Sādhanā (2020) 45:109
https://doi.org/10.1007/s12046-020-1308-5

Particle swarm optimization and feature selection for intrusion detection system

NILESH KUNHARE, RITU TIWARI and JOYDIP DHAR*

Atal Bihari Vajpayee Indian Institute of Information Technology and Management, Gwalior, Gwalior, India
e-mail: nilesh954@gmail.com; tiwariritu2@gmail.com; jdhar@iiitm.ac.in

MS received 5 September 2019; revised 13 January 2020; accepted 15 January 2020; published online 7 May 2020

Abstract. The network traffic in the intrusion detection system (IDS) has unpredictable behaviour due to the high computational power. The complexity of the system increases; thus, it is required to investigate the enormous number of features. However, the features that are inappropriate and (or) have some noisy data severely affect the performance of the IDSs. In this study, we have performed feature selection (FS) through a random forest algorithm for reducing irrelevant attributes. It makes the underlying task of intrusion detection effective and efficient. Later, a comparative study is carried through applying different classifiers, e.g., *k* Nearest Neighbour (*k*-NN), Support Vector Machine (SVM), Logistic Regression (LR), decision tree (DT) and Naive Bayes (NB) for measuring the different IDS metrics. The particle swarm optimization (PSO) algorithm was applied on the selective features of the NSL-KDD dataset, which cut down the false alarm rate and enhanced the detection rate and the accuracy of the IDS as compared with the mentioned state-of-the-art classifiers. This study includes the accuracy, precision, false-positive rate and the detection rate as performance metrics for the IDSs. The experimental results show low computational complexity, 99.32% efficiency and 99.26% detection rate on the selected features (=10) out of a complete set (= 41).

Keywords. Particle swarm optimization; feature selection; machine learning classifiers; intrusion detection system.

1. Introduction

The intrusion detection system (IDS) has become a powerful tool that monitors malicious activities and triggers alerts to detect suspicious attacks. The intrusions make the system unpredictable for network traffic because of its nonlinear behaviour [1, 2]. The reason behind the requirement of an IDS is that the security principles: confidentiality, integrity and availability, are compromised due to intrusions and (or) attacks like spoofing, traffic analysis, cyber-attacks and other harmful vulnerabilities [3]. The IDS is partitioned into two broad categories: signature-based and anomaly-based IDSs [4]. The signature-based system is further referred to as misuse-based detection. This approach compares the signature of the recognized malicious activities and triggers alert whenever the match is found. Hence, these types of systems can diagnose perceived attacks with a low false alarm rate [5]. Anomaly-based systems are competent to encounter zero-day attacks. This approach observes the pattern of the systems; whenever any system deviates from the regular pattern, it triggers alert. This approach undergoes a high false-positive rate (FPR) [6, 7]. The IDS can be further divided into two parts: host-based (HIDS) and network-based (NIDS). The HIDS monitors the individual hosts and raises the alerts based on the local host system calls, log files, application logs and other host activities. However, NIDS monitors all the traffic passing through the entire network system; whenever any malicious activity matches with known attacks in the network, alert is raised and sent to the administrator to take appropriate actions. With the increase in the number of features, the complexity of the system increases. Hence, it is complicated for the IDS to examine the large volume of data. The selection of useful and critical features is essential for the detection of intrusions in the field of information security. In an attempt to make an effective and adequate IDS, there is a requirement for the identification of significant features before pre-processing. Identification of the valuable features is complicated as the dataset is expressed by various relevant, irrelevant and extravagant features, which increase the computation complexity for the analysis of intrusions [8–10]. The fundamental principle of the feature selection (FS) is to improve the quality and performance of the predictor. This research study proposes the optimal selection of features for elaborating the performance of the IDSs having low computational complexity because of the classification of a set of simplified features. Hence, FS is used to enhance the

*For correspondence

performance of classifiers and scale down the attributes that are not relevant before pre-processing. FS is performed on the NSL-KDD dataset [11], which is a modified version of the KDD Cup 99 dataset [12], and the proposed method results in higher accuracy in detection of intrusions and also cuts down the false alarm rate because of the use of a reduced number of features.

2. Description of NSL-KDD dataset

The NSL-KDD is a modified variant of KDD Cup 99 dataset. The dataset has 41 attributes, partitioned into 5 classes, which are normal and 4 attack groups, and they are described later. The 42nd attribute is the class attribute, which contains information about these groups; this attribute has positive or negative instances.

DoS (Denial of Service) includes the definition of attacks where the services of authentic users are restricted. Examples: smurf, teardrop, SYN flooding and neptune.

U2R (User to Root) includes the definitions where the attacker controls the local machines by exploiting vulnerabilities over it. Examples: rootkit, spy, buffer-overflow and SQL attacks.

R2L (Root to Local) includes the definition of attack where the attacker tries to yield access over the remote machine in an unauthorized way. Examples: warezmaster, Imap, multihope and spy.

Probe attack includes the definition where the attacker gathers the information about the network by traffic analysis. Examples: port-scan, satan, ping-sweep and nmap.

The attributes of the dataset are further divided into four labels [13, 14].

Basic: These attributes have separate transmission control protocol (TCP) connections. The number of attributes that belong to this label is 9 and they are illustrated in table 1.

Domain Knowledge: They are the attributes that are inside the connections. The number of attributes that belong to this label is 13 and they are explained in table 2.

Traffic: This group contains the attributes that are enumerated using windows of 2-s duration. The number of attributes that belong to this label is 9. Attributes under this group label are described in table 3.

Host: This group contains attributes that are represented to evaluate attacks survival for greater than 2 s. The number of attributes that belong to this label is 10. Attributes under this label are described in table 4.

The statistics of the records in KDD Cup 99 and NSl-KDD dataset are presented in figure 1 and 2, respectively. The dataset is divided into four types of attacks for training and testing sets. The training set has 22 attack types, and the testing set has increased 17 attack varieties. Figure 3 presents the frequency distribution of attacks in NSL-KDD dataset.

3. Related work

Ganapathy *et al* [15] proposed the FS algorithm as well as classifier using Support Vector Machine (SVM). The paper also represented the inspection of FS and classification strategies for intrusion detection. Moreover, soft computing techniques were used to highlight the research challenges in intrusion detection. Ahmad and Amin [16] used particle swarm optimization (PSO) algorithm for FS, and PCA for feature transformation. The theoretical method was introduced for the detection of intrusion using SVM classifier on KDD Cup 99 dataset. This research work is further extended using the neural network on NSL-KDD dataset. Franco *et al* [17] presented classification using a Self-Organized Map (SOM) and FS performed using Fisher discriminant rate algorithm using 17 features of the dataset. Eesa *et al* [18] propose cuttlefish optimization algorithm for intrusion detection, where FS is also used, and decision tree (DT) is applied for classification purposes. This method improves true positive rate (TPR) and accuracy, and reduces false alarm rate. Chebrolu *et al* [19] introduced FS using classification and regression tree with a Bayesian network for the effectiveness of performance and detection accuracy in

Table 1. Basic label attribute description of the NSL-KDD dataset.

Name	Description	Type
Duration	Magnitude of the association in seconds	Continuous
Protocol type	The type of protocol (TCP, UDP)	Discrete
Service	Network services occupied by the destinations	Discrete
Flag	Identify the connection’s status error or normal	Discrete
Source byte	Bytes transferred from origin to destination	Continuous
Destination byte	Bytes transferred from destination to origin	Continuous
Land	1 if the association is for the same port/host to/from, 0 contrarily	Discrete
Wrong fragment	Statistics of associations for wrong snippets	Continuous
Urgent	Statistics of urgent packets	Discrete

Table 2. Domain Knowledge label attribute description of the NSL-KDD dataset.

Name	Description	Type
Hot	Statistics of hot benchmarks	Discrete
Number of failed logins	Statistics of incorrect login attempts	Discrete
Logged in	Denotes 1 for logged, 0 contrarily	Discrete
Number of compromised	Statistics of arbitrated conditions	Discrete
Root shell	1 if root shell captured, 0 contrarily	Discrete
Su attempted	1 if command attempted, 0 contrarily	Discrete
Number of root	Statistics of root access	Discrete
Number of file creations	Statistics of activities of file creations	Discrete
Number of shells	Statistics of shell prompts	Discrete
Number of access files	Statistics of access control files activities	Discrete
Number of outbound cmds	Outward-bound information in an FTP session	Discrete
Is host login	1 if host login, 0 contrarily	Discrete
Is guest login	1 if guest login, 0 contrarily	Discrete

Table 3. Traffic label attribute description of the NSL-KDD dataset.

Name	Description	Type
Count	Represents associations with the same host within last 2 s	Discrete
Service count	Represents connections for last 2 s of the same port number	Continuous
Serror rate	Represents synchronous errors by flag bit	Discrete
Service serror rate	Statistics of associations that contain SYN errors	Discrete
Rerror rate	Statistics of associations that contain REJ deviations	Discrete
Service rerror rate	Number of associations that contain REJ errors	Discrete
Same service rate	Number of associations for the equivalent service	Discrete
Different service rate	Number of associations for dissimilar services	Discrete
Service different host rate	Statistics of association of different hosts	Discrete

Table 4. Host label attribute description of the NSL-KDD dataset.

Name	Description	Type
Destination host count	Statistics for destination host	Discrete
Destination host service count	Associations having same port number	Discrete
Destination host same service rate	Statistics of connections having same terminus host	Discrete
Destination host different service rate	Different service rate for terminus host	Discrete
Destination host same source port rate	Port numbers for the attribute destination host service count	Discrete
Destination host service different host rate	Different host rate for terminus host	Discrete
Destination host serror rate	Number of activated flags for the attribute destination host count	Discrete
Destination host service serror rate	Service serror rate for terminus host	Discrete
Destination host rerror rate	Rerror rate for terminus host	Discrete
Destination host service rerror rate	Service serror rate for terminus host	Discrete

the IDS on KDD Cup 99 dataset. Zhang *et al* [20] presented methods with a rough set using genetic algorithm (GA) [21] for classification rules. The algorithms used for machine learning and data mining, including SOM, DT, *K* Means and SVM, have been used for the extensive analysis of NSL-KDD and KDD Cup 99 datasets for the exhaustive study [22]. Tsai *et al* [23] discuss several machine learning techniques used for the identification of intrusions in the system. The paper describes various classifiers, including single, multiple and ensemble classifiers. Modi and Patel [24] deployed IDS on cloud computing using different classifiers DT, Bayesian and Associative. The framework

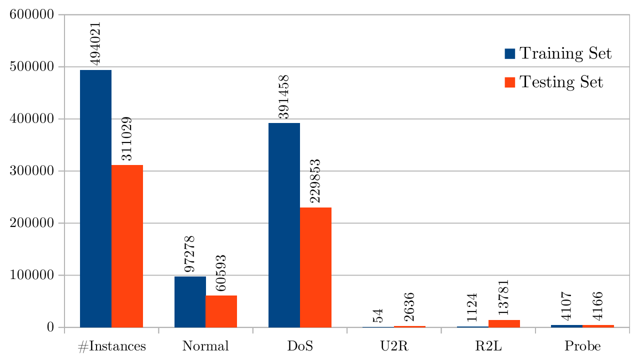

Figure 1. KDD dataset distribution.

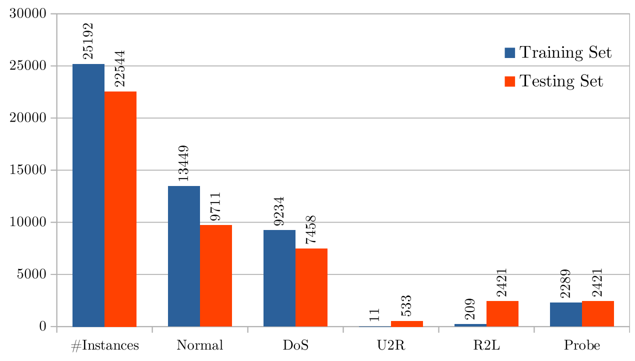

Figure 2. NSL-KDD dataset distribution.

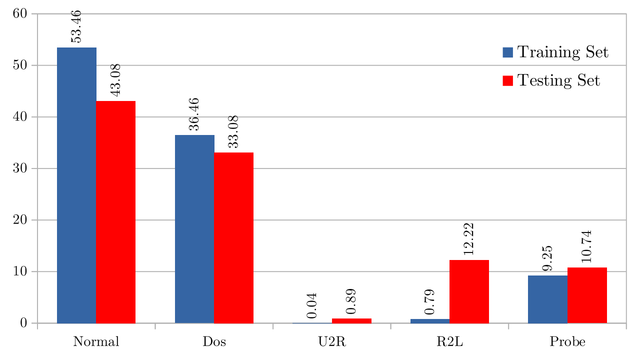

Figure 3. Frequencies of attack distribution in NSL-KDD dataset.

was designed to identify distributed attacks in both real-time and offline environments. Seth and Chandra *et al* [25] used Binary Grey Wolf Optimization (BGWO) algorithm for key FS with a neural network classifier. Mazini *et al* [26] use the AdaBoost algorithm for improved detection rate (DR) and accuracy; for FS they deploy Artificial Bee Colony (ABC) algorithm. Kumar and Sharma [27] propose an integrated framework that automatically detects, triggers the alarm, classifies and mitigates the software vulnerabilities. Alzubi *et al* [28] used a modified BGWO algorithm for the detection of attacks with feature reduction technique. Bharathy and Basha [29] used Multiple Criteria Linear Programming (MCLP) model for the classification and PSO algorithm for tuning the parameters to design network intrusion detection. Xue *et al* [30] suggested a self-adaptive PSO over enormous amount of features for dimensionality reduction. The experiments were performed over 12 datasets to represent the effectiveness of the algorithm. Bostani and Sheikhan *et al* [31] implemented binary gravitational search algorithm along with mutual information for FS to achieve better results from existing algorithms. SVM was used for feature reduction and neural network was implemented to rank the importance of selected features from DARPA dataset in [32]. Xue *et al* [33] introduced a self-adaptive differential evaluation (SaDE) for FS and *k* Nearest Neighbour (*k*-NN) for evaluating the different performance measures of IDS using the KDD Cup 99 dataset on wireless sensor networks. Wu *et al* [34] and Yu *et al* [35] proposed online feature selection (OFS) methods in the field of data mining and machine learning. Recently, evolutionary computation (EC) techniques were used for FS due to its ability of global optimization. Some of them include the firefly algorithm [36, 37] and the PSO [38–40].

4. Machine learning classifiers

We have used the following machine learning classifiers for the analysis of training and testing set of NSl- KDD dataset.

SVM: Cortes and Vapnik [41] developed the SVM classifier for binary classification of data into two classes. It contains the two hyperplanes, and the margin between them should be large. The basic principle behind this is to maximize the separating margin between negative and positive classes in order to derive a hyperplane. In SVM the mapping of the input vector is carried out to higher dimensional feature space, and the optimal separating hyperplane is accomplished in surpassing the dimensional feature space. The SVM has low rationalization error or does not deteriorate from the obstacle of overfitting to the training dataset [42]. When a model performs poorly on instances that are not commenced in the training, the set is said to have high generalization error or overfitting. The parameter penalty factor concedes users to make a trade-off between the width of the decision boundary and the number of misclassified samples. It is an effective technic for solving regression and classification problems. Later on, Zhang and Wang [43] developed a Computer-aided Design (CAD) system for the classification of images into two classes. They use 126 samples of brain Magnetic Resonance (MR) images in which 28 samples have Alzheimer' s disease (AD), and remaining samples are normal scans. Here, Displacement Field (DF) is employed for tracking the morphometry from normal brains to AD brains. Further, they have used the SVM and its two variants, namely Generalized Eigen-Value Proximal SVM (GEPSVM) and Twin SVM (TSVM), for classification of images.

Naive Bayes (NB): In most of the cases, it is troublesome to express the probabilistic relationship among the variables in cases where the variables have causal or statistical relations between them. In computing, the variable may influence others. The probabilistic model is designed to exploit the causal or statistical relationship of random variables of the problem, referred to as Naive Bayesian networks [44, 45]. In this case, it is expected that attributes are independent and this classifier performs well when combined with some attribute selection measures.

k Nearest Neighbour (k-NN): It is a non-parametric and most transparent machine learning technique for the classification and regression of samples [46, 47]. It performs the computation of approximate separation between several points on the input vector, and the unlabelled position designated to its *k*-NN. The *k* parameter represents the number of observations most adjacent to the given observation in testing or validation dataset. In this classifier, a new point is taken and classified according to the majority of the votes obtained for the nearest point in the training data. To measure the similarity between two points, Euclidean distance is used as the distance metric.

DT: In DT a sample is classified through a sequence of decisions, where the prevailing decision supports making the consecutive decision [48]. The classification of the sample takes place from the root node to the leaf node, where classification category is represented by each leaf node. Each node represents the attributes of the sample.

Logistic Regression (LR): This analysis studies about the association between the categorical dependent variable and independent variables, and takes place only for binary values such as 0 or 1, yes or no. It follows a binomial distribution where individual characteristics are the basis for creating the chance for the outcome of the model [49].

Random forest (RF): It is a machine learning classifier that incorporates the DT and ensemble learning [50, 51]. The forest consists of several trees, and the attributes are selected randomly as input. In this algorithm, a bunch of DTs along with an arbitrary subgroup of the data are created using a bagging approach. This algorithm is considered to be the most effective for the solution of almost any prediction task. For any problem related to classification and regression, it can be used. It is a combination of tree predictors where every tree confides on the values of an arbitrary vector sampled separately with the equal division for all trees in the forest. This algorithm split into two phases. In the first phase, '*i*' random trees are initiated, which generate the RF. The second phase combines all DTs that have the same test features. The final prediction consists of the DT that appears most frequently or the assessment of each DT. This algorithm operates well even for inconsistent data to generate better accuracy, is simple to use and also gives assessment on what variables are necessary for the classification purpose. It executes accurately on enormous databases while provoking an internal impartial evaluation of the generalization error. It also contributes a procedure for unbalanced datasets for stabilizing error in class population. The random algorithm specifically benefits data scientists to redeem data preparation time, as there is no requirement for any input preparation and can examine numerical data and specific features without scaling or transformation. In this ensemble method, several DTs are operated by training them, and the class having a majority over all the trees is returned. RF is slightly better than SVMs [52]. A collection of trees is constructed for the generation of forest having the controlled variance. The prediction of this classifier is decided by the weighted or majority voting. A flow chart of the algorithm is presented in figure 4.

Definition of RF: It uses trees $g_k(A, \theta_k)$ as k^{th} base learners where θ_k' s are a stack of random variables and they are independent for k = 1,...,K. For training data $D = (a_1, b_1), \ldots, (a_N, y_N)$ where $a_i = (a_i, 1, \ldots, a_i, p)^T$ represents the m predictors and b_i represents the response, and a

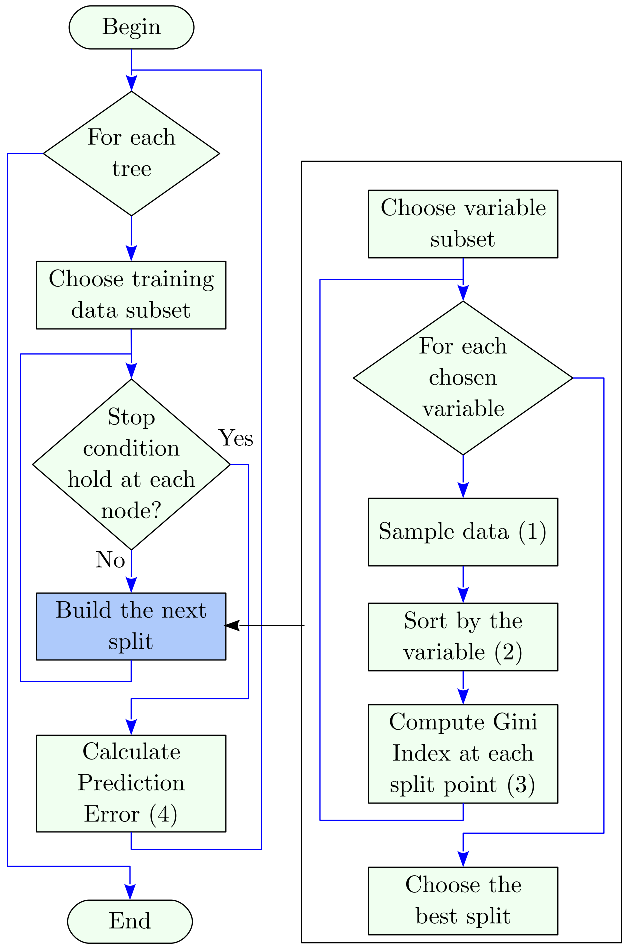

Figure 4. Flow chart of random forest algorithm.

specific realization θ_k of Θ_k, the fitted tree is represented as $\hat{g}_k(a, \theta_k, D)$. While this formulation is derived from [53, 54], in practice the random component θ_k is not treated explicitly. However, it is used to implant randomness in two phases implicitly. In the first phase, which includes bagging, an independent bootstrap pattern is taken from the original data to fit into each tree. The bootstrap sampling involves randomization, which gives one part of θ_k . In the second phase, during random selection, the best split is found from the subsets of r predictor variables instead of all m predictors, separately at each node while splitting a node. The sampling of predictors gives the remaining part of θ_k by randomization. A pseudo-code of RF is depicted in Algorithm 1. As discussed, this algorithm has several trees and each tree is constructed by following steps:

1. Assumptions: training cases = X and the classifier consists of a number of variables = Y.
2. For the determination of the decision at a node of the tree, we have y number of input variables, and $y < Y$.
3. In this tree, the selection of a training set is performed by taking a bootstrap sample, i.e., selecting x times with restoration from all X available training cases. The estimation of the tree' s error is performed using the rest of the cases.
4. Randomly, y variables are selected for each node of the tree to perform the decision at that node. The best split is calculated by these y variables in the training set.
5. Each tree is completely grown up and not sheared. A new sample is predicted by pushing it down the tree. The label is assigned to the training sample at the end of the terminal node. The iteration continues for all trees, and the prediction of RF is expressed by calculation of the average vote of all the trees.

The benefits of this algorithm are the following:

1. Applicable for both classification and regression problems.
2. Handles categorical predictors naturally.
3. Computationally quick and straightforward to fit, even for significant problems.
4. No explicit distributional assumptions (non-parametric).
5. It can handle highly nonlinear communication and classification boundaries.
6. Variable selection is automatic.
7. Handles missing values through surrogate variables using proximities.
8. Outlier detection, visualization and unsupervised learning.

Algorithm 1: Procedural steps of random forest algorithm.

```
1  Let $D = (a_1, b_1), \ldots, (a_N, b_N)$ represent the training data, with $a_i = (a_{i,1}, \ldots, a_{i,m})^T$
2  for $k = 1$ to $K$ : do
3      Proceed with taking a bootstrap pattern $D_k$ of size $N$ from $D$
4      Using the bootstrap pattern $D_k$ as the training data, fit a tree using binary recursive partitioning
5      Initiate all the observations within a single node
6      for each unsplit node do
7          repeat
                i. Select the $r$ predictors randomly from the $m$ usable predictors
                ii. Calculate the best binary division among all binary divisions on the $r$ predictors from step i
                iii. Divide the node into two child (descendant) nodes using the division of step ii
8          until termination criteria met;
9      end
10 end
11 To make a prediction at a new point $a$
       • $\hat{f}(a) = \frac{1}{K} \sum_{k=1}^{K} \hat{g}_k(a)$ for regression
       • $\hat{f}(a) = arg\ max_b \sum_{k=1}^{K} I(\hat{g}_j(a) = b)$ for classification
         where $\hat{g}_k$ is the prediction of the counter variable at $a$ using the $j$th tree
```

5. Swarm intelligence

Swarm intelligence has emerged as a population-based and nature-inspired algorithm with the capability of solving complex problems that provides robust, fast and low-cost solutions [55]. It is a subdivision of artificial intelligence that involves the collective behaviour of social insects, including ants, termites, bird flocks and honey bees. The social animals have limited capabilities, unsophisticatedly interact either directly or indirectly with each other by specific behavioural patterns for survival. The direct interaction includes communication over audio or video, such as flutter dance of honey bees. The indirect communication exists when an agent changes the environment and other agents acknowledge the changed environment, like ants depositing pheromone trails on their way in order to search for food sources. Swarm intelligence is used for the optimization of the problems including data analysis, scheduling, bioinformatics, machine learning, operations research and medical informatics and also in the field of business and finance. The author Bonabeau *et al* [56] defined swarm intelligence in very simple words as the popular way of simple and common intelligence of social agents. The parameters used for finding a set of solutions in swarm intelligence are the following. *Proximity principle*: effective computation for the execution of time and space solutions.

Quality principle: interaction of quality features of the environment.

Different response principle: existence of solutions beyond narrow channels.

Stability principle: stability of actions based on the changing conditions.

Adaptability principle: ability to adapt new conditions based on effective time and space computations.

6. Particle swarm optimization (PSO) algorithm

It was recommended by R Eberhart and J Kennedy in 1995 [57]. The PSO algorithm implemented in various streams of engineering is explained in [58] and is derived from the behaviour of bird flocking. It involves the adaption of rapidly changing interactions and movements of social animals like fishes and birds. A flow diagram of the PSO algorithm is presented in figure 5. The PSO algorithm combines the experiences learned when working together as a group and personal experiences. The optimized solutions are achieved by the flocking behaviour of birds. The birds follow some predetermined path for the destined food resources. This path, considered as the shortest path, is also referred to as the personal best solution (pbest) of the particle. Each particle looks for the best solution in the search space by observing its own flying experiences and experiences of others in the group. Another best fitness value is achieved by observing any of the particles in the group near the range of that particle. This is referred to as the gbest. Each particle has its associated velocity for the acceleration towards achieving the pbest and gbest. The basic concept of PSO is to achieve global optimal solution, thereby moving each particle towards pbest and gbest with arbitrary weight at every step. This algorithm gives improved exploration and exploitation [59, 60].

The equation for PSO algorithm is represented as

$$\begin{aligned} v_{kd}(t+1) =& v_{kd}(t) + c_1\mathbf{R}_1(p_{kd}(t) - x_{kd}(t)) \\ &+ c_2\mathbf{R}_2(p_{gd}(t) - x_{kd}(t)) \end{aligned} \tag{1}$$

$$x_{kd}(t+1) = x_{kd}(t) + v_{kd}(t+1) \tag{2}$$

where $v_{kd}(t)$ represents the velocity of the k^{th} particle in d^{th} dimension in t^{th} iteration, C_1 and C_2 are the acceleration factors, respectively, in personal and social cognizance directions, $x_{kd}(t)$ represents the position of k^{th} particle in d^{th} dimension in t^{th} iteration, $p_{kd}(t)$ represents the pbest position of the k^{th} particle in d^{th} dimension in t^{th} iteration and $p_{gd}(t)$ represents the gbest position obtained by the complete population till t^{th} iteration; R_1 and R_2 are random numbers between 0 and 1 used to overcome premature convergence.

7. Proposed methodology

7.1 *Preprocessing*

The classifiers led to false alarms due to the rough features. Hence, preprocessing of the dataset is the necessary part. Moreover, some typical features increase the computation time and memory resources that are unavoidable by the classification techniques. In the NSL-KDD dataset, rough features are expressed as

$$rs = \{fs_1, fs_2, fs_3, fs_4, \ldots, fs_n\}, \tag{3}$$

where n is 41, which represents 41 features in the dataset. The typical features are eliminated from rough features due to the overhead and redundancy. The modified rough features are represented as

$$rs' = \{fs_1, fs_2, fs_3, fs_4, \ldots, fs_p\}, \tag{4}$$

where p represents the number of rough features after elimination. Moreover, appended preprocessing is needed to obtain the optimized feature set based on their importance in the dataset. For this purpose, feature reduction and feature transformation are necessary, which are explained in the next section.

7.2 *FS method*

In this section, we propose the FS method using a mathematical equation. FS is represented as 6-tuplet: FS =

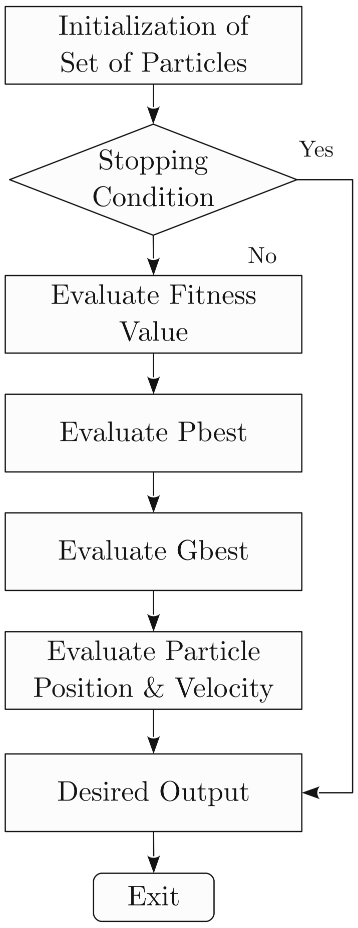

Figure 5. Flow diagram of the PSO algorithm.

$\{D, V, C, S, f_s, E\}$ where D is a dataset, $D = \{a_1, a_2, \ldots, a_k\}$ with k instances, V is set of features, $V = \{e_1, e_2, \ldots, e_f\}$ with f number of features, C is a target class, $C = \{c_1, c_2, \ldots, c_n\}$ with n classification of target classes, S (search space) is a distribution of set V that contains all subdivisions that we can construct by V, $S = \{s_1, s_2, \ldots, s_l\}$ $(l = 2^n - 1)$ with $s_i = \{e_p, e_q, \ldots, e_r\}$ $(l \leq p \neq q \neq r \leq n)$, E is an evaluation measure and function f_s represents the transformation of FS: $f_s : V \rightarrow S$. FS is the process of eliminating extravagant attributes and obtaining the optimized subset(s) from the dataset [61]. The objective of this technique is to select a subset of features that increase the efficiency, reduce the computation complexity and enhance the predictive accuracy. The steps performed in FS are presented in figure 6 and explained here.

1. The generation module provokes the next successor from the original feature set.
2. The estimation module calculates the applicability of the subsets using several measuring parameters.
3. The endpoint where the subset features are recognized as optimal.
4. The validation module to check the validity of the feature subset.

7.3 *Proposed model*

In this paper, we have performed FS of attributes before pre-processing. The process of selecting essential features from the original is termed as FS, and it is compulsory because of the perseverance of misleading and insignificant features in the dataset. The RF algorithm is used for performing FS, which is one of the most prominent machine learning algorithms. This algorithm has easy interpretability, low over-fitting and exemplary predictive performance. The interpretability is represented by deriving the importance of each variable on the DT. The process of selecting features using RF is referred to as embedded methods. This method is the amalgamation of wrapper methods and quality filters. RF inheres 4–1200 trees; each of them undergoes the observation of random extraction from the dataset as well as features. Decorrelated trees are less prone to over-fitting because every tree does not observe the features of others. Every tree undergoes a sequence of yes–no questions that are based on a combination of features. At every node of the problem the tree is divided into two buckets, each of them observing the similarities among themselves and differences in other buckets. This presents the importance and purity of every feature, which is derived from each bucket. Hence, RF classifier is applied to the training and testing set of the NSL-KDD dataset; the classifier selects top 10 features based on the importance and eliminates the insignificant features. Figure 7 presents the proposed methodology. The list of selected features is shown in figure 8 based on their importance in the dataset and described in section 2. The proposed PSO algorithm is blueprinted by Algorithm 2.

The training, as well as the testing set of the dataset, has 41 attributes; hence, excluding the 1 class attribute, there are 41 attributes for the training set. The onehot encoding is the feature used to convert categorical variables into a format suitable for machine learning algorithms for improving better job predictions. These 41 attributes

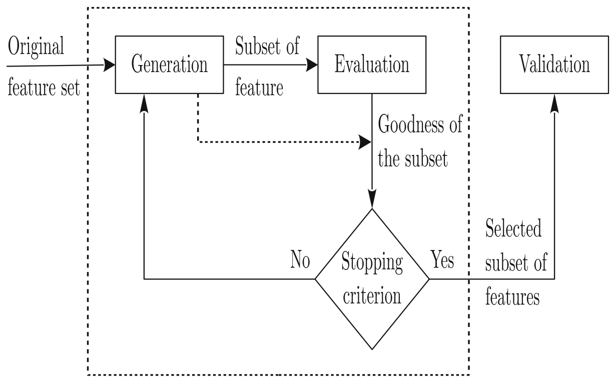

Figure 6. Feature selection process [62].

Algorithm 2: Proposed methodology.

```
   Input  : NSl-KDD dataset and preprocessing 42 features $f_1, f_2, \ldots f_{42}$
   Output: Display the classification results for accuracy, detection rate, false alarm rate, $F_1$-score of different classifier
1  Function Process_model ()
2      Perform FS on NSL-KDD dataset (having 42 features) using RF algorithm    (see Algorithm 1)   /* it
         selects the 10 features feature vector */
3      Calculate the classification accuracy through various existing machine learning classifiers, e.g., SVM, LR, NB,
         DT and k-NN on selected features
4      Perform the classification through PSO-based classifier with varying values of inertia weight, and control
         parameter $C_1$ and $C_2$ for cognitive and social component, and calculate the respective performances for
         mentioned measures
5      Compare the results.   /* it shows that the PSO-based classifier obtained considerable improvement for all
         measures as compared with the state-of-the-art classifiers */
6  end
```

undergo onehot encoding and 74 attributes are created. These attributes are used as an input layer in the neural network; the hidden layer has 20 neurons, which is used to decide the computation. The forward propagation is used as an objective function. This computes the forward propagation of the neural network as well as the loss. It receives a set of parameters that must be rolled back into the corresponding weights and biases. The PSO algorithm is implemented after the forward propagation on the selective features of the dataset with different iterations with a finite number of particles to observe the best accuracy among all the iterations.

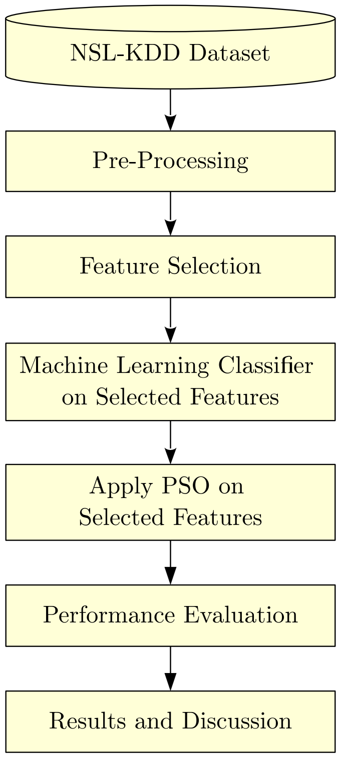

Figure 7. Proposed research methodology.

8. Performance measure

The performance measurement is achieved by calculating the FPR and TPR. FPR is also designated as false alarm rate (FAR) or failure rate; it is the classification of benign traffic as malicious. It can be formulated as the number of incorrect detection of normal records as intrusions divided by the total number of normal records. The DR is the number of correctly identified positive instances divided by the total number of instances identified as positive. The DR is also referred to as TPR or recall or sensitivity. The following parameters are used for performance measurement:

$$\begin{aligned}
\text{True Positive Rate (TPR)} &= \frac{\text{TP}}{\text{TP+FN}} \times 100,\\
\text{True Negative Rate (TNR)} &= \frac{\text{TN}}{\text{TN+FP}} \times 100,\\
\text{False Negative Rate (FNR)} &= 100 - \text{TPR} \times 100,\\
\text{False Positive Rate (FPR)} &= 100 - \text{TNR} \times 100,\\
\text{precision} &= \frac{\text{TP}}{\text{TP+FP}} \times 100,\\
\text{overall accuracy} &= \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \times 100,\\
\mathcal{F}_1\text{-score} &= \frac{2\text{TP}}{2\text{TP+FP+FN}} \times 100.
\end{aligned}$$

9. Analysis and results

The FS method eliminates unnecessary attributes from the dataset. Removing these unwanted attributes is a must because they decrease the accuracy of the algorithm, which we use to predict something in the future. As the number of features increases in the dataset, the search space also increases. The FS is a challenging task because it is relevant to the search space. It acts as a bridge for the extraction of features and pre-processing. In this study, we have performed FS using RF algorithm to cut down the number of attributes from the dataset to enhance the computation power and remove the redundant attributes that affect the

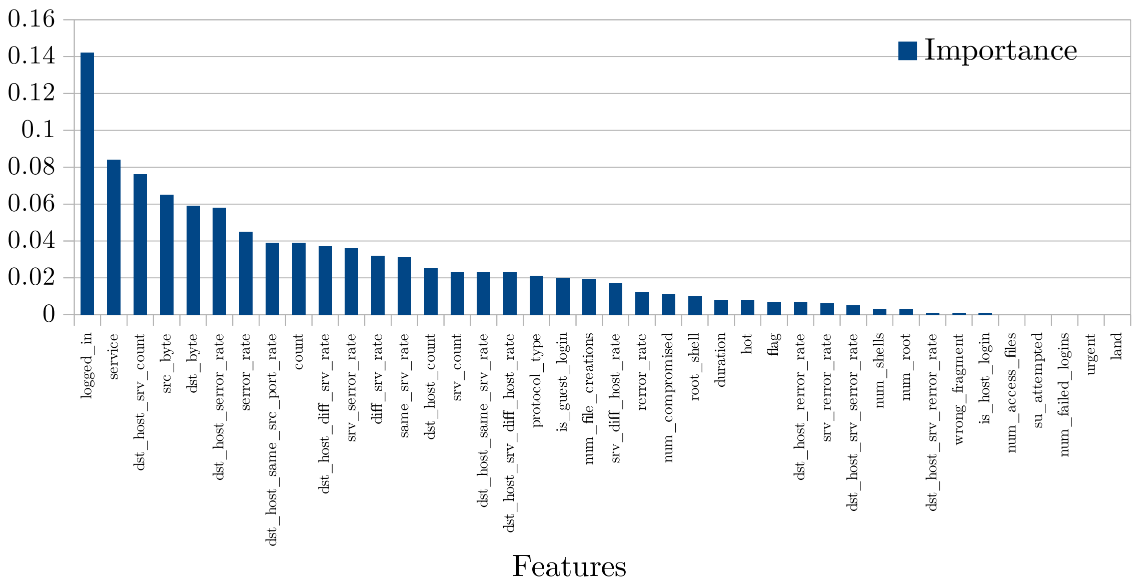

Figure 8. List of selected attributes representing their importance using random forest algorithm.

Table 5. Comparison of the proposed technique with machine learning classifiers in training set.

Methods	TPR	TNR	FPR	FNR	Precision	$\mathcal{F}_1$-score	Accuracy
SVM	97.31	96.13	3.87	2.69	96.08	96.69	96.71
NB	96.96	96.32	3.68	3.04	96.29	96.62	96.63
DT	96.43	97.02	2.98	3.57	97.02	96.73	96.73
k-NN	97.04	97.24	2.76	2.96	97.24	97.11	97.18
LR	96.57	95.15	4.85	3.43	95.07	95.82	95.85
Proposed approach	99.26	99.38	0.62	0.74	99.37	99.31	99.32

Table 6. Comparison of the proposed technique with machine learning classifiers in testing set.

Methods	TPR	TNR	FPR	FNR	Precision	$\mathcal{F}_1$-score	Accuracy
SVM	87.47	84.28	18.72	12.53	77.80	82.35	85.51
NB	86.77	81.74	18.26	13.23	73.57	79.35	83.37
DT	81.35	81.87	18.13	18.65	74.97	78.03	81.66
k-NN	81.86	84.47	15.53	18.14	77.59	83.48	86.66
LR	83.07	85.04	14.96	16.93	79.95	81.48	84.22
Proposed approach	89.12	87.02	12.98	10.88	81.11	84.92	87.83

efficiency of the system. After performing FS on the dataset, we have applied the PSO algorithm on the dataset, including training set and testing set, for enhancing the DR and optimization of the IDS. We have calculated various performance measures, including the precision, DR and accuracy of the system, based on the confusion matrix of the results, which are enlisted in table 5 and 6 using the FS method with the proposed PSO algorithm, which gives the best accuracy of 99.32% in training set and 87.83% in testing set, which is considered to be the optimized accuracy as compared with other machine learning classifiers,

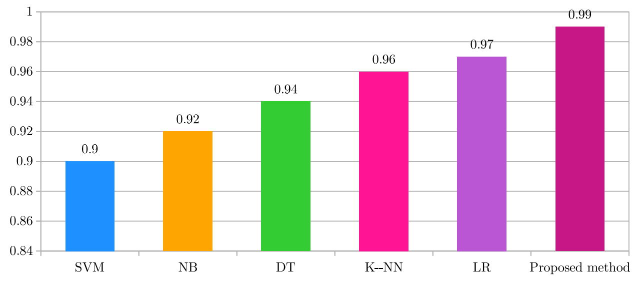

Figure 9. Comparison of accuracy of machine learning classifiers with the proposed method.

Table 7. Observations of the PSO algorithm using fixed number of particles with increased iterations.

Particles	Iterations	TPR	TNR	FPR	FNR	Precision	$\mathcal{F}_1$-score	Accuracy
2800	20	98.31	97.24	2.76	1.69	97.20	97.75	97.77
2800	21	98.68	97.13	2.87	1.32	97.08	97.87	97.89
2800	22	98.67	97.00	3.00	1.33	96.94	97.80	97.82
2800	23	99.01	97.13	2.87	0.99	97.07	98.03	98.05
2800	24	98.66	97.35	2.65	1.34	97.31	97.98	98.00
2800	25	98.26	97.34	2.66	0.74	97.27	98.25	98.27
2800	26	99.14	97.25	2.75	0.86	97.20	98.16	98.18
2800	27	98.61	97.07	2.93	1.39	97.01	97.80	97.83
2800	28	99.26	99.38	0.62	0.74	99.37	99.31	99.32
2800	30	98.85	97.04	2.96	1.15	96.98	97.90	97.93
2800	31	97.80	97.13	2.87	2.20	97.10	97.49	97.46
2800	32	98.31	95.87	4.13	2.69	95.79	96.54	96.58

including NB, SVM, DT, LR and k-NN algorithms. The comparison is presented in figure 9. This algorithm is superior to machine learning classifiers because PSO is a bio-inspired search technique that is simple and easy to implement, which involves a single operator for updating solutions. This algorithm has been very effective in a wide variety of applications, and has the ability to produce good solutions at a very low computational cost. A slight change in the parameters of this algorithm results in better performance and results of the systems [65–67]. It is robust and parallel computation can be performed. Accurate mathematical models can be solved efficiently. This algorithm converges rapidly without overlapping and mutation, and also has a higher probability of finding global optima. [68, 69].

To analyse the performance of the proposed work, we examine it on several combinations of the PSO parameters. We found that highest accuracy is obtained on the following PSO parameters: $C_1 = 0.5$, $C_2 = 0.3$ and $W = 0.9$. The tested results for different parameters are presented in table 10. Moreover, to obtain the perfect empirical combination of the number of particles and the number of iterations, we perform several preliminary experiments and find that 2800 number of particles and 28 number of iterations give the concluding performance remarks (see table 7). However, a change of parameters may result in different observations in a different scenario. Thus, the parameter optimization of the PSO algorithm is still an open area of research. The change of accuracy with respect to the iterations is depicted in figure 10. In this figure, we observe that the detection accuracy fluctuates within a small margin (approximately ±0.5) till the 27th iteration. This small change is because the search particles oscillate between their personal and cognizance; thus their respective positions get updated in a small range of search space (exploitation). However, some particles may iteratively keep the momentum towards the earlier best position (possible because the inertia weight is 0.9), and escape out from local optimum. Hence, they also guide other particles to move towards this newly discovered position (exploration). This is observed in figure 10 by a sudden increase in accuracy at 28th iteration. Moreover, excessive exploration of the search space also guides the particles to move towards nonoptimal area. Thus, we find that the accuracy suddenly drops between the 28th and 30th iterations. This algorithm is also tested for a different minimal set of

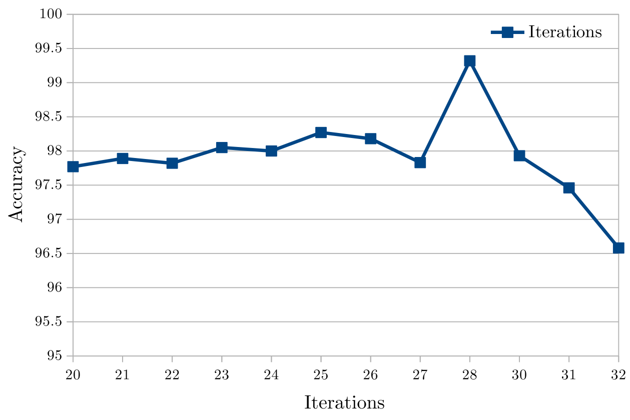

Figure 10. Accuracy observed using different iterations of the proposed PSO algorithm.

Table 8. Observations of the PSO algorithm with different feature sizes.

# Features	TPR	TNR	FPR	FNR	Precision	$\mathcal{F}_1$-score	Accuracy
10	*99.26*	99.38	0.62	0.74	99.37	99.31	99.32
12	98.63	97.18	2.82	1.37	97.13	97.87	97.89
15	99.05	97.11	2.89	0.95	97.04	98.03	98.05
18	98.43	96.59	3.41	1.57	96.55	97.48	97.50
20	99.51	97.24	2.76	0.49	95.77	97.61	97.66
22	98.93	96.52	3.48	1.07	96.42	97.66	97.69
25	99.17	96.43	3.57	0.83	96.68	97.92	97.95

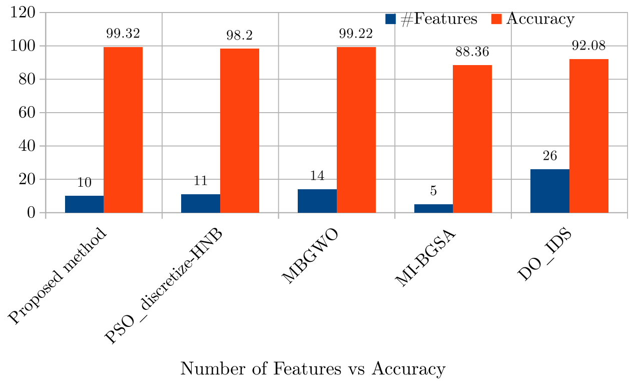

Figure 11. Comparison of accuracy with other algorithms.

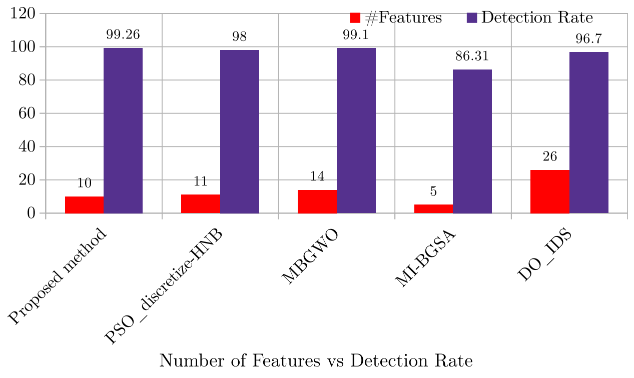

Figure 12. Comparison of detection rate with other algorithms.

Table 9. Comparison of the results with other algorithms.

Methods	Accuracy	Detection rate	# Features
PSO-discretize-HNB [63]	98.20	98.00	11
MBGWO [28]	99.22	99.10	14
MI-BGSA [31]	88.36	86.31	5
DO_IDS [64]	92.08	96.70	26
Proposed approach	99.32	99.26	10

features like 12, 15, 18, 20, 22 and 25 features using the same configuration of the PSO to compare the results to selected 10 features (see table 8). The comparison of accuracy and DR with existing methods is depicted in figure 11 and 12, respectively, which shows that the proposed technique has a better DR and accuracy than those from other methods. In addition, the number of selected features is lesser in the proposed PSO algorithm. The algorithm improves the DR when the PSO algorithm is applied to the reduced features of the NSL-KDD dataset. Hence, this algorithm gives optimized results in terms of the number of features. table 9 presents the comparison with existing algorithms, which indicates that this algorithm provides better results with 10 features only (table 10).

Table 10. Empirical parameter settings of the PSO algorithm.

C_1	C_2	W	Accuracy
0.7	0.5	0.9	97.55
0.7	0.5	0.8	96.42
0.7	0.6	0.9	98.18
0.6	0.5	0.9	97.88
0.5	0.3	0.9	99.32

10. Conclusion and future scope

This paper discusses the implementation of PSO algorithm with FS to enhance the accuracy and DR of the IDS. We have used only 10 features from NSL-KDD dataset in an attempt to remove the extra and noisy attributes with negative encounter on the pursuance of the system. Simplification of the dataset is the main objective of FS method by reducing its dimensionality and identification of the optimal subset of features. The RF algorithm is used to select top 10 features out of 41. The PSO algorithm is applied to the selected 10 features with a distinctive number of iterations with definite particles to achieve the optimized result. The number of iterations ranges from 20 to 28, and the number of particles is fixed to 2800. The best accuracy and DR are observed at 28 iterations with 2800 particles. The results are compared to those from SVM, DT, NB, LR and *k*-NN algorithms on training and testing sets of the dataset. The accuracy and other performance measures observed are better than those from other algorithms. This algorithm is also compared to existing algorithms where FS method is implemented on the same dataset; the results show that this algorithm gives the best accuracy with only 10 features as compared with other algorithms.

References

[1] Aghdam M H and Kabiri P 2016 Feature selection for intrusion detection system using ant colony optimization. *IJ Netw. Secur.* 18(3): 420–432

[2] Modi C, Patel D, Borisaniya B, Patel H, Patel A and Rajarajan M 2013 A survey of intrusion detection techniques in cloud. *J. Netw. Comput. Appl.* 36(1): 42–57

[3] Myerson J M 2002 Identifying enterprise network vulnerabilities. *Int. J. Netw. Manag.* 12(3): 135–144

[4] Liao H J, Lin C H R, Lin Y C and Tung K Y 2013 Intrusion detection system: a comprehensive review. *J. Netw. Comput. Appl.* 36(1): 16–24

[5] Bhuyan M H, Bhattacharyya D K and Kalita J K 2014 Network anomaly detection: methods, systems and tools. *IEEE Commun. Surv. Tutor.* 16(1): 303–336

[6] Teodoro P G, Verdejo J D, Fernandez G M and Vazquez E 2009 Anomaly-based network intrusion detection: techniques, systems and challenges. *Comput. Secur.* 28(2): 18–28

[7] Sperotto A, Schaffrath G, Sadre R, Morariu C, Pras A and Stiller B 2010 An overview of ip flow-based intrusion detection. *IEEE Commun. Surv. Tutor.* 12(3): 343–356

[8] Xue B, Zhang M and Browne W N 2013 Particle swarm optimization for feature selection in classification: a multi-objective approach. *IEEE Trans. Cybern.* 43(6): 1656–1671

[9] Xue B, Zhang M and Browne W N 2014 Particle swarm optimisation for feature selection in classification: novel initialisation and updating mechanisms. *Appl. Soft Comput.* 18: 261–276

[10] Yang H, Lyu M R and King I 2013 Efficient online learning for multitask feature selection. *ACM Trans. Knowl. Discov. Data* 7(2): 1–6

[11] Dhanabal L and Shantharajah S P 2015 A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. *Int. J. Adv. Res. Comput. Commun. Eng.* 4–6: 446–452

[12] Tavallaee M, Bagheri E, Lu W and Ghorbani A A 2009 A detailed analysis of the KDD cup 99 data set. In: *Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1–6

[13] Maza S and Touahria M 2019 Feature selection for intrusion detection using new multi-objective estimation of distribution algorithms. *Appl. Intell.* 49(1): 1–21

[14] Alzubi Q M, Anbar M, Alqattan Z N M, Al-Betar M A and Abdullah R 2019 Intrusion detection system based on a modified binary grey wolf optimisation. *Neural Comput. Appl.*, pp. 1–13

[15] Ganapathy S, Kulothungan K, Muthurajkumar S, Vijayalakshmi M, Yogesh P and Kannan A 2013 Intelligent feature selection and classification techniques for intrusion detection in networks: a survey. *EURASIP J. Wirel. Commun. Netw.* 1: 242–255

[16] Ahmad I and Amin F 2014 Towards feature subset selection in intrusion detection. In: *Proceedings of the 7th IEEE Joint International Information Technology and Artificial Intelligence Conference*, pp. 68–73

[17] Franco E D L H, Garcia A O, Lopera J O, Correa E D L H and Palechor M F 2015 Implementation of an intrusion detection system based on self organizing map. *J. Theor. Appl. Inf. Technol.* 71(3): 324–334

[18] Eesa A S, Orman Z and Brifcani A M A 2015 A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. *Expert Syst. Appl.* 42(5): 2670–2679

[19] Chebrolu S, Abraham A and Thomas J P 2005 Feature deduction and ensemble design of intrusion detection systems. *Comput. Secur.* 24(4): 295–307

[20] Zhang L, Zhang G, Yu L, Zhang J and Bai Y 2004 Intrusion detection using rough set classification. *J. Zhejiang Univ. Sci. A* 5(9): 1076–1086

[21] Deb K 1999 An introduction to genetic algorithms. *Sadhana* 24(5): 293–315

[22] Kaushik S S and Deshmukh P R 2011 Detection of attacks in an intrusion detection system. *Int. J. Comput. Sci. Inf. Technol.* 2(3): 982–986

[23] Tsai C F, Hsu Y F, Lin C Y and Lin W Y 2009 Intrusion detection by machine learning: a review. *Expert Syst. Appl.* 36(10): 11994–12000

[24] Modi C and Patel D 2018 A feasible approach to intrusion detection in virtual network layer of cloud computing. *Sadhana* 43(7): 114

[25] Seth J K and Chandra S 2016 Intrusion detection based on key feature selection using binary GWO. In: *Proceedings of the 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 3735–3740

[26] Mazini M, Shirazi B and Mahdavi I 2018 Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and adaboost algorithms. *J. King Saud Univ. Comput. Inf. Sci.*, pp. 541–553

[27] Kumar M and Sharma A 2017 An integrated framework for software vulnerability detection, analysis and mitigation: an autonomic system. *Sadhana* 42(9): 1481–1493

[28] Alzubi Q M, Anbar M, Alqattan Z N M, Al-Betar M A and Abdullah R 2019 Intrusion detection system based on a modified binary grey wolf optimisation. *Neural Comput. Appl.*, pp. 1–13

[29] Bharathy A M V and Basha A M 2017 A multi-class classification MCLP model with particle swarm optimization for network intrusion detection. *Sadhana* 42(5): 631–640

[30] Xue Y, Xue B and Zhang M 2019 Self-adaptive particle swarm optimization for large-scale feature selection in classification. *ACM Trans. Knowl. Discov. Data* 13(5): 1–27

[31] Bostani H and Sheikhan M 2017 Hybrid of binary gravitational search algorithm and mutual information for feature selection in intrusion detection systems. *Soft. Comput.* 21(9): 2307–2324

[32] Sung A H and Mukkamala S 2003 Identifying important features for intrusion detection using support vector machines and neural networks. In: *Proceedings of the Symposium on Applications and the Internet*, pp. 209–216

[33] Xue Y, Jia W, Zhao X and Pang W 2018 An evolutionary computation based feature selection method for intrusion detection. *Secur. Commun. Netw.*, pp. 1–10

[34] Wu Y, Hoi S C, Mei T and Yu N 2017 Large scale online feature selection for ultra-high dimensional sparse data. *ACM Trans. Knowl. Discov. Data* 11(4): 48

[35] Yu K, Wu X, Ding W and Pei J 2016 Scalable and accurate online feature selection for big data. *ACM Trans. Knowl. Discov. Data* 11(2): 16

[36] Yang X S 2010 *Nature-inspired metaheuristic algorithms*. Luniver Press, pp. 1–75

[37] Zhang Y, Song X and Gong D 2017 A return cost-based binary firefly algorithm for feature selection. *Inf. Sci.* 418: 561–574

[38] Bharti K K and Singh P K 2016 Opposition chaotic fitness mutation based adaptive inertia weight BPSO for feature selection in text clustering. *Appl. Soft Comput.* 43: 20–34

[39] Zhang Y, Gong D, Hu Y and Zhang W 2015 Feature selection algorithm based on bare bones particle swarm optimization. *Neurocomputing* 148: 150–157

[40] Xue B, Zhang M and Browne W N 2014 Particle swarm optimisation for feature selection in classification: novel

initialisation and updating mechanisms. *Appl. Soft Comput.* 18: 261–276

[41] Cortes C and Vapnik V 1995 Support-vector networks. *Mach. Learn.* 20(3): 273–297

[42] Smola A J and Scholkopf B 2004 A tutorial on support vector regression. *Stat. Comput.* 14(3): 199–222

[43] Zhang Y and Wang S 2015 Detection of Alzheimer's disease by displacement field and machine learning. *PeerJ* 3: e1251

[44] Pearl J 2014 *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, pp. 1–551

[45] Zhang H 2004 The optimality of naive Bayes. *AA* 1(2): 1–6

[46] Fukunage K and Narendra P M 1975 A branch and bound algorithm for computing *k*-nearest neighbors. *IEEE Trans. Comput.* 7: 750–753

[47] Altman N S 1992 An introduction to kernel and nearest-neighbor nonparametric regression. *Am. Stat.* 46(3): 175–185

[48] Safavian S R and Landgrebe D 1991 A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man Cybern.* 21(3): 660–674

[49] Hosmer Jr D W, Lemeshow S and Sturdivant R X 2013 *Applied logistic regression*. John Wiley & Sons, vol. 398, pp. 1–511

[50] Liaw A, Wiener M 2002 Classification and regression by random forest. *R News* 2(3): 18–22

[51] Zhang H and Singer B H 2010 *Recursive partitioning and applications*. Springer Science & Business Media, pp. 1–258.

[52] Lorena A C, Jacintho L F, Siqueira M F, Giovanni R D, Lohmann L G, Carvalho A C D and Yamamoto M 2011 Comparing machine learning classifiers in potential distribution modelling. *Expert Syst. Appl.* 38(5): 5268–5275

[53] Breiman L 2001 Random forests. *Mach. Learn.* 45(1): 5–32

[54] Cutler A, Cutler D R and Stevens J R 2012 Random forests. *Ensemble Mach. Learn.* 45(1): 157–175

[55] Kennedy J 2006 Swarm intelligence. In: *Handbook of Nature-inspired and Innovative Computing*, pp. 187–219

[56] Bonabeau E, Marco D R D F, Dorigo M and Theraulaz G 1999 *Swarm intelligence: from natural to artificial systems*. Oxford University Press, vol. 1, pp. 1–320

[57] Kennedy J 2010 Particle swarm optimization. In: *Encyclopedia of Machine Learning*, pp. 760–766

[58] Zhang Y, Wang S and Ji G 2015 A comprehensive survey on particle swarm optimization algorithm and its applications. *Math. Probl. Eng.*, pp. 1–39

[59] Mirjalili S, Wang G G and Coelho L S 2014 Binary optimization using hybrid particle swarm optimization and gravitational search algorithm. *Neural Comput. Appl.* 25(6): 1423–1435

[60] Kumar D and Ramakrishnan A G 2016 Binary classification posed as a quadratically constrained quadratic programming and solved using particle swarm optimization. *Sadhana* 41(3): 289–298

[61] Singh P, Verma A and Chaudhari N S 2015 Feature selection based classifier combination approach for handwritten devanagari numeral recognition. *Sadhana* 40(6): 1701–1714

[62] Dash M and Liu H 1997 Feature selection for classification. *Intell. Data Anal.* 1(4): 131–156

[63] Elngar A, Mohamed D and Ghaleb F 2013 A real-time anomaly network intrusion detection system with high accuracy. *Inf. Sci. Lett.* 2(2): 49–56

[64] Ren J, Guo J, Qian W, Yuan H, Hao X and Jingjing H 2019 Building an effective intrusion detection system by using hybrid data optimization based on machine learning algorithms. *Secur. Commun. Netw.s*, pp. 1–12

[65] Gudise V G and Venayagamoorthy G K 2003 Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks. In: *Proceedings of the IEEE Swarm Intelligence Symposium, SIS03*, pp. 110–117

[66] Sierra M R and Coello C A C 2006 Multiobjective particle swarm optimizers: a survey of the state-of-the-art. *Int. J. Comput. Intell. Res.* 2(3): 287–308

[67] Robinson R and Samii Y R 2004 Particle swarm optimization in electromagnetics. *IEEE Trans. Antennas Propag.* 52(2): 397–407

[68] Abdmouleh Z, Gastli A, Brahim L B, Haouari M and Al-Emadi N A 2017 Review of optimization techniques applied for the integration of distributed generation from renewable energy sources. *Renew. Energy* 113: 266–280

[69] Wang Z, Zhang Q and Zhang D 2007 A PSO based web document classification algorithm. In: *Proceedings of the Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007)*, vol. 3, pp. 659–664