sensors

MDPI

Article

Enhanced Anomaly Detection System for IoT Based on Improved Dynamic SBPSO

Asima Sarwar [1], Abdullah M. Alnajim [2,*], Safdar Nawaz Khan Marwat [1], Salman Ahmed [1], Saleh Alyahya [3] and Waseem Ullah Khan [1]

[1] Department of Computer Systems Engineering, University of Engineering and Technology, Peshawar 25120, Pakistan; asima.sarwar@uetpeshawar.edu.pk (A.S.); safdar@uetpeshawar.edu.pk (S.N.K.M.); sahmed@uetpeshawar.edu.pk (S.A.); waseem@uetpeshawar.edu.pk (W.U.K.)
[2] Department of Information Technology, College of Computer, Qassim University, Buraydah 51452, Saudi Arabia
[3] Department of Electrical Engineering, College of Engineering and Information Technology, Onaizah Colleges, Onaizah 56447, Saudi Arabia; saleh.alyahya@oc.edu.sa
* Correspondence: najim@qu.edu.sa

Abstract: The Internet of Things (IoT) supports human endeavors by creating smart environments. Although the IoT has enabled many human comforts and enhanced business opportunities, it has also opened the door to intruders or attackers who can exploit the technology, either through attacks or by eluding it. Hence, security and privacy are the key concerns for IoT networks. To date, numerous intrusion detection systems (IDS) have been designed for IoT networks, using various optimization techniques. However, with the increase in data dimensionality, the search space has expanded dramatically, thereby posing significant challenges to optimization methods, including particle swarm optimization (PSO). In light of these challenges, this paper proposes a method called improved dynamic sticky binary particle swarm optimization (IDSBPSO) for feature selection, introducing a dynamic search space reduction strategy and a number of dynamic parameters to enhance the searchability of sticky binary particle swarm optimization (SBPSO). Through this approach, an IDS was designed to detect malicious data traffic in IoT networks. The proposed model was evaluated using two IoT network datasets: IoTID20 and UNSW-NB15. It was observed that in most cases, IDSBPSO obtained either higher or similar accuracy even with less number of features. Moreover, IDSBPSO substantially reduced computational cost and prediction time, compared with conventional PSO-based feature selection methods.

Keywords: anomaly detection; Internet of Things; intrusion detection system; IoT security

Citation: Sarwar, A.; Alnajim, A.M.; Marwat, S.N.K.; Ahmed, S.; Alyahya, S.; Khan, W.U. Enhanced Anomaly Detection System for IoT Based on Improved Dynamic SBPSO. *Sensors* **2022**, *22*, 4926. https://doi.org/10.3390/s22134926

Academic Editors: Paolo Bellavista and Omprakash Kaiwartya

Received: 27 May 2022
Accepted: 26 June 2022
Published: 29 June 2022

1. Introduction

With the rise of the Internet, there has been an immense surge in Internet-based services [1]. As a result, many of the physical systems or devices that are connected to the Internet can easily be operated and managed remotely. Client behaviour can then be monitored and documented, future decisions can be predicted, and useful services provided [2]. The Internet of Things (IoT) is used in a variety of fields, including the smart home, smart city, smart healthcare, smart factories, smart supply chain, and smart retail. Figure 1 depicts a few of IoT applications that may be found in everyday life. The goal of such a smart environment is to make people's lives more productive and add value by addressing issues related to living conditions [3]. However, because of increased interconnectedness, the network has become more complicated, making network security more difficult to sustain. Intruders consider security lapses to be an invitation to discover and exploit vulnerabilities in IoT networks. However, network security breaches can result in significant financial losses for businesses and consumers. Hence, it is essential to design

a system that will ensure the security of the IoT network. Many tools and techniques are available to combat various cyber-attacks, such as spam filters, firewalls, anti-malware, intrusion detection systems (IDSs), intrusion prevention systems (IPSs), and so on [4].

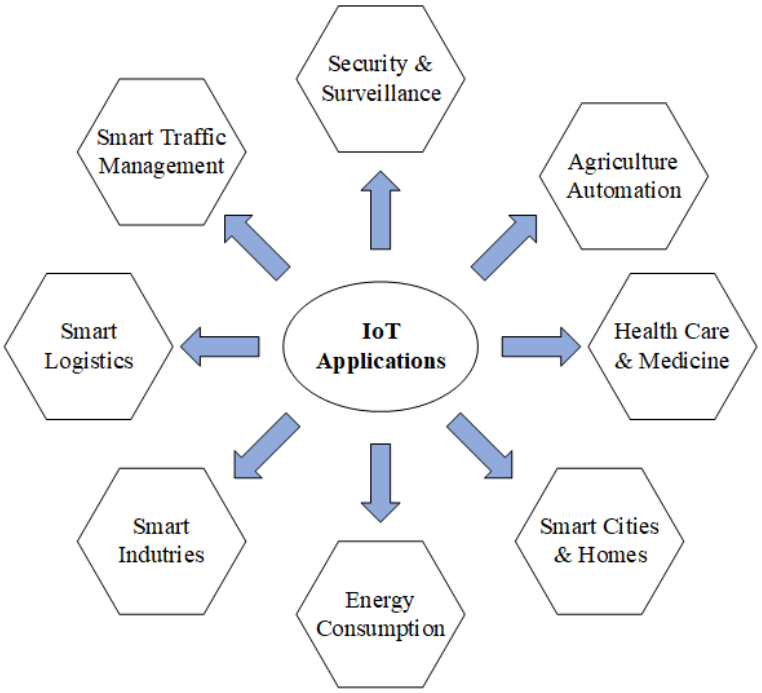


Figure 1. Potential IoT applications.

To ensure the security of an IoT network, an IDS can be an extremely effective and crucial solution. There are three key phases in the operation of an IDS. The first of these is monitoring, which is based on network or host sensors. The second phase is analysis, which involves feature extraction and pattern recognition. Finally, the third phase is detection, which detects any anomalies in a network.

Intrusion detection systems can be classified into two main groups: signature-based intrusion detection systems (SIDS) and anomaly-based intrusion detection systems (AIDS). Traditional SIDS methods involve examining network packets and attempting to match patterns to a signature database. A machine-learning (ML) approach is used in AIDS to train the model in normalised behaviour. Network activities are then compared with that normal behaviour. Anomaly-based intrusion detection systems are considered as a dynamic approach to anomaly detection, applying behaviour-oriented detection.

The AIDS strategy has in fact received more attention than any other approach [5]. The capacity to detect unknown or zero-day attacks is the main benefit of AIDS. The majority of researchers choose anomaly detection, since it appears to be the most viable means [6,7]. However, designing efficient IDS for IoT devices remains challenging, due to the following reasons:

(a) **Cyber-security datasets**
The majority of existing datasets are outdated and may be inefficient for grasping the behavioural patterns of modern cyber-attacks. Moreover, there is a dearth of knowledge about the characteristics of recent attacks and their patterns of occurrence.

(b) **Handling quality problems in Cyber-security datasets**
Cyber-security datasets may be incomplete, unbalanced, noisy, or contain inconsistent instances related to a particular security incident. The quality of the learning process, and performance of ML-based models is affected by such dataset issues [8].

(c) **Low processing ability**
Internet of Things devices are lightweight and energy-constrained with low compu-

tational capacity. However, teal-time data-processing is required by ML algorithms, which presents a problem to the implementation of such resource-constrained devices.

(d) **Low memory capacity**
Data is created in diverse ways in the IoT context, necessitating huge memory in IoT devices. As a result, being able to offer an efficient solution for varied data poses a hurdle.

Moreover, employing all features in the design of an IDS can lead to the introduction of redundant and irrelevant features into the model. Therefore, feature optimization must be used to achieve good IDS performance [9]. There are three main approaches to feature optimization. The filter-based approach evaluates features according to predefined metrics, often using information theory. In contrast, a wrapper and embedded approach will evaluate features using an ML algorithm. In this current study, a wrapper-based feature optimization technique was used, specifically IDSBPSO as it gives efficient results as compare to other feature optimization methods [10]. SBPSO is a recently proposed BPSO variant that updates a particle's position, using the flipping probability rather than velocity. In SBPSO, a stickiness parameter is employed to maintain the momentum that is characteristic of PSO, meaning that a particle will tend to adhere to the position to which it has recently moved. PSO is a population-based stochastic optimization algorithm. Due to its easy feature-coding, computational reasonability, few parameters, and less demanding execution to address and select critical feature problems, the PSO algorithm is considered efficient and robustness to control parameters. There are various publicly available datasets for IoT networks, which include DARPA98, KDDCUP99, CAIDA (2007), ISCX 2012, ADFA-WD (2014), ADFA-LD (2014), CISIDS 2017, DS2OS (updated 20218), BOT-IoT (updated 2020) UNSW-NB15, and IoTID20.

The following are the contributions of the paper:

- The proposed IDSBPSO is based on a novel approach of dynamic bit-masking strategy to reduce the search space of the SBPSO. This approach iteratively applies a mask to features after a certain number of generations, in order to prevent those features from evolving further. Using such a method throughout the evolutionary process can significantly reduce the search space, allowing the IDSBPSO to identify better solutions within a smaller search space.
- Some parameters are set to dynamic, in order to investigate how this strategy can help balance exploration with exploitation, thereby further improving the searchability of SBPSO for the problems of optimising feature selection.
- The proposed strategy would be implemented on two IoT network datasets for feature optimization, since this strategy is proposed for the design of an anomaly detection system for IoT networks, as a means of reducing the computational cost of such networks when using devices of a constrained nature.

The proposed FS model will be tested on the 2 datasets, IoTID20 and UNSW-NB15. The proposed model obtain comparable or higher accuracy with reduced computational cost and less number of features compared to benchmark PSO based methods. The remainder of this paper is organised as follows: the literature review is presented in Section 2; Section 3 discusses the proposed framework architecture; Section 4 describes the implementation and evaluation of the results of the system experiments, and Section 5 concludes the paper, also making recommendations for future work.

2. Literature Review

Internet of Things network security remains a consistent research topic for security analysers. Hence, numerous IDSs have been proposed, based on various types of feature optimization and reduction methodologies. In [11], the authors propose a novel two-tier classification model based on ML methodologies, for example, the Naive Bayes, K nearest neighbors (KNN) classifier with certainty factor voting, and linear discriminant analysis (LDA) for feature reduction. This model has a high detection rate for sophisticated

attacks like User to Root (U2R) and Remote to Local (R2L), namely 34.81% and 67.16%, respectively. Conversely, in [12], the authors propose an effective deep learning approach: a self-taught learning (STL) IDS. The NSL-KDD dataset was used in the above-mentioned study, but the authors suggest a hybrid method for more accurate results. In [13], the authors suggest a feature selection technique using filter and wrapper methods, but these are computationally expensive. Meanwhile, in [14], the authors propose three IDS on K-means clustering, a decision tree, and a hybrid of these methods to achieve a maximum detection rate of 70–93%.

In [15], however, the authors propose a hybrid deep network, combining Convolutional Neural Network (CNN) with a gated recursive unit to detect intrusion. A PSO algorithm was utilised in the resulting study to select relevant features from the data, and a developing system successfully performed the feature selection and classification process automatically. Meanwhile, in [16], the authors present a semi-supervised ML technique for distributed denial of service (DDoS) detection, based on network entropy estimation, co-clustering, information gain ratio, and an extra-tree algorithm. This demonstrated good accuracy but with increased complexity. Conversely, in [17], the authors employed a variety of feature selection strategies, including a correlation coefficient, gain ratio, and information gain. The suggested experiment was carried out on random forest, rotation forest, and random committee classifiers.

Meanwhile, in [18], the authors present a feature selection-based IDS. The feature classification algorithm was based on a linear correlation coefficient. The cuttlefish algorithm was also used in this method to select features based on filter and wrapper, respectively. The FCC-CFA (feature grouping according to the linear correlation coefficient-cuttlefish algorithm) approach was created to extract the optimal subset of features from the dataset. This is a hybrid form of filter and wrapper method, retaining the advantages of each. The KDD Cup99 dataset was then used to test the suggested approach. The results of utilising the FGLCC-CFA algorithm revealed that compared with the CFA and FGLCC algorithms, the hybrid method was able to improve the accuracy and detection rate, while also reducing the number of false alarms.

In contrast, using a two-phase approach, the authors in [19] propose a hybrid intrusion detection model. Here, the first phase consisted of feature selection and the second, detecting an attack. A wrapper method called MGA-SVM was applied in the first phase. With multi-parent crossover and multi-parent mutation, this model combines the characteristics of SVM and GA (MGA). In the second phase, an artificial neural network (ANN) was used to detect attacks, and PSO was employed to improve the suggested model's performance. The proposed name of this model is therefore MGA-SVMHGS-PSO-ANN. It has a high detection accuracy of 99.3%, according to data from the NSL-KDD dataset.

On the other hand, specifically for lightweight IoT devices, the performance of a lightweight ML-based IDS was tested in [20], using a new feature selection technique. The technique was verified with a public dataset, acquired from an IoT environment for this work. In the above model, a new feature selection approach, referred to as correlated-set thresholding on gain-ratio (CST-GR), is proposed to create a lightweight system, while also positively affecting the detection rate.

In [21], however, the authors propose supervised ML algorithms to create a three-layer intrusion detection system, capable of detecting a variety of cyber-attacks in IoT networks. The resulting solution was tested in a smart home scenario with eight IoT gadgets. In [22], the authors designed a bottom-up EI architecture and proposed novel data driven dynamical control strategy. Moreover, Intelligent controllers augmented by deep reinforcement learning (DRL) techniques are adopted and the concept of curriculum learning (CL) is integrated into DRL to improve the sample efficiency and accelerate the training process. Similarly, in [23], the authors created a novel hybrid intrusion detection system (HIDS) for IoT threats. The developed HIDS ensemble was utilised to secure IoT devices by merging SIDS with AIDS. The results of the generated model revealed that the

HIDS was superior in its performance. Conversely, the model could not detect various types of attack on the IoT system.

According to the research cited above, various FS methods have been used in the past but when the data dimensionality increases then it cause serious challenge for optimisers, as search space increases dramatically. Choosing the right characteristics to maximise classification accuracy for anomaly detection in IoT networks, while at the same time reducing computational cost and prediction time, would still appear to be a work in progress. Various research exists on the design of anomaly detection systems for IoT networks, but these either use benchmark PSO-based methods, or a hybrid of optimization algorithms for feature selection. To close the gap in the literature, this study therefore provides an intelligent system, which uses novel approach to reduce search space and increase the exploration and exploitation ability of optimizer to select optimal features, while obtaining comparable or higher accuracy with reduced computational cost and prediction time.

3. The Proposed Model

This section proposes an enhanced approach to the design of an efficient and accurate IDS for IoT networks, using an IDSBPSO as an approach to feature selection. Particle swarm optimization (PSO) is a population-based stochastic optimization algorithm, proposed by Eberhart and Kennedy in 1995 [24]. Because of its easy feature-coding, computational reasonability, few parameters, and less demanding execution to address and select critical feature problems, the PSO algorithm is considered efficient [25]. The originally proposed PSO was a continuous one (CPSO), used to tackle a variety of continuous issues. The main drawback of PSO is that if a particle gets stuck in a local minimum (optimal), all the other particles will converge to that local minimum, resulting in erroneous solutions. Thus, before expanding the network, it is necessary to preserve particle diversity [26].

Particles are employed in the PSO method to represent solutions from the population of particles in the relevant space. This population is referred to as a swarm. Each particle in the swarm is represented by vector $x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,d})$, where d represents the number of features in the dataset, and each particle has d dimensional velocity $v_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,d})$. To enhance efficiency, PSO works randomly and travels in the search space to find relevant features by updating velocity and position with iterations. At each iteration, the particles' velocity and position are updated according to *pbest* and *gbest*, which are the best personal and global fitness values up until that iteration. According to [27], the position and velocity of particles is updated as in (1) and (2).

$$v_{i,d}^{k+1} = wv_{i,d}^{k} + c_1 r_1 (pbest_{i,d}^{k} - x_{i,d}^{k}) + c_2 r_2 (gbest_{i,d}^{k} - x_{i,d}^{k}) \quad (1)$$

$$x_{i,d}^{k+1} = x_{i,d}^{k} + v_{i,d}^{k+1} \quad (2)$$

where k represents k_{th} iteration and d represents d_{th} feature in the vector space. In addition, w represents the inertia factor that will give weightage to the previous velocity, and c_1 and c_2 are acceleration coefficients that give weightage to the cognitive and social term in the updated velocity. Meanwhile, r_1 and r_2 are uniform random numbers within [0, 1].

Velocity has three components, as illustrated in (1). The first component is momentum, depicting the influence of the present direction. Varying particles usually have different momentums, which help keep the swarm diverse, especially when everyone shares their finest experiences. Furthermore, momentum is the only factor that will allow a particle to continue seeking better solutions, once it has arrived at the best point discovered by the swarm so far. Conversely, the other two are cognitive and social components which guide particles towards an optimal experience, as well as that of each particle's neighbours.

Binary PSO was developed to solve combinatorial problems, including job-shop scheduling and feature selection. In BPSO, rather than adding velocity to position, in order

to obtain a new position, velocity is used to determine the probability of achieving the corresponding updated position values [27], which can be seen in (3).

$$x_d^{k+1} = \begin{cases} 1 & rand() \leq s(v_{i,d}^{k+1}) \\ 0 & otherwise \end{cases} \tag{3}$$

$$s(v_d^{k+1}) = 1/e^{-v_d^{k+1}} \tag{4}$$

Sticky BPSO (SBPSO) is a recently proposed BPSO variant that updates a particle's position, using the flipping probability rather than velocity. In SBPSO, a stickiness parameter is employed to maintain the momentum that is characteristic of PSO, meaning that a particle will tend to adhere to the position to which it has recently moved [28]. This is illustrated in (5).

$$x_d^{k+1} = \begin{cases} 1 - x_{i,d}^k & rand() \leq p_{i,d}^{k+1} \\ x_{i,d}^k & otherwise \end{cases} \tag{5}$$

where rand () is a random value in [0, 1] from the uniform distribution. Moreover, $p_{i,d}^{k+1}$ is the flipping probability of the i^{th} particle in the d dimension [28], which may be written mathematically as per (6).

$$p_{i,d}^{k+1} = n_s(1 - st_{i,d}^k) + n_p|pbest_{i,d} - x_{i,d}^k| + n_g|gbest_d - x_{i,d}^k| \tag{6}$$

where $st_{i,d}^k$ denotes the stickiness parameter of the i^{th} particle on the d^{th} dimension. Here, $pbest_{i,d}$ denotes the personal best of the i^{th} particle on the d^{th} dimension, and $gbest_d$ denotes the global best. Meanwhile, n_s, n_p, and n_g are the three parameters that give weightage to the particle's stickiness ability and its tendency to move towards pbest and gbest. The stickiness parameter, $st_{i,d}^k$ lowers over time, indicating that a bit is more likely to cling to its new position. According to [28], the updated $st_{i,d}^k$ mechanism is illustrated in (7).

$$st_{i,d}^k = \begin{cases} s_{i,d}^k - 1/M & x_{i,d}^{k+1} = x_{i,d}^k, s(v_{i,d}^k) > 0 \\ 1 & x_{i,d}^{k+1} \neq x_{i,d}^k \end{cases} \tag{7}$$

where M is the step parameter determining stickiness ability, which decreases from 1 to 0 as the number of iterations increases. Initially, $s_{i,d}^k = 1$ was set for $k = 0$. Dynamic SBPSO is a further variant of the SBPSO variant, proposed to control the exploration and exploitation ability of particles. In dynamic SBPSO, n_s, n_p, and n_g are used to increase exploration at the outset and increase exploitation at the end. Here, n_s and n_g linearly decrease in relation to an increase in the number of iterations, which can be seen in (8) and (10), respectively. Meanwhile, n_p linearly increases alongside the rising number of iterations, which can be seen in (9).

$$n_s = n_{s_{max}} - k/k_{max}(n_{s_{max}} - n_{s_{min}}) \tag{8}$$

$$n_p = n_{p_{min}} + k/k_{max}(n_{p_{max}} - n_{p_{min}}) \tag{9}$$

$$n_g = n_{g_{max}} - k/k_{max}(n_{g_{max}} - n_{g_{min}}) \tag{10}$$

where $n_{s_{max}}$ and $n_{s_{min}}$ are the maximum and minimum values for the n_s factor, $n_{p_{max}}$ and $n_{p_{min}}$ are the maximum and minimum values for the n_p factor, and $n_{g_{max}}$ and $n_{g_{min}}$ are the maximum and minimum values for the ng factor. Ultimately, k represents the k^{th} iteration, and $kmax$ represents the maximum number of iterations. The values applied for all these parameters can be seen in the subsection, 'Parameter Setup'.

Traditionally, during the evolutionary process, a BPSO algorithm searches in a fixed d-dimensional space (where d represents the number of original features). When d is large, setting a high number of particles or generations in the PSO algorithms demands significant processing resources. As a result, it is advantageous to include a search space reduction strategy, which can lower the computational resources required for the PSO applied to the feature selection task.

In this study, the dynamic bit-masking strategy was combined with DSBPSO. This first involved extracting information from the *pbests* of particles to determine which bits

should be masked. During the evolutionary process, the number of selected traits of all particles decreases. Even before the halting criterion is met, noisy or irrelevant features can be determined. After a certain number of generations, if a feature (bit) is not selected by all *pbests* in the swarm, it is very probable that this feature is useless, since solutions containing this feature are very likely to be eliminated for their poor fitness. The parameter that decides when a mask should be updated is μ. In this study, the mask update approach was adopted, because a bit is masked if it is not selected by all pbests in the swarm. This can be seen in Algorithm 1.

Algorithm 1: Search Space Reduction Strategy

Data: Particles pbests $P\_B=\{pbest_1, pbest_2, pbest_3,\ldots, pbest_v\}$, particles positions $P\_P=\{X_1^k, X_2^k, X_3^k,\ldots, X_v^k\}$, $X_i^k = \{x_{i,1}^k, x_{i,2}^k, x_{i,3}^k,\ldots, x_{i,N}^k\}$ where i={1,2,3,...,V}, unmasked bits $U\_B=\{d_1, d_2, d_3,\ldots, d_w\}$

Result: Updated U_B and P_P set

```
for d ∈ U_B do
    if Σ_{i=1}^{v} pbest_{i,d} = 0 then
        U_B ← U_B\{d};
        for i ← 1 to V do
            x_{i,d}^k ← 0;
        end
    end
end
return Updated U_B and P_P set;
```

In this algorithm, the *pbests* of particles are represented by a set, P_B, and the mask is denoted by U_B, given that each element in this set corresponds to an unmasked bit. The U_B set is updated by obtaining information from the pbest of each swarm. A bit is removed from U_B if it is not selected by all *pbests* in the swarm. During the algorithm's evolutionary phase, the set is updated. Some bits in U_B are masked each time the mask-update mechanism is run. The mask-update approach ensures a reduced search space, because only the bits in U_B can evolve. The position-updating mechanism can then be rewritten as in (11) [28].

$$x_{i,d}^{k+1} = \begin{cases} 1 - x_{i,d}^k & rand() < x_{i,d}^{k+1}, d \in U\_B \\ x_{i,d}^k & rand() \geq x_{i,d}^{k+1}, d \in U\_B \\ 0 & d \notin U\_B \end{cases} \quad (11)$$

According to the third condition, if $d \notin U\_B$, the position of that particle is assigned a value of 0, meaning that it is eliminated from the search space to reduce computational time and resources. This improvement can be seen in Figure 2, where the grey blocks show the improved SBPSO strategies.

The overall IDSBPSO-based feature selection procedure can be seen in Algorithm 2. The proposed approach first adopts a search space reduction strategy to reduce the number of features involved in the iteration update, and the mask is updated every $\mu.K$ iterations, with K as the maximum number of iterations.

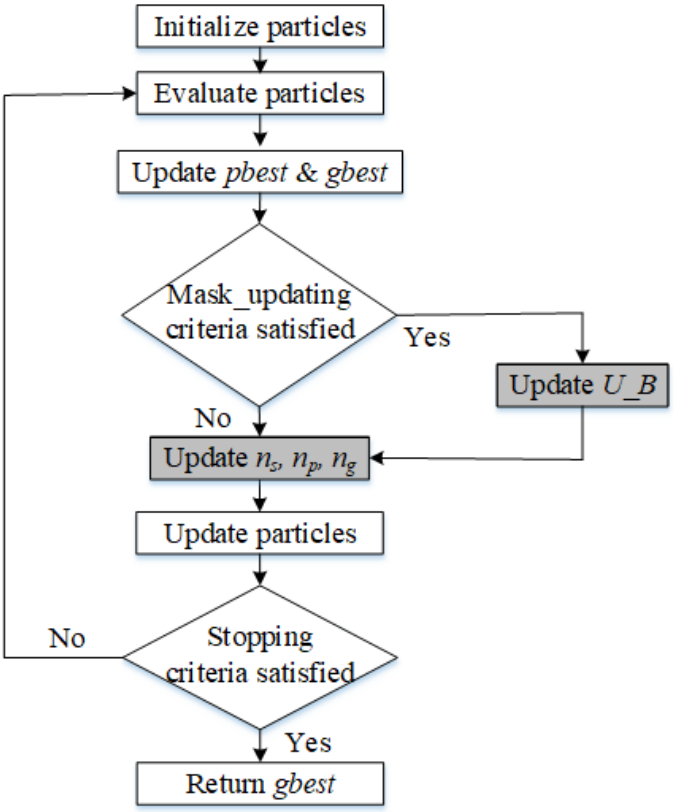


Figure 2. Flowchart for the IDSBPSO.

Algorithm 2: Pseudocode of the IDSBPSO-Based Feature Selection Method

Data: The training dataset with N features, maximum number of generations K, swarm size L;
Result: A set F_s of a selected features;
$k \leftarrow 0$, $U\_B \leftarrow 1, 2, \ldots, N$;
${P_1}^k \leftarrow (0)_x N$, ${S_1}^k \leftarrow (1)_x N$, i=1,2,…,L;
$S^k = \{{X_1}^k, {X_2}^k, \ldots, {X_L}^k\} \leftarrow$ Initialize each particle's position;
Evaluate the fitness value of each particle in S^k using Equation (10);
Update pbests and gbest ;
while $k < K$ **do**
  if *mask updating condition satisfied* **then**
    $U_B \leftarrow$Update U_B using **Algorithm 1**;
    $S^{k+1} \leftarrow S^k$;
  end
  for *particle* $i \leftarrow 1$ *to* L **do**
    $P_i^{k+1} \leftarrow$Update $i's$ probability vector using Equation (6)
    $X_i^{k+1} \leftarrow$Update i's position using Equation (11)
    $S_i^{k+1} \leftarrow$Update i's stickiness parameter vector using Equation (7)
    Evaluate the fitness value of i using Equation (12);
    Update pbest of i^{th} particle using best fitness value;
    Update gbest using X_i^{k+1} ;
  end
  $S^{k+1} = \{X_1^{k+1}, X_2^{k+1}, \ldots, X_L^{k+1}\}$;
  $k \leftarrow$k +1;
end
$F_s \leftarrow$ Decode gbest; ;
return F_s;

Figure 3 depicts the framework of the proposed model for a network IDS using IDSBPSO. The proposed system comprises a number of phases to obtain good accuracy and network suitability, as explained in this section.

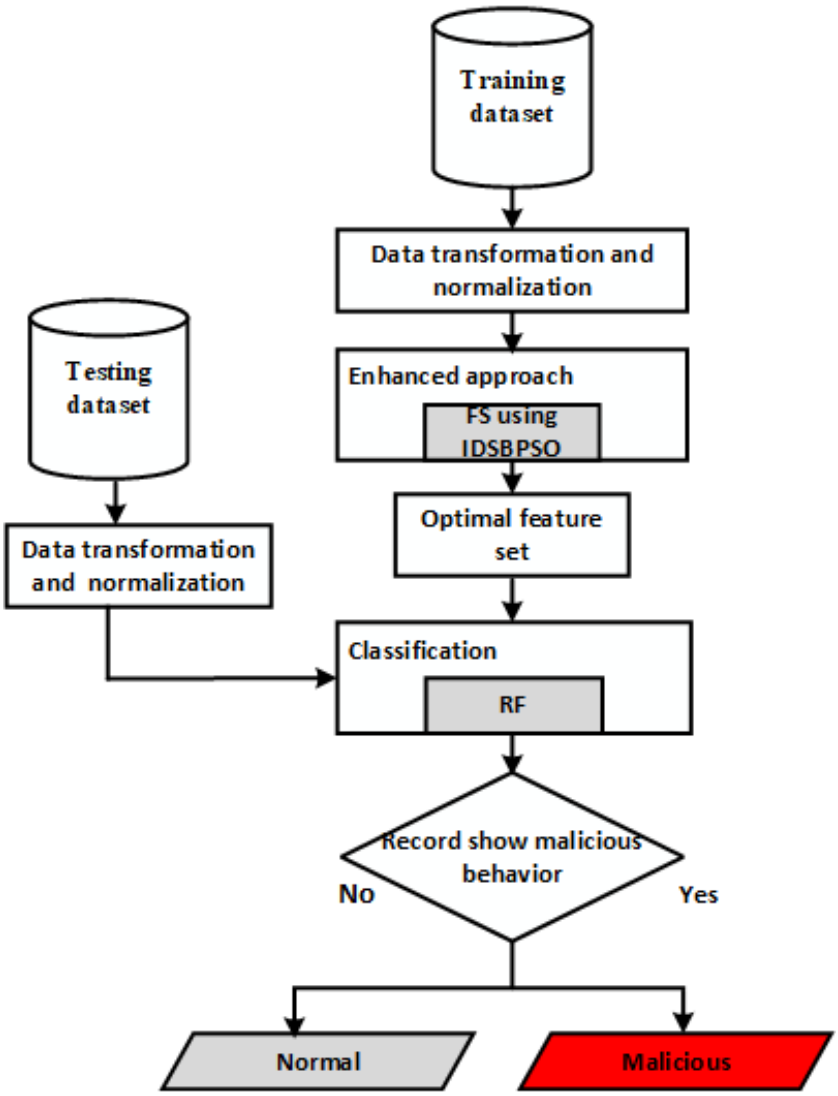


Figure 3. Working architecture of enhanced IDS, using IDSBPSO.

The two IoT datasets included IoTID20 and UNSW-NB15. The IoTID20 dataset was generated in 2020 [29] and contains a total of 83 network features. These network features can be seen in Table 1. There are also three label features in this dataset: binary, category, and sub-category, and four main attacks: Scan, Mirai, denial of service (DoS), and man in the middle (MITM). These attacks and their subcategories can be seen in Table 2.

Table 1. Features of the IoTID20 dataset.

Flow ID	Src IP	Src Port
Dst IP	Dst Port	Protocol
Timestamp	Flow Duration	Tot Fwd Pkts
Tot Bwd Pkts	TotLen Bwd Pkts	TotLen Fwd Pkts
Fwd Pkt Len Min	Fwd Pkt Len Max	Fwd Pkt Len Mean
Fwd Pkt Len Std	Bwd Pkt Len Max	Bwd Pkt Len Min
Bwd Pkt Len Mean	Bwd Pkt Len Std	Active Min
Active Max	Idle Mean	Idle Max

Table 1. *Cont.*

Flow IAT Max	Flow IAT Min	Fwd IAT Max
Fwd IAT Tot	Fwd IAT Mean	Fwd IAT Std
Fwd IAT Max	Fwd IAT Min	Bwd IAT Tot
Bwd IAT Mean	Bwd IAT Std	Bwd IAT Max
Bwd IAT Min	Fwd PSH Flags	Bwd PSH Flags
Fwd URG Flags	Bwd URG Flags	Bwd Header Len
Fwd Header Len	Fwd Pkts/s	Bwd Pkts/s
Pkts Len Min	Pkts Len Max	Pkt Len Mean
Pkt Len Std	Pkt Len Var	FIN Flag Cnt
Active Std	SYN Flag Cnt	RST Flag Cnt
PSH Flag Cnt	ACK Flag Cnt	URG Flag Cnt
CWE Flag Count	ECE Flag Cnt	Down/Up Ratio
Pkt Size Avg	Fwd Seg Size Avg	Bwd Seg Size Avg
Fwd Bytes/b Avg	Fwd Pkts/b Avg	Fwd Blk Rate Avg
Bwd Bytes/b Avg	Fwd Pkts/b Avg	Bwd Blk Rate Avg
Subflow Fwd Bytes	Subflow Bwd Bytes	Subflow Fwd Bytes
Subflow Fwd Bytes	Init Fwd Win Bytes	Init Bwd Win Bytes hline
Fwd Act Data Pkts	Fwd Seg Size Min	Active Mean
Idle Std	Idle Max	-

Table 2. Attack categories on the IoTID20 dataset.

Scan	**Mirai**	**DoS**	**MITM**
Host Port OS	Brute Force, HTTP Flooding, UDP Flooding	Syn Flooding	ARP Spoofing

The UNSW-NB15 is an advanced dataset used for IDS research. It is widely referenced in the literature. The UNSW-NB15 contains 42 network features, as listed in Table 3. There are two label features in this dataset: binary and category. Moreover, nine attacks may be seen in Table 4. Authentication, confidentiality, integrity, and availability are among the security needs targeted by these attacks. Accurate detection of these attacks is critical, as the consequences for IoT applications can be disastrous [30].

Table 3. Features of UNSW-NB15 dataset.

dur	proto	service
state	spkts	dpkts
sbytes	dbytes	rate
sttl	dttl	sload
dload	sloss	dloss
sinpkt	dinpkt	sjit
djit	swin	stcpb
dtcpb	dwin	tcprtt
synack	ackdat	smean
dmean	trans_depth	response_body_len
ct_srv_src	ct_state_ttl	ct_dst_ltm
ct_src_dport_ltm	ct_dst_sport_ltm	ct_dst_src_ltm
is_ftp_login	ct_ftp_cmd	ct_flw_htp_mthd
ct_src_ltm	ct_srv_dst	is_sm_ips_ports

Table 4. Attack categories of UNSW-NB15.

Generic	Exploits	Fuzzers
DoS	Reconnaissance	Analysis
Backdoor	Shellcode	Worms

Real-life datasets are high-dimensional because they incorporate vibrant information, received from a variety of IoT devices and sensors. When creating an ML model, it is essential to choose a set of meaningful, non-redundant features, because the quality of the

features will reduce the performance of the ML classifier [31,32] and the data will be unsuitable for IoT devices to work on. For this purpose, IDSBPSO-based feature optimization was used in this study. The feature optimization problem may be formulated in different ways. In many cases, there is a need to optimise features to reduce computational cost, while also increasing performance accuracy to enhance the generalisation capability. When choosing the best optimization technique based solely on prediction accuracy, performance will vary between the training and test sets [33]. Therefore, there are two main goals in feature selection: to improve classification performance and reduce the number of selected features. In [34], the aggregate fitness function is used to select best features with no change in accuracy, which can be shown in (12).

$$l = a \times l_1 + (1 - a) \times l_2 \tag{12}$$

where l_1 is the error rate, a is a constant giving weightage to the terms, and l_2 is the percentage of selected features, which can be seen in (13).

$$l_2 = p/n \tag{13}$$

where p represents the total number of selected features out of a total of N features. The value of a has been adjusted to 0.8 since it was suggested as being between 0.7 and 0.9 [35]. The selected features were then entered into the ML classifier. Random forest classification is used, this being a group of tree-structured classifiers in an ensemble technique. Each tree is built with a decision tree and different bootstrap sample from the original data. Each node of trees only selects a limited selection of features for the split. Out-of-bag (OOB) evaluation, which is an unbiased estimator of generalisation error, is performed on the learning samples that are not selected using the bootstrap. When a new sample needs to be classified after the forest has been built, it is fed into each tree in the forest. Each tree then casts a unit vote for a specific class, indicating the tree's judgement. When compared to typical ML classifiers, ensemble classifiers are strategies that may be adopted to build a powerful classifier with improved classification accuracy. The mathematical expression representing the model can be seen in (14).

$$C(x) = sign \sum_{j=1}^{m} (C_j(x)) \tag{14}$$

where j represents each classifier and m represents the total number of classifiers included in the classification or voting.

Random forest has the following advantages:

- It demonstrates excellent performance in accuracy on structured data.
- It is computationally efficient and can run on large-scale datasets with high dimensions.
- In most cases, it does not overfit and is robust against noise.
- It can handle unbalanced datasets.

4. Implementation and Evaluation of Results

This section discusses the experimental setup, evaluation metrics used to check the proposed model's performance, parameter setup, and experimental results, ending with an evaluation of the results of the proposed model.

(a) **Experimental Setup**

The suggested model's performance was evaluated on a Dell computer, running Microsoft Windows 10 Professional with Intel (R) Core (TM) i7-6500U and CPU at 2.50GHz 2.60 GHz, 2 cores and 4 logical processors, and 16 GB RAM. Feature selection and classification algorithms were implemented in the Python programming language (version 3.8). Anaconda Navigator was installed on the above-mentioned machine for the experimental setup.

(b) **Evaluation Metrics**

The performance of the proposed ML model may be evaluated using the following parameters: accuracy (AC), precision (PR), recall (RC), and F1-score $(F1S)$ [36]. The $F1S$ is the harmonic mean of PR and RC. Meanwhile, AC, PR, RC, and $F1S$ are computed as follows:

$$AC = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

$$PR = \frac{TP}{TP + FP} \tag{16}$$

$$RC = \frac{TP}{TP + FN} \tag{17}$$

$$F1S = \frac{2 \times (PR \times RC)}{PR + RC} \tag{18}$$

where each element in the above equations can be defined as follows:

- True Positive (TP): indicating that both the actual and predicted values are positive.
- True Negative (TN): indicating that both the actual and predicted values are negative.
- False Positive (FP): indicating that the actual value is negative, but the model predicted positive.
- False Negative (FN): indicating that the actual value is positive, but the model predicted negative.

In addition, computational time was used as an evaluation parameter to verify the efficiency of the proposed model, as the model is being proposed for energy constrained IoT devices.

(c) **Parameter Setup**

In IDSBPSO, the swarm size (total no. of particles to select best solution) was set at L = 20, the maximum number of generations was set at K = 50, and the step parameter (determines the stickness ability) was set at M = 50, as used in [37]. The parameter for updating mask μ was 0.25, given that this has been found to produce good results. Moreover, inertial weight n_s decreases from 0.9 to 0.2 and is calculated using (8). The acceleration constant np increases from 0.5 to 2.5 and decreases from 2.5 to 0.5 for n_g, using (9) and (10).

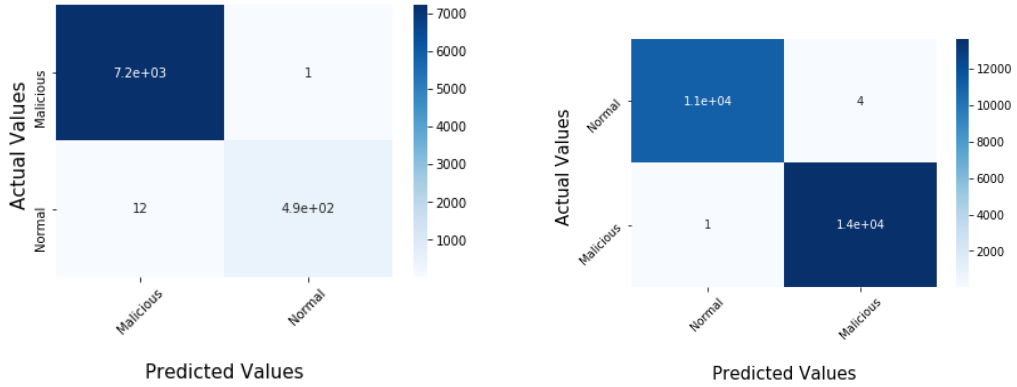
(d) **Experimental Results**

The experiment was carried out on the IoTID20 and UNSW-NB15 dataset, using the train test split validation method to conduct a detailed performance evaluation of the ML algorithms. The IoTID20 dataset contained 625,783 instances and the UNSW-NB15 dataset contained 2,540,044. Out of this data, 70% was used for training and 30% for validating the model. The binary classification of performance based on AC, PR, RC, and $F1S$ for both datasets can be seen in Table 5 for the normal and malicious network traffic using the proposed method. From the table, it is clear that for both network traffic datasets, the malicious behaviour is detected with almost 100% accuracy over the 20 runs.

Figure 4 illustrates the confusion matrix for the binary classification performance for both the IoTID20 and UNSW-NB15 datasets. From Figure 4, it is clear that FN and FP rates are very low which indicates good accuracy and low false alarm rate. While Figure 5 shows that the particle converges to optimal features rapidly with updated number of iterations using proposed IDSBPSO for both datasets.

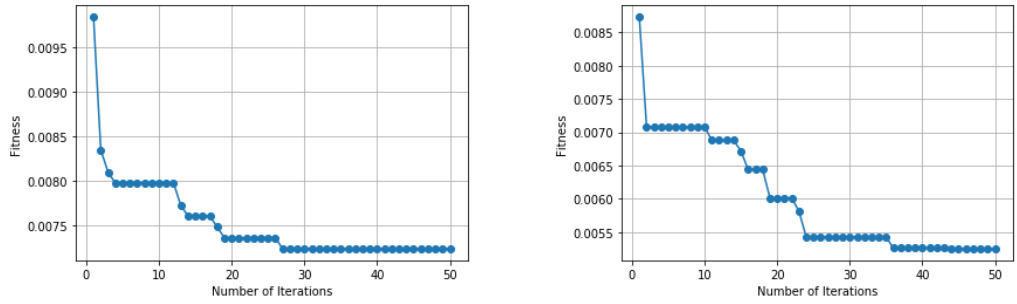
Table 5. Binary classification of normal and malicious traffic.

Traffic Category	AC	PR	RC	F1S
		IoTID20		
Normal	0.98	1.00	0.98	0.99
Malicious	1.00	1.00	1.00	1.00
		UNSW-NB15		
Normal	1.00	1.00	1.00	1.00
Malicious	1.00	1.00	1.00	1.00



(**a**) (**b**)

Figure 4. Confusion matrix for binary classification. (**a**) IoTID20 dataset; (**b**) UNSW-NB15 dataset.



(**a**) (**b**)

Figure 5. Convergence curve for binary classification. (**a**) IoTID20 dataset; (**b**) UNSW-NB15 dataset.

Meanwhile, Table 6 shows the category classification performance of the proposed model, using the evaluation parameters: *AC*, *PR*, *RC*, and *F1S*. From the table, it is clear that for both network traffic datasets, mostly attacks are detected with good accuracy except Mirai Ack flooding, Analysis, Backdoor, DoS, and Worms.

In addition, Figures 6 and 7 show the confusion matrix for the category classification performance for both the IoTID20 and UNSW-NB15 datasets, respectively. From the figures, it is clear that in IoTID20 dataset, attacks are detected with good accuracy except Mirai Ack flooding, While in UNSW-NB15 attacks such as Analysis, Backdoor, DoS, and Worms are detected with low accuracy of classification. Similarly, Figure 8 shows that the particle converges to optimal features rapidly for both the datasets with the updated number of iterations using the proposed IDSBPSO.

Figure 9 subsequently shows the number of selected features from the total number of features in both the IoTID20 and UNSW-NB15 datasets. There are total 83 features in IoTID20 datasets out of which only 30 optimal features are selected for training the model. Similarly, in UNSW-NB15, there are total 42 features out of which 15 have been selected. Figure 10 then illustrate the random forest prediction time in the proposed model for both the binary and category classification of the IoTID20 and

UNSW-NB15 datasets. As UNSW-NB15 dataset is larger as compared to IoTID20 dataset, therefore model takes more prediction time on it.

Table 6. Category classification of different attacks.

Traffic Category	AC	PR	RC	F1S
		IoTID20		
DoS Sync flooding	1.00	1.00	1.00	1.00
MITM ARP Spoofing	0.92	0.93	0.90	0.92
Mirai Ack flooding	0.34	0.35	0.34	0.34
Mirai-HTTP Flooding	0.94	0.92	0.96	0.94
Mirai Host brute force	0.96	0.95	0.97	0.96
Mirai-UDP Flooding	0.80	0.79	0.80	0.80
Normal	0.98	0.99	0.97	0.98
Scan Host port	0.65	0.73	0.56	0.64
Scan port OS	0.85	0.82	0.88	0.85
		UNSW-NB15		
Analysis	0.10	0.11	0.09	0.10
Backdoor	0.03	0.03	0.03	0.03
DoS	0.38	0.39	0.34	0.37
Exploits	0.73	0.70	0.76	0.73
Fuzzers	0.84	0.84	0.85	0.84
Generic	0.99	0.99	0.98	0.99
Normal	1.00	1.00	1.00	1.00
Reconnaissance	0.82	0.83	0.80	0.81
Shellcode	0.60	0.64	0.56	0.60
Worms	0.25	0.67	0.15	0.25

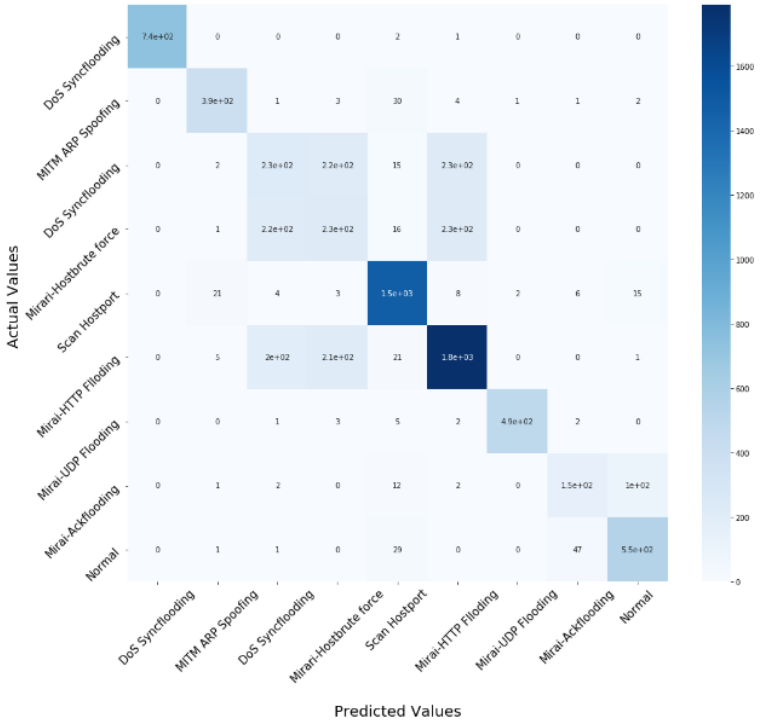


Figure 6. Confusion matrix for the multiclass classification of IoTID20.

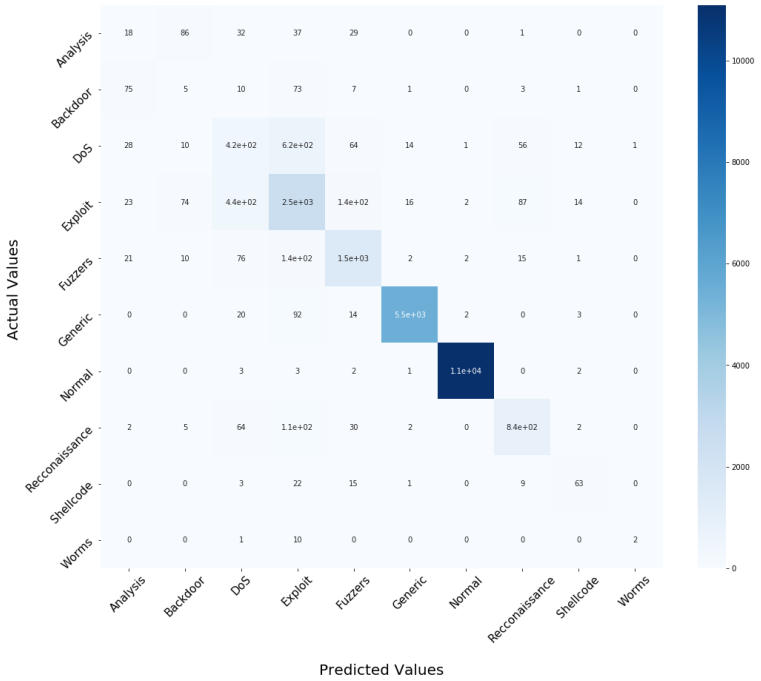
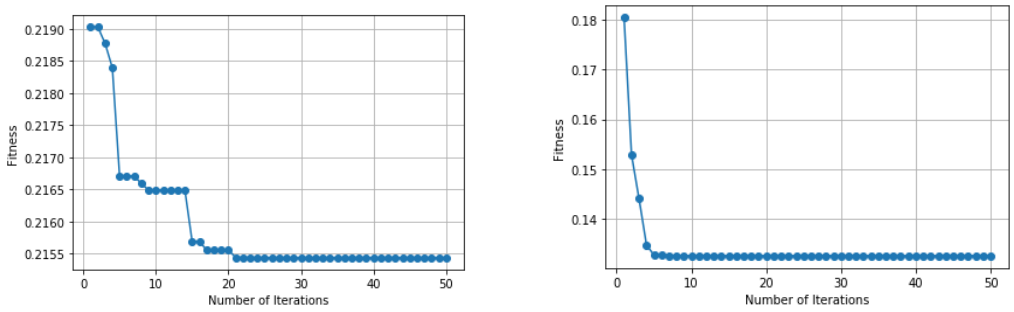


Figure 7. Confusion matrix for the multiclass classification of UNSW-NB15.



(**a**) (**b**)

Figure 8. Convergence curve for multiclass classification. (**a**) IoTID20 dataset; (**b**) UNSW-NB15 dataset.

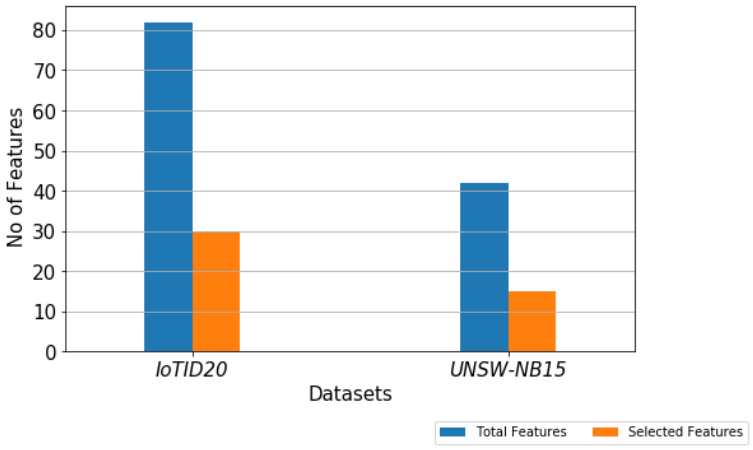
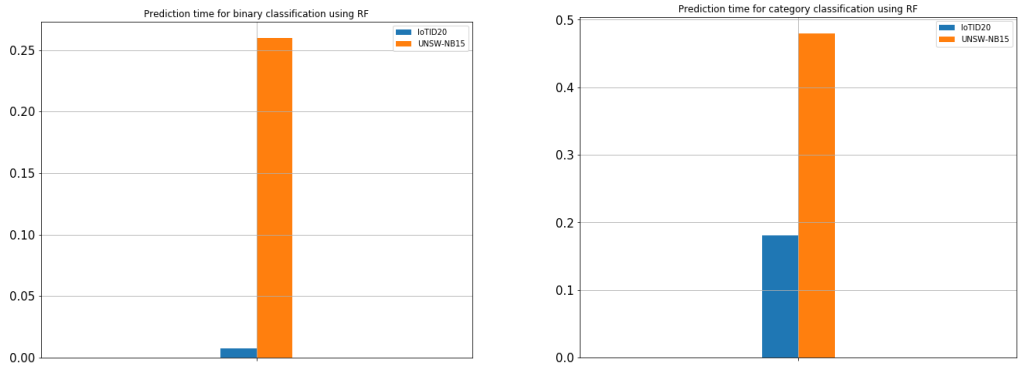


Figure 9. No. of selected features out of total features of IoTID20 and UNSW-NB15.



(**a**) (**b**)

Figure 10. Prediction time (min) for IoTID20 and UNSW-NB15. (**a**) Binary classification; (**b**) Category classification.

(e) **Evaluation of Results**

Tables 7 and 8 show the results of comparing the IDSBPSO with PSO-based benchmark methods. These PSO-based benchmark methods include SBPSO [37], DSBPSO [27], Up BPSO (UBPSO) [38], Quantum BPSO (QBPSO) [39], Sequential Forward Selection (SFS) [40] and Sequential Backward Selection (SBS) [40]. The results represent the mean value of 20 runs. The results in bold indicate improved computational time. As it can be seen in the table that IDSBPSO takes less time for FS as compared to other state of the art PSO based methods with almost similar accuracies and number of selected features for both datasets. Also the results indicate the improvement in computational time of IDSBPSO for both datasets.

Table 7. Results of the evaluation of binary classification.

Method	AC	FS	Computation Time (min)
	IoTID20		
SBPSO	99.80%	30	5.2
DSBPSO	99.84%	29	5.2
UBPSO	95.20%	34	5.8
QBPSO	98.35%	32	5.4
SFS	91.00%	25	5.0
SBS	86.56%	39	5.1
IDSBPSO	99.84%	30	**4.8**
	UNSW-NB15		
SBPSO	99.99%	17	42
DSBPSO	99.99%	21	39
UBPSO	98.43%	24	35
QBPSO	99.90%	18	33
SFS	87.64%	14	**30**
SBS	85.00%	29	34
IDSBPSO	99.95%	13	32

Table 8. Results of the evaluation of category classification.

Method	AC	FS	Computation Time (min)
	IoTID20		
SBPSO	79.12%	42	6.4
DSBPSO	79.00%	34	6.1
UBPSO	78.46%	40	6.4
QBPSO	79.03%	38	6.3
SFS	62.00%	30	**5.7**
SBS	60.89%	45	6.1
IDSBPSO	78.46%	37	6.0
	UNSW-NB15		
SBPSO	89.72%	19	45.3
DSBPSO	89.57%	19	38.7
UBPSO	86.90%	23	30.6
QBPSO	89.56%	21	29.9
SFS	79.45%	19	**27.0**
SBS	75.00%	25	28.2
IDSBPSO	89.52%	21	29.6

In the tables, it can be seen that the proposed IDSBPSO performs better in terms of accuracy rate and computational cost compared to most of the other PSO-based feature selection methods. IDSBPSO shows slightly more computational cost compared to SFS and SBS but has higher accuracy. This denoted that IDSBPSO is less efficient in terms of computational cost compared to SFS and SBS but better in terms of accurate prediction compared to SFS and SBS. The proposed model obtains a slightly lower accuracy compared with SBPSO and DSBPSO. This means that in some instances, IDSBPSO may remove some informative features, resulting in decreased accuracy compared to SBPSO and DSBPSO. It can be seen from the accuracy results of IoTID20 and UNSW-NB15 datasets that accuracy on UNSW-NB15 is greater compared to IoTID20 dataset. As UNSW-NB15 is a larger dataset compared to IoTID20 dataset so it may be possible that the proposed approach incorrectly mask some main features from already smaller dataset.

In short, the proposed IDSBPSO algorithm obtains higher accuracy while selecting fewer features with reduce computational cost compared with most of the state of the art PSO-based FS methods.

5. Conclusions

In this paper, an improved binary PSO algorithm called IDSBPSO is proposed for feature selection in classification. To improve feature selection performance, two mechanisms were adopted for IDSBPSO: a search space reduction strategy and a dynamic strategy to manage the contributions of momentum, pbest, and gbest to the movement of particles, thereby resulting in a balance between exploration and exploitation during the evolutionary process. The proposed method is used to design an anomaly based intrusion detection system for IoT networks due to its less demanding computational cost. Comparison was made on the basis of accuracy, precision, detection rate, F1 score, and computational time. The experimental results for two IoT network datasets demonstrated the effectiveness and efficiency of IDSBPSO. In most cases, IDSBPSO outperformed benchmark PSO-based feature selection methods by obtaining better or similar accuracy with less number of features. In particular, IDSBPSO significantly reduced computational time, compared with benchmark PSO-based feature selection methods, as it is designed for energy-constrained IoT devices.

Although the proposed IDSBPSO algorithm significantly reduced computational time, compared with the benchmark PSO algorithms it was still found to consume a significant amount of computational time, because a wrapper-based technique require extensive

computational time. Category classification accuracy of some attacks is not good. The proposed approach works better for large dimensional datasets while it is not more suitable for those datasets with less dimensions as it removes some informative features from them, which results in lower accuracy. Thus, in future, the authors will seek to improve the accuracy of subcategory classification and further reduce computational time. Moreover, IDSBPSO will be used in other applications, as this research was restricted solely to IoT network security. The performance of the proposed algorithm may also be tested using various other classifiers.

Author Contributions: Conceptualization, A.S., A.M.A., S.A. (Saleh Alyahya) and W.U.K.; Data curation, A.S., S.N.K.M. and W.U.K.; Formal analysis, A.M.A., S.N.K.M. and S.A. (Salman Ahmed); Investigation, S.A. (Salman Ahmed) and S.A. (Saleh Alyahya); Methodology, A.S., S.A. (Salman Ahmed) and W.U.K.; Project administration, S.N.K.M., S.A. (Salman Ahmed) and S.A. (Saleh Alyahya); Resources, A.M.A., S.A. (Salman Ahmed) and S.A. (Saleh Alyahya); Software, A.S. and S.N.K.M.; Supervision, A.M.A., S.N.K.M., S.A. (Salman Ahmed) and W.U.K.; Validation, A.S., S.N.K.M. and W.U.K.; Visualization, A.M.A., S.N.K.M. and S.A. (Saleh Alyahya); Writing—original draft, A.S. and W.U.K.; Writing—review & editing, A.S., S.A. (Salman Ahmed) and W.U.K. All authors have read and agreed to the published version of the manuscript.

Funding: The researchers would like to thank the Deanship of Scientific Research Qassim University for funding the publication of this project.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The IoTID20 dataset supporting this study was obtained from https://sites.google.com/view/iot-network-intrusion-dataset/home. This is newly developed data, generated in 2020. The UNSW-NB15 dataset was obtained from Kaggle https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
IDS	Intrusion Detection Systems
PSO	Particle Swarm Optimization
IDSBPSO	Improved Dynamic Sticky Binary Particle Swarm Optimization
IPS	Intrusion Prevention Systems
SIDS	Signature-based Intrusion Detection Systems
AIDS	Anomaly-based Intrusion Detection Systems
ML	Machine Learning
SBPSO	Sticky Binary Particle Swarm Optimization
BPSO	Binary Particle Swarm Optimization
KNN	K Nearest Neighbors
LDA	Linear Discriminant Analysis
U2R	User to Root
R2L	Remote to Local
STL	Self Taught Learning
CNN	Convolutional Neural Network
DDoS	Distributed Denial of Service
FGLCC-CFA	Feature Grouping according to the Linear Correlation Coefficient-Cuttlefish Algorithm
ANN	Artificial Neural Network
CST-GR	Correlated Set Thresholding on Gain Ratio
DRL	Deep Reinforcement Learning
HIDS	Hybrid Intrusion Detection Systems

CL	Curriculum Learning
CPSO	Continuous Particle Swarm Optimization
PBest	Personal Best
Gbest	Global Best
DoS	Denial of Service
MITM	Man in the Middle
OOB	Out-of-Bag
DRL	Deep Reinforcement Learning
AC	Accuracy
PR	Precision
RC	Recall
F1S	F1-Score

References

1. Gupta, A.; Tewari, B.B. Security, privacy and trust of different layers in Internet of Things framework. *Future Gener. Comput. Syst.* **2020**, *108*, 909–920.
2. Tewar, A.; Gupta, B.B. A novel ECC-based lightweight authentication protocol for Internet of Things devices. *Int. J. Higperformance Comput. Netw.* **2019**, *15*, 106–120. [CrossRef]
3. Thakare, A.; Lee, E.; Kumar, A.; Nikam, V.B.; Kim, Y.G. PARBAC: Priority-attribute-based RBAC model for azure IoT cloud. *IEEE Internet Things J.* **2020**, *7*, 2890–2900. [CrossRef]
4. Ferrag, M.A.; Maglaras, L.; Ahmim, A.; Derdour, M.; Janicke, H.; RDTIDS: Rules and decision tree-based intrusion detection system for internet-of things networks. *Future Internet* **2020**, *12*, 44. [CrossRef]
5. Almomani, I.; Qaddoura, R.; Habib, M.; Alsoghyer, S.; Al Khayer, A.; Aljarah, I.; Faris, H. Android ransomware detection based on a hybrid evolutionary approach in the context of highly imbalanced data. *IEEE Access* **2021**, *9*, 57674–57691. [CrossRef]
6. Karami, A.; Guerrero-Zapata, M. A fuzzy anomaly detection system based on hybrid PSO-Kmeans algorithm in content-centric networks. *Neurocomputing* **2015**, *149*, 1253–1269. [CrossRef]
7. Amouri, A.; Alaparthy, V.T.; Morgera, S.D. A machine learning based intrusion detection system for mobile Internet of Things. *Sensors* **2020**, *20*, 461. [CrossRef]
8. Sarker, I.H.; Kayes, A.S.M.; Badsha, S.; Alqahtani, H.; Watters, P. Cybersecurity data science: An overview from machine learning perspective. *J. Big Data* **2020**, *7*, 41. [CrossRef]
9. Keserwani, P.K.; Govil, M.C.; Pilli, E.S.; Govil, P. A smart anomaly-based intrusion detection system for the Internet of Things (IoT) network using GWO–PSO–RF model. *J. Reliab. Intell. Environ.* **2021**, *7*, 3–21. [CrossRef]
10. Khraisat, A.; Alazab, A. A critical review of intrusion detection systems in the Internet of Things: Techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. *Cybersecurity* **2021**, *4*, 18. [CrossRef]
11. Pajouh, H.H.; Dastghaibyfard, G.H.; Hashemi, S. Two-tier network anomaly detection model: A machine learning approach. *J. Intell. Inf. Syst.* **2017**, *48*, 61–74. [CrossRef]
12. Al-Qatf, M.; Lasheng, Y.; Al-Sabahi, K. Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. *IEEE Access* **2018**, 6, 52843–52856. [CrossRef]
13. Ghazy, R.A.; El-Rabaie, E.S.M.; Dessouky, M.I.; El-Fishawy, N.A.; El-Samie, F.E.A. Feature selection ranking and subset-based techniques with different classifiers for intrusion detection. *Wirel. Pers. Commun.* **2020**, *111*, 375-393. [CrossRef]
14. Shukla, P. ML-IDS: A machine learning approach to detect wormhole attacks in Internet of Things. In *Intelligent Systems Conference (IntelliSys)*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 234–240.
15. Ullah, A.; Javaid, N.; Samuel, O.; Imran, M.; Shoaib, M. CNN and GRU based deep neural network for electricity theft detection to secure smart grid. In *International Wireless Communications and Mobile Computing (IWCMC)*; IEEE: Piscataway, NY, USA, 2020; pp. 1598–1602.
16. Idhammad, M.; Afdel, K.; Belouch, M. Semi-Supervised machine learning approach for DDoS detection. *Appl. Intell.* **2018**, *48*, 3193–3208. [CrossRef]
17. Latah, M.; Toker, L. Towards an efficient anomaly-based intrusion detection for software-defined networks. *IET Netw.* **2018**, *7*, 453–459. [CrossRef]
18. Mohammadi, S.; Mirvaziri, H.; Ahsaee, M.G.; Karimipour, H. Cyber intrusion detection by combined feature selection algorithm. *J. Inf. Secur. Appl.* **2019**, 44, 80–88. [CrossRef]
19. Hosseini, S.; Zade, B.M.H. New hybrid method for attack detection using combination of evolutionary algorithms, SVM, and ANN. *Comput. Netw.* **2020**, 173, 107–168. [CrossRef]
20. Alyahya, S.; Khan, W.U.; Ahmed, S.; Marwat, S.N.K.; Habib, S. Cyber secure framework for smart agriculture: Robust and tamper-resistant authentication scheme for IoT devices. *Electronics* **2022**, *11*, 963. [CrossRef]
21. Anthi, E.; Williams, L.; Słowinska, M.; Theodorakopoulos, G.; Burnap, P. A supervised intrusion detection system for smart home IoT devices. *IEEE Internet Things J.* **2019**, *6*, 9042–9053. [CrossRef]
22. Hua, H.; Qin, Z.; Dong, N.; Qin, Y.; Ye, M.; Wang, Z.; Chen, X.; Cao, J. Data-Driven dynamical control for bottom-up energy internet system. *IEEE Trans. Sustain. Energy* **2022**, *13*, 315–327. [CrossRef]

23. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J.; Alazab, A. A novel ensemble of hybrid intrusion detection system for detecting Internet of Things attacks. *Electronics* **2019**, *8*, 1210. [CrossRef]
24. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
25. Wu, S.X.; Banzhaf, W. The use of computational intelligence in intrusion detection systems: A review. *Appl. Soft Comput.* **2010**, *10*, pp. 1–35. [CrossRef]
26. Bharti, V.; Biswas, B.; Shukla, K.K. A Novel Multiobjective GDWCN-PSO Algorithm and Its Application to Medical Data Security. *ACM Trans. Internet Technol.* **2021**, *21*, 1–28. [CrossRef]
27. Nguyen, B.H.; Xue, B.; Andreae, P.; Zhang, M. A new binary particle swarm optimization approach: Momentum and dynamic balance between exploration and exploitation. *IEEE Trans. Cybern.* **2019**, *51*, 589–603. [CrossRef]
28. Li, A.; Xue, B.; Zhang, M. Improved binary particle swarm optimization for feature selection with new initialization and search space reduction strategies. *Appl. Soft Comput.* **2021**, *106*, 107302. [CrossRef]
29. Alkahtani, H.; Aldhyani, T.H.H. Intrusion detection system to advance Internet of Things infrastructure-deep learning algorithms. *Complexity* **2021**, *2*, 18. [CrossRef]
30. Qaddoura, R.; Al-Zoubi, A.M.; Almomani, I.; Faris, H. Predicting different types of imbalanced intrusion activities based on a multi-stage deep learning approach. In Proceedings of the International Conference on Information Technology (ICIT), Amman, Jordan, 14–15 July 2021; p. 858863.
31. Long, J.; Zhang, S.; Li, C. Evolving deep echo state networks for intelligent fault diagnosis. *IEEE Trans. Ind. Inf.* **2020**, *16*, 4928–4937. [CrossRef]
32. Long, J.; Mou, J.; Zhang, L.; Zhang, S.; Li, C. Attitude data-based deep hybrid learning architecture for intelligent fault diagnosis of multi-joint industrial robots. *J. Manuf. Syst.* **2020**, *61*, 736–745. [CrossRef]
33. Sarhani, M.; Vob, S. PSO-Based cooperative learning using chunking. In Proceedings of the International Conference on Learning and Intelligent Optimization, Athens, Greece, 24–28 May 2020; pp. 278–288.
34. Sarhani, M.; Vob, S. Chunking and cooperation in particle swarm optimization for feature selection. In *Annals of Mathematics and Artificial Intelligence*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 1–21.
35. Vignolo, L.D.; Milone, D.H.; Scharcanski, J. Feature selection for face recognition based on multi-objective evolutionary wrappers. *Expert Syst. Appl.* **2013**, *40*, 5077–5084. [CrossRef]
36. Abdulhamme, R.; Musafer, H.; Alessa, A.; Faezipou, M.; Abuzneid, A. Features dimensionality reduction approaches for machine learning based network intrusion detection. *Electronics* **2019**, *8*, 322. [CrossRef]
37. Nguyen, B.H.; Xue, B.; Andreae, P. A novel binary particle swarm optimization algorithm and its applications on knapsack and feature selection problems. In *Intelligent and Evolutionary Systems*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 319–332.
38. Liu, J.; Mei, Y.; Li, X. An analysis of the inertia weight parameter for binary swarm optimization. *IEEE Trans. Evol. Comput.* **2016**, *20*, 666–681. [CrossRef]
39. Jeong, Y.; Park, J.; Jang, S.; Lee, K.Y. A new quantum-inspired binary PSO: Application to unit commitment problems for power systems. *IEEE Trans. Power Syst.* **2010**, *25*, 1486–1495. [CrossRef]
40. Kohavi, R.; John, G.H. Wrappers for feature subset selection. *Artif. Intell.* **1997**, *97*, 273–324. [CrossRef]