DOI: 10.15849/IJASCA.240730.05

Int. J. Advance Soft Compu. Appl, Vol. 16, No. 2, July 2024
Print ISSN: 2710-1274, Online ISSN: 2074-8523
Copyright © Al-Zaytoonah University of Jordan (ZUJ)

Hybrid Supervised Machine Learning-based Intrusion Detection System of Internet of Things

Mohamed Abdullah Shenify, Amjad Saeed Alghamdi, Adil Fahad Alharthi

Department of Computer Science, College of Computing and Information, Albaha University
e-mail: maalshenify@bu.edu.sa, 443022340@stu.bu.edu.sa, afalharthi@bu.edu.sa

Abstract

The Internet of things (IoT) is an embedded network of networked computing devices found in everyday objects, allowing them to send and receive data and makes our lives comfortable. Therefore, IoT is a very important technology and at the same time security has become a challenge, due to several issues, including poor interoperability, security flaws, privacy concerns, and lack of industry standards. Cyber-attacks on the IoT may impact security and privacy. To address the IoT-related security issue, nowadays, intrusion detection system (IDS) and machine learning technique are commonly used. In addition, to speed up the detection process, optimal features selection techniques are incorporated into the IDS. This paper proposes an efficient IoT IDS based on a hybrid feature selection and supervised machine learning-based technique. To examine the robustness of the proposed IDS, experiments are carried out on two datasets, i.e. NSL-KDD and UNSW-NB15. The proposed IDS achieves a detection accuracy rate of 99.3% on the NSL-KDD dataset, and 99.4% on UNSW-NB15 dataset. The proposed IDS was also compared to the deep learning-based IDS and found out that the proposed IDS achieves better performance in term of accuracy and learning runtime.

Keywords: *features selection, machine learning, SMO, SVM, intrusion detection system (IDS), internet of things (IoT).*

1 Introduction

Recently, the IoT is considered a critical technology, and thus its security has become challenging for researchers. This challenge is due to several issues, consisting of inadequate industry standards, security holes, privacy issues, and poor interoperability, which may leads to serious cyber-attacks [1].

The speed of data transfer in the IoT is a basic requirement, and the transfer of this data at a high speed and in large quantities of data leads to a fragility in security, which makes it vulnerable for hacking [2].To address the security issues of the IoT networks/systems, there are need for IDS and machine learning to solve these issues [3,4]. Machine learning helps us to solve the problems by detecting the malware in encrypted traffic, identify insider threats, predict online "bad neighborhoods" to keep users safe, or protect cloud data by identifying suspicious user behavior by continuously learning through analyzing data and identifying patterns [5,6,7].

Received 4 January 2024; Accepted 25 May 2024

Fig. 1 illustrates the IoT networks/systems' vulnerability to hacker attacks. The services provided by the IoT system can be accessed through internet connection, thus, the system is exposed to any possible attacks. The existing IDS may not be able to distinguish the attack traffic from legitimate users' traffic, which results in a low accuracy detection rate.

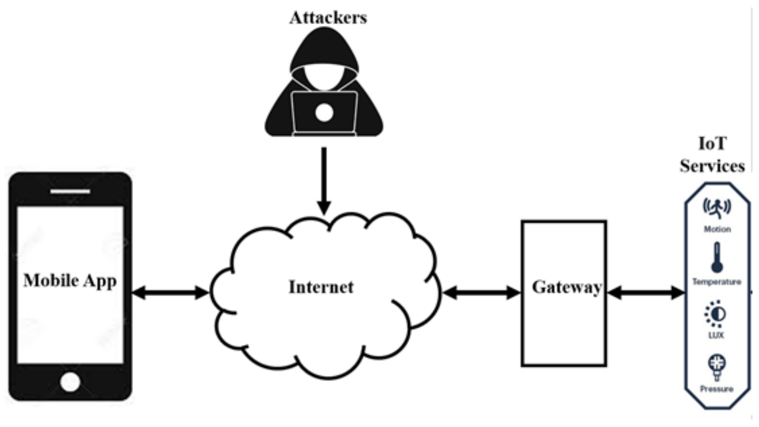

Fig. 1. Problem of low accuracy detection on IoT networks/systems

Fig. 2 illustrates the process of incorporating the machine learning as classifier into IoT IDS. The IoT devices collect the captured data and store them into a database. Then the data is cleaned for inputting to the machine learning classifier. The classifier produces two classes of traffic data, i.e.: attack and normal traffics.

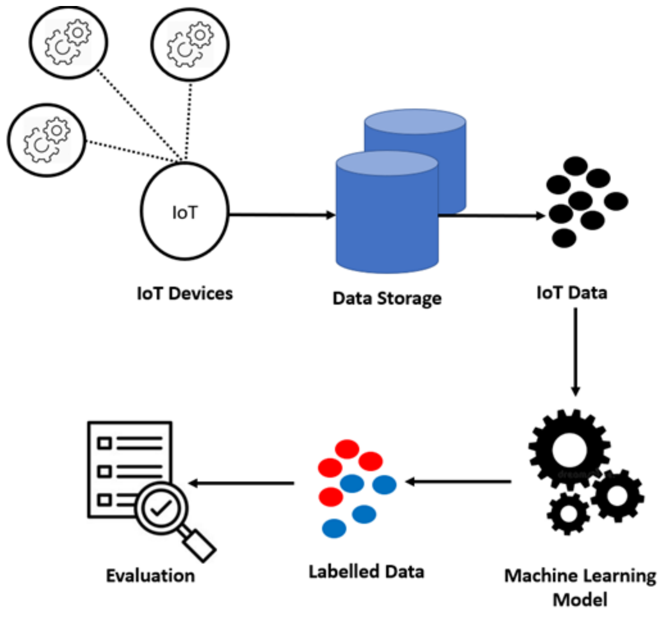

Fig. 2. Incorporating the machine learning into IoT IDS

Many works on securing IoT networks have been carried out, however, the challenges remain there especially in term of accuracy, false positive rate and slow response time, due to the limitation of the IoT devices on computing resources such as: power, processor, memory and storage. IDS and machine learning are considered the best to be employed to serve the IoT. The IDS is used to detect anomaly/attack traffic in the network. Detecting harmful contents embedded within traffic using machine learning guarantees a high accuracy of detection. Consequently, low accuracy in building IDS systems is critical, and the need for high-accuracy systems is crucial. The most key factor that degrades the accuracy of detection of anomaly/attack traffic in IDS is the training of machine learning system on low quality datasets (i.e., the dataset is not prepared well) and the present of irrelevant and redundant features which in turn leads to low accuracy and high computational time in detecting attacks.

The rest of the paper is structured as follows. Section 2 provides theoretical background and related works, followed by the proposed method in Section 3. Section 4 presents experimental results and discussion. Lastly, Section 5 provides conclusion and future works.

2 Theoretical Background and Related Work

This section discusses theoretical background on feature extraction/selection and supervised machine learning, followed by some related works on utilizing feature extraction/selection and supervised machine learning on solving issues of IoT IDS.

2.1. Feature extraction/selection

In network traffic analysis, feature extraction and selection are crucial processes for tasks like intrusion detection, traffic classification, and anomaly detection. They involve transforming raw network data into a more manageable and informative format for machine learning algorithms.

Feature extraction involves identifying and extracting relevant characteristics from this raw data that can be used to differentiate between different types of traffic or identify potential security threats. Common features extracted from network traffic include: packet headers (Source and destination IP addresses, source and destination port numbers, protocol type (TCP, UDP, ICMP, etc.), packet size, flags (SYN, ACK, FIN, etc.); flow statistics (Number of packets in a flow, total bytes transferred, flow duration, inter-arrival time between packets, incoming or outgoing packet; Application Layer Features (Extracted from higher-level protocols like HTTP or DNS, may include URLs, user agents, and content type).

Feature selection (FS) identifies a subset of features that are most relevant to the specific task at hand, because not all extracted features are equally important or informative. The feature selection aims to reduce computational complexity, improve model performance and enhance interpretability. Common feature selection techniques include: Filter methods, Wrapper methods and embedded methods. Some machine learning algorithms like decision trees perform automatic feature selection during training. The choice of features and selection techniques depends on the specific application and the type of network traffic being analyzed. Domain expertise is often valuable in feature selection, as human understanding of network protocols and security threats can guide the selection process.

When it comes to feature selection, there are several algorithms and techniques available to subset the feature space and improve the performance and efficiency of machine learning models. This study uses SMO as it is a global optimization technique. The fission-fusion social structure that spider monkeys exhibit during their foraging behavior served as the model for SMO. The notions of self-organization and work division are fundamental to swarm intelligence. Swarm intelligence-based algorithms, or SMOs, have gained traction in the last several years and are currently being used for a wide range of engineering optimization problems. The SMO algorithm's fundamental characteristic is that it iteratively chooses only subsets of size two to push chunking to the limit before optimizing the target function using the provided features. Due to its lack of a requirement, this technique can analytically solve the problems [8].

2.2. Spider Monkey Optimization (SMO)

The SMO algorithm is a relatively new optimization algorithm inspired by the social foraging behavior of spider monkeys [8]. It falls under the category of swarm intelligence algorithms, which utilize a population-based approach to find optimal solutions in complex search spaces. The SMO leverages the concept of fission-fusion social structure, i.e.: break down into smaller subgroups for foraging and then come together to share information, to achieve a balance between exploration (finding new areas of the search space) and exploitation (focusing on promising areas) during the optimization process. Components of the algorithm are:

- Initialization: A population of "spider monkeys" (potential solutions) is randomly initialized within the search space. The size of the population is a parameter of the algorithm. Each spider monkey represents a candidate solution to the optimization problem being addressed.
- Local Leader Phase: Each spider monkey interacts with its immediate neighbors and updates its position based on the best solution (local leader) it finds in its vicinity. This promotes local exploration within subgroups
- Global Leader Phase (Optional): If a spider monkey's position hasn't improved for a certain number of iterations, it considers the overall best solution found so far (global leader). This global leader can trigger a split, depending on the specific implementation.
- Local Leader Learning Phase: To maintain diversity in the population and prevents premature convergence on suboptimal solutions, each spider monkey updates its position based on a combination of: Its current position, the local leader it encountered, and a self-confidence factor (also called inertia weight)
- Global Leader Learning Phase (Optional): A similar learning phase can be applied based on the global leader, allowing monkeys to potentially jump to more promising areas of the search space.
- Local Leader Decision Phase: If the local leader has not improved for a specific number of iterations, it can decide to split the subgroup it belongs to. This creates smaller, diverse subgroups that can explore different areas of the search space.
- Global Leader Decision Phase (Optional): If the global leader hasn't improved for a certain number of iterations, it can trigger a population-wide split, essentially restarting the search with more diverse subgroups.

Overall the SMO goals are balancing exploration and exploitation, maintaining population diversity and leverage social learning.

2.3. Supervised machine learning

Machine Learning is field of computer science concerned with creating algorithms that can learn from data without explicit programming. Supervised means that the learning process is supervised by providing labeled data, where each data point has a corresponding label or desired output. Supervised learning is a powerful tool that has revolutionized various industries. Key concepts in supervised learning are:

- Labeled Data: The core of supervised learning is the use of labeled data that consists of input features and corresponding outputs. The model learns the relationship between these features and outputs.
- Training: The labeled data is used to train the machine learning model. During training, the model adjusts its internal parameters based on the provided examples.
- Prediction: Once trained, the model can make predictions for new, unseen data.

Nevertheless, Supervised Learning has challenges i.e.: data dependence and over-fitting.

2.4. Support vector machine (SVM)

The SVM is a supervised learning algorithm that analyzes data for classification and regression tasks and a fundamental machine learning algorithm with a wide range of applications, which is often used in conjunction with other machine learning techniques for optimal performance. SVM algorithm is a tool that helps to draw the best dividing line in a space with multiple dimensions (e.g.: height, width, and depth) to separate different things. This line is called a hyperplane. The SVM is able to find the most important data points, such as cornerstones, to build the best possible hyperplane. Fig. 3 shows an example where a hyperplane neatly separates two categories [9].

The SVM operates by finding the optimal hyper-plane (a decision boundary in high-dimensional space) that separates data points belonging to different classes with the maximum margin. This margin refers to the distance between the hyper-plane and the closest data points from each class, called support vectors. These support vectors are the data points closest to the hyper-plane from each class. They are crucial for defining the decision boundary and influence the SVM's model. This definition captures the essence of SVMs: their focus on finding an optimal separation boundary based on maximizing the margin and their reliance on support vectors to achieve that separation. Choosing the right kernel function is crucial for effective SVM applications.

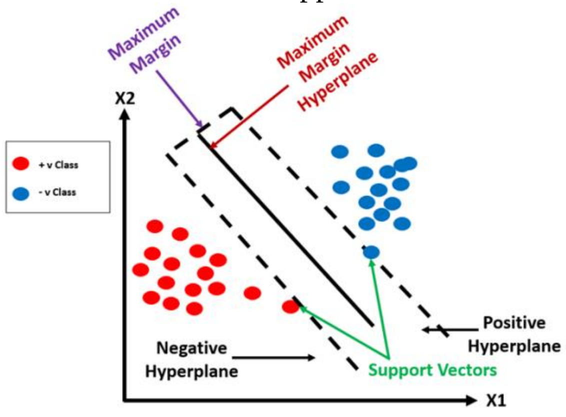

Fig. 3. Support Vector Machine

2.5. Related Works

There some researches on feature extraction and selection in IoT network traffic in the literatures. Susanto, et al. [10] has discussed the dimensionality reduction effect on IoT intrusion detection system performance; dimensionality reduction using fast ICA for IoT botnet detection [11]; Karim, et al. discuss multifactor-based clustering scheme for internet of vehicles [12]. Stiawan et al. propose machine learning-based IoT IDS, using K-Means classifier [13], and using Deep Autoencoder as feature selection and Artificial Neural Network as classifier [14]. Sharipuddin, et al. [15], combine deep learning and feature extraction algorithm for intrusion detection on heterogeneous IoT network, while Stiawan, et al. [16], Kurniabudi et al. [17] and Abbas, et al. [18] discuss ensemble machine learning-based IDS for IoT networks. Table 1 summarizes research works on machine learning-based IoT IDS.

Table 1: Summary on machine learning-based IoT IDS

Method	Ref.#	Dataset	Note
Machine learning (ML)	[19],	KDD Cup'99	Bidirectional Recurrent Neural Network
	[20]	IoTID20	Random Forest classifier
	[13]	Created from real traffic	K-Means clustering
ML+FS	[14]	MedBIoT	Artificial Neural Networks + Deep Autoencoder
Deep learning	[21]	Bot-IoT	Convolution Neural Network
	[22]	TON_IoT_Weather	Deep Belief Network
	[23]	CICIDS2017	Hybrid Weighted Deep Belief Network
Deep learning + FS	[24]	NSL-KDD, UNSW-NB15	Semi-parallel deep neural network (SPDNN)+ SMO
	[25]	NSL-KDD, IoTID20	Multilayer Perception, J48, and IBk + Information Gain
Ensemble	[26]	UNSW-NB15, NIMS	AdaBoost ensemble learning method using 3 machine learning techniques: decision tree, Naïve Bayes, and artificial neural network,
	[27]	NSL-KDD, UNSW-NB15	Automatic Model Selection Method
	[17]	CICIDS2017	PSO Search and Random Forest
Ensemble + FS	[28]	CICIDS2017, NSL-KDD, and UNSW-NB15	ensemble classifier using K-means, One-Class SVM, DBSCAN, and Expectation-Maximization

3 The Proposed Method

This paper proposes an efficient IoT IDS based on a hybrid feature selection and supervised machine learning-based technique. The proposed feature selection method adopts Spider Monkey Optimization (SMO) algorithm to discard irrelevant and redundant features. Furthermore, a sampling method based on Weighted Prototype approach is applied to reduce the dataset complexity and enhance the quality of data by selecting significant representative instances. The proposed IoT IDS is then evaluated using optimal features and the representative instances on two datasets to measure its robustness and efficiency. Fig. 4 shows the architecture of the proposed IoT IDS, where the details of the components are described in the following sections.

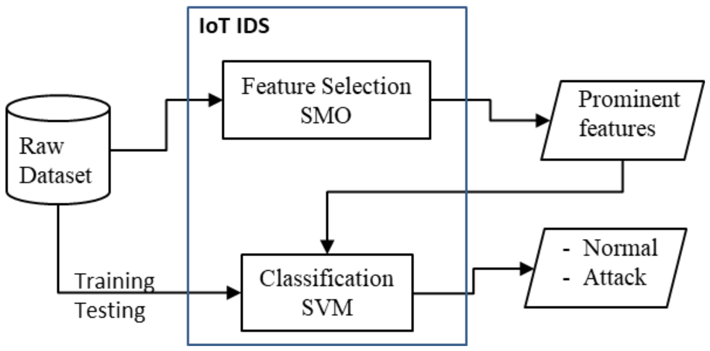

Fig. 4. The proposed IoT IDS architecture

Fig. 5 depicts the steps of the development of the proposed IoT IDS. Each step in Fig. 5 is explained in detail in the following subsections.

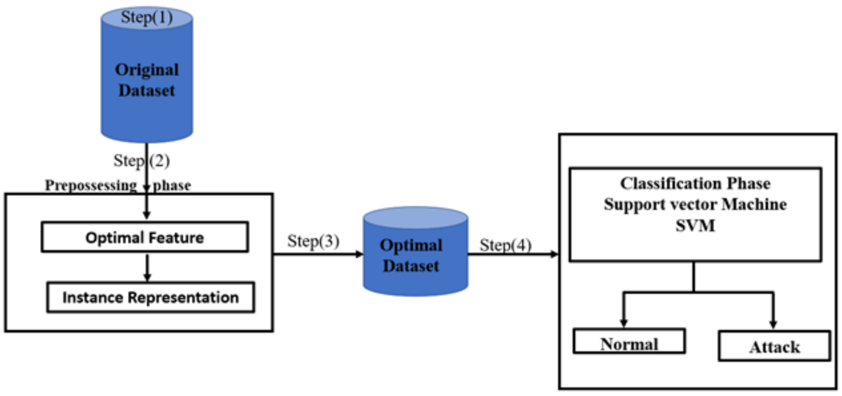

Fig. 5. Development steps of the proposed IoT IDS

3.1. Step 1: Dataset preparation

NSL-KDD dataset

The NSL-KDD dataset is often preferred over the other dataset because it has 41 features and 125972 records. The NSL-KDD dataset was specifically developed to address some of the inherent problems of the KDD'99 dataset [29]. While it is true that the KDD data set has been criticized for not being a perfect representation of existing real networks, it is still considered a valuable resource for researchers in the field of intrusion detection. Furthermore, The NSL-KDD dataset is often preferred over the other dataset because it allows for experiments to be run on the entire dataset, rather than just a small portion to obtain in more consistent and comparable evaluation results [30]. The dataset is divided into two categories the first category is normal traffic which contains 67,342 records and the second category is attack traffic, 58,630 records. The percentage of benign and attack data is relatively close, i.e.: 53%: 47%.

UNSW-NB15 dataset

The UNSW-NB15 network dataset contains a comprehensive data for network intrusion detection systems. Nine distinct attacks are included such as: DoS, malware, backdoors, and fuzzers. The dataset has 2,000,000 records with 14 features [31,32]. The dataset contains the imbalanced data where 93% of the records are normal class and only 7% of the records are the attack class.

3.2. Step 2: Pre-processing phase

In this step, a stable environment for the dataset for the purpose of efficient work is created. The pre-processing phase is divided into two main steps, i.e.: features optimization and instance reduction.

Feature optimization

Machine learning model converts a collection of input data points, or features, into a predictor or target variable. To accurately anticipate the target variable of given new data and an unknown target, this procedure seeks to train the model a pattern or mapping between these inputs and the target variable [33]. A model that can predict with the maximum precision level for any given data collection is created. There are numerous aspects in machine learning that affect how well the model performs. These aspects often consist of the algorithm choice, the parameters used in algorithm, the quality of the dataset and the features used to train the model [34,35], Feature selection helps in avoiding noise and considering only useful data to solve the problem that is attempted to solve. In this step, the model was trained using relevant data only. This paper uses SMO algorithm as feature selection method as shown in Algorithm 1.

Algorithm 1 Spider Monkey Optimization algorithm (SMO)
```
Input: Orginal_Data_Set
Processing:
InitializeDataSet(Orginal_Data_Set,Max_Number_of_
Iterations,Data_Set_Size)
while iterations < Max_Number_of_ Iterations:
EvaluateFitness( )
Data_Set = GetOptimalAndWorstDataSat()
for each Column in Data_Set:
if Column! = "best":
Optimal_Data_Set=UpdateFeatures(Column)
Iterations++
end while
Output: Optimal Data Set
```

Instance reduction

It is well known that to avoid excessive storage and time complexity and, potentially, to improve generalization accuracy by avoiding noise and over-fitting, it is frequently necessary to either reduce the initial training set by removing some instances before the learning phase or to modify the instances using a new representation. [36,37]. This paper uses Weighting Prototype technique where the weights for each instance are calculated in terms of both nearest neighbors using gradient descent and then removes any instances with weights that are more than a predetermined threshold [38]. The steps are described in full in Algorithm 2.

Algorithm 2 Instance Weight Prototype
```
Input: Orginal_Data_Set
Processing:
Orginal_Data_Set_Rows = GetRows(Orginal_Data_Set)
InitializeDataSet(Orginal_Data_Set,Max_Number_of_
Iterations,Data_Set_Rows_Size)
while iterations < Max_Number_of_ Iterations:
EvaluateFitness( )
Optimal_Worst_Data_Set_Rows                                    =
GetOptimalAndWorstDataSatRows(Orginal_Data_Set_Rows)
for each Row in Optimal_Worst_Data_Set_Rows:
if Row! = "best":
Optimal_Data_Set_Rows = UpdateFeatures(Row)
Iterations++
end while
Optimal_Data_Set = GetDataSet(Optimal_Data_Set_Rows)
Output: Optimal Data Set
```

3.3. Step 3: Optimal Dataset

After the last step is completed, we get an optimal dataset (i.e. valid dataset) which is cleaner, more stable, and fuller of relevant data. Algorithm 3 presents the steps in details.

Algorithm 3 Optimal Data Set
```
Input: Orginal_Data_Set
Processing:
Orginal_Data_Set = ReadDataSet (file_name.csv)
If ( IsDataSetNull (Orginal_Data_Set))
```

```
Orginal_Data_Set = ReprocessDataSet (file_name.csv)
ShowDataSet(Orginal_Data_Set)
Optimal_Columns = DecreaseColumnsBySMOAlgorithms (Orginal_Data_Set)
Optimal_Rows = DecreaseRows (Optimal_Columns)
Optimal Data Set = ReprsentDataSet (Optimal_Columns, Optimal_Rows )
Output: Optimal Data Set
```

3.4. Step 4: Classification using SVM

This paper uses Linear SVM (LSVM) and two non-linear SVM, i.e.: Polynomial SVM (PSVM) and Sigmoid SVM (SSVM) as the classifier for the proposed IoT IDS. The kernel function for the non-linear SVM is expressed as in (1) [39,40].

$$K(x, x_i) = sum(x * x_i) \tag{1}$$

The Polynomial kernel function with degree *d* is represented in (2). Degree *d* has a default value of 2.

$$K(x, x_i) = 1 + sum(x * x_i)^{\,d} \tag{2}$$

The Sigmoid kernel function is represented in (3).

$$K(x, x_i) = tanh(\alpha x_i \bullet x_j + \beta) \tag{3}$$

The classifiers are implemented as Algorithm 4.

Algorithm 4. SVM
```
Input: Optimal_Data_Set
Processing:
Kernal_Type = SetKernelType ()
SVM_Classifier = GetSVMClassifier (Kernal_Type)
Classified_Data_Set = SVM_Classifier.Train (Optimal_Data_Set)
Output: Classified_Data_Set
```

3.5. Performance Metrics

The confusion matrix in machine learning and artificial intelligence refers to the various scenarios that can arise with a particular classifier when considering the classifier's output and the actual class values. This makes it a useful framework for analyzing the degree to which a classifier can distinguish between records belonging to distinct classes [41]. The confusion matrix is established using four terms: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Five metrics are used for evaluating the performance of the proposed IDS as represented by (4) – (8) [42].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F1\_Score = \frac{2TP}{2TP + FP + FN} \tag{7}$$

$$Error\_rate = 1 - Accuracy \tag{8}$$

4 Results, Analysis and Discussions

The proposed IDS is implemented using Python 3.10.1 programming language. It is implemented on a Lenovo Laptop equipped with hexa-core Intel(R) Core (TM) i7-9750HF CPU whose Base Frequency is 2.60 GHz. The RAM capacity is 16 GB, and the hard drive capacity is 1TB. The operating system is Microsoft Windows 11 Home (x64).

4.1. Optimal Features Results

On NSL-KDD dataset, 21 features are selected from the MSO algorithm implementation, i.e.: *duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, logged_in, count, serror_rate, same_srv_rate, diff_srv_rate, dst_host_diff_serv_rate, dst_host_same_src_port_rate, dst_host_diff_src_port_rate, dst_host_serror_rate, dst_srv_host_serror_rate* and *dst_host_rerror_rate*. While on UNSW-NB15 dataset, 7 features are selected, i.e.: *IPv4_src_addr, L4_src_port, IPv4_dst_addr, L4_dst_port, protocol, TCP_flag, attack.*

Table 2 shows the selected optimal dataset as the result of features selection using MSO method and records reduction using Weighting Prototype method for NSL-KDD and UNSW-NB15 datasets.

Table 2: Optimal Datasets Result

	NSL-KDD		UNSW-NB15	
	Original	Result	Original	Result
Features #	42	21	14	7
Records #	125,972	73,234	2,000,000	1,351,287

On UNSW-NB15 dataset, 980,078 records are identified as normal records, 68,497 records are attack records, while 302,712 records are undefined. Only two classes of data are considered, i.e.: normal and attack classes. Thus, the total records used for the experiment on UNSW-NB15 are 1,048,575 records.

4.2. Classification Results

Observations on the values of TP, FP, TN, and FN during the experiment are tabulated in Table 3.

Table 3 :Values of Confusion matrix

	LSVM		PSVM		SSVM	
	NSL-KDD	UNSW-NB15	NSL-KDD	UNSW-NB15	NSL-KDD	UNSW-NB15
TP	64312	950676	61418	922156	58654	894491
TN	55992	66442	53472	64449	51066	62516
FP	3030	29402	5924	57922	8688	85587
FN	2638	2055	5158	4048	7564	5981

Fig. 6 displays the detection performance measurements of the proposed IoT IDS on detecting traffic attacks on NSL-KDD dataset. The result graph containing accuracy, precision, recall, and F1-score that provide valuable insights into the performance of a classification model of the IoT IDS.

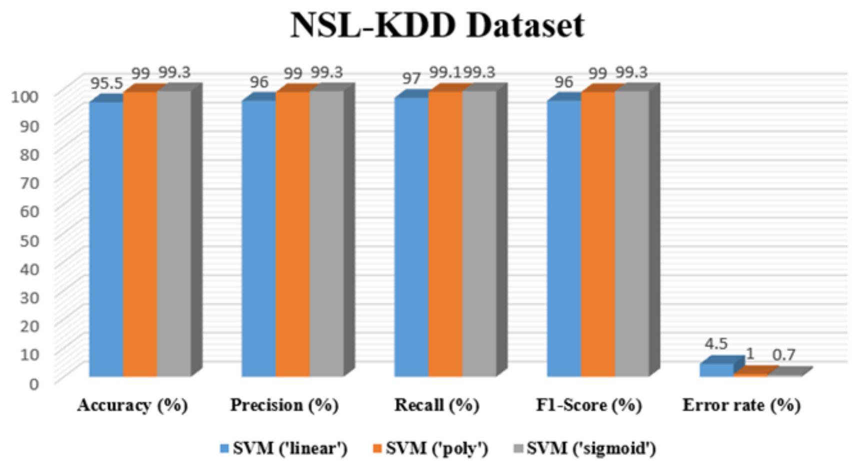

Fig. 6. Performance evaluation results on NSL-KDD dataset

The results of accuracy shown in Fig. 6 give a general idea of how many predictions the model got right. The precision and recall do not deviate significantly, since the NSL-KDD dataset is balance dataset. The figure also shows a low error rate, which means the proposed IoT IDS is very good at making accurate predictions. In fact, the SSVM version of the IoT IDS only had a 0.7% error rate, which shows it is highly effective at classifying data correctly. This low error rate suggests the system has learned the right patterns from the data and can be generalized even to entirely new data it has not seen before. Even better, the error rate is significantly lower than LVSM and PSVM, ndicating a major improvement in performance.

Fig. 7 shows the detection performance measurements of the proposed IoT IDS on detecting traffic attacks on UNSW-NB15 dataset. In general, the proposed IoT IDS is better than the performance on the SDL-KDD dataset. Even the UNSW-NB15 dataset is imbalanced dataset, the precision and recall curves does not deviate significantly, indicating the three classifier models succeed dealing with the imbalanced data. In addition, F1-scores are also very good, providing a more balanced view of the model's performance for imbalanced datasets.

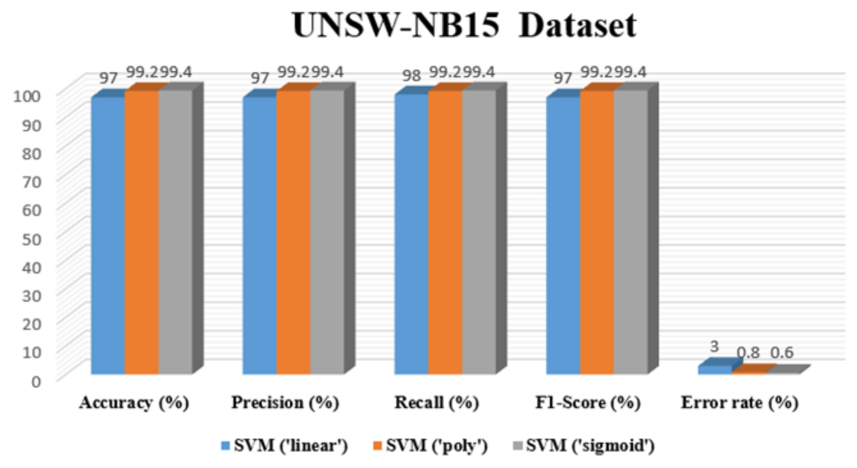

Fig. 7. Performance evaluation results on UNSW-NB15 dataset

Table 4 displays the comparison of the accuracy detection with other works.

Table 4. accuracy comparison

Ref.#	Method	Dataset	Accuracy
[19]	Machine learning	KDD Cup’99	99.04%
[20]	(ML)	IoTID20	95.8%
[13]		Created from real traffic	99.94%
[14]	ML+ FS	MedBIoT	99.72%
[21]		Bot-IoT	99.99%
[22]		TON_IoT_Weather	99.99%
[23]		CICIDS2017	99.99%
[24]	Deep learning + FS	**NSL-KDD,**	**99.2%**
		UNSW-NB15	**99.3%**
[25]		NSL-KDD, IoTID20	99.98%
[26]	Ensemble	UNSW-NB15, NIMS	95.25%
[27]		NSL-KDD, UNSW-NB15	91.9%
[17]		CICIDS2017	99.9%
[28]	Ensemble + FS	CICIDS2017	99.99%
		NSL-KDD	**99.3%**
[**xx**]*		**UNSW-NB15**	**99.4%**

*the work in this paper.

4.3. Runtime Results

The runtime of machine learning models on specific dataset can be a deciding factor on the choice of algorithms, especially for benchmarking and comparison purposes. Table 5 shows the runtime for executing the proposed classifier, which compared with the classifier in [24].

Table 5: Runtime for executing the classifier

Ref.#	Classifier Method	Dataset	Runtime			Detection time (ave.)
			Preprocessing (seconds)	Training (minutes)	Testing (minutes)	
[24]	SPDNN+SMO	NSL-KDD,	27	35	13	3.8 ms
		UNSW-NB15	56	78	50	2.7 ms
[xx]*	SSVM+SMO	NSL-KDD	39	30	9	2.9 ms
		UNSW-NB15	90	57	41	1.2 ms

*the work in this paper.

4.4. Discussion

The SMO algorithm as feature selection method and the Weight Prototype algorithm as instance reduction method provide very good dimensional reduction of the datasets, while preserving the accuracy detection rate. The SMO algorithm decreases the features number up to 50%, while the Weight Prototype algorithm reduces the instance dimension with 41.86% and 32.44% for NSL-KDD and UNSW-NB15 datasets, respectively (refer to Table 2). Table 3 shows the values of confusion matrix for the three types of SVM classifier obtained from observation during the experiments. Then using equations (4) – (8) the performance metrics are computed. The best accuracy is achieved by Sigmoid SVM, i.e.: 99.3% on NSL-KDD dataset and 99.4% on UNSW-NB15 dataset, as confirmed by Fig. 5 and Fig. 6.

From Table 4, it can be seen, the IDSs that perform better than the proposed IDS are implemented on different datasets. The IDSs that run on NSL-KDD dataset, i.e. [24], [26] and [27] perform under the proposed IDS. To have apple-to-apple comparison, the authors re-implement the classifier of Otoum, et al. [24] on UNSW-NB15 datasets. The proposed IDS performs slightly better. The implementation of the proposed IDS on both datasets results in a good accuracy, which represents the robustness of the proposed IDS as NSL-KDD dataset is balanced dataset with more features, while UNSW-NB15 dataset contains imbalanced records. In order word, the proposed IDS works well also on imbalanced dataset.

Table 5 compares the proposed IDS using SSVM classifier with Otoum's IDS on temporal dimension, where the pre-processing time, training time, and testing time are involved. It is obvious that the proposed IDS underperforms in terms of pre-processing time due to the proposed IDS deploys two algorithms to reduce the dataset dimensional, while the Otoum's IDS only deploys SMO. On the other hand, the proposed IDS outperforms the Otoum's IDS in both cases (i.e., training on NSL-KDD dataset and UNSW-NB15 dataset) and at level of training time and testing time. The reason behind this is related to the effective pre-processing step used in the proposed IDS, which leads to decrease the dataset dimension while avoiding cures of dimensionality issue. The proposed IDS is speeded up by 23 minutes and 9 minutes for training and testing times respectively, when compared to the Otoum's IDS running on the UNSW-NB15 dataset. Detection time on UNSW-NB15 dataset are better than NSL-KDD dataset, due to its data dimensional is lower.

5 Conclusion

The study has developed effective IoT IDS based on a hybrid feature selection and supervised machine learning. The proposed IDS has been implemented on balanced and imbalanced datasets, and it performs very well, thus, the proposed IDS is considered robust. To continue this work, we intend to make the UNSW-15 dataset as balance dataset (i.e., the attacks data and normal data ratios are close). Moreover, we plan to study CICDDoS2019 Dataset and IoTID20 dataset by deep learning models using the Wrapper method.

References

[1] Stoyanova, M., Nikoloudakis, Y., Panagiotakis, S., Pallis, E., & Markakis, E. K. (2020). A survey on the internet of things (IoT) forensics: challenges, approaches, and open issues. *IEEE Communications Surveys & Tutorials*, 22(2), 1191-1221.

[2] Sarker, I. H., Khan, A. I., Abushark, Y. B., & Alsolami, F. (2022). Internet of things (iot) security intelligence: a comprehensive overview, machine learning solutions and research directions. *Mobile Networks and Applications*, 1-17.

[3] Zeadally, S., & Tsikerdekis, M. (2020). Securing Internet of Things (IoT) with machine learning. *International Journal of Communication Systems*, 33(1), e4169.

[4] Mahdavinejad, M. S., Rezvan, M., Barekatain, M., Adibi, P., Barnaghi, P., & Sheth, A. P. (2018). Machine learning for Internet of Things data analysis: A survey. *Digital Communications and Networks*, 4(3), 161-175.

[5] Berman, D. S., Buczak, A. L., Chavis, J. S., & Corbett, C. L. (2019). A survey of deep learning methods for cyber security. *Information*, 10(4), 122-.

[6] Karim, S. M., Habbal, A., Chaudhry, S. A. and Irshad, A. (2023). BSDCE-IoV: Blockchain-based secure data collection and exchange scheme for IoV in 5G environment. *IEEE Access*,11, 36158-36175.

[7] Askar, N.A., Habbal, A., Mohammed, A. H., Sajat, M. S., Yusupov, Z. and Kodirov, D. (2022). Architecture, protocols, and applications of the internet of medical things (IoMT). *Journal of Communications*, 17(11), 900-918.

[8] Akhand, M. A. H., Ayon, S. I., Shahriyar, S. A., Siddique, N., & Adeli, H. (2020). Discrete spider monkey optimization for travelling salesman problem. *Applied Soft Computing*, 86, 105887.

[9] Mohammadi, M. Rashid, T. A., Karim, S. H. T., Aldalwie, A. H. M., Tho, Q. T., Bidaki, M., Rahmani, A. M., Hosseinzadeh, M. (2021). A comprehensive survey and taxonomy of the SVM-based intrusion detection systems, *Journal of Network and Computer Applications*, 178, 102983.

[10] Susanto, Stiawan, D., Arifin, M.A.S, Idris, M.Y. and Budiarto, R. (2021). Impact of dimensionality reduction on performance of IoT intrusion detection system, in *Internet of Things: Research and Practical Insights book*, Bharat Bhushan, Sudhir Kumar Sharma, Bhuvan Unhelkar, Muhammad Fazal Ijaz, Lamia Karim (Eds.), CRC Press, Taylor & Francis Group. Ch. 6, 101-124.

[11] Susanto, Stiawan, D. Rini, D.P., Arifin, M.A.S., Idris, M.Y., Alsharif, N. and Budiarto, R. (2022). Dimensionality Reduction with Fast ICA for IoT Botnet Detection. *Journal of Applied Security Research*, Aug. 2022.

[12] Karim, S. M., Habbal, A., Hamouda, H., Alaidaros, H. (2023). A secure multifactor-based clustering scheme for internet of vehicles, *Journal of King Saud University - Computer and Information Sciences*, 35(10), 2023, 101867.

[13] Stiawan, D., Suryani, M.E., Susanto, Idris, M.Y., Aldalaeien, M., Alsharif, and Budiarto, R. (2021). Ping Flood Attack Pattern Recognition Using a K-Means Algorithm in an Internet of Things (IoT) Network, *IEEE Access*, Augt. 2021, 9, 116475-116484.

[14] Stiawan, D. Susanto, Bimantara, A. Idris, M.Y. and Budiarto, R. (2023). IoT botnet attack detection using deep autoencoder and artificial neural network, *KSII Transactions on Internet and Information Systems*, 17(5), May 2023, 1310-1338.

[15] Sharipuddin, Winanto, E.A., Purnama, B. Kurniabudi, Stiawan, D., Hanapi, D., Idris, M.Y., Kerim, B., Budiarto, R. (2021). Enhanced Deep Learning Intrusion Detection in IoT Heterogeneous Network with Feature Extraction, *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, 9(3), 747-757.

[16] Stiawan, D., Heryanto, A., Berdadi, A., Rini, D.P., Subroto, Kurniabudi, Idris, M.Y., Abdullah, A.H., Kerim, B., Budiarto, R. (2021). An approach for optimizing ensemble intrusion detection systems, *IEEE Access*, 9, 6930-6947.

[17] Kurniabudi, Stiawan, D. Darmawijoyo, Idris, M.Y., Defit, S., Triana, Y.S., Budiarto, R. (2022). Improvement of attack detection performance on the internet of things with pso-search and random forest, *Journal of Computational Science*, 64, 101833.

[18] Abbas, A., Khan, M.A., Latif, S. et al. (2022), A new ensemble-based intrusion detection system for internet of things. *Arab Journal of Science and Engineering*, 47, 1805–1819.

[19] Dushimimana, A., Tao, T., Kindong, R., & Nishyirimbere, A. (2020). Bi-directional recurrent neural network for intrusion detection system (IDS) in the internet of things (IoT). *International Journal of Advanced Engineering Research and Science*, 7, 524-539.

[20] Hussein, A. Y., Falcarin, P., & Sadiq, A. T. (2021). Enhancement performance of random forest algorithm via one hot encoding for IoT IDS. *Periodicals of Engineering and Natural Sciences,* 9(3), 579-591.

[21] Idrissi, I., Azizi, M., & Moussaoui, O. (2021). Accelerating the update of a DL-based IDS for IoT using deep transfer learning. *Indonesian Journal of Electrical Engineering and Computer Science,* 23(2), 1059-1067.

[22] Alabadi, M., Habbal, A. and Guizani, M.(2024). An Innovative Decentralized and Distributed Deep Learning Framework for Predictive Maintenance in the Industrial Internet of Things. *IEEE Internet of Things Journal*, doi: 10.1109/JIOT.2024.3372375..

[23] Maseer, Z. K., Yusof, R., Mostafa, S. A., Bahaman, N., Musa, O., & Al-rimy, B. A. S. (2021). DeepIoT. IDS: hybrid deep learning for enhancing IoT network intrusion detection. *Computers, Materials and Continua*, 69(3), 3945-3966.

[21] Otoum, Y., Liu, D., & Nayak, A. (2022). DL-IDS: a deep learning–based intrusion detection framework for securing IoT. *Transactions on Emerging Telecommunications Technologies,* 33(3), e3803.

[25] Albulayhi, K., Abu Al-Haija, Q., Alsuhibany, S. A., Jillepalli, A. A., Ashrafuzzaman, M., & Sheldon, F. T. (2022). IoT intrusion detection using machine learning with a novel high performing feature selection method. *Applied Sciences,* 12(10), 5015.

[26] Moustafa, N., Turnbull, B., & Choo, K. K. R. (2018). An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things. *IEEE Internet of Things Journal,* 6(3), 4815-4830.

[27] Alhowaide, A., Alsmadi, I., & Tang, J. (2021). Ensemble detection model for IoT IDS. *Internet of Things,* 16, 100435.

[28] Jaw, E., & Wang, X. (2021). Feature selection and ensemble-based intrusion detection system: an efficient and comprehensive approach. *Symmetry*, 13(10), 1764.

[29] Almaiah, M. A., Almomani, O., Alsaaidah, A., Al-Otaibi, S., Bani-Hani, N., Hwaitat, A. K. A., et al. (2022). Performance investigation of principal component analysis for intrusion detection system using different support vector machine kernels. *Electronics*, 11(21), 3571.

[30] NSL-KDD website, online [Available]: https://github.com/Mamcose/NSL-KDD-Network-Intrusion-Detection/blob/master/NSL_KDD_Train.csv

[31] Sharma, S., Gigras, Y., Chhikara, R., & Dhull, A. (2019). Analysis of NSL KDD dataset using classification algorithms for intrusion detection system. *Recent Patents on Engineering*, 13(2), 142-147.

[32] UNSW_NB15 website, online [Available]: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15

[33] Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).* 9(1), 381-386.

[34] Kasongo, S. M., & Sun, Y. (2020). Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. *Journal of Big Data*, 7, 1-20.

[35] Umar, M. A., & Zhanfang, C. (2023). Effects of Feature selection and normalization on network intrusion detection. *Authorea Preprints*.

[36] Miloudi, S., Wang, Y., & Ding, W. (2021). An improved similarity-based clustering algorithm for multidatabase mining. *Entropy*, 23(5), 553.

[37] Abdulhammed, R., Musafer, H., Alessa, A., Faezipour, M., & Abuzneid, A. (2019). Features dimensionality reduction approaches for machine learning based network intrusion detection. *Electronics*, 8(3), 322.

[38] Jiangzhong Cao, Zijie Yao, Lianggeng Yu, Bingo Wing-Kuen Ling, (2023), WPE: Weighted prototype estimation for few-shot learning, *Image and Vision Computing*, 137, 2023, 104757.

[39] Shankar, K., Lakshmanaprabu, S. K., Gupta, D., Maseleno, A., & De Albuquerque, V. H. C. (2020). Optimal feature-based multi-kernel SVM approach for thyroid disease classification. *The journal of supercomputing*, 76, 1128-1143.

[40] Roy, A., & Chakraborty, S. (2023). Support vector machine in structural reliability analysis: A review. *Reliability Engineering & System Safety*, 109126.

[41] Zoghi, Z., & Serpen, G. (2021). Unsw-nb15 computer security dataset: Analysis through visualization. *arXiv preprint arXiv:2101.05067*.

[42] Tuan, T.A., Long, H.V., Son, L.H. et al. (2020). Performance evaluation of Botnet DDoS attack detection using machine learning. *Evolutionary Intelligence*, 13, 283–294 (2020).

Notes on contributors

Mohammed A. Shenify is an Associate Professor at the Department of Computer Science, Albaha University, Saudi Arabia. His main teaching and research interests include Natural Language Processing, Artificial Intelligence and Machine Learning. He has published several research articles in international journals of computer sciences and information technology.

Amjad S. Alghamdi is currently a student at Master in Computer Science program of Department of Computer Science, Albaha University, Saudi Arabia. Her research interests include Cybersecurity, IoT networks/systems, Machine Learning and Artificial Intelligence.

Adil F. Alharthi is an Associate Professor at the Department of Computer Science, Albaha University, Saudi Arabia. His main teaching and research interests include Intrusion Detection System, Network Traffic Analysis and Machine Learning. He has published several research articles in international journals of computer sciences and information technology.