Journal of Computer and Communications, 2022, 10, 59-73
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Task-Specific Feature Selection and Detection Algorithms for IoT-Based Networks

Yang Gyun Kim[1], Benito Mendoza[1], Ohbong Kwon[1], John Yoon[2]

[1]Department of Computer Engineering Technology, New York City College of Technology (CUNY), New York, USA
[2]Department of Math/Computer Science/Cybersecurity, Mercy College, Dobbs Ferry, USA
Email: yakim@citytech.cuny.edu, bmendoza@citytech.cuny.edu, okwon@citytech.cuny.edu, jyoon@mercy.edu

How to cite this paper: Kim, Y.G., Mendoza, B., Kwon, O. and Yoon, J. (2022) Task-Specific Feature Selection and Detection Algorithms for IoT-Based Networks. *Journal of Computer and Communications*, **10**, 59-73.
https://doi.org/10.4236/jcc.2022.1010005

Received: September 11, 2022
Accepted: October 24, 2022
Published: October 27, 2022

Abstract

As IoT devices become more ubiquitous, the security of IoT-based networks becomes paramount. Machine Learning-based cybersecurity enables autonomous threat detection and prevention. However, one of the challenges of applying Machine Learning-based cybersecurity in IoT devices is feature selection as most IoT devices are resource-constrained. This paper studies two feature selection algorithms: Information Gain and PSO-based, to select a minimum number of attack features, and Decision Tree and SVM are utilized for performance comparison. The consistent use of the same metrics in feature selection and detection algorithms substantially enhances the classification accuracy compared to the non-consistent use in feature selection by Information Gain (entropy) and Tree detection algorithm by classification. Furthermore, the Tree with consistent feature selection is comparable to the ensemble that provides excellent performance at the cost of computation complexity.

Keywords

Cybersecurity, Features Selection, Information Gain, Particle Swarm Optimization, Intrusion Detection System, Machine Learning, Decision Tree, Network Attacks, IoT Network

1. Introduction

We live in a world where we are surrounded and embedded in a vast network of gadgets, machines, objects, and people. The Internet of Things (IoT) allows these physical objects ("things") and people to be connected to the internet and each other. Each "thing" (your thermostat or smart fridge, the elevator on your building, or some machines in a factory) is equipped with sensors and actuators that allow it to collect and exchange data. The IoT is evolving swiftly; it is becoming a sophisticated, more intelligent, sensitive, and responsive ecosystem. Soon, the "things" will be able to think, feel and act as we humans do. They will be able to learn from us and each other. They will be able to adapt to our needs and preferences. They will be able to interact with us in ways that are natural and intuitive [1] [2] [3].

However, security concerns are worth considering when implementing an IoT solution. First, IoT devices are often connected to the internet, which means they are vulnerable to attacks from cybercriminals. IoT devices may collect and store sensitive data, which unauthorized individuals could access and use if the devices are not adequately secured. Additionally, IoT devices may be used to control physical systems, such as industrial equipment or vehicles, which could pose a safety risk if the devices are hacked or malfunctioned [4] [5]. To help ensure the security of an IoT solution, it is essential to choose devices that have been designed with security in mind.

In cybersecurity, artificial intelligence has been used to create detection and predictive models of cyberattacks, creating systems that can automatically identify and block malicious activity. Especially machine learning (ML) approaches have proved helpful in cybersecurity because they can learn from data and identify patterns that humans may not be able to see [6] [7].

An intrusion Detection System (IDS) is the primary form of security for many organizations. IDS can be used to detect attempts to exploit vulnerabilities in software or hardware and, thus, detect and avoid a wide variety of attacks, such as denial of service attacks, viruses, worms, and buffer overflows [8]. IDS is essential to a comprehensive security strategy. However, building and maintaining efficient IDS is a demanding task that requires constant updates. Most IDS are data-driven models, and the amount of data produced by an IoT system can be overwhelming.

One of the challenges in applying MLto IDS is to select the right set of features to feed into an ML model. An MLalgorithm can capture unimportant patterns and learn from noise if too many unnecessary features exist; the selected attack's features must be of the highest quality. The method of reducing and selecting the input attack features to a model by using relevant data and getting rid of noise in data is called Feature Selection. Feature selection (FS) establishes a subset of relevant features that can deselect irrelevant and redundant features while increasing the discerning ability of the feature set [7].

FS is an optimization problem that can be categorized into two parts: search strategy and evaluation strategy in a supervised model. The two parts are further classified into optimal and heuristic for the search and filter and wrapper methods for the evaluation. The difference between the filter and the wrapper methods is that the filter is independent of a classification (*i.e.*, detection) algorithm. In contrast, the wrapper is related to a classification algorithm, where the FS utilizes some of the classification's metrics to select a subset of features. Our method is based on the hybrid of the wrapper and the meta-heuristic algorithm rather than the optimal algorithm, *i.e.*, an exhaustive search compensates for the high computation cost.

To the best of our knowledge, this is the first attempt to address the consistent metrics of feature selection and detection algorithms to maximize classification accuracy. In [9], a method to select the most relevant features using PSO is presented. The approach relies on correlation-based metrics and performs the fusion of Tree-based Classifiers (ensemble) with the NSL-KDD dataset. In general, ensemble models are known to perform better than a single model [9]. However, training several models and collecting and averaging the output of multiple models is computationally expensive, so an ensemble may be unsuitable for resource constraint devices. [10] utilizes PSO for feature selection by selecting the most relevant feature subsets after optimizing the maximum class variance within normal and anomaly (*i.e.*, distance), using the NSL-KDD dataset, and then applying PSO + Tree and PSO + KNN (K-Nearest Neighbor) for binary classification. The result shows that PSO + KNN outperformed the PSO + Tree classifier algorithm in identifying network anomalies. Again, the ensemble classifier strategy also outperforms single machine learning techniques. [11] proposes a combination of filter and wrapper approaches to achieve smaller feature subsets with better classification performance in a shorter time. [11] and [12] select relevant feature subsets close to the class label using Information Gain. In addition, [11] employs the local search heuristic as a first stage (*i.e.*, filter) to find a better personal best using IG and then applies PSO, in which the fitness (*i.e.*, wrapper) performs by keeping a distance with different class and keeping close the same class. However, [11] is still based on distance, although the detection algorithm is a classification for the disease names. Therefore, all related works above are considered as non-consistent algorithms between feature selection and detection. This paper analyzed multi-Gigabyte data generated from IoT devices [13] and selected relevant features to push the IoT device's intrusion detection rate up to a maximum possible point where it can be reached by choosing the input attack features that are directly associated with the detection algorithms.

2. Feature Selection

2.1. The Description of the IoT-23 Dataset

One of the significant research challenges in IoT-based networks is the lack of a comprehensive network-based dataset that reflects current network traffic scenarios, a wide variety of low-footprint intrusions, and in-depth structured information about the network traffic [14]. Nonetheless, the IoT-23 dataset provides data on an IoT-based network with 23 scenarios involving Windows and Linux OSs in diverse IoT-related attack types. Many network scenarios published in 2020 and collected for one year are awaiting analysis. The IoT-23 data were collected for at least one hour and up to 112 hours, depending on how quickly the pcap file size increased. The first three scenarios involve only normal data for three IoT devices: Amazon Echo device, Philip Hue device, and Somfy door lock device. The remaining 20 scenarios include normal and attack data along with different periods and attack types while naming each scenario a specific attack type: Mirai for scenarios 1 through 5, 14, and 15, Torri for scenarios 6 and 7, Trojan for scenario 8, Gagfyt for scenario 9, Kenjiro for scenarios 10 and 12, Okiru for scenario 11, Hakaj for scenario 13, IRCBot for scenario 16, Linux Mirai for scenario 17, Linux Hajime for scenario 18, Mushstik for scenario 19, and Hide and Seek for scenario 20. The three devices are real, so the scenarios are not simulated or emulated, and the dataset is generated from a real environment. Although there are only three devices, the diverse attack types (IoT botnet, IoT malware, and data breaches) reflect a modern updated footprint attack environment for an IoT-based network. This paper investigated some scenarios where feature selection by both IG and PSO was evaluated. Based on the feature selection, Decision Tree (Tree in short), as a classifier, was investigated for performance comparison with the minimum select feature set by each feature algorithm.

2.2. The Data-Processing of the IoT-23 Dataset

Among the 23 features and labeled data in the IoT-23 dataset, the 21st feature is ignored due to infrequent data elements. The 22nd feature is for binary classification, which is also dismissed as our multi-class classification goal. Furthermore, the first and second features are discarded due to collection purpose attributes. The first feature is Unix time for starting and ending each flow collection, and the second is for universal ID with string. Therefore, there are a total of 18 features and labeled data. Data preparation is performed by converting data form (e.g., float, string, Boolean, etc.) into a machine-readable format using label encoding. The data is then standardized for all features, although Tree is insensitive to the scaling in data magnitude [12]. This paper uses label encoding rather than a one-shot, in which one single part is converted into machine-readable 126 inputs (features) (e.g., a "history" feature has 126 elements), thereby increasing the number of features/inputs and further increasing the complexity. The label encoding converts each feature data into a numeric form in a machine-readable format.

This paper analyzed some of the scenarios in the IoT-23 dataset using Matlab due to the ease of data preparation. When importing the data into Matlab, this paper had to use the "table" data type to import the 16th and the 2nd scenarios because of the extreme volume of data. However, this paper could import all other systems as "string" data types, which does not require tedious data preparation when transforming the data into machine-readable form. Although the "table" data type requires the least memory space compared to other data types, such as "string", "category", and "numeric", it requires time-consuming data preparation. With our goal of multi-class classification, the labeled class for each scenario would be parsed appropriately since each scenario has varying attack types from the least two to the maximum of seven classes. The detailed procedure of the data-preprocessing pipeline and the whole flow of the proposed framework is shown in Figure 1.

2.3. Information Gain-Based Feature Selection

Two feature selection algorithms were investigated: IG and PSO. As the Tree algorithm is built along with the entropies of features, *i.e.*, IG could contribute to boosting the detection rate accuracy for the Tree algorithm. Surprisingly, building the Tree structure by using the ranks of the entropies in features is irrelevant to the performance of the Tree due to the different metrics; entropies form the tree topology, and the metric for detection by the Tree is classification. The intrinsic aim of PSO is to improve computational efficiency by finding sub-optimal value/s of the feature space [11]. The metric of the PSO this paper proposed uses misclassification to be consistent between feature selection and detection algorithm. Thus, PSO would be favorable to Tree detection as PSO performs feature selection on detection algorithm metrics (*i.e.*, misclassification). Our primary concern is the consistent performance of the feature selection method and detection algorithm to maximize the overall detection performance while keeping the computation complex low.

The paper calculates IG in some scenarios with the number of data records and attack types. Each scenario has a different number of data records and other attack types that produce different IG values. For example, although there is the same number of data records and features, the IG for binary and multiclass classifications should be different due to the $freq(C_j)$ in Equation (1). The IG depends on the number of attributes in the labeled class, in which each attribute in the label is calculated as its probability of occurrence in Equation (2) [15]. Furthermore, the IG is also influenced by the number of features and data records. Based on the IG, those features in the top 10% to 40% of the IG values are selected

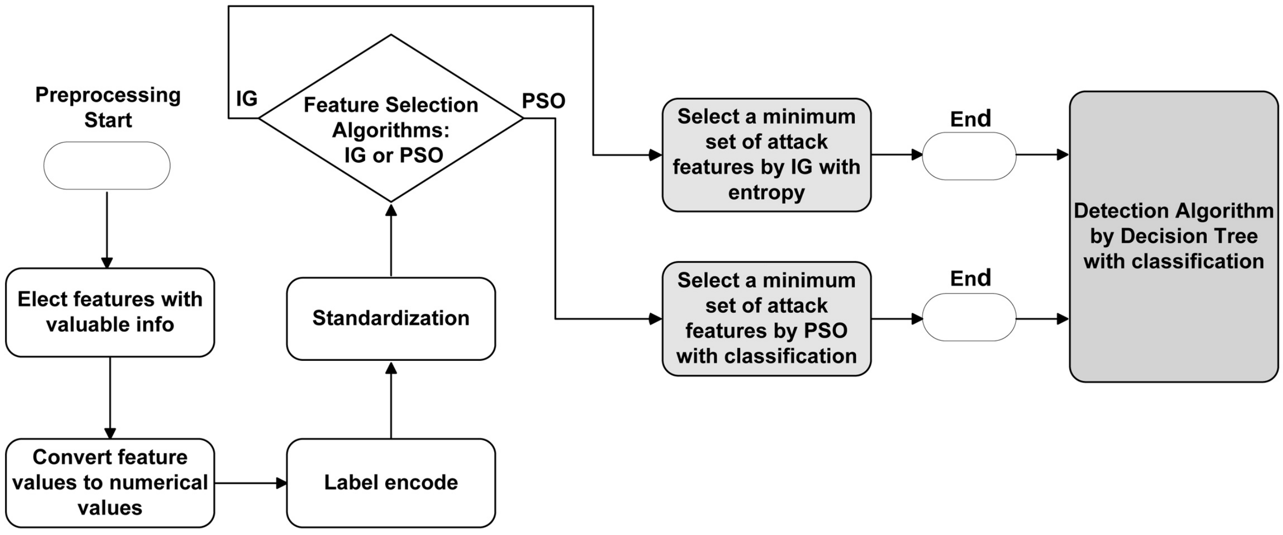

Preprocessing Start
Elect features with valuable info
Convert feature values to numerical values
Label encode
Standardization
Feature Selection Algorithms: IG or PSO
IG
PSO
Select a minimum set of attack features by IG with entropy
Select a minimum set of attack features by PSO with classification
End
End
Detection Algorithm by Decision Tree with classification

Figure 1. The proposed architecture for a task-specific intrusion detection system.

above the line [16] for each scenario. D is the number of records, and T is one of the attributes (one of the labeled classes). The IG this paper uses is defined as follows.

$$Info(D) = -\sum_{j=1}^{m} \frac{freq(C_j, D)}{|D|} \log_2 \left\{ \frac{freq(C_j, D)}{|D|} \right\} \tag{1}$$

$$Info(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} info(T_i) \tag{2}$$

$$\textbf{\textit{Information Gain}}(A_i) = Info(D) - Info(T) \tag{3}$$

2.4. PSO-Based Feature Selection

Biology-inspired methods have a tremendous impact on designing network security systems. They have developed novel and effective protection schemes due to the increased deployment and widespread use of computer systems. As traditional approaches often suffer from scalability problems to cope [9]. Thus, it is essential to consider biologically based systems as sources of inspiration when designing new processes. Therefore, Particle Swarm Optimization (PSO) has been widely applied in an ML-IDS as one of many existing biology-inspired meta-heuristic algorithms. For PSO, each particle, a potential solution (x_a^i), moves around toward a global target in a search space ($x_{ad}^i \in \Re^2$, where d is dimension) while sharing its own experience (*i.e.*, a personal best since i iteration, P_a^i) with that of other particles. Each particle will compromise his future movement by acknowledging others' movement behaviors. Among the others' movements, the best movement (*i.e.*, a global best since i iteration, p_g^i) toward a global target will be selected, which will be the leader, and others will adjust their movements accordingly. The amount of the movement for each particle is determined by their current position and velocity as following Equations (4) and (5) [17].

$$v_{ad}^i = w^i * v_{ad}^{i-1} + c_1^i * r_1 * \left(p_{ad}^{i-1} - x_{ad}^{i-1}\right) + c_2^i * r_2 * \left(p_{gd}^{i-1} - x_{ad}^{i-1}\right) \tag{4}$$

$$x_{ad}^i = x_{ad}^{i-1} + v_{ad}^i \tag{5}$$

In Equation (4), w^i is inertia weight that constitutes the global search ability (exploration) and the local search ability (exploitation). The "cognitive" coefficient c_1^i and the "social" coefficient c_2^i represent how fast each particle moves towards p_{ad}^{i+1} and p_{gd}^{i+1} positions. r_1 and r_2 are random numbers uniformly distributed within [0,1) and a more detailed explanation refers to [17]. The pseudocode for the PSO-based feature selection algorithm is as follows in **Figure 2**.

3. Performance Evaluation

Tree detection algorithms are performed with a minimal set of attack features after the feature selection by IG and PSO-based. During the PSO-based feature selection, particle population is 20, and the other hyper parameter values of c_1^i, c_2^i, *w*, etc. refer to [17]. The experiment scenario or setting and conditions for

```
Input: The number of attack features
Output: The best set of attack features
1 Randomly initialize particles
2 While do until one of the stopping criteriaismet
3     For $i$ = 1 to Population Size do
4         If $F_i$ is better than pbest, then update pbest
5             if $F_i$ is better than gbest, then update gbest
6             End
7         End
8         Update velocity of particle $i$ by Equation (4)
9         Update position of particle $i$ by Equation (5)
10    End
11    Return the position of gbest
12 End

  Stopping condition: 100 epochs, zeros misclassification
                      or gradient $10^{-6}$

  $F_i$ = w*misclassification of Train Data + (1−w)*misclassification
   Test Data, where, w is 0.8
```

Figure 2. Pseudo-code for PSO-based feature selection.

detection are as follows. The Tree algorithm uses the hyperparameter-tuned Tree due to the varying number of data records and classes. The values of the hyperparameters tend to grow deeply as the number of data sample records increases. In addition, as the number of data records for each scenario significantly varies, the optimized Tree algorithm is performed for detection accordingly. Support Vector Machine (SVM) is used as a Linear kernel with the cross-entropy loss function whether it can segment the separability of the dataset linearly. Ensemble is used Bagged Tree where the number of learners is 30 and maximum number of splits vary due to different number of data records. The data for detection is divided into training, verification, and test data. This paper used ten cross-validation folds where the validation check is completed after each epoch (iteration) to determine whether the model is appropriate, avoiding underfitting or overfitting. Select scenarios in the IoT-23 dataset are analyzed via IG and PSO algorithms for feature selection, and Tree for detection is considered for in-depth analysis. The simulation work was carried out on a machine with 128 GB of RAM, a 7-core CPU and an 8 GB GPU.

The 19th scenario (3-1) has 156,103 data records with four classes: Attack, Benign, C&C and Part OfAHorizontalPortScan, while the pcap size is 56 MB [13]. The most important feature by IG is "conn_state" (10), shown in Figure 3. The selection of features by IG has been done with the above line (threshold) for features 8, 9, 10, and 14, as shown in Figure 3. In Figure 4, the number 1 to 18 indicates feature numbers and the number for each feature refers to particle's position, optimal of which is the minimum 0 (origin but not x and y); the smallest number shows feature 14 and the next is feature 10, and the algorithm stops for the gradient stopping condition (10^{-6}). Therefore, the most important feature by PSO-based is "history" (14) and feature selection has been selected 10 and 14, shown in Figure 4.

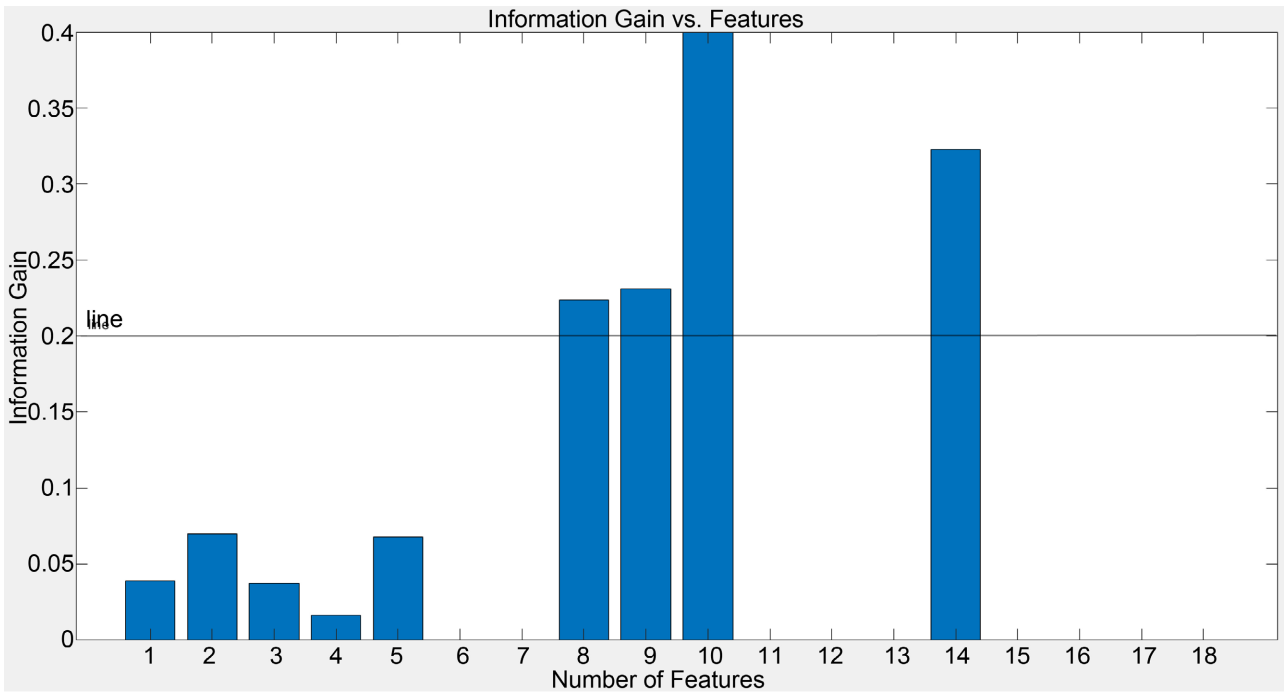
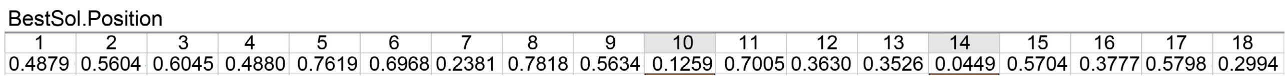


Figure 3. Feature selection by IG for scenario 19th.


BestSol.Position

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0.4879	0.5604	0.6045	0.4880	0.7619	0.6968	0.2381	0.7818	0.5634	0.1259	0.7005	0.3630	0.3526	0.0449	0.5704	0.3777	0.5798	0.2994

Figure 4. PSO-based feature selection for scenario 19th.

For a fair comparison between the IG and PSO-based, Support Vector Machine (SVM) with multi-classification has been performed and the left-side shows 13 misclassifications, as shown in **Figure 5**, while the right-side exhibits 8 misclassifications, as shown in **Figure 6**. The different feature selection algorithm reveals different detection outcomes, showing that the PSO-based provides better input attack features than the IG-based.

For the detection by Tree from the those select features by IG and PSO-based, 15 misclassifications for the non-consistent algorithm exhibit in **Figure 7**; two Benign misclassifies as Attack and one C&C as Attack. Four Attack misclassifies as Benign and 5 C&C as Benign. One Attack misclassifies as C&C and one Benignas C&C. One C&C misclassifies as PartOfAHorizontalPortScan. On the other hand, for the PSO-based consistent detection algorithm, there are 4 misclassifications in **Figure 8**; one C&C misclassified as Attack, two C&C as Benign, one C&C as PartOfAHorizontalPortScan.

Although features 10 and 14 are common between the non-consistent and consistent algorithms, the feature set for IG and PSO varies wildly depending on the algorithm, and our study was motivated by the need for careful feature selection of the appropriate algorithm through experimentation. The consistent feature selection-and-detection algorithm shows about twice in Tree and one and a half better performance in SVM than the non-consistent algorithm. Note that

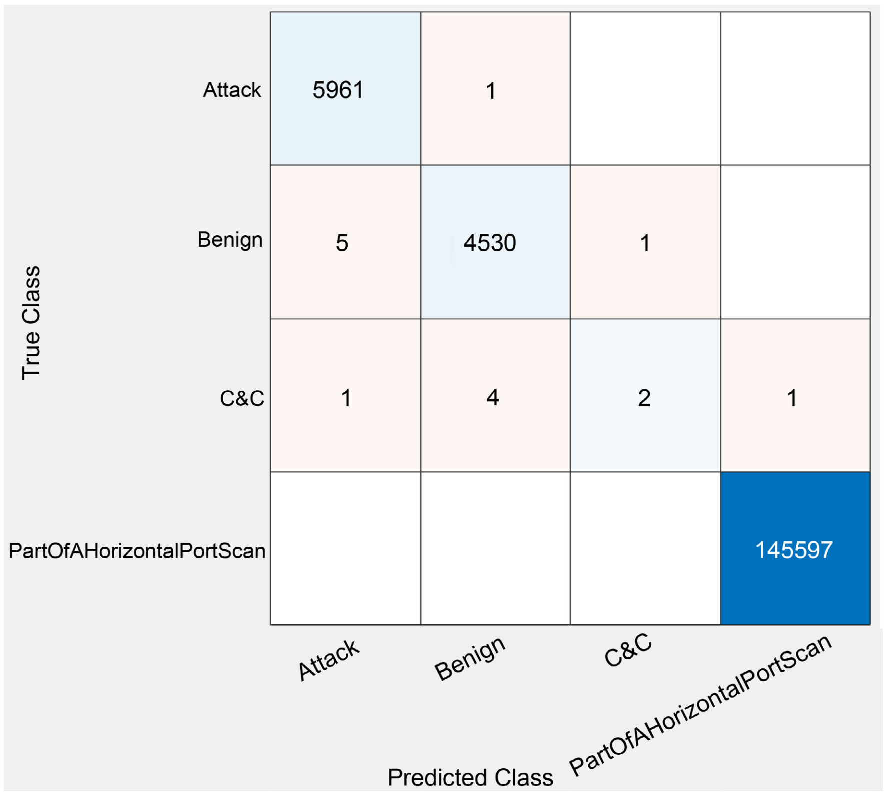


Figure 5. SVM performance with features 8, 9, 10, and 14 by IG-based.

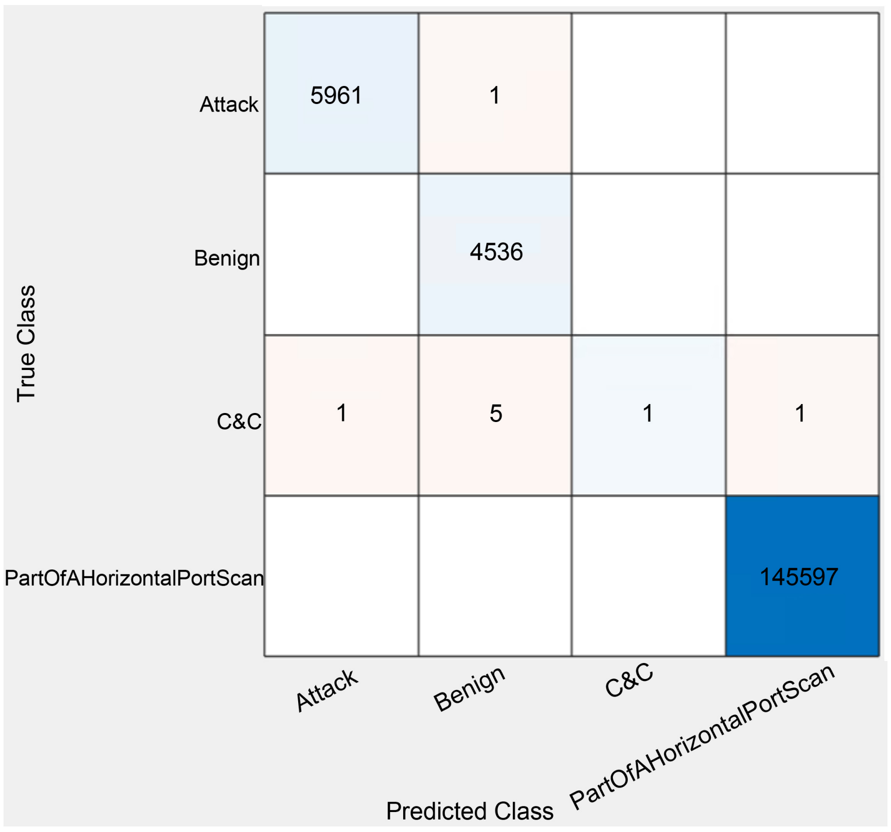


Figure 6. SVM performance with features 10 and 14 by PSO-based.

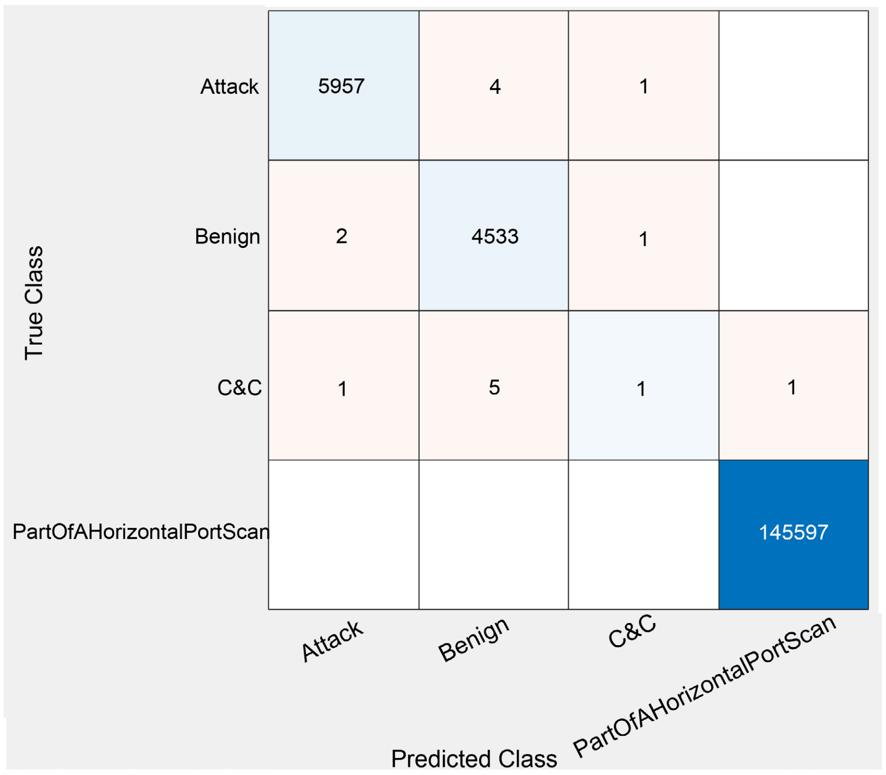


Figure 7. Tree performance for the non-consistent algorithm with features 8, 9, 10, and 14.

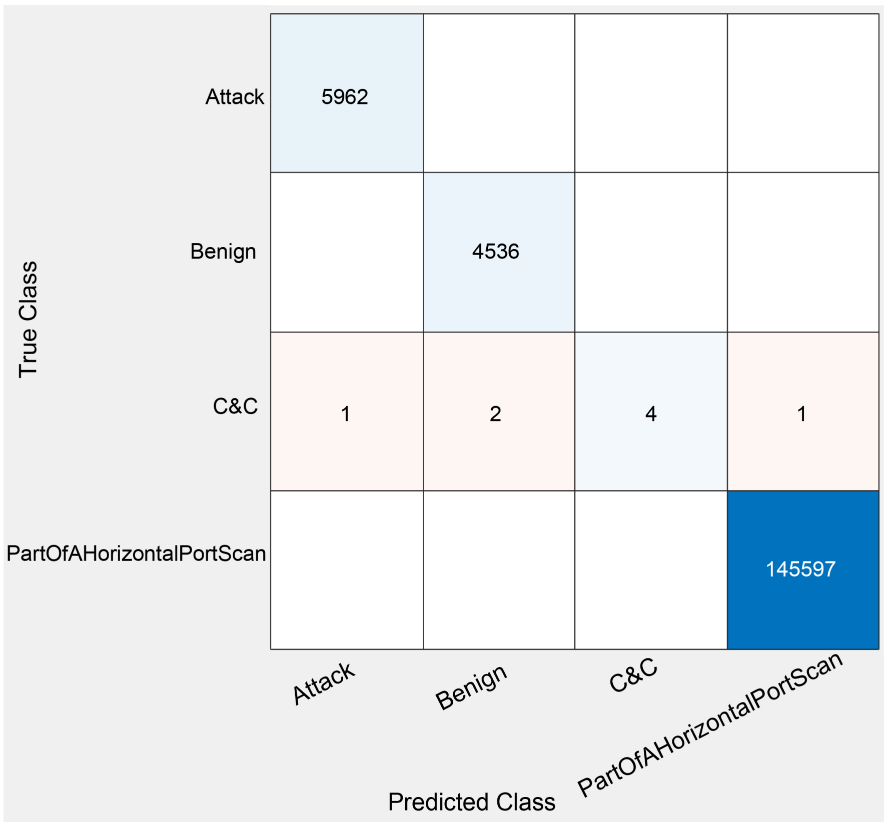


Figure 8. Tree performance for the consistent algorthm with features 10 and 14.

attack refers to implementation but does not capture by the application layer. Detection rate for both non-consistent and consistent algorithms is 100% due to rounding off where the denominator dominates over the numerator, especially for a quite large number of data records. In this scenario, the optimal number of features for the non-consistent and consistent algorithms are 4 and 2, respectively.

The 10th scenario (17-1) has 54,659,855 data records with seven classes: Attack, Benign, C&CHeartBeat, DDoS, Okiru, PartOfAHorizontalPortScan, and PartOfAHorizontalPortScanAttack while the pcap size is 7.8 GB [13]. The most important feature by IG is id.resp_p (4) and features 2 and 4 are selected shown in **Figure 9**. The detection with the IG by Tree shows a total of 3404 misclassification in **Figure 10**; 18 PartOfAHorizontalPortScan misclassifies as Benign. 3 attack misclassifies as PartOfAHorizontalPortScan, 3380 Benign as PartOfAHorizontalPortScan, 2PartOfAHorizontalPortScanAttack as PartOfAHorizontalPortScan. One attack misclassifies as PartOfAHorizontalPortScanAttack and 3 PartOfAHorizontalPortScan as PartOfAHorizontalPortScanAttack. On the other hand, PSO-based feature selection selects features 2, 4, and 14.Surprisingly, PSO-based consistent algorithm shows a total of 7 misclassification in **Figure 11** compared to the 3404 misclassifications for the non-consistent algorithm; one Benign misclassified as Attack, 4 attack as Benign, one PartOfAHorizontalPortScanAttack as PartOfAHorizontalPortScan, and one PartOfAHorizontalPortScan as PartOfAHorizontalPortScanAttack. In this scenario, the optimal number of features for the non-consistent and consistent algorithm are 2 and 3, respectively. Again, detection rate (classification) for both non-consistent and consistent algorithms is 100%.

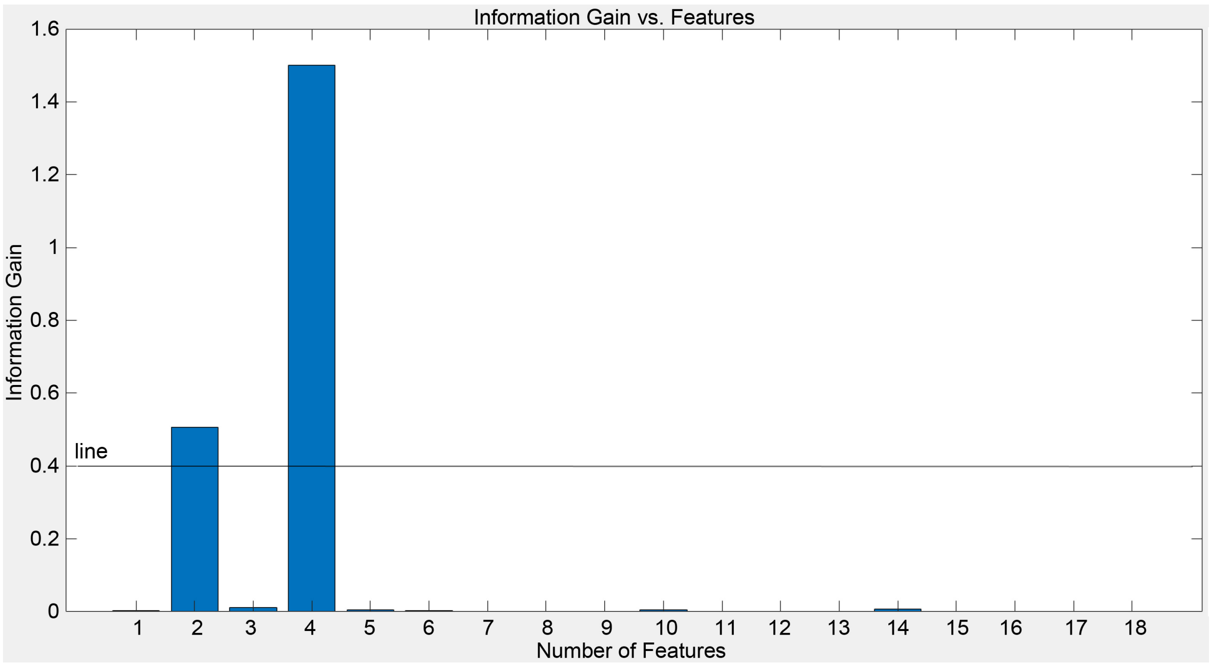


Figure 9. Information gain for scenario 10th.

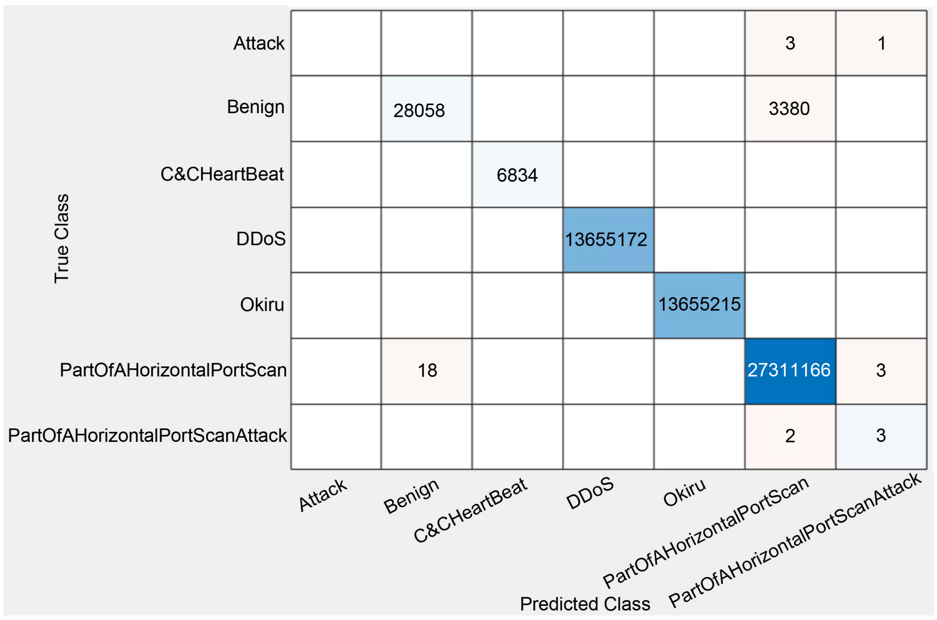


Figure 10. Tree performance for the non-consistent in scenario 10th.

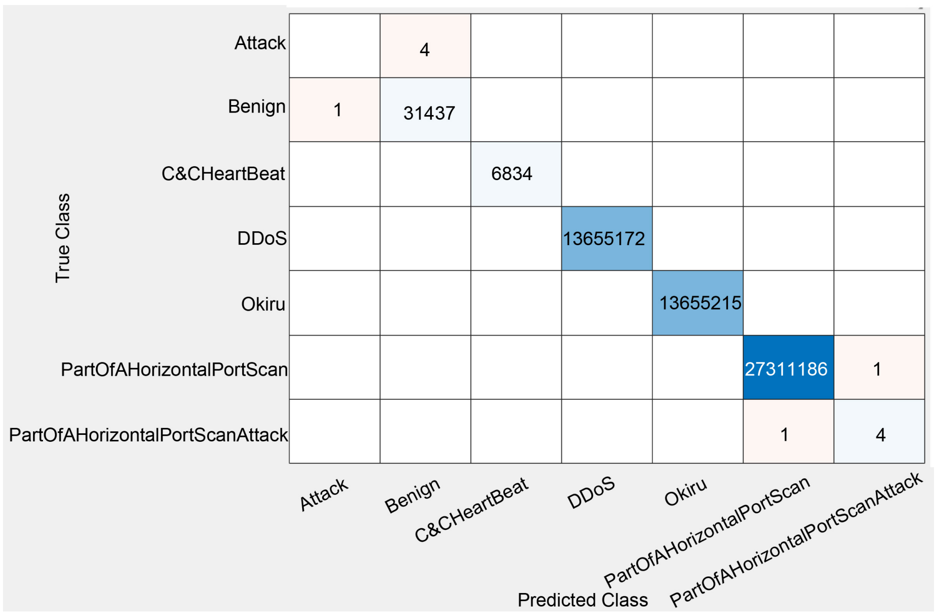


Figure 11. Tree performance for the consistent in scenario 10th.

For further analysis, ensemble ML for both non-consistent and consistent algorithms is evaluated whether it may be unsuitable for IoT-device-based Network. The performance of the non-consistent represents 3404 misclassifications in Figure 12 while the computation time 8371.4 seconds. On the other hand, the consistent algorithm has only 6 misclassifications in Figure 13 while the computation time is 8118.2 seconds. However, when compared directly to the Tree, the detection accuracy is nearly the same, but the computation time for the ensemble is significantly increased, as the computation time for the Tree is 934.13 seconds that is about tenfold less than that of the ensemble.

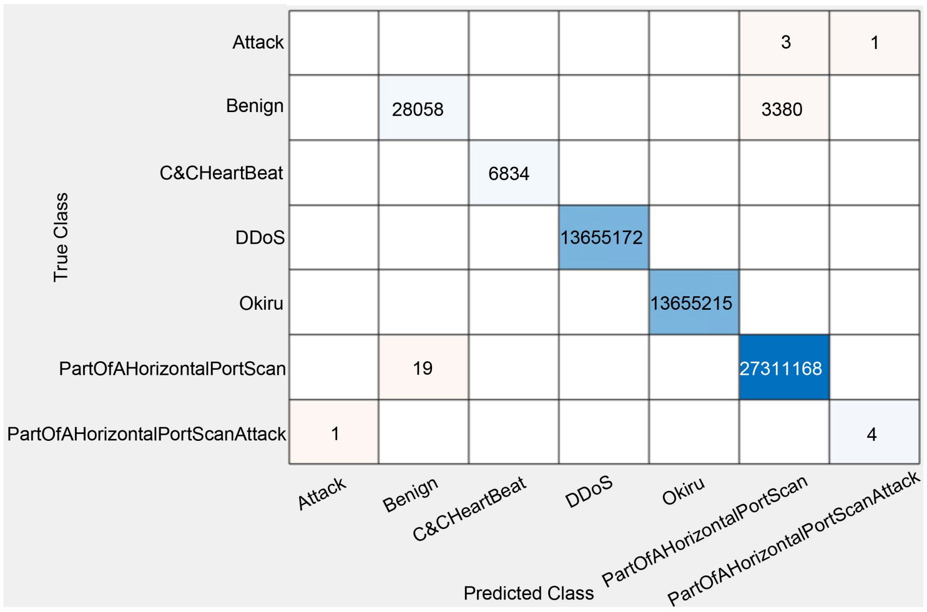


Figure 12. Ensemble performance for the non-consistent in scenario 10th.

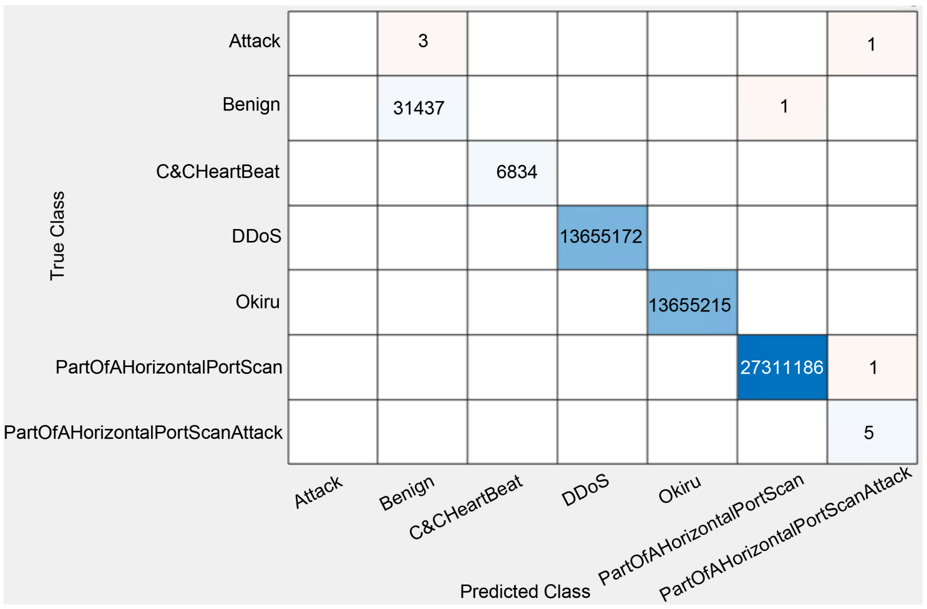


Figure 13. Ensemble performance for the consistent in scenario 10th.

Our prior study [12] and [18] supported that the NSL-KDD dataset with increased inputs/features showed the highest detection performances. For example, for the KDD dataset, the optimal number of features was 15, and the performance of Tree degraded when there were either less than or more than 15 inputs. In contrast, the detection accuracy for Tree in the IoT-23 dataset with fewer features resulted in higher performance than with all features.

4. Conclusion

This paper studies two feature selection algorithms: Information Gain and PSO-based feature selection, to select a minimum number of attack feature sets. The metric of the former is related to entropy, while that of the latter is misclassification. The detection algorithm, Tree, performs multi-classification among diverse attack types for Intrusion Detection System. This shows that the consistent use of the same metrics in feature selection and detection algorithms enhances the classification accuracy when compared to the non-consistent use algorithm counterpart. This paper reveals that feature selection should be made with the most relevant feature for the detection algorithm's metrics to maximize the overall performance when applying ML to the IDS in an IoT-based network since the consistent use algorithm selects the most useful features for the detection model. In addition, the ensemble is evaluated to determine whether it is suitable for an IoT-based network due to its complexity. The performance improvement by the ensemble is insignificant compared to Tree while its computation time is significant. Additional detection algorithms should be evaluated in diverse scenarios of IoT-23 datasets and other datasets for future studies. The current paper studies both Tree and SVM algorithms in select scenarios of IoT-23 datasets.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Painuly, S., Sharma, S. and Matta, P. (2021) Future Trends and Challenges in Next Generation Smart Application of 5G-IoT. 2021 *5th International Conference on Computing Methodologies and Communication* (*ICCMC*), Erode, 8-10 April 2021, 354-357. https://doi.org/10.1109/ICCMC51019.2021.9418471

[2] Li, Y., Chen, W., Ding, Y., Qie, Y. and Zhang, C. (2022) A Vision of Intelligent IoT—Trends, Characteristics and Functional Architecture. 2022 *International Wireless Communications and Mobile Computing* (*IWCMC*), Dubrovnik, 30 May-3 June 2022, 184-189. https://doi.org/10.1109/IWCMC55113.2022.9824304

[3] Shafique, K., Khawaja, B.A., Sabir, F., Qazi, S. and Mustaqim, M. (2020) Internet of Things (IoT) for Next-Generation Smart Systems: A Review of Current Challenges, Future Trends and Prospects for Emerging 5G-IoT Scenarios. *IEEE Access*, **8**, 23022-23040. https://doi.org/10.1109/ACCESS.2020.2970118

[4] Georgiana Dorobantu, O. and Halunga, S. (2020) Security Threats in IoT. 2020 *International Symposium on Electronics and Telecommunications* (*ISETC*), Timisoara, 5-6 November 2020, 1-4. https://doi.org/10.1109/ISETC50328.2020.9301127

[5] Iqbal, W., Abbas, H., Daneshmand, M., Rauf, B. and Bangash, Y.A. (2020) An In-Depth Analysis of IoT Security Requirements, Challenges, and Their Countermeasures via Software-Defined Security. *IEEE Internet of Things Journal*, **7**, 10250-10276. https://doi.org/10.1109/JIOT.2020.2997651

[6] Olowononi, F.O., Rawat, D.B. and Liu, C. (2021) Resilient Machine Learning for Networked Cyber Physical Systems: A Survey for Machine Learning Security to Securing Machine Learning for CPS. *IEEE Communications Surveys & Tutorials*, **23**, 524-552. https://doi.org/10.1109/COMST.2020.3036778

[7] Zhang, Pan, L., Han, Q.-L., Chen, C., Wen, S. and Xiang, Y. (2022) Deep Learning

Based Attack Detection for Cyber-Physical System Cybersecurity: A Survey. *IEEE*/*CAA Journal of Automatica Sinica*, **9**, 377-391. https://doi.org/10.1109/JAS.2021.1004261

[8] Kayacik, Gunes, H. and Nur Zincir-Heywood, A. (2009) Current Challenges in Intrusion Detection Systems. In: Tiako, P.F., Ed., *Software Applications*: *Concepts*, *Methodologies*, *Tools*, *and Applications*, IGI Global, Hershey, 458-466. https://doi.org/10.4018/978-1-60566-060-8.ch031

[9] Adhi Tama, B. and Rhee, K.H. (2015) A Combination of PSO-Based Feature Selection and Tree-Based Classifiers Ensemble for Intrusion Detection Systems. In: Park, D.-S., *et al*., Eds., *Advances in Computer Science and Ubiquitous Computing*, Springer, Berlin, 489-495. https://doi.org/10.1007/978-981-10-0281-6_71

[10] Ogundokun, R.O., Awotunde, J.B., Sadiku, P., Adeniyi, E.A., Abiodun, M. and Dauda, O.I. (2021) An Enhanced Intrusion Detection System Using Particle Swarm Optimization Feature Extraction Technique. *Procedia Computer Science*, **193**, 504-512. https://doi.org/10.1016/j.procs.2021.10.052

[11] Tran, B., Zhang, M. and Xue, B. (2016) A PSO Based Hybrid Feature Selection Algorithm for High-Dimensional Classification. 2016 *IEEE Congress on Evolutionary Computation* (*CEC*), Vancouver, 24-29 July 2016, 3801-3808. https://doi.org/10.1109/CEC.2016.7744271

[12] Kim, Y.G., Kwon, O. and Yoon, J. (2020) Proportional Voting Based Semi-Unsupervised Machine Learning Intrusion Detection System. *Journal of Computer Science and Information Technology*, **8**, 1-9. https://doi.org/10.15640/jcsit.v8n2a1

[13] Garcia, S., Parmisano, A. and Erquiaga, M.J. (2020) IoT-23: A Labeled Dataset with Malicious and Benign IoT Network Traffic (Version 1.0.0) [Data Set].

[14] Shafiq, M., Tian, Z., Sun, Y., *et al*. (2020) Selection of Effective Machine Learning Algorithm and Bot-IoT Attacks Traffic Identification for Internet of Things in Smart City. *Future Generation Computer Systems*, **107**, 433-442. https://doi.org/10.1016/j.future.2020.02.017

[15] Güneş Kayacık, H., Nur Zincir-Heywood, A. and Heywood, M.I. (2005) Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets. *Proceedings of the 3rd Annual Conference on Privacy*, *Security and Trust*, Vol. 94, 85-89.

[16] Igbe, O., Darwish, I. and Saadawi, T. (2016) Distributed Network Intrusion Detection Systems: An Artificial Immune System Approach. 2016 *IEEE First International Conference on Connected Health*: *Applications*, *Systems and Engineering Technologies* (*CHASE*), Washington DC, 27-29 June 2016, 101-106. https://doi.org/10.1109/CHASE.2016.36

[17] Kim, Y.G. and Lee, M.J. (2014) Scheduling Multi-Channel and Multi-Timeslot in Time Constrained Wireless Sensor Networks via Simulated Annealing and Particle Swarm Optimization. *IEEE Communications Magazine*, **52**, 122-129. https://doi.org/10.1109/MCOM.2014.6710073

[18] Kwon, O., Kim, Y.G. and Mendoza, B. (2021) Machine Learning-Based Intrusion Detection System for Home/Enterprise Internet Network. UKC, Los Angeles.