Efficient Intrusion Detection System for IoT Environment

Rehab Hosny Mohamed[1], Faried Ali Mosa[2], Rowayda A. Sadek[3]
Information Technology Department, Faculty of Computers & Artificial Intelligence, Beni-Suef University, Cairo, Egypt[1, 2]
Information Technology Department, Faculty of Computers & Artificial Intelligence, Helwan University, Cairo, Egypt[3]

***Abstract*—These days, the Internet is subjected to a variety of attacks that can harm network devices or allow attackers to steal the most sensitive data from these devices. IoT environment provides new perspective and requirements for Intrusion detection due to its heterogeneity. This paper proposes a newly developed Intrusion Detection System (IDS) that relies on machine learning and deep learning techniques to identify new attacks that existed systems fail to detect in such an IoT environment. The paper experiments consider the benchmark dataset ToN_IoT that includes IoT services telemetry, Windows, Linux operating system, and network traffic. Feature selection is an important process that plays a key role in building an efficient IDS. A new feature selection module has been introduced to the IDS; it is based on the ReliefF algorithm which outputs the most essential features. These extracted features are fed into some selected machine learning and deep learning models. The proposed ReliefF-based IDSs are compared to the existed IDSs based correlation function. The proposed ReliefF-based IDSs model outperforms the previous IDSs based correlation function models. The Medium Neural Network model, Weighted KNN model, and Fine Gaussian SVM model have an accuracy of 98.39 %, 98.22 %, and 97.97 %, respectively.**

Keywords*—*Intrusion detection systems (IDSs); TON IoT dataset; machine learning; deep learning; ReliefF

I. Introduction

The Internet of Things (IoT) environment describes a network of physical objects that includes software, sensors, and other technologies to connect and transfer data with systems and devices on the Internet. IoT represents the interconnection of heterogeneous entities, where the term “entity” refers to a human being, a sensor, or possibly anything that can request or provide a service [1]. The Internet of Things (IoT) has become one of the most important technologies of the 21st century, so we need to protect data traveling between devices. IoT security is an ongoing research topic that's gaining increasing attention in governmental, industrial, and academic research. IoT security requirements are Confidentiality, Integrity, Availability, Authentication, and Authorization. There are multiple IDS systems that are used to protect networks from attacks, but some systems can’t identify the newest attacks. We provide in this paper an intrusion detection system to detect attacks, the system is developed using deep learning and machine learning techniques with an efficient feature selection using the ReliefF algorithm. The proposed model is based on the Windows 10 subset of the ToN-IoT dataset. A comparison is carried out with the existed work which selects features from Windows 10 dataset using the correlation function.

In this paper, we focus on Windows 10 dataset as the windows 10 operating system is widely used in personal computers. This dataset contains new attacks for IIoT networks which can’t be detected by traditional IDSs so this paper presents a new IDS system for IoT networks by using machine learning and deep learning algorithms. Many datasets have redundant or noisy data which may cause poor learning performance and consume time for training so the feature selection techniques are used to remove irrelevant and noisy attributes from the dataset. In this paper, we apply the ReliefF feature selection method on the windows 10 dataset.

The paper structure is as follows: Section II reviews the IDS backgrounds, Datasets, feature selection techniques, and Machine learning and deep learning algorithms. Section III discusses Related Works on IDS. The proposed model is explained in Section IV. Following that the Evaluation Metrics and Experimental Results are described in Section V. Finally, Section VI describes the conclusions of the future work.

II. Background

Intrusion detection systems (IDSs) are tools or software that can secure or defend your networks from intrusions. The purpose of IDS is to recognize various types of attacks and usage of the computer that cannot be known via firewalls. This is often very important to achieve a high level of protection versus action which compromises the availability, integrity, or confidentiality of a computer system [2]. IDS monitors network traffic for suspicious activities and issues alerts when such activities are detected. Some intrusion detection systems can do actions if anomalous traffic is detected such as blocking traffic sent from a suspicious IP address. IDS sometimes is classified based on its location; Host Intrusion Detection Systems (HIDS) and Network Intrusion Detection Systems (NIDS) [3]. HIDS monitors and analyzes host activities, application logs, system calls, and modifications that occur on system files to identify intrusions such as login to unprivileged data [4]. Network-based IDS monitors the network traffic by using techniques such as packet sniffing to collect network traffic and detect attacks such as DOS attacks or port scans [4]. There are two types of Network-based IDS Statistical anomaly IDS and Pattern matching IDS [3]. On the other hand, IDS is practically classified as signature-based IDS and anomaly-based IDS. Anomaly-based IDS is significantly used for detecting zero-day attacks. Efficient IDS development depends on selecting suitable machine learning or deep learning models as well as the feature selection technique in order to enhance the performance and reduce the computations.

Feature selection is an important step that plays a significant role in the development of an effective IDS. Feature selection is categorized into three methods like filter methods, wrapper methods, and embedded methods. The filter methods select the most discriminating features by data character. Features are ranked according to specific criteria and then the highest-ranked features are selected. There are many types of filter methods that have been used, such as ReliefF, F-statistic, mRMR, correlation, and information gain. Wrapper methods use the intended learning algorithm to evaluate the features. Examples of wrapper methods are Sequential forward selection and Sequential backward selection. Embedded methods perform feature selection in the model construction process examples of embedded methods are Elastic Net and Ridge Regression [5]. The most widely used feature selection methods are The ReliefF (Relief-F) [6], Correlation-Based Feature Selection [7, 8], and Information Gain Ratio-based feature selection [9, 10].

Various network datasets were prepared for the evaluation of IDS, such as KDDCUP99, NSLKDD [11], UNSW-NB15 [12], Ton-IoT, and ISCX [13].

There are several techniques of classification like decision trees, naïve Bayes (NB), neural networks (NN), support vector machines (SVM), Random Forest (RF), rule-based system, and nearest neighbor (KNN) [14,15]. Every algorithm utilizes learning methods to create a classification model. However, proper classification techniques should handle the training data and should be able to accurately identify a class of records that have not ever been seen before [4].

Deep Learning a branch of Machine Learning, has also begun to be widely used to implement IDS. There are many types of deep learning algorithms that can be used for intrusion detection systems such as recurrent neural network, deep belief network, Long Short Term Memory, restricted Boltzmann machine, deep auto-encoder, self-taught learning, convolutional neural network, deep migration learning, and replicator neural network [16, 17].

III. Related Work

There are lots of researches that use some methodologies and algorithms to overcome the most advanced attacks in the network. These researches trend to intrusion detection systems (IDSs) by using machine learning and deep learning algorithms with different feature selection methods to select the most important features from the datasets. The related works are explained in this section as follows:

Senthilnayaki Balakrishnan et al [18], the authors developed a new feature selection method based on Information Gain Ratio. They had named it an Optimal Feature Selection algorithm that could select an optimal number of features from the dataset. They used the KDD Cup dataset. They used Support Vector Machine and Rule-Based Classification algorithms to classify data as normal or as attacks data. The proposed feature selection and classification algorithms had achieved the best results.

S. Ramakrishnan et al. [19], the authors developed an IDS to detect attacks or normal data from the KDD Cup 99 dataset. They first selected the most important attributes by using the entropy-based feature selection method. The Fuzzy Control Language was used to classify data as normal or attack data. The results showed that the proposed system reduced the computational time and achieved high accuracy.

Peilun Wu et al. [20], in this paper, the authors developed an IDS system called Densely-ResNet. They selected the important attributes using the correlation function. The UNSW-NB15 was used to evaluate the performance of Densely-ResNet. The results showed that the Densely-ResNet had achieved high accuracy and reduced false alarm rate.

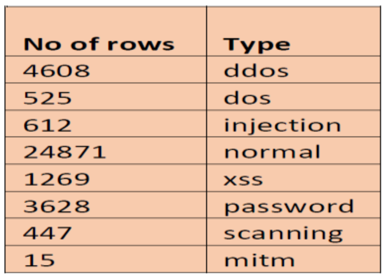
Shahid Latif et al. [21], the authors proposed an IDS that utilized a deep random neural network to protect IIoT systems. However, the system had been tested with the UNSW NB15 dataset. This dataset was used to observe its applicability or feasibility to IIoT. The results showed a better detection accuracy with a low false alarm rate.

Merna Gamal et al [14], the authors proposed a hybrid intrusion detection system using machine learning and deep learning techniques to benefit from the advantage of deep learning and machine learning. They used 10% of the KDDcup1999 dataset, and applied CNN to extract features from the KDDcup1999 dataset, then used machine learning techniques (SVM, KNN) to classify the data. The experiment results showed that the proposed model had achieved the best accuracy.

Prabhat Kumar et al [22], the authors developed an ensemble learning and fog-cloud architecture-driven cyberattack detection framework for IoMT networks. They used a ToN-IoT dataset that was collected from a large-scale and heterogeneous IoT network. Results showed that the proposed system could achieve high accuracy of 96.35%, a high detection rate of 99.98%, and a reduction in false alarm rate by 5.59%.

IV. Proposed Model

This section provides our proposed model. We select features from windows 10 datasets using the ReliefF algorithm. We apply the ReliefF algorithm to remove irrelevant variables from Windows 10 dataset. Windows 10 dataset contains 125 attributes and two attributes one called “Label” whose values are (0 or 1), 0 for normal data and 1 for attack data, and “type” attribute contains normal data and seven attack categories (DDOS, DOS, Injection, MITM, Password, Scanning, and XSS). Before selecting features from the windows 10 dataset, we do preprocess on a dataset. The “type” feature refers to attack categories, these categories are converted from text data to numeric data from 2 to 8 respectively, and normal data are converted to 1. Fig. 1 shows the number of rows for each type of data in the dataset. Then we divide the dataset into training and testing data using holdout validation, 80 % of the dataset is training data and 20 % of the dataset is used as testing data.



No of rows	Type
4608	ddos
525	dos
612	injection
24871	normal
1269	xss
3628	password
447	scanning
15	mitm

Fig. 1. Number of Rows for each Type of Data.

The ReliefF algorithm is used to select features from the windows 10 dataset. After selecting features, we apply deep learning and machine learning techniques to classify the data as normal or attacks data. The Matlab tool and Machine Learning Toolbox (Classification Learner) are used to simulate algorithms such as KNN, SVM, and NN. We apply different types of KNN algorithm (such as Weighted KNN and Medium KNN), SVM (like Linear SVM and Fine Gaussian SVM), and Neural Network (like Medium NN and Bilayered NN). Also, we apply the LSTM algorithm to the dataset. Fig. 2 shows the flowchart of our proposed model.

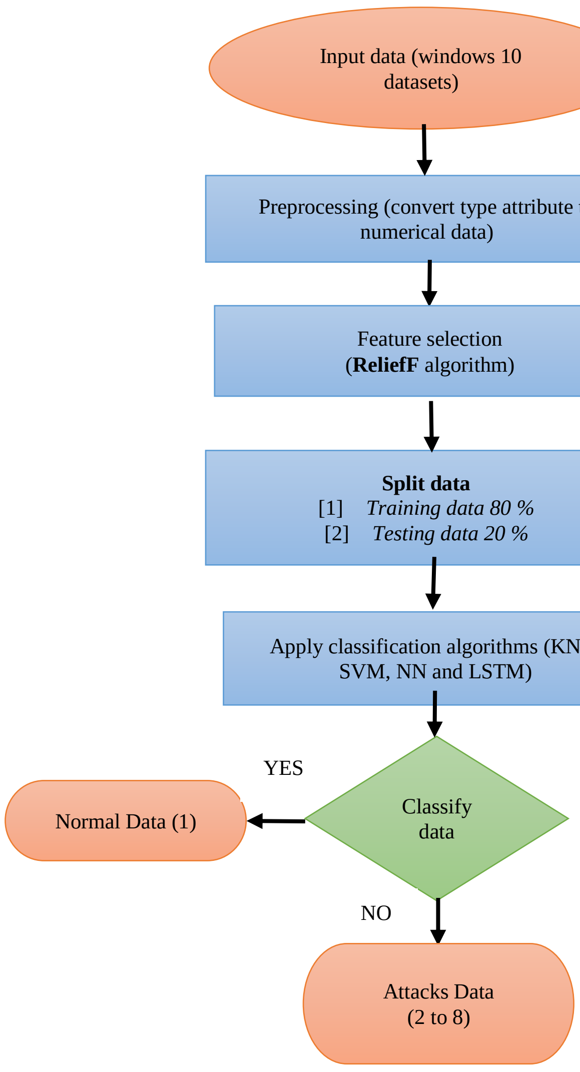

Input data (windows 10 datasets)
Preprocessing (convert type attribute to numerical data)
Feature selection (ReliefF algorithm)
Split data
[1] Training data 80 %
[2] Testing data 20 %
Apply classification algorithms (KNN, SVM, NN and LSTM)
Classify data
YES
Normal Data (1)
NO
Attacks Data (2 to 8)

Fig. 2. Flowchart of Proposed Model.

V. Simulation and Experimental Results

Experiments are carried out to examine the performance of the proposed model.

A. *Ton-IoT Datasets Description*

Ton-IoT datasets were collected from Telemetry datasets of IoT services, Operating systems datasets of Windows and Linux, as well as datasets of Network traffic. The Windows datasets were generated using the virtual machines running Windows 7 and Windows 10 and incorporated the collections of data from multiple sources, including memory, process, processor, and hard drive of the systems. Windows 10 dataset contained 125 attributes and two attributes one called "Label" which values are (0 or 1), 0 for normal data and 1 for attack data, and the "type" attribute contained normal data and seven attack categories.

We apply our proposed model to Windows 10 dataset. We select features from it using the ReleifF algorithm then we apply deep learning and machine learning techniques to classify the data as normal or attack data. We divide the dataset using holdout validation as the following; 80 % of the dataset is used for training (28780 data sample records). 20 % of the dataset is used for testing (7195 data sample records).

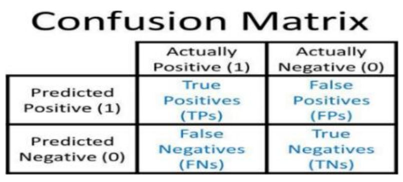
B. *Simulation Environment Properties*

The proposed system is implemented using Matlab 2021a, Classification learner App is set to apply imported and preprocessed instances of the dataset the processor unit (CPU) is Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz 2.40 GHz, main memory(RAM) is 16.0 GB and the operating system is Windows 10 Pro.

C. *Evaluation Metrics*

There are performance metrics are calculated to examine the model performance. These metrics are accuracy, recall, precision rate and, F1-Measure ought to be calculated for evaluation. These all performance metrics can be calculated by using a confusion matrix.

- Confusion matrix is a table or array that represents the performance of a classification model on testing data for which the true values are identified. Fig. 3 shows the confusion matrix.



Confusion Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Fig. 3. Confusion Matrix.

- Accuracy is the ratio of true detection over the whole instances.

$$\boldsymbol{accuracy} = \frac{\text{TP+TN}}{\text{totalSample}} \quad (1)$$

- Recall is how often does it predict correctly. Also known as Sensitivity or True Positive Rate (TPR).

$$\boldsymbol{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (2)$$

- Precision indicates how often it is accurate when it is predicted to be accurate.

$$\boldsymbol{precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{3}$$

- F1-measure is the average of recall and precision weight. The mathematical representation of all measures can be deduced from the confusion matrix.

$$\boldsymbol{F - Mesure} = 2 * \frac{precision * Recall}{precision + Recall} \tag{4}$$

Area Under ROC Curve (AUC) is the performance metric for binary classification issues. AUC represents the work of the model for distinguishing between the negative classes and positive classes. An area of 1.0 acts as a model in which all predictions are made perfectly. An area of 0.5 acts as a model that is as good as random. The ROC can be classified into specificity and sensitivity.

D. Feature Selection

The paper in [23] selects features from Windows 7 and Windows 10 datasets using a correlation function that selects the most correlated features in datasets, and the authors suggest applying machine learning and deep learning algorithms to selected features. In our proposed model we suggest modifying the feature selection step, we apply the ReliefF algorithm to remove irrelevant variables from Windows 10 dataset. We compare our proposed model with the work implemented in the paper [23].

We use the ReliefF algorithm because it is a widely used filter-based feature selection method that finds the best feature subset by calculating the features' weights. The ReliefF algorithm's advantages are that more robust and can deal with incomplete and noisy data. It consumes less computational resources and is not limited to two-class problems, but it can work with multiple classes [6, 24]. The ReliefF algorithm first selects R samples randomly from the training set, and then it finds Near Hits in the same class, finds Near Miss in other classes, and updates each feature weight by rules. After repeating the above action m times, all feature weights will be obtained. The higher the feature weight is, the more useful the feature is for classification. The reverse is also true. The following steps are the pseudocode of the ReliefF algorithm:

Pseudocode *for* ReliefF *Algorithm:*

```
Input: a vector of attribute values and the class value
for each training instance
Output: the vector W of estimations of the attributes qualifies.

Step 1: put up all of the weights $W[A] := 0.0$;
Step 2: for i:= 1 to $m$ do begin
Step 3: randomly select an instance $R_i$;
Step 4: find $k$ nearest hits $H_j$;
Step 5: for each class $C \neq \text{class}(R_i)$ do
Step 6: from class $C$ find $k$ nearest misses $M_j(C)$;
Step 7: for $A$:= 1 to a do
Step 8: $W[A] := W[A] - \sum_{j=1}^{K} \frac{\text{diff}(A, Ri, Hi)}{m.K} +$
Step 9: $\sum_{C \neq \text{class}(Ri)} [\frac{P(C)}{1-P(\text{class}(Ri))} \sum_{j=1}^{K} \frac{\text{diff}(A, Ri, Mj(C))}{m.K}$
Step 10: end;
```

E. Result and Discussion

ReliefF algorithm estimates the weight of each feature and sorts features according to their weights. Fig. 4 shows the most important weights of 20 selected features, the highest feature weight is equal to 0.257.

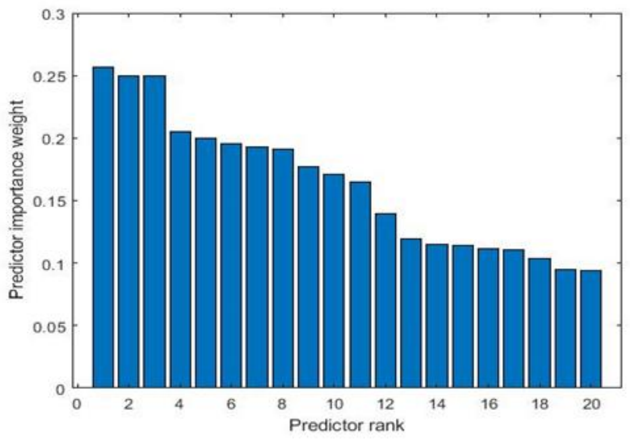


Fig. 4. Weights of Selected Features.

Tables I and II show features selected in the windows 10 dataset using the correlation function in [23] and features selected using the ReliefF algorithm in our proposed model, respectively. The selected features will be then used for training and testing machine learning and deep learning algorithms to evaluate their efficiency in classifying the dataset as normal or attacks data.

After selecting features, we apply deep learning and machine learning techniques to classify the data as normal or as attacks data. The algorithms applied are KNN, SVM, Neural Network, and LSTM. We apply different types of KNN algorithms (such as Weighted KNN and Medium KNN), SVM (like Linear SVM and Fine Gaussian SVM), and Neural Network (like Narrow NN and Bilayered NN). Table III shows the performance metrics of classification algorithms applied to the features selected in [23] in the windows 10 dataset. Table IV shows performance metrics of classification algorithms applied to the features selected using the ReliefF algorithm in windows 10 datasets.

TABLE I. FEATURES SELECTED FROM WINDOWS 10 DATASETS USING CORRELATION FUNCTION

Feature selection method	Selected Features
correlation function [23]	1. Network I.Intel.R 82574L GNC.Current Bandwidth 2. Network I.Intel.R 82574L GNC.Packets Sent Unicast.sec 3. Network I.Intel.R8 2574L GNC. Packets Sent.sec 4. LogicalDisk. Total Disk Read Bytes.sec 5. LogicalDisk Total. Avg.Disk.Bytes Transfer 6. Memory.Modified. Page List Bytes 7. Memory.Pool.Paged Bytes 8. Memory.Page Reads.sec 9. Process IO Data Operations sec 10. Processor pct Processor Time

TABLE II. FEATURES SELECTED FROM WINDOWS 10 DATASETS USING THE RELIEFF ALGORITHM

Feature selection method	Selected Features
ReliefF algorithm in proposed model	1. MemorySystemDriverResidentBytes 2. Process_Virtual_Bytes 3. Process_Virtual_BytesPeak 4. MemoryLong_TermAverageStandbyCacheLifetime_s 5. Process_Pool_PagedBytes 6. Process_HandleCount 7. Network_I_IntelR_82574L_GNC_TCP_APS 8. Process_WorkingSet 9. MemorySystemCacheResidentBytes 10. Process_ThreadCount 11. MemoryPct_CommittedBytesInUse 12. Process_Working_Set_Peak 13. MemoryPoolNonpagedBytes 14. MemoryStandbyCacheReserveBytes 15. MemorySystemDriverTotalBytes 16. Process_Working_Set_Private 17. MemoryCacheBytesPeak 18. MemoryCommitLimit 19. MemoryStandbyCacheCoreBytes 20. MemoryPoolPagedResidentBytes

TABLE III. PERFORMANCE METRICS OF CLASSIFICATION ALGORITHMS IN FEATURES SELECTED USING CORRELATION FUNCTION IN [23]

Classification algorithm	Accuracy (%)	Precision	Recall	F_measure	AUC
Medium KNN Model	93.47	0.664	0.957	0.785	0.99
Weighted KNN Model	94.02	0.686	0.956	0.799	0.99
Linear SVM Model	75.48	0.006	0.006	0.006	0.97
Fine Gaussian SVM Model	91.22	0.594	0.926	0.724	0.99
Bilayered NN Model	92.23	0.622	0. 956	0.753	0.99
Medium NN Model	94.12	0.689	0.962	0.803	0.99

TABLE IV. PERFORMANCE METRICS OF CLASSIFICATION ALGORITHMS IN FEATURES SELECTED USING THE RELIEF ALGORITHM IN THE PROPOSED MODEL

Classification algorithm	Accuracy (%)	Precision	Recall	F_measure	AUC
Medium KNN Model	98.15	0.994	0.979	0.987	0.99
Weighted KNN Model	98.22	0.991	0.983	0.987	0.99
Linear SVM Model	96.54	0.985	0.964	0.975	0.99
Fine Gaussian SVM Model	97.97	0.992	0.978	0.985	0.99
Bilayered NN Model	98.17	0.996	0.977	0.987	1.00
Medium NN Model	**98.39**	0.996	0.980	0.988	1.00

TABLE V. PERFORMANCE METRICS OF LSTM IN FEATURES SELECTED USING THE RELIEFF ALGORITHM AND CORRELATION FUNCTION

LSTM model	Accuracy	Precision	Recall	F_measure
ReliefF algorithm	70 %	0.72	0. 94	0.81
Correlation function	68.9 %	0.71	0.95	0.81

Long Short Term Memory algorithm (LSTM) is also applied to features of windows 10 which are selected from using the correlation function in the paper [23] and the ReliefF algorithm in our proposed model. Table V shows the comparison of performance metrics of the LSTM model applied to the features selected using the correlation function in the paper [23] and the ReliefF algorithm in our proposed model in windows 10 datasets.

The results showed that the accuracy of the LSTM model is 68.9 % of features selected from using the correlation function, and the accuracy of the LSTM model is 70 % of features selected from using the ReliefF algorithm and the Precision is 0.72 in our proposed model unlike in the paper [23] is 0.71. This indicates that our proposed model has achieved the best results.

The results of Table III and Table IV showed that our proposed model has achieved the best results, as in our proposed model the Medium NN Model has best results than in results in the paper [23]. The accuracy of the Medium NN Model in our proposed model is 98.39 and the Precision is 0.996 unlike in the paper [23] the Medium NN Model has achieved 94.12% and Precision is 0.689. Also Weighted KNN Model and Bilayered NN Model accuracy is 98.22%, and 98.17%, respectively. Unlike in [23], the Weighted KNN Model and Bilayered NN Model accuracy is 94.02%, and 92.23%, respectively.

The accuracy of the Linear SVM algorithm in our proposed model is 96.54 %, but in paper [23] the accuracy is 75.48%. The reason this algorithm has achieved lower results than other machine learning algorithms is that this algorithm works more efficiently with two classes, but if it works with a dataset that has multiple classes may achieve low accuracy, and it isn't suitable for large datasets.

LSTM also has archived low accuracy as it doesn't work efficiently with a large number of samples. As the number of samples increases, the accuracy of LSTM tends to decrease. Figure 5 shows a graphical representation of the accuracy of algorithms applied to features selected from the windows 10 dataset. This comparison shows that our proposed model has achieved the best results than the results of [23].

Fig. 6 and 7 show the graphical representation of performance metrics of our proposed model and of [23], respectively.

Fig. 6 and 7 show that our proposed model has achieved high values of performance metrics against another work [23].

All results of our proposed model are better than the results of [23]. It indicates that the ReliefF algorithm selects the most important features and it can work with multiple classes.

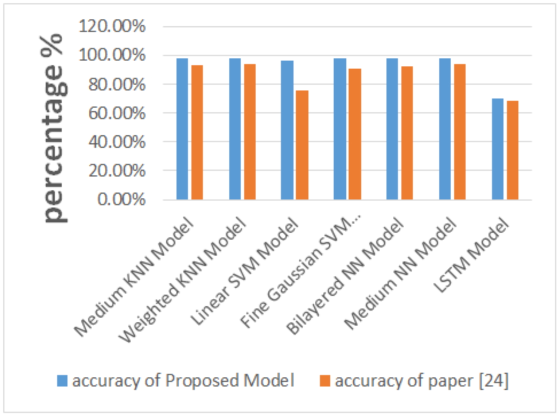


Fig. 5. Accuracy Comparison between Paper [23] and our Proposed Model.

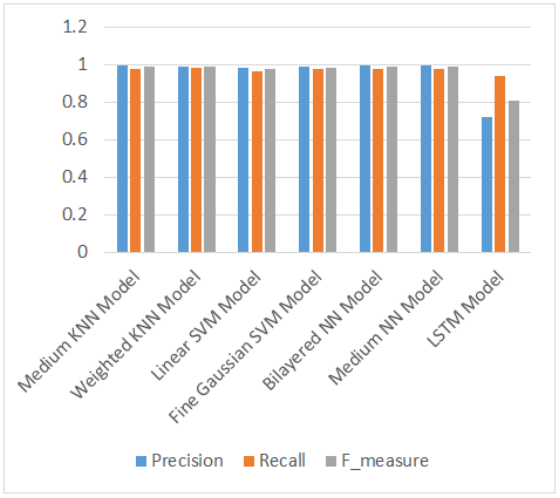


Fig. 6. Performance Metrics of our Proposed Model.

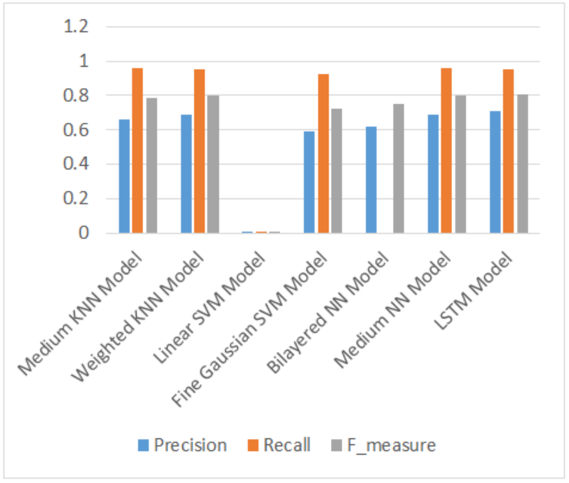


Fig. 7. Performance Metrics of Paper [23].

VI. Conclusion

This paper has introduced an efficient IDS for the IoT environment. The paper uses the benchmark dataset ToN_IoT which includes IoT telemetry data. ReliefF algorithm is proposed to efficiently select the most vital features to get more enhanced performance and computations reduction. A comparison is carried out with the correlation function by applying machine learning and deep learning algorithms to the selected features from the windows 10 dataset. The proposed model has been applied to the windows 10 subset. The experimental results reveal that our proposed model has achieved better results than theirs. Our proposed model has high accuracy such as using Medium Neural Network, Weighted KNN, and Fine Gaussian SVM has 98.39 %, 98.22 %, and 97.97 % respectively. In the future, we will use Transfer Learning to enhance the accuracy of the Long Short Term Memory (LSTM) model.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," Computer Networks 54, no. 15 (2010): 2787-2805.

[2] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," Cybersecurity (2019).

[3] J. Surana, J. Sharma, I. Saraf, N. Puri, and B. Navin “A Survey On Intrusion Detection System,” International Journal of Engineering Development and Research (2017).

[4] A.J. Deepa and V. Kavitha, “A Comprehensive Survey on Approaches to Intrusion Detection System,” Procedia Engineering (2012).

[5] J. Miao and L. Niu, “A Survey on Feature Selection,” Information Technology and Quantitative Management (ITQM 2016).

[6] M. ROBNIK and I. KONONENKO, “Theoretical and Empirical. Analysis of ReliefF and RReliefF,” Machine Learning (2003).

[7] Y. Chen, Y. Li, X. Cheng, and L. Guo,” Survey and Taxonomy of Feature Selection Algorithms in Intrusion Detection System,” Information Security and Cryptology, Lecture Notes in Computer science (2006).

[8] M. A. Hall, “Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning,” In Proc. of the 17th Int. Conf. on Machine Learning. Morgan Kaufmann Publishers Inc (2000).

[9] M. Trabelsia, N. Meddouria, and M. Maddouri, “A New Feature Selection Method for Nominal Classifier based on Formal Concept Analysis,” International Conference on Knowledge Based and Intelligent Information and Engineering Systems (September 2017).

[10] M.sujatha, and Dr. G. Lavanya Devi, “Feature Selection Techniques using for High Dimensional Data in Machine Learning,” International Journal of Engineering Research & Technology (IJERT) (2013).

[11] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," IEEE Symposium on Computational Intelligence for Security and Defense Applications (2009).

[12] N. Moustafa, and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Dataset," Military Communications and Information Systems Conference (MilCIS). IEEE (2015).

[13] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," ICISSP (2018).

[14] M. Gamal, H. Abbas, and R. A. Sadek, “Hybrid Approach for Improving Intrusion Detection Based on Deep Learning and Machine Learning Techniques,” Springer (2020).

[15] O. Muhammad Altoumi Alsyaibani, E. Utami, and A. Dwi Hartanto, “Survey on Deep Learning Based Intrusion Detection System,” Telematika (2021).

[16] A. Mehnas Muthalib, S. Theres Mathew, S. Tom Mathew, R. Issac, and D. Sunny, “A SURVEY ON DEEP LEARNING ALGORITHMS,” International Research Journal of Engineering and Technology (IRJET, 2020).

[17] A. Aldweesh, A. Derhab, and A. Z. Emam, “Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues,” Journal Pre-proof (2019).

[18] S. Balakrishnan, V. K, and K. A, “Intrusion Detection System Using Feature Selection and Classification Technique,” International Journal of Computer Science and Application (IJCSA) Volume 3 Issue 4, November 2014.

[19] S. Ramakrishnan1 and S. Devaraju, “Attack’s Feature Selection-Based Network Intrusion Detection System Using Fuzzy Control Language,” Taiwan Fuzzy Systems Association and Springer-Verlag Berlin Heidelberg (2016).

[20] P. Wu, N. Moustafa, S. Yang, and H. Guo, “Densely Connected Residual Network for Attack Recognition,” IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) (2020).

[21] S. Latif, Z. Idrees, Z. Zou, and J. Ahmad. Drann, “A deep random neural network model for intrusion detection in industrial iot,” International Conference on UK-China Emerging Technologies (UCET) IEEE (2020).

[22] P. Kumar, G. P. Gupta, and R. Tripathi, “An ensemble learning and fog-cloud architecture-driven cyber-attack detection framework for IoMT networks,” Computer Communications (2021).

[23] N. Moustafa, M. Keshky, E. Debiez and H. Janicke, “Federated TON IoT Windows Datasets for Evaluating AI-based Security Applications,” IEEE 19th International Conference on Trust, Security, and Privacy in Computing and Communications (2020).

[24] Z. Huang, C. Yang, X. Zhou, and T. Huang, “A hybrid feature selection method based on binary state transition algorithm and ReliefF,” IEEE Journal of Biomedical and Health Informatics (2018).