2, 2) as a frequent data point and some infrequent data points which

Table 4 Stage II clusters

Notation	Data point represented
$C2_1$	(105, 0)
$C2_2$	(105, 105)
$C2_3$	(105, 145)
$C2_4$	(105, 146)
$C2_5$	(105, 147)
$C2_6$	(28, 0)
$C2_7$	(1, 0)
$C2_8$	(1, 1)
$C2_9$	All other values

Table 5 Stage III clusters

Stage II cluster	Data points that represent stage III clusters	Number of sub-clusters
$C2_1$	(1, 1), (2, 2), (3, 3), (4, 4), other	5
$C2_2$	(1, 1), (2, 2), (3, 3), other	4
$C2_3$	(1, 1), (2, 2), (3, 3), (4, 4)	4
$C2_4$	(1, 1), (2, 2), (3, 3), (4, 4), other	5
$C2_5$	(1, 1), (2, 2), (3, 3), (4, 4), other	5
$C2_6$	(2, 2), other	2
$C2_7$	other	1
$C2_8$	other	1
$C2_9$	other	1

resulted in two sub-clusters. The clusters $C2_7$, $C2_8$ and $C2_9$ do not have any frequent data point and hence they have only one sub-cluster representing all the infrequent data points.

A general observation from Table 5 is that, the attributes A_7 and A_8 take up the same values except in a few cases and the values of the attributes A_7 and A_8 are generally small. The attribute A_7, 'count', indicates the number of connections to the same host as the current connection in the past 2 sec. The attribute A_8, 'srv_count', indicates the number of

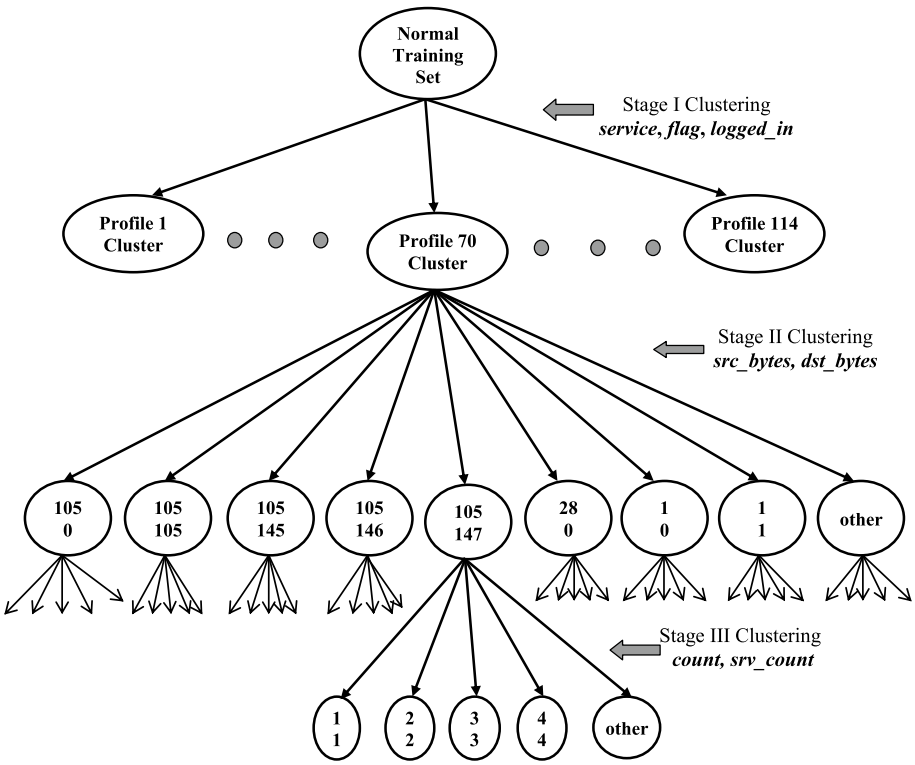

Fig. 5 Multistage Clustering

connections to the same host using the same service as the current connection in the past 2 sec. If A_7 and A_8 are same, it means in the past 2 s, all the connections to that host used the same service. The smaller values for the attributes A_7 and A_8 indicate that only a few connections are established to the same host in a very short period of time.

After finalizing the number of sub-clusters to be formed for each stage II cluster, the next step is to assign the training instances represented by the data points of each stage to a stage III cluster based on the values of the attributes A_7 and A_8. After assigning a training instance to a stage II cluster, the corresponding values of the attributes A_9 and A_{10} are stored in the sub-cluster without repetitions. The multistage clustering algorithm is presented in Algorithm 1 and is pictorially represented in Fig. 5.

Algorithm 1: MultistageClustering

Input: Stage I cluster $C1_{70}$ that represents the profile A_1='private', A_2='SF' & A_3=0
Output: Clusters of stages II and II denoted as $C2$ & $C3$
1: **Let** $P(x, y)$ represent a data point in $C1_{70}$
2: **Let** n represent the number of elements in $C1_{70}$
3: Find the number of occurrences of each data point in $C1_{70}$
4: Let j represent the number of frequent data points in $C1_{70}$
5: Create $j + 1$ stage II clusters $C2$, with j clusters to represent each frequent data point and one cluster to represent all the infrequent data points
6: **for** every instance $C1_{70}[i]$, $i \leftarrow 1$ *to* n **do**
7: Based on A_5, A_6 find the fitting cluster $C2_k$, $k = 1, 2, 3, \ldots, j + 1$
8: Store the values of A_7, A_8 in the corresponding cluster $C2_k$ if it is not already present
9: **end for**
10: **for** each stage II cluster $C2_k$, $k \leftarrow 1$ *to* $j - 1$ **do**
11: **Let** x represent the number of elements in $C2_k$
12: Find the no of occurrences of each data point in $C2_k$
13: Let m represent the no of frequent data points in $C2_k$
14: Create $m + 1$ stage III clusters $C3$, with m clusters for each frequent data point and one cluster for all the infrequent data points
15: **for** every instance $C2_k[i]$, $i \leftarrow 1$ *to* x **do**
16: Based on A_7, A_8 find the fitting cluster $C3_y$, $y \leftarrow 1$ *to* $m + 1$
17: Store the values of A_9, A_{10} in the corresponding cluster $C3_y$ if it is not already present
18: **end for**
19: **end for**

6.2 MCBOD Testing Phase

Once formation of clusters of different stages is over, the testing phase begins. All the instances in the testing set are processed sequentially. First, the attributes $A_1, A_2 \& A_3$ are checked to find the corresponding profile. If no such profile exists, then it is an intrusion. If the profile is found, then check the value of the attribute A_6. If $A_6 \neq 0$, it is an outlier. Otherwise, check whether it is profile 70 or not. If the instance is not of profile 70, then proceed with CBOD testing to identify whether it is an intrusion or not. If the testing instance is of profile 70, then find the corresponding stage II cluster based on the attributes $A_4 \& A_5$. Next, pick the matching stage III cluster, C on the basis of the attributes $A_7 \& A_8$. Let (x, y) represent the values of the attributes $A_9 \& A_{10}$. Check whether (x, y) matches with any of the data points in C. If a match is found, then the instance is classified as a normal event.

Otherwise, it is classified as an intrusion. This procedure is shown pictorially in Fig. 6 for better understanding.

6.3 Experimental Setup

The experiments were carried out on a desktop with Intel Core i7-2600 processor @ 3.40 GHz with 2 GB RAM running Windows 7 Professional. WEKA is used for feature selection. The proposed MCBOD method for classifying normal and intrusive traffic connections were implemented using MATLAB (MATrixLABoratory) R2011.

Two methods are commonly used for testing the performance of any classifier – Cross validation and Holdout. In all our experiments, we used Holdout method and not cross validation as the NSL-KDD dataset comes with separate training and testing sets.

6.4 Performance Analysis of MCBOD

The time taken for training and testing phases is around 375 s and 40 s respectively. The proposed MCBOD method yielded 99.52% DR, 1.15% FAR and 99.22% ACC. The MCBOD approach is also a two-class classifier. To demonstrate the effectiveness of MCBOD approach in identifying all types of attacks, Table 6 shows the split-up of DR for the four classes of attacks.

On comparing Tables 3 and 6, it can be inferred that there is a significant increase in the DR of R2L attacks because of the specially designed MCBOD approach to identify *snmpgetattack* and *snmpguess* attacks. It is also interesting to note that the FAR is significantly reduced to 1.15%. This was possible by the thorough analysis of instances of Profile 70 by including four features and by splitting them into different sub-classes. Table 7 shows the DR of the R2L attacks. All the *snmpguess* attacks were detected by MCBOD. But, 27 *snmpgetattacks* were missed.

The attacks of all types that went undetected are listed in Table 8. From the table it can be inferred that 12 known attacks and 50 unknown attacks went undetected.

To prove the effectiveness of the proposed approach, it is compared with six recent anomaly detection methods in the literature whose results were reported to be obtained on the same data set and is tabulated in Table 9. From the table, it is clear that the proposed MCBOD method has produced the highest DR compared to the existing methods. It can be seen that two methods MOAD and TVCPSO have shown a very low FAR, but their DR is also less, which is not appealing.

To further substantiate the supremacy of the proposed method, its performance is analyzed with respect to measures like sensitivity, specificity, accuracy and precision. Sensitivity, which is also called true positive rate or detection rate or recall, signifies how efficiently intrusions are identified. On the other hand, specificity, which is also called true negative rate, denotes how efficiently the normal instances are identified. Accuracy shows how well normal events are classified as normal and anomalous events are classified as intrusions. Precision represents the fraction of alarms that are true. If FAR increases, precision decreases. Some methods may yield a high DR while suffering from high FAR. On the other hand, some methods may reduce false alarms very well while having low DR. The better method can be identified using F-Measure, which reveals the balance between precision and recall which in turn conveys the tradeoff between DR and FAR.

Figure 7 graphically represents the performance analysis. From the graph, it is evident that the proposed method has the highest values for sensitivity, accuracy and F-Measure.

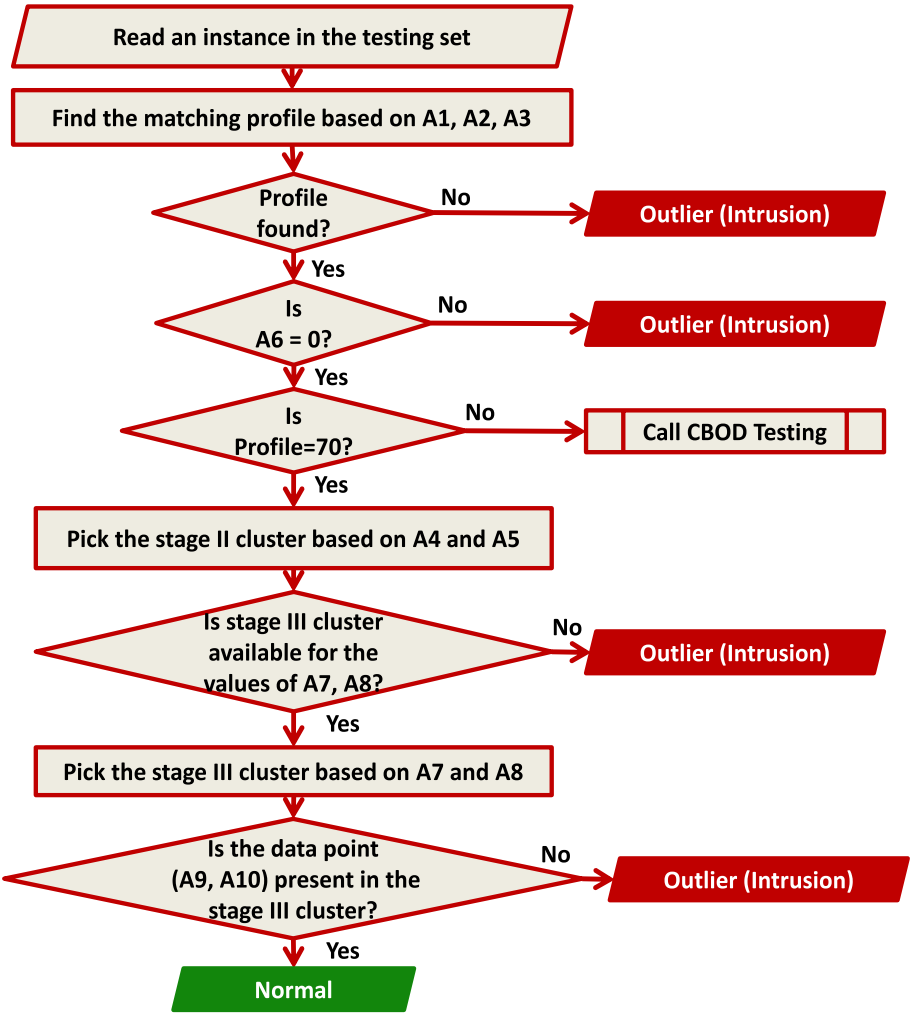

Fig. 6 Steps in MCBOD Testing

The existing methods MOAD and TVCPSO have slightly higher specificity and precision, because of the less false alarms produced. But they show less F-Measure which is not appreciable. F-Measure, the tradeoff between DR and FAR, is the highest for the proposed method which confirms its supremacy. Thus, the proposed method has shown winning results by reasonably reducing the FAR while maintaining a very high DR.

7 Conclusion & Future Scope

This paper addresses the challenge of reducing FAR in anomaly-based IDS while maintaining a high DR. The proposed multistage clustering approach, an extension of CBOD approach, was specifically designed to handle R2L attacks. The design of MCBOD depends mainly on clustering and frequency analysis. The additional levels of clustering

Table 6 Detection rate—MCBOD

Attack class	Detection rate (%)
Dos attacks	99.81
Probe attacks	99.75
R2L attacks	98.73
U2R attacks	96.50

Table 7 Detection rate of R2L attacks—MCBOD

Attack name	Actual present	Detected	Missed	Detection rate %
guess_passwd	1231	1231	0	100
ftp_write	3	3	0	100
Multihop	18	15	3	83.33
imap	1	1	0	100
warezmaster	944	942	2	99.79
phf	2	1	1	50
snmpgetattack	178	151	27	84.83
snmpguess	331	331	0	100
named	17	15	2	88.24
sendmail	14	14	0	100
xlock	9	9	0	100
xsnoop	4	4	0	100

Table 8 Attacks missed by MCBOD

Attack type	Attack name	Actual present	Detected	Missed
Known attacks				
Probe	satan	735	734	1
R2L	multihop	18	15	3
R2L	warezmaster	944	942	2
R2L	phf	2	1	1
U2R	rootkit	13	8	5
Total		1712	1700	12
Unknown attacks				
DoS	apache2	737	733	4
DoS	mailbomb	293	283	10
Probe	saint	319	314	5
R2L	snmpgetattack	178	151	27
R2L	named	17	15	2
U2R	xterm	13	12	1
U2R	httptunnel	133	132	1
Total		1690	1640	500

paved way to closely represent the properties of the normal connections and to arrive at a clear boundary between normal and intrusive behaviour. The effectiveness of the proposed method is substantiated with careful experiments and is found to be superior to the existing

Table 9 Comparison of DR & FAR with existing methods

Method	DR (%)	FAR (%)
PCA & Bayesian SOM [17]	97	7
K-means clustering & SVM [18]	96.26	3.7
OS-ELM [19]	99.01	1.74
MOAD [20]	97.23	0.98
TVCPSO [21]	97.03	0.87
IG-BAL-SVM [22]	99.15	1.9
ENSEMBLE [31]	99.10	1.23
TSE-IDS [32]	94.60	8.10
DNN [33]	92	8
ANFIS [34]	95.80	3.45
The proposed method MCBOD	99.51	1.15

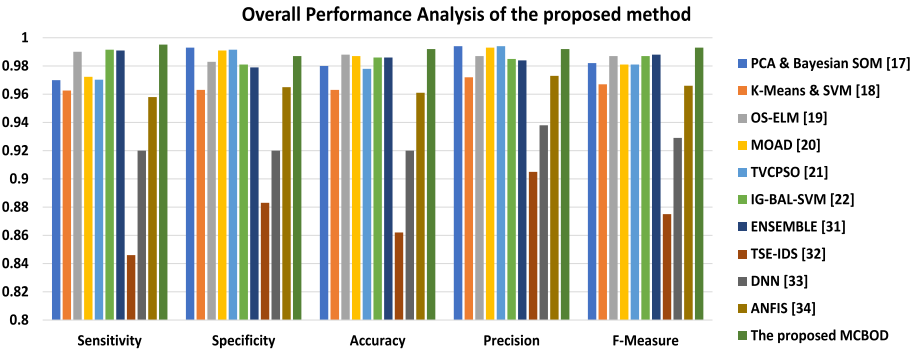

Fig. 7 Performance Analysis of the proposed MCBOD method with existing methods

methods in the literature by projecting a good balance between DR and FAR. Obviously, the proposed method fills the gaps identified in the earlier research.

This research was targeted towards finding whether a traffic connection is intrusion or not, which is a two-class problem. This can be extended to a five-class problem to detect the exact type of intrusion, i.e., to detect whether it is a normal connection or DoS/Probe/R2L/U2R attack. In addition, as a future work, the method proposed in this paper can be applied in real time. This can be done by first capturing live network traffic using the packet analyzer tcpdump and then by extracting the necessary features and by applying the proposed MCBOD method on the extracted features. The findings of this research work are significant and will surely guide future researchers in fruitful directions.

Author's Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by JRB, MN, DSI and DSP. All authors read and approved the manuscript.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data Availability The datasets analysed during the current study are available in https://web.archive.org/web/20150205070216/http://nsl.cs.unb.ca/NSL-KDD/ and also in the KAGGLE repository, https://www.kaggle.com/hassan06/nslkdd.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

References

1. Yeung D. Y., Chow C. (2002). "Parzen-window network intrusion detectors", In: Object recognition supported by user interaction for service robots, IEEE, vol. 4, pp. 385–388
2. Buczak, A. L., & Guven, E. (2015). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials, 18*(2), 1153–1176.
3. Ahmad I., Abdullah A. B., Alghamdi A. S., (2010). "Remote to Local attack detection using supervised neural network", In IEEE International Conference for Internet Technology and Secured Transactions, pp. 1–6.
4. Denning, D. E. (1987). An intrusion-detection model. *IEEE Transactions on Software Engineering, 2*, 222–232.
5. Lazarevic A., Ertoz L., Kumar V., Ozgur A., Srivastava J. (2003). "A comparative study of anomaly detection schemes in network intrusion detection", In Proceedings of the 2003 SIAM international conference on data mining, Society for Industrial and Applied Mathematics, pp. 25–36.
6. Tavallaee, M., Stakhanova, N., & Ghorbani, A. A. (2010). "Toward credible evaluation of anomaly-based intrusion-detection methods." *IEEE Transactions on Systems, Man and Cybernetics Part C (Applications and Reviews), 40*(5), 516–524.
7. Gogoi, P., Borah, B., & Bhattacharyya, D. K. (2010). Anomaly detection analysis of intrusion data using supervised & unsupervised approach. *Journal of Convergence Information Technology, 5*(1), 95–110.
8. Bhuyan M. H., Bhattacharyya D. K., Kalita J. K. (2011). "NADO: Network anomaly detection using outlier approach", In Proceedings of the International Conference on Communication, Computing & Security, ACM, pp. 531–536, 2011.
9. Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network anomaly detection: methods, systems and tools. *IEEE Communications Surveys & Tutorials, 16*(1), 303–336.
10. Nalini, M., & Anbu, S. (2014). Anomaly detection via eliminating data redundancy and rectifying data error in uncertain data streams. *International Journal of Applied Engineering Research, 9*(24), 30795–30812.
11. Nalini M., Priyadarsini U. (2019). "To improve the performance of wireless networks for resizing the buffer", In Proceedings of the 1st International Conference on Innovations in Information and Communication Technology, pp. 1–5, IEEE, 2019.
12. Nalini, M., & Chakram, A. (2019). "Digital risk management for data attacks against state evaluation." *International Journal of Innovative Technology and Exploring Engineering, 8*, 197–201.
13. Lee W., Stolfo S. (1998). "Data mining approaches for intrusion detection", In *Proceedings of USENIX Security,* pp. 79–93.
14. Boudia, M. A., Hamou, R. M., & Amine, A. (2017). A new meta-heuristics for intrusion detection system inspired from the protection system of social bees. *International Journal of Information Security and Privacy (IJISP), 11*(1), 18–34.
15. Arul R., Moorthy R. S., Bashir A. K., (2019) "Ensemble learning mechanisms for threat detection: A Survey", In *Machine Learning and Cognitive Science Applications in Cyber Security,* IGI Global, pp. 240–281.
16. Blazquez-Gracia A., Conde A., Mori U., Lozano J. A. "A review on outlier/anomaly detection in time series data" arXiv preprint arXiv:2002.04236 (2020).
17. De la Hoz, E., De la Hoz, E., Ortiz, A., Ortega, J., & Prie, B. (2015). PCA filtering and probabilistic SOM for network anomaly detection. *Neurocomputing, 164*, 71–81.
18. Mohamad Tahir H., Hasan W., Md Said A., Zakaria N. H., Katuk N., Kabir N. F., Omar M. H., Ghazali O., & Yahaya N. I., (2015). "Hybrid machine learning technique for intrusion detection system", In *Proc. ICOCI,* pp. 464–472.

19. Singh, R., Kumar, H., & Singla, R. K. (2015). An intrusion detection system using network traffic profiling and online sequential extreme learning machine. *Expert Systems with Applications, 42*(22), 8609–8624.
20. Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2016). A multi-step outlier-based anomaly detection approach to network-wide traffic. *Information Science, 348*, 243–271.
21. Bamakan, S. M. H., Wang, H., Yingjie, T., & Shi, Y. (2016). An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization. *Neurocomputing, 199*, 90–102.
22. Enache A. C., Sgarciu V., (2015) "Anomaly intrusions detection based on support vector machines with an improved bat algorithm", In *Proc. CSCS*, pp. 317–321.
23. Hassan, D. (2017). Cost-sensitive access control for detecting remote to local (R2L) and user to root (U2R) attacks. *International Journal of Computer Trends and Technology (IJCTT), 43*(2), 124–129.
24. Paliwal, S., & Gupta, R. (2012). Denial-of-service, probing & remote to user (R2L) attack detection using genetic algorithm. *International Journal of Computer Applications, 60*(19), 57–62.
25. Revathi, S., & Malathi, A. (2014). Effective analysis on remote to user (R2L) attacks using random forest algorithm. *International Journal of Engineering Sciences & Research Technology, 3*(5), 317–319.
26. Jeya, P. G., Ravichandran, M., & Ravichandran, C. S. (2012). Efficient classifier for R2L and U2R attacks. *International Journal of Computer Applications, 45*(21), 28–32.
27. Nguyen V.Q., Nguyen V. H., Le-Khac N. A., Cao V. L., (2020) "Clustering-Based Deep Autoencoders for Network Anomaly Detection", in *International Conference on Future Data and Security Engineering,* pp. 290–303, Springer, Cham.
28. Pu, G., Wang, L., Shen, J., & Dong, F. (2020). A hybrid unsupervised clustering-based anomaly detection method. *Tsinghua Science and Technology, 26*(2), 146–153.
29. Li, M., Kashef, R., & Ibrahim, A. (2020). Multi-level clustering-based outlier's detection (MCOD) using self-organizing maps. *Big Data and Cognitive Computing, 4*(4), 24.
30. Elmogy, A., Rizk, H., & Sarhan, A. M. (2021). OFCOD: On the fly clustering based outlier detection framework. *Data, 6*(1), 1–20.
31. Aljawarneh, S., Aldwairi, M., & Yassein, M. B. (2018). Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *Journal of Computational Science, 25*, 152–160.
32. Tama, B. A., Comuzzi, M., & Rhee, K. H. (2019). TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system. *IEEE Access, 7*, 94497–94507.
33. Mohammed, B., & Gbashi, E. K. (2021). Intrusion detection system for NSL-KDD dataset based on deep learning and recursive feature eimination. *Engineering and Technology Journal, 39*(7), 1069–1079.
34. Manimurugan, S., Majdi, A. Q., Mohammed, M., Narmatha, C., & Varatharajan, R. (2020). Intrusion detection in networks using crow search optimization algorithm with adaptive neuro-fuzzy inference system. *Microprocessors and Microsystems, 79*, 103261.
35. Gogoi, P., Bhattacharyya, D. K., Borah, B., & Kalita, J. K. (2011). A survey of outlier detection methods in network anomaly identification. *The Computer Journal, 54*(4), 570–588.
36. Beulah, J. R., & Punithavathani, D. S. (2015). Outlier detection methods for identifying network intrusions—A survey. *International Journal of Applied Engineering Research, 10*(19), 40488–40496.
37. Hassani M., Seidl T., (2011) "Network intrusion detection using a secure ranking of hidden outliers", In *Proceedings of the Seventh International Computing Conference in Arabic,* pp. 1–10.
38. NSL-KDD Dataset [Online] Available: https://web.archive.org/web/20150205070216/http://nsl.cs.unb.ca/NSL-KDD/
39. Hasan, M. A. M., Nasser, M., Ahmad, S., & Molla, K. I. (2016). Feature selection for intrusion detection using random forest. *Journal of Information Security, 7*(3), 129–140.
40. Beulah, J. R., & Punithavathani, D. S. (2018). A hybrid feature selection method for improved detection of wired/wireless network intrusions. *Wireless Personal Communications, 98*(2), 1853–1869.
41. Hall M.A. (1999) "Correlation-based feature selection for machine learning" Ph.D. dissertation, Dept. of Computer Science, The University of Waikato, Hamilton.
42. Le Cessie, S., & Van Houwelingen, J. C. (1992). Ridge estimators in logistic regression. *Applied Statistics, 41*(1), 191–201.
43. Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning, 6*(1), 37–66.
44. Kohavi R. (1996) "Scaling up the accuracy of naïve-Bayes classifiers: A decision tree hybrid", In *Proc. International Conference on KDD,* pp. 202–207.

45. Beulah, J. R., & Shalini Punithavathani, D. S. (2020). An efficient mixed attribute outlier detection method for identifying network intrusions. *International Journal of Information Security and Privacy (IJISP), 14*(3), 115–133.
46. Kemiche M., Beghdad R. (2014). "CAC-UA: A communicating ant for clustering to detect unknown attacks", In *Proceedings of Science and Information Conference*, IEEE, pp. 515–522,

Dr. J. Rene Beulah is an Assistant Professor in the Department of Computing Technologies, SRM Institute of Science & Technology, Kattankulathur, India. She received her B.E. in Computer Science and Engineering from Manonmaniam Sundaranar University, Tirunelveli, India in 2004 and M.E. in Computer Science and Engineering from Anna University, Chennai, India in 2006. She was awarded Ph.D. in the Faculty of Information and Communication Engineering from Anna University, Chennai in 2018. She has 8 years of teaching and 4 years of research experience. Her research interest includes Network Security and Data Mining.

Dr. M. Nalini is an Assistant Professor in the Department of Computer Science and Engineering, Saveetha School of Engineering, Thandalam, Chennai, India. She received her B.E. in Computer Science and Engineering from Anna University, Chennai in 2010 and M.Tech. in Computer Science and Engineering from B.S.Abdur Rahman Crescent Institute of Science & Technology, Chennai, India in 2012. She was awarded Ph.D. in Computer Science and Engineering from St. Peter's Institute of Higher Education and Research, Chennai, India in 2018. She has 8 years teaching experience. Her research interests include Data Mining, Big Data Analytics and Networking. She has published many articles in reputed Journals.

Dr. D. Shiny Irene is an Assistant Professor in the Department of Computing Technologies, SRM Institute of Science & Technology, Kattankulathur, India. She received her B.E. in Computer Science and Engineering from Anna University, Chennai, India in 2011 and M.E. in Computer Science and Engineering from Anna University, Chennai, India in 2013. She was awarded Ph.D. in the Faculty of Information and Communication Engineering from Anna University, Chennai in 2021. She has 8 years teaching experience. Her area of interests include Data Mining, Machine Learning, Big Data Analytics and Mobile Computing. She has published many articles in reputed Journals.

Dr. D. Shalini Punithavathani was the Principal of Government College of Engineering, Tirunelveli, India. She received her B.Sc. in 1979 from Sarah Tucker College, affiliated to Madurai Kamarajar University, India, B.Tech. in Electronics in 1982 from Madras Institute of Technology, affiliated to Anna University, Chennai, India and M.E. in Computer Science and Engineering in 1990 from Government College of Technology, affiliated to Bharathiar University, Coimbatore, India. She got her Ph.D. entitled "Study and Implementation of IPv4 to IPv6 translation techniques" in 2010 from Anna University, Chennai, India. She has 34 years of academic experience. Her research interests include Computer Networks, Mobile Computing, Network Security and Data Mining. She has published many articles in International Journals.

Authors and Affiliations

J. Rene Beulah[1] · M. Nalini[2] · D. Shiny Irene[1] · D. Shalini Punithavathani[3]

M. Nalini
nalini.tptwin@gmail.com

D. Shiny Irene
dshinyirene@gmail.com

D. Shalini Punithavathani
shalini329@gmail.com

1 Department of Computing Technologies, College of Engineering and Technology, Faculty of Engineering and Technology, SRM Institute of Science and Technology, Kattankulathur, Chennai, Tamilnadu, India

2 Department of Computer Science and Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, India

3 Department of Computer Science and Engineering, Government College of Engineering, Tirunelveli, India