

Major Project Report On

Performance Comparison of Various Machine Learning Techniques for Data Oriented and Packet Oriented Dataset

Submitted by

Nilita Anil Kumar (16IT122)

Nandini AV (16IT120)

VIII SEM B.Tech (IT)

Under the guidance of

Dr. Bhawana Rudra

Dept of IT, NITK Surathkal

in partial fulfillment for the award of the degree

of

Bachelor of Technology

in

Information Technology

at



**Department of Information Technology
National Institute of Technology Karnataka, Surathkal.**

June 2020

CERTIFICATE

This is to certify that the project entitled **“Performance Comparison of Various Machine Learning Techniques for Data oriented and Packet oriented dataset”** is a bonafide work carried out by **Nandini AV and Nilita Anil Kumar**, student of **Fourth** year B.Tech (I.T), Department of Information Technology, National Institute of Technology Karnataka, Surathkal, during the **8th** term of the academic year 2019 – 2020. It is submitted to the Department in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology. I certify that they have carried out the work in his/her own capacity and has successfully completed the work assigned to them. The aforementioned work was carried out during the period of **January 2020 to April 2020**

Place: Surathkal

Date:

(Signature of the Guide/Mentor
with Institution seal)

Abstract

The Internet of Things (IoT) paradigm has recently evolved into a technology for building smart environments. Security and privacy are considered key issues in any real-world smart environment based on the IoT model. The security vulnerabilities in IoT-based systems create security threats that affect smart environment applications. Thus, there is a crucial need for intrusion detection systems (IDSs) designed for IoT environments to mitigate IoT-related security attacks that exploit some of these security vulnerabilities. We proposed a Hybrid Intrusion Detection System which is a combination of Host Based IDS (HIDS) and Network Based IDS (NIDS). The system can be defined for anomaly detection and malware detection. To demonstrate the feasibility of the proposed MIB-IoT, we applied various classification algorithms- supervised and unsupervised algorithms on data oriented dataset and packet oriented dataset respectively. The experiment results showed that SVM and Fuzzy C-means performed the best compared to other algorithms.

Table Of Figures

Title	Page
Fig 4.1.1 Temperature Distribution	18
Fig 4.1.2 Humidity Distribution	18
Fig 4.1.3 Light Distribution	18
Fig 4.1.4 Voltage Distribution	18
Fig 4.2.1.1 Elbow curve for Kmeans	20
Fig 4.2.3.1 Dendrogram	23
Fig 5.1 PCA	25
Fig 5.2 Elbow Curve	25
Fig 5.3 K distance plot	26
Fig 5.4 Dendrogram	27
Fig 5.5 FPC for different cluster numbers	28
Fig 5.6 Graph for FPC for different cluster numbers	28
Fig 5.7 FPC for different cluster numbers for over-sampled data	28
Fig 5.8 Graph for FPC for different cluster numbers for over-sampled data	28

Table of Contents

1. Introduction	2
1.1 Motivation	2
2. Literature Survey	4
2.3 Problem Statement	7
2.4 Objective	7
3. Methodology and Framework	8
3.1 Architecture	8
4. Implementation	10
4.1 Dataset	10
4.1.1 Dataset used for HIDS	10
4.1.2 Dataset used in NIDS	12
4.1.3 Interquartile Rule for Outliers	14
4.2 Algorithms	15
4.2.1 Algorithms used for HIDS	15
4.2.1.1 K- Means Algorithm	15
4.2.1.2 Fuzzy c-means (FCM)	16
4.2.2 Algorithms used for NIDS	17
4.2.2.1 Naive Bayes	17
4.2.2.2 Support Vector Machine (SVM)	17
4.2.2.3 Logistic Regression	18
4.2.2.4 KNN	18
4.4 Up Sample and Down Sample	19
5. Results and Analysis	20
5.1 Network Based IDS	20
5.2 Host Based IDS	22
6. Conclusion and Future work	24
References	25

1. Introduction

One of the goals of smart environments is to improve the quality of human life in terms of comfort and efficiency. The Internet of Things (IoT) paradigm has recently evolved into a technology for building smart environments. The systems involved in the Internet of Things (IoT) network typically consist of numerous devices equipped with a variety of computing, power, sensor, and communication capabilities. Security and privacy are considered key issues in any real-world smart environment based on the IoT model. The security vulnerabilities in IoT-based systems create security threats that affect smart environment applications. Thus, there is a crucial need for intrusion detection systems (IDSs) designed for IoT environments to mitigate IoT-related security attacks that exploit some of these security vulnerabilities. Due to the limited computing and storage capabilities of IoT devices and the specific protocols used, conventional IDSs may not be an option for IoT environments. Machine learning can be used in IDS, but due to the volatile nature of the IoT devices or, it is very difficult to conceptualize a universally working machine learning (ML) model across various IoT devices [1]. The features used in models and their behaviors can dynamically be changed with devices and applications. To address this issue, we must consider the use of profiles to represent various IoT devices and applications as opposed to building a single universal ML model [2].

To show the feasibility of the proposed idea, we evaluated the performances of various machine learning algorithms with datasets. We propose a host based IDS to detect the anomaly by using unsupervised algorithm on dataset consisting of a normal operation dataset and a hardware fault dataset [5]; We also examine performance of various ML classification algorithms for attack detection[6] on Bot-IoT Dataset.

1.1 Motivation

A smart environment that integrates IoT technology is considered to be a complex system because it consists of different products from different companies based on different technologies that do not share a universal language. Therefore, standardization is another important aspect of security in IoT systems. Creating a standard IoT architecture based on one standard technology for all vendors and manufacturers would enhance the

interoperability of the security functionalities of all objects and sensors in an IoT system. Such standardization will greatly facilitate IoT network security.

A single successful penetration of one or more end devices can threaten the security of an entire IoT system and cause harm to its applications and services, especially from an industrial point of view. Thus, the implementation of a strong security mechanism in an IoT system depends on the strength of the security for individual IoT devices.

Due to the volatile nature of the IOT system, anomaly data can change in different environments. So, we have created a Host-Based Intrusion Detection System to find out the anomaly in data captured by the sensor using clustering methods. Also, incorporated Network Based Intrusion Detection System using supervised algorithms to detect malicious data.

2. Literature Survey

In [10], Roman Fernando and Paul Stacey, they have discussed a novel approach to security detection using streaming data analytics to classify and detect security threats in their early stages. The volatile nature of the IoT environment makes discovery difficult. Within an IoT system, there may be a high degree of volatile behavior. This volatility merely represents the digital artifacts of a chaotic physical world augmented with sensing and communication technologies. The diagnostic system proposed here is developed to identify abrupt changes in the individual features and their ever-changing dependencies. The exploratory study described here uses a Spatio-temporal methodology to characterize network behaviors. Once characterised, anomalous behaviours can be identified by calculating a similarity/distance metric to previously identified behaviours (e.g. attacks, intrusions or malfunctioning machinery).

In [16] The proposed system generates profiles of normal applications using a multi-modal probability distribution. The proposed classification framework is then extended to detect distributed denial of service attacks from the collected statistical information at the flow level. To demonstrate the feasibility of the proposed system, they evaluate its performance using five real-world traffic datasets. The experiment results show that the proposed method is capable of achieving an accuracy of over 97%, whereas the misclassification rate is only 2.5%. After collecting the data, the proposed framework extracts packet-level features. Packet-level features are used to generate fingerprints for traffic applications. Here, they utilize a histogram-based fingerprinting approach. Then, the fingerprints (transformed dataset) are clustered using an unsupervised approach, and the output clusters are mapped to their respective application traffic with assistance from the labeled training data.

In [14], it discussed abnormal behavior detection for IoT applications. IoT smart sensors, which are equipped with high computing power and communication capability; moreover, it is possible that one sensed data can be modified instead of all sensed data (e.g., a sensor reporting temperature, humidity, light, and voltage) for malicious purposes. It affects the detection accuracy of abnormal behavior and the degradation of abnormal behavior detection from both machine learning algorithms, such as the k-Means algorithm and support vector

machine (SVM), was observed. The k-Means algorithm and SVM were used to detect one sensed data modification out of 4 possible data points from one sensor and the results demonstrated that the k-Means algorithm (92%) had less affection than the SVM (69.5%) in terms of detection accuracy.

In [7], Yair Meidan et al address the challenge of IoT device identification within a network by analyzing and classifying network traffic data. Even if the prefixes of MAC addresses can be used to identify the manufacturer of a particular device, there is no standard to identify brands or types of devices. However, high-level network statistics, such as the “ratio between incoming and outgoing bytes” and the “average time to live” have been used to identify malicious traffic in network communications and have proposed to use such network features to identify IoT devices.

In [13], “Profiling-based classification algorithms for security applications in Internet of Things” , Eunil seo et al propose a “Management Information Base for IoT (MIB-IoT)” by extending conventional MIB to a more generalized structure in order to represent not only the structured properties of network objects but also the best machine learning model for each network object in a systematic fashion. MIB-IoT profiles can be defined for various applications such as abnormal behavior detection, malicious behavior detection, and even data source identification. To demonstrate the feasibility of the proposed MIB-IoT, they apply various classification algorithms on datasets consisting of normal operation data and hardware fault data and also, a hybrid of the real modern normal and the contemporary synthesized attack activities of the network traffic. The experiment results show that the classification algorithm using MIB-IoT is capable of achieving an accuracy of 99.81% for malicious behavior detection and 78.51% accuracy for categorising normal and hardware data.

A short comparison between anomaly IDS techniques with a focus on the advantages and disadvantages of each technique.

Technique	Advantages	Disadvantages
Data mining[24]	1- Models are created Automatically 2- Applicable in different environments 3- Suitable for online datasets	1- Based on historical Data 2- Depends on complex algorithms
Machine learning [25]	1- High detection Accuracy 2- Suitable for massive data volumes	1- Requires training Data 2- Long training time
Statistical model[26]	1- Suitable for online Datasets 2- System simplicity	1- Based on historical Behavior 2- Detection accuracy depends on statistical and mathematical operations
Rule model [27]	1- Suitable for online Datasets 2- System simplicity	1- Based on a set of Rules 2- High false positive rate
Payload model [28]	1- High detection accuracy for known attacks	1- Privacy issues 2- Long processing time
Protocol model [29]	1- High detection accuracy for a specific type of attack	1- Designed for a specific type of protocol
Signal processing Model [30] [31]	1- High detection accuracy	1- Depends on complex pattern-recognition Methods 2- Low false positive rate

2.3 Problem Statement

The security vulnerabilities in IoT-based systems create security threats that affect smart environment applications. Thus, there is a crucial need for intrusion detection systems (IDSs) designed for IoT environments to mitigate IoT-related security attacks that exploit some of these security vulnerabilities. Most existing systems for Identification of security breaches make use of network traffic analysis but that is not sufficient in case of IOT Systems. We intend to develop a system system which can address the security vulnerabilities due to the volatile nature of Iot along with the traditional attacks.

2.4 Objective

The major objective of the project is to harness the properties of unsupervised and supervised learning to detect attack and anomalies. We use this analysis to further suggest which machine learning model is best suitable for Network based IDS and Host Based IDS.

3. Methodology and Framework

3.1 Architecture

We aim to build an integrated IoT Abnormal behavior detection system system based on machine learning. Considering the Volatile behavior of the Iot System it is not possible to use only one kind of data set, to ensure the complete security of the system is necessary to do analysis at network layer and Data Layer. The security vulnerabilities in IoT-based systems create security threats that affect smart environment applications. Thus, there is a crucial need for intrusion detection systems (IDSs) designed for IoT environments to mitigate IoT-related security attacks that exploit some of these security vulnerabilities. Due to the limited computing and storage capabilities of IoT devices and the specific protocols used, conventional IDSs may not be an option for IoT environments

The basic architecture of the Iot system has multiple layer, we require different approaches to address the issues in different layer FIG 3.1.1. The security challenges in IoT systems are related to

security issues arising in the different IoT layers. Physical damage, hardware failure, and power limitations are challenges faced in the physical layer. DoS attacks, sniffing, gateway attacks, and unauthorized access are challenges relevant to the network layer. Malicious code attacks, application vulnerabilities, and software bugs are challenges faced in the application layer [22].

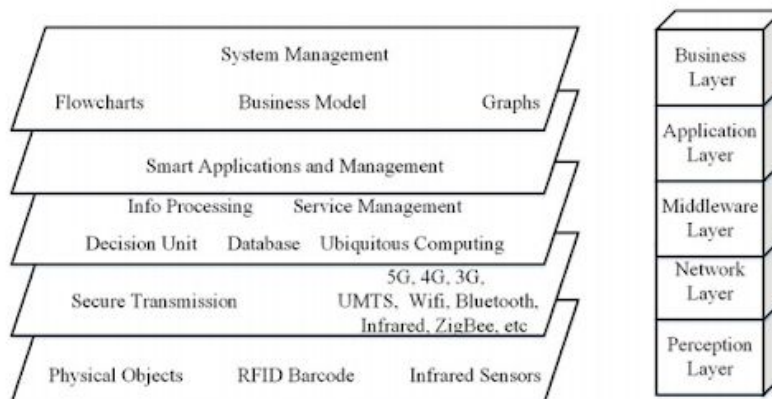


FIG 3.1.1 The Layer of Iot System[21]

IDS is a tool for monitoring traffic data to identify and protect against intrusions that threaten the confidentiality, integrity, and availability of an information system. The operations of an IDS can be divided into three stages. The first stage is the monitoring stage, which relies on network-based or host-based sensors. The second stage is the analysis stage, which relies on feature extraction methods or pattern identification methods. The final stage is the detection stage, which relies on anomaly or misuse intrusion detection. An IDS captures a copy of the data traffic in an information system and then analyzes this copy to detect potentially harmful activities [23].

Here we propose a Hybrid Intrusion Detection System where we make use of machine learning algorithms to develop the IDSs. In our system we have used unsupervised learning to implement host based system and supervised learning for network based. It difficult to conceptualize a universally working machine learning (ML) model across various IoT devices. The features used in models and their behaviors can dynamically be changed with devices and applications [12].

In the Host based system we focus on detecting anomalies caused by hardware fault which would ultimately lead to abnormally functioning of the Iot System. The Network Based IDS we aim at detection of various attacks using supervised learning. We have conducted a comparative study to select the best machine learning model suitable for both host based IDS and network based IDS. The FIG describes the architecture of the proposed system. For host based IDS we have used a dataset of sensor values from Intel lab berkeley and for Network based IDS we have used UNSWNB 15 dataset.

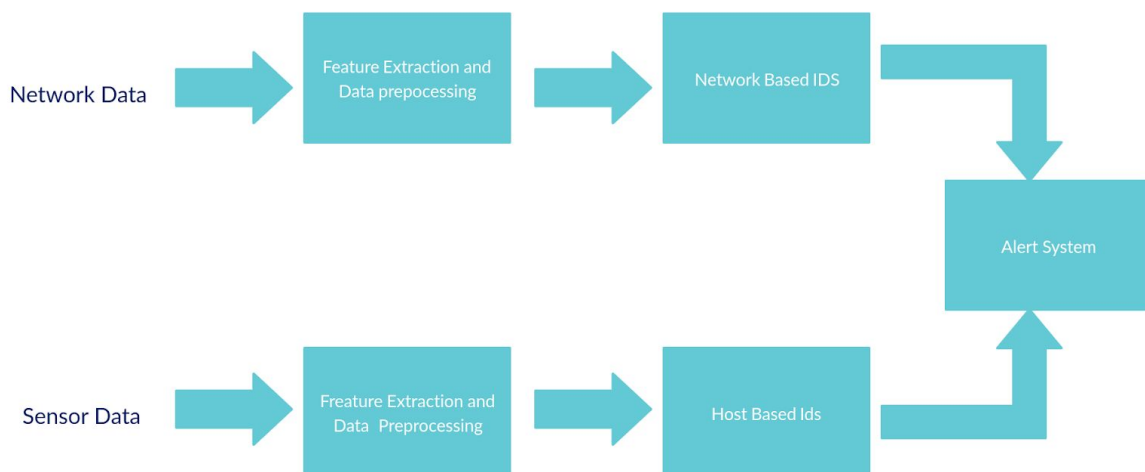


FIG 3.1.2 Architecture of the system

4. Implementation

4.1 Dataset

We have incorporated two datasets into our project. A raw data collected from the Intel Berkeley Research Lab [5] has been used for the HIDS and BoT-IoT dataset for NIDS.

4.1.1 Dataset used for HIDS

The raw data has been collected from the Intel Berkeley Research Lab [5]. This is available Freely and collected from 54 sensors deployed in the Intel Berkeley Research lab between February 28th and April 5th, 2004. Mica2Dot sensors with weatherboards collected timestamped topology information, along with humidity, temperature, light and voltage values once every 31 seconds. Data was collected using the TinyDB in-network query processing system, built on the TinyOS platform. In this data epoch is a monotonically increasing sequence number from each mote. Two readings from the same epoch number were produced from different nodes at the same time. There are some missing epochs in this data set. Moteids range from 1-54; data from some motes may be missing or truncated. Temperature is in degrees Celsius. Humidity is temperature corrected relative humidity, ranging from 0-100%. Light is in Lux (a value of 1 Lux corresponds to moonlight, 400 Lux to a bright office, and 100,000 Lux to full sunlight.) Voltage is expressed in volts, ranging from 2-3; the batteries in this case were lithium ion cells which maintain a fairly constant voltage over their lifetime; note that variations in voltage are highly correlated with temperature.

We created our custom dataset by keeping this as the base and adding class labels. We have divided the dataset into two major classes - Normal and Hardware.

Obtain Normal class from the dataset:

- $D^{Normal} = D_1^N, D_2^N, D_3^N, D_4^N, \dots, D_n^N$ where n presents 1520099 dataset entries out of the original 2,210,084 after removing hardware fault data.
- $D_i^N = X_i, y_i$ where D_i^N is the ith data entry of the dataset, X_i is the vector of the input data, and y is the output data: normal operation data as a data feature.
- $X_i = x_{i1}, x_{i2}, \dots, x_{id}$ where d is four representing the four attributes: temperature, humidity, light, and voltage.

- $y_i = y_i$ where y_i has one output and y_i represents the feature of data entry as a normal operation data.

The provided dataset has abnormal values, such as a high temperature of 122 degrees Celsius, and this hardware fault data is caused by the sensor battery. The FIG. 4.1.1 shows the outlier distribution which would be considered a hardware fault. To Detect the Hardware fault we have used Outlier Detection that is IQR. The outlier Points have been labeled as hardware Fault. Dataset for Hardware is specified as follows:

- $D^{\text{Hardware}} = D^H_1, D^H_2, D^H_3, \dots, D^H_n$ where n represents 930159 dataset entries out of 2,210,084.
- $D^H_i = X_i, y_i$ where D^H_i is the i th data entry of the dataset, X_i is the vector of the input data, and y_i is the output data: hardware fault data.
- $X_i = x_{i1}, x_{i2}, \dots, x_{id}$ where d is four representing the four attributes: temperature, humidity, light, and voltage.
- $y_i = y_i$ where y_i has one output and y_i represents the feature of data entry as a hardware fault data.

Challenges faced while creating a dataset for the malicious type

To generate a malicious dataset, we assume a service compromising scenario in which a malicious attacker purposefully manipulates the temperature value on purpose. In order to emulate such a scenario, one of the data inputs, temperature, is changed from that in the D^{Normal} . According to the measured temperature scope (0 - 40 Celsius) during normal operation, only temperatures between 30 and 40 are randomly decreased by a range of 0 to 5 for ten days out of a total of thirty-eight days.

- $D^{\text{Malicious}} = D^M_1, D^M_2, D^M_3, \dots, D^M_n$, where n represents 523,591 dataset entries, which are modified data out of normal operation data for the purpose of compromising a service.
- $D^M_i = X_i, y_i$ where D^M_i is the i th data entry of the dataset, X_i is the vector of the input data, and y_i is the output data: malicious data.
- $X_i = x_{i1}, x_{i2}, \dots, x_{id}$, where d is four representing the four attributes: temperature, humidity, light, and voltage
- $y_i = y_i$, where y_i represents the feature of data entry as malicious data.

But while applying the unsupervised algorithm, for separating the normal, malicious and hardware data, the algorithm overlapped the outputs of normal dataset and malicious dataset. Most of the normal dataset and malicious dataset were clustered together. We came to this conclusion as the outputs showed homogeneity as a high value but completeness as a very low value. This tells us that all of the observations with the same class label are in the same cluster but all members of the same class are not in the same cluster.

As a solution to this problem we have developed a NIDS using the Bot-IoT Dataset.

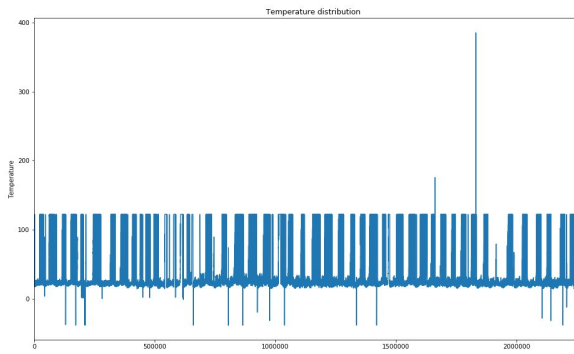


Fig 4.1.1 (a) Temperature Distribution

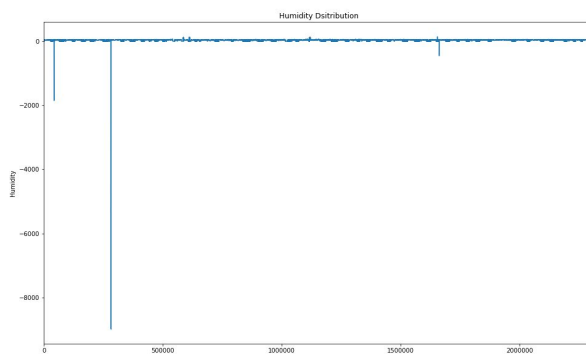


Fig 4.1.1 (b) Humidity Distribution

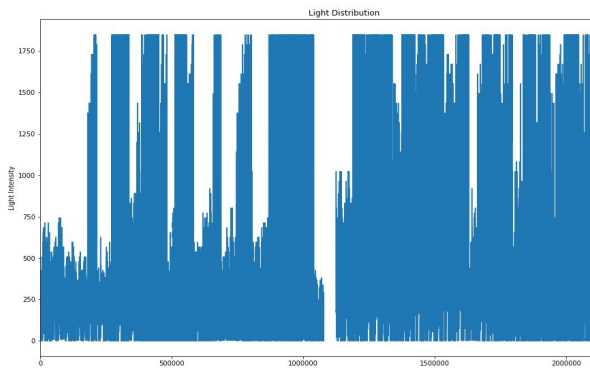


Fig 4.1.1 (c) Light Distribution

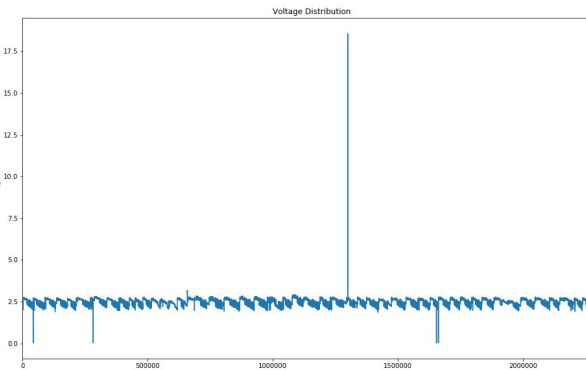


Fig 4.1.1 (d) Voltage Distribution

FIG 4.1.1 Outlier Distribution

4.1.2 Dataset used in NIDS

The Dataset used for the NIDS is the IOT network dataset. The IoT scenarios in the testbed of the dataset:

1. A weather station , which generates information on air pressure, humidity and temperature.
2. A smart fridge, which measures the fridge's temperature and when necessary adjusts it below a threshold.
3. Motion activated lights , which turn on or off based on a pseudo-random generated signal.
4. A remotely activated garage door, which opens or closes, based on a probabilistic input.

5. A smart thermostat , which regulates the house's temperature by starting the Air-conditioning system.

These testbed scenarios are similar to the data used in Hostbased IDs which mainly corresponds to temperature and humidity. In the dataset instances are labeled with '1' and normal ones are labeled with '0' for training and validating machine learning models through a binary classification. In addition to that, we have further introduced attack category and subcategory attributes, which could be used for training and validating multiclass classification models. Along with this classification there is also a subcategory column which corresponds to the type of attack. We have utilised this to create the labels of attacks which are used for classification.

We have divided the dataset into 5 classes based on the column 'category' namely DoS, DoS, Normal, Theft, Reconnaissance . We use various classification algorithms on this data set. After data cleaning and under sampling each class has close to 90000 data points. Using Random Forest(Gini measure) we estimated the importance of the feature in the dataset FIG 4.1.2 shows the graph describing the importance of each feature. The last three features min, sport and drate have gini value less than 0.05 hence these are dropped as the feature

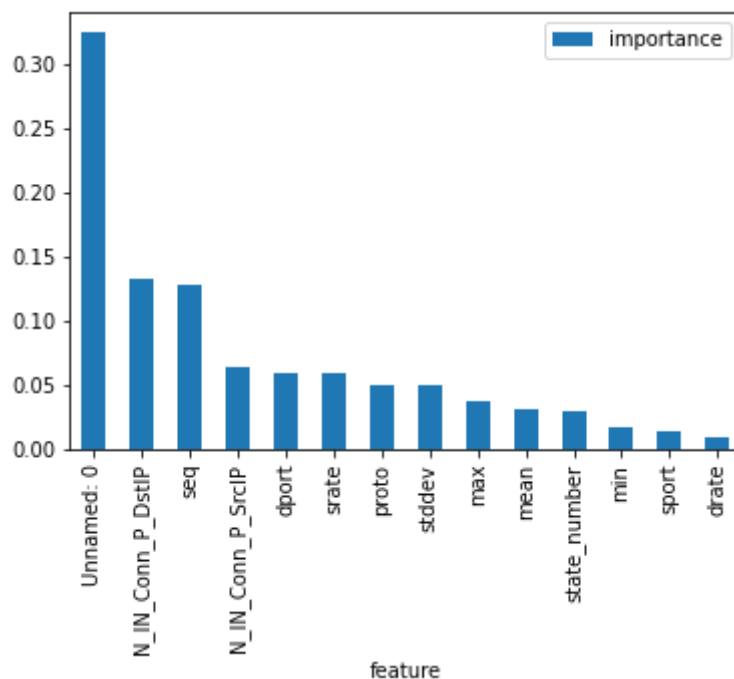


FIG 4.1.2 Feature Distribution

4.1.3 Interquartile Rule for Outliers

Any set of data can be described by its five number summary. These five numbers, in ascending order, consist of:

- The minimum, or lowest value of the dataset
- The first quartile Q1 - this represents a quarter of the way through the list of all the data
- The median of the data set - this represents the midpoint of the list of all of the data
- The third quartile Q3 - this represents three quarters of the way through the list of all the data
- The maximum, or highest value of the data set.

These five numbers can be used to tell us quite a bit about our data. For example, the range, which is just the minimum subtracted from the maximum, is one indicator of how spread out the data set is.

Similar to the range, but less sensitive to outliers, is the interquartile range. The interquartile range is calculated in much the same way as the range. All that we do is subtract the first quartile from the third quartile: $IQR = Q3 - Q1$.

The interquartile range shows how the data is spread about the median. It is less susceptible than the range to outliers.

The interquartile range can be used to help detect outliers. All that we need to do is to is the following:

1. Calculate the interquartile range for our data
2. Multiply the interquartile range (IQR) by the number 1.5
3. Add $1.5 \times (IQR)$ to the third quartile. Any number greater than this is a suspected outlier.
4. Subtract $1.5 \times (IQR)$ from the first quartile. Any number less than this is a suspected outlier.

It is important to remember that this is a rule of thumb and generally holds. In general, we should follow up in our analysis. Any potential outlier obtained by this method should be examined in the context of the entire set of data.

4.2 Algorithms

In our project for Host- based IDS we have used two unsupervised algorithms:

1. K- means algorithm
2. Fuzzy C-means algorithm

And for Network- based IDS we have used four supervised algorithms:

1. Naive Bayes
2. Support Vector Machine (SVM)
3. Logistic Regression
4. k- Nearest Neighbor (kNN)

4.2.1 Algorithms used for HIDS

4.2.1.1 K- Means Algorithm

K-means algorithm is an iterative algorithm that tries to partition the dataset into Kpre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make the inter-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

The way K-means algorithm works is as follows:

1. Specify number of clusters K.
2. Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without replacement.
3. Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn't changing.
 - Compute the sum of the squared distance between data points and all centroids.
 - Assign each data point to the closest cluster (centroid).
 - Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.

The approach K-means follows to solve the problem is called Expectation-Maximization. The E-step is assigning the data points to the closest cluster. The M-step is computing the centroid of each cluster.

Elbow Method

K-means requires k as an input and doesn't learn it from data, there is no right answer in terms of the number of clusters that we should have in any problem. Sometimes domain knowledge and intuition may help but

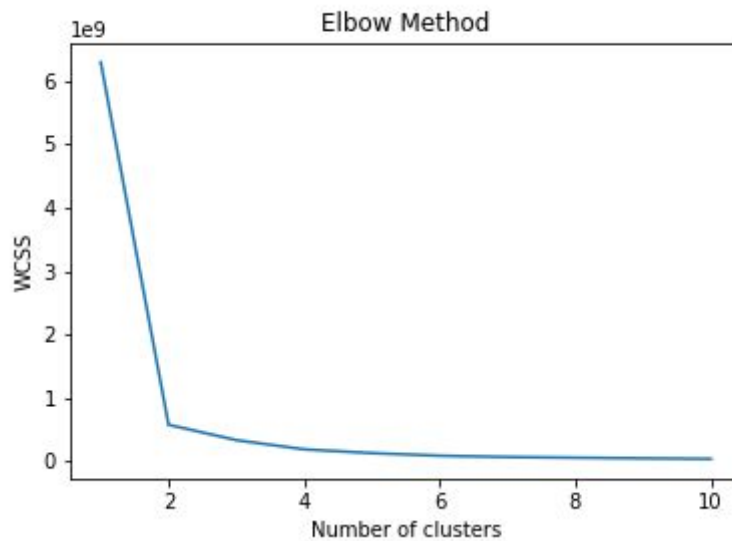


FIG 4.2.1.1 Elbow curve for Kmeans

usually that is not the case. In the cluster-predict methodology, we can evaluate how well the models are performing based on different K clusters since clusters are used in the downstream modeling. Elbow method

gives us an idea on what a good k number of clusters would be based on the sum of squared distance (SSE) between data points and their assigned clusters' centroids. We pick the spot where SSE starts to flatten out and form an elbow. From the Fig 4.2.1 The curve starts to flatten out after cluster 2, so the optimal number of clusters of the given dataset is three.

4.2.1.2 Fuzzy c-means (FCM)

In K-means algorithm each data point belongs to one cluster whereas in Fuzzy clustering (also referred to as soft clustering or soft k-means) is a form of clustering in which each data point can belong to more than one cluster. Fuzzy logic principles can be used to cluster multidimensional data, assigning each point a membership value for each cluster center from 0 to 100 percent and the point will be assigned to the cluster for which it has the maximum

membership value. Fuzzy c-means clustering is accomplished via `skfuzzy.cmeans`, and the output from this function can be repurposed to classify new data according to the calculated clusters (also known as prediction) via `skfuzzy.cmeans_predict`.

4.2.2 Algorithms used for NIDS

4.2.2.1 Naive Bayes

A Naive Bayes classifier is a probabilistic machine learning model that's used for classification tasks. The crux of the classifier is based on the Bayes theorem.

Using Bayes theorem, we can find the probability of A happening, given that B has occurred. Here, B is the evidence and A is the hypothesis. The assumption made here is that the predictors/features are independent. That is, the presence of one particular feature does not affect the other. Hence it is called naive.

The main advantage of Naive bayes is that that algorithm is fast hence can be deployed in real time and it is suitable for multiclass predictions.

4.2.2.2 Support Vector Machine (SVM)

In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall. on-linear SVM means that the boundary that the algorithm calculates doesn't have to be a straight line. The benefit is that you can capture much more complex relationships between your data points without having to perform difficult transformations on your own. The downside is that the training time is much longer as it's much more computationally intensive.

4.2.2.3 Logistic Regression

Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. This function has a nice property, basically it maps any value to the range [0,1] which is appropriate to handle probabilities during the classification. For example in the case of a binary classification $g(X)$ could be interpreted as the probability to belong to the positive class. In this case normally you have different classes that are separated with a decision boundary which is basically a curve that decides the separation between the different classes. Following is an example of a dataset separated in two classes.

Logistic regression is a statistical model that in its basic form uses a logistic function to model a dependent variable. In the multi class logistic regression python Logistic Regression class, multi-class classification can be enabled/disabled by passing values to the argument called ‘multi_class’ in the constructor of the algorithm. In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the ‘multi_class’ option is set to ‘ovr’ and uses the cross-entropy loss if the ‘multi_class’ option is set to ‘multinomial’.

4.2.2.4 KNN

K-nearest neighbors (KNN) algorithm is a type of supervised ML algorithm which can be used for both classification as well as regression predictive problems.

K-nearest neighbors (KNN) algorithm uses ‘feature similarity’ to predict the values of new data points which further means that the new data point will be assigned a value based on how closely it matches the points in the training set. We can understand its working with the help of following steps –

Step 1 – For implementing any algorithm, we need a dataset. So during the first step of KNN, we must load the training as well as test data.

Step 2 – Next, we need to choose the value of K i.e. the nearest data points. K can be any integer.

Step 3 – For each point in the test data do the following –

- 3.1 – Calculate the distance between test data and each row of training data with the help of any of the methods namely: Euclidean, Manhattan or Hamming distance. The most commonly used method to calculate distance is Euclidean.

- 3.2 – Now, based on the distance value, sort them in ascending order.
- 3.3 – Next, it will choose the top K rows from the sorted array.
- 3.4 – Now, it will assign a class to the test point based on the most frequent class of these rows.

Step 4 – End

4.4 Up Sample and Down Sample

Most machine learning algorithms work best when the number of samples in each class are about equal. This is because most algorithms are designed to maximize accuracy and reduce error. Therefore, we oversample or downsample the data. Oversampling can be defined as adding more copies of the minority class. Undersampling can be defined as removing some observations of the majority class.

Here we will use imblearn's SMOTE or Synthetic Minority Oversampling Technique for upsampling. SMOTE uses a nearest neighbors algorithm to generate new and synthetic data we can use for training our model. Again, it's important to generate new samples in the training set to ensure our model generalizes well to unseen data.

The new instances are not just copies of existing minority cases; instead, the algorithm takes samples of the feature space for each target class and its nearest neighbors, and generates new examples that combine features of the target case with features of its neighbors. This approach increases the features available to each class and makes the samples more general.

SMOTE takes the entire dataset as an input, but it increases the percentage of only the minority cases. For example, suppose you have an imbalanced dataset where just 1% of the cases have the target value A (the minority class), and 99% of the cases have the value B. To increase the percentage of minority cases to twice the previous percentage, you would enter 200 for SMOTE percentage in the module's properties.

A simple under-sampling technique is to under-sample the majority class randomly and uniformly. This can potentially lead to loss of information. But if the examples of the majority class are near to others, this method might yield good results.

5. Results and Analysis

The Dataset in both the cases has been split in to train and test based in 80:20 ration.Accuracy has been used as evaluation metrics in both the cases.

5.1 Network Based IDS

The measures used to assess NIDS performance depend on four factors, namely, the numbers of true positives (α), true negatives (δ), false positives (γ) and false negatives (β).We have used multiple classification algorithms to identify the best suitable algorithm. The algorithm we explored are KNN, SVM, logistic Regression, Naive Bayes the results are as follows

	Test	Test %	Train	Train %	
NB	0.846	84.560	0.876	87.620	False Positive 0.1092 False Negative 0.1092 True Positive 0.8908
SVM	0.975	97.470	0.988	98.820	False Positive 0.042 False Negative 0.042 True Positive 0.958
LR	0.942	94.170	0.969	96.910	False Positive 0.0588 False Negative 0.0588 True Positive 0.9412
KNN	0.949	94.949	0.962	96.174	False Positive 0.0504 False Negative 0.0504 True Positive 0.9496

Ditribution of Test and Train



FIG 5.1.1 Distribution of Accuracy of NIDs

The result above shows that the models are giving accuracy above 90% . SVM give the best result with 98.82 % Train Accuracy and 97.47% Test Accuracy. FIG 5.1.1 describes the accuracy distribution. And Fig 5.1.2 shows the confusion matrix for the above described algorithms.

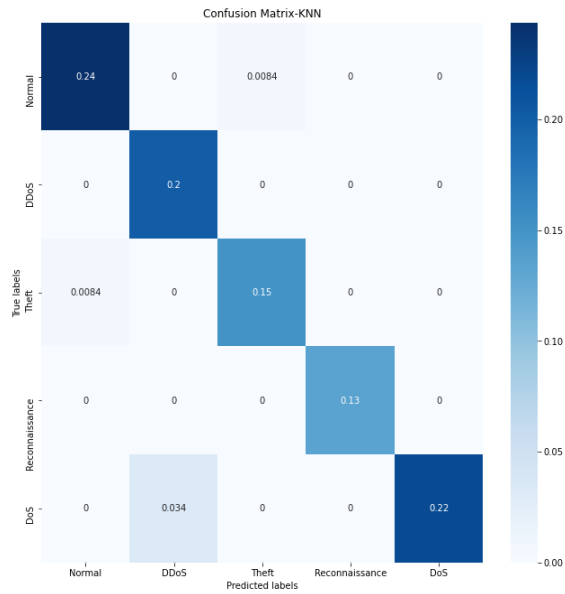


FIG 5.1.2 (a) KNN

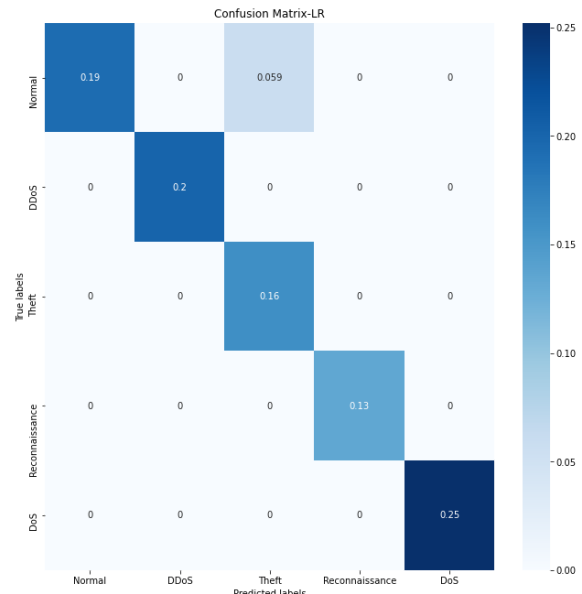


FIG 5.1.2 (b) LR

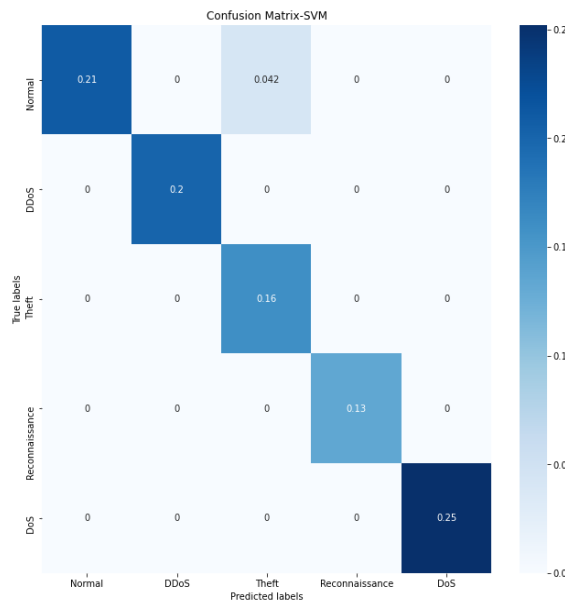


FIG 5.1.2 (c) SVM

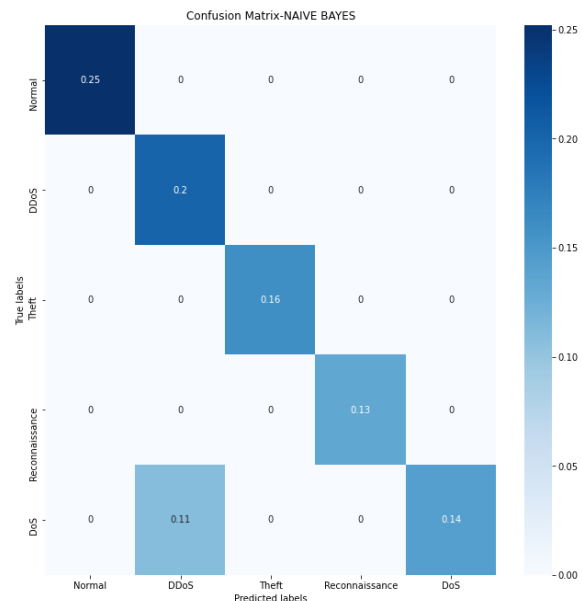


FIG 5.1.2 (d) NB

FIG 5.1.2 Confusion matrix

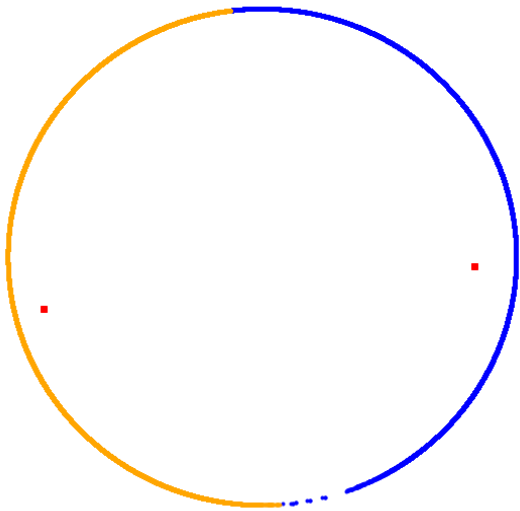
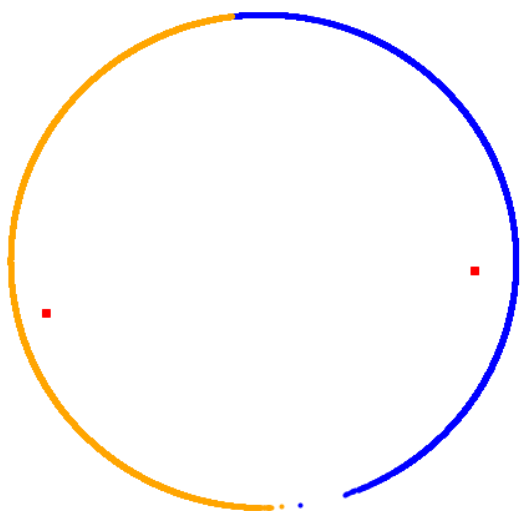
5.2 Host Based IDS

For model evaluation, we have assigned cluster labels - normal and hardware to the data oriented dataset. To calculate the accuracy for k-means and fuzzy c-means, we have compared the assigned labels with the original labels.

K- means

Train	Test
Accuracy = 92.56	Accuracy = 90.743

Fuzzy C-means

Train	Test
<p>Centers = 2; FPC = 0.88</p> 	<p>Centers = 2; FPC = 0.88</p> 
Accuracy = 92.428491470283	Accuracy = 92.39141968607413

Comparison between test and train accuracies of the algorithms

Train and Test

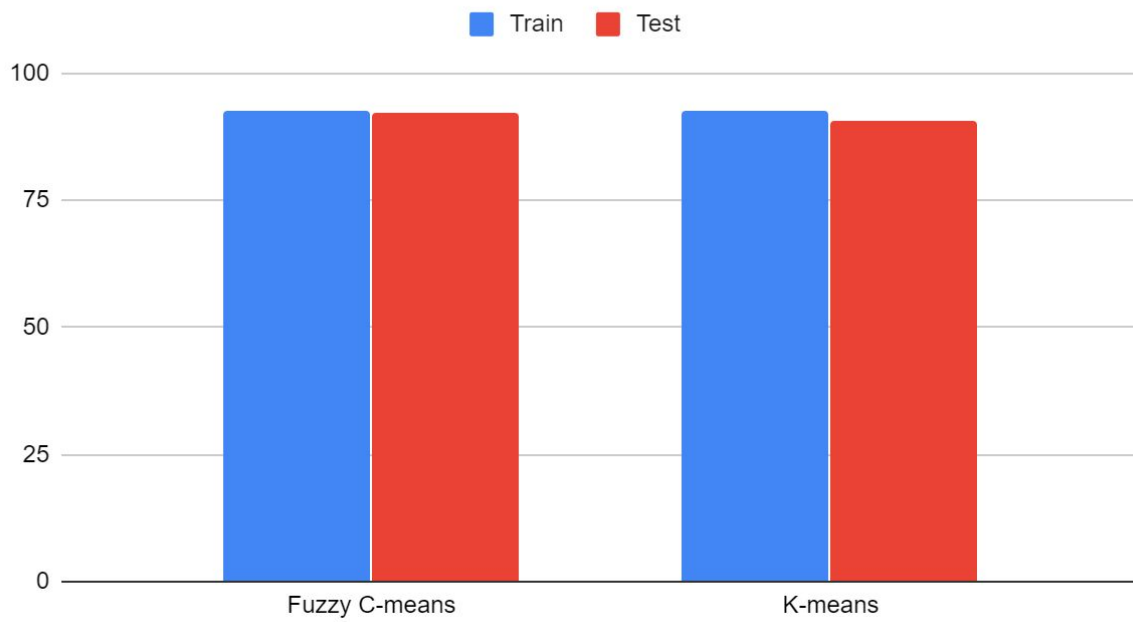


FIG 5.2.1 Accuracy Distribution

6. Conclusion and Future work

As the numbers of IoT users, services, and applications increase, an urgent need for a robust and lightweight security solution that is suitable for use in IoT environments is emerging. Furthermore, IoT networks are the basis of smart environments; thus, any deficiencies in the security of these IoT networks will directly influence the smart environments on which they are based. Attacks such as DoS, DDoS, and Reconnaissance attacks affect the services and applications offered in IoT-based smart environments; thus, the security of IoT environments is a very serious issue.

We have carefully developed a hybrid IDS, an autonomous IDS that can detect intrusions without human intervention is required for application in the IoT environment. The hybrid IDS consists of a Host- based IDS (HIDS) and a Network- based IDS (NIDS). HIDS is to detect anomalies in the traffic using unsupervised algorithms - Fuzzy C-means, K-means. NIDS is used for malware detection using supervised algorithms like - Naive Bayes, Support Vector Machine, Logistic Regression and k-Nearest Neighbor. We have also compared the algorithms using metrics - TPR, FPR and accuracies and come to a conclusion for the best algorithm that can be used in each of the IDSs. For NIDS, SVM is considered to be the most apt algorithm among others and for HIDS, it is Fuzzy C-means.

In the future, we are planning on implementing the HIDS that will be compatible with 6LowPAN protocol to detect DOS attacks. In an IOT- based smart environment, the energy consumption processing time and performance overhead of an IDS are also of critical interest. Because of the power and memory limitations of IoT devices, these metrics are very important to the QoS of an IoT system. So we will be comparing the algorithms based on these metrics too.

References

- [1] A. L. Samuel, “Some studies in machine learning using the game of checkers,” IBM Journal of Research and Development, vol. 3, no. 3, pp.210–229, Jul 1959.
- [2] H. M. B. D. Cardinaux F., Brownsell S., “Modelling of behavioural patterns for abnormality detection in the context of lifestyle reassurance.” In: Ruiz-Shulcloper J., Kropatsch W.G. (eds) Progress in Pattern Recognition, Image Analysis and Applications. CIARP 2008. Lecture Notes in Computer Science, vol. 5197, 2008.
- [3] M. Rose and K. McCloghrie, “Structure and identification of management information for tcp/ip-based internets,” IETF, STD 16, RFC 1155, May 1990. [Online]. Available: <https://www.rfc-editor.org/info/rfc1155>
- [4] M. Hildebrandt, S. Gutwirth, M. Hildebrandt, and S. Gutwirth, Profiling the European citizen: cross-disciplinary perspectives, 1st ed. Springer Publishing Company, Incorporated, 2008.
- [5] S. Madden. (2003) Intel berkeley research lab data. [Online]. Available:<http://db.csail.mit.edu/labdata/labdata.html>
- [6] R. Yan, T. Xu, and M. Potkonjak, “Data integrity attacks and defenses for intel lab sensor network,” in Proceedings of the 2015 IEEE 2Nd World Forum on Internet of Things (WF-IoT), ser. WF-IOT ’15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 721–726.[Online]. Available: <http://dx.doi.org/10.1109/WF-IoT.2015.7389143>
- [7] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, “Profiliot: a machine learning approach for iot device identification based on network traffic analysis,” in Proceedings of the Symposium on Applied Computing, ser. SAC ’17. New York, NY, USA: ACM, 2017, pp. 506–509.
- [8] A. Serbanati, C. M. Medaglia, and U. B. Ceipidor, “Building blocks of the internet of things: State of the art and beyond,” Deploying RFID-Challenges, Solutions, and Open Issues, InTech, 2011.
- [9] R. Clarke, “Profiling: a hidden challenge to the regulation of data surveillance,” Journal of Law and Information Science, vol. 4, p. 403, 1993.
- [10] R. Ferrando and P. Stacey, “Classification of device behaviour in internet of things infrastructures: towards distinguishing the abnormal from security threats,” in Proceedings of

the 1st International Conference on Internet of Things and Machine Learning, ser. IML '17. New York, NY, USA: ACM, 2017, pp. 57:1–57:7.

[11] S. Myagmar, A. J. Lee, and W. Yurcik, "Threat modeling as a basis for security requirements." Symposium on Requirements Engineering for Information Security (SREIS), Aug 2005. [Online]. Available: <http://d-scholarship.pitt.edu/16516/>

[12] P. Ji and M. Szczodrak, "A multivariate model for data cleansing in sensor networks," Mar 2019.

[13] R. Yan, T. Xu, and M. Potkonjak, "Data integrity attacks and defenses for intel lab sensor network," 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), pp. 721–726, Dec 2015. [14] S. Lee, S. Wi, E. Seo, J. Jung, and T. Chung, "Profiot: Abnormal behavior profiling (abp) of iot devices based on a machine learning approach," in 2017 27th International Telecommunication Networks and Applications Conference (ITNAC), Nov 2017, pp. 1–6.

[15] Y. x. Xie, X. g. Chen, and J. Zhao, "Data fault detection for wireless sensor networks using multi-scale PCA method," 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC), pp. 7035–7038, Aug 2011.

[16] M. E. Ahmed, S. Ullah, and H. Kim, "Statistical application fingerprinting for ddos attack mitigation," IEEE Transactions on Information Forensics and Security, vol. 14, no. 6, pp. 1471–1484, Jun 2019.

[17] K. McCloghrie and M. Rose, "Management information base for network management of tcp/ip-based internets: Mib-ii," STD 17, RFC 1213, Mar 1991. [Online]. Available: <https://www.rfc-editor.org/info/rfc1213>

[18] L. Pan and J. Li, "K-nearest neighbor based missing data estimation algorithm in wireless sensor networks," Wireless Sensor Network, vol. 2, pp. 115–122, Jan 2010.

[19] G. Kortuem, F. Kawsar, D. Fitton, and V. Sundramoorthy, "Smart objects as building blocks for the internet of things," IEEE Internet Computing, vol. 14, no. 1, pp. 44–51, Jan 2010.

[20] N. V. Chawla, K. W. Bowyer, L. O'Hall, W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," Journal of artificial intelligence research, 321–357, 2002.

[21] Khan R, Khan S, Zaheer R, Khan S (2012) Future internet: The internet of things architecture, possible applications and key challenges. In: 2012 10th International Conference on Frontiers of Information Technology. IEEE, Islamabad. pp 257–260

- [22] Kumar S, Vealey T, Srivastava H (2016) Security in internet of things: Challenges, solutions and future directions. In: 2016 49th Hawaii International Conference on System Sciences (HICSS), Koloa. pp 5772–5781
- [23] Anwar S, Mohamad Zain J, Zolkipli MF, Inayat Z, Khan S, Anthony B, Chang V (2017) From intrusion detection to an intrusion response system: Fundamentals, requirements, and future directions. *Algorithms* 10(2):1–24
- [24] Feng W, Zhang Q, Hu G, Huang JX (2014) Mining network data for intrusion detection through combining SVMs with ant colony networks. *Futur Gener Comput Syst* 37:127–140
- [25] Namdev N, Agrawal S, Silkari S (2015) Recent advancement in machine learning based internet traffic classification. *Procedia Comput Sci* 60:784–791
- [26] Muzammil MJ, Qazi S, Ali T (2013) Comparative analysis of classification algorithms performance for statistical based intrusion detection system.
- [27] In: 2013 3rd IEEE International Conference on Computer, Control and Communication (IC4), Karachi. pp 1–6
- [28] Mabu S, Chen C, Lu N, Shimada K, Hirasawa K (2011) An intrusion-detection model based on fuzzy class-association-rule mining using genetic network programming, Vol. 41
- [29] Xu C, Chen S, Su J, Yiu SM, Hui LCK (2016) A survey on regular expression matching for deep packet inspection: Applications, algorithms, and hardware platforms. *IEEE Commun Surv Tutor* 18(4):2991–3029
- [30] Davis JJ, Clark AJ (2011) Data preprocessing for anomaly based network intrusion detection: A review. *Comput Secur* 30(6–7):353–375
- [31] Vancea F, Vancea C (2015) Some results on intrusion and anomaly detection using signal processing and NEAR system. In: 2015 38th International Conference on Telecommunications and Signal Processing (TSP). IEEE, Prague. pp 113–116