Int. J. Appl. Math. Comput. Sci., 2023, Vol. 32, No. 2, 313–326
DOI: 10.34768/amcs-2023-0023

FEATURE OPTIMIZATION USING A TWO–TIER HYBRID OPTIMIZER IN AN INTERNET OF THINGS NETWORK

AKHILESHWAR PRASAD AGRAWAL [a,*], NANHAY SINGH [b]

[a]AIACT&R
Guru Gobind Singh Indraprastha University
Dseu, Delhi, India
e-mail: kpw.ce08@gmail.com

[b]Department of Computer Science and Engineering
Netaji Subhas University of Technology (East Campus)
Geeta Colony, Delhi 110031, India
e-mail: nsingh1973@gmail.com

The growing use of the Internet of Things (IoT) in smart applications necessitates improved security monitoring of IoT components. The security of such components is monitored using intrusion detection systems which run machine learning (ML) algorithms to classify access attempts as anomalous or normal. However, in this case, one of the issues is the large length of the data feature vector that any ML or deep learning technique implemented on resource-constrained intelligent nodes must handle. In this paper, the problem of selecting an optimal-feature set is investigated to reduce the curse of data dimensionality. A two-layered approach is proposed: the first tier makes use of a random forest while the second tier uses a hybrid of gray wolf optimizer (GWO) and the particle swarm optimizer (PSO) with the k-nearest neighbor as the wrapper method. Further, differential weight distribution is made to the local-best and global-best positions in the velocity equation of PSO. A new metric, i.e., the reduced feature to accuracy ratio (RFAR), is introduced for comparing various works. Three data sets, namely, NSLKDD, DS2OS and BoTIoT, are used to evaluate and validate the proposed work. Experiments demonstrate improvements in accuracy up to 99.44%, 99.44% and 99.98% with the length of the optimal-feature vector equal to 9, 4 and 8 for the NSLKDD, DS2OS and BoTIoT data sets, respectively. Furthermore, classification improves for many of the individual classes of attacks: denial-of-service (DoS) (99.75%) and normal (99.52%) for NSLKDD, malicious control (100%) and DoS (68.69%) for DS2OS, and theft (95.65%) for BoTIoT.

Keywords: IoT, anomaly mitigation, GWO, feature optimization, PSO.

1. Introduction

Internet of Things (IoT) applications have grown in popularity in the recent decade thanks to the introduction of low-cost sensor devices which record IoT data, and powerful servers which process these data to generate actionable events. One of these IoT applications is a smart air monitoring system, which uses numerous IoT sensors to track, measure, and record the amount of dangerous chemicals in the air (IoT data). The information gathered by the sensors is then sent to intelligent IoT servers, which estimate the air quality using various machine learning (ML) algorithms and alert the populace if air pollution levels exceed a predetermined threshold.

Though IoT applications are very useful, they are prone to security threats like worms, DoS, backdoors and other attacks by malicious entities. Such threats have the potential to devastate IoT services and smart devices in significant ways. For mitigating such attacks on IoT applications, one of the solutions is to prevent them through the use of an anomaly-based intrusion detection system (AIDS). Figure 1 shows a section of AIDS that consists of AIDS nodes (or network monitoring gateways or edge nodes) deployed at the gateway of each IoT sensor node. They record and analyze incoming network traffic packets to classify access attempts as elements of normal or anomaly classes. Although the AIDS is a useful

*Corresponding author

solution to detect attacks, it suffers from the problem of data dimensionality. The deployed AIDS nodes often lack the processing power to process huge data sets.

This problem can be mitigated by modifying the architecture of the AIDS, where all the data recorded by AIDS nodes are sent to the cloud node. Figure 1 depicts this architecture. The cloud node then executes optimization algorithms (proposed in the present work) to reduce and optimize the independent feature vector of the network traffic. This significantly reduces the effective size of the data set. This reduced optimal-feature vector (demonstrated in Fig. 2, where 1 represents selection and 0 represents rejection of a feature) is then communicated to each AIDS node. These nodes execute the ML algorithm (using only this optimal-feature vector) to test the incoming traffic and classify it as normal or anomaly. In the present work, our focus is to develop algorithms for the cloud node to effectively reduce and optimize the independent feature vector of the network traffic and improve the performance in terms of accuracy and other metrics.

The present research contributes in the following ways:

(a) We propose a novel algorithm called GWO-PSO (RGPO) which utilizes the concept of four leader wolves of gray wolf optimization (GWO) for updating the position of particles in particle swam optimization (PSO). Here, the weights assigned to leader wolves differ in accordance with their status within the hierarchy. Additionally, differential weights are assigned to the global-best and local-best positions in the proposed RGPO velocity equation.

(b) Algorithm H2TO is proposed, which employs the random forest in the first tier, and the RGPO algorithm in the second tier.

(c) A novel statistic called the reduced feature to accuracy ratio (RFAR) is proposed for comparing the proposed work with other similar results.

(d) The suggested approach is evaluated using three different data sets—NSLKDD, DS2OS, and BoTIoT, and the results show an improvement over existing works.

2. Literature review

The literature has provided numerous ML and deep learning (DL) solutions for feature selection and classification tasks for various applications. Kusy and Zajdel (2021) used a fusion of three different techniques to optimize the feature-length using a convolutional neural network (CNN) as the classification technique. The weighted wrapper concept was used in this paper, and a comparison was made with sequential wrapper methods. Siwek and Osowski (2016) applied feature selection on a vast air pollution data set to derive meaningful insights and predict the coming days' pollution index. Two methods were applied by the authors: the genetic algorithm and stepwise fit, to find the best predictive feature set.

These methods have also shown promise in the field of network and IoT security. Bhattacharjya (2022) studied the use of the blockchain technology in CPS and IoT architectures, maintaining the CIA triad in data communication. Gu and Lu (2021) used the NSLKDD data set to test their ML based approach and reported better results. Shafiq *et al*. (2022) employed a pretrained auto encoder to transfer the characteristics to similar IoT nodes for detection purposes. The data sets of the Mirai and Bashlite regarding infected devices were used for validation purposes. A finding was considerable reduction in complexity due to the use of pre-trained models in other models. In the work of Kim and Heo (2022), unrelated features were selected using correlation coefficients and the Boruta algorithm was applied to hydraulic IoT sensor data. Linear discriminant analysis (LDA), linear regression (LR), and a support vector classifier (SVC) were used for the classification task giving TPR performance up to 94%. In their study, Kumar *et al*. (2021b) combined several methodologies, such as the information gain and the correlation coefficient measure, to choose the most crucial feature set using the AND function. Through the use of the kNN, extreme gradient boosting (XGboost), etc., the classification was accomplished. Perceptron and hybrid deep neural networks were used on the DS2OS data set by Huma *et al*. (2021) and the accuracy achieved was 98%. Pahl and Aubet (2018) applied the idea of clustering for categorization. In particular, they combined the standard deviation with the BIRCH and k-means concepts. They evaluated their suggested methodology on IoT microservices and discovered an accuracy of up to 96.3%. Kumar *et al*. (2021a) used an ensemble of kNN, naive Bayes (NB), XGBoost and random forest (RF) classifiers in a fog environment. They tested their method on the DS2OS data set, achieving the accuracy up to 99.41%.

Numerous literary works have made substantial use of swarm intelligence algorithms. The binary version of the PSO method was created by Kennedy and Eberhart (1997) to enable feature selection. In their research, the authors used the coding mechanism to mask and unmask the features. In the work of Safaldin *et al*. (2021), a variety of wolf densities in GWO were taken into account, and the findings were promising. Another swarm intelligent algorithm, GWO, was proposed by Mirjalili *et al*. (2014). GWO is based on the concept of collective hunting by leader wolves, i.e., a collective search process. Singh and

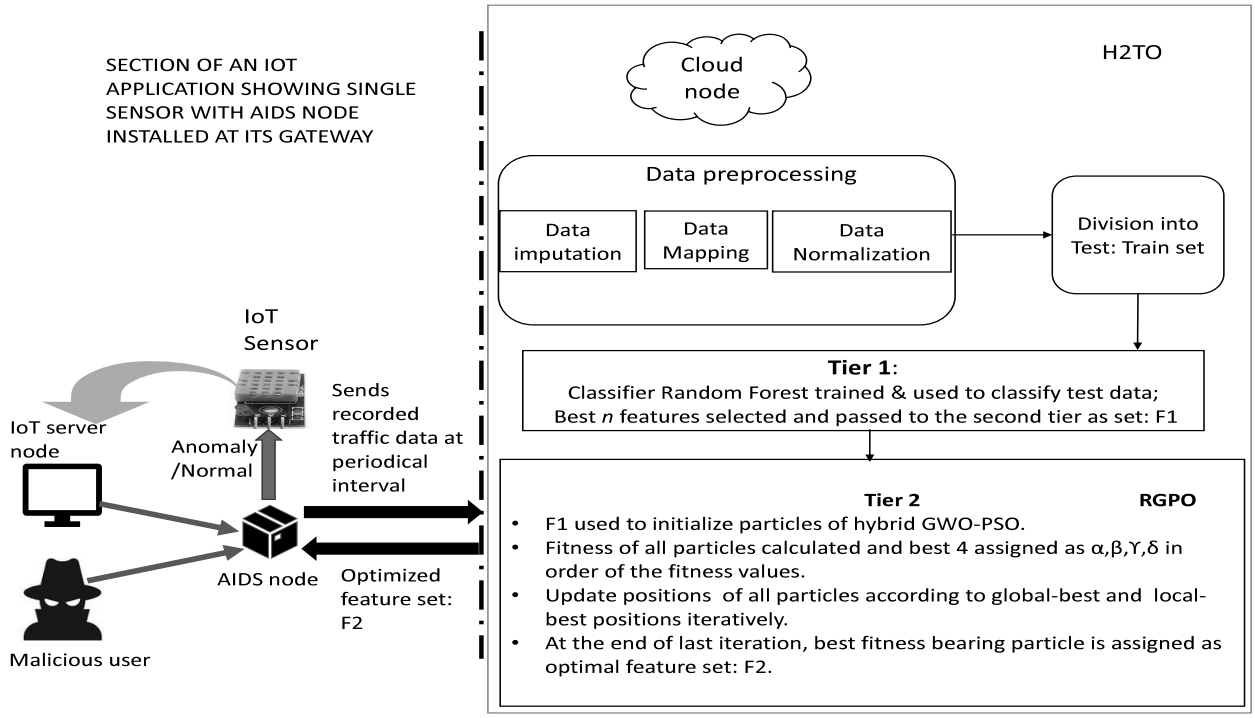

Fig. 1. Two-tier framework of the system.

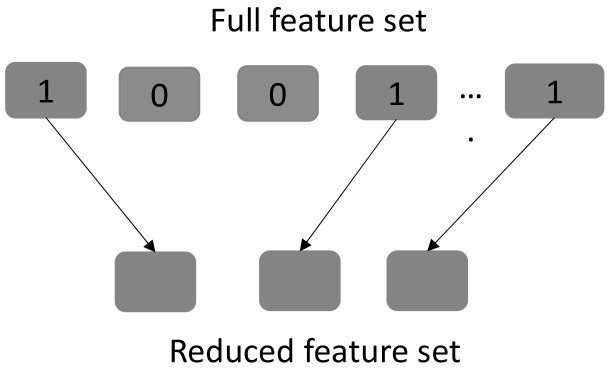

Fig. 2. Feature optimization process.

Singh (2017) developed HPSOGWO by combining GWO and PSO. Using an iterative process, Chopra *et al*. (2016) merged PSO and GWO. They utilized the output of one to initialize the population of the other.

PSO was applied to various neural networks to yield better results. Kowalski and Słoczyński (2021) modified PSO in terms of regularization control, geometric swarm centre determination, etc., and applied it to find an optimal configuration of a fuzzy flip-flop, producing the least training error. In the work of Carvalho and Ludermir (2007), the concept of PSO-PSO was developed, wherein inner PSO was used to optimize the weights of an MLP neural network while outer PSO was used to optimize its architecture. In the work of Band *et al*. (2020), PSO and a deep neural network were ensembled together to model gully erosion susceptibility using 13 independent features.

As regards work done on the DS2OS and BoTIot data sets, Hasan *et al*. (2019) used the random forest and decision tree techniques to classify the data of DS2OS. For classifying the BoTIoT data sets, Soe *et al*. (2020) used a correlation coefficient in conjunction with J48, an RF and a very fast decision tree (VFDT). Most of the works improved the classification efficiency; however, they suffer from the problem of data dimensionality.

3. Proposed work

3.1. Definitions and formal representation.

IoT sensor node: an electronic device which is deployed in the field to measure and record data.

IoT data: data recorded by the IoT sensor node.

IoT server: an intelligent device which processes IoT data and generates actionable events.

AIDS node (network monitoring gateway/edge node): an electronic device that records and analyzes incoming network traffic packets to classify them into normal or

Algorithm 1. H2TO.

Input: $\mathcal{D} = \{I_1, I_2, \ldots, I_p, Class\}$
1. Load the data set $\mathcal{D}$
2. $\mathcal{D} \leftarrow$encoding($\mathcal{D}$)
3. $\{I_1, I_2, \ldots, I_p\} \leftarrow$ Scale/Normalize$\{I_1, I_2, \ldots, I_p\}$
4. Split $\mathcal{D}$ as $\{\mathcal{D}^{Train} \cup \mathcal{D}^{Test}\}$

Tier 1:
5. Select best n features using RFMDA metrics:
   5.1 Train the classifier $\mathcal{C}_1$(RF)
   5.2 Predict the class of test data.
   5.3 Calculate prediction accuracy.

Output of 1st tier:
$\vec{F}_{reduced} \subseteq \{I_1, I_2, \ldots, I_p\}$

Tier 2:
6. Call algorithm **RGPO** with parameters:
   6.1 $\vec{F}_{reduced}$,
   6.2 Fitness function fit_{fn} and
   6.3 ML classifier $\mathcal{C}_2$ =kNN

Output of 2nd tier:
$\vec{F}_{final} \leftarrow \Psi_\alpha$,
Accuracy, DR, FPR, FNR, TNR, PR, F1.

Table 1. Cardinality of the data sets.

NSLKDD	Probe-11656, DoS-45927, U2R-52, R2L-995, Normal-67343
DS2OS	DoS-5780, MC-889, MO-805, Probe-342, Normal-347935, Scan-1547, Spy-532, WS-122
BoTIoT	DDoS-1926624, DoS-1650260, Recon.-91082, Normal-477, Theft-79

anomaly classes.

Cloud node: a computationally powerful node which runs optimization techniques to optimize the input feature vector.

Input feature vector: $\mathcal{D} = \{I_1, I_2, \ldots, I_p, Class\}$, where $\mathcal{D}$ represents the entire data set, $\{I_1, I_2, \ldots, I_p\}$ is the set of independent features, I_j are n-dimensional column vectors of $(n \times 1)$ dimension for $j = 1, 2, \ldots, p$, $Class$ is the dependent label, being an n-dimensional column vector and $Class \in \{C_1, C_2, \ldots, C_k\}$, where C_i are distinct classes present under label $Class$ for $i = 1, 2, \ldots, k$.

3.2. Data set description. In the present work, three data sets: DS2OS (Pahl and Aubet, 2018), NSLKDD (Tavallaee *et al.*, 2009) and BoTIoT (Koroniotis *et al.*, 2019) are used. Table 1 describes these data sets.

NSLKDD: The NSLKDD data set is a collection of multiple different incursions that were simulated in a military network environment. It is a modified version of the original KDD Cup 1999 data set. The authors can thoroughly compare their work with that of others using this well-known legacy network data set. The NSLKDD data set's independent input feature vector is made up of 41 different features or traffic attributes. Attacks can be divided into four primary categories: DoS, root to local (R2L), user to root (U2R), and probe.

DS2OS: In DS2OS, traces originated from the application layer and were captured in an IoT context, distinguishing them significantly from standard network traces. The independent input feature vector of the DS2OS data set consists of 12 distinct features/traffic attributes. DoS, data type-probing (probe), malicious control (MC), malicious operations (MO), scan, spying and wrong set-up (WS) are attack types in DS2OS.

BoTIoT: BoTIoT was recorded in a real-world network setting with both botnet and common traffic. Forty three unique features or traffic attributes make up the independent input feature vector of BoTIoT. The four types of attacks are denial-of-service (DoS), distributed-denial-of-service (DDoS), reconnaissance, and theft.

Therefore, a thorough validation of the proposed work using three different natured, different timeline-based, different layered and different sized data sets enable comprehensive analysis.

3.3. Pre-processing of the data set. This first step is subdivided into the following three substeps:

Missing fields in datasets: There are no missing fields in the BoTIoT and NSLKDD data sets, whereas in the DS2OS data set, missing fields are imputed with "missing".

Mapping from non-numerical form to numerical form: In the NSLKDD data set, non-numerical features are "service", "flag" and "protocol type", while in the DS2OS these are "source ID", "source Type" and others. These attributes must be transformed into numerical form. In this work, mapping is accomplished using ordinal encoding, so UDP is assigned the value of 1 and TCP as 2 for the feature "protocol type". The ordinal encoding technique does not produce a sparse data set, unlike one hot encoding.

Data normalization: It is important to scale the values of a feature using normalization or scaling techniques to provide consistency to the otherwise large range of values. The feature values in the current study are scaled using the

Algorithm 2. RGPO.

Input: Max_{iter}, N_p, $\vec{F}_{reduced}$, Fit_{fn}, $\mathcal{C}_2$

1. Initialize each particle Ψ_k, $k = 1$ to N_p as
   1.1 $\vec{F}_{left} = \{I_1, I_2, \ldots, I_p\} - \{\vec{F}_{reduced}\}$
   1.2 $\Psi_k pos = \{x = 1 \ \forall x \in \vec{F}_{reduced}\} \cup \{y = 0 \text{ or } 1 \ \forall y \in \vec{F}_{left}\}$
   1.3 $\Psi_k vel$ = random()
   1.4 $LoBest_k pos = \Psi_k pos$

2. **for** *all* Ψ_k **do**
   2.1 Fit_k =Compute fitness of $\Psi_k pos$
   2.2 $LoBest_k fit = Fit_k$

3. Sort(Fit_k) in order and assign:
   3.1 $\Psi_\alpha = \Psi_k pos$ having best(Fit_k)
   3.2 $\Psi_\beta = \Psi_k pos$ having 2nd best(Fit_k)
   3.3 $\Psi_\delta = \Psi_k pos$ having 3rd best(Fit_k)
   3.4 $\Psi_\omega = \Psi_k pos$ having 4th best(Fit_k)

4. $GlBestpos$ = Use Eqn. (13)
5. $GlBestfit$ = Compute fitness of $GlBestpos$
6. Initialize ξ, η_1,η_2

7. **while** *iteration* $< Max_{iter}$ **do**
   7.1 **for** *all* Ψ_k, $k = 1$ *to* N_p **do**
      7.1.1 Initialize x_1, x_2
      7.1.2 $\Psi_k vel$ = Eqn. (14)
      7.1.3 $\Psi_k pos$ = Eqn. (16)
      7.1.4. Calculate Fit_k=Compute fitness of $\Psi_k pos$
      7.1.5 **if** $Fit_k < LoBest_k fit$ **then**
         7.1.5.1 $LoBest_k fit = Fit_k$
         7.1.5.2 $LoBest_k pos = \Psi_k pos$
   7.2 Sort(Fit_k) in order and update:
      7.2.1 $\Psi_\alpha = \Psi_k pos$ having best(Fit_k)
      7.2.2 $\Psi_\beta = \Psi_k pos$ having 2nd best(Fit_k)
      7.2.3 $\Psi_\delta = \Psi_k pos$ having 3rd best(Fit_k)
      7.2.4 $\Psi_\omega = \Psi_k pos$ having4th best(Fit_k)

   7.3. $GlBestNpos$ = Eqn. (13)
   7.4. $GlBestNfit$ = Compute fitness of $GlBestNpos$
   7.5. **if** $GlBestNfit < GlBestfit$ **then**
      7.5.1 $GlBestfit = GlBestNfit$
      7.5.2 $GlBestpos = GlBestNpos$

Output: $\vec{F}_{final} \leftarrow \Psi_\alpha$,
Accuracy, DR, FPR, FNR, TNR, PR, F1.

min-max scaling method

$$I_{scale} = \frac{I - \min(I)}{\max(I) - \min(I)}.$$

Here, I is the original value, I_{scale} is the scaled value, $\min(I)$ is the minimum value and $\max(I)$ is the maximum value.

3.4. H2TO. Algorithm 1 shows the pseudocode of the H2TO algorithm. $\mathcal{D}^{Train}$ represents the training set of the data set, $\mathcal{D}^{Test}$ represents the testing set of the data set, $\vec{F}_{reduced}$ represents the intermediate reduced feature vector obtained from the first tier, $\vec{F}_{left}$ represents features other than those present in $\vec{F}_{reduced}$, $\vec{F}_{final}$ represents the final reduced optimal feature vector, $\mathcal{C}_1$ represents Classifier 1, $\mathcal{C}_2$ represents Classifier 2, Fit_{fn} represents the fitness function and Ψ_α represents particle α. The algorithm returns accuracy, detection rate (DR), false positive rate (FPR), false negative rate (FNR), true negative rate (TNR), precision (PR) and F1-score (F1). The pre processing processes used on the data set are displayed in Steps 1–3. To divide the modified data set into training and test sets in Step 4, random selection is applied. According to other relevant works in the literature, the ratio of 80:20 (train:test) is taken into account for the current investigation for all three data sets.

The transformed data are then passed through two tiers: Tier 1 and Tier 2. Step 5 implements Tier 1 which outputs feature set $\vec{F}_{reduced}$ using a random forest classifier and RFMDA metrics. RFMDA (random forest mean decrease accuracy) ranks all the features using the concept of permutation (Hur *et al*., 2017). For ranking each feature, accuracy is measured before and after permuting the feature (under test), and based on the difference in values the importance of that feature is ascertained. Kumar *et al*. (2021b) showed how RFMDA is useful to get a meaningful feature set. Regarding the random forest, classification of data instances is made using a collection of decision trees and a collective voting pattern. As a result, it typically provides good accuracy, is able to prevent overfitting and is free from the bias of a single decision tree, which is advantageous for an intrusion detection system.

Step 6 implements Tier 2, which obtains this $\vec{F}_{reduced}$ feature set and utilizes it to initialize the RGPO particle population. Specifically, PSO particles are initialized by taking union of $\vec{F}_{reduced}$ and random selection over $\vec{F}_{left}$. Thus, instead of using the original method of complete randomization, we have used partial intelligent initialization. Notably, intelligent initialization of particles is necessary to improve algorithmic convergence and performance (Tian, 2018), as here, the search mechanism of particles starts from intelligently initialized positions.

With unique train:test sets created by using the random selection strategy before each run, the H2TO algorithm is run ten times to avoid a chance bias. Out of these 10 runs, the shortest feature vector with the best accuracy is chosen. Table 6 shows these optimal-feature

vectors. To validate the effectiveness of the obtained feature vectors, statistical analysis is performed, whose results are given in Section 4.4.

3.5. RGPO Algorithm.

3.5.1. Prequisite. The global-best and local-best positions in PSO are, respectively, the historical best positions examined by the swarm and the historical best positions explored by the k-th particle at any instance (Kennedy and Eberhart, 1997). The position update of the k-th particle depends on its local-best and global-best position. It is important to note that this global-best is based on the historical best position occupied by a single particle of the swarm, considering all the particles. The following equations depict this update process in PSO:

$$\Psi_k pos(iter+1) = \Psi_k pos(iter) + \Psi_k vel(iter+1), \quad (1)$$

$$\begin{aligned}&\Psi_k vel(iter+1)\\ &= \xi \times \Psi_k vel(iter)\\ &\quad + \eta_1 \times x_1 \times (LoBest_k pos(iter) - \Psi_k pos(iter))\\ &\quad + \eta_2 \times x_2 \times (GlBestpos(iter) - \Psi_k pos(iter)),\end{aligned} \quad (2)$$

where $\Psi_k pos$ is the k-th particle position vector, $\Psi_k vel$ is the k-th particle velocity vector, $LoBest_k pos$ is the local-best position vector of k-th particle, $GlBestpos$ is the global-best position vector over all the particles, ξ is the inertial-weight parameter, η_1 and η_2 are optimization parameters, x_1 and x_2 are numbers $\in$ [0,1] and $\times$ denotes the multiplication.

Unlike in PSO, in GWO the updating of every k-th wolf (or particle in PSO) is dependent on three best valued leader wolves instead of a single wolf (Mirjalili *et al*., 2014). The update here is given as

$$\Psi_k pos(iter+1) = \frac{\tau_{k1} + \tau_{k2} + \tau_{k3}}{3}, \quad (3)$$

$$\tau_{k1} = \Psi_\alpha - A_1 \times Dist_\alpha, \quad (4)$$

$$\tau_{k2} = \Psi_\beta - A_2 \times Dist_\beta, \quad (5)$$

$$\tau_{k3} = \Psi_\delta - A_3 \times Dist_\delta, \quad (6)$$

$$A_i = 2 \times a \times random1_i - a, \quad i = 1,2,3, \quad (7)$$

where $Dist_\alpha$, $Dist_\beta$, $Dist_\delta$ are the distances of the k-th wolf from the α, β, δ wolves,

$$Dist_\alpha = |C_1 \times \Psi_\alpha - \Psi_k pos|, \quad (8)$$

$$Dist_\beta = |C_2 \times \Psi_\beta - \Psi_k pos|, \quad (9)$$

$$Dist_\delta = |C_3 \times \Psi_\delta - \Psi_k pos|, \quad (10)$$

$$C_i = 2 \times random2_i, \ \ i = 1,2,3. \quad (11)$$

Here, Ψ_α, Ψ_β and Ψ_δ represent positions of wolves α, β and δ, respectively, a linearly decreases from 2 to 0 over the course of iterations, $random1_i$ and $random2_i$ are random numbers uniformly distributed over $[0,1]$ for $i = 1,2,3$.

3.5.2. Proposed alterations. In the current study, the modifications in PSO create a wider search space as well as improve the exploitation process as detailed below.

(i) We use four leader particles α, β, δ and ω to compute the global-best position at each iteration, in contrast to the classic PSO, where a single particle is used to determine the global-best position. This improves the exploration process of the algorithm, where dependence on one particle is substituted by the collection of four leader particles, thus yielding a wider search space to explore the optimal solution.

(ii) The relative ranking concept is applied for assigning differential weights to the leaders, α, β, δ and ω, in order of their ranks, with the maximum weight assigned to α followed by β and so on. This improves the focus of the exploration process. If all the leaders are given equal weights, the exploration process gets affected by less-accurate leaders in the same proportion as high-accurate leaders.

(iii) Global-best and local-best positions are assigned differential weights while updating the velocity of the k-th particle. This is done to assign higher importance to the global-best position, which is better than or equal to all the local-best positions at any given instance. This helps in further improving the focus of the exploration process to reach the optimal solution.

The following equations model this:

$$\begin{aligned}&GlBestpos(iter+1)\\ &\qquad = \frac{4 \times \Psi_\alpha + 3 \times \Psi_\beta + 2 \times \Psi_\delta + \Psi_\omega}{10}.\end{aligned} \quad (12)$$

The following equation converts $GlBestpos(iter+1)$ to binary form:

$$\begin{aligned}&GlBestpos(iter+1)\\ &= \begin{cases} 1 & \text{if } sigmoid\left(\frac{4\times\Psi_\alpha+3\times\Psi_\beta+2\times\Psi_\delta+\Psi_\omega}{10}\right) \geq rand,\\ 0 & \text{otherwise.}\end{cases}\end{aligned} \quad (13)$$

Similarly, the velocity is updated as

$$\begin{aligned}&\Psi_k vel(iter+1)\\ &= \xi \times \Psi_k vel(iter)\\ &\quad + \eta_1 \times x_1 \times (u \times LoBest_k pos(iter) - \Psi_k pos(iter))\\ &\quad + \eta_2 \times x_2 \times (v \times GlBestpos(iter) - \Psi_k pos(iter)),\end{aligned} \quad (14)$$

where

$$u + v = 1. \quad (15)$$

Finally, $\Psi_k pos(iter+1)$ is converted to binary form as

$$\Psi_k pos(iter+1) = \begin{cases} 1 & \text{if } sigmoid(\Psi_k pos(iter) \\ & +\Psi_k vel(iter+1)) \geq rand, \\ 0 & \text{otherwise.} \end{cases} \quad (16)$$

Algorithm 2 presents the pseudocode of RGPO. In the algorithm, $GlBestfit$ denotes the global-best fitness value, $LoBest_k fit$ denotes the local-best fitness value of k-th particle, $GlBestNpos$ denotes the new global-best position and $GlBestNfit$ denotes the new global-best fitness value at each iteration. The inputs to the algorithm are the maximum number of iterations Max_{iter}, the number of particles of PSO N_p, reduced feature set $\vec{F}_{reduced}$, classifier $\mathcal{C}_2$ (here a kNN) and fitness function Fit_{fn}. The kNN has the advantage of dynamically classifying new instances effectively as it does not generate any discriminative function (Kumar *et al.*, 2021a). This is beneficial for a system like the AIDS, where new instances keep generating with an increase in traffic. RGPO returns a reduced optimal feature set, accuracy and other performance metrics as output. Step 1 of the algorithm initializes the position vector of each particle. Specifically, these are initialized by taking the union of $\vec{F}_{reduced}$ and random selection over $\vec{F}_{left}$. Further, the initialized position of each particle is also stored as its local-best position. Step 2 computes the fitness value of each particle and stores it as its local-best fitness value.

Step 3 sorts the computed fitness values. It further assigns the best particle position (particle having the best fitness value) to particle α, second the best to β, third best to δ and fourth best to ω. At this point Step 4 uses Eqn. (13) to compute the global-best position. Step 5 computes the fitness value of the global-best position. Step 6 initializes the parameters of the algorithm. Iterations start from Step 7, where in each iteration the velocity and position of k-th particle are updated using equations as specified in the algorithm. The fitness of all particles is recomputed; based on this the local-best position of each particle is updated. The particles are sorted in order of their fitness values and, based on this, reassignment of α, β, δ and ω is done. The global-best position is recomputed based on the reassigned leader particles. At the end of the last iteration, the position vector of α is returned as the optimal feature vector.

Table 2 lists the parameters used in the proposed work. The parameter values shown in the table are an outcome of experimental analysis.

3.6. Fitness function used. The fitness value of a given solution is measured using the fitness function

Table 2. Parameter table.

Parameters used	Specification
First layer ML algorithm	RF
Number of trees in RF	500
Second layer search algorithm	RGPO
ML technique used in wrapper approach	kNN
Fitness function	Eqn. (17)
Number of particles used in PSO	5,7,10,12
Value m in fitness function	0.9995
Value n in fitness function	0.0005
Number of leader particles	4
Value of inertial weights ξ	0.7
Values of η_1, η_2	0.5, 0.5
Values of u, v	0.4, 0.6

Table 3. Overall metrics for all classes.

Data set	Feats.	Acc.	DR	FPR	PR	F1
NSLKDD	9	99.44	99.43	0.34	99.43	99.43
DS2OS	4	99.44	99.45	19.39	99.44	99.44
BoTIoT	8	99.98	99.98	0.18	99.98	99.98

Table 4. Descriptive statistical results.

	Optimal feature set		All feature set	
Data set	mean	sd	mean	sd
NSLKDD	99.44	0.02	99.46	0.03
DS2OS	99.44	0.03	99.44	0.02
BoTIoT	99.98	0.01	99.98	0.01

Table 5. Two sample t-test.

Data set	t-value	p-value	C.I-95%
NSLKDD	-2.01	0.05	$[-0.03, 0.0]$
DS2OS	-0.64	0.53	$[-0.02, 0.01]$
BoTIoT	-1.53	0.14	$[-0.01, 0.0]$

Table 6. Optimal-feature sets returned by the algorithm.

Data set	Feats.	Feature specification
NSLKDD	9	protocol_type,land, is_host_login, count, rerror_rate, same_srv_rate, dst_host_count, dst_host_srv_count, dst_host_rerror_rate
DS2OS	4	sourceID, destinationServiceType, operation, value
BoTIoT	8	proto, mean, stddev, srate, drate,AR_P_Proto_P_SrcIP, AR_P_Proto_P_DstIP, Pkts_P_State_P_Protocol_P_SrcIP

employed in optimization problems, and the solutions are updated as a result. The fitness function to be minimized is (Alzubi *et al*., 2020)

$$Fit_{fn} = m \times (1 - Acc) + n \times (len), \qquad (17)$$

where Acc denotes the accuracy, len denotes the length of the feature vector, m and n are variables used to assign different weights to the terms of accuracy and feature length in the above equation, subject to the constraint that $m + n = 1$.

3.7. Performance metrics used. For experimental evaluation and comparison, the following metrics are used: accuracy, DR, FPR, FNR, TNR, PR and F1. These common measures are employed to verify algorithms. These measurements, however, fall short of fully capturing an algorithm's performance. For example, if a proposed algorithm is able to reduce the length of the feature vector, but in the process, accuracy is compromised by a small proportion, there is no metric to establish supremacy or inferiority of such an algorithm in comparison with other existing algorithms. Motivated by this problem, we have defined a novel metric:

$$\text{RFAR} = \frac{OF_{len}}{AF_{len}} \times \frac{1}{OF_{acc}} \times 100, \qquad (18)$$

where OF_{len}, AF_{len} and OF_{acc} are the lengths of the optimal-feature vector and the all-feature vector as well as accuracy obtained with the optimal feature vector, respectively. A lower value of RFAR indicates better performance.

4. Experimental results and a discussion

4.1. Overall performance of the algorithm. The overall values of accuracy, length of the feature vector, FPR, DR, F1-score, and PR for the three data sets are displayed in Table 3. H2TO performs well overall, returning the accuracy, DR, precision, and F1-score values above 99%. H2TO is therefore effective in detecting anomalies in network traffic.

4.2. Iterationwise performance metrics. The convergence of H2TO can be gauged by plotting the iterationwise results of various metrics (Fig. 3). For NSLKDD and BoTIoT, the plot till the 30-th iteration is shown, while for DS2OS, the plot till the 14-th iteration is shown as the convergence towards the optimal solution is faster in the case of DS2OS. For all the data sets, the feature-length and FPR broadly tend to reduce as the iteration progress. Similarly, for DR, accuracy, precision and F1-score, the graph shows an overall upward trend as iterations progress.

4.3. Performance of algorithm with varying values of N and k. Experiments were conducted for $k = 1, 3, 5, 7$ and $N = 5, 7, 10, 12$. Figure 4 shows the variation in results with values of $k = 1, 3, 5, 7$ (keeping N as 10), $N = 5, 7, 10$ and 12 (keeping k as 1), as they yield better results compared with other parameter combinations. As seen in the figure, NSLKDD and BoTIoT produce the best performance for $k = 1$ and $N = 10$ in terms of the accuracy, feature-length and other metrics. However, for DS2OS, the best accuracy of 99.44% is obtained for $k = 5$ and $N = 10$, with the feature-length of 5. For the same data set, $k = 1$ and $N = 10$ yield an accuracy of 99.44%; however, in this case, the feature-length is reduced to the value of 4.

4.4. Statistical analysis. Table 6 shows the lists of optimal feature sets returned by the algorithm. Correlation graphs of the above optimal-feature sets were plotted for NSLKDD, DS2OS and BoTIoT as shown in Figs. 5, 6 and 7, respectively. The figures reflect the comparison between the correlation graph of all-feature set with that of the optimal feature set. By observing the color palette, it is noticed that the white color is reduced in the optimal feature set correlation graph as compared to the all-feature set correlation graph, meaning that the highly correlated features were filtered out in the optimal feature set in the case of all the data sets. These highly correlated features have little utility for training the ML model.

For statistical analysis of the accuracy obtained with the optimal feature set as compared with that obtained with the all-feature set, a sample size of 20 runs is considered.

Table 4 compares the statistical results in terms of accuracy and the standard deviation for the optimal feature set and the all-feature set. For all the data sets, the accuracy obtained with the optimal feature set is marginally lower than that obtained with the all-feature set; however, the proposed algorithm is able to reduce the length of the feature vector by a substantial margin.

Table 5 gives the two-sample t-test results for the hypothesis

$$H_0 : Accuracy(\text{all-features}) = Accuracy(\text{optimal-features})$$

with a confidence level of 95%. Since the p-value exceeds 0.05 for all the data sets, this implies statistical evidence to support the hypothesis.

Figure 8 shows the box plots of accuracy for the three data sets. In the case of NSLKDD, the minimum, maximum, median and mean values of accuracy obtained with the optimal-feature set are 99.39%, 99.47%, 99.44% and 99.44%, respectively. With the all-feature set these are 99.41%, 99.49%, 99.45% and 99.46%, respectively. In

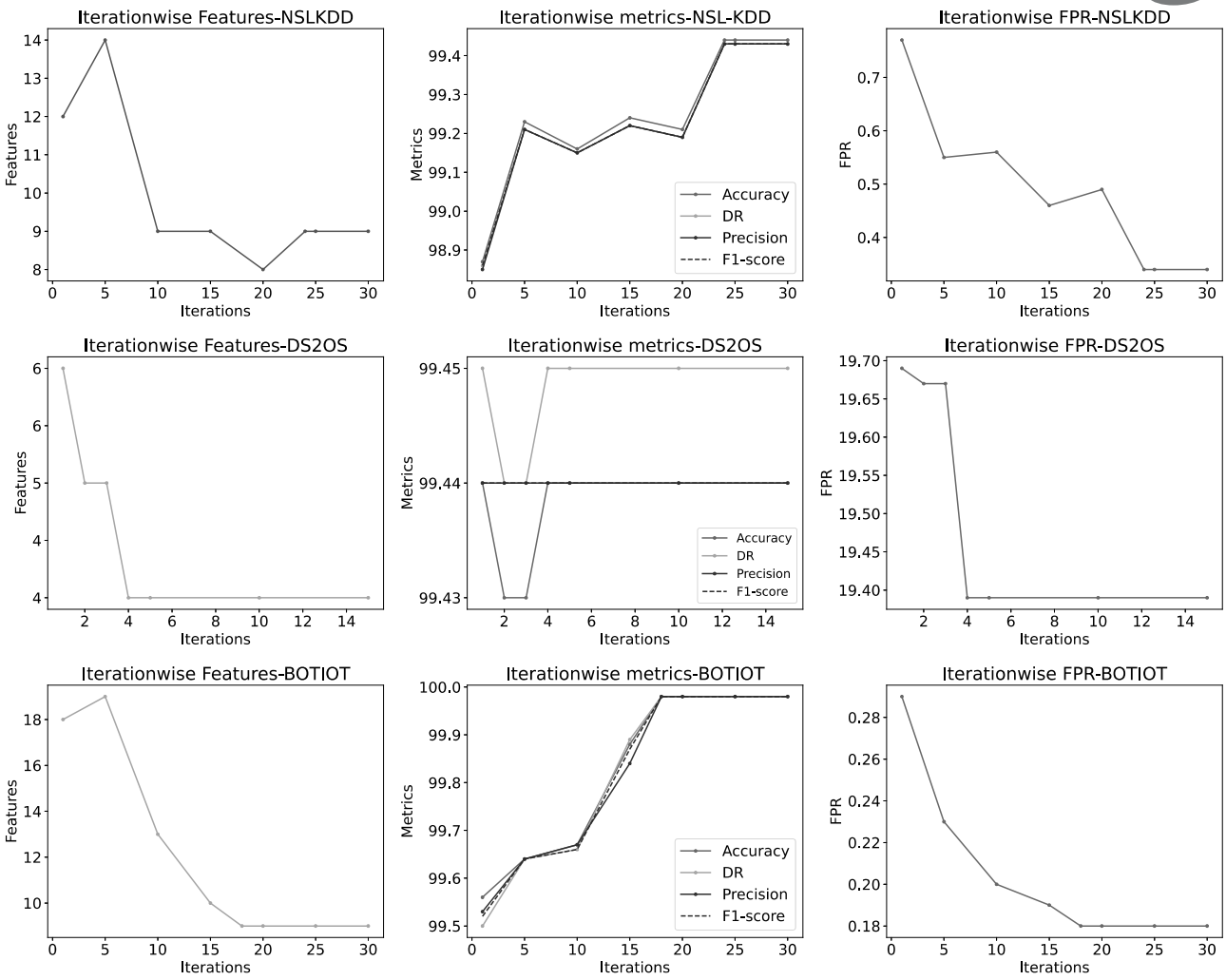

Fig. 3. Iterationwise variation in results.

the case of DS2OS, with the optimal-feature set, they are 99.37%, 99.48%, 99.44% and 99.44%, respectively. Then with the all-feature set they are 99.39%, 99.48%, 99.44% and 99.44%, respectively. In the case of BoTIoT, with optimal feature set these are 99.95%, 99.99%, 99.98% and 99.98%, respectively, with the all-feature set 99.97%, 99.99%, 99.98% and 99.98%, respectively. The figures show the consistency in the results of various runs except for a few outliers.

4.5. Comparison of the proposed work with related research. Tables 7, 8 and 9 depict the classwise comparison of the proposed H2TO with other algorithms in terms of the detection rate. Regarding NSLKDD, the proposed algorithm performs best in terms of detecting the DoS and normal classes, producing values of 99.75% and 99.52%, respectively, while the worst DR is reported for the U2R attack type with the value of 44.44%. Regarding DS2OS, the proposed algorithm outperforms in terms of detecting MC and DOS attack types, yielding the values of 100% and 68.69%, respectively. For the Probe, Scan, MO and WS classes, the DR values are 100%. However, it underperforms for the Spying class, giving the value of 97.48%, which is lower than for a few other algorithms. For BoTIoT, the proposed algorithm outperforms in terms of detecting the theft attack type, returning the highest value of 95.65%. For the classes DoS, Reconnaissance and Normal, the returned DR values are 100%. The only class where the algorithm underperforms is the DDoS class, returning the value of 99.97%. Thus, the proposed work is able to outperform the existing techniques for most of the classes.

Similarly, Tables 10, 11 and 12 compare the proposed algorithm with the existing works in terms of accuracy, the length of the feature vector and RFAR. Regarding NSLKDD, Wei *et al*. (2020) achieve the best accuracy value of 99.47%. However, in this case, the length of the feature vector is as high as 24 features. In terms of the minimal feature length, Kunhare *et al*. (2020)

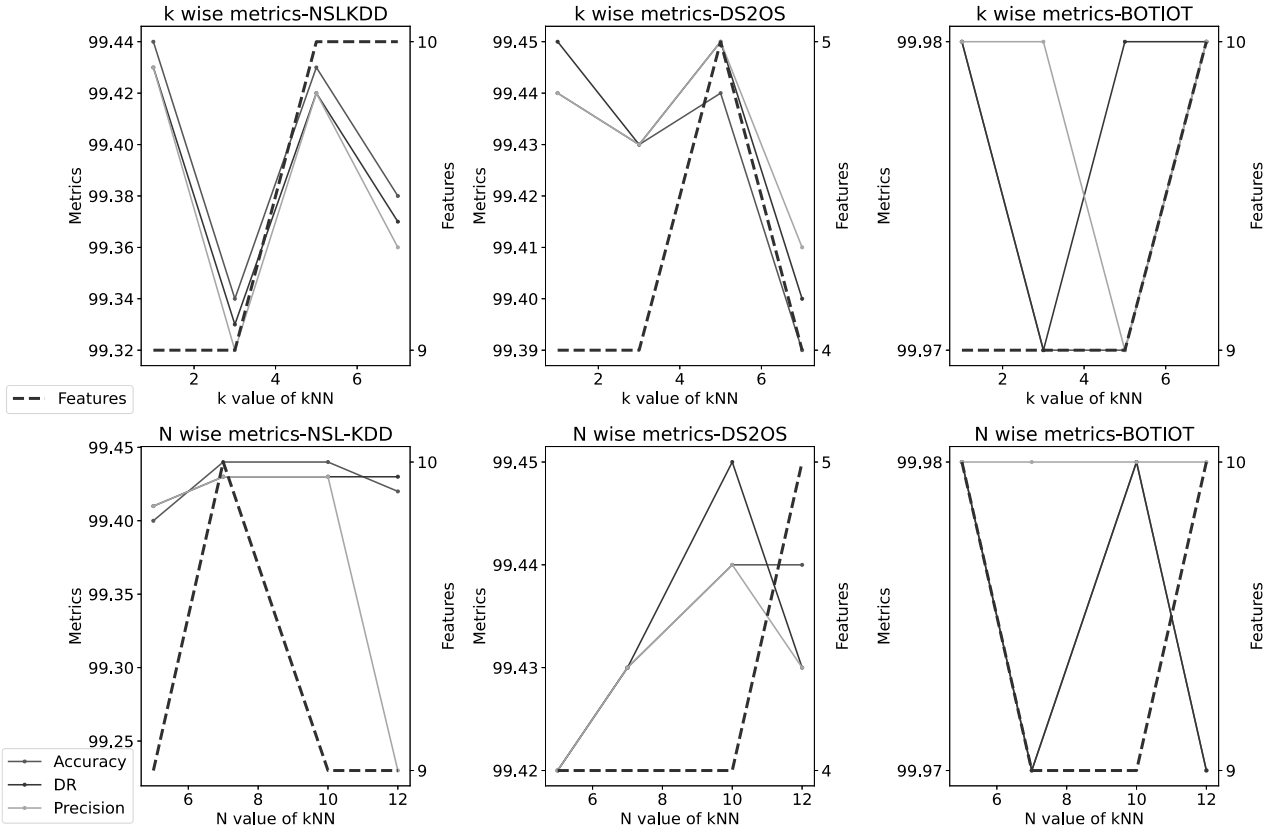

Fig. 4. Variation in performance of H2TO with k and N.

achieve a value of 10; however, in this case, the accuracy drops to 99.32%. H2TO, on the other hand, achieves the accuracy of 99.44% with the number of features being nine only. Thus, it can be concluded that, although the accuracy using the proposed algorithm drops marginally as compared with the results of Wei *et al.* (2020), the feature-length improves substantially. Regarding DS2OS, the proposed algorithm is able to outperform in terms of both accuracy and the feature length. In this case, the accuracy improves by 0.01% over the second best value of 99.43%, while the feature set is optimized to four features. Regarding BoTIoT, the best accuracy is achieved by Kumar *et al.* (2016) and Ashraf *et al.* (2022) as 99.99%, with the length of the feature vector being 10. Considering the minimal feature length, Soe *et al.* (2020) perform best using only eight features, but the accuracy is only 99.1%. Using the proposed algorithm, we achieved accuracy up to 99.98% (marginally lower compared with 99.99%) with the length of the feature vector equal to 8 (Soe *et al.*, 2020). Thus, we are able to reduce the length of the feature vector, besides retaining almost the best accuracy (unlike Soe *et al.* (2020), who compromised the accuracy for optimizing the length of the feature vector).

5. Conclusions and future work

In the present study, we worked towards optimizing the length of the feature vector without compromising other performance metrics to reduce the curse of data dimensionality related to huge network traffic. The decreased optimal-feature vector can then be used by low-performance devices (AIDS nodes) to identify attacks. A two-tier algorithm termed H2TO is suggested to accomplish this goal, and it makes use of the proposed RGPO algorithm, an amended version of PSO, to return an ideal feature vector along with other crucial metrics. We also presented a novel statistic, called RFAR, for comprehensive comparison with related publications. An experimental analysis pointed to better results than those produced earlier. With regard to the DoS and normal classes in NSL-KDD, MC and the DoS class in DS2OS, and theft class in BoTIoT, the detection rate specifically improves. In terms of overall performance, H2TO outperforms other works for NSLKDD in terms of the length of the feature vector, in addition to obtaining accuracy that is very near to the best. The feature vector's length was substantially optimized for DS2OS, and accuracy was increased. The accuracy and length

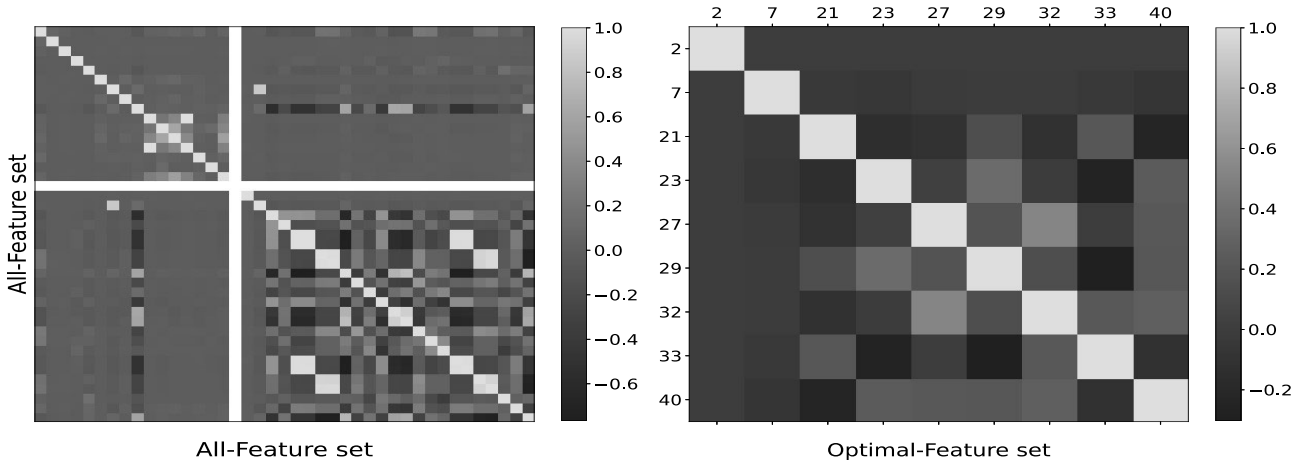

Fig. 5. Correlation graphs of the optimal-feature set vs. the all-feature set for NSLKDD.

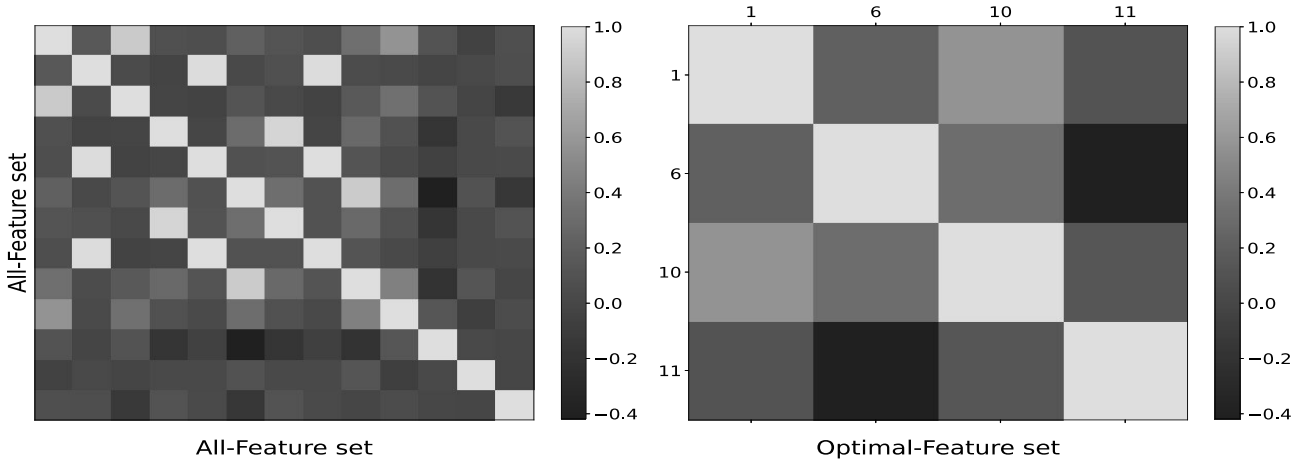

Fig. 6. Correlation graphs of the optimal-feature set vs. the all-feature set for DS2OS.

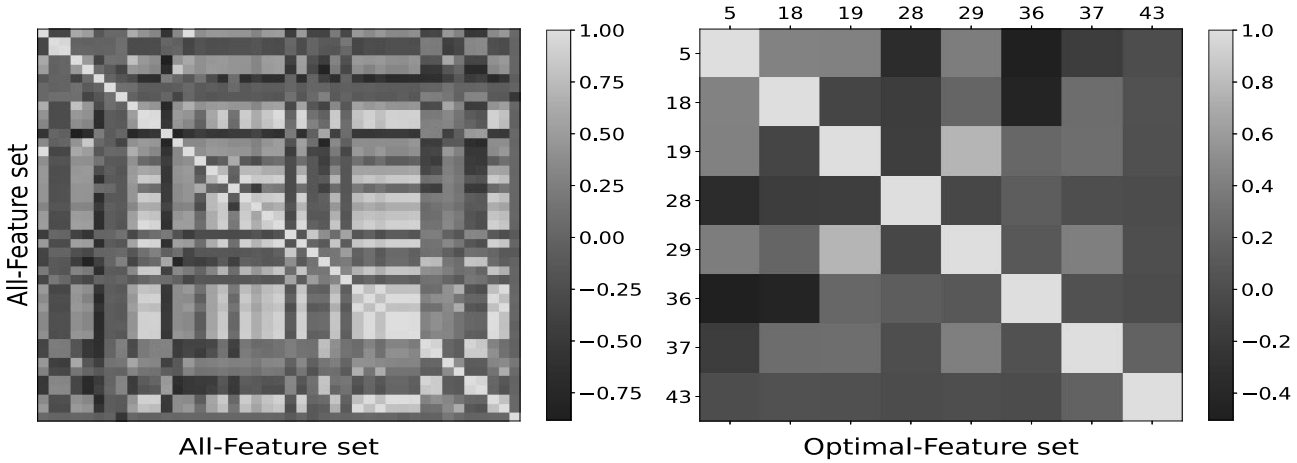

Fig. 7. Correlation graphs of the optimal-feature set vs. the all-feature set for BoTIoT.

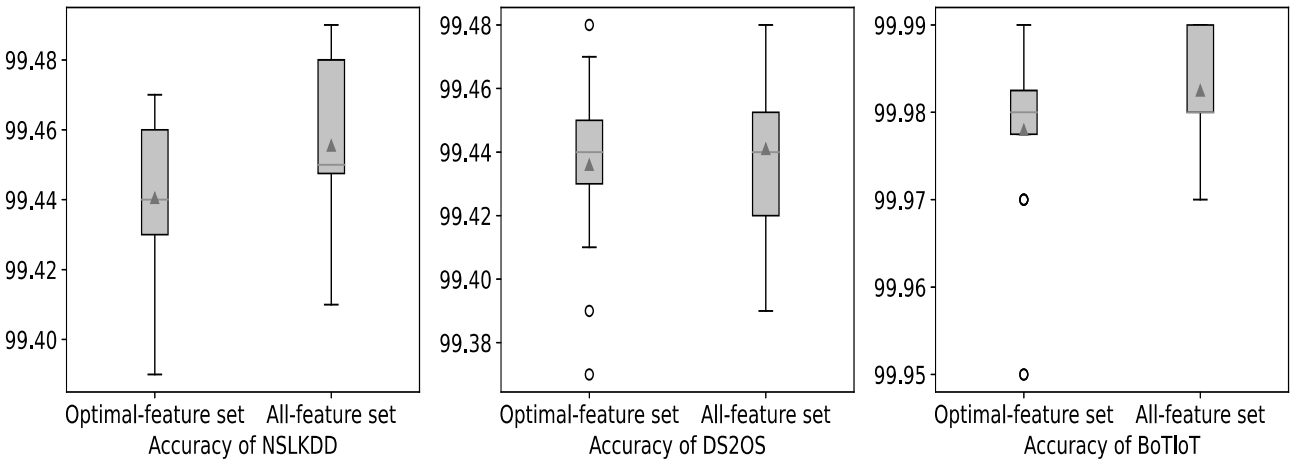

Fig. 8. Accuracy metric box plots for the optimal-feature set vs. the all-feature set.

of the feature vector were successfully optimized for BoTIoT.

As for the future, parameter tuning remains an open problem for researchers. This will help in further improving the performance of present algorithms. Also, for a few classes like U2R, accuracy and the detection rate can be improved with further research.

References

Alazzam, H., Sharieh, A. and Sabri, K.E. (2020). A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer, *Expert Systems with Applications* **148**: 113249.

Alzubi, Q.M., Anbar, M., Alqattan, Z.N., Al-Betar, M.A. and Abdullah, R. (2020). Intrusion detection system based on a modified binary grey wolf optimisation, *Neural Computing and Applications* **32**(10): 6125–6137.

Ashraf, E., Areed, N.F., Salem, H., Abdelhay, E.H. and Farouk, A. (2022). FIDChain: Federated intrusion detection system for blockchain-enabled IoT healthcare applications, *Healthcare* **10**(6): 1110.

Band, S.S., Janizadeh, S., ChandraPal, S., Saha, A., Chakrabortty, R., Shokri, M. and Mosavi, A. (2020). Novel ensemble approach of deep learning neural network (DLNN) model and particle swarm optimization (PSO) algorithm for prediction of gully erosion susceptibility, *Sensors* **20**(19): 5609.

Bhattacharjya, A. (2022). A holistic study on the use of blockchain technology in CPS and IoT architectures maintaining the CIA triad in data communication, *International Journal of Applied Mathematics and Computer Science* **32**(3): 403–413, DOI: 10.34768/amcs-2022-0029.

Carvalho, M. and Ludermir, T.B. (2007). Particle swarm optimization of neural network architectures and weights, *7th International Conference on Hybrid Intelligent Systems (HIS 2007), Kaiserslautern, Germany*, pp. 336–339.

Chen, L., Li, Y., Deng, X., Liu, Z., Lv, M. and Zhang, H. (2022). Dual auto-encoder GAN-based anomaly detection for industrial control system, *Applied Sciences* **12**(10): 4986.

Chopra, N., Kumar, G. and Mehta, S. (2016). Hybrid GWO-PSO algorithm for solving convex economic load dispatch problem, *International Journal of Research in Advent Technology* **4**(6): 37–41.

Gao, X., Shan, C., Hu, C., Niu, Z. and Liu, Z. (2019). An adaptive ensemble machine learning model for intrusion detection, *IEEE Access* **7**: 82512–82521.

Gu, J. and Lu, S. (2021). An effective intrusion detection approach using SVM with naïve Bayes feature embedding, *Computers & Security* **103**: 102158.

Gu, J., Wang, L., Wang, H. and Wang, S. (2019). A novel approach to intrusion detection using SVM ensemble with feature augmentation, *Computers & Security* **86**: 53–62.

Hasan, M., Islam, M.M., Zarif, M.I. and Hashem, M.M. (2019). Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches, *Internet of Things* **7**: 100059.

Huma, Z.E., Latif, S., Ahmad, J., Idrees, Z., Ibrar, A., Zou, Z., Alqahtani, F. and Baothman, F. (2021). A hybrid deep random neural network for cyberattack detection in the industrial Internet of Things, *IEEE Access* **9**: 55595–605.

Hur, J.H., Ihm, S.Y. and Park, Y.H. (2017). A variable impacts measurement in random forest for mobile cloud computing, *Wireless Communications and Mobile Computing* **2017**, Article ID: 6817627.

Kennedy, J. and Eberhart, R.C. (1997). A discrete binary version of the particle swarm algorithm, *1997 IEEE International conference on Systems, Man, and Cybernetics, Orlando, USA*, pp. 4104–4108.

Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J. and Alazab, A. (2019). A novel ensemble of hybrid intrusion detection system for detecting Internet of Things attacks, *Electronics* **8**(11): 1210.

Table 7. Classwise comparison of the proposed H2TO with other algorithms in terms of the detection rate (NSLKDD).

Works	DoS	Probe	Normal	R2L	U2R
Gao *et al.*, 2019	84.37	87.11	94.33	55.27	25.00
Pajouh *et al.*, 2017	84.68	79.76	94.56	34.81	67.16
Su *et al.*, 2020	87.55	85.76	97.50	44.25	20.95
Tian *et al.*, 2020	98.04	98.26	94.90	95.24	75.50
Mahfouz *et al.*, 2020	99.20	91.60	99.50	55.10	39.30
KNN (Kumar *et al*., 2021b)	99.69	98.25	98.00	95.23	**93.36**
RF (Kumar *et al*., 2021b)	99.71	**99.41**	98.01	**97.08**	93.22
Proposed work	**99.75**	98.41	**99.52**	91.44	44.44

Table 8. Classwise comparison of the proposed H2TO with other algorithms in terms of the detection rate (DS2OS).

Works	Probe	MC	Dos	Scan	MO	WS	Spying	Normal
KNN (Kumar *et al*., 2021b)	**100**	99	66	98	**100**	**100**	**100**	**100**
Hasan *et al.*, 2019	92	97	66	95	**100**	**100**	93	**100**
Xgboost (Kumar *et al*., 2021b)	**100**	99	66	**100**	**100**	**100**	**100**	**100**
RF (Kumar *et al*., 2021b)	**100**	97	66	97	**100**	**100**	**100**	**100**
Hasan *et al.*, 2019	92	92	66	71	56	**100**	4	**100**
Pahl and Aubet, 2018	**100**	32	68	54	66	**100**	13	95
Proposed work	**100**	**100**	**68.69**	**100**	**100**	**100**	97.48	99.99

Table 9. Classwise comparison of the proposed H2TO with other algorithms in terms of the detection rate (BoTIoT).

Works	DoS	DDoS	Reconn.	Normal	Theft
RF (Kumar *et al*., 2021b)	99	**100**	**100**	**100**	93
Soe *et al.*, 2020	**100**	**100**	**100**	0	0
Zeeshan *et al.*, 2021	99.76	99.87			
Shafiq *et al.*, 2020	**100**	98	81	75	93
XGboost (Kumar *et al*., 2021b)	**100**	**100**	**100**	**100**	93
Proposed work	**100**	99.97	**100**	**100**	**95.65**

Kim, D. and Heo, T.Y. (2022). Anomaly detection with feature extraction based on machine learning using hydraulic system IoT sensor data, *Sensors* **22**(7): 2479.

Koroniotis, N., Moustafa, N., Sitnikova, E. and Turnbull, B. (2019). Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: BOT-IOT dataset, *Future Generation Computer Systems* **100**: 779–796.

Kowalski, P.A. and Słoczyński, T. (2021). A modified particle swarm optimization procedure for triggering fuzzy flip-flop neural networks, *International Journal of Applied Mathematics and Computer Science* **31**(4): 577–586, DOI: 10.34768/amcs-2021-0039.

Kumar, P., Gupta, G.P. and Tripathi, R. (2021a). A distributed ensemble design based intrusion detection system using fog computing to protect the Internet of Things networks, *Journal of Ambient Intelligence and Humanized Computing* **12**(10): 9555–9572.

Kumar, P., Gupta, G. and Tripathi, R. (2021b). Toward design of an intelligent cyber attack detection system using hybrid feature reduced approach for IoT networks, *Arabian Journal for Science and Engineering* **46**(4): 3749–3778.

Kunhare, N., Tiwari, R. and Dhar, J. (2020). Particle swarm optimization and feature selection for intrusion detection system, *Sādhanā* **45**(1): 1–14.

Kusy, M. and Zajdel, R. (2021). A weighted wrapper approach to feature selection, *International Journal of Applied Mathematics and Computer Science* **31**(4): 685–696, DOI: 10.34768/amcs-2021-0047.

Mahfouz, A.M., Venugopal, D. and Shiva, S.G. (2020). Comparative analysis of ML classifiers for network intrusion detection, *2020 4th International Congress on Information and Communication Technology, London, UK*, pp. 193–207.

Mirjalili, S., Mirjalili, S.M. and Lewis, A. (2014). Grey wolf optimizer, *Advances in Engineering Software* **69**: 46–61.

Pahl, M. and Aubet, F. (2018). All eyes on you: Distributed multidimensional IoT microservice anomaly detection, *14th International Conference on Network and Service Management (CNSM), Rome, Italy*, pp. 72–80.

Pajouh, H.H., Dastghaibyfard, G. and Hashemi, S. (2017). Two-tier network anomaly detection model: A machine learning approach, *Journal of Intelligent Information Systems* **48**(1): 61–74.

Table 10. Accuracy, features, RFAR comparison (NSLKDD).

Works	Accuracy	Features	RFAR
Gu and Lu, 2021	99.36	41	1.01
Alazzam *et al.*, 2020	86.90	18	0.51
Gu *et al.*, 2019	99.41	41	1.01
Wei *et al.*, 2020	**99.47**	24	0.59
Safaldin *et al.*, 2021	96.00	12	0.31
Kumar *et al.*, 2021b	98.67	18	0.45
Kunhare *et al.*, 2020	99.32	10	0.25
Proposed work	99.44	**9**	**0.22**

Table 11. Accuracy, features, RFAR comparison (DS2OS).

Works	Accuracy	Features	RFAR
KNN (Kumar *et al.*, 2021b)	99.40	6	0.50
Chen *et al.*, 2022	99.10	12	1.01
Hasan *et al.*, 2019	99.35	11	0.92
RF (Kumar *et al.*, 2021b)	99.40	6	0.50
Hasan *et al.*, 2019	99.05	11	0.93
Pahl Aubet, 2018	96.30	12	1.04
Xgboost (Kumar *et al.*, 2021b)	99.43	6	0.50
Proposed work	**99.44**	**4**	**0.34**

Table 12. Accuracy, features, RFAR comparison (BoTIoT).

Works	Accuracy	Features	RFAR
Koroniotis *et al.*, 2019	99.70	10	0.23
Khraisat *et al.*, 2019	99.70	13	0.30
ANN (Ashraf *et al.*, 2022)	**99.99**	10	0.23
Xgboost (Ashraf *et al.*, 2022)	98.96	10	0.24
Kumar *et al.*, 2021b	**99.99**	10	0.23
Soe *et al.*, 2020	99.10	**8**	0.19
Proposed work	99.98	**8**	**0.19**

Safaldin, M., Otair, M. and Abualigah, L. (2021). Improved binary gray wolf optimizer and SVM for intrusion detection system in wireless sensor networks, *Journal of Ambient Intelligence and Humanized Computing* **12**(2): 1559–1576.

Shafiq, M., Tian, Z., Sun, Y., Du, X. and Guizani, M. (2020). Selection of effective machine learning algorithm and BoT-IoT attacks traffic identification for Internet of Things in smart city, *Future Generation Computer Systems* **107**: 433–442.

Shafiq, U., Shahzad, M.K., Anwar, M., Shaheen, Q., Shiraz, M. and Gani, A. (2022). Transfer learning auto-encoder neural networks for anomaly detection of DDoS generating IoT devices, *Security and Communication Networks* **2022**, Article ID: 8221351.

Singh, N. and Singh, S.B. (2017). Hybrid algorithm of particle swarm optimization and grey wolf optimizer for improving convergence performance, *Journal of Applied Mathematics* **2017**, Article ID: 2030489.

Siwek, K. and Osowski, S. (2016). Data mining methods for prediction of air pollution, *International Journal of Applied Mathematics and Computer Science* **26**(2): 467–478, DOI: 10.1515/amcs-2016-0033.

Soe, Y., Feng, Y., Santosa, P., Hartanto, R. and Sakurai, K. (2020). Towards a lightweight detection system for cyber attacks in the IoT environment using corresponding features, *Electronics* **9**(1): 144.

Su, T., Sun, H., Zhu, J., Wang, S. and Li, Y. (2020). BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset, *IEEE Access* **8**: 29575–29585.

Tavallaee, M., Bagheri, E., Lu, W. and Ghorbani, A.A. (2009). A detailed analysis of the KDD CUP 99 data set, *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, Canada*, pp. 1–6.

Tian, D. (2018). Particle swarm optimization with chaos-based initialization for numerical optimization, *Intelligent Automation & Soft Computing* **24**(2): 331–342.

Tian, Q., Han, D., Li, K.C., Liu, X., Duan, L. and Castiglione, A. (2020). An intrusion detection approach based on improved deep belief network, *Applied Intelligence* **50**(10): 3162–3178.

Wei, W., Chen, S., Lin, Q., Ji, J. and Chen, J. (2020). A multi-objective immune algorithm for intrusion feature selection, *Applied Soft Computing* **95**: 106522.

Zeeshan, M., Riaz, Q., Bilal, M.A., Shahzad, M.K., Jabeen, H., Haider, S.A. and Rahim, A. (2021). Protocol-based deep intrusion detection for DoS and DDoS attacks using UNSW-NB15 and BoT-IoT data-sets, *IEEE Access* **10**: 2269–83.

Akhileshwar Prasad Agrawal received his MTech degree from IIIT, Allahabad, India, his PhD from GGSIPU, Delhi, and his lecturer qualification at Delhi Skill and Entrepreneurship University, Government of NCT of Delhi. His current research interests include swarm intelligent algorithms, machine learning algorithms and feature selection techniques.

Nanhay Singh received his PhD degree from Kurukshetra University. He works as a professor at the Netaji Subhas University of Technology (East Campus), Delhi (formerly AIACTR). His area of interest includes the Internet of Things, machine learning algorithms and cloud computing.

Received: 26 July 2022
Revised: 25 October 2022
Re-revised: 25 November 2022
Accepted: 12 January 2023