Detecting malicious IoT traffic using Machine Learning techniques

Bhuvana JAYARAMAN*, Mirnalinee THANGA NADAR THANGA THAI, Anirudh ANAND, Karthik Raja ANANDAN

Sri Sivasubramaniya Nadar College of Engineering, Chennai, Tamil Nadu, India

bhuvanaj@ssn.edu.in*, mirnalineett@ssn.edu.in, anirudh19015@cse.ssn.edu.in, karthikraja19048@cse.ssn.edu.in

***Corresponding author:** Bhuvana JAYARAMAN
bhuvanaj@ssn.edu.in

Abstract: Internet of Things (IoT) generates huge amount of data, that needs to communicate between the IoT enabled devices. These communications are vulnerable to security attacks and are malicious enough to cause harm to connected devices. The invasive communication and security breaches have to be identified and should be dealt with in order not to cause further damage and consequences. The objective of this work is to distinguish intentional communications from insecure communications between the IoT devices. The intentional communications can be different from the insecure communications in their patterns. Artificial intelligence-based machine learning approaches have the technologies to identify patterns of the intentional or insecure communications. In this paper, Random Forest, Decision Tree, SVM and 1DCNN have been used to discriminate patterns belonging to intended and unintended messages. To evaluate this technique, IoT-23 dataset is used, the proposed machine learning based approach obtaining a performance of 99.25% accuracy with the benchmark dataset. The proposed approach is compared with the state-of-the-art methods. It is observed that the proposed Random Forest method outperforms the existing ones with sufficient patterns to identify. To enhance the performance of the poorly performing classifiers on the imbalanced dataset, a potential solution to be applied on this dataset is also explored and proposed.

Keywords: Deep Learning, Exploratory Data Analysis, IoT Security, Machine Learning, Traffic Classification.

1. Introduction

Internet of Things: Evolution of the Internet with its emerging technologies has taken the world into Internet-of-Things (IoT) which has improved the quality of life and in turn the world economy. Almost 35 billion heterogeneous IoT devices have been connected to the network, there is an unprecedented increase in number of IoT devices at the rate of 5 billion per year. Physical objects are connected to each other in a timeless fashion that sense and control remotely via the network. IoT connects domain specific applications with domain independent services, where sensors and actuators communicate directly with each other to provide services. The six components of IoT (Kolisnyk, 2021) are identification, sensing and control, communications, computation, services and semantics. Ubiquitous codes (ucode) and electronic product codes (EPC) are generally used for the unique identification of connected physical objects. IoT sensors measure/ collect data and send it to the cloud or database to provide services via lossy and noisy communication channels. These connected devices run real-time operating systems throughout their activation to process the collected data. Semantics ensure the gathered data is provided to the correct resource to deliver the appropriate services.

Communication in IoT has started to attract attention when heterogeneous things are connected to the network. Communication usually uses different technologies, namely, radio frequency identification (RFID), wireless sensor network (WSN) with Wi-Fi, 3G, 4G, 5G internet supporting small to large networks (Burhan et al., 2018).

Eavesdropping, replay, time attack, Denial of Service (DoS), Man-in-The-Middle, storage space attack, cross site scripting, malicious code (Burhan et al., 2018), Malicious Insider Attack are some of the security attacks the IoT communication is facing. These attacks are being handled through various security solutions, namely cryptography hash-based solution, secure authorization, embedded security, identity-based management, intrusion detection systems and access control mechanisms.

https://doi.org/10.33436/v33i4y202304

1.1. Problem statement

Apart from these techniques for managing security threats to IoT devices, in order to maintain the performance, classification of network traffic becomes essential. Communication security and privacy of the shared data are the consequences of network traffic classification. Such traffic classification finds its application in IoT based smart environments, like smart cities, smart homes, smart cars etc.

1.2. Contributions to the advancement of the field

1. This work has proposed a system that automatically detects and identifies the IoT communications as benign and malicious.
2. Explored various machine learning and deep learning algorithms to find the suitable algorithm for IoT traffic classification.
3. Detailed analysis on the outcomes of the algorithms.
4. Proposed a solution to handle the data imbalance problem in IoT-23 dataset.

2. Background

Network traffic classification is essential for security purposes only and is necessary for monitoring, accounting, allotting resources to the network and availability of services as well. In general, classification of traffic can be performed through several methods, namely port based, statistical methods, behavioral based and payload-based methods. Both machine learning (Liu et al., 2021a) and deep learning-based approaches (Abdalgawad et al., 2021), (Shahraki et al., 2021), (Yue et al., 2021), are widely used using the mentioned methods. This work is based on behavioral classification approaches where the pattern of the traffic will be analyzed among the IoT devices. DDoS attacks, intrusions, abnormal or malicious activities are prone to happen in any IoT based networking communications.

The communication between IoT devices and cloud processors are vulnerable to possible attacks, namely DOS, Jamming, and Buffer Overflow. The attacks are classified using conventional machine learning algorithms (Fatayer et al., 2021), like learning Vectors Quantization (LVQ), Radial basis function (RBN) and Multilayer Perceptron (MLP). The classifiers identify the type of security attacks that happened on the communications. KDD Cup 99 dataset was used to evaluate the Artificial Neural Network (ANN) classification algorithms, where MLP got 99.86% of accuracy in shorter time.

A deep learning based malicious traffic identification with attention mechanism was proposed as (Liu et al., 2021b) Hierarchical Attention Gated Recurrent Unit (HAGRU). The used datasets are NSLKDD, CIC-IDS2017 and CSE-CIC-IDS2018, and learning samples are subjected to a series of preprocessing stages, namely, digitizing, normalization, sampling, data segmentation and processing of missing values. The pre-processed data goes into GRU, an attention layer and a MLP to detect malicious traffic. However, HAGRU fails to handle the unbalanced dataset.

Three supervised learning algorithms, namely SVM, XGBoost and Light Gradient Boosting Machine (LightGBM) were used to classify the (Bansal et al., 2017) subset of IoT-23 dataset. The training instances are fed to the classifiers, where the subset of IoT23 labelled by the authors as 21-1 has achieved 100% F1 score by all the three classifiers. Similar works are found in (Alhowaide et al., 2021; Khandait et al., 2021; Ullah et al., 2017;).

The following subsection discusses the various loss functions that contribute to the performance of the machine and deep learning approaches.

2.1. Loss functions

The performance of any machine learning algorithm depends on the loss function that helps update the weights and parameters. Loss functions used in binary classification and regression problems are log loss, hinge loss, Exponential loss, Mean square error, Mean absolute error, Huber loss (Wang et al., 2020).

Let I be the input space and J be the output space, L be the set of labels and is finite $\{L_1,L_2,...,L_n\} \subseteq J$. A supervised learning system will map I to J by applying a function f_L on every input sample I to be mapped to J. If the prediction system is a binary classification then the $L = \{L_1,L_2\}$ will be subset of L. $L_1 = 0$, $L_2 = 1$ constitute the output space.

For every I_{new}, the classifier applies the function $f_L(I,J)$ to predict a J_{new}, where $J_{new} = 1$ when $f_L(I,J) > 0$ otherwise $J_{new} = 0$ and is given by $f_L(I,J) \in RdX\{1,0\}$ or $f_L : I \rightarrow J$. If D is the dataset with n samples and given by $D = \{(I,J)\}n$. Let κ be the other parameters of the classifier, the mapping function can be represented as $f_L(I : \kappa)$.

The classifier predicts the output J_{new} which should be close to the actual J, the deviation between J_{new} and J will form the error. It's a measure that tells how good the model has learned from the data. Loss is calculated using a loss function or an error function, the scope of which is a single training sample. The average of the losses calculated over all the samples of the dataset is called the cost function. Any supervised learning algorithm will minimize this cost function with the help of an optimization function during the training.

With an array of classifiers in machine learning, a single loss function is not sufficient, instead it is selected on the basis of the type of classifier used, type of the underlying data, presence of outliers in the dataset, ease of computing the gradient. The two different classifications of loss are classification loss and regression loss.

2.2. Loss functions in binary classification

2.2.1. Binary Cross-Entropy / log loss

This loss function is commonly used in binary classification problems that measures the deviation between two probability distributions. When the difference is small, the prediction and the actual truth are similar, otherwise not. When the projected probability differs from the actual probability, the Binary Cross-Entropy increases, where the optimal value for cross entropy of a perfect model will be 0.

$$H(p) = -\frac{1}{N}\sum_{m=1}^{N} J\log\left(p(J)\right) + \left(1-J\right)\log\left(1-p(J)\right) \tag{1}$$

where J is the label and p(J) is the predicated probability of the sample being L_1 for all the samples N. It is assumed that $L_1 = 1$ and $L_2 = 0$ as the two labels. As per the Equation 1 for every sample with $L_1 = 1$, this expression adds log(p(J)) to the loss and for every sample with $L_2 = 0$, $\log(1- p(J))$ is added to the loss. This means that the probability of the expected value will be penalised according to the distance from the actual value. calculate the cross-entropy, as per the expression, wrong predictions are penalized and the probability of the correct prediction is 1, then the loss becomes 0 and vice versa.

2.2.2. Hinge log loss

Hinge loss is the second loss commonly used in classification, especially in SVM classifier for binary classification. It is based on the concept of maximum margin.

$$H\left(f_L(I,J)\right) = \max\left\{0,1-J\,f_L(I,J)\right\} \tag{2}$$

where $f_L(I,J)$ is the prediction made by the classification model. Hinge loss can be used when the target values are in the set 1, -1 and therefore 0 is mapped to -1. More errors will be accumulated when the sign of the actual and predicted values differ.

2.3. Loss functions in regression problems

2.3.1. Mean Square Error loss (MSE)

When the probability distribution of the predicted or dependent variable exhibits a Gaussian distribution, MSE is widely used. The cost function is computed by the square of the difference between actual and predicted values.

$$H\left(f_L(I,J)\right)=\frac{1}{N}\sum_1^N\left(J-f_L(I,J)\right)^2 \tag{3}$$

N is the size of each batch. Squaring operation maintains the positive outcome. This loss will penalize the classifier for larger errors with large errors and hence more significance to the outliers. Due to the quadratic nature of the loss function, it has one global minimum and no local minimum.

A variant of this function is Mean Squared Logarithmic Error (MSLE), which avoids penalizing the classifier by a large error. This function takes the logarithmic value of the predicted and the actual values and calculates the square of their difference for every sample.

$$H\left(f_L(I,J)\right)=\frac{1}{N}\sum_1^N\left(\log(J+1)-\log\left(f\_L(I,J)+1\right)\right)^2 \tag{4}$$

2.3.2. Mean Absolute Error loss (MAE)

An alternative loss function that calculates the absolute difference between expected and actual values, which is frequently used in regression problems in MAE.

$$H\left(f_L(I,J)\right)=\frac{1}{N}\sum_1^N\left|J-f\_L(I,J)\right|^2 \tag{5}$$

MAE is more powerful and better than MSE, but has local minima and the gradient stays high throughout the training. This does not allow it to converge and therefore may need a dynamic learning rate.

2.3.3. Huber loss (Smooth Mean Absolute Error)

Huber loss combines the calculation of both MSE and MAE. A threshold or limit value decides whether to use MSE or MAE. When the error is less than the threshold E, MSE will be used.

Huber loss is not sensitive to outliers and does not have local minima. But it needs the additional parameter ϵ to be optimized.

$$H\left(f_L(I,J)\right)=\begin{cases}\frac{1}{N}\left(J-f_L(I,J)\right)^2 & for\ \left|J-f_L(I,J)\right|\le\varepsilon\\ otherwise & \\ \varepsilon\left(\left|J-f_L(I,J)\right|-\frac{1}{2}\varepsilon\right)^2 & \end{cases} \tag{6}$$

3. Proposed system

The proposed system is intended to identify communications between the IoT devices as benign or malicious. Traffic packets with the necessary information will be used as features to identify the patterns and classify them. This work used both machine-learning and deep learning classifiers to the IoT traffic classification. Decision tree, Random Forest, SVM and Convolution Neural Network (CNN) are used as binary classifiers to predict the communication. IoT communications made available as dataset undergo several stages of pre-processing steps to generate and prepare the salient feature vectors. The feature vectors are then learned by four classifiers, namely Decision tree, Random Forest, SVM and CNN. The best acceptable classifier for identifying secure IoT communications is identified after the models have been evaluated for their performance using evaluation measures. The overview of the proposed system is shown in Figure 1.

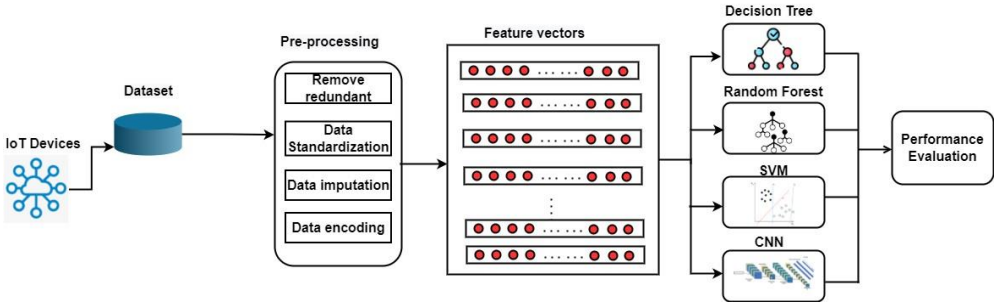

Figure 1. Overview of proposed system

3.1. Pre-processing of the inputs

The csv files were aggregated into 1 common data frame. The data frame was inspected using preliminary data-processing techniques. Columns and rows that were redundant were deleted. The data frame was standardized by converting all non-values to NaN. The missing values were extrapolated from known data. For example, the missing values in the 'duration' were replaced by the most frequently reported time, while missing values in the 'service' column were replaced with a new category. The labels were renamed as either 'Malicious' or 'Benign'. Columns with categorical data were replaced with dummy codes using the *get dummies* function. Finally, the non-numeric data were encoded using Python's Label Encoder function.

An objective and representative sample is obtained from a broader population by using the statistical sampling approach known as stratified sampling. This approach separates the population into strata, or subgroups, according to specific characteristics or features that are essential to current research or study. To choose a sample, a random or systematic sampling technique is used within each stratum, treating each as a separate and distinct population. Since each subgroup is guaranteed sufficient representation in the final sample, this method is especially helpful when there is a great deal of heterogeneity within the population. It was observed that the proportion of benign and malicious data were imbalanced. Hence stratified sampling was used in this work. 70% and 30% of the data were used to train and test the model respectively.

The choice of Random Forest, Decision Tree, SVM and 1D CNN is based on their ability to handle pattern recognition, their diverse learning approaches, their suitability for high-dimensional and potentially non-linear data and their proven track record in similar classification tasks. Because of their distinct advantages, each of these methods are used to classify IoT communications. SVM effectively handles high-dimensional and non-linear data, Random Forest and Decision Trees offer interpretability and robustness, while 1D CNN is ideally suited for sequential data with complicated patterns. The particulars of the IoT dataset and the trade-offs between interpretability, computation involved and accuracy have also influenced the choice of algorithm.

3.2. Convolution Neural Networks (CNN)

Convolution Neural Networks are mostly used for 2D and 3D data objects. Here we have used a 1D CNN to determine its efficiency in classifying a row of data to one of 2 classes. A CNN consists of 3 layers namely CNN layer, Pooling layer, flatten layer. The feature map is passed to the following layer by the convolution layer once it has learned the patterns from the input data. Pooling layer reduces the dimensions of the input, so that less but significant data is available to be flattened. Flatten layer is used to convert multidimensional feature map from the output of the previous layer to a single dimensional vector. Here a 1D CNN with 64 filters of size two and a maxpool layer of size two were used. The output of this layer is given to a dense layer whose output is sent to sigmoid activation for classification. The loss function used is the binary cross entropy, instead of other loss functions which improved accuracy by 20%. The Adam optimizer is used to optimize the loss function, and the classifier is trained for 20 epochs. Protocols and service columns are converted into one hot encoded set of columns using get dummies function, with which the accuracy grew up to 93%. Without this change, the accuracy reduces back to 78%. We can clearly see that splitting into more columns gives more accuracy, i.e. giving each category a separate node gives better accuracy than giving multiple labels through a single node.

3.3. Decision tree

The interior nodes of the decision tree algorithm are decision nodes, while the leaf nodes are classifier nodes, which makes it a tree-structured classifier. The decision rules are represented as branches and outcomes are represented as each leaf node. The edges descending from the tree's nodes represent the potential classes to which the instance may belong, and each node serves as a test case for an attribute. Every subtree placed at the new node goes through this recursive procedure once again. Essentially, the algorithm learns to make efficient splits so that the leaf nodes contain negligible amounts of impurity, i.e., minimal entropy.

However, it is important that the leaf nodes have small amounts of impurities, as otherwise there may be signs of model over-fitting the dataset. The problem of over-fitting can be resolved by pruning the tree when the threshold impurity is reached.

3.3.1. Training the decision tree classifier

Stratified sampling was used to account for the imbalance in the training dataset. 70% and 30% of the data were used to train and test the model respectively. The model achieved an accuracy of 98.79% and an F1-score of 99.24%.

3.4. Random Forest

Random Forest, a supervised learning method, builds a forest out of a group of decision trees that were typically trained using the bagging technique. The main idea behind the bagging method is integrating the learning models to enhance the outcome. The random forest algorithm classifies an entity into a particular category based on the final inference of the majority of decision trees. The internal nodes of each decision tree are called decision nodes, while the leaf nodes are the classification nodes. The trees in the Random Forest branch out in such a way that the information gained is maximum (or) the decrease in entropy is the highest. The model has an accuracy of 98.8% and precision of 99.3.

3.5. Support Vector Machine (SVM)

A most common and widely used machine learning algorithm, Support Vector Machine (SVM), draws a hyperplane as a decision boundary in a N dimensional feature space to classify them into classes. The different kernel functions of SVM define the similarity of the features points and transform them, so that a hyper plane can be drawn to separate them. When there are many hyperplanes, the SVM chooses a hyperplane with maximum margin. SVM is applied with various kernels namely rbf, poly, linear and sigmoid to study the performance of the approach on the dataset. The regularization parameter is fixed as 1, with kernel coefficient gamma set to 1 / (n features * X). The other parameter is 1e−3 as tolerance to stop the iterations of SVM without any limitation on the maximum number of iterations.

4. Performance and discussion

4.1. Dataset and its insight

This work has used the IoT-23 dataset released by Stratosphere Laboratory, CTU University, Czech Republic (Parmisano et al., 2020) that has network traffic collected from various IoT devices. There are totally 23 traffic captures or Scenarios available from the communication of Internet of Things, which are real-time traffic communications with the labels benign or malicious. Out of 23 captures, 20 are captured from malware infected IoT devices and the remaining 3 are collected from benign IoT devices.

The dataset has 500 hours of traffic with 325 million flows that are annotated and has 760 million packets taken during 2018 and 2019 in the lab. The 20 scenarios of malicious include the varieties of malware that are tabulated in Table 1.

Table 1. Various malware families of IoT-23 dataset

Malware	Description
Mirai	Converts Linux based system into bots, killing processes to TCP
Torii	A sophisticated attack on IoT devices on unexpected levels
Gagfyt	Exploits shell vulnerabilities, targets the embedded systems
Kenjiro	Variant of Hakai Malware
Okiru	Exploits embedded devices with ARC processors
Hakai	DDOS malware that affects the routers
IRC Bot	Trojan that uses IRC servers and access MSN messenger contacts
Hajime	Creates peer to peer botnets, targeting several CPU architectures
Muhstik	DDOs malware that mines cryptocurrencies
Hide & seek	Similar to worms and assigns random IP addresses for victims

The dataset has 21 features or columns, 17 out of them are categorical in nature and 4 are numerical types. The features include time of capture, its identity, IP address of the compromised devices, its port number, IP address of the device from where the data is captured, duration of the attack, application, network protocol to which the attack is associated with, the quantum of data sent and received in terms of packets and bytes, connection state, its history, its origin and response's origin, bytes missing in packets, along with the label as malicious or benign and if the scenario is malicious its type is present in that learning instance.

An exploratory data analysis is performed on the IoT-23 dataset to get the insights and relationship between the features of the communication. From the correlation heap map on the features of the dataset shown in Figure 2, it can be observed that there exists a poor correlation between the features like id orig ip, id resp ip, missed bytes, orig pkts. They can be seen in darker shade in the correlation map, with very less to no correlation among themselves. Resp pkts, resp ip bytes can be seen in lighter shades with high correlation among them. These characteristics of the features will help to analyze what features will contribute or will not contribute to the decision making of the classifiers.

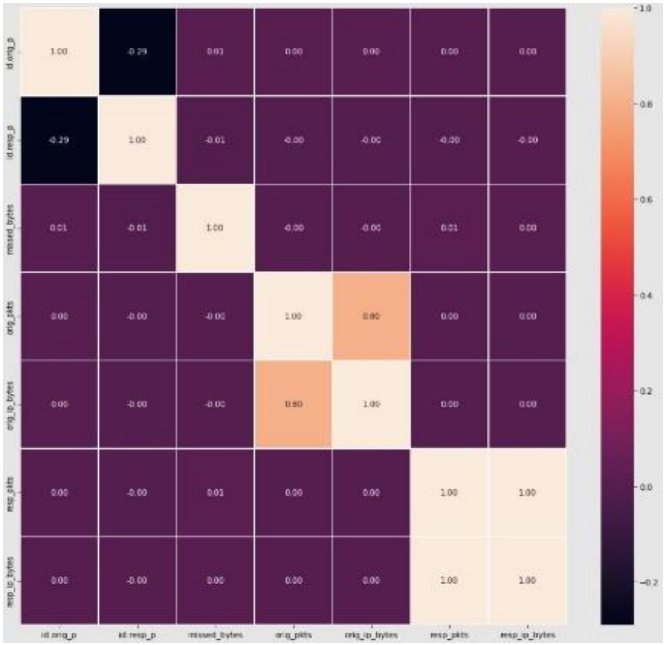

Figure 2. Correlation heat map of the features of IoT-23 dataset

The histogram of the samples of benign and malicious class labels are shown in Figure 3. Class A refers to malicious samples and class B refers to benign samples. It can be observed that the class samples are not balanced, which may affect the performance of the models that classify them.

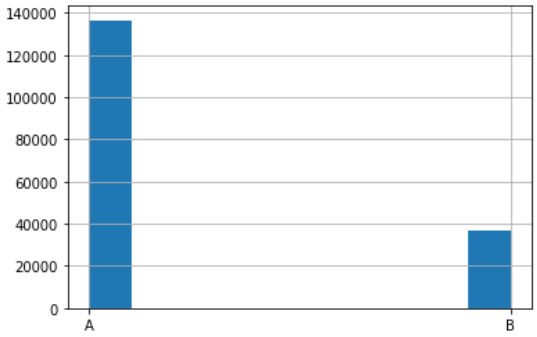

Figure 3. Histogram of benign and malicious classes of IoT-23 dataset

4.2. Performance metrics

All the algorithms in this work are trained with the same set of instances, and their performance is observed on the basis of evaluation metrics, namely accuracy, Recall, Precision and F1 score along with False Positive Rate (FPR), False Negative Rate (FNR) and False Discovery Rate (FDR). From the F1 scores obtained from the four classifiers, it can be observed that Random Forest has outperformed well compared to the other 3 classifiers. Decision Tree has got the second best F1 score. Among the four models SVM failed to identify the IoT traffic as benign and malicious.

The inherent nature of Random Forest works well with all possible type of features, namely categorical, numerical or binary features without any necessary pre-processing. The features are not necessarily transformed or scaled. It has also handled the outliers in the dataset of IoT-23. The nature of the random forest has tackled the class imbalance problem and has built-in techniques to reduce the overall error rate. Because it averages over several trees, the problem of overfitting has also been reduced. These have proved that Random Forest is a more robust model than decision tree for classifying the IoT communications into benign and malicious. SVM fails to perform due to the class imbalance problem and presence of outliers as shown from Table 2.

Table 2. Performance metrics

Classifier	Accuracy	F1	Precision	Recall	Specificity
Decision Tree	0.9879	0.9924	0.9894	0.9953	0.9605
CNN 1D	0.9390	0.9627	0.9301	0.9976	0.9172
Random Forest	0.9891	0.9931	0.9936	0.9926	0.9762
SVM	0.7887	0.8819	0.7888	1.0	-

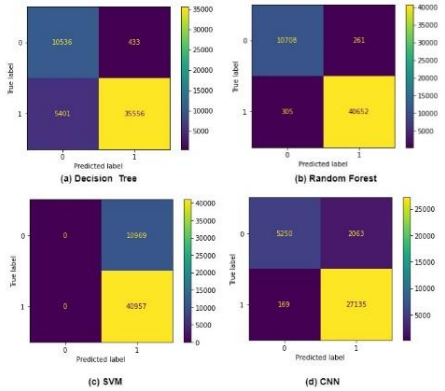

Figure 4. Confusion matrix of all classifiers

The confusion matrix of all the four models is shown in Figure 4. The confusion matrix of the random forest shows that the number of both false positives and false negatives is very small compared to the other classifiers, and SVM can be seen with poor classification. From Table 3, the Random Forest had the lowest false discovery rate (FDR) of 0.0063, showing that the Random Forest classifier identified 0,6% of the malicious traffic as benign. False positive rate (FPR) has also been the lowest of all the other classifiers. The performance metrics is visualized in the Figure 5, showing the near equal performance of the classifiers used to identify the IoT communications.

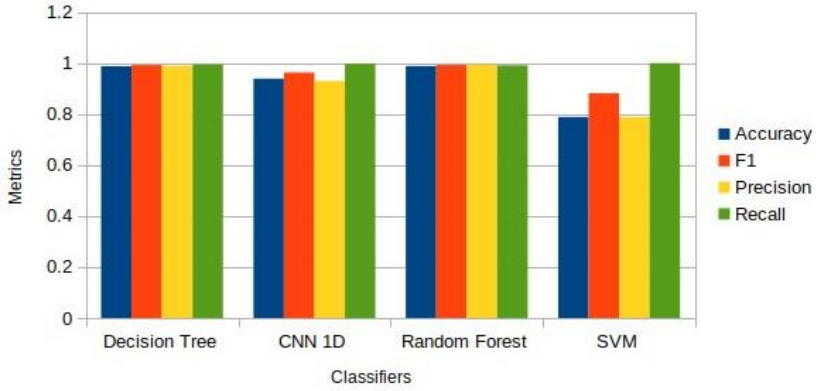

Figure 5. Metrics of all classifiers

Table 3. Other related metrics

Classifier	FPR	FNR	FDR	MCC
Decision Tree	0.0395	0.1318	0.0120	0.7333
CNN 1D	0.2821	0.0061	0.0706	0.8025
Random Forest	0.0237	0.0074	0.0063	0.9673
SVM	1.0	0.0	0.2112	0.0

Matthews Correlation Coefficient (MCC) is also calculated for all the binary classifiers that measure the correlation coefficient between the predicted and the true class Eqn. 7. The higher the coefficient, the higher the correlation and hence the better the prediction of the classifier.

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP) + (TP + FN) + (TN + FP) + (TN + FN)}} \tag{7}$$

Random forest bindings between predicted and the true class tops the chart supporting the other metrics calculated, where CNN has the second highest MCC score than the usual metrics like accuracy, F1, recall and precision.

4.3. Comparison of proposed with existing works

The proposed system for traffic classification is compared with similar works on the same IoT-23 dataset and is shown in Table 4. Almost all the works on this dataset have reported near 100% performance on F1 score. (Bansal et al., 2017) used only a subset of IoT-23 and has achieved a F1 score of 100% and (Austin et al., 2021) has obtained 97.3% using Random Forest and 92.35% using Linear SVM and observed no preprocessing was done on the features of the dataset. As similar to this work, SVM (Austin et al., 2021) has registered a lesser performance than other models.

Table 4. Comparison of proposed with existing works

Similar work	Algorithm	F1 Score
(Bansal et al., 2017)	XGBoost	100
(Austin et al., 2021)	Random forest	97.3%
(Austin et al., 2021)	Linear SVM	92.35%

4.4. Observations and solution to handle poor performance of classifiers

Among the machine learning and deep learning algorithms used to classify the IoT traffic for malicious and benign communications, the following behaviours were observed:

1. Random Forest, with its versatile nature of algorithm design, handled the data with or without data pre-processing. As an ensemble learner, random forest use several weak learners as decision trees and bring together the prominent features. Hence the class imbalance has no effect on the performance of the model.

2. The Decision Tree has classified the communications and was observed as the second-best algorithm after the Random Forest classifier with CNN serving as the third best approach.

3. SVM failed to produce a better performance, more false positives and false negatives were observed, which may not be suitable for a dataset like IoT-23, which has several outliers, a poor correlations coefficient among the features and, prominently, an imbalanced dataset.

The classifier that learns on an imbalanced data set, will fail to learn the data from the class with lesser samples. When such a model is tested, it will only predict the sample as being one of the majority classes since it has learnt the minor class as noise, which may lead to more false positives and false negatives (Hasanin et al., 2019a; Hasanin et al., 2019b; Hasanin et al., 2020). This kind of behaviour is unacceptable in real time applications. Hence, the following solutions are recommended to handle the class imbalance data:

1. Data level approaches.

2. Algorithm level approaches.

3. Hybrid approaches.

Data-level approaches will be more suitable for the IoT-23 dataset. One particular type of data level techniques is random under-sampling, which reduces the number of samples from the majority class, is a viable solution to handle class imbalance problem of IoT-23 dataset. The other data level approach, random over-sampling, which repeats the data from the minority class, may not be the solution to improve the performance of the classifiers, since it may lead to model overfitting and cause poor generalization.

5. Conclusion

Due to the advancement of technologies and their adoption at a faster pace, they not only increase productivity but also have their vulnerabilities. In this paper, the machine learning techniques to detect the communications between the IoT devices and classified them into IoT benign and IoT malicious insecure traffic were explored. After preprocessing, the learning samples of IoT-23 dataset are modeled by decision tree, random forest, SVM and CNN classifiers. The Random Forest has outperformed in detecting the secure and insecure traffic with 99.31% F1 score, when compared to SVM, decision tree and CNN. SVM was observed to be the least performing classifier, since it is sensitive to the outliers, has poor correlation between the features and mainly due to the class imbalance problem. The solution to overcome this class imbalance problem has also been proposed, which can be incorporated as the extension of this work to enhance the performance of the other classifiers. Unsupervised representational learning can also be attempted to build a model that discriminates the malicious from the secure communications of IoT devices.

REFERENCES

Abdalgawad, N., Sajun, A., Kaddoura, Y., Zualkernan, I. A. & Aloul, F. (2021) Generative deep learning to detect cyberattacks for the IoT-23 dataset. *IEEE Access*. 10, pp. 6430-6441. doi:10.1109/ACCESS.2021.3140015.

Alhowaide, A., Alsmadi, I. & Tang, J. (2021) Towards the design of real-time autonomous IoT NIDS. *Cluster Computing*. 26(1), 1-14.doi:10.1007/s10586-021-03231-5.

Austin, M. (2021) *IoT Malicious Traffic Classification Using Machine Learning*. Graduate Theses, Dissertations, and Problem Reports. 8024.West Virginia University.

Bansal, A. & Mahapatra, S. (2017) A comparative analysis of machine learning techniques for botnet detection. In *Proceedings of the 10th International Conference on Security of information and Networks*. pp. 91-98. doi:10.1145/3136825.3136874.

Burhan, M., Rehman, R. A., Khan, B. & Kim, B. S. (2018) IoT elements, layered architectures and security issues: A comprehensive survey. *Sensors*. 18(9), 2796.doi:10.3390/s18092796.

Fatayer, T. S. & Azara, M. N. (2019) IoT secure communication using ANN classification algorithms. In *2019 International Conference on Promising Electronic Technologies (ICPET), 23-24 October 2019, Gaza, Palestine, IEEE.* pp. 142-146. doi: 10.1109/ICPET.2019.00033.

Hasanin, T., Khoshgoftaar, T. M., Leevy, J. L. & Bauder, R. A. (2019a) Severely imbalanced big data challenges: investigating data sampling approaches. *Journal of Big Data*. 6(1), 107. doi:10.1186/s40537-019-0274-4.

Hasanin, T., Khoshgoftaar, T. M., Leevy, J. L. & Seliya, N. (2019b) Investigating random undersampling and feature selection on bioinformatics big data. In *2019 IEEE Fifth International Conference on Big Data Computing Service and Applications (Big Data Service), 04-09 April 2019, Newark, CA, USA, IEEE.* pp. 346-356. doi: 10.1109/BigDataService.2019.00063.

Hasanin, T., Khoshgoftaar, T. M., Leevy, J. L. & Bauder, R. A. (2020) Investigating class rarity in big data. *Journal of Big Data*. 7(1), 1-17. doi:10.1186/s40537-020-00301-0.

Khandait, P., Hubballi, N. & Mazumdar, B. (2021) IoTHunter: IoT network traffic classification using device specific keywords. *IET Networks*. 10(2), 59-75. doi:10.1049/ntw2.12007.

Kolisnyk, M. (2021) Vulnerability analysis and method of selection of communication protocols for information transfer in Internet of Things systems. *Radioelectronic and computer systems*. 1, 133-149.doi:10.32620/reks.2021.1.12.

Liu, D., Xu, X., Liu, M. & Liu, Y. (2021a) Dynamic traffic classification algorithm and simulation of energy Internet of things based on machine learning. *Neural Computing and Applications*. 33, 3967-3976. doi:10.1007/s00521-020-05457-7.

Liu, X. & Liu, J. (2021b) Malicious traffic detection combined deep neural network with hierarchical attention mechanism. *Scientific Reports*. 11(1), 12363. doi:10.1038/s41598-021-91805-z.

Parmisano, A., Garcia, S. & Erquiaga, M. J. (2020) *A labeled dataset with malicious and benign IoT network traffic*. Stratosphere Laboratory: Praha, Czech Republic.

Shahraki, A., Abbasi, M., Taherkordi, A. & Kaosar, M. (2021) Internet traffic classification using an ensemble of deep convolutional neural networks. *In Proceedings of the 4th FlexNets Workshop on Flexible Networks Artificial Intelligence Supported Network Flexibility and Agility*. Association for Computing Machinery, New York, NY, United States, 2021. pp. 38-43. doi: 10.1145/ 3472735.3473386.

Ullah, I. & Mahmoud, Q.H. (2021) Network traffic flow based machine learning technique for IoT device identification. In *2021 IEEE International Systems Conference (SysCon), 15 April 2021 - 15 May 2021, Vancouver, BC, Canada. IEEE.* pp. 1-8. doi: 10.1109/SysCon48628.2021.9447099.

Wang, Q., Ma, Y., Zhao, K. & Tian, Y. (2022) A Comprehensive Survey of Loss Functions in Machine Learning. *Annals of Data Science*. 9, 187-212.doi:10.1007/s40745-020-00253-5.

Yue, Y., Li, S., Legg, P. & Li, F. (2021) Deep learning-based security behaviour analysis in IoT environments: A survey. *Security and communication Networks*.1-13. doi:10.1155/2021/8873195.

Bhuvana JAYARAMAN is an Associate Professor in the Department of Computer Science and Engineering with 22 years of experience in teaching. Before joining SSN College of Engineering in 2006, she worked as an Assistant professor in AVC College of Engineering for 8 years. She received her Ph.D. from Anna University, Chennai in 2015, with master's degree, M.E. in CSE from Annamalai University, Chidambaram in 2004, with First class and Distinction. She completed B.E. in CSE from the University of Madras in 1998. Her research interests include Deep Learning, Multiobjective Optimization, Memetic Algorithms, Evolutionary Algorithms, Machine Learning.

Mirnalinee THANGA NADAR THANGA THAI is a Professor at SSN College of Engineering, Chennai, India, and is currently the head of the department of Computer Science and Engineering. She received her B.E. degree from Bharathidasan University, Trichy, M.E. degree from the College of Engineering, Guindy, Anna University, Chennai, and Ph.D. from Indian Institute of Technology Madras (IITM), Chennai, India. Her research interests include Computer Vision, Machine Learning, Green Networks and Software Defined Networks. Seven research scholars have completed PhD under her supervision, and she is currently guiding seven more scholars. She has completed three research projects and has published about 80 papers in international journals and conferences. She has reviewed several papers in international journals and chaired several sessions in conferences.

Anirudh ANAND is a skilled and experienced fresher in ML and full-stack technologies. He recently completed his bachelor's degree in Computer Science and Engineering at SSN College of Engineering, Chennai, India. He has worked on ML, CV, NLP tasks, and in particular been interested in and working on differential Privacy.

Karthik Raja ANANDAN, a skilled fresher in ML and full-stack technologies, recently completed his Bachelor's degree in Computer Science and Engineering at SSN College of Engineering, Chennai, India. He is passionate about using technology to solve real-world problems and have used ML, CV and NLP to develop solutions for various problems. He is greatly interested in LLMs.