EKSPLOATACJA I NIEZAWODNOŚĆ MAINTENANCE AND RELIABILITY

Eksploatacja i Niezawodnosc – Maintenance and Reliability

Volume 24 (2022), Issue 4

journal homepage: http://www.ein.org.pl

Article citation info:
Agrawal AP, Singh N. Double Layered Priority based Gray Wolf Algorithm (PrGWO-SK) for safety management in IoT network through anomaly detection . Eksploatacja i Niezawodnosc – Maintenance and Reliability 2022; 24 (4): 641–654, http://doi.org/10.17531/ein.2022.4.5

Double Layered Priority based Gray Wolf Algorithm (PrGWO-SK) for safety management in IoT network through anomaly detection

Indexed by: Web of Science Group

Akhileshwar Prasad Agrawal [a,*], Nanhay Singh [a]

[a] Guru Gobind Singh Indraprastha University, Dept of Computer Science and Engg., Ambedkar Institute of Advanced Communication Technologies and Research (now NSUT-E), Geeta Colony, Delhi-110031, India

Highlights

- Proposed swarm intelligent based PrGWO resulting in optimal feature set.
- Contributed by improving classification of individual classes present in datasets.
- Overall the accuracy improved for two of the datasets, with third very close to best.
- Proposed new fitness function resulting in inclusive performance measurement.

Abstract

For mitigating and managing risk failures due to Internet of Things (IoT) attacks, many Machine Learning (ML) and Deep Learning (DL) solutions have been used to detect attacks but mostly suffer from the problem of high dimensionality. The problem is even more acute for resource starved IoT nodes to work with high dimension data. Motivated by this problem, in the present work a priority based Gray Wolf Optimizer is proposed for effectively reducing the input feature vector of the dataset. At each iteration all the wolves leverage the relative importance of their leader wolves' position vector for updating their own positions. Also, a new inclusive fitness function is hereby proposed which incorporates all the important quality metrics along with the accuracy measure. In a first, SVM is used to initialize the proposed PrGWO population and kNN is used as the fitness wrapper technique. The proposed approach is tested on NSL-KDD, DS2OS and BoTIoT datasets and the best accuracies are found to be 99.60%, 99.71% and 99.97% with number of features as 12,6 and 9 respectively which are better than most of the existing algorithms.

Keywords

Gray Wolf Optimizer, anomaly detection, feature selection, predictive maintenance.

This is an open access article under the CC BY license (https://creativecommons.org/licenses/by/4.0/)

1. Introduction

Internet of things (IoT) application is ever increasing since its inception due to its widespread use in areas of smart buildings, smart vehicles, smart highways and wireless sensor network which will grow exponentially in coming years [27]. The network layer is particularly more vulnerable to attacks due to attackers' ability to launch it at several locations. Various famous attacks aimed at the network layer are categorized into Denial of service (DoS) attacks, User to Root (U2R) attacks etc [8]. Apart from the network layer, attacks may be directed at the application layer or other layers, too. Over the years research efforts have been aimed at improving the algorithms for classification and thereby improving Intrusion detectin system (IDS), however still there are lot of problems left to be adequately addressed in order to make the system robust.

Problems and motivation:

1. Developing a feature selection technique to find the most optimal reduced feature vector. The requirement for optimized feature vector is necessary to a) reduce the training time of the model and b) test the incoming traffic in real time.
2. Developing an IDS that can improve classification accuracy, detection rate, false positive rate and other important performance metrics using this optimal reduce feature vector.

Objective:

Motivated by the problems listed above, the authors intend to devise an IDS technique/methodology **to reduce and find optimal feature vector giving maximum fitness in terms of accuracy, Detection rate (DR) and False positive rate (FPR)** for a given traffic/dataset. Thus the present problem can be conceived as a **multiobjective optimization** problem. This optimal feature vector can then be used for classification on the IoT nodes. The IoT nodes based on the trained model can then test for incoming traffic using this optimal reduced feature set (instead of considering the entire feature set)in real time.

Contribution:

1. A priority based Gray wolf Optimizer with SVM and kNN (PrGWO-SK) is proposed, in which the support vector Machine (SVM) alongwith first layer fitness function is used to find the reduced feature vector which then acts as feeder vector for the initialization of Priority based Gray wolf optimizer (PrGWO) wolves/particles.

(*) Corresponding author.
E-mail addresses: A.P. Agrawal (ORCID: 0000-0001-6366-0311): kpw.ce08@gmail.com, N. Singh (ORCID: 0000-0002-3303-1386): nanhay.singh@nsut.ac.in

2. PrGWO is proposed wherein the intra-group priority of different leader hunting wolves is used by other wolves to update their own positions during each iteration. The leader wolves uses the same concept in a modified form to update their own positions.
3. In the present work, two fitness functions are proposed to be used in the two different layers. The first layer fitness function focuses solely on the accuracy while the second layer fitness function balances not only the requirement of accuracy but considers other metrics like detection rate, false positive rate and length of feature vector.
4. PrGWO-SK was tested on three datasets-legacy NSL-KDD dataset and newly captured DS2OS, BoTIoT datasets using various metrics- Accuracy,FPR, Recall, Precision and F1-score and the results were found to be encouraging.

2. Related work

The IDS system working on resource staved IoT nodes has limited capacity. Antunes Rodrigues et al. [3] emphasized on preventive maintenance of sensor nodes in industrial application using Ishikawa diagram and FMECA. M. Almiania et al. [2] used the recurrent neural network concept to classify the data but it suffered from delays and was not amenable to real time applications. Tian et al. [39] used the deep belief network alongwith non-Mean gaussian distribution to classify the network packets. However, feature selection was not considered here. Baranowski [4] used the Bayesian workflow to predict attacks and failures in IoT particularly considering device variance. Modi et al. [22] developed a framework which used Snort and combined different classifiers, viz Decision tree, Bayesian & Associative. However, this was done for cloud network which is not resource starved as IoT nodes. Sivapalan et al. [32] suggested using lightweight Neural network to detect attacks in the wearable IoT(ECG). Gao et al. [6] in their paper proposed to use memory augmented autoencoder for detecting attacks in time series sensor data. A very important supervised technique is SVM which is used to draw hyper-plane between different classes. In one of the papers, the authors analysed threats using artificial neural networks [12].Vijayanand et al. [40] used the technique of SVM in the field of mesh networks security and found it to be efficient in distinguishing the attack from normal types. Similarly, E.Shams et al. [30] in their paper used the SVM technique to implement security mechanism in vehicular adhoc network. To reduce the training time, subsampling technique was used to filter out the less useful data thus validating the utility of SVM in different applications. In the literature, extensive research has been done in the field of metaheuristic algorithms for traversing the search space and converging to a solution. Tama et al. [35] in their paper have used ensemble technique to classify the data. In this paper feature selection was accomplished through Ant colony optimization and PSO. However the work suffers from methodological complexity in using a large number of algorithms. Kunhare et al. [19] used the random forest algorithm for feature selection coupled with the PSO algorithm. The accuracy and number of features were optimally generated, however the comparison with other work was not extensive. Wei et al. [41] in their paper used the jaccard fitness function for evaluating the optimality of the feature set. The accuracy was measured to be very good but the number of features increased. J.Gu et al. [9] used the concept of Naive Bayes to enhance the differences in the feature values to enable SVM to clearly distinguish between normal and attack types. Though the accuracy improved, the number of features was neglected. T.Wisanwanichthan et al. [42] in their paper used double layered approach i.e. SVM and Naive Bayes to classify the data. In their work the emphasis was on R2L and U2R but overall the accuracy suffered. Inspired by swarm intelligence concept, Mirjalili S. et al. [21] developed gray wolf optimization algorithm which used swarm technique to combine multiple greedy best solutions to update the subsequent solutions. Here the optimal solution was reached by calculating the fitness function. E. Emary et al. [5] in their paper has proposed the binary version of Gray wolf optimizer for feature selection. The proposed approach was validated by using it over a number of datasets. However, for NSL-KDD dataset no experiments were performed. M. Safaldin et al. [28] in their work used five leader wolves to guide the new positions of all wolves instead of four. The paper discussed about the enhancement ratio concept and compared the results according to population size. Apart from the above mentioned works, some of the important related work is depicted in the Table 1.

3.1. Proposed PrGWO-SK

Fig.1 depicts the proposed methodology of PrGWO-SK in graphical form. The proposed methodology starts with acquisition of net-

Table 1. Related work at a glance

References	Year	Main Technique	Feature sel.	Important notes
Pajouh et al. [25]	2017	Linear discriminant analysis	Yes	Both NB and kNN used for two tier feature selection and classification
Shone et al. [31]	2018	S-NDAE	Yes	Used S-NDAE and RFA
Yao et al. [45]	2019	HMLD	Yes	Hybrid feature selection and hybrid classification through SVM and ANN
Gu et al. [10]	2019	DT-EnSVM2	No	Used SVM along with feature transformation
Kumar et al. [18]	2021	Xgboost and kNN	Yes	Used information gain, correlation coefficient to reduce the features based on importance
Gao et al. [7]	2019	Adapting ensemble	Yes	Used multiple ML algorithms like RF, DNN, kNN, RF and using the voting mechanism for ensembling them
Golrang et al. [8]	2020	NSGAII-ANN	Yes	Proposed the use of Random Forest and multi-objective fitness criteria for FS
Wisanwanichthan et al. [42]	2021	DLHA	Yes	Division of features was done to use Naive Bayes with SVM
Gu et al. [9]	2020	NB-SVM2	No	Used the Naive Bayes concept to enhance the features of data for classification by SVM
Teng et al. [38]	2018	DT	Yes	Used the SVM along with DT for adaptive classification
Alazzam et al. [1]	2020	Sigmoid_PIO	Yes	Pigeon inspired algorithm for feature selection.
Tama et al. [35]	2019	PSO	Yes	Used two stage classifier ensembles for anomaly detection.
Wei et al. [41]	2020	NIA+GHSOM-pr	Yes	Used Jaccard's coefficient for measuring fitness
Safaldin et al. [28]	2020	GWOSVM-IDS	Yes	Modified the GWO algorithm to detect anomalies

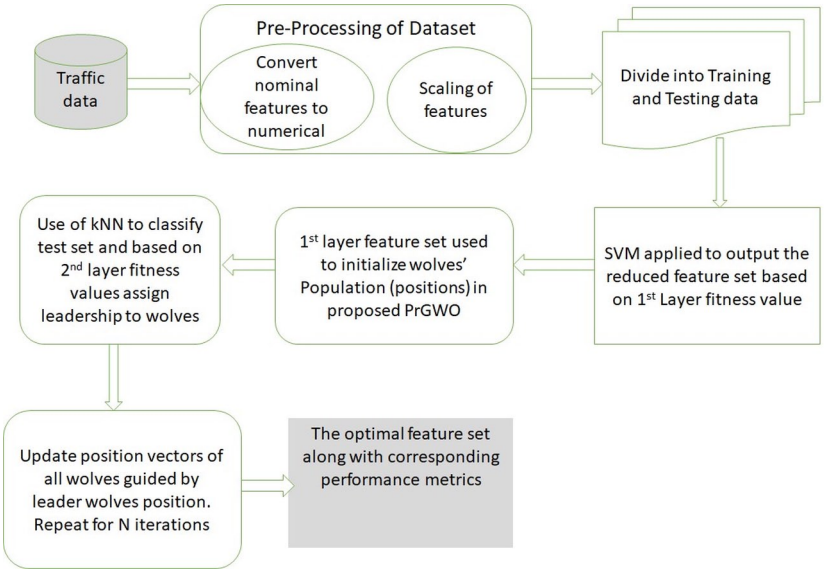

Fig. 1. Methodology of the PrGWO-SK algorithm

work traffic data for which NSL-KDD[36][37], BoTIoT[17] and DS2OS[23] datasets are used. NSL-KDD dataset is a legacy and widely used network dataset for intrusion detection work thus making it possible to compare the present work with others work in a comprehensive way. The DS2OS dataset is a collection of traces captured at the application layer in IoT environment while the BoTIoT dataset is a relatively recent dataset captured by designing network environment. Thus the testing of proposed approach on three different datasets-each one having their unique characterstic- helped validate the proposed work. Secondly, the used dataset is pre-processed using section 3.2 to convert into a form acceptable to the algorithm. Thirdly the data set is divided into 80% training and 20% testing subsets through random selection. Fourthly, the data is passed to the first layer (SVM with Radial basis Function(rbf)) which used first layer fitness function for generating reduced feature set. The SVM with rbf kernel has the advantage of seperating non-linearly seperable, closely related classes in dataset. The reduced feature set is passed to the second layer which used this to initialise the population of search agents of proposed algorithm 2 (These initialized search agents represent distinct feature sets). Initialization of search agents is an important step in swarm intelligence algorithms thus this work focusses on this aspect specifically. For achieving this two layered model is used in PrGWO-SK. Here the first layer exclusively considers accuracy as the basis for classification as it is one of the most important metrics in context of security mechanism. Thus the objective of first layer is solely to provide good solution for initialization of swarm population.

In the second layer the focus is not only on accuracy but also on other performance metrics, especially the length of feature vector finally selected. Thus, the second layer seeks to optimize both the length of feature vector alongwith accuracy and other important measures. To reiterate this can then be conceived as the multi objective optimization problem.

The algorithm then used these initialized search agents coupled with kNN as wrapper technique for generating classification results. kNN is used for its utility to evolve with new datapoints i.e new attack and normal access types as it does not involve explicit function generation. Thus it can give flexible non linear decision boundaries in a simple way. The relative performance of these distinct feature sets is evaluated through use of second layer fitness function. Accordingly four top search agents are assigned as leader wolves. In each subsequent iteration the position of each search agent is updated with the guidance of assigned leader wolves (four best search agents of the previous iteration). At the end of last iteration the feature set corresponding to best search agent is assigned as the optimal feature vector and passed to the real world classifier for detection.

Algorithm 1 gives step by step account of PrGWO-SK. Notations used in the algorithm are specified below. W_i (i=1..N) denotes the wolves, FNR stands for False negative rate, TNR for True negative rate; rest of the notations are defined in the algorithm itself. In the first phase/layer data preprocessing is done using techniques of section 3.1. Also the number of wolves participating in the algorithm are specified as parameter. This population of wolves is initialized randomly and algorithm 2 called with following parameters- Initialized population vector, fitness function 1 and wrapper method as SVM. The algorithm 1 stored the returned reduced feature vector. In the second layer/phase the feature vector returned by the first layer is used to initialize the population of wolves. All features of the returned feature vector are mandatorily selected union random selection over rest of the features. Once again the algorithm 2 is called with this set of initialized population, fitness function 2 and wrapper method kNN. The returned feature vector by algorithm 2 is stored as optimal feature vector and given to the real world classifier for classification using only the features present in this optimal feature vector.

3.2. Preparing dataset and preprocessing

NSL-KDD: Published in 2009, it is an enhanced version of the CUP99 dataset from KDD. Before the assessment of NSL-KDD, several researchers used the KDD'CUP99 dataset. But KDD'CUP99 has many duplicates and this makes the dataset redundant and biased towards some of the attacks. NSL-KDD has got 41 independent features and one dependent feature. These 41 independent features can broadly be classified as- basic features, content features and traffic features. Regarding dependent feature it is categorised into 39 attack types. Broadly the attack types can be categorised into DoS, U2R, Remote to local (R2L) and Probe types. Additionally one type present is 'normal' to denote normal class. Regarding approximate distribution of various attack and normal types it is as:
Normal=53.45%, Probe=9.25%, DoS=36.45%, U2R=0.04% and R2L=0.78%.

DS2OS: Published in 2018, this open-source data set was obtained via Kaggle. In a virtual IoT environment, the distributed smart space orchestration system (DS2OS) is used to create the dataset. The entire virtual architecture, which is a collection of many micro-services in an IoT context. In the DS2OS dataset there are 12 independent features- majority having nonnumerical values. Regarding attack types these can be categorised into 7 types- DoS, Malicious control (MC), Malicious Operations (MO), Probe, Scan, Spy and Wrongsetup (WS). Regarding approximate distribution they are as:
Normal=97.2%, DoS=1.59%, MC=0.25%, MO=0.23%, Probe=0.09%, Scan=0.43%, Spy=0.14% and WS=0.03%.

BoTIoT: Published in 2019, the UNSW Canberra Cyber Range Center's practical network configuration was built in order to build the BoT-IoT dataset. A workable substitute for IoT solutions, this data collection is produced utilising the message queuing telemetry transport (MQTT) protocol, which connects machine-to-machine interactions. In total there are 43 independent features while the attack categories present in BoTIoT dataset are 4 in number- DoS, Distributed denial of service (DDoS), Reconnaissance and theft. Regarding approximate distribution of attack and normal types it is as:

Normal=0.012%, DDoS=52.5%, DoS=44.9%, Recon=2.4% and theft=0.002%.

As discussed before, all datasets are preprocessed before they can be used in the actual ML algorithm. Below described techniques are used to transform them in a form amenable to classification algorithms:

Handling Missing values: Very often the dataset contains few missing values which affect the classification model's performance. In case of NSL-KDD and BoTIoT there are no missing values, while for DS2OS dataset the missing values are replaced with "missing" term.

Feature mapping: A few independent features like protocol type, flag and service are of nominal type which needs to be converted into numerical type. In the literature approaches like one hot encoding have been used. However, one hot encoding makes the dataset much more sparse thereby increasing the computation complexity. Therefore, in the present work ordinal encoding is used for converting nominal into numerical values. For example for protocol_type attribute the original values are converted as: TCP=1, UDP=2, ICMP=3. The same technique is used for other features also.

Feature normalization: Feature normalization or scaling is required to infuse uniformity in the values of features. In absence of feature normalization the higher values tend to dominate the trained model parameters. In the present work, all the independent features are scaled to a uniform level using min-max scaling technique:

$$F = \frac{f - f_{min}}{f_{max} - f_{min}} \tag{1}$$

Here, f is feature which is normalized, f_{max} is a feature's maximum value, f_{min} is its minimum value present in the dataset.

After scaling is done the next step is to split entire data set into training and test subdataset. Random selection technique is used for dividing the entire dataset into 80:20 ratio.The algorithm was repeated for 10 runs with different train:test subset using randomization. This is required to avoid chance selection bias in results.

Algorithm 1: PrGWO-SK

Input:

D = $D_i \cup D_d$ where \⋆ D_i are the independent features and D_d the label in the dataset ⋆\

Output:

$F_{optimal}$: The set of optimal features

Steps:

1. Load .csv file

2. D= ordinal encoding(D) \⋆ Convert the non-numeric data into distinct numeric types. ⋆\

3. D_i = D.drop(label) \⋆ Get the matrix of independent features ⋆\

4. D_d = D.drop(D_i) \⋆ Get the column vector of dependent feature or label ⋆\

5. D_i = Normalize(D_i) \⋆ Scale the dataset D_i to create uniformity among various features.

6.Initialize N value \⋆ No. of wolves or search agents used ⋆\

7. Repeat for 10 times:

7.1. [D_{train}, D_{test}] = Random selection on ($D_i \cup D_d$) \⋆ Random selection to create training:test as 80:20

Layer 1:

7.2. Initialize and call PrGWO with

W_i (i=1..N) = [X_i^d] where $X_i^d \in \{0,1\}$. \⋆ d is the vector dimension and i is the search agent no. ⋆\,

$Fitness = fit_{1layer}$ and

Classifier= SVM (rbf)

7.3.F_{SVM} = W[alpha]

Layer 2:

7.4. Initialize and call PrGWO with:

$W_i(i = 1....N) = [F_{SVM} \cup (D_i - F_{SVM}) \in \{0,1\}]$,

$Fitness = fit_{2layer}$ and

Wrapper Method = kNN

7.5.$F_{optimal}$ = W[alpha]

Output $F_{optimal}$, Acc, DR, FPR, FNR, TNR, Precision, F1-score.

3.3. Proposed PrGWO

Background: The Gray wolf optimizer algorithm [21] is inspired by the metaheuristic approach which searches through the solution space for optimal solutions. Notionally, here a pack of leader wolves is responsible for guiding the followers' pack towards the prey's location and then hunting it down. The leader pack consists of best three solution vectors and are called as alpha, beta, delta wolves. Though all wolves start from a random solution vector but they move towards optimal solution vector through help of their leader wolves.

The update equation for all wolves was given as:

$$W_i(iter+1) = \frac{\omega 1 + \omega 2 + \omega 3}{3} \tag{2}$$

Here:

$$\omega 1 = \left| W[alpha] - A_1 . D[alpha] \right| \tag{3}$$

$$\omega 2 = \left| W[beta] - A_2 . D[beta] \right| \tag{4}$$

$$\omega 3 = \left| W[delta] - A_3 . D[delta] \right| \tag{5}$$

where W_i(iter+1) is the i^{th} wolf position in next iteration, $W[alpha], W[beta], W[delta]$ are the positions of the Alpha, Beta and Delta wolves, A_i are the factor values and D[alpha], D[beta], D[delta] are the distance vectors of i^{th} wolf from alpha, beta and delta wolves' position vector respectively.

For adapting the general algorithm having continuous values to feature selection task with each feature represented by only binary variable $\in 0,1$ the above algorithm was modified [5]. In this algorithm the mapping was done from continuous values to the binary values. In the modified scenario, the update equation was written as:

$$W_i(iter+1) = crossover(\omega 1, \omega 2, \omega 3) \tag{6}$$

Here crossover is defined as :

$$crossover(\omega1,\omega2,\omega3)=\begin{cases}\omega1 & if \quad rand<\frac{1}{3}\\ \omega2 & if \quad \frac{1}{3}\le rand<\frac{2}{3}\\ \omega3 & if \quad rand\ge\frac{2}{3}\end{cases} \tag{7}$$

$$\omega1=\begin{cases}1 & if \quad (W[alpha]+step[alpha])\ge1\\ 0 & otherwise\end{cases} \tag{8}$$

$$step[alpha]=\begin{cases}1 & if \quad sigmoid(10(A_1.D[alpha]-.5))>rand\\ 0 & otherwise\end{cases} \tag{9}$$

Same equations (6)-(9) are used for other leader wolves- beta, delta and omega with relevant substitutions.

PrGWO: In PrGWO, firstly instead of using only three leaders four active leader wolves have been taken. To each non-leader wolf this provides more variety; availability of increased number of leader position vectors for updating their own position vector.

Secondly, the criteria of relative importance of individual four leaders in influencing updation of positions of all the other non-leader wolves was incorporated. For instance, alpha wolf is the best leader among all leader wolves hence it needs to be given higher weight influence than the other three. Similarly beta, delta and omega wolves need to be given weightage in order of their importance. Unlike proposed in the present paper, the original GWO and bGWO had given equal weight influence to all the wolves disregarding the fact that the alpha wolf is the most important leader in terms of acquiring best position followed by beta, delta and omega wolves respectively. In this proposed algorithm, different impact factor to different wolves viz alpha, beta, delta and omega wolves is proposed.

The mathematical model used in the proposed algorithm is as:

$$W_i(iter+1)=\begin{cases}\omega1 & if \quad rand<\frac{1}{2}\\ \omega2 & if \quad \frac{1}{2}\le rand<\frac{3}{4}\\ \omega3 & if \quad \frac{3}{4}\le rand<\frac{9}{10}\\ \omega4 & otherwise\end{cases} \tag{10}$$

$$\omega1=\begin{cases}1 & if \quad (sigmoid(W[alpha]-A_1D[alpha]))\ge rand\\ 0 & otherwise\end{cases} \tag{11}$$

$$\omega2=\begin{cases}1 & if \quad (sigmoid(W[beta]-A_2D[beta]))\ge rand\\ 0 & otherwise\end{cases} \tag{12}$$

$$\omega3=\begin{cases}1 & if \quad (sigmoid(W[delta]-A_3D[delta]))\ge rand\\ 0 & otherwise\end{cases} \tag{13}$$

$$\omega4=\begin{cases}1 & if (sigmoid(W[omega]-A_4D[omega]))\ge rand\\ 0 & otherwise\end{cases} \tag{14}$$

ωi=1...4 are binary vectors and W_i (iter+1) are the updated position vectors of individual non-leader wolves. In equation no.10 different weights of influence have been assigned to different leader wolves based on their relative importance. For example, if the random number (generated in range [0,1]) is less than 0.5 the updation of non-leader wolf is based on alpha wolf. This weight of influence is reduced respectively for the beta, delta and omega wolves as 0.25, 0.15 and 0.10 respectively. This is contrary to the original bGWO where the weight influence for all the wolves was uniformly distributed as 0.33.

Thirdly, for alpha, beta, delta and omega wolves (leaders guiding the other wolves)a different criteria for updating of their own positions is proposed. Each leader wolf updates its position relative to its own, its predessesors and its immediate successor. This was done to ensure that already better position holding wolves do not get deviated under influence of other less efficient successor leader wolves except its immediate successor. Immediate succcessor was required as in absence of it the alpha wolf position would not have updated and exploration of search space would have been restricted. Accordingly, updation of alpha wolf position was done relative to its own and beta wolf's position, updation of beta wolf's position was done with respect to alpha, beta and delta wolves' position, delta wolf position was done with respect to positions of alpha, beta, delta and omega wolves while for omega wolf it was with respect to all leader wolves. For updation of alpha,beta, delta wolves equations 15-17 were used.

$$W[alpha](iter+1)=Crossover(\omega1,\omega2) \tag{15}$$

$$W[beta](iter+1)=Crossover(\omega1,\omega2,\omega3) \tag{16}$$

$$\mathrm{W}[delta](iter+1)=Crossover(\omega1,\omega2,\omega3,\omega4) \tag{17}$$

$$Crossover(\omega1,\omega2)=\begin{cases}\omega1 & if \quad rand<\frac{2}{3}\\ \omega2 & otherwise\end{cases} \tag{18}$$

$$Crossover(\omega1,\omega2,\omega3)=\begin{cases}\omega1 & if \quad rand<\frac{1}{2}\\ \omega2 & if \quad \frac{1}{2}\le rand<\frac{4}{5}\\ \omega3 & otherwise\end{cases} \tag{19}$$

$$Crossover(\omega1,\omega2,\omega3,\omega4)=\begin{cases}\omega1 & if \quad rand<\frac{1}{2}\\ \omega2 & if \quad \frac{1}{2}\le rand<\frac{3}{4}\\ \omega3 & if \quad \frac{3}{4}\le rand<\frac{9}{10}\\ \omega4 & otherwise\end{cases} \tag{20}$$

Omega wolf can also be updated similar to delta wolf using equation 17. From eq 18-20 it can be seen that for updation of alpha wolf's position, ratio of 0.66:0.33 in terms of weight of influence is used between alpha and beta wolves. Similarly for beta wolf's updation ratio used is 0.5:0.3:0.2. In case of the delta wolf's updation, the weights of influence used are 0.5:0.25:0.15:0.10.

Fig. 2 depicts the PrGWO in graphical form while algorithmic depiction is presented as Algorithm 2. In this algorithm the initialized N feature vectors corresponding to N wolves are used to create N models. These models are then applied to the test data to classify the data. Based on the classification report and fitness function, fitness value of each wolf's feature vector is calculated. The fitness values are sorted in ascending order and the position/feature vector corresponding to the best fitness value is assigned as alpha position. Similarly the

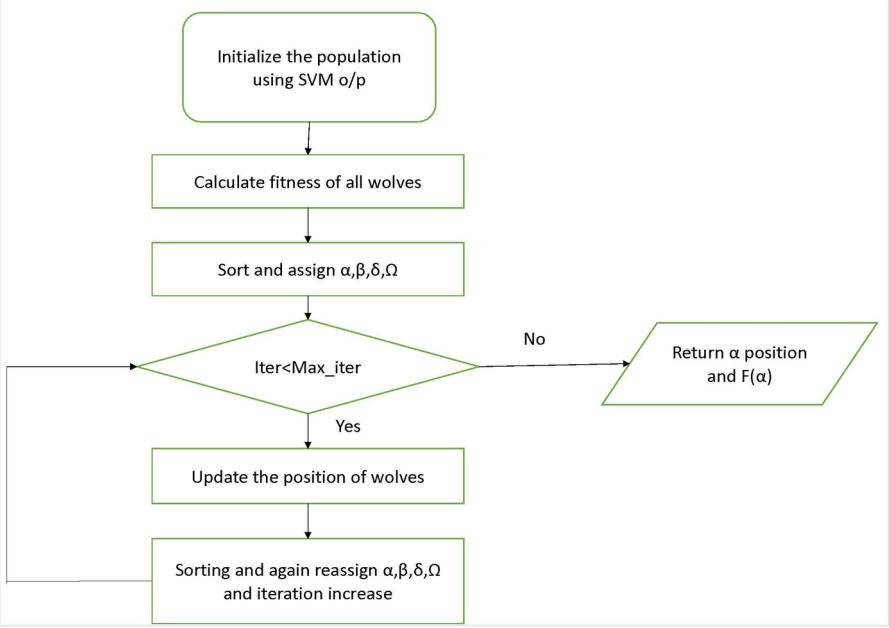

Fig. 2. Flowchart of PrGWO

second best is assigned as beta, third best as delta and fourth one as omega. In the subsequent iteration these values(alpha, beta, delta and omega) are used to update the position vectors of other wolves and their own using equations enlisted in the algorithm. Again the fitness value of each wolf's updated feature vector is calculated. Based on these newly sorted fitness values, reassignment of alpha, beta, delta and omega is done. The cycle is repeated till the last iteration. At the end of last iteration the alpha position vector is returned as optimal reduced feature vector.

3.4. Proposed fitness function

Fitness function has an important role in determining the optimal number of features for use in classification algorithms like SVM and kNN without compromising on performance metrics. For systematic initialization of the Gray wolves in the second layer, SVM is used as the wrapper method in the first layer. In this first layer, the accuracy measure has been used for getting the reduced feature set. This is done to ensure that the most important measure of efficiency is accounted for exclusively in the first layer; the other measures being accommodated in the second layer later on without sacrificing accuracy.

Thus the fitness function used in first layer is:

$$fit_{1layer} = 1 - \left(\frac{cor}{tot} \right) \tag{21}$$

where fit_{1layer} is the fitness function of first layer, *cor* is the correctly predicted instances and *tot* are the total number of instances. As for the second layer wherein PrGWO along with kNN is being used there are other important metrics like Detection rate, False positive rate and number of features which needs to be incorporated to make the fitness function truly inclusive. However, recognizing the accuracy as the most important measure for ensuring good security here again, a fitness model is proposed which gives higher weightage to the accuracy.

Algorithm 2: PrGWO

Input:
Iteration = max_iteration
Initialized W_i (i=1....N)
FitnessFn
Wrapper method {SVM,kNN}

Output:
W[alpha] \⋆ Optimal feature set got after the algorithm is complete ⋆\
f(W[alpha]) \⋆ Fitness value corresponding to optimal feature set ⋆\

Steps:
1.Initialize A
2. For all W_i
2.1 $Model_i$ = Classifier(D_{train})
2.2 $Predict_i$ = Classifier(Model$_i$, D_{test})
2.3 Fitness value $fit_i = Fitnessfn(Predict_i)$
3. Sort(fit_i),corresponding W_i and assign:
W[alpha]= W_i corresponding to best(fit_i)
W[beta] = W_i corresponding to 2^{nd} best(fit_i)
W[delta]= W_i corresponding to 3^{rd} best(fit_i)
W[omega]= W_i corresponding to 4^{th} best(fit_i)
4. While(iterations < max_iteration)
4.1. For i=1 to N do
if (W_i ==W[alpha]) Apply eq. 15
elseif (W_i ==W[beta]) Apply eq. 16
elseif (W_i ==W[delta] or W[omega]) Apply eq. 17
Else Apply eq. 10
Endif
endfor
4.2.Update A
4.3.Repeat step 2.1-2.3
4.4.Sort(fit_i),corresponding W_i and reassign W[alpha], W[beta], W[delta] and W[omega].
Return W[alpha] and f(W[alpha])

The proposed fitness function is as:

$$fit_{2layer} = a.error + \left(\frac{1-a}{2}\right).DR_{error} + \left(\frac{1-a}{2}\right).FPR + \left(b.Nfeat\right) \quad (22)$$

$$error = 1 - \frac{cor}{tot} \quad (23)$$

$$DR_{error} = 1 - \frac{Tpos}{Tpos + Fneg} \quad (24)$$

$$FPR = \frac{Fpos}{Tneg + Fpos}. \quad (25)$$

where fit_{2layer} is the fitness function of second layer, $Tpos$ is True-Positive cases, $Fneg$ is false negative, $Fpos$ is false-positive and $Tneg$ is true-negative. The last term $Nfeat$ is the number of features. Multiplying by b factor is necessary to scale this term to the level of other terms in the expression. Lastly, in the model the goal of the algorithm is to minimize the fitness function. The least valued fitness function will be assigned as alpha wolf and its space position will be the best feature set.

3.5. Performance metrics used

Present work was tested using various quality parameters. Equation(26)-(34) depicts these parameters:

$$Acc = \frac{True\_pos + True\_neg}{True\_pos + False\_pos + True\_neg + False\_neg} \quad (26)$$

$$DR = \frac{True\_pos}{True\_pos + False\_neg} \quad (27)$$

$$FPR = \frac{False\_pos}{False\_pos + True\_neg} \quad (28)$$

$$TNR = \frac{True\_neg}{False\_pos + True\_neg} \quad (29)$$

$$FNR = \frac{False\_neg}{True\_pos + False\_neg} \quad (30)$$

$$PR = \frac{True\_pos}{False\_pos + True\_pos} \quad (31)$$

$$F1 - score = \frac{2 \times DR \times PR}{DR + PR} \quad (32)$$

$$Feat = No.\ of\ \ features \quad (33)$$

$$Ratio\left(R\right) = \frac{Acc}{Feat} \quad (34)$$

True positive signifies that the data point was predicted as malicious and actually it was malicious. False Positive signifies that the data point was predicted as malicious, however actually it was not. True negative signifies that the data point was predicted as not malicious and actually it was not malicious. False negative signifies that the data point was predicted as not malicious, however it was actually malicious.

Detection Rate (DR) is also known as recall and TPR. It identifies the number of times the classifier predicted a 'positive' result over the number of times positive results were to be predicted.

Precision (PR) determines the measure of correctness of results got. Thus it uses true positive over true positive and false positive combined.

F1-score is used when class distribution is not balanced. It calculates the weighted average of detection rate and precision.

4. Experimental results and discussion:

Experiments were performed using Matlab 2021 and Python programming language on anaconda distribution over the Intel Core i7-10th generation machine.

Table 2. Table of parameters

Parameter name	Parameter values
No. of runs	10
No. of population-N	5,7,10
No. of iterations	25
Technique for feature selection in 1st layer	SVM(RBF)
Fitness function for SVM-RBF	fit_{1layer}
Technique used FS and classification in second layer	PrGWO with kNN
Fitness function for PrGWO-kNN	fit_{2layer}
a in fitness function- fit_{2layer}	0.7
b in fitness function fit_{2layer} for optimal accuracy	0.0001
b in fitness function for optimal no. of features	0.001

4.1. Parameters used:

Various parameters were tested experimentally for finding the best combination of parameters. Table 2 illustrates these parameters. The algorithm was run 10 times, Number of wolves were taken to be 5,7 and 10, No of iterations within each run was 25, SVM was used with RBF, two layers were used with different fitness functions and values of b were taken as 0.001 and 0.0001. The reason for taking different values of b is to give different weightage to number of features in the fitness function. In table 3 variation in results due to size of population is depicted. Notationwise NoF in the table stands for No. of features. Here 3 values of N were taken: 5,7 and 10 wolves. The results were compared not only in regard to DR, FPR, F1-score etc but also accuracy and length of feature set finally selected by the algorithm. The results show that when the size of population was 7, the length of feature set was least and accuracy was maximum. Only two parameters-FPR and TNR were found to be better in case of N=10. Similarly the 'k' value in the kNN was experimented for k=1,3 and 5. Table 4 shows the variation in results with change in k value. For k=1, the experiments yielded best results in terms of accuracy i.e 99.60% alongwith other metrics However, the best length of feature vector i.e 8 was found for k=3 though the accuracy and other metrics were not better than k=1.

Hence based on experimental results it is appropriate to remark that a cost benefit analysis needs to be done in terms of accuracy and number of features while choosing value of k for real world scenarios.

Table 3. Variation of results with value of 'N' for NSL-KDD

N	Accu-racy	NoF	DR	FPR	TNR	FNR	Preci-sion	F1
5	99.46	13	99.44	0.34	99.66	0.55	99.45	99.44
7	**99.60**	**12**	**99.61**	0.28	99.72	**0.39**	**99.61**	**99.61**
10	99.47	14	99.45	**0.27**	**99.73**	0.54	99.46	99.46

Table 4. Variation of results with value of 'k' for NSL-KDD

k	NoF	Accuracy	DR	FPR	TNR	FNR	Precision	F1
1	12	**99.60**	**99.61**	**0.28**	**99.72**	**0.39**	**99.62**	**99.61**
3	**8**	99.32	99.33	0.53	99.47	0.67	99.30	99.32
5	10	99.26	99.26	0.53	99.47	0.74	99.25	99.25

Table 5. Classwise performance of PrGWO-SK for NSL-KDD dataset

Class	DR	FPR	TNR	FNR	Precision	F1 score
Normal	99.76	0.478	99.52	0.24	99.58	99.67
DoS	**99.89**	0.062	99.94	**0.11**	**99.89**	**99.89**
Probe	98.75	0.061	99.94	1.25	99.39	99.07
R2L	88.58	**0.036**	**99.964**	11.41	95.57	91.94
U2R	66.67	0.043	99.957	33.33	42.10	51.61

Table 6. Classwise performance of PrGWO-SK for DS2OS dataset

Class	DR	FNR	FPR	TNR	Precision	F1-score
DoS attack	99.45	0.55	0.0015	99.9985	99.48	99.46
Data Probing	99.75	0.25	0.0002	99.9998	99.76	99.75
Malicious control	99.6	0.4	0.0013	99.9987	99.41	99.50
Malicious operation	99.8	0.2	0.0025	99.9975	98.91	99.35
Scan	98.73	1.27	0.0054	99.9946	98.65	98.69
Spying	**99.84**	**0.16**	**0**	**100**	**99.77**	**99.80**
Wrongsetup	99.74	0.26	**0**	**100**	99.75	99.74
Normal	99.81	0.19	0.19	99.81	99.76	99.78

Table 7. Classwise performance of PrGWO-SK for BoTIoT dataset

Class	DR	FNR	FPR	TNR	Precision	F1-score
DoS	99.97	0.03	**0.0001**	**99.9999**	99.81	99.89
DDoS	**99.98**	**0.02**	**0.0001**	**99.9999**	**99.92**	**99.95**
Reconnaissance	99.89	0.11	0.0008	99.9992	99.79	99.84
Theft	96.2	3.8	0.0045	99.9955	99.15	97.65
Normal	99.2	0.8	0.0009	99.9991	98.9	99.05

4.2. Performance of PrGWO-SK for different classes:

The performance of present approach was tested on different classes of dataset. Table 5 shows the measures for the Detection rate, False positive rate, F1-score, Precision etc for all the classes. The experimental results show DR to be highest for DoS attacks being 99.89% while for R2L and U2R the DR was least among all. However, the False positive rate was found to be the best for these two classes viz. 0.036% and 0.043% among all. Considering the precision, the two classes -Normal and DoS performs to the extent of 99.58% and 99.89%.

As regards DS2OS dataset table 6 depicts the classification results for various classes present in the dataset. Most of the performance metrics shows best results for the spying class. In case of FPR and TNR both Wrongsetup and Spying classes shows the results as 0 and 100% respectively, i.e no false alarm is generated if the classes are neither Wrongsetup or Spying. However, the algorithm performs worst for the Scan class where the Detection rate was measured as 98.73% in contrast to the other classes where the DR was measured more than 99%.

Similar to DS2OS and NSl-KDD the classwise performance of the BoTIoT dataset was also evaluated. From table 7 it can be seen that for DDoS the algorithm performs best in terms of DR at 99.98%, FPR at 0.0001%, precision at 99.92% and F1-score at 99.95%. The class DoS shows results very close to the DDoS class. The least Detection rate was recorded for class 'Theft' at 96.2% which shows the difficuly in correctly predicting this class of attack.

4.3. Convergence of algorithm:

Table 8 depicts the various performance measures iteration-wise for b=0.0001. For this value of b the algorithm focusses more on getting optimal accuracy even at the cost of increasing the length of feature vector.

Similarly table 9 depicts the experimental results for b=0.001. Here the focus shifts more towards optimizing the length of feature vector.

Table 8. Iteration wise results for NSL-KDD with b=0.0001

Iteration	Features	Accuracy	DR	FPR	Precision	F1-score
1	19	99.506	99.510	0.306	99.500	99.505
5	15	99.569	99.572	0.268	99.581	99.576
10	15	99.565	99.568	**0.241**	99.577	99.572
15	13	99.577	99.579	0.275	99.586	99.582
16	13	99.577	99.579	0.275	99.586	99.582
17	12	99.577	99.579	0.275	99.586	99.582
18	13	99.577	99.579	0.255	99.586	99.582
19	13	99.577	99.579	0.255	99.586	99.582
20	13	99.577	99.579	0.255	99.586	99.582
21	13	99.577	99.579	0.255	99.586	99.582
22	13	99.585	99.587	0.257	99.597	99.592
23	**12**	**99.603**	**99.606**	0.284	**99.616**	**99.611**
24	12	99.603	99.606	0.284	99.616	99.611
25	12	99.603	99.606	0.284	99.616	99.611

Table 9. Iteration wise results for NSL-KDD with b=0.001

Iteration	Feature	Accuracy	DR	FPR	Precision	F1-score
1	18	98.810	98.790	0.560	98.890	98.840
5	14	98.790	98.800	0.501	98.870	98.835
10	14	99.202	99.193	0.565	99.191	99.192
15	14	99.202	99.193	0.565	99.191	99.192
16	13	99.212	99.193	0.565	99.207	99.200
17	13	99.212	99.193	0.565	99.207	99.200
18	15	99.345	99.337	**0.425**	99.340	99.339
19	12	99.345	99.337	0.441	99.338	99.338
20	11	99.345	99.337	0.447	99.338	99.338
21	8	99.342	99.343	0.445	99.331	99.337
22	8	99.342	99.343	0.445	99.331	99.337
23	8	99.342	99.343	0.445	99.331	99.337
24	**8**	**99.361**	**99.367**	**0.568**	**99.366**	**99.367**
25	8	99.361	99.367	0.568	99.366	99.367

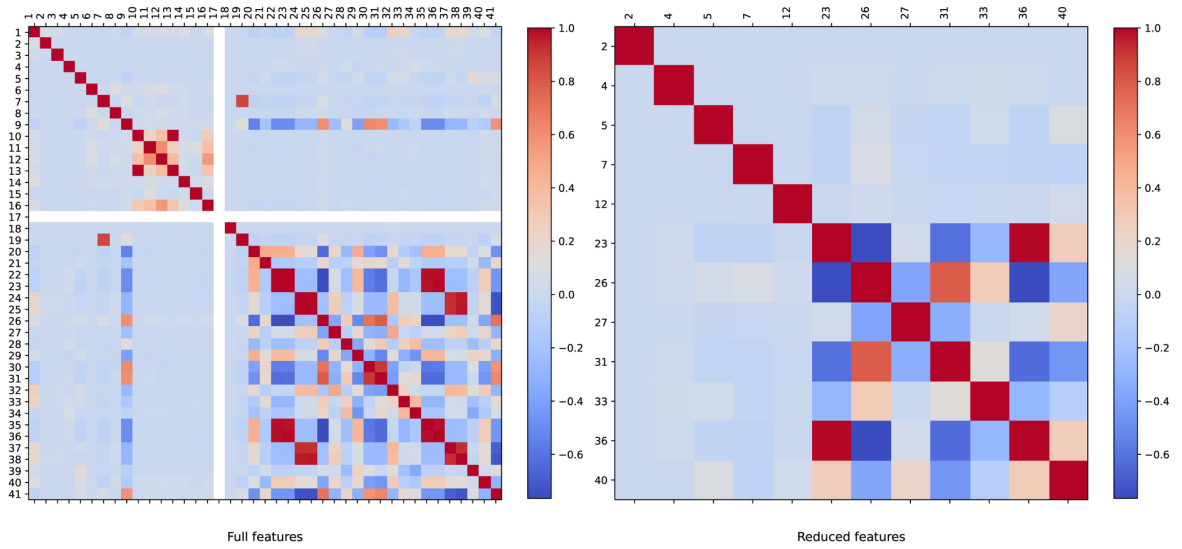

Fig 3. Original vs Reduced features Correlation heatmap of the NSL-KDD dataset

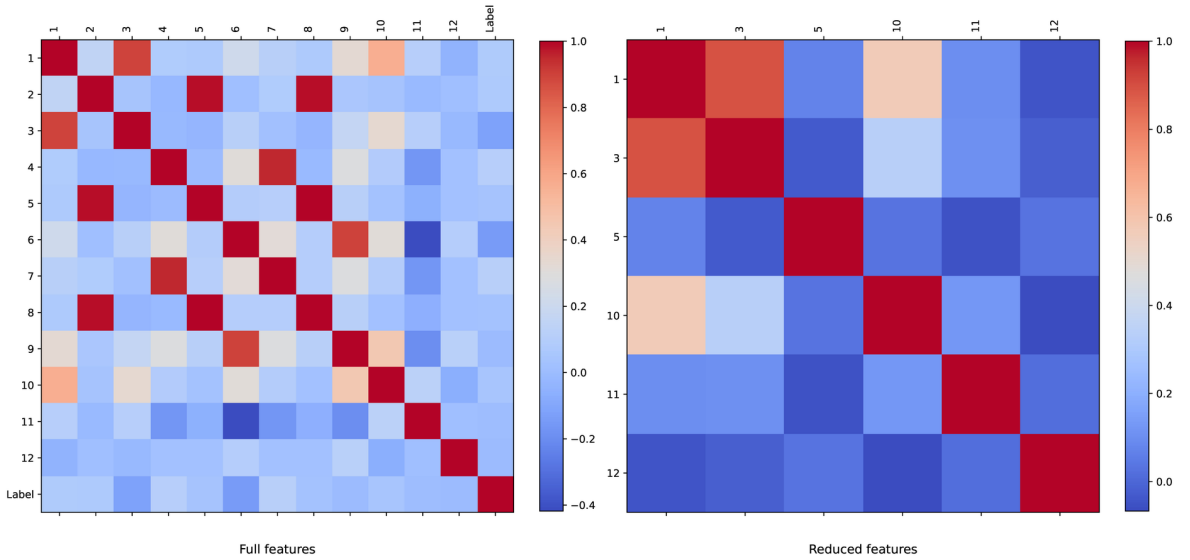

Fig 4. Original vs Reduced features Correlation heatmap of the DS2OS dataset

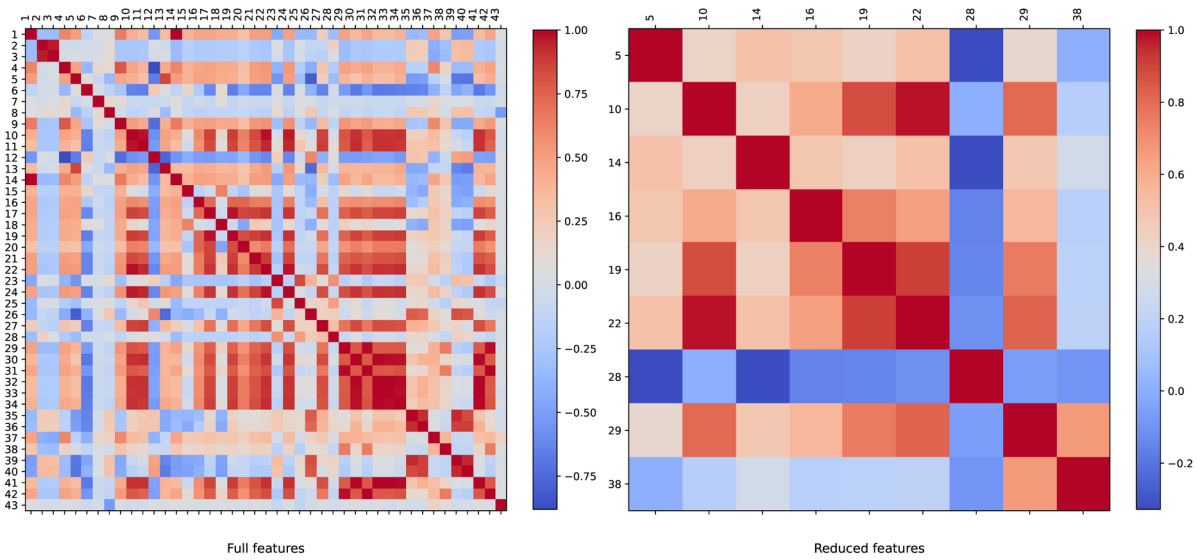

Fig 5. Original vs Reduced features Correlation heatmap of the BoTIoT dataset

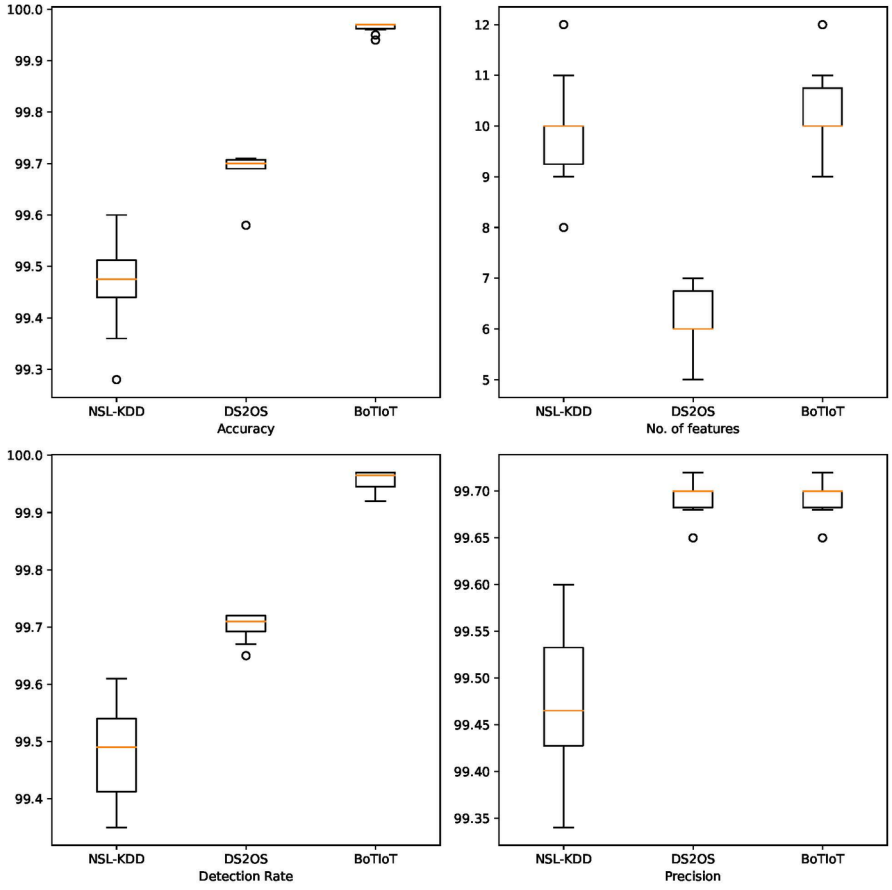

Fig 6. Boxplots showing variation of experimental results for NSL-KDD, DS2OS and BoTIOT

4.4. Correlation of original dataset vs reduced dataset

Figures 3, 4 and 5 shows the correlation heatmap for NSL-KDD, DS2OS and BoTIoT datasets respectively. The figures show both the correlation matrix of the original feature set vis-a-vis reduced feature set. Correlation is an important measure of finding the similarity between the features in pair. For optimizing the length of feature vector, similar or correlated features needs to be removed as they do not add any additional characterstic for classification task. The figures mentioned above shows how the final feature set outputted by the proposed algorithm does not contain the strongly correlated features i.e the algorithm is largely able to remove the correlated features. Thus this test can be taken as statistical test for validating the proposed approach.

4.5. Consistency of experimental results through box plots

The values shown in the tables in this paper are the results of the best run during 10 runs of the algorithm. However, an algorithm's robustness and efficiency can be measured by its consistency over several runs. Fig 6 depicts the median and the interquartile range of the different runs of experimental results graphically. For NSL-KDD dataset

Table 10. DR -Classwise comparison of existing algorithm with proposed algorithm for NSL-KDD dataset

Work	Normal	Probe	DoS	U2R	R2L
Pajouh et al. [25]	94.56	79.76	84.68	67.16	34.81
Wu et al. [43]	95.31	83.4	83.49	11.55	24.69
Gao et al. [7]	94.33	87.11	84.37	25	55.27
Yang et al. [44]	97.38	73.94	81.09	6.5	17.25
Zhang et al. [47]	97.03	80.38	83.95	32.84	11.26
Pajouh et al. [26]	94.43	87.32	88.2	70.15	42
Tian et al. [39]	94.9	98.26	98.04	75.5	95.24
Su et al. [34]	97.5	85.76	87.55	20.95	44.25
Mahfouz et al. [20]	99.5	91.6	99.2	39.3	55.1
RF [18]	98.01	**99.41**	99.71	93.22	**97.08**
Xgboost [18]	99	96.86	93.68	83.92	80.11
KNN [18]	98	98.25	99.69	**93.36**	95.23
Proposed work	**99.76**	98.75	**99.89**	66.67	88.58

Table 11. DR-Classwise comparison of existing algorithm with proposed algorithm for DS2OS dataset

Works	DoS	Probe	MC	MO	Scan	Spying	WS	Normal
Pahl et al. [24]	68	**100**	32	66	54	13	**100**	95
Hasan et al. [11]	66	92	97	**100**	95	93	**100**	**100**
Hasan et al. [11]	66	92	92	56	71	4	**100**	**100**
RF [18]	66	**100**	97	**100**	97	**100**	**100**	**100**
Xgboost [18]	66	**100**	99	**100**	**100**	**100**	**100**	**100**
KNN [18]	66	**100**	99	**100**	98	**100**	**100**	**100**
Proposed work	**99.45**	99.75	**99.6**	99.8	98.73	99.84	99.74	99.81

Table 12. DR-Classwise comparison of existing algorithm with proposed algorithm for BoTIoT dataset

Work	Normal	DDoS	DoS	Reconnaissance	Theft
Shafiq et al.[29]	75	98	**100**	81	93
Soe et al. [33]	0	**100**	**100**	**100**	0
RF [18]	**100**	**100**	99	**100**	93
Xgboost [18]	**100**	**100**	**100**	**100**	93
Proposed Work	99.2	99.98	99.97	99.89	**96.2**

Table 13. Comparative analysis of works reported in the literature for NSL-KDD

IDS	No. of features	Accuracy	Ratio
GHSOM [46]	41	96.02	2.34
A-GHSOM [15]	41	96.63	2.35
Kayacik [16]	41	90.04	2.19
DT-EnSVM2 [10]	41	99.41	2.42
NB-SVM2 [9]	41	99.36	2.42
S-NDAE+RF [31]	28	85.42	3.05
PCA+GHSOM-pr [13]	30	87.23	2.90
FDR+ kernel PCA [14]	23	90	3.91
NGSA +GHSOM-pr [41]	27	98.61	3.65
GWOSVM-IDS [28]	12	96	8
NNIA+GHSOM-pr [41]	24	99.47	4.14
kNN+GR+RFMDA [18]	18	98.67	5.48
Decision tree based [38]	16	98.38	6.15
SIGMOID_PIO [1]	18	86.9	4.83
RF+PSO [19]	10	99.32	9.93
Proposed PrGWO-SK(b=0.001)	8	99.361	12.42
Proposed PrGWO-SK(b=.0001)	12	99.60	8.30

Table 14. Comparative analysis of works reported in the literature for DS2OS

IDS	No. of features	Accuracy	Ratio
Pahl et al. [24]	12	96.3	8.025
Hasan et al. [11]	11	99.35	9.031
RF [18]	6	99.4	16.566
Xgboost [18]	6	99.43	16.571
KNN [18]	6	99.4	16.566
Proposed work	6	99.71	16.618

Table 15. Comparative analysis of works reported in the literature for BoTIoT

IDS	No. of features	Accuracy	Ratio
Shafiq et al. [29]	43	98.35	2.287
Soe et al. [33]	8	99.1	12.387
RF [18]	10	99.99	9.999
Xgboost [18]	10	99.99	9.999
KNN [18]	10	85.92	8.592
Proposed work	9	99.97	11.107

the median value of accuracy is 99.475%, while the minimum value and maximum values are 99.6% and 99.28% respectively. However, this minimum value is the outlier as shown in the box plot and the mean is 99.463%. Similarly for DS2OS, the median, mean, minimum and maximum values are 99.7%, 99.688%, 99.58% and 99.71% respectively. For BoTIoT these values are 99.97%, 99.964%, 99.94% and 99.97%. The above results for all the datasets shows the consistency in performance of the algorithm except one or two outliers. Considering the number of features, NSL-KDD has median of 10, mean of 9.8 , minimum as 8 and maximum as 12. For DS2OS dataset these are 6, 6.2, 5 and 7 and for BoTIoT these are 10, 10.2, 9 and 11 respectively in order of median, mean, minimum and maximum. Similarly for DR, NSL-KDD has median of 99.58%, mean of 99.48%, minimum as 99.35% and maximum as 99.61%. For DS2OS median is 99.71%, mean is 99.70, minimum is 99.65% and maximum is 99.72%. For BotIoT median is 99.94%, mean is 99.96%, minimum is 99.92% and maximum is 99.97%. Regarding Precision values for NSLKDD median is 99.58%, mean is 99.47%, minimum is 99.34% and maximum is 99.6%. For DS2OS median is 99.7%, mean is 99.69%, minimum is 99.65% and maximum is 99.72%. Lastly for BoTIoT median is 99.96%, mean is 99.96%, minimum is 99.95% and maximum is 99.97%.

4.6. Comparison of Detection rates achieved by different algorithms with proposed algorithm for different classes:

For measuring effectivess of any algorithm it is important to compare it with other existing algorithms in terms of common performance metrics. Table 10, 11 and 12 shows the classwise performance of the proposed algorithm compared to other algorithms. Table 10 shows that the proposed approach outperforms for 'Normal' and 'DoS' classes giving 99.76% and 99.89% while for other classes different algorithms performs better. Table 11 shows that the proposed approach performs best for the DoS and MC attacks. For DoS it gives DR as 99.45% which is much better than the second best of 66%. For MC it gives 99.6%. Similarly, table 12 shows that the proposed approach performs best in terms of detecting "theft" attack type giving DR as 96.2% while the second best was 93%.

4.7. Comparison in terms of Accuracy and Length of feature set of different algorithms with proposed algorithm:

Tables 13, 14 and 15 shows the comparison of the proposed approach with others' work in terms of accuracy, number of features and the ratio. Table 13 shows that in terms of the accuracy two best performing algorithms are NNIA+GHSOM-pr (99.47%) and DT-EnSVM2 (99.41%). However, proposed approach has been able to outperform these efficient algorithms measuring 99.60%. In terms of length of feature vector the best performing algorithm is RF+PSO giving output feature vector of length 10. In this case also the proposed work outperforms RF+PSO giving vector of length 8. However, in this case the accuracy reduces to 99.36%.

Table 14 shows that most of the algorithms gave accuracy close to 99.43% with length of feature vector as low as 6. Though PrGWO-SK was not able to reduce the length of feature vector further below 6 but the accuracy showed substantial improvement measuring 99.71%.

Similarly table 15 shows the results for BoTIoT dataset. Here the best accuracy was achieved by Kumar et al. as 99.99% with feature length as 10. Considering the optimal feature length the best was achieved as 8 by Soe et al. However, the accuracy measured in this case was only 99.1%. Proposed approach was able to measure accuracy as 99.97% with length of feature vector as 9. Thus the proposed approach was able to outperform these algorithms - first algorithm in terms of length without compromising much on accuracy and second algorithm in terms of accuracy by a substantial margin.

5. Conclusions and limitations

This paper aimed to find reduced optimal feature vector of a traffic dataset to reduce the computational complexity by removing the data dimensionality curse. A two layered structure was used - first layer employed the SVM technique while second layer used the kNN as wrapper technique. For searching the solution space swarm intelligence based modified form of GWO called PrGWO was proposed alongwith two different fitness functions. The effectiveness of the experimental results were established through use of metrics - Accuracy, Detection rate, FPR, TPR, Precision. Through extensive experiments it was found that the proposed methodology and algorithm performed better for several individual classes -Normal, DoS for NSl-KDD, DoS and MC for DS2OS datasets and Theft for BoTIoT dataset. In terms of overall accuracy, the PrGWO-SK performed better for NSL-KDD at the same time length of feature vector was also reduced. For DS2OS, though the length of feature vector could not be reduced still notable increase in accuracy was witnessed. In case of BoTIoT the combination of accuracy and length of feature vector was effectively optimal. Althougth the present work performs better as mentioned above, it has shown limitations in detecting classes like U2R , R2L where the DR was not found to be better than some of the existing algorithms. Hence as a future work classification of classes like U2R and R2L can be pursued futher. Moreover, the authors intend to take up parameter tuning as further research objective in future.

References

1. Alazzam H, Sharieh A, Sabri KE. A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer. Expert systems with applications 2020; 148: 113249, https://doi.org/10.1016/j.eswa.2020.113249.
2. Almiani M, AbuGhazleh A, Al-Rahayfeh A, Atiewi S, Razaque A. Deep recurrent neural network for IoT intrusion detection system. Simulation Modelling Practice and Theory 2020; 101: 102031, https://doi.org/10.1016/j.simpat.2019.102031.
3. Antunes Rodrigues J, Torres Farinha J, Mendes M, Mateus R, Marques Cardoso A. Short and long forecast to implement predictive

maintenance in a pulp industry. Eksploatacja i Niezawodnosc – Maintenance and Reliability 2022; 24(1): 33–41, http://doi.org/10.17531/ein.2022.1.5.
4. Baranowski J. Predicting IoT failures with Bayesian workflow. Eksploatacja i Niezawodnosc – Maintenance and Reliability 2022; 24 (2): 248–259, http://doi.org/10.17531/ein.2022.2.6.
5. Emary E, Zawbaa HM, Hassanien AE. Binary grey wolf optimization approaches for feature selection. Neurocomputing 2016; 172: 371-381, https://doi.org/10.1016/j.neucom.2015.06.083.
6. Gao H, Qiu B, Barroso RJ, Hussain W, Xu Y, Wang X. TSMAE: a novel anomaly detection approach for internet of things time series data using memory-augmented autoencoder. IEEE Transactions on Network Science and Engineering(Early Access) 2022: 1, https://doi.org/10.1109/TNSE.2022.3163144.
7. Gao X, Shan C, Hu C, Niu Z, Liu Z. An adaptive ensemble machine learning model for intrusion detection. IEEE Access 2019; 7: 82512-82521, https://doi.org/10.1109/ACCESS.2019.2923640.
8. Golrang A, Golrang AM, Yildirim Yayilgan S, Elezaj O. A novel hybrid IDS based on modified NSGAII-ANN and random forest. Electronics 2020; 9(4): 577, https://doi.org/10.3390/electronics9040577.
9. Gu J, Lu S. An effective intrusion detection approach using SVM with naïve Bayes feature embedding. Computers & Security 2021; 103: 102158, https://doi.org/10.1016/j.cose.2020.102158.
10. Gu J, Wang L, Wang H, Wang S. A novel approach to intrusion detection using SVM ensemble with feature augmentation. Computers & Security 2019; 86: 53-62, https://doi.org/10.1016/j.cose.2019.05.022.
11. Hasan M, Islam MM, Zarif MI, Hashem MM. Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. Internet of Things 2019; 7: 100059, https://doi.org/10.1016/j.iot.2019.100059.
12. Hodo E, Bellekens X, Hamilton A, Dubouilh PL, Iorkyase E, Tachtatzis C, Atkinson R. Threat analysis of IoT networks using artificial neural network intrusion detection system. 2016 IEEE International Symposium on Networks, Computers and Communications (ISNCC), Hammamet, IEEE 2016: 1-6, https://doi.org/10.1109/ISNCC.2016.7746067.
13. Hoz ED, Hoz ED, Ortiz A, Ortega J, Martínez-Álvarez A. Feature selection by multi-objective optimisation: Application to network anomaly detection by hierarchical self-organising maps. Knowledge Based Systems 2014; 71: 322-338, https://doi.org/10.1016/j.knosys.2014.08.013.
14. Hoz ED, Ortiz A, Ortega J, Hoz ED. Network anomaly classification by support vector classifiers ensemble and non-linear projection techniques. 2013 International Conference on Hybrid Artificial Intelligence Systems, Berlin, Springer 2013: 103-111, https://doi.org/10.1007/978-3-642-40846-5_11.
15. Ippoliti D, Zhou X. An adaptive growing hierarchical self organizing map for network intrusion detection. 2010 IEEE 19th International Conference on Computer Communications and Networks, Zurich, IEEE 2010: 1-7, https://doi.org/10.1109/ICCCN.2010.5560165.
16. Kayacik HG, Zincir-Heywood AN, Heywood MI. A hierarchical SOM-based intrusion detection system. Engineering applications of artificial intelligence 2007; 20(4): 439-51, https://doi.org/10.1016/j.engappai.2006.09.005.
17. Koroniotis N, Moustafa N, Sitnikova E, Turnbull B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-Iot dataset. Future Generation Computer Systems 2019; 100: 779-796, https://doi.org/10.1016/j.future.2019.05.041.
18. Kumar P, Gupta GP, Tripathi R. Toward design of an intelligent cyber attack detection system using hybrid feature reduced approach for iot networks. Arabian Journal for Science and Engineering 2021; 46(4): 3749-3778, https://doi.org/10.1007/s13369-020-05181-3.
19. Kunhare N, Tiwari R, Dhar J. Particle swarm optimization and feature selection for intrusion detection system. Sādhanā 2020; 45(1): 1-4, https://doi.org/10.1007/s12046-020-1308-5.
20. Mahfouz AM, Venugopal D, Shiva SG. Comparative analysis of ML classifiers for network intrusion detection. 2020 4th International Congress on Information and Communication Technology, Singapore, Springer 2020: 193-207, https://doi.org/10.1007/978-981-32-9343-4_16.
21. Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Advances in Engineering Software 2014; 69: 46-61, https://doi.org/10.1016/j.advengsoft.2013.12.007.
22. Modi C, Patel D. A feasible approach to intrusion detection in virtual network layer of Cloud computing. Sādhanā 2018; 43(7): 1-6, https://doi.org/10.1007/s12046-018-0910-2.
23. Pahl MO, Aubet FX. DS2OS traffic traces . [https://www.kaggle.com/francoisxa/ds2ostraffictraces].
24. Pahl MO, Aubet FX, Liebald S. Graph-based IoT microservice security. 2018 IEEE/IFIP Network Operations and Management Symposium(NOMS), Taipei, IEEE 2018: 1-3, https://doi.org/10.1109/NOMS.2018.8406118.
25. Pajouh HH, Dastghaibyfard G, Hashemi S. Two-tier network anomaly detection model: a machine learning approach. Journal of Intelligent Information Systems 2017; 48(1): 61-74, https://doi.org/10.1007/s10844-015-0388-x.
26. Pajouh HH, Javidan R, Khayami R, Dehghantanha A, Choo KK. A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. IEEE Transactions on Emerging Topics in Computing 2016; 7(2): 314-23, https://doi.org/10.1109/TETC.2016.2633228.
27. Rathore S, Park JH. Semi-supervised learning based distributed attack detection framework for IoT. Applied Soft Computing 2018; 72: 79-89, https://doi.org/10.1016/j.asoc.2018.05.049.
28. Safaldin M, Otair M, Abualigah L. Improved binary gray wolf optimizer and SVM for intrusion detection system in wireless sensor networks. Journal of Ambient Intelligence and Humanized Computing 2021; 12(2): 1559-76, https://doi.org/10.1007/s12652-020-02228-z.
29. Shafiq M, Tian Z, Sun Y, Du X, Guizani M. Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city. Future Generation Computer Systems 2020; 107: 433-42, https://doi.org/10.1016/j.future.2020.02.017.
30. Shams EA, Rizaner A, Ulusoy AH. Trust aware support vector machine intrusion detection and prevention system in vehicular adhoc networks. Computers & Security 2018; 78: 245-254, https://doi.org/10.1016/j.cose.2018.06.008
31. Shone N, Ngoc TN, Phai VD, Shi Q. A deep learning approach to network intrusion detection. IEEE transactions on Emerging topics in Computational Intelligence 2018; 2(1): 41-50, https://doi.org/10.1109/TETCI.2017.2772792.
32. Sivapalan G, Nundy KK, Dev S, Cardiff B, John D. ANNet: a lightweight neural network for ECG anomaly detection in IoT edge sensors. IEEE Transactions on Biomedical Circuits and Systems 2022; 16(1): 24-35, https://doi.org/10.1109/TBCAS.2021.3137646.
33. Soe YN, Feng Y, Santosa PI, Hartanto R, Sakurai K. Towards a lightweight detection system for cyber attacks in the IoT environment using corresponding features. Electronics 2020; 9(1): 144, https://doi.org/10.3390/electronics9010144.

34. Su T, Sun H, Zhu J, Wang S, Li Y. BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset. IEEE Access 2020; 8: 29575-29585, https://doi.org/10.1109/ACCESS.2020.2972627.
35. Tama BA, Comuzzi M, Rhee KH. TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system. IEEE Access 2019; 7: 94497-94507, https://doi.org/10.1109/ACCESS.2019.2928048.
36. Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD CUP 99 data set. 2009 IEEE symposium on Computational Intelligence for Security and Defense Applications, Ottawa, IEEE 2009: 1-6, https://doi.org/10.1109/CISDA.2009.5356528.
37. Tavallaee M, Bagheri E, Lu W, Ghorbani AA. The NSL-KDD data set. [https://www.unb.ca/cic/datasets/nsl.html].
38. Teng S, Wu N, Zhu H, Teng L, Zhang W. SVM-DT-based adaptive and collaborative intrusion detection. IEEE/CAA Journal of Automatica Sinica 2017; 5(1): 108-18, https://doi.org/10.1109/JAS.2017.7510730.
39. Tian Q, Han D, Li KC, Liu X, Duan L, Castiglione A. An intrusion detection approach based on improved deep belief network. Applied Intelligence 2020; 50(10): 3162-3178, https://doi.org/10.1007/s10489-020-01694-4.
40. Vijayanand R, Devaraj D, Kannapiran B. Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. Computers & Security 2018; 77: 304-314, https://doi.org/10.1016/j.cose.2018.04.010.
41. Wei W, Chen S, Lin Q, Ji J, Chen J. A multi-objective immune algorithm for intrusion feature selection. Applied Soft Computing 2020; 95: 106522, https://doi.org/10.1016/j.asoc.2020.106522.
42. Wisanwanichthan T, Thammawichai M. A Double-Layered Hybrid Approach for Network Intrusion Detection System Using Combined Naive Bayes and SVM. IEEE Access 2021; 9: 138432-138450, https://doi.org/10.1109/ACCESS.2021.3118573.
43. Wu K, Chen Z, Li W. A novel intrusion detection model for a massive network using convolutional neural networks. IEEE Access 2018; 6: 50850-50859, https://doi.org/10.1109/ACCESS.2018.2868993.
44. Yang Y, Zheng K, Wu C, Niu X, Yang Y. Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks. Applied Sciences 2019; 9(2): 238, https://doi.org/10.3390/app9020238.
45. Yao H, Wang Q, Wang L, Zhang P, Li M, Liu Y. An intrusion detection framework based on hybrid multi-level data mining. International Journal of Parallel Programming 2019; 47(4): 740-758, https://doi.org/10.1007/s10766-017-0537-7.
46. Yu Z, Tsai JJ, Weigert T. An adaptive automatically tuning intrusion detection system. ACM Transactions on Autonomous and Adaptive Systems 2008; 3(3): 1-25, https://doi.org/10.1145/1380422.1380425.
47. Zhang C, Ruan F, Yin L, Chen X, Zhai L, Liu F. A deep learning approach for network intrusion detection based on NSL-KDD dataset. 2019 IEEE 13th International Conference on Anti-counterfeiting, Security, and Identification (ASID), Xiamen, IEEE 2019: 41-45, https://doi.org/10.1109/ICASID.2019.8925239.