INVESTIGATING THE DIRECT DERIVATION OF DEVICE IDENTITY FROM LOW-LEVEL HARDWARE DEVICES

Pooja Rajesh Khanna

A Thesis Submitted to the University of Kent for the Degree of Doctor of Philosophy in the Subject of Electronic Engineering

October 2022

Abstract

The security and privacy are one of the top priorities in the growing field of Internet of Things (IoT), as industries are expanding to incorporate wireless technology to generate valuable data. Adversaries attack the vulnerable node in the system to gain a wider access into the network. This thesis utilises the premise of ICMetrics technology to introduce a novel technique to secure low – level hardware used to build IoT applications. Previously, the technique has been used on off – the shelf IoT devices to generate keys and use it for authentication.

Building on the work of ICMetrics technology, this thesis aims to cover a research gap by using cost – effective, low – resource embedded devices like Raspberry Pi and the basis of the ICMetrics technology i.e., using internal characteristics of the devices and utilise it to identify the devices uniquely. The work performed in this thesis aims to explore the operational features (memory – based) from the devices to generate the device identity.

Based on the Literature Review, Raspberry Pi has served as a gateway for several large – scale attacks by using its vulnerabilities. Hence, we aim to exploit the uniqueness shown by these devices based on the working, environment where these devices are deployed and the application/processes running on the devices by exploiting dynamic operational features from them. The key contribution of this thesis is exploring features that are multimodal in nature and by utilizing the relationship between the modes, a novel classification technique is built which takes in account the multimodal nature of the features to gain higher accuracy in identifying devices.

Hence, making the widely used Raspberry Pi devices more secure and in case of any attacks, the characteristics of the devices change prevents the critical data leak from the devices & network.

Acknowledgement

This thesis work was supported by project INCASE: Industry 4.0 via Networked control applications and Sustainable Engineering which was funded by Interreg IV 2 Seas Programme (North). I would also like to express my sincere thanks to the University of Kent for supporting we with the resources required for my research work as well as for helping me to have an excellent research experience.

I would like to first thank my supervisor Prof. Gareth Howells, Professor of Secure Electronic Systems and Director of Research and Innovation, University of Kent for guiding me throughout my research period and providing excellent support. I feel privileged to have make most out of his experience and excellent mentorship which helped me to have an excellent research experience and contribute towards the field of cyber security. I would also like to thank my co-supervisor Dr Konstantinos Sirlantzis, Senior Lecturer in Intelligent Systems for providing me guidance, feedback, and support.

I would also like to specially thank my friends Priyanka and Naveen who advised me during the coding mishaps. I am grateful to my friends and colleagues at University of Kent who have made these research years fun, memorable and a great learning experience.

A very special thanks to my better half, who has been my personal cheerleader and has supported me through emotional rollercoasters with a great deal of love, encouragement, and proof – reading my work.

I am infinitely grateful to my parents, who believed in me and provided me with unconditional support. I am blessed to have you both guide me, encourage me in whatever I do. I would not have made it through without you. Thank you for being my calm in the midst of chaos.

Thesis Contributions

JOURNAL PAPERS

[1] P. R. Khanna, G. Howells, and P. I. Lazaridis, "Design and Implementation of Low-Cost Real-Time Energy Logger for Industrial and Home Applications," *Wireless Personal Communications,* vol. 119, no. 3, pp. 2657-2674, 2021.

[2] S. Yadav, P. R. Khanna, and G. Howells, "Device Authentication Using Wavelet Based Features," 2022.

Conference Papers:

[3] P. R. Khanna and G. Howells, "A novel cost-effective Pressure Sensor based Smart Car park system," in *2020 XXXIIIrd General Assembly and Scientific Symposium of the International Union of Radio Science*, 2020: IEEE, pp. 1-4.

[4] P. R. Khanna and G. Howells, "Using Dynamic Operational features to Identify Embedded Devices," in *2021 XXXIVth General Assembly and Scientific Symposium of the International Union of Radio Science (URSI GASS)*, 2021: IEEE, pp. 1-4.

[5] S. Yadav, P. Khanna, and G. Howells, "Robust Device Authentication Using Non-Standard Classification Features," 2021.

[6] S. Yadav, P. R. Khanna, and G. Howells, "Device Identification Using Discrete Wavelet Transform," in *2021 International Conference on Engineering and Emerging Technologies (ICEET)*, 2021: IEEE, pp. 1-6

Table of Contents:

List of Acronyms

ADICOV	Approximate of a Distribution for a given COVariance
AES	Advanced Encryption System
ARP	Address Resolution Protocol
ASIC	Application Specific Integrated Circuit
AWGN	Additive White Gaussian Noise
CARS	Context Aware Recommendation System
CART	Classification and Regression Trees
COTS	Commercial off-the-shelf
CRP	Challenge Response Pair
CSES	Centre for Strategy & Evaluation Services
CST-GR	correlated – set thresholding on gain – ratio
CWT	Continuous Wavelet Transform
db	Daubechies Wavelet
DCMS	Department for Digital, Cultural, Media & Sports
DCT	Discrete Cosine Transform
DES	Data Encryption System
DL	Deep Learning
DoS	Denial of Service
DP	Differential Privacy
DPCM	Differential Pulse Code Modulation
DTW	Dynamic Time Warping
ECC	Elliptic Curve Cryptography
ED	Eigen Decomposition
EM	Electromagnetic
EPC	Electronic Product Code
ES	Evolution Strategies
F1	Feature 1
F2	Feature 2
FFT	Fast Fourier Transform
FN	Fiestel Networks
FPGA	Field Programmable Gate Array
FT	Fourier Transform
GAN	Generative Adversarial Network
GNB	Gaussian Naive Bayes
HMM	Hidden Markov Models
ICMetrics	Integrated Circuit Metrics
IDC	International Data Corporation
IDS	Intrusion Detection Systems
IoT	Internet of Things
JPEG	Joint Photographic Experts Group
KPSS	Kwiatkowski, Phillips, Schmidt, and Shin
LED	Light Encryption Device
LMT	Logistic Model tree

LPF	Low Pass Filter
LR	Logistic Regression
ML	Machine Learning
MMHCI	Multi-modality human-computer interface
MMVGD	Multimodal Multivariate Gaussian Distribution
PC	Program Counter
PCM	Pulse Code Modulation
PDF	Probability Density Function
PDF	Probability Density Function
PDP	Probability Distribution Plane
PeGS	Perturbed Gibbs Sampler
PMI	Perturbed Multiple Imputation
PMML	Predictive Model Markup Language
PUF	Physically Unclonable Functions
Rbf	
RF	Random Forest
RFID	Radio Frequency Identification
RPi	Raspberry Pi
RSA	Rivest - Shamir - Adleman
SDDL	Synthetic Data Definition Language
SDV	Synthetic Data Vault
SNORT	Signature or Anomaly based detection
SoC	System-on-chip
SP	Synthpop
SPN	Substitution Permutation Network
SRPL	Secure Routing Protocol
SSD	Secure Software Development
SSDLC	Secure Software Development Life Cycle
SSE	Sum of Square Error
SSH	Secure Shell
STFT	Short Time Fourier Transform
SVD	Singular Value Decomposition
SVM	Support Vector Machine
sym	Symlet Wavelet
SYN	Synchronization
VAE	Variational Autoencoder
VFDT	Hoeffding tree
WSN	Wireless Sensor Network
WT	Wavelet Transform

List of Tables

List of Figures

1 INTRODUCTION

The advances in the technology have been immense since the development of IoT (Internet of Things). With the advancement in technology the concept of security becomes one of the most crucial aspects of any IoT application. The introduction of the automated devices which contribute to making life easier for general population does come with certain consequences. Amongst all the technical advances, Internet of Things has made the greatest impact on us by generating data from sensor – based network or devices. Hence, making security one of the major concerns for the experts. The IoT industry is anticipated to build about 75.44 billion connected devices by the end of 2025 [7, 8]. The International Data Corporation (IDC) estimates connected devices would generate 175 zettabytes of information. The growth of the analysed data that is 'touched' by cognitive systems will grow by a factor of 100 [9].

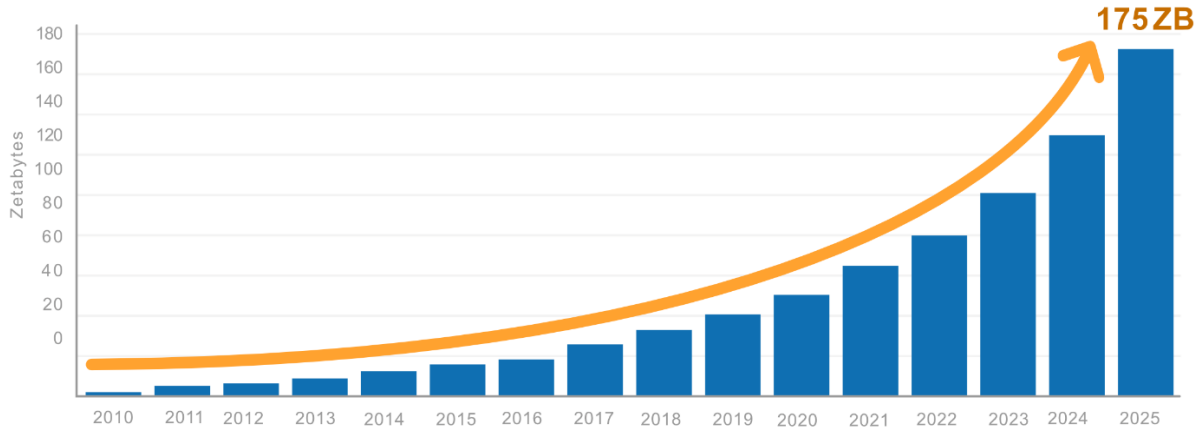


Figure 1 Annual Size of the Global Datasphere[9]

The rise of connected devices affects various domains i.e., rise of cognitive systems which would be powered with data, data privacy would be of greater significance and security systems in place would reflect the importance of this data[9]. When the data – driven applications are built using data in combination with pattern recognition techniques, it is extremely important to build security measures to prevent data leaks or eavesdropping.

These IoT applications can be perceived as a conjunction between numerous physical and virtual entities working together with each other, the integration of these entities offers a wide range of services [10]. There are several state-of-the-art security techniques available to users and big data firms. Any security techniques chosen by any firm, or by the users depends

upon what we are planning to secure. Whether it be using encryption algorithm, a cryptographic technique or any algorithm used depends on the resources available to utilise.

The devices using heavy encryption and security algorithms have numerous resources and computational power to utilise which in-turn makes expensive devices. The larger and expensive devices already have number of resources available and compatibility with other frameworks. Recently, there has been an increase in the resource constrained embedded IoT devices that use Bluetooth, Wi – Fi, Zigbee, Ethernet etc to connect the sensor – based network to the cloud for various services. Figure 2 below gives an example of the computational power and cost based on the type of devices used.

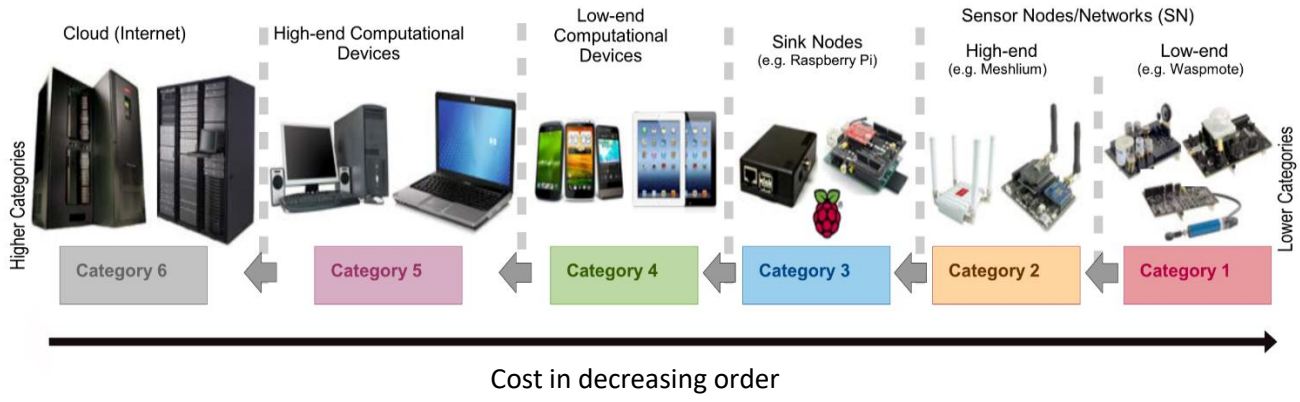


Figure 2 Computational Power and Cost of IoT devices categorically[11]

IoT devices have covered a portion of various industrial domains namely healthcare, agriculture, manufacturing, finance etc, which are responsible to collect data and sensitive information. Each device having substantial computational power, has the capability to implement any state-of-the-art cryptographic, encryption etc techniques on the device. The complexity of the device is proportional to its cost. This makes the cost of the device is a major factor considered by users depending on the scale the device would be used at. Hence, the off – the – shelf devices for the industries and home – based applications available on market, are easy to use, lower in cost and accessible to users are popular. The lower – cost of these devices have a greater risk of security breach. The article [12] addresses the risk of using off – the – shelf at the price of security. These devices are widely used by various businesses where the security risks are often neglected.

Hence, a novel technology was developed by Metrarc [13] for commercial off – the – shelf (COTS) devices to derive the encryption keys to secure the systems using their properties. Novel security solutions were developed by them for various sectors like, m – commerce, online banking, e – commerce and IoT. This technique developed is called ICMetrics (Integrated Circuit Metrics). As the technology advances, there is a need to ensure the authenticity and integrity of the digital systems. This is required to control, manage, and verify their identity. The importance of protecting each node in IoT network is critical as one faulty or compromised node can take the whole network down or gain access to the information communicated by them as these devices are communicating independently. The work presented in this thesis, uses ICMetrics as a basis for simpler devices like Raspberry Pi and use this for identification and authentication purposes. By using the basis of ICMetrics system, a novel method of classification is proposed which uses multimodal features from Raspberry Pi Devices.

Before this Research, the ICMetrics technique was used for generating unique identifiers using the internal hardware and software-based data/features to generate these identifiers and encryption keys. These keys are directly derived from the behaviour displayed by the device and are not stored onto it. Typically, the technique has been used in various low resource off the shelf complex embedded devices. These are sensor-based devices used to generate the device identity using the features derived by using the processing core information of the off the shelf health monitoring devices, FPGA, apple watches, servers etc. The aim is to use the features extracted from the processor or the sensors attached to them to generate the unique device identifier. This is to ensure secure communication using the encryption keys and device signature which is generated using its behavioural characteristics and hardware characteristics. There is no storage of the encryption keys or templates, this prevents the unauthorised access to connected devices. This prevents the tampered device to be characterised as the authorised users as the tampering will change the behaviour of the device.

The features extracted from the devices are used in combination with Machine Learning (ML) techniques to perform the device identification by using state of the art ML algorithms. Hence, this novel technique adds a layer of security to commercial off – the – shelf devices

which protects these devices from accessing the information and furthering the attacks in the network.

The work presented in this thesis uses the simpler devices which are Raspberry Pi based, where the features are extracted and used with novel classification technique which are then benchmarked with state-of-the-art ML techniques to perform device classification for device identification.

1.1 Research Motivation

The Literature Review on conventional security techniques applied on IoT devices, some of the crucial problems were found in context of security for low – resource embedded devices. These devices also have limited computational capabilities so any security technique to be used on devices should be lightweight so, it does not use up all the available resources to the device. Another factor of looking into low – resource devices is that these are cost effective devices used for various applications. Hence, the security protocols when applied should not affect the overall cost of the application immensely.

The work in this thesis addresses these problems to form a few research questions and perform analysis to find a solution. The research questions put together are as follows:

1. Is it possible to identify simpler embedded devices as an attempt to build a layer of security?
    a. Can characteristics from a low – resource, simple embedded devices like Raspberry Pi generate a device signature?
    b. Can highly unstable dynamic multimodal features can be utilised for classification algorithms to improve performance?
2. Is the proposed classification capable of identifying modes in a multimodal distribution correctly?
    a. How does the accuracy trend using the proposed algorithm vary with considerable number of devices.

1.2 THREAT MODEL

Raspberry Pi devices are used as a stand – alone device or a part of a wider Sensor – based Network. The Wireless Sensor Networks (WSNs) are highly prone to active and passive attacks i.e., Denial of Service (DoS), Node Replication, Node capture attack etc, with the aim of compromising the device and infecting the network. This consequently leads to more severe attacks such as malware installation, data breach from crucial machines in the network. The technique proposed in the thesis, understands the vulnerabilities of the Raspberry Pi devices where physical and network-based attacks are impossible to prevent, aims to secure these devices to block the hack moving laterally through the network by using a second layer of security using ICMetrics technique to transport data securely to any server/database further deeper into the network by identifying the devices based on characteristics.

Node capture attacks are common in WSNs and target the weakest node in the network (in terms of security) to gain information about the network either actively or passively. This type of attack enhances the possibility of eavesdropping attack throughout the network since each node in WSN is part of a same network where the encrypted data packets are passed throughout the network easily intercepted and can be used to gain access to the keys, operations, protocols set up for the system. The next scenario is when the attackers collect all the needed information, they use it to plant either malicious scripts to convert the nodes into malware botnet or query various nodes to be further informed about the operations.

A similar attack published in [14] was observed on the Android devices where users downloaded an app named 'Swing VPN' to protect their data has been modifying user's devices into a malware botnet. The app pings a particular website repeatedly every 10 seconds which when scaled up by substantial number of people using the app builds up considerable amount of traffic. Therefore, launching a DDoS attack using abundant devices. These events provide a wider perspective of how the cost – effective solution can prove to be more treacherous than beneficial. Hence, we use the operational features from the low – level hardware devices to distinctly identify the devices to secure them and further enhance the security by confirming the data received from edge devices is from a reliable source.

1.3 THESIS STRUCTURE

Chapter 2 – Literature Review

The chapter contains an extensive information on kinds of security threats occurring on each layer of the IoT device, the state – of – the – art cryptographic techniques and where they are used. Then the properties of security are explored and all the techniques i.e., cryptographic techniques and generalised security attacks are compiled as a base to understand which attacks are affecting Raspberry Pi's. Given that Raspberry Pi's are resource constrained devices, we explore the other kinds of security measures which are beneficial for the RPi (Raspberry Pi) based devices.

Then we introduce the two methods PUFs (Physically Unclonable Functions) and ICMetrics (Integrated Circuit Metrics) technique, then review the previously done work using these techniques.

Chapter 3 – ICMetrics Device Identification Technique

This chapter explains the ICMetrics technique to use the features from the Raspberry Pi. As previously the technique was used for commercial off – the – shelf devices, the features from RPi are examined thoroughly. The RPi based devices are prototypes built at the university and Part – time work at British Telecom. These are presented in chapter 6.

Since the features are highly dynamic and multimodal in nature, we propose a novel classification technique **Multimodal Multivariate Gaussian Distribution (MMVGD)** which takes in account the multimodality of the features. This technique is benchmarked with the state-of-the-art linear classification algorithms.

Chapter 4 – Alternative Device Recognition using wavelet – based features

This chapter explores wavelets for device recognition, where we examine the wavelet-based features for device recognition. Prior to this research, the wavelets are used extensively in various domains like audio processing, Machine vision, signal processing, human activity recognition etc.

Wavelets have proven to yield good results for non – stationary data. The KPSS test was performed to prove that the data is non – stationary in nature. Hence, these are two reasons

we use wavelets for device recognition. Another reason to use wavelets is the processing time. Using simpler wavelets to convert the data into frequency domain, also reduces the number of samples. Since the classification is the main aim for this thesis, we have not compared the processing times for the algorithms. The wavelet – based features are subjected to the proposed classification method MMVGD for classification and benchmark it by using the state-of-the-art classification techniques.

Chapter 5 – Synthetic Dataset Generation – Multivariate Multimodal Features

This chapter presents an algorithm to generate Multivariate Multimodal Dataset using Python programming language. The algorithm is developed to validate the outcomes of proposed classification method to detect modal boundaries and the statistics per modal combination.

It also implements the manual and automated generation of the data. The chapter also addresses a few case scenarios to consider when looking at multimodal data. The part of the work presented in this chapter is to validate the proposed classification method that addresses multimodal data for classification and its ability to handle large number of devices.

2 LITERATURE REVIEW

In the recent years, as the technology advances the devices become smaller in size with greater capabilities. The greater capabilities include sensing, control, interconnectivity between devices and provides services to each other. The things with these abilities are known as smart objects or connected things. The word 'smart' indicates its capabilities to convert the raw data acquired from the environment to information to draw the decisions based on the data [15]. There are connected devices in every single sector i.e., Healthcare, smart homes, transportation, smart industries etc. With the exponential increase in the connected devices in the past few years, the concern for security threats have been recognised. The IoT creates a network of devices which generate, and share data based on their unique environments. Each unique ecosystem contains sensors, embedded devices, connectivity, and decision making with minimal user interaction. Hence, the need for securing the information in the exchange of data is crucial in protecting people and devices, as a small shift in its behaviour can change the entire working of the system [16].

The Figure 3 below shows the various sectors of the IoT which offer wide range of services. Each sector on its own provides various devices supporting each industry and increasing quality of lifestyle. These devices are compatible and share information which were acquired via various sensors.

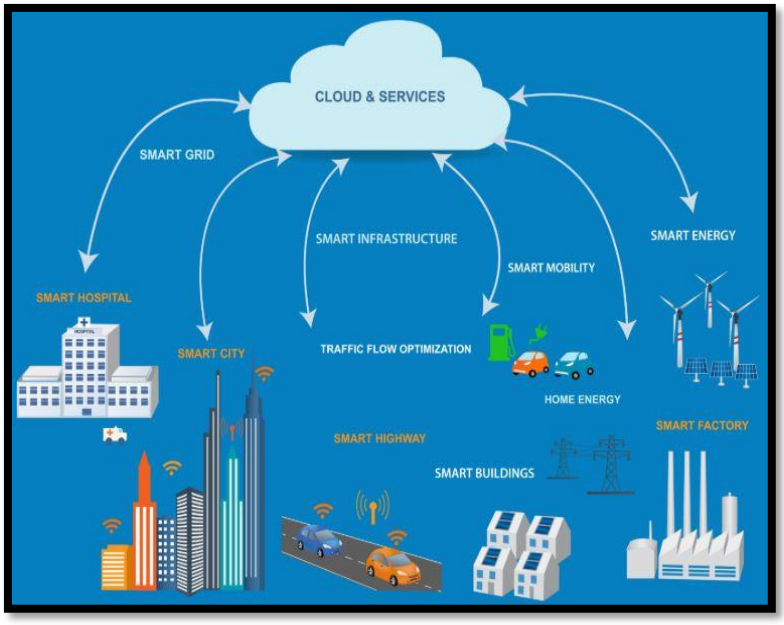


Figure 3 Sectors of IoT

An IoT application can be classified as a multi-layered system as each layer serves a different purpose. The layers can be defined as physical layer, network layer, abstraction layer, application/user layer. The physical layer provides the edge devices like sensors, smart monitors, wearable devices etc which help in acquiring the data from surroundings. The network layer consists of the transfer of information/data and can also be referred as a communication layer. The abstraction layer is a cloud – based layer which accumulates the data received from the devices and hence contains the information from the raw data received. The last layer is the application/user layer where the data analytics is performed. This will help make sense of the data and can be customised for services to subscribers[17].

This chapter presents a generalised information on Security Properties for IoT Devices, Attacks that affect the IoT architecture, Impact of these attacks. Since the work in this thesis is based on ICMetrics technique, which is extensively used for encryption, so we also explore the attacks on cryptographic system, list legacy cryptographic techniques used, review circumstances of its usage and resources needed. Alternative to the conventional cryptographic techniques we also review PUFs and where they are used, its computational cost, actual additional cost for installation and build a comparison with ICMetrics technique. Lastly, previous work on ICMetrics is reviewed to completely understand when, where and how this technique is applied previously to learn from it.

2.1 SECURITY PROPERTIES:

1. Confidentiality: This property ensures the access of the information be only given to the users allowed to view the data and be protected from eavesdroppers. As it may be, that the data is sensitive in nature and only be available to the authorized devices and personnel.[18, 19]
2. Integrity: The property ensures that the data after transmission or storage, resembles the original information without unapproved changes[18]
3. Availability: Ensures the availability of data and its services each time its needed i.e., the WSN's are accessible for communication of data and each of the edges use network resources. Hence are available for transmitting and receiving data even in the

presence of DoS attacks. These services are essential for any IoT device to survive and its data available to the authorized users. [18, 19]

4. Auditability: The services/actions performed by the devices should be documented from time-to-time and be available to look-up and verify.[18, 20]
5. Privacy: It ensures the secrecy of the data and it only be available and controlled by the authorized users. It follows the privacy guidelines and facilitates users to manage their personal data[18, 20].
6. Accountability: The property name suggests that any users of the system must be held accountable for their individual actions[18, 20].
7. Non – repudiation: During any communication between edge devices or users of an IoT system, an entity cannot deny that the message was initiated by them. Hence, any device or user cannot deny the actions executed by them[18, 21].
8. Authorization: A property that suggests the clients should accept the necessary rights of any IoT device and system to access its practices and resources.[18]
9. Authentication: Any user of a system whether it be an entity (human) or device in a network should be able to verify its identity with each other. The authentication is done while transmitting or receiving the data and should be able to validate the source of data. Hence, authentication, one of the properties of a secure IoT system[18, 19]
10. Nonce: A lot of key generation and encryption schemes rely on random bit stream creation which is fuelled by generating random numbers. A nonce value is the unique value sent with each transaction. This is used for altering and accountability prevent attacks[18]
11. Routing security: A secure point – to – multipoint, multipoint – to – point, point – to – point communication via an SRPL (Secure Routing Protocol) to protect the data. The protocol, SRPL applies a concept of rank threshold and hash chain authentication. This can prevent some routing attacks encountered by the network[18].
12. Software Security: The security issues with software i.e., attacks on the code, exposing system's vulnerabilities etc are the main issues. These can be prevented by using a secure software development process like SSDLC (Secure Software Development Life Cycle) [22]. Paper [22] also reviews other models of software development life cycle based on security and also works in developing its security. The software development process consists of assessing codes, defining security architecture, accumulating

security needs etc. The SSDLC process uses SSD (Secure Software Development) process.[18]

2.2 SECURITY IN IOT ARCHITECTURE

Rapid growth in the field of IoT is observed in the past few years. This swift development of the applications comes with a cost i.e., developing devices cheaper and quickly without taking necessary precautions for security of the data. The Figure 4 below shows the attacks on each layer of an IoT system and how they disrupt the working of system.

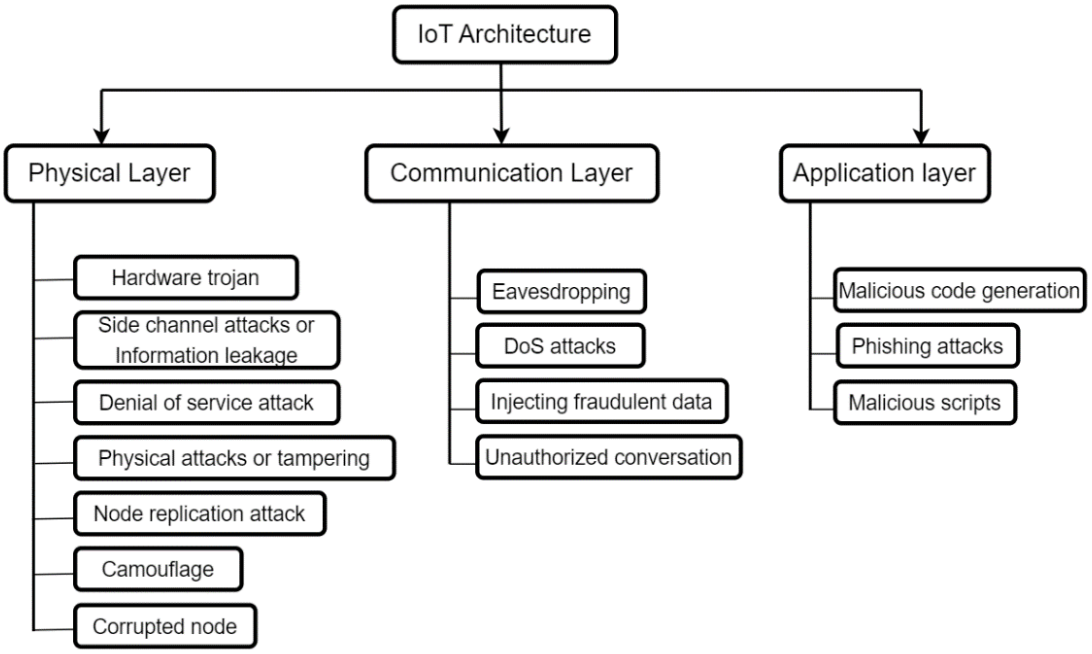


Figure 4 Security Threats on IoT architecture on each layer

The concern is with the hardware and software security of the system. As mentioned above each IoT device is a multi – layered system, each layer is prone to attacks that take advantage of vulnerabilities and gain access. While understanding the security issues related to Raspberry Pi is very prone to physical layer attacks, communication layer and application layer. While recognising the security issues of using Raspberry Pi on various layers, this thesis focuses on addressing the security issues based on the communication layer to prevent a widespread attack on the full network. To address these, we need to be aware of the types of attacks that can be mitigated with the proposed ICMetrics based approach on the

communication layer of the system. Hence, we describe the security attacks encountered at communication layer of the system:

1. Communication Layer*:* The network layer and abstraction layer can be grouped as a communication layer (data transfer layer). The attacks on this layer are basically about getting the access to a network as a user and capture the details of the system like cryptographic keys, information being transmitted etc.
    a. **Eavesdropping:** This is also known as sniffing attack. As it signifies listening to confidential exchanges over the communication channels. This can lead to unencrypted data leak like passwords, username etc[18, 23, 24].
    b. **DoS Attacks:** Jamming the radio signal and stopping the transmission is one of the most common DoS attacks. There are two kinds of jamming: 1. Continuous Jamming (Complete blockage of data transmission) – blocks all data transmission, 2. Intermittent Jamming (Block all signal periodically) - the transmission occurs in a periodical manner to decrease the performance[17, 25].
    c. **Injecting Fraudulent Data:** The attack suggests the attacker introduces the counterfeit data in three ways: 1. By insertion – introducing new data packets (which seems legitimate) to the network which contains malicious information and has ability to regenerate, 2. By manipulation – This involves intercepting the packets and modifying its data, 3. By replication (replay) – the previously sent/routed packets are resent repeatedly between two points/nodes [17].
    d. **Routing Attacks:** Routing attacks in the communication layer are based on spoofing, redirection, misdirecting a packet to a new destination by changing the routing details in the network or dropping the packets too. This is done to ensure trafficking in the network or loss of data[17, 18].
    e. **Unauthorized Conversation:** In any sensor-based network, edge devices share the data across the nodes or access other node's data. Each IoT system comprises of both secure and vulnerable nodes where each node should only communicate with subset of nodes it wants to share data with. For e.g. If an unsecure node shares (gets) the data (from) with every node, hence an adversary can easily take the network down through an unsecure node.

These attacks listed above largely affect Raspberry Pi devices and the scale of the attack depends on where the devices are deployed in the network. These attacks can be mitigated with the security model proposed in the thesis with potential to identify the Raspberry Pi devices based on their individuality in the way they utilise the resources for a task.

2.3 IMPACT OF ATTACKS ON IOT DEVICES:

An extensive study was performed by CSES (Centre for Strategy & Evaluation Services) for consumer IoT devices for DCMS (Department for Digital, Cultural, Media & Sports) is presented in [26]. They presented a study that examines the scale and nature of vulnerabilities in the consumer side of IoT by performing analysis in four areas, of which two of them relevant to our research are consumer vulnerabilities in IoT devices, its impact, and future risks. They performed an extensive literature review using 80 sources and to support the analysis 23 interviews were performed with stakeholders. These stakeholders are representatives in businesses using IoT devices or consumers or people with industrial associations. The interviews were done telephonically, and two online surveys were conducted using business owners and consumer target groups. The types of attacks on communication layer of IoT devices is mentioned in section 2.2. Hence, this section discusses the likelihood of types of vulnerabilities exploited, damage caused and their impact.

The vulnerabilities are utilized by the hackers to not only damage the devices but use the access for more attacks to gain better access to the information or device functions. The Figure 5 below shows the graphical representation of which vulnerabilities in consumer based IoT devices are most likely to be exploited.

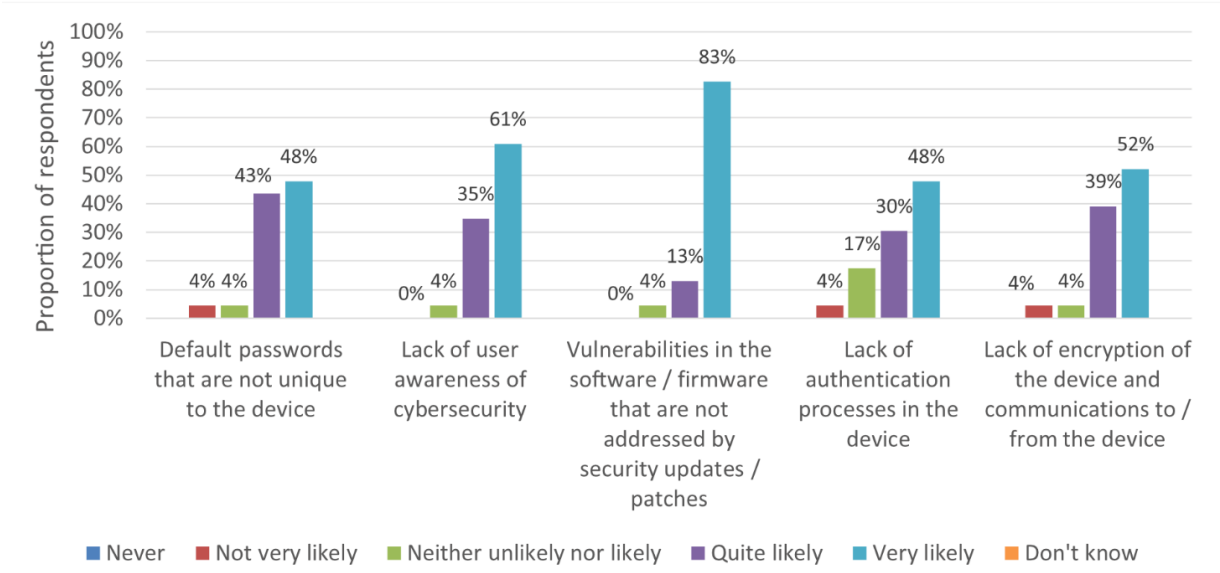


Figure 5 Perceived likelihood of vulnerabilities exploited in IoT devices by CSES using surveys conducted[26]

The survey done and literature reviewed by CSES states these are the five criteria according to the business owners and consumers which vulnerabilities are exploited. These vulnerabilities are also explored based on impact of these attacks. The cyber-attacks can be harmful to an individual with a potential of minor impact to serious data breach or financial loss. The interviews conducted by CSES with cyber security representatives and tech who highlight the potential impacts at three levels i.e., Personal level, Business level and National level. Using the surveys and the data collected by CSES, assessment of the perceived potential impacts by the attacks on the consumer based IoT devices. The Figure 6 below shows the statistics of how the impacts on the IoT attacks are perceived in decreasing order.

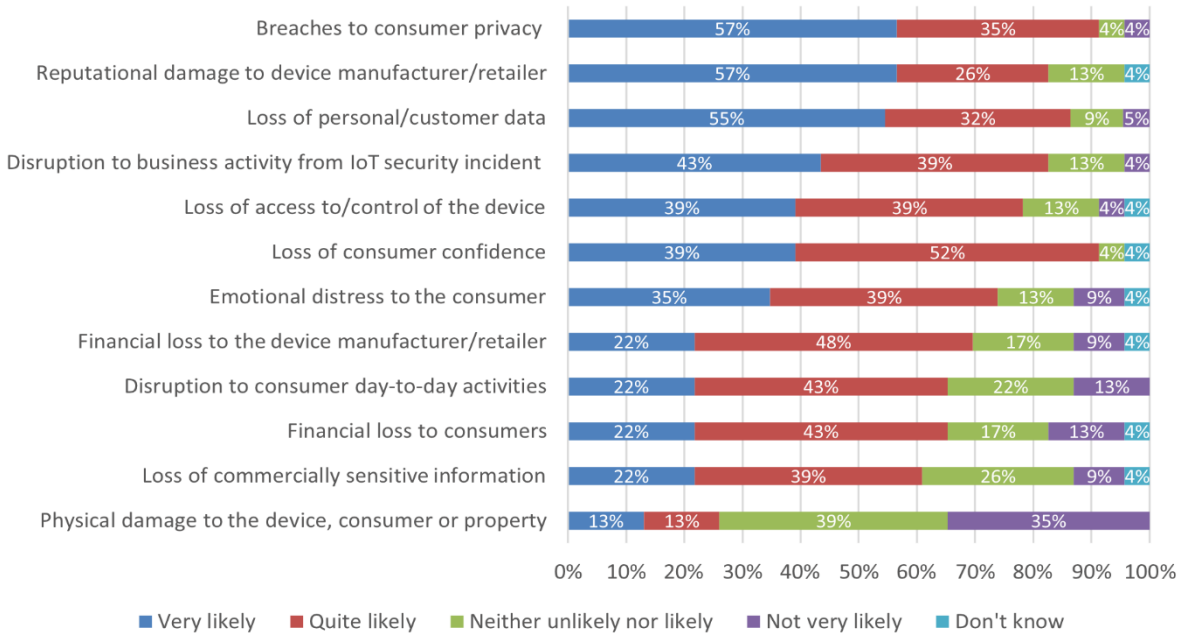


Figure 6 Perceived likelihood of impacts by an attack on IoT devices by CSES using surveys conducted[26]

These are the impacts of an attack on IoT devices. There is also a need to understand how much damage is caused by each of the attacks. Each of the attacks as mentioned previously can have any of the three scales of impact i.e., Personal, Business and National level. The Figure 7 below shows the impact of the IoT attacks with respect to the damage perceived, this represented graphically. This analysis is derived using the surveys conducted by the CSES.

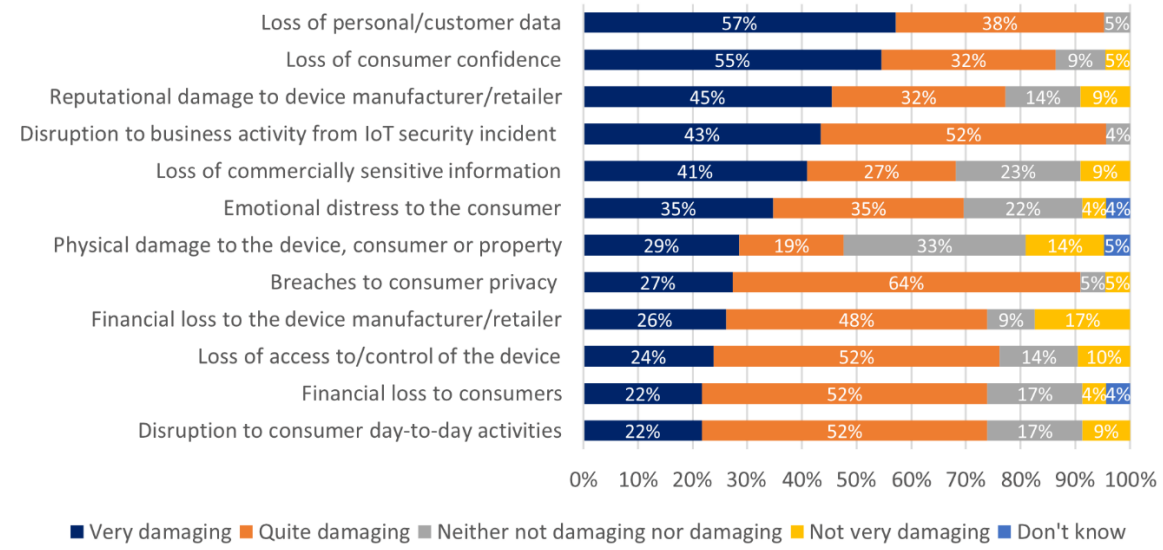


Figure 7 Perceived likelihood of damage caused by an attack on IoT devices by CSES using surveys conducted[26]

The IoT devices are used at a very large scale and the usage is set to increase exponentially, which leads to increase in scale of probable attacks caused by hackers. Hence, there is a need

to keep focus on cyber security measures to be taken for the devices to protect individuals and their data from potential attacks. The sections below recommend some security techniques that can be used on IoT devices to protect the data and their function to minimise the damage in case of any attacks.

2.4 ATTACKS ON CRYPTOSYSTEM:

The attack on cryptosystems suggests the ideas of exposing the confidentiality of the keys used for secure exchange. Hence, an attempt is based on exposing the keys of any system to compromise them. The cryptography attacks are different in a way as some of these would not get detected after the keys are replicated. Hence, all the malicious activity disappears after the keys are intercepted or if the system was ever infected.

There are number of cryptographic attacks are described in [27-29]. As there are various types of key interception techniques, implementing it is a complex ask. Hence, each attack is chosen by analysing the weakness of the system. The following list are potential techniques used by adversaries on an IoT system to capture the keys that lead to secure communication.

- A cryptosystem can be conquered by using many methods like man in the middle attack, brute force, active or passive attacks [30] etc. To conquer these attacks, techniques like adding salt, increasing the key size etc, can be used.
- A software can be infected by a malware that can intercept the key and convey it to the adversary. These kinds of attacks can be difficult to expose as the user might not be aware of its presence. Paper [31] describes a 'black-box' system for using a cryptographic technique exposes the system to various attacks. A 'black-box' system means trusting the internals of device entirely and its negative impact on cryptography.
- A possible attack on a cryptographic system is assuming false identity of a person with a public key and a claim that it belongs to them. A certificate is issued by the authorities as evidence to ensure the identity is not falsified. There have been cases where the experts have released the certificates to the counterfeiters, which has led to a certification revoking list on the browsers.

- The algorithmic complexity is the key aspect of any cryptographic algorithm. Importance of this is understood during the key generation as it will be easier for someone to intercept it. The paper[32] describes the security provided private – public key combination technique for encryption and decryption to provide best possible security.
- Manipulation, eavesdropping, persuasion etc are some of the few methods to get access to the keys from respective owners. Hence, it is important to maintain the secrecy of the keys.

These are the main types of attacks that can be used by an adversary to attack the cryptosystems of any device. These are the basic techniques to intercept keys, detailed version of the techniques is present in [21, 27-31, 33].

2.5 CRYPTOGRAPHIC TECHNIQUES:

The cryptographic techniques like encryption, secure transmission protocols are used to keep the data secure during transmission and are effective against routing attacks or eavesdropping during communications. There are numerous encryption methods like RSA, AES, MD5, SHAI etc are used to secure the data transmission over a network. The techniques to encrypt and decrypt data for the traditional wired network and not for the latest edge IoT devices. These are pocket-sized and battery operated which limits the resources needed like memory, battery, and high-power processors to run these algorithms, hence light-weight cryptographic techniques are considered for these devices. The main aspects of cryptography are confidentiality, integrity and authentication[34]. Hence data transfer is performed in a way which makes sense only to the sender and the receiver[34]. There are two types of cryptographic algorithms i.e., Asymmetric and Symmetric cryptography.

1. Lightweight Cryptography: The cryptography technique is considered for the devices with limited resources at its disposal, i.e., lesser computational power, battery etc. To develop the lightweight techniques for the IoT devices, a clear assessment of the hardware and software capabilities are performed. This evaluates the energy expenditure, time – taken to encrypt the data and capabilities of the processor[34]. The paper [35] presents a range of lightweight cryptography techniques that covers variety of techniques to secure IoT applications from

Fiestel Networks (FN) and Substitution Permutation Network (SPN). The FN and SPN are significant classes of symmetric block ciphers[36].

- AES – 128: A widely used lightweight block cipher. The algorithm is based on the SPN structure. The 128 bits block size of the AES be used for three different key sizes 128 bits, 192 bits and 256 bits[37].
- HIGHT: A lightweight cryptography technique suitable for low – resource hardware. For a 128 – bit key length it uses a 64-bit block and uses simple arithmetic operations like addition and logical operations like bitwise rotation, XOR etc. These are used by simple RFID systems or small IoT devices[38].
- LED (Light Encryption Device): The encryption has a block size of 64 – bits and has multiple key sizes 64, 80, 96 or 128 bits. It provides a good performance for software aspects[35]. The cipher can be successfully implemented on a miniature device and uses a 4 – bit PRESENT S box[39].
- Speck: An FN based technique to best work on both hardware and software but best adapted to be used on software execution on embedded devices. The technique uses simple logical and arithmetic operations. The logical operations consist of bitwise AND and simple round function left circular shift bitwise XOR[35, 40].
- Simon: It uses an FN structure, which provides a decent diffusion between non – linear and linear confusion operation. The composition of nonlinear function is based on two rotations, a bitwise AND or a bitwise XOR[35, 40]. Works effectively on hardware and software but commendable results on hardware implementations.

Paper [35, 39] describe more lightweight cryptographic techniques available to be used on constrained IoT applications.

2. Asymmetric Cryptography: The Asymmetric cryptography is also known as public key cryptography[34]. The technique uses two different keys i.e., a pair of public and private keys for encryption and decryption[21]. Each person has their own set of public and private keys where the public keys are available to anyone. During the data transmission, the sender encrypts the data with the recipient's public key. The decryption is performed by the recipient using their own private key[34]. The system does not require an exchange of keys amongst the sender and the receiver and hence can protect the integrity of the data during transmission. The following are the popular asymmetric algorithms:

- RSA (Rivest - Shamir - Adleman): An asymmetric algorithm developed by Rivest – Shamir – Adleman[41] which uses a block – cipher-based system. The algorithm is used to encrypt the session key which is used for secret key encryption (protects the message reliability) or message's hash value (its digital signature) [42]
- Diffe – Hellman: The algorithm is also known as a public key exchange algorithm[21] also used commercially. The importance of the algorithm depends upon keeping the secrecy of the key and secure the key trade amongst the parties.
- Elliptic Curve Cryptography (ECC): The ECC uses elliptic curves where the coefficients and variables are limited to a finite field[21]. It has relatively smaller key sizes but offer comparable to RSA and other Public Key Cryptographic techniques. The algorithm was designed for devices with limited resources. [42]

3. Symmetric Cryptography: The Symmetric cryptography uses a single key of encryption and decryption. The data is encrypted with a secret key before transmission by the sender and would be decrypted using the same key by the receiver[34]. This is also known as Secret Key Cryptography. The following are the symmetric algorithms:
    - Data Encryption System (DES): A block cipher-based encryption system invented by IBM in the 1970s. The block – cipher encrypts the data in blocks, where the size of each block is 64 – bit for a 56 – bit key and the rest of the bits are used to check parity[21]. It was specifically designed for fast hardware and slow software implementations.
    - Advanced Encryption System (AES): An AES is a block – cipher technique widely used in business-related applications. It uses 128-, 192- or 256-bit block size and key sizes can be 128, 192 or 256 bits. The technique is based on substitution permutation computing[21]. AES uses one permutation and three substitutions, in total uses four stages per round.
    - Triple DES: The technique is a variant of DES which applies three different 56 – bit keys for making three encryption or decryption goes over a block of data.

The main cryptography techniques can be classified into three categories namely, lightweight, symmetric, and asymmetric techniques. There are many more subcategories of cryptographic techniques on each of these. Papers [21, 34, 35, 42] have extensive details of the

cryptographic attacks on the devices. The IoT devices are resource constrained devices as they are meant to support one application with various capabilities at a time. Hence, most systems that attempt to use cryptography, use lightweight cryptographic techniques or hybrid cryptographic techniques to let the security overhead running on the limited resource device in efficient manner.

Authors of [43] reviewed a few cryptographic techniques on Raspberry Pi 3 benchmarked with respect to Arduino results which were collected during literature survey. These are compared based on power consumption, time, energy cost. The paper compares hashing, digital signature, asymmetric and symmetric cryptographic algorithms. They only compare the power consumption and throughputs (data encrypted per second with one joule of energy) using the security algorithms, there is no mention of running the security overheads with the applications. Authors make various conclusions based on their experiment results i.e., firstly, the best symmetric algorithm for RPi3 is AES, where AES-128 produces higher throughput and gives better energy performance than AES-256, AES-192. The DES algorithm works three times faster than 3DES and works 65% slower than the AES. Secondly, the hashing algorithms exhibit behaviour like the symmetric algorithm where SHA-1 and MD5 are much faster than others but these are not recommended due to security flaws. Thirdly, the ECDH algorithm works better than the RSA for key – generation/exchange. Lastly, amongst digital signature and verification algorithms, ECDSA performs better than DSA & RSA and RSA is better for verification. While comparing the performances (for digital signature and verification) using RSA, DSA and ECDSA, the ECDSA performs much better on average.

Another paper [44] discusses the comparison between the AES (Algoritma Advanced Encryption Standard) and DES (Data Encryption Standard) based on their performance in terms of speed and memory usage. The AES algorithm encrypts the data faster and uses more memory space. The AES uses the binary and hexadecimal numbers for encryption whereas DES uses binary. The authors also conclude AES to be the better algorithm as it supports 128 – bit length key whereas DES has 64 – bit key length.

Similarly, paper [45] compares the lightweight cryptographic algorithms built for low resource embedded devices and compared each of their performance on two platforms i.e., windows machine and Raspberry Pi 3. Since these two devices are majorly different in terms resources available, it is crystal clear that algorithms perform better on windows as RPi 3 is a low

resource device compared to a windows machine. Similarly in [46] the authors compare the lightweight cryptographic techniques on Macbook Pro and Raspberry Pi 4. Even though the comparison of implementing these algorithms is on these two devices, they also compare the algorithms efficiency of each algorithm on respective devices individually to find the most suitable technique to be used on resource – constrained devices. The authors conclude that the lightweight crypto techniques are superior to the traditional techniques while working with resource constrained devices in terms of power consumption, speed, and efficiency. The paper specifically mentions the SPECK and SIMON type ciphers have 128-bit keys and work efficiently using python for implementation. The main concern for the cryptographic overheads is not only the encryption time but also the time taken to fully execute the algorithm and the energy consumption during that execution. **Since, the work done in the thesis solely focuses on the Raspberry Pi devices and developing techniques to identify Pi based devices which in-turn can be used for securing them.**

2.6 ATTACKS / SECURITY ON RASPBERRY PI

This section shows the vulnerabilities in Raspberry Pi devices that have been exploited in the past and techniques used previously to prevent or intercept the attacks. The section also provides information on which techniques used to secure these devices and strategies to protect the data leaks. The work presented in this thesis is majorly based on exploring Raspberry Pis for device identification which can be used to secure these devices. Hence, this section describes the previously tracked attacks on Raspberry Pi devices and how extensive its damage can be.

The notion of developing cloud-based smart devices comes with the security issue when connecting to a network. This section presents the examples of security issues or malware encountered by raspberry Pi devices when connected to the internet. The Raspberry Pi is a simple versatile embedded device which can host a variety of applications. Hence, the applications are cost effective and should be in a secure environment.

The cloud – based edge platforms are exposed to thermal attacks. Paper [47] studies the thermal attacks on a Raspberry Pi 4 by using a stress-ng [48] tool to perform stress test on linux – based systems. The authors perform three experiments, to evaluate the effect of

thermal attacks. The first experiment measures the temperature of the Pi while the CPU attacks the system with various frequencies. Second experiment focuses on cooling strategies for edge devices. The last experiment measures the effect of attacks on power consumption and temperature, where the effects are maximized when the device is working at full power.

The Denial of Service (DoS) attack is also one of the most common attacks on embedded devices. The authors in paper [49] have a Raspberry pi based smart farming system, which remotely monitors various aspects of a farm like soil moisture, delivering pesticide, crop status etc. The attackers exploit the systems weaknesses to interrupt normal processing of the system and intercept the data. The paper demonstrates the DoS attack (Wi-Fi Deauthentication Attacks), to disrupt the Wi-Fi signal to disrupt the data transmission from sensor to Pi and unable to upload the data and hence, brings down the system.

The authors of paper [50] performs a vulnerability analysis on raspberry Pi model B+ for cold boot attacks. These attacks are usually performed on variety of computing devices which attack the RAM of the device by bringing the device down to extreme cold temperature (by using the Air Duster in this case) setting before switching off the power of the device which leaves the data on for numerous seconds and hence can be retrieved during a cold boot attack. To perform the susceptibility to cold boot attacks on Pi, the authors analysed the booting sequence on raspberry Pi to check the process that initialises the RAM. The Raspberry Pi has a ROM boot for initial booting sequence which reads the second bootloader from its SD card and lastly, the third bootloader from the SD card to SDRAM. This leads to reading the config.txt and hence running the kernel.img . The adversary after lowering down the RAM temperature, replaces the victim's SD card to recover the RAM data from original program.

One of the major concerns of connecting any device over the internet is the susceptibility of accidentally downloading malware or getting attacked by it. The authors in the paper [51] discuss various malware and how it affects/infects the devices. Here the authors set up a honeypot system to capture them from the network. A honeypot configuration works with cowrie [52] setup on an Amazon AWS server. It is a fully interactive Secure Shell (SSH) and Telnet honeypot. These protocols are used for remote access, are usually already configured in most IoT devices and hence used to attack the networks and generate Mirai botnet [53]. The Mirai botnet is a malware that affects the IoT devices and bound them as DDoS botnet. The ports that are open to remote access are used for these

attacks, usually get access by trying default credentials. The largest DDoS attack was performed by Mirai botnets in history[54]. Authors of [51] also caught a malware with their honeypot called 'UNIX_PIMINE.A' which was originally designed for Raspberry Pi by Trend Micro [55]. Here the static and dynamic analysis are performed to analyze the effects of the malware on the devices. The static analysis reveals the malware after infection, replicates itself and kills the ongoing processes on the system. The dynamic analysis depicts after the malware was executed, the Pi reboots and creates a backdoor on the device. The paper also describes the various malware caught by the honeypot system.

The IoT devices deployed sometimes have limited resources to implement any Intrusion Detection Systems (IDSs). The lightweight detection system is implemented by authors in paper [56] which includes a machine learning based IDS that uses different feature sets to detect each kind of attack. The system detection is lightweight and hence, able to execute on a Raspberry Pi. They also suggest a new feature selection criterion called correlated – set thresholding on gain – ratio (CST – GR). The research focuses on three kinds of attacks i.e., DDoS attacks, Probing attacks (reconnaissance) and Information theft attacks.

- DDoS attacks: These are used to disrupt the services for the users and are launched in combination with many conceded hosts also known as bots.
- Probing attacks (reconnaissance): It uses malware to gather the target data using remote scanning. These are also categorized into subclasses namely, port scanning and OS fingerprinting.
- Information theft attacks: The attacker tries to steal sensitive information which could be further categorized into keylogging or data theft.

Here the tree-based Machine Learning (ML) classifiers like J48, Random Forest (RF), Logistic Model tree (LMT) and Hoeffding tree (VFDT) implement the detection system with around 99.4% accuracy with each classifier.

The Raspberry Pi based IDS's focus on several types of attacks encountered by the Pi. The table below shows the gist of the detection systems on Raspberry Pi and the kinds of threats they encounter.

References	Pi - Model	Type of Threats	Detection Tools
Zitta, et al. [57]	RPi – 3	Port Scanning	Suricata
Fu, et al. [58]	RPi – 3	Replay, Jam and Fake attack	Automata Theory (FSM)
Kyaw, et al. [59]	RPi – 2 B	Port Scanning, Synchronization (SYN) flood, Address Resolution Protocol (ARP) spoofing	Snort IDS/ Bro IDS
da Silva Cardoso, et al. [60]	RPi – 3 B	Port Scanning, ICMP Flood, UDP Flood, SYN Flood	CEPIDS
von Sperling, et al. [61]	RPi – 3 B	Man In the Middle attack, DNS cache Poisoning, DoS	Sniffing Software (Python and DPKT library)
Visoottiviseth, et al. [62]	RPi – 3	SQLi, Evil Twin, DDos/Dos or Password based attacks	Snort (Signature or Anomaly based detection)
Visoottiviseth, et al. [63]	RPi – 3 B+	DDos/Dos, Bruteforce password, SQL injections, XSS	SPIDAR, Snort (Signature or behavior-based attack with ML)

Table 1 Summary of types of attacks and detection tools used on Pi.

The papers [47, 49-51, 53, 54, 56-63] have extensive details about the attacks on Raspberry Pis. Each paper discusses the attacks that a Pi is prone to and offers a few countermeasures to detect the attack before the information is stolen or the device is down. Hence the security measures and the attacks, specifically on Raspberry Pi, are discussed in this section. A few research studies have a Machine Learning (ML) based intrusion detection system / recognizing attacks and protecting data being transmitted. The Malware affecting Raspberry Pis in this section affects the internal working of the system, acquires space, and uses default credentials to gain access to a single device or the network of devices. Therefore, an idea of utilizing physical characteristics of devices arises from the biometrics used for people recognition. This idea of utilizing Physical characteristics for generating unique identification of devices gives rise to the concept of Physically Unclonable Functions (PUFs). The next section reviews the previous work done and how the characteristics are utilized to generate device identity in various situations for increased security.

2.7 PHYSICALLY UNCLONABLE FUNCTIONS (PUFs)

The PUFs are presented as a lightweight and robust hardware – based security solution for edge devices. The PUF technique is lightweight compared to classic cryptographic techniques and works best for resource – constrained devices [64]. Initial PUF based authentication works in two phases i.e. enrolment and authentication [65]. During the enrolment phase the PUF the PUF circuit connected to server where the server challenges and the PUF circuit responds. This process of the server challenging and receiving corresponding response by circuit, is called Challenge Response Pair (CRP). These CRPs are stored by the server and during the authentication phase, the server sends an arbitrary PUF challenge to the device, where device (where the PUF circuit is connected after enrolment phase) responds which should match the response stored in the server, Figure 8 below shows the working of a PUF circuit. A recent survey conducted by authors in [66] provides literature on various types of PUFs that have been previously used namely Ring Oscillator PUF, Arbiter PUF, Optical PUF, Hybrid PUF, SRAM PUF, MRAM PUF etc. These are hardware based PUFs that generate a tangible measurement of any device/sensor which are specific to a particular situation of the device. These are like biometric features of a human[67]. These are derived from the physical properties of a device and cannot be derived from cryptographic reductions. PUFs of any device are extremely useful as they are nearly impossible to clone as these are particularly tied to the physical nature of the device. As PUFs are developed using the physical traits of any device, if the device is tampered with to gain access, the properties of the device will vary and hence the PUF related to it. This prevents the user from being accurately authenticated.

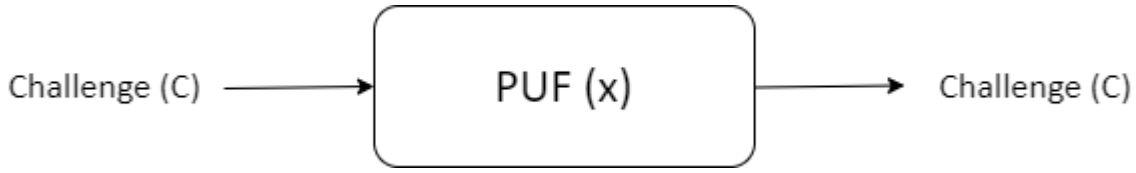


Figure 8 PUF circuit Challenge Response Pair (CRP)

When a PUF is called upon with a question, it provides a user with an answer which is based on the unique persona of the device [68] hence producing a unique output as a response from the question or information requested by the device. Any device characteristics to be used as a stable PUF has following properties:

- The characteristics used should be unique.
- It should have stability (i.e., produce the same output whenever called upon) and is unpredictable.
- It should yield the same results when repeated for a device.

Any board or device undergoes stress during the manufacturing process which yields unique sensor bias or offsets per device. These are used as identifiers of a device. Paper [69] proves that while connecting and soldering sensors on a board, a stress is introduced on it which is unique per device and affects it uniquely. Hence, it becomes a distinguishing property of that device. The robust identification or signature of a device here is done by generating a sensor fingerprint by using the calibration data from accelerometer of the system and analyzing data from speaker – microphone system in frequency domain. The authors also examine the sensors that are commonly available on mobile devices and which of their characteristics can be used distinctively. Similarly, [70] presents the identification of the system via the SRAM memory used as a PUF for a device. The characteristics of SRAM memory in the device (from 180 nm to 65 nm) are studied during powering – up of each device. The authors analyze the power-up sequence of the devices provided, the external conditions like environmental temperature, applied voltage are noted. The PUFs hence, can be used as an improved securing technique instead of traditional cryptographic techniques and are also used for safe key storage, authentication, key zeroization[71], generating keys[72] etc. Papers [67, 70-72] contains an extensive discussion on PUF based applications and various techniques to generate them.

The PUFs are an alternative to traditional cryptographic techniques as they are lightweight and highly compatible with IoT devices as the computational overhead is lesser than traditional techniques. Even though PUFs are supposed to be unique as they are hardware – based, they are highly dependent on the environment they are in. As described in [66] there are various types of attacks that can be used against PUF system. These can be either active or passive adversaries and consider two kinds of scenarios i.e., When communication between devices is intercepted (passive) or Physical access is gained by an adversary (active).

Various attacks like Brute force, non – invasive, invasive, semi – invasive and Machine Learning attacks. Most popular attacks in case of the effectiveness are Machine Learning based attacks on PUFs. These attacks are passive attacks and intercept the communication between the device and server. The adversaries use Machine Learning (ML) techniques to predict CRPs on the servers. Any Machine Learning based attacks work on a presumption that the adversary has gained access to the subsets of CRPs which then are used to generate a mathematical model from the data for which ML models are effective tool.

Paper [73] investigates and reviews ML based attacks on various types of PUFs. For simulating the ML attacks, the CRPs were collected from two sources: 1) FPGA and ASIC (Application Specific Integrated Circuit) based silicon CRPs. 2) Using Pseudorandom simulations with additive delays. They examined the XOR Arbiter, Arbiter, Feed – Forward Arbiter, and Ring Oscillator based PUFs. The authors concluded that strong PUFs that are investigated, 'within a certain size and complexity' can be predicted with 99% accuracy. The ML models used are Logistic Regression (LR), Support Vector Machine (SVM) and Evolution Strategies (ES). Similar experiments are performed with the exact same ML models in [74], but the only difference is the sources the CRPs were collected from.

A recent and extensive survey is performed by authors of [66] that shows the ML based attacks on various PUF based architecture. They also provide a few solutions that can be used to prevent these attacks. One of the methods is to increase the bit length of an Arbiter PUF. However, a PUF – based circuit is designed to maintain challenges to a limited amount in a certain amount of time for every single authentication process. The CRPs should never be used twice to prevent an ML based attack, meaning considerable number of CRPs can help prevent the predictability in PUF generation. Similarly, another method to prevent these is proposed in paper [75] by using reconfigurable design where the numbers of CRPs are increased without affecting the computational overhead whereas paper [76] uses obfuscation for challenge response protocol using random number generator, control unit and a PUF chip. This does not use conventional cryptographic techniques to prevent increased computational cost.

These attacks are based on getting the CRPs and building a model to predict CRPs to intercept communication line. An alternative technique to this is the popular ICMetrics (Integrated Circuit Metrics). The ICMetric is a cryptographic technique comparable to the

PUFs. While PUFs only utilize the hardware characteristics of the system, the ICMetric technique can utilize the data from the software running on the system in combination with its effects on hardware of the device. This in turn generates a unique identifier of the device which is used in authentication / identification or encryption of the system. As this technique does not support storing of keys on the device, which prevents the data interception. The next section describes the existing literature on the ICMetric technique.

2.8 INTEGRATED CIRCUIT METRICS (ICMETRICS):

The ICMetric technique is a cryptographic technique can be used to secure and identify IoT devices. The traditional cryptographic algorithms involve the storage of encryption/decryption keys. The probability of the keys intercepted by adversaries increases when the device is compromised. Hence, the ICMetric technique works as an alternative option to storing the keys to ensure the security of the data even when the device is compromised. The attackers compromise the device in attempt to intercept the secret key to decrypt the data. The sections above describe the wide range of attacks on cryptographic systems and Raspberry Pis to gain access to the sensitive data generated by the IoT devices.

The ICMetric technology [77-80] is not only used as an alternative method of cryptography to store - key cryptography but is also used to identify devices by performing extensive analysis on the features of each device. It eliminates the concept of storing the keys which prevents any access of information to the adversaries. This technology is comparable to any biometric systems as they authenticate / identify people based on their unique features i.e., fingerprints, facial recognition etc. Similarly, ICMetric tech uses device features to generate a unique device signature / fingerprint to identify itself. The technology uses distinctive properties by using features from each device and produces an ICMetric.

Previously, the ICMetric technique is used to add a layer of security on off-the-shelf embedded devices by using lower – level features of the embedded devices. The idea behind using low – level device features is the complexity to replicate the functioning of the device by an adversary which eliminates various kind of attacks like identity theft, key generation etc. Hence using the data from the devices to generate a stable ID from the working of the devices and to further reduce the vulnerabilities by generating keys which are not stored in

the system. Various techniques have been employed in the past decade using ICMetrics as a basis to increase security in vulnerable devices.

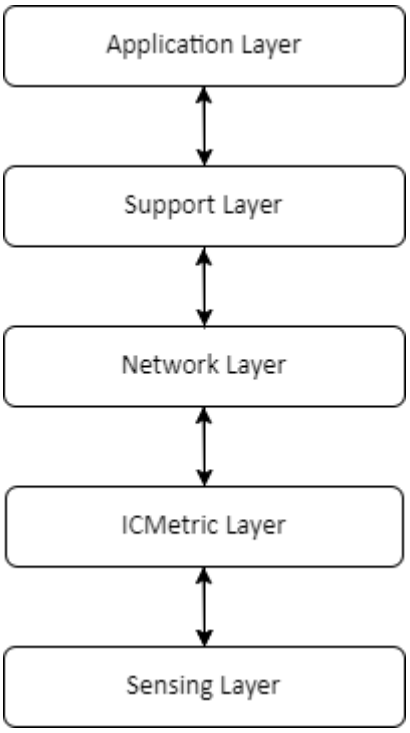


Figure 9 ICMetric based IoT system [81]

Authors in [82] explore the Program Counter (PC) as a potential for ICMetric by using the tracing methods namely single stepping and sampling. Here, they generate the program profiles using each tracing method with the raw data obtained and conclude that PC values need further analysis to be used as an ICMetric feature to build a system. Similarly, in [83] employ various algorithms on same hardware device to simulate its performance. They make use of the Program Counter (PC) values as ICMetric features are extracted while running various algorithms. The paper concludes that the sample size of the features affects the ICMetric system ‘s performance. The main purpose of the collected features is its ability to identify the devices uniquely, hence the frequency domain of PC values is explored and considered as appropriate feature to identify the devices effectively.

The characteristics derived through a device can also be used to generate a stable key. The ICMetrics technique offers a stable generation of the encryption keys for the device and offers the easier integration of the new devices to the network. Paper [84] offers a technique to integrate unseen devices to secure healthcare-based communications. The integration is done via performing analysis to firstly confirm the similarities i.e., if the unseen device is the

same as the devices in the network. If the first statement is true, perform the analysis to generate a basis number as an identification for the device by using the feature values. The paper also describes two techniques to combine features (for basis number) labelled as: concatenation and addition technique to effectively identify devices. It also explores the correlation amongst sample size (number of samples used to generate a key) and accurately identifying the device. Not all the features extracted from the devices are useful to generate an ICMetric. The features having certain characteristics are used to generate the ICMetric. Usually features representing some stability and unimodal/ gaussian distribution are used. In certain cases, even though hard to model, multimodal distributions are used and offer excellent information about the relationship amongst the features. This could be used as an advantage to generate the identity of the devices.

Initial works with the technique as documented in [80, 85-88] are based on understanding and extracting low – level circuit features, combining them to generate the stable keys, identifying the devices correctly and lastly, adding a layer of security using generated keys. The paper [85] investigates the need of appropriate number of samples for the features that demonstrate varied distributions. Here the authors use a cluster detection technique by using the Probability density function and detect the number of modes using the peak detection technique by using Peak – Trough detection algorithm[89]. This cluster detection technique is used on multimodal feature sets to identify the distribution of the data and its ability to be linked, as well as identified correctly. If the sample size of the data is smaller, it will result in misidentification of the device as the model would be trained on small data. Hence, it depicts the importance of choosing a larger dataset and analysing the correlation between the features and their modes. Whereas paper [87] explores the methods to generate a stable encryption key. The technique combines the feature values using either concatenation or addition technique, after the normalisation to generate a basis number. This basis number is used to correctly identify the device and converted to gray codes where its stability is tested based on number of samples used to generate it. Generating the keys or a key – pair, based on the unique basis number for each device. Similarly, in [88] authors utilise the ICMetrics approach to extract the low – level device data via reconfiguration with the intention of identifying the device and securing the System-on-chip (SoC) based devices. It uses the address distributions used by the software running on the device. By generating the address

distribution, any abnormal behaviour can be detected after the profiles are generated. Also provides assurance of saving 30% of area by using this feature extraction technique mentioned in the paper. System expects to use the properties of the system combined with properties gained from other significant code data are used to form the encryption keys. In [86], the devices are identified based on each of their behavioural personality. It performs the feature extraction for ten different software assignments and performs the address reports for each of them to identify it by profiling each of these tasks by analysing it. Hence, using systems lower level (hardware and software) features are used to generate unique ICMetric and provide robust security against attacks/tampering of the system.

The practical applications of this technology applied in healthcare sector was first performed on an intelligent powered wheelchair [90] to secure the communication with the database and amongst wheelchairs. By encrypting the transmitted data, the attacks like data interception, device cloning etc required security of the device is achieved. The main purpose of the SYSIASS project is three tiered i.e., developing smart wheelchair with assisted navigation, securing the data transfer amongst them and design a multi-modality human-computer interface (MMHCI)[91, 92]. Here the focus is on the ICMetric technology used here to secure the data transmission to secure the susceptible nodes / points of data leakage. Paper proposes the direct encryption of the data samples acquired from the device that generates its signature / identity. Similarly, papers [78, 79, 93] showcase the implementation ICMetrics technology on the healthcare applications to secure the data transfer by generating unique keys.

Some of the recent work on the ICMetrics is based on exploiting the inbuilt sensors to employ an extra layer of authentication system on the IoT applications. Authors in [94] utilise the in-built accelerometers (mechanical MEMs). These sensors are extremely sensitive in nature meaning, the assembling and hence their baseline/offset values become ‘device specific’, which also includes axis calibration. It also includes the data from tap detection mechanism as input data from the system to develop a multi-factor authentication which in turn can generate a device signature. Hence, becoming a useful data source for generating ICMetric system. The multi – factor authentication includes the encrypted key and PIN (data from tap detection) to be verified. The authors also tested the robustness of the key regeneration technique with 94.5% accuracy. Paper [95] uses the MEMS sensor present in intelligent

wheelchairs to generate ICMetric number and integrate it with the dataset. A network simulator version two (ns-2) is used to generate the trace file from which the raw dataset is extracted. The authors extract bias readings, to generate the readings for ICMetric number generation which is then integrated with the dataset. This data is split into training (60%), testing (20%) and validation (20%) set. The proposed system uses the SVM for training and testing the data. The results are then compared to the system using ICMetric number integrated dataset to system using raw dataset. The proposed system with ICMetrics is concluded to be superior as it yields higher accuracy 99.77% and low error rate of 0.26%.

The use of accelerometer data has been widely used in ICMetric technology as it can be unique since the data depends upon personal usage. This has been observed in various applications previously where sensor data is utilised to generate a unique identifier. Recent work on ICMetrics is described in paper [96] which builds a hybrid security system for Unmanned Aerial Vehicles (UAV) to implement anomaly detection algorithm using ICMetrics. The authors make use of gyroscope and accelerometer data to generate a unique ICMetric number and further integrate it to the dataset. This data is applied to a DNN (Deep Neural Network) to differentiate between normal and abnormal cases and achieved 99% accuracy. Earlier works of the authors in [97] built an intelligent Intrusion Detection scheme based on Magnetometer Sensors in Vehicular ad hoc networks (VANETs) for self – driving vehicles. The technique uses the ICMetrics technology to generate a unique identifier using a magnetometer sensor. The characteristics of a magnetometer are extracted to create a ICMetric basis number in combination with the traditional Intrusion Detection System (IDS) to build a communication system with better security. The proposed system extracts the features that indicate the normal/abnormal behaviour generated using a network simulator. These are used to identify external/internal attacks on the systems. The results from the proposed detection system are compared with the normal IDS. Where the proposed technique gives an average percentage of false alarm rate of 1.21% with the accuracy fluctuating between 98.78% - 99.77% to the normal IDS technique gives a false alarm rate of 12.2% and accuracy fluctuating between 85.02% - 98.45%. Similar approach is used in by the authors in [98] by using infra – red sensors to build an Intelligent Security System for VANETs. The approach used here is exactly same as in [97], but the ICMetric basis number generated here is through Infrared sensor combined with traditional ID by extracting 16 significant

features. The simulation tools used here are SUMO (Simulation Urban Mobility) and MOVE (Mobility Vehicles) to understand the real – life scenarios. The generated data is extracted from the trace files. The focus remains on the infrared sensors to generate a unique ICMetric Basis number for device identification. The results indicate that the system employing the ICMetric IDS works much better than the normal IDS with false alarm rate at 3.69% for system using ICMetrics and 12.24% using normal IDS.

Each Wireless Sensor Network (WSN) consists of several nodes depending on the application they are employed to run. These can either be critical applications like health monitoring or everyday applications. Each node can act as an entry point such as to perform a large-scale attack. Paper [99] proposes the usage of ICMetrics to build an authentication technique without storing the keys. It uses the system properties to generate a unique basis number based on the characteristics of each node. Hence, any hardware and software-based changes encountered in the node is reflected in building a basis number. This enhances the security of the complete network as when the attacker tries to launch a complete network attack or a node capture attack or eavesdrop to gain access to the cryptographic key;. each of these attacks changes the working of the system/ uses more resources of the system or changes the hardware configuration. This altered configuration will be detected on the node and the node will be excluded to gain access to the network.

ICMetrics is highly utilised in sensor – based off the shelf embedded systems, authors in paper [100] use FPGA as a sensor based device HMM to generate synthetic PDFs from raw sensor data to increase stability. The study is used to generate an n – bit ID from the data extracted from the on – chip hardware macro blocks or ICs. The key issue addressed in the study is the data extracted is not stable or consistent in nature. The authors here make use of opto – couplers which generate data from light induced variations and convert it to clean signal with a purpose of generating stable synthetic PDFs from raw sensor data. Initially the HMM is trained on a raw sensor dataset and store its parameters, which is then used on a new set of sensor data to generate a state prediction and build synthetic PDF. These synthetic PDFs are utilised with the ICMetric approach to generate stable IDs.

The ICMetric technique utilises the data from the system and generate a behavioural model which leads to accurately identify the data, given the appropriate number of samples were used to model the behaviour. The unique basis number is generated to identify the devices, but the basis number cannot be directly used as a cryptographic key as they have smaller size and are less random in nature. The cryptographic keys are generated by combining the features. The main purpose of using the ICMetric technique is to add a layer of security on an existing IoT based communication system, which should work effortlessly even with the low resource devices. Hence boosting the existing IoT system capabilities with limited effects upon it. ICMetrics is an alternative to the PUF – based authentication technique. The main advantage of the ICMetrics technique is that it does not store the keys generated by the devices used for authentication and does not require additional hardware to support the technique. This prevents various attacks where stored keys can be intercepted by invasive or non – invasive attacks. The ICMetrics can be prone to attacks based on analysis if they are generated using simple features [101]. Section 2.9 describes the generalised version of steps or phases involved in ICMetrics technique as used by the authors in the literature.

2.9 ICMETRIC GENERATION

This section describes the generalised procedure to generate ICMetrics used by the authors in the literature to generate keys. The Integrated Circuit Metric generation is predominantly dependent on the features recorded from the device while in different modes when deployed i.e., sleep mode, active mode, data transfer mode etc. From the description based on the previous section, we know that ICMetrics uses the combination of hardware and software characteristics to generate keys for authorization. For effective key generation, the data is recorded as a background process while the device is working in its environment. Users collect the hardware and software data characteristics from the devices while they are deployed. These device characteristics are also known as features of the device, hence used in generating the signature of the device. To generate an ICMetric of a device, the first step is to extract the features from the device and perform premature statistical analysis of the data. Generating ICMetric is a two – part process. The first part is called calibration phase and second part is the operational phase.

- Calibration Phase:
    1. The calibration phase is applied once per device.
    2. Appropriate features are selected from the data obtained for analysis in frequency domain.
    3. Frequency distributions are generated for each individual feature, histograms are generated and statistically analysed.
    4. The feature distributions for each feature are normalised to generate normalisation maps.
- Operational Phase:
    1. The operational phase is applied each time a key is generated.
    2. Generate normalisation maps to get data for key generation.
    3. Apply Key generation techniques.

The next chapter explores the novel application of ICMetrics technology as the part of this thesis for Device Identification in detail for low resource embedded devices like Raspberry Pi using dynamic features. Typically, the ICMetrics technique is used on off the shelf devices with complicated hardware and various sensors. This technique lays a foundation for various cryptographic services and provides security without storing any encryption/authentication information on a device, hence overcoming the data leakage and key theft type attacks. It is also known as a study of device characteristics to generate a unique ICMetric. The next chapter presents the novel technique which utilizes the basis of ICMetrics technique, uses the dynamic characteristics / features of the device and addresses the multimodal behaviour for better classification.

3 ADAPTING ICMETRICS FOR LOW – LEVEL HARDWARE DEVICES

In the past decade, the work on ICMetrics technique on COTS (Commercial Off The Shelf) devices has become popular in terms of securing these devices. Identifying devices online or device fingerprinting is an upcoming field with applications in various industries such as fraud detection, Forensics, authentication, prevent hacking etc. The embedded devices like health or sensor – based monitors etc, even though encrypted, can easily be recognised when connected to a network which are usually used for identification are easily intercepted by foes as the identifying features lack complexity. The ICMetric technology is used as device identification technique which is also used for authentication, encryption, and key generation. This novel technique was previously applied on off the shelf complex connected devices for various cryptographic services. The technique depends on system features for suitable ICMetric generation.

The features are to be examined appropriately as they play a very vital role in identifying devices as misinterpreting them could cause false identification for example, an imposter could be misidentified as a device on network and get critical information. The other simpler embedded devices like Raspberry Pi are extremely versatile in nature and can host about any IoT application. The credit of growing connected devices since the past decade can partially be given to the versatility of the Pi being able to integrate with various sensors and host numerous services. These Pis have a very basic security layer; hence these are easily hackable or can easily be accessed when connected to a network.

Recently, NASA's Lab was subjected to an attack where data was leaked. The reports found a **Raspberry Pi** was used as a vulnerable node to gain access to the systems[102]. The adversaries took advantage of the feeble security infrastructure of the device to gain access of the system and due to lower internal security, the access to other data sources on network easily available. The attack happened in April 2018, where 500 MB of sensitive data was stolen and according to [103], the report mentions the threat was undetected for 10 months. Therefore, one of the objectives of the work performed in the thesis is to investigate the Raspberry Pis to add a layer of security to these low – level embedded devices. The Raspberry

Pi devices are being used extensively to develop various applications at a larger scale to build a network and as previously mentioned in the literature, an attack on Raspberry Pis can cause extensive damage. Hence, this chapter in the thesis examines the Raspberry Pi devices based on its memory usage. So, the next part in this chapter investigates Raspberry Pi hardware, collects the memory usage data also known as features, examines them and processes them to be able to use for generating device identity.

3.1 RASPBERRY PI HARDWARE ANALYSIS (FEATURE BASED)

This section provides the complete procedure in detail used to build identity of the devices. From downloading the features from the device to subjecting them for classification, each step is explained in detail i.e., how the features are pre – processed/ used to identify these devices. A Raspberry Pi is a cheap credit - card sized computer that runs on a linux operating system. The Pi can be used as a stand – alone device for many physical computing applications. The easy accessibility, usability, and compatibility of Pi with sensors, other peripherals etc. overlooks the security issues related to the device because of the pros of using this device. The security issues feel smaller initially when weighted against the advantages of device but has greater repercussions down the lane protecting the data the device generates. Since Raspberry Pi can host many applications, it can have multiple devices on a network monitoring various things or monitoring same things at different locations. Since the Pi is a limited resource device, the features used for ICMetric generation are chosen and analysed carefully. The criteria of selecting features are mentioned below in section 3.4 named 'Feature Selection'. The identification of each device should be generated on a molecular level; therefore, the system memory usage data of each device are analysed.

To generate the device identification, the system memory features are used. All the system memory features were chosen for initial investigation as it gives the information on memory used, free memory and total memory. This information provides the data used by different systems of the Raspberry Pi when programs are running on it. The system memory information is downloaded during the functioning of the device so any change in the working of the device is reflected at the system level i.e., in the system memory features. It is understandable that as these features reflect the amount of processing memory the program on the device uses the features can be highly dynamic in nature.

The memory features were chosen to investigate for ICMetrics instead of sensor readings because sensor readings can be replicated if used in similar environment. So, investigating memory readings made much sense as they are affected by all the processes and are hard to replicate as each programmer programs the same application differently. This means the program size, variables used, processing time, other background processes needed to run the fully functional application, etc would be unique but needs further investigation.

The Table 2 below shows all the memory features collected for initial examination and their description:

Sr. No.	Feature Name	Feature Description
1.	MemTotal	Total usable device RAM except the reserved memory bytes
2.	MemFree	The measure of unused physical RAM
3.	MemAvailable	Total memory available for initiating new programs without swapping
4.	Buffers	Temporary storage for raw disk blocks
5.	Cached	Physical Memory being used as cached memory
6.	SwapCached	The measure of memory when moves to swap memory and back to main memory
7.	Active	Memory that is actively being used and not reclaimed
8.	Inactive	The memory that was used less lately and can be reclaimed
9.	Active(anon)	Measure of both anonymous and shared memory in use or was used recently before moving to swap
10.	Inactive(anon)	The measure of both anonymous and tmpfs/shmem memory in line for expulsion
11.	Active(file)	The measure of file cached memory that is in or was recently used when system reclaimed memory
12.	Inactive(file)	The measure of file cached memory recently stacked from the disk or is in line for reclaim
13.	Unevictable	Measure of un-evictable memory as its locked by applications run by user
14.	Mlocked	The memory locked by user run applications
15.	SwapTotal	The memory available to swap
16.	SwapFree	Amount of swap free measured
17.	Dirty	The amount memory in line to be returned to the disk
18.	Writeback	Memory currently being written back to the disk
19.	AnonPages	Memory used by pages are mapped onto userspace page tables & have no backup in files
20.	Mapped	Memory used for libraries (or any files) and are mmaped
21.	Shmem	Memory used as shared memory (shmem) is measured
22.	KReclaimable	Memory allocated to Kernel, includes the SReclaimable value.
23.	Slab	This is a measurement of cached data from the kernel
24.	SReclaimable	A portion of Slab memory that is reclaimable
25.	SUnreclaim	A portion of Slab memory unreclaimable even if there is memory shortage
26.	KernelStack	The total allocated memory for each task of the system by kernel stack allocations
27.	PageTables	Memory assigned lowest page table level
28.	NFS_Unstable	The NFS pages memory delivered to server but is not committed to the stable storage
29.	Bounce	Bounce buffers is the memory usage for block device
30.	WritebackTmp	Memory used for Temporary writeback buffers by FUSE
31.	CommitLimit	The total memory available to be assigned to the system
32.	Committed_AS	Estimation of memory to complete the task assigned which also includes the swap memory and worst-case scenario value
33.	VmallocTotal	Memory assigned to virtual address space
34.	VmallocUsed	Memory used by virtual address space
35.	VmallocChunk	Largest memory chunk assigned, are adjacent to each other in a block format that is free
36.	Percpu	Percpu assigns the memory to back percpu allocations. This excludes the metadata cost.
37.	CmaTotal	The memory that has been assigned to Contiguous Memory Allocation
38.	CmaFree	The amount of free memory for Contiguous Memory Allocation

Table 2 Feature Description for all collected features

Memory Features of the Pi, even though unstable are selected as they reflect even the slightest change when the system is running. The data is collected by continuously running the linux command **"/proc/meminfo"** which yields the 38 features listed above. The next section describes the technique from start to finish, which includes the data collection, data analysis, data selection, data processing and classification technique.

The device used here is Raspberry Pi 2 B where the above-mentioned features are collected. The intention is to confirm if the Pi's can be identified or differentiated by the working of the device. To prove if multiple devices, even though running a same program/application can be distinguished by the way they work when they are deployed on the same network. The Figure 10 below shows the Raspberry Pi 2 B that were used as a part of data collection.

Figure 10 Raspberry Pi 2 Model B

Here six identical Raspberry Pi's running the same application are used for data collection. For proper analysis, each device should have sufficient samples for it working in various modes when subjected to data collection i.e., for example when the code is running (sensing phase), during data transmission, idle, processing the data etc. Hence, for data collection, each device is subjected to work in various modes and samples were extracted as a background process which are then analysed. The number of samples used for analysis is crucial as it captures the

true nature of the data and their dependencies. Here we do not keep the set number of samples but keep minimum number of samples as 1500. Number of samples kept is more as features are volatile in nature. After collecting the data, we eliminate the top 100 and bottom 100 to use the stable data. Each feature is then analysed, and its frequency distribution is generated to initially understand its nature in general. The next section 'Feature Extraction' shows how the characteristics of the data is extracted from the device.

3.2 Feature Extraction

Features are the building blocks of any ICMetric system that provides the underlying information on the functioning of the device and contributes to generating an identity of the device. The number of features collected are examined to choose and use the valuable features. The 1500 samples are selected to observe the variations in the features caused during the working of the device. Each feature is then analysed, and the frequency distribution per feature is observed. The RPi devices are used in various applications. The data collection is performed on the Pi's which are pre – engaged in running the applications they were designed for. The aim is to not disturb the normal working of the system to collect the data.

The code running on Raspberry Pi 2 B (with 16GB SanDisk Memory card), runs as a background process, meaning we do not intervene with the normal working of the device. Hence separate data collection code is built which is set up in a way that it is triggered when the device is booted. The device runs Raspbian Desktop version and saves the data locally on the SD card. The features are extracted using Node – Red [104]. Node – Red is used so it can be configured to start on the device start – up and can run the data collection in the background without intervening with the working of sensors, its processing and transmission. This gives assurance of capturing the changes in the data. The key reason we use the memory – based operational features is because each time the application running on Pi uses the memory to record, process or transmit the data; changes are appropriately reflected. The Figure 11 below shows the snapshot of the Node – Red code used for data collection.

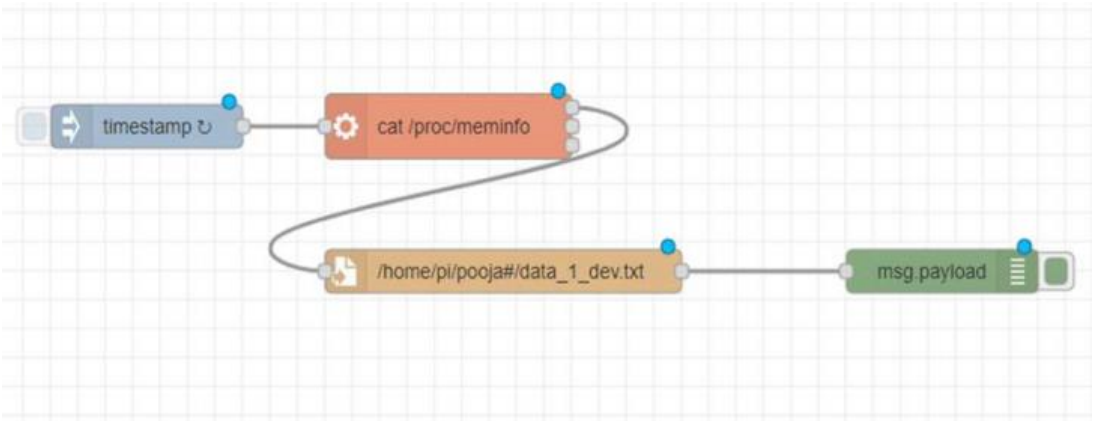


Figure 11 Node - Red flow for collecting data

The Figure 11 above shows the code used to collect the information from the device continuously when its ON and stores it in a .txt file. A set of 38 features are recorded, each time the program loops over. There is no delay introduced in the code to assure no critical changes in the data are missed because of the delays. All the features are collected initially to further analyse the data to gain better understanding of the correlation factor amongst features and its variability in context to each other. The data collected is then extracted to an excel file and the data is pre-processed to perform the analysis.

The next section describes the pre – processes performed for feature selection and analysis of the data selected which is then subjected to classification. Each collected feature is a subject for initial descriptive analysis to see the difference in the amongst the same feature from different devices.

3.3 Feature Analysis

The features mentioned in table are subjected to analysis once transferred to an excel file. This section describes the pre – processes applied on the features, analysing the feature distribution and challenges to model them. The most important part of performing feature analysis is examining the features in multidimensional space to understand and exploit their interaction with each other. Each feature mode is examined in context to the other modes in the feature set to work out the relationship amongst the modes. This gives an insight on various ways the features are correlated to each other.

3.3.1 Pre – Processing:

The pre – processing of the data is performed on all the features to identify the usable features. The initial examination includes performing the descriptive statistical analysis i.e., finding out the Mean, Median, Standard deviation, Mode, correlation, covariance of all the features of the device. Further, we eliminate the static features, as these remain same throughout the data collection process and across various devices. After eliminating the static features, the next step is to perform the exploratory data analysis i.e., generating PDF graph for feature distribution and examining features in multidimensional space (after feature selection) on remaining features using python in jupyter notebook. This is documented in the next section as data behaviour and then the select appropriate features.

3.3.2 Data Behaviour:

After the static features are eliminated, remaining features are examined by themselves first to understand the viable features to be selected as a basis for better classification. Initial step requires each feature's probability distribution to be examined. The importance of this step is to understand the variance/ variation each feature exhibits during the working of a device. All remaining features are analysed individually and each of their distribution is noted, and the nature of the feature is deduced. Each feature is to be observed individually as we are using memory features for device identification, and these are highly dynamic in nature and are based on the intensity and number of processes running on any stand – alone Raspberry Pis. This leads to selecting appropriate features amongst the given features. Each of the feature distribution is visualised and characterised. The appropriate features are selected to generate a feature set to form a basis of classification. The Table 3 below lists the distribution acquired by each feature when collected from the 6 Raspberry Pi devices:

Sr. No.	Feature Name	Feature Distribution (PDF Graph)	Feature Characteristics
1.	MemFree	Unimodal/ Multimodal	Dynamic
2.	MemAvailable	Multimodal	Dynamic
3.	Buffers	Multimodal	Dynamic
4.	Cached	Bi modal/Multimodal	Dynamic
5.	SwapCached	Unimodal/Multimodal	Dynamic
6.	Active	Unimodal/Multimodal	Dynamic
7.	Inactive	Bimodal/Multimodal	Dynamic
8.	Active(anon)	Bimodal/Multimodal	Dynamic
9.	Inactive(anon)	Bimodal/Multimodal	Dynamic
10.	Active(file)	Multimodal/Bimodal	Dynamic
11.	Inactive(file)	Bimodal/Multimodal	Dynamic
12.	SwapTotal	Constant graph	Static
13.	SwapFree	Constant graph/ Multimodal	Almost Static
14.	AnonPages	Multimodal	Dynamic
15.	Mapped	Bimodal	Dynamic
16.	Shmem	Unimodal	Static
17.	KReclaimable	Bimodal/Multimodal	Dynamic
18.	Slab	Bimodal	Stable Dynamic
19.	SReclaimable	Bimodal/Multimodal	Stable Dynamic
20.	SUnreclaim	Bimodal	Stable Dynamic
21.	Committed_AS	Bimodal	Dynamic
22.	CmaFree	Unimodal/Bimodal	Stable Dynamic

Table 3 Probability Distribution for each feature

The importance of examining each feature is because we select the highly dynamic features as these are volatile and are hard to be replicated using different devices. The Table 3 above shows the probability density of the features observed in each device (not separately but summarised). For example, let's take MemFree feature, here the corresponding feature description states that unimodal/multimodal, which means in all 6 devices when MemFree feature is observed, it has either unimodal or multimodal distribution. Hence the next step is to select valuable features from these initial examinations and subject them to a further analysis i.e., feature selection criteria and examining them in multidimensional space.

3.4 FEATURE SELECTION:

The features mentioned in the table above depicts the distribution of each feature which gives an insight on how the features change during any part of code execution running on it. Each features probability distribution is described as observed in all 6 devices. These features reflect the changes that occur during the code execution on the stand-alone Raspberry Pi devices. The importance of selecting quality features is easily reflected when these are subjected to generate an identity of the device. Hence, we select the features which reflect the working of the device and are related to each other, so we can exploit that relationship to gain better classification results than the standard features provide. Since all the features are multimodal, we exploit the relationship between them by examining features in multidimensional space (this is explained in section 3.4.1 Addressing Multimodality).

Criteria for feature selection:

- Low intra – sample variance and high inter – sample variance: This is a measurement of variability observed in a feature of a device and variability observed in a feature across all the devices. This is used as a preliminary analysis to select adequate features to build a model.
- Features should be Dynamic in nature: Dynamicity depends on the unpredictable nature of the features. As they highly depend on the working of the devices. The aim is to use the features highly affected by the working of the device and are correlated to each other.
- Consistency in the feature distribution observed in all devices: A consistent distribution of a feature across all devices is observed to verify the dynamic nature of the features and correlation amongst them.

Four features were selected namely, MemAvailable, Inactive(file), Slab and SUnreclaim. Each of these features are analysed and their distribution is visualised to understand their nature. The following Figure 12, Figure 13, Figure 14 and Figure 15 depict the **probability distribution** observed in all the devices, **where x – axis is the sample value and y axis are the probability density value**:

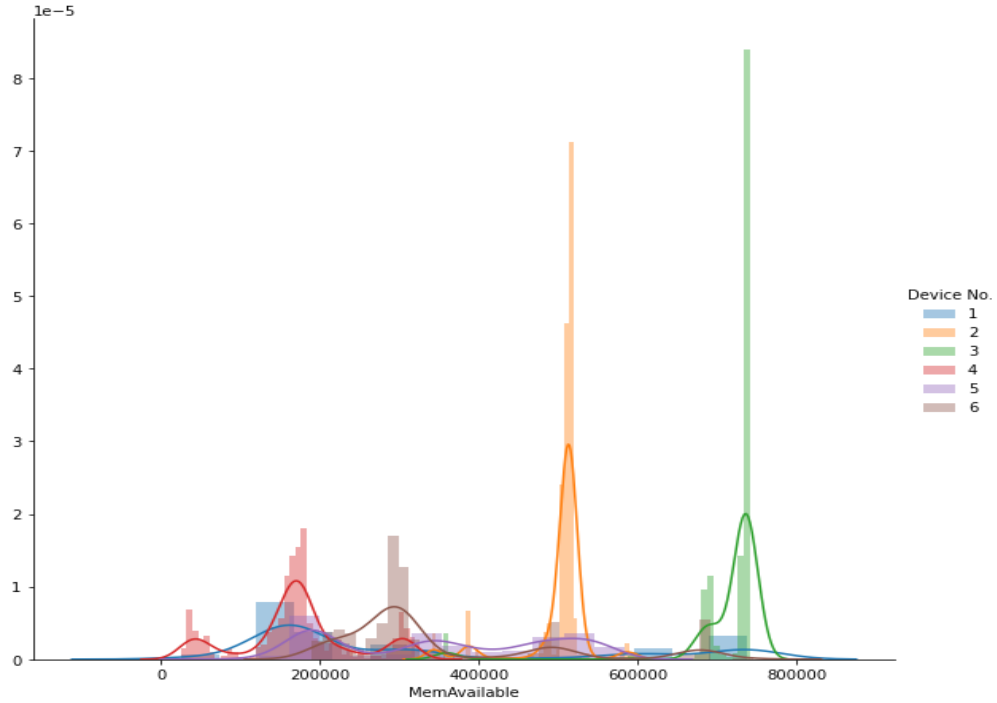


Figure 12 PDF graph for feature MemAvailable

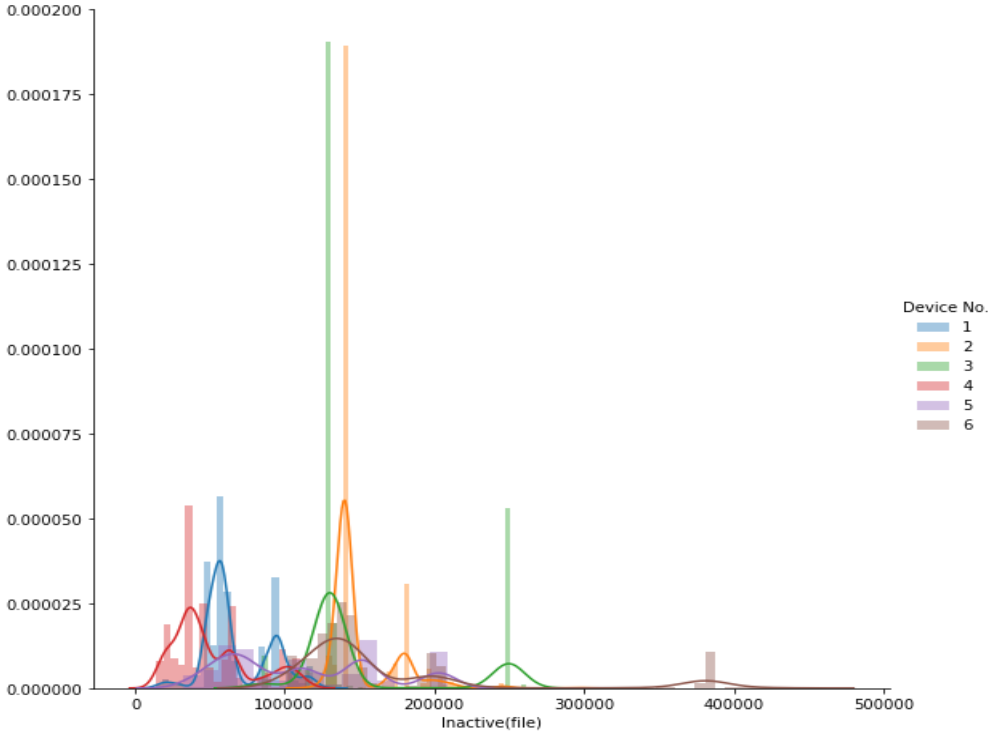


Figure 13 PDF graph for feature Inactive(file)

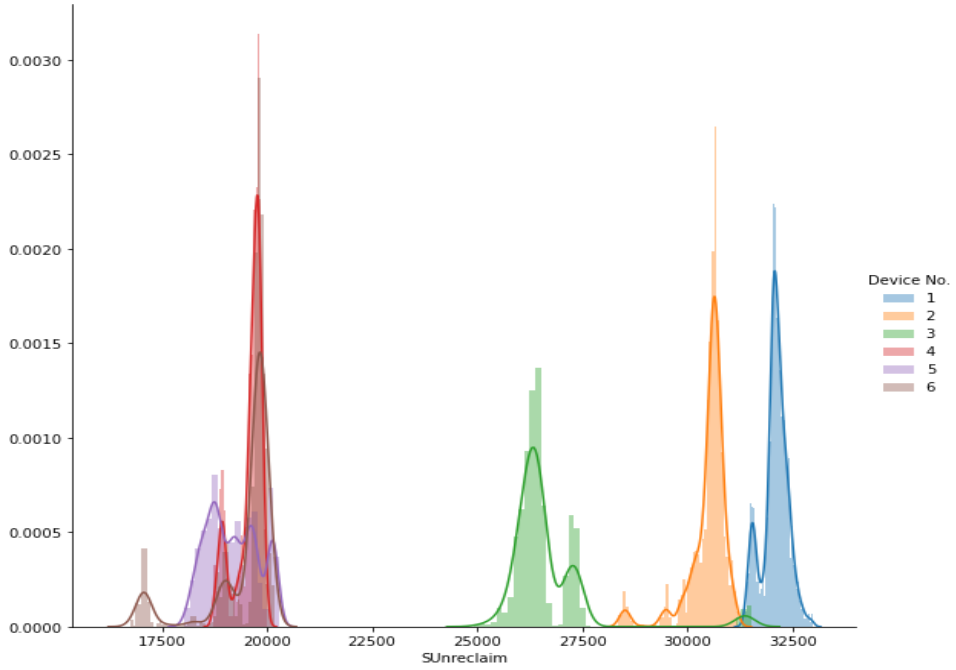


Figure 14 PDF graph for feature Sunreclaim

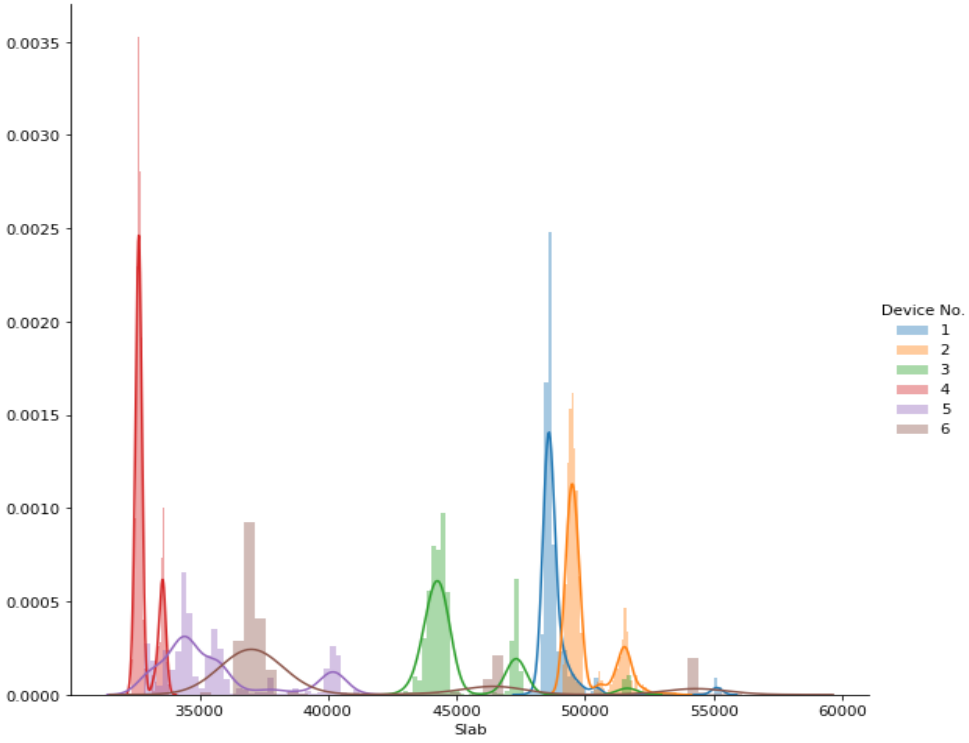


Figure 15 PDF graph for feature Slab

Figure 12, Figure 13, Figure 14 and Figure 15 show the feature distribution for each selected feature in all the devices. The graphical representations of the selected features are shown to understand the problems when using multimodal features. Each of these graphs show the probability density of each feature for all devices and there is a significant overlap amongst them. In the graphs **x – axis show the feature values and y – axis shows the probability**

density. As observed, the features selected are bimodal / multimodal and a challenge to model properly. The following observations are made in multimodal features:

- The overlapping amongst the features in different devices
- Highly dynamic nature of the features

The multimodal features are difficult to model for following reasons:

- The feature distribution cannot be accurately described using general statistical data unlike unimodal features which are easily represented using mean, covariance, variance etc.
- The overlapping amongst features, makes it difficult to generate classification boundaries.
- Multiple multimodal features (i.e., for multivariate datasets), can be misidentified due to excessive overlapping as observed in our dataset from Figure 12, Figure 13, Figure 14 and Figure 15.
- These reasons are why unimodal features are preferred over multimodal features to use for data classification.

Therefore, we look at the Bimodal/Multimodal features in multidimensional space and hence, their relationship with each other. This is described in next section with an example.

3.4.1 Addressing Multimodality:

Traditionally the features selected for classification are termed as stable and are usually unimodal in nature. A unimodal feature can be described as a feature with one peak i.e., when subjected to Probability Distribution Function (PDF), its graph shows a single peak as shown in the Figure 16 below. So, instead of considering traditional unimodal features, the dynamic features are used for device detection.

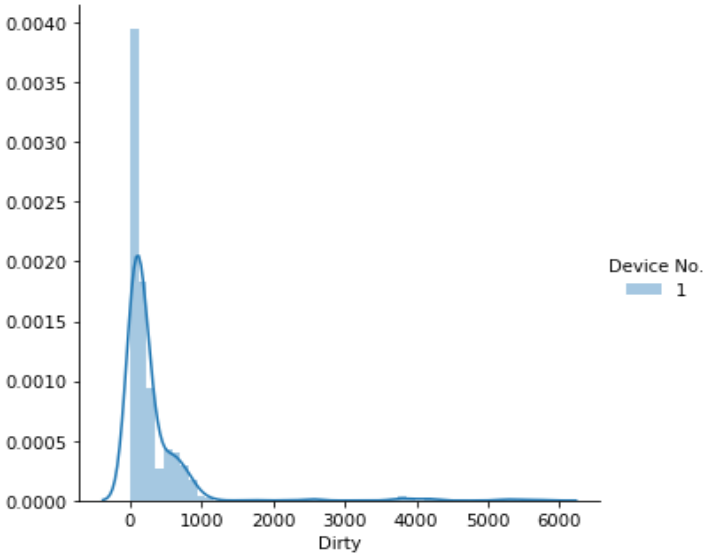


Figure 16 Feature (Dirty) Depicts Unimodal nature

Here we use multimodal features as the features that exhibit unimodal behaviour are static in nature and do not meet the criteria mentioned in feature selection section.

Figure 12, Figure 13, Figure 14 and Figure 15, show distribution of each feature from each device is presented in a different colour. Each colour represents same feature across all the devices. Viewing the same feature across all the devices gives the understanding of the extent of overlapping of the features, their nature, and ideas to model them. Since there is too much overlapping amongst features so, we exploit the relationship between the selected features. The features show similar behaviour across all the devices but do not always have same number of modes across all devices. Hence, when building a model, the key step is to identify the number of modes that are present in a feature. The steps below address the multimodal features and convert them to modal combinations with the help of thresholds generated via peak trough algorithm. By addressing the multimodal nature of features, we take care of the overlapping issue. The overlapping of data between devices per feature, is differentiated by establishing a relationship between the modes, which means by creating the combination of modes that we mentioned in step 3, we can show the relationship between the modes of features (these are unique per device). To address the multimodal nature of the features, following operations are performed:

1. The first step is to **identify** the number of modes for each feature per device. This is done via subjecting each feature per device to peak – trough algorithm[89] .

2. Secondly, these multimodal features are **divided** into the number of modes which were identified by using peak trough algorithm. Here, each feature will **generate thresholds** equal to the number of modes present in the feature. These thresholds will create modal boundaries. The peak – trough algorithm identifies the throughs in a particular feature distribution. The feature distribution is identified by using the histogram function which shows where the troughs lie in the data which in turn determines the modal boundaries i.e., thresholds.
3. Next step is to exploit the relationship between the features and their modes to build the model. **For example**, considering a dataset with two features namely F1 and F2 which are bimodal and multimodal in nature (shown in Figure 17 below).

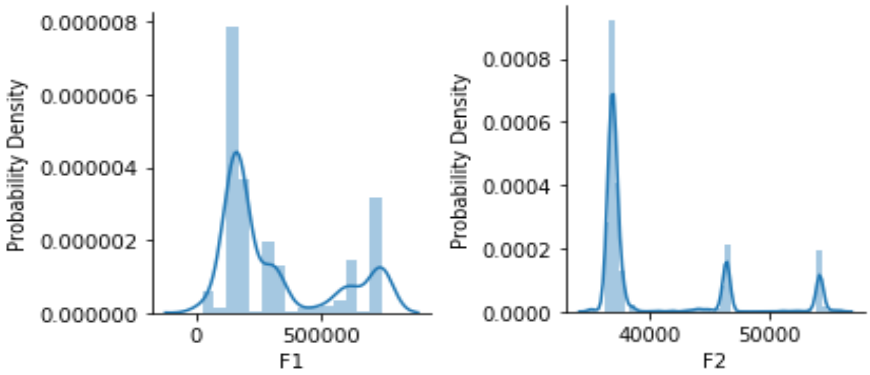


Figure 17 PDF graphs for F1 and F2 showing modes individually.

4. As observed from Figure 17, F1 has two modes, let us denote these as F1 – M1 and F1 – M2 and Feature2 (F2) has three modes (M1, M2, M3), and similarly are denoted as F2 – M1, F2 – M2, F2 – M3. Establish a relationship between the modes of these features by generating modal combinations. For this, example dataset with two features (as mentioned in 3) is taken in account to generate and exemplar model for generating modal combinations. Using these two features, 6 modal combinations are generated from this example dataset and are as follows:
    a. [F1-M1, F2-M1]
    b. [F1-M1, F2-M2]
    c. [F1-M1, F2-M3]
    d. [F1-M2, F2-M1]
    e. [F1-M2, F2-M2]

f. [F1-M2, F2-M3]

5. Each of these combinations contains data which is unimodal in nature and represent only device 1. This process is carried out per device with n features and the number of combinations (6 in the above example) is dependent upon the number of modes in each feature. These modal combinations are also referred to as 'converted gaussians'.
6. Sort the samples with respect to thresholds per modal combination of features for that device. There may be some empty combinations (i.e., no samples in that combination), which can be eliminated as soon as identified.
7. These generated modal combinations for all datasets generate multiple levels of identity for each device.
8. The model is then trained using each of these modal combinations. So, when the data is sampled for testing, each test sample vector is first checked that which modal combination the sample belongs to and is tested against it. There is a possibility where the test sample appear to belong to more than one modal combinations.
9. The sample vector is then tested against those distributions to generate the probability and assigned to the combination which generates maximum probability against it.

For example, if the values of F1 totally overlap for device 1 and device 2, then we factor in the F2 for respective devices. This takes in account the relationship between F1 and F2 for each device, hence showing if the overlapped values in F1 might not be related to the same values of F2 for ever device. Using this relationship of each feature with the other per device, we exploit the distinctive working pattern of each device on the lower hardware level and properly modelling the dynamic nature of the features selected.

3.5 Experimental Methodology using Proposed Classification method (MMVGD)

This section describes the novel classification algorithm from start to finish for testing the data against the training samples. The Algorithm is divided into two stages:

Pre-processing:

1. Data collection on Raspberry Pi via Node-Red i.e., Feature extraction.

2. In the next step, we select four features to generate a feature set.
3. Divide the data into training and testing sets with 80:20 ratio.
4. Generate a probability distribution for each feature.
    a. Check if the distribution is unimodal, bimodal, or multimodal.
5. If the distribution has more than one mode, apply peak – trough algorithm [89] to find the modal boundaries which are referred as mode thresholds.
6. Generate modal combinations to divide the data per device into multiple 'converted gaussians'. These are converted with the help of modal thresholds generated by using peak trough.
7. Each of these converted gaussians can act as multiple multivariate distributions per device, like the way it is represented in Gaussian Mixture Models[105] using clusters of data.
8. Each of these multivariate gaussians are represented by its mean (μ) and covariance (∑) matrix.

Classification:

9. Compute Probability of each test sample against modal combinations the test feature vector it is suspected to belong to per device using 'multivariate_normal.logpdf'
10. Store the probability generated per device and assign the device with maximum probability output.
11. Repeat for all devices and find max probability generated which is our prediction data for classifier.
12. Generate a confusion matrix which shows the number of correctly predicted samples and show the number of misclassified samples.
13. These results are then benchmarked against state-of-the-art classification techniques.

The next section discusses the results obtained when subjected to proposed classification technique.

3.6 EXPERIMENT RESULTS

The classification is carried out using the raw features collected from raspberry pi working in two modes for device identification. The selected features are subjected to pre-processing steps. It is crucial to take in account the multimodal nature of the features to get better results. The experiment is performed using the features collected from 6 Raspberry Pi's. The data is collected in two different Pi usage scenarios:

Scenario 1: When the device is actively using its sensors and transmitting the data (not all the time) i.e., the devices are working in all the modes it can like transmitting, idle and sensing.

Scenario 2: When the device is actively deployed but only running the background processes.

These two modes are chosen for following reasons:

- It is unlikely that any device is continuously transmitting information without breaks. Even the applications that use the Raspberry Pis as designated gateways, have designated intervals for transmitting data.
- When the device is continuously using it sensors, it also has a program on it which take processing time, memory, and power these factors affect the device memory features.
- There are times when devices are idle in a network, so we use the second scenario as a device only running background processes.
- Lastly, when there is a network of devices, there is a possibility of a few devices being toggled from one mode to another whereas the other devices remain idle.

Hence, these two scenarios are used together to observe how the data behave when subjected to classification, get an insight on how the samples are misclassified while using multimodal data and if devices using different mechanisms can be differentiated or not.

The Table 4 below shows the device number and the scenario used to collect its data.

Device Number	Data Collection conditions
1	Scenario 1
2	Scenario 1
3	Scenario 1
4	Scenario 2
5	Scenario 2
6	Scenario 2

Table 4 Data collection scenario for each device

The classification metrics contains four parameters:

- Accuracy: Accuracy is defined as the percentage of correct predictions made by the classifier with respect to all the test data.
- Precision: The precision is the measure of correctly classified data to their respective classes i.e., it is the measure of true positives for each class.
- Recall: It is the measure of the number of True positive samples were recalled. To capture all the cases of a class correctly.
- F1 score: The F1 – score represents both the Precision and Recall. It can be said as the harmonic mean of the two entities.

Usually, the measure of accuracy is used to define the validity of the model. But since the test data we have has unequal proportions of the classes in question, accuracy percentage alone cannot be used as a measure to best describe the model. Hence, we can either use recall or precision as a measure to describe the results of the model correctly.

We compare the proposed MMVGD classifier to the standard sklearn classifiers subjecting them to the raw features, without taking in account the multimodal nature of the features. This is done to understand the importance of taking in account the feature distribution of the selected features. Firstly, we look at the confusion matrix generated by each standard classification method to see which of the device's data are misclassified the most and compare them with proposed classification method introduced. When the data is misclassified as the other device, a sense of it can be made looking at the features together from both devices i.e., the actual device the test set is from and the predicted device to where it belongs. The figures below show confusion matrices for each standard classifier and proposed classification methods.

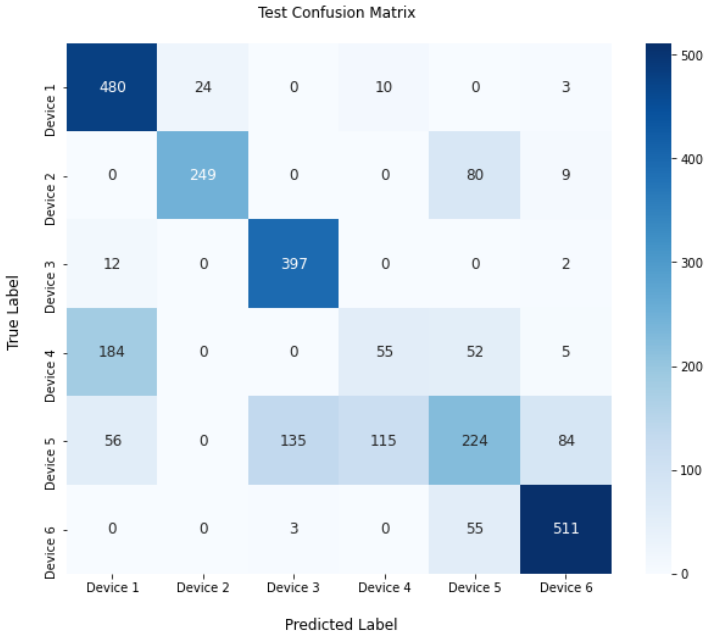


Figure 18 Confusion Matrix for LR

The Figure 18 above represents confusion matrix generated when data is subjected to LR classification technique.

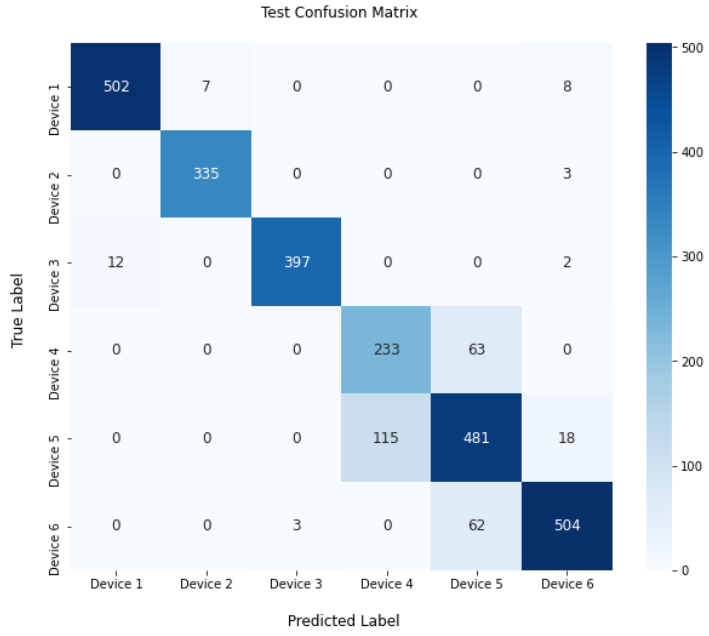


Figure 19 Confusion Matrix for SVM

The Figure 19 above represents confusion matrix generated when data is subjected to SVM classification technique.

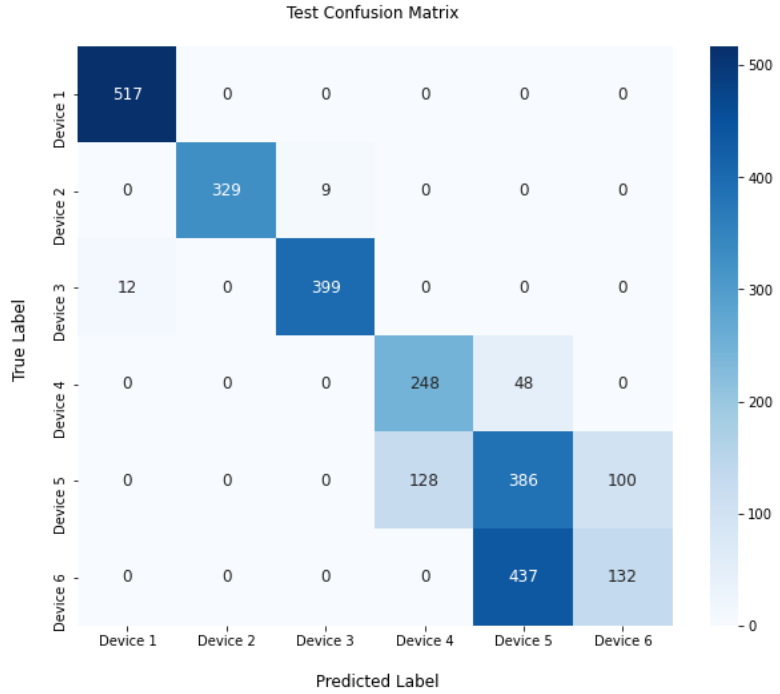


Figure 20 Confusion Matrix for GNB

The Figure 20 above represents confusion matrix generated when data is subjected to GNB classification technique.

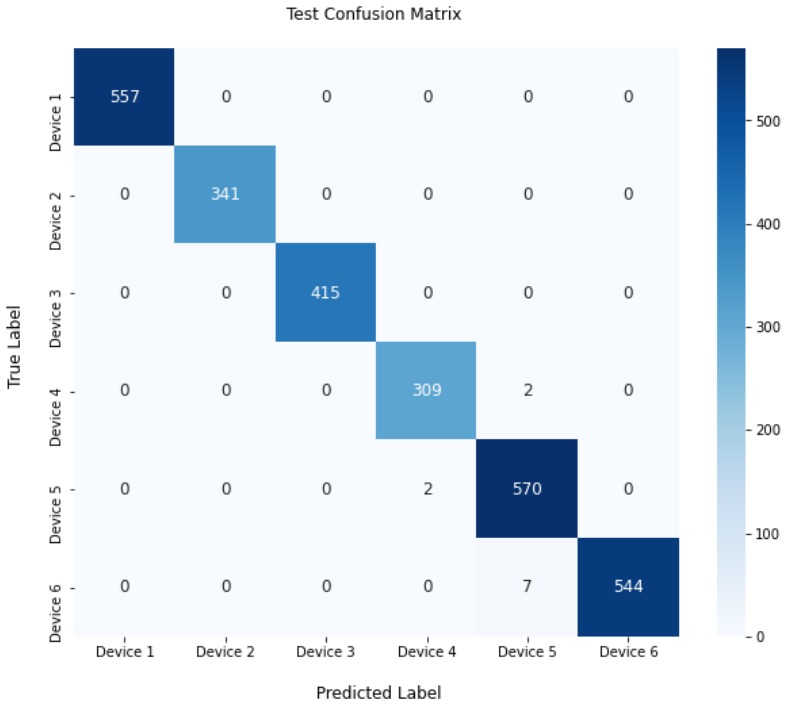


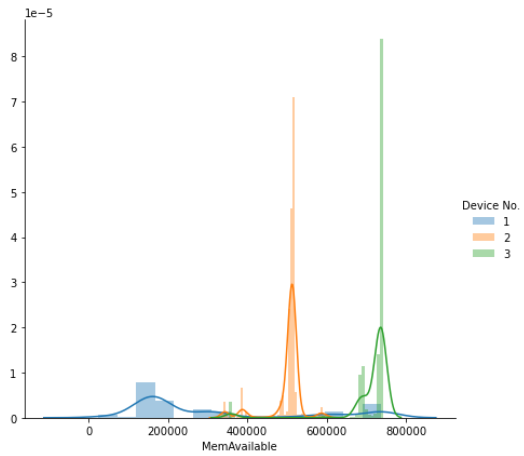
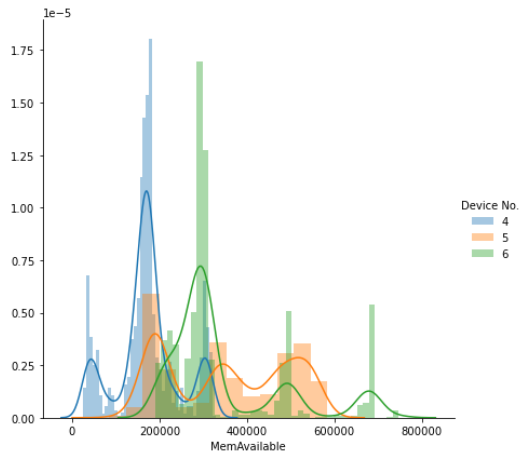
Figure 21 Confusion Matrix for MMVGD (Proposed classification Technique)

The Figure 21 above represents confusion matrix generated when data is subjected to MMVGD classification technique.

As observed from the confusion matrices above, we can say that novel classification technique (taking in account the multimodal nature of the features) developed, yields the best results as Figure 21, the confusion matrix for MMVGD, has least number of misclassified samples whereas, from Figure 18, it is obvious that LR classification technique gives out most misclassified samples.

The LR classifier works poorly when the features exhibit nonlinearity and are highly correlated. Since the features also exhibit massive overlapping amongst devices, this classifier tends to yield poor results. The GNB classifier provides better results than LR, Figure 20, we can see that the incorrectly recognised samples are mostly amongst devices 4, 5 and 6. Similarly the SVM classification gives better results than GNB, but mostly misclassified samples occur amongst devices 4, 5 and 6 and a few misclassified samples between classes 1, 2 and 3.

Drawing from the confusion matrix to account for misclassification we look at each feature from the two groups simultaneously and understand why classification is better amongst devices of group 1 (consists of devices 1 – 2 – 3) than group 2(consists of devices 4 – 5 – 6).

Feature Number	Group 1 (Devices 1 – 2 – 3)	Group 2 (Devices 4 – 5 – 6)
1.	 *Figure 22 Feature distribution for MemAvailable in device 1, 2 & 3*	 *Figure 23 Feature distribution for MemAvailable in device 4, 5 & 6*

2.

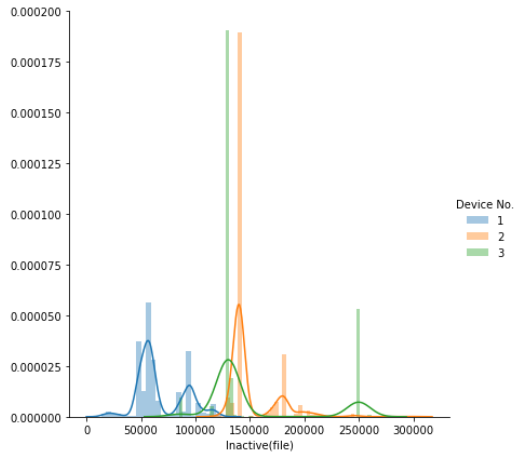


Figure 24 Feature distribution for Inactive(file) in device 1, 2 & 3

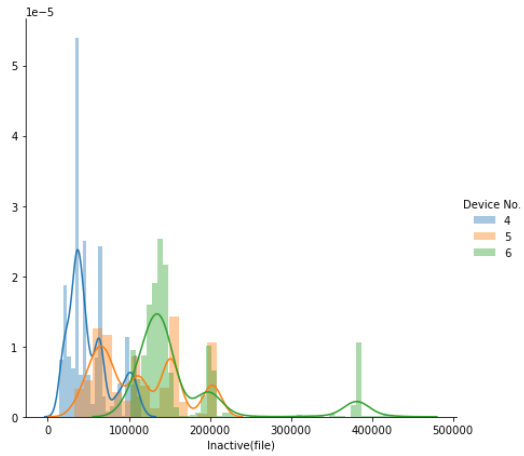


Figure 25 Feature distribution for Inactive(file) in device 4, 5 & 6

3.

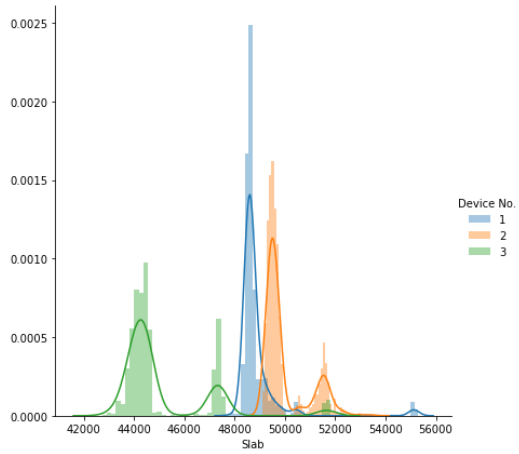


Figure 26 Feature distribution for Slab in device 1, 2 & 3

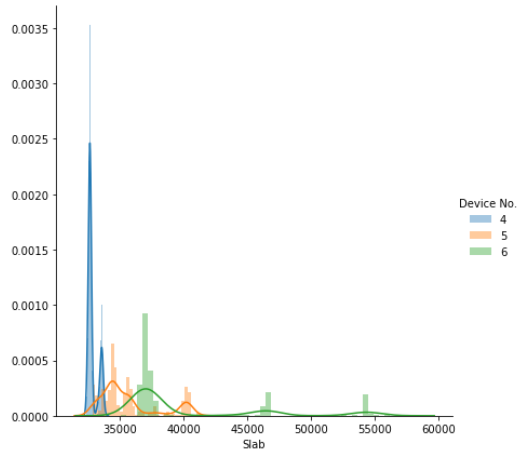


Figure 27 Feature distribution for Slab in device 4, 5 & 6

4.

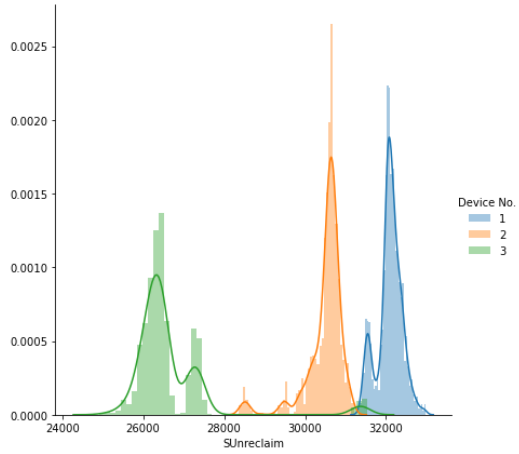


Figure 28 Feature distribution for SUnreclaim in device 1, 2 & 3

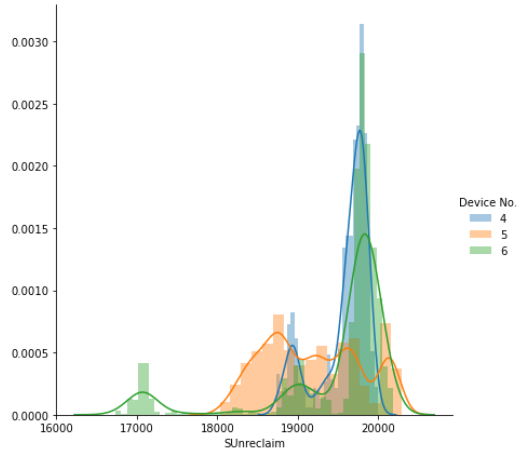


Figure 29 Feature distribution for SUnreclaim in device 4, 5 & 6

Figure 30 Comparison between each feature distribution between two device groups using different scenarios for data collection.

From Figure 30**Error! Reference source not found.**, the feature distribution for each feature, we observe the extensive amount of overlapping amongst features of each device. The maximum misclassified samples are from group 2. By comparing these two groups of data side by side we can see the difference in the amount of overlapping. Hence, justifies the poor results from the standard classifiers. Hence, we develop a technique to utilise multimodal features to convert them into modal combinations (explained in upper sections) i.e., exploiting the relationship between each mode of features and subject them to classification. Hence proposed classification (MMVGD) works best when taken these things into account. This method gives us minimum number of misclassified samples and gives the total accuracy of 99.6% with precision of 100% for devices 1, 2, 3 and precision of 99%, 98% and 100% for devices 4, 5, 6 respectively. Even though we encounter some misidentified samples, 99.6% of samples are correctly identified. The validity of the results is proved by using the standard classification techniques for benchmarking the results.

The proposed classification technique uses multimodal features, even though unconventional but yield better results than other linear classification techniques. These are presented below in a tabular format to compare the results.

The Table 5 below compares the precision measure of each classifier subjected to the data per device to understand the model and correctly analyse the results.

Classifier		Gaussian Naive Bayes (sklearn.naive_bayes import GaussianNB)			(rbf kernel) Support Vector Machine (From sklearn.svm import SVC)			Logistic Regression (from sklearn.linear_model import LogisticRegression)			MMVGD (Multimodal Multivariate Gaussian Distribution) **Proposed Algorithm**		
		Precision (P), Recall(R), F1-Score(F)											
		P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
Device No.	1	98%	100%	99%	98%	97%	97%	66%	93%	77%	100%	100%	100%
	2	100%	97%	99%	98%	99%	99%	91%	74%	82%	100%	100%	100%
	3	98%	97%	97%	99%	97%	98%	74%	97%	84%	100%	100%	100%
	4	66%	84%	74%	67%	79%	72%	31%	19%	23%	99%	99%	99%
	5	44%	63%	52%	79%	78%	79%	55%	36%	44%	98%	100%	99%
	6	57%	23%	33%	94%	89%	91%	83%	90%	86%	100%	99%	99%
Overall Accuracy (%)		**73.6%**			**89.3%**			**70%**			**99.6%**		

Table 5 Overall classification Metrics for all classifiers

Hence, the Proposed classification works best with the dynamic features and their unpredictable nature. Even though the classification technique works effectively, it is important to understand the limitations. We only have 6 devices which were used to derive data. Therefore, not enough datasets to make a definitive conclusion. This is one of the reasons why synthetic data is highly needed.

Another reason to need the synthetic data is to prove if the system built accurately identifies the number of modes, modal boundaries and properly identifies the descriptive statistics of the device. So, a validation of the system is performed by generating synthetic dataset (Chapter 6) and compare the derived data from the system with the original parameters used to generate synthetic dataset.

3.7 SUMMARY

This chapter presents the details of the entire classification process i.e., from data collection to data identification. The work presented in this chapter has a two – part novelty:

1. The first novelty is using low – level hardware (Raspberry Pi) to collect the data and using Dynamic operational features.

2. The second novelty is building a classification method to address multimodal features called MMVGD.

The dataset collected exhibits each device's behaviour during its usage. All the features from the dataset were analysed and the static ones were eliminated. Then the dynamic features were further analysed to check the distribution of the features and select the appropriate ones. The features exhibit multimodal behaviour. So, a need to address this for better identification of data. Generally, while using multimodal features, the techniques used to build a model usually use a two dimensional or three dimensional boundaries to separate the overlapping data observed from multiple features. The proposed technique is comprised of observing the data in multidimensional space but making linear boundaries by using modal thresholds and exploiting modal relationships. Hence, we compare MMVGD (proposed method) to the standard classifiers as in theory they work similarly.

The technique used to address multimodal features looks at each feature individually to get their distributions and then observe how each feature's mode is related to another by generating modal combinations. The modal combinations utilise the relationship amongst the features to build multiple identities per device.

This technique was developed to utilize dynamic multimodal features which are mostly eliminated due to their unpredictable nature and overlapping of the data. Each modal combination's distribution parameters are then used to generate probabilities when compared with a test sample. The technique gives us 99.6% accuracy which is highest when compared to other standard classification methods.

3.8 CONCLUSION

The work in this chapter presents a novel classification technique to address dynamic features that are multimodal in nature (which are normally rejected) that reflect the internal working of the Raspberry Pi devices. The data is observed in multidimensional space to exploit the relationship amongst modes observed in features by building modal combinations (explained in the chapter above). These modal combinations are built using peak – trough algorithm and

using troughs to build this combination by dividing the datasets via thresholds into multiple identities (by looking at distribution parameter for each modal combination) for one device.

By addressing the multimodality in each feature simultaneously, it is observed that the proposed MMVGD classification technique achieves 99.6% accuracy results and to validate this result, we use standard classifiers and subject them to data collected to benchmark these results. The SVM classifier shows comparable result with 89.3% accuracy.

Even with the higher accuracy percentage using MMVGD, it is understandable that the number of devices the data was extracted from are limited. Hence, the conclusions inferred from the results or datasets cannot be stated undoubtedly. There is also a question to how the proposed classification technique handles large number of devices and if it perceives the modal boundaries of each feature correctly. These questions the authenticity of the classification model built and to validate the parameters derived from it.

Therefore, there is a need to validate the proposed validation method and answer the questions described above. So, a synthetic data generation algorithm (chapter 5) to mimic the multimodal distributions observed in the features of real Raspberry Pi devices. This is done to validate the perception of data by the classification algorithm (MMVGD) to identify the devices and how the classification algorithm handles large number of devices.

4 ALTERNATIVE DEVICE RECOGNITION USING WAVELET-BASED FEATURES

This chapter presents an alternative technique on identifying devices using wavelets. The previous technique has some limitations. The previous techniques look at raw multimodal features and use the PDF (Probability Density Functions) to understand the dynamic behaviour. The technique defined in chapter 3, eliminates the modal combinations with small number of samples which may lead to eliminating the samples reaching peaks or samples found during state transition i.e., states or events that occur less frequently which might be sometimes critical to differentiating between/ identifying devices or it can be noise between device's state transition. Therefore, using wavelets also reduces the training time comparatively as it reduces the dataset, which reduces the processing time. From the literature on wavelets, it is evident that wavelets have been extensively used for identification basis. Hence, we aim to use wavelets with Raspberry Pi data in the field for identification.

An integral part of building a smart system using machine learning techniques is to automatically recognise the patterns exhibited by anything. It can also be viewed as a method to segmenting, distinguishing, and identifying the patterns in the data by using special machine learning algorithms. Machine Learning is used in various applications such as Biometrics, image classification, medical applications, signal recognition etc. Over the previous decade, the wavelet theory has progressed immensely in the field of Signal Processing. Wavelets are the tools for analysing and investigating issues in various disciplines like music, computer vision, criminology, finance etc for spectral estimation, compression, classification, denoising etc[106].

4.1 INTRODUCTION TO WAVELETS

Wavelets are known as short time Fourier Transforms (STFT), this represents the frequency components of the signals in time domain. This localization and decomposition in time domain of a signal gives it the ability to deal with stationary and non – stationary data. The wavelet transform (WT) has a benefit of decomposing a time series into multiple sub – series. WT is performed in two domains i.e., continuous wavelet transforms (CWT), and discrete

wavelet transform (DWT). Since the dataset used for this thesis is discrete data, DWT is used to analyse the dataset. DWT essentially decomposes the data into high frequency and low frequency components which are also known as detail and approximate coefficients respectively [107]. A wavelet can be described as a small wave that comprises of oscillating attributes with its energy focused on a place in time, this is a valuable attribute for wavelet transform[108]. It grants the complex information to be decomposed and expressed in simpler form at various scales and points which can be reconstructed to the original data with high accuracy. Wavelet Transform is a tool used for processing signals in various scientific domains. This technique converts a time – domain signal to a time – frequency domain which are represented by wavelet coefficients which define the signal, which are associated to the mother wavelet function which is finite in time[109]. Wavelet Transform provide a better alternative than Fourier transform (FT) as FT provides higher resolution, either in time or frequency domain rather than higher resolution in both using WT. The mother wavelet and the level of decomposition of a signal are two main parameters of a wavelet transform, it uses the mother wavelet as a basis function. It performs the transform by shifting and dilating the function with respect to the time axis, the coefficients derived from the transformation are in time – frequency domain where the amplitudes are directly proportional to anomalies in the signal[109]. Many types of wavelet transforms are available to use but most used are Discrete Wavelet Transform (DWT) and Continuous Wavelet Transform (CWT).

4.2 Discrete Wavelet Transform (DWT)

Discrete Wavelet Transform is used extensively for classification/Recognition purposes. The DWT technique apart from recognition, is used in minimizing noise in the signals or data. It has the capabilities to compress the data while still retaining the important information by manifesting the signal's local time and frequency characteristics. Therefore, DWT and its variants are one of the most used techniques for signal processing. DWT is applied on a discretely sampled time series data that uses linear time i.e., time taken to process the data is proportional to the size of the input. The technique decomposes the signal into two parts namely approximation and detail coefficients[110]. The level of decomposition can pe provided during the coefficient calculation. The level of decomposition can be shown in the Figure 31 below:

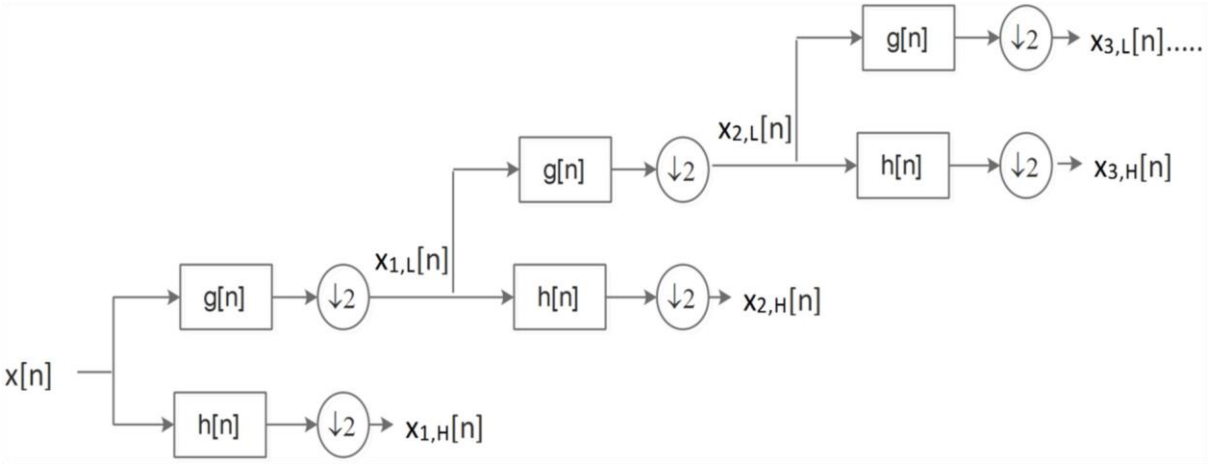


Figure 31 Multilevel DWT decomposition

Computationally, DWT works by filtering the signal through the high pass and low pass filters i.e., dividing the data into two-part frequency components. The approximate coefficients are the components from the low pass filter whereas the detail coefficients are the components from high pass filter, effectively dividing the signal into two-part frequency components. This constitutes the first level of DWT coefficients, then for the second level of coefficients, the approximate coefficients are passed through new set of filters and subsampled further by factor of 2 using half the cut – off from previous filter. With each increasing filtering level using DWT, dataset is down sampled which leads to less processing time for classification of data. For Example, let's take an application like event detection, when using the DWT technique, the down sampling of data, allows us to investigate high – frequency and low – frequency components separately which makes it easier to investigate two bands of frequency components. This filtering process uses two equations respectively for low pass and high pass filtering i.e., by using equations 1 and 2 to generate respective coefficients. The technique was first applied by Mallat [111] on the discrete dataset with t time points. DWT is easier to implement and computationally efficient compared to CWT. The variation is detected with more detail information at a particular wavelet level.

$$Approximate: \quad y[n] = \sum_{k=-\infty}^{\infty} x[k]g[2n-k]$$

Equation 1 Equation to calculate Approximate Coefficients

$$Detail: \quad y[n] = \sum_{k=-\infty}^{\infty} x[k]\ h[2n-k]$$

Equation 2 Equation to calculate Detail Coefficients

From the equations above, the g and h components correspond to the low – pass filter and high – pass filter respectively and are derived from mother wavelet.

4.3 CONTINUOUS WAVELET TRANSFORM (CWT)

DWT works over certain number of wavelet resolution levels unlike CWT. CWT provides the capability and flexibility to choose the wavelet scale in continuous domain. CWT can also be defined as cross correlation between two continuous signals, one being the original signal and second being the wavelet[112], this is represented by following equation:

- x(t) represents the signal.
- Ψ is a reference/ mother wavelet and Ψ^* is its conjugate.
- a and b are scales and time shifts of the reference wavelet
- t represents the time.
- T(a,b) represents the time – scale denotation of the signal.

$$T(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} x(t)\psi * \frac{(t-b)}{a}\, dt$$

Equation 3 Equation for CWT

The investigation of frequency component dictates the choosing of the length of the wavelet used for cross correlation. The longer wavelets are used to identify and investigate the low frequency components to improve frequency localization at the cost of time and similarly, the shorter wavelets are used to investigate higher frequency components that improve the localization of time at the cost of frequency.

Various reference wavelets are used to employ the frequency detection. These wavelets exhibit oscillatory patterns with the energy of one unit and whose mean is zero. The stretching and translation capabilities of mother wavelets, enables the technique to be used for analysis of signals from various domains. The CWT can also be reversed i.e., the inverse function to reconstruct the decomposed signal to build the original. Even though CWT has advantages,

using this technique for device recognition as the CWT is computationally expensive and requires more memory power. Hence, DWT technique is chosen to be explored further over CWT.

The next section reviews the applications where the wavelet transform is used to effectively classify data and identify events. The applications reviewed are like the work performed in the thesis.

4.4 Applications for Wavelet Transform

Wavelets are used in a variety of application domains for various purposes like Human Activity Recognition, Image/Video Processing, Event Detection, Condition Monitoring, Speech Recognition etc. It is impossible to mention each domain and technique used. Hence, we review the relevant applications where WT are used as building blocks for **identification or classification purposes** across various applications which are also applied to build a model to accurately predict the correct outcome.

Wavelets are used in following applications:

1. Signal Processing
    - **Audio Signals:** Wavelets can analyse non – stationary signals like audio signals. One – dimensional WT are popularly used to extract information and denoise the audio data. Simply subjecting audio signals to a Low Pass Filter (LPF) is not sufficient for restoring and enhancing the audio signals. Hence, the ability to generate multiple resolutions of time – frequency bands make the wavelets an asset to analyse audio signals. Paper [113] describes an audio signal constitutes of three parts i.e. transient, noise and harmonics which are separable/recognisable using wavelets. Denoising is performed by wavelet packet decomposition with frequency closely resembles critical subbands. Lapped cosine transform is used for tonal part estimation which is followed again by noise filtering. Lastly, the dyadic DWT is used to separate the transient part of the signal. This yields to the approximate and detail coefficients split

over lower levels. These coefficients are useful in estimating the variance of noises in various subbands.

2. Image/Video Processing
    - **Image/Video Processing:** Using wavelets for image/video processing works similarly to the compression techniques used for images and videos. The need of compression is to store/transfer the data and to reduce the redundancy [114]. It provides better quality of images after compression. Two types of techniques used in image compressions transformations and non – transforms. For transformations, the techniques used are Discrete Cosine Transform (DCT), Fast Fourier Transform (FFT), Wavelet Transform and Joint Photographic Experts Group (JPEG) and non – transform techniques like Differential Pulse Code Modulation (DPCM) and Pulse Code Modulation (PCM)[115]. Image compression is one of the most popular techniques used image/video-based applications.
    - **Machine Vision:** The authors in paper [116] have developed a machine vision system that detects fabric-defect in real-time using a DWT based algorithm. For their research work, an undyed raw denim fabric was used, and its real time images were recorded using a user interface developed in MATLAB. The feed forward neural network was used to classify the fabric images that were detected. The proposed algorithm comprised of a filter to remove the noise, DWT, Soft wavelet thresholding (SWT), double thresholding binarization, and dilation and then followed by an erosion operation. The approach was to first decompose the de-noised image frame (using Wiener filtering) into sub-images with the help of “db3-type” wavelets. After this, DWT is used to obtain the approximate image and the detail image from the four sub-images at a particular level of resolution. The noise is then removed from the approximate image using SWT. Then, with the help of averaging filter, the convolution of fabric sample frame is performed to obtain a smooth image. The next step involved the use of 2-D filter for enhancing the obtained image, where the texture in the background is attenuated and the required defective is accentuated. The resulting filtered image is then converted into a binary form.

For the binarization process, the authors used double thresholding method proposed in [117-119]. The proposed algorithm in [116], claimed to chieve an accuracy of 93.4 % to detect the defected and defect-free regions of the fabric, whereas it achieved 96.3% accuracy in classifying the fabric denim defects.

3. **Human Activity Recognition (HAR):** Recently, wavelets have been used extensively in HAR applications. The authors in [120] uses DWT coefficients as feature of an object. Furthermore, multi-class classifiers are used so that different activities or motions in the video frame can be classified effectively. Moreover, the discrete wavelet transform technique have also been utilised for liver and brain tumour classification [121]. Here, the authors propose a hybrid technique called CNN-DWT-LSTM, which combines the power of convolution neural network (CNN) in feature extraction, the power of discrete wavelet transform (DWT) in signal processing, and the power of long short-term memory (LSTM) in signal classification. The classification of the computed tomography (CT) images of the affected liver and Magnetic Resonance (MR) images of the brain tumours can be implemented using the proposed CNN-DWT-LSTM technique. It uses CNN to extract the features, and then uses DWT to have reduced dimension as well as strengthen the feature extraction performance. The classification is then followed by LSTM. The work proposed in [121]suggests having achieved 99.1 % accuracy and claims to be more efficient than the existing popular and powerful classifiers like K-nearest neighbours (KNN) and support vector machine (SVM).

   Another interesting study presented in [122] uses DWT for feature extraction to provide diagnosis of epileptic seizures using EEG data. The paper also demonstrates the results of the investigation performed using four feature extraction technique, namely, DWT -db4, DWT-db2, DWT-coif1 and Mel Frequency Cepstral Coefficients (MFCC), along with seven machine learning classifiers. This study also presents a comparison of the all the four feature extraction techniques used in the case of imbalanced as well as balanced datasets. According to this study, in the case of imbalanced datasets, DWT-db2 used with Random Forest (RF) performed the best, achieving classification accuracy of 98.99 %. However, DWT-db4 used with SVM or RF, also showed promising results by achieving the classification accuracy of 98.90 %.

Whereas in the case of balanced datasets, DWT-db4 with SVM achieved 98.99% of classification accuracy in comparison of the DWT-db2 with RF, who achieved 98.56%. A similar study presented in [123] uses DWT and RF algorithm for the classification of ECG beats. In this study, classification of five types of ECG beats was performed using 99.8% accuracy.

4. **Identifying load:** A useful study presented in [124], shows how DWT can be used to detect the load signatures from various electronic appliances. Their research work focussed on extracting features using DWT and then compare accuracies of different classifier used for appliances recognition. The load signature of each appliance was first recorded in terms of real power for number of observations. It involved using DWT, to reduce the data size, such that the approximate coefficients were extracted that represented the signal. The dataset is then divided into training set and test set. The classification was performed using 9 different classifiers, and its comparison in terms of accuracy and time taken is also presented in their work. The classifiers that were used in the experiment include K-nearest neighbours (KNN), Support vector machine (SVM), Decision tree classifier (DT), Random Forest classifier (RF), Adaboost classifier (ADA), Gradient Boosting classifier (GB), Gaussian NP (GNB) and Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA). These classifiers were implemented in the python with the help of sklearn module. The experimental results suggests that DT and GB both achieve 100% accuracy, however, DT was faster compared to the latter as it took 1.07 seconds to map all the database compared to the 115.70 seconds taken by the GB. KNN achieved an accuracy of 98.93% and took around 2.27 seconds, while the RF was the fastest amongst the classifiers used as it took 0.86 seconds, although the accuracy was lower than the KNN. QDA was the poor performing candidate amongst the other classifier as it achieved less than 20% accuracy. Thus, the proposed technique in [6], has a good potential in monitoring the load of the appliance, which in turn, could be used to save energy and reduce its consumption.

These applications are reviewed to understand how wavelets are applied per application and design our own algorithm to properly exploit the uniqueness from the features collected from

the Raspberry Pi devices. There is a two – part novelty here; first to use the simpler devices like Raspberry Pis to differentiate amongst them and second subjecting dynamic features to DWT along with the proposed classification technique MMVGD (introduced in chapter 3). The reviewed papers also depict the similarity i.e., identification of a certain entity, detecting an event etc. The next section provides the motivation to use wavelets for device identification.

4.5 METHODOLOGY

This section provides all the tests and observations made, used as explanation for using wavelets. It also describes each test performed and observation made. The features described in previous chapter are the ones used for further examination using wavelets. Wavelets are also used for non – stationary process, hence we need to check if the features of the multivariate dataset are either stationary or non – stationary in nature. This is done to understand the justification behind the lower results obtained by conventional classification methods. Hence, we perform the KPSS (Kwiatkowski, Phillips, Schmidt, and Shin)[125] test for stationarity. This test is used to test the stationarity. For multivariate data, each feature is tested for stationarity.

4.5.1 KPSS Test:

The KPSS test is performed using the inbuilt python library called statsmodels [126]. The KPSS test is used to test stationarity in a way for an observable time – series by testing it for a null hypothesis against alternative of a unit root. This means the absence of unit root is the null hypothesis. The test suggests that the absence of unit root means the absence of trend – stationarity. To determine the presence of randomness in the data we use the measurement of unit root. Any time – series is called stationary when its statistical properties remain constant over time. These are basic statistical properties like variance, correlation, mean etc. This means that relying on basic statistical properties is not practical. The table below shows the KPSS test performed on each selected feature of six devices the data is collected from. The p – value from the test is used to make inferences from the test.

KPSS Test Results				
Device 1				
Feature Name	*MemAvailable*	*Inactive(file)*	*Slab*	*SUnreclaim*
Test Statistics	5.43	5.01	0.525	0.854
p – value	0.01	0.01	0.01	0.01
Lags Used	31.00	31.00	31.00	31.00
Critical Value (10%)	0.347	0.347	0.347	0.347
Critical Value (5%)	0.463	0.463	0.463	0.463
Critical Value (2.5%)	0.574	0.574	0.574	0.574
Critical Value (1%)	0.739	0.739	0.739	0.739
Device 2				
Feature Name	*MemAvailable*	*Inactive(file)*	*Slab*	*SUnreclaim*
Test Statistics	0.754	3.338	3.238	2.545
p – value	0.01	0.01	0.01	0.01
Lags Used	26.00	26.00	26.00	26.00
Critical Value (10%)	0.347	0.347	0.347	0.347
Critical Value (5%)	0.463	0.463	0.463	0.463
Critical Value (2.5%)	0.574	0.574	0.574	0.574
Critical Value (1%)	0.739	0.739	0.739	0.739
Device 3				
Feature Name	*MemAvailable*	*Inactive(file)*	*Slab*	*SUnreclaim*
Test Statistics	0.496	4.181	1.606	0.542
p – value	0.01	0.01	0.01	0.01
Lags Used	28.00	28.00	28.00	28.00
Critical Value (10%)	0.347	0.347	0.347	0.347
Critical Value (5%)	0.463	0.463	0.463	0.463
Critical Value (2.5%)	0.574	0.574	0.574	0.574
Critical Value (1%)	0.739	0.739	0.739	0.739
Device 4				
Feature Name	*MemAvailable*	*Inactive(file)*	*Slab*	*SUnreclaim*
Test Statistics	4.01	4.43	3.971	4.596
p – value	0.01	0.01	0.01	0.01
Lags Used	25.00	25.00	25.00	25.00
Critical Value (10%)	0.347	0.347	0.347	0.347
Critical Value (5%)	0.463	0.463	0.463	0.463
Critical Value (2.5%)	0.574	0.574	0.574	0.574
Critical Value (1%)	0.739	0.739	0.739	0.739
Device 5				
Feature Name	*MemAvailable*	*Inactive(file)*	*Slab*	*SUnreclaim*
Test Statistics	7.503	8.321	3.534	8.659
p – value	0.01	0.01	0.01	0.01
Lags Used	31.00	31.00	31.00	31.00
Critical Value (10%)	0.347	0.347	0.347	0.347
Critical Value (5%)	0.463	0.463	0.463	0.463
Critical Value (2.5%)	0.574	0.574	0.574	0.574
Critical Value (1%)	0.739	0.739	0.739	0.739
Device 6				
Feature Name	*MemAvailable*	*Inactive(file)*	*Slab*	*SUnreclaim*
Test Statistics	5.408	4.607	5.199	4.458

p – value	0.01	0.01	0.01	0.01
Lags Used	31.00	31.00	31.00	31.00
Critical Value (10%)	0.347	0.347	0.347	0.347
Critical Value (5%)	0.463	0.463	0.463	0.463
Critical Value (2.5%)	0.574	0.574	0.574	0.574
Critical Value (1%)	0.739	0.739	0.739	0.739

Table 6 KPSS test results on each selected feature

The Table 6 above shows the results of KPSS test. The KPSS test is inferred using the p – value of the test. If the p – value is less than the alpha value (0.05), which is pre – defined, then the null hypothesis is rejected. Following are the conditions for non – stationary data:

- Critical Value (5%) < Test Statistic
- P – value < 0.05

This is true for each feature of each device. Hence from KPSS test, we conclude that the data is non – stationary. Hence, we use wavelets to derive characteristics and build basis for classification model.

4.5.2 Methodology Used:

This section explains the methodology using wavelets is the same as the methodology used in the ICMetrics chapter with some additional steps. The main difference between them is using wavelets to generate the features used for the classification. Below is the flowchart that shows step by step procedure to use DWT on the dataset to get the coefficients and further analyse them.

The flow chart in Figure 32 gives the entire working of the system. The steps represent the overview of the system design. Each of the processes shown in the flowchart, which are repeated from the previous chapter. The additional steps here are applying the wavelets to the dataset. The features collected from Raspberry Pi are the memory based operational features which are subjected to DWT using simple wavelets.

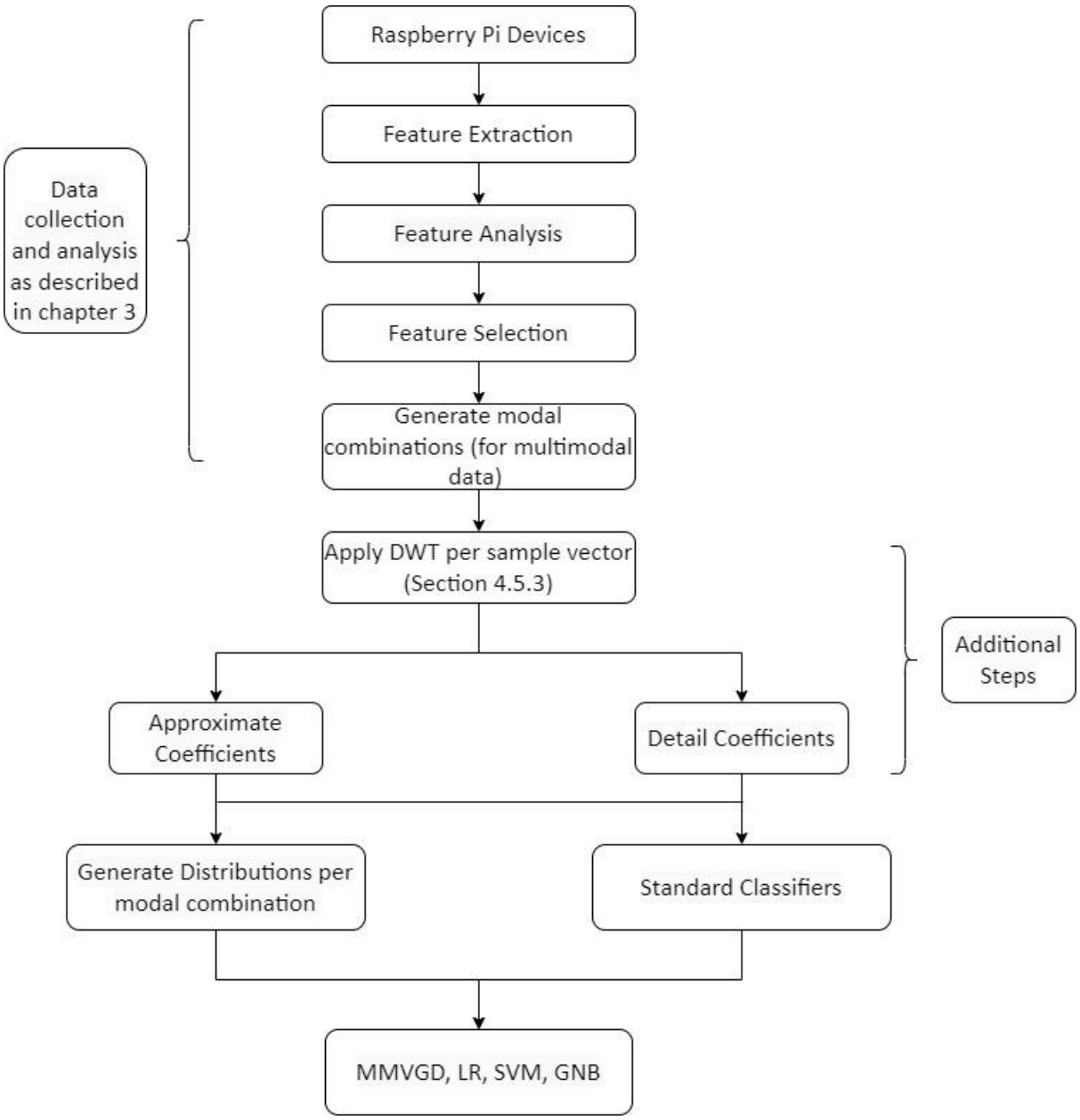


Figure 32 Flowchart showing each step of the system design using wavelets for classification.

There 38 features in total which are collected and each one is examined individually. This is also described in Table 2 and Table 3 shows the distribution of each feature after eliminating the static features. We select four features out of 38, and subject them to Discrete Wavelet transform. Since the data is non – stationary, wavelets are used to convert the data in frequency domain to perform the classification.

4.5.3 Applying the Wavelets

This section provides the detailed explanation on the process of applying wavelets. From chapter 3 it was observed that the relationship amongst features is of grave importance, Hence the wavelets are used across the feature vector to get the coefficients. Since the features do not exhibit any observable patterns visually like periodicity, dependency etc. from the plots of the data. The novelty in this chapter is using Wavelet Transform across the features to observe the variation in patterns exhibited amongst features of the dataset.

The flowchart above shows the step – by – step procedure of using wavelets. The steps listed here are followed in sequence to apply wavelets:

i. The initial steps from the flowchart are the same as followed from chapter 3 i.e., feature extraction, feature analysis and feature selection. These three steps are applied exactly as mentioned in chapter 3 without any changes.
ii. After feature selection, the first step is to apply peak – trough to build modal combinations to address multimodal features. The generation of these modal combinations are addressed in section 3.4.1
iii. Each modal combination is then subjected to Discrete Wavelet Transform (DWT), one sample vector at a time.
    a. For instance, let us consider a sample vector, X = [188172, 64892, 55072, 31988]
    b. Applying Haar wavelet transform to the sample vector X.
    c. Let us calculate the Approximate (A) and Detail (D) coefficients using the following equations:

    1st Level Coefficients:

$$A(1) = \frac{X(2k)+X(2K+1)}{\sqrt{2}} \qquad (K= 0,1,2,3) \text{ ----------(1)}$$

$$D(1) = \frac{X(2k)-X(2K+1)}{\sqrt{2}} \qquad (K= 0,1,2,3) \text{ ----------(2)}$$

    d. Hence the resulting coefficients are:

    Approximate (A): [178943.2705, 61560.71637]

Detail (D): [87172.12398, 16322.8529]

iv. Each modal combination's distribution is then generated. (Separate distributions for Approximate and Detail coefficients are generated)

v. For testing phase, each sample vector gives two arrays of coefficients. One array consists of resulting approximate coefficients and other of detail

vi. Then the resulting approximate coefficients from the sample vector are tested against approximate modal combination distribution. This step is repeated separately for detail coefficients i.e., to test detail coefficient samples against the detail distributions from training samples.

vii. All the steps from above are repeated using three wavelets i.e., Symlet, Daubechies and Haar wavelets. The criteria for choosing these wavelets are as follows:

- Sym and Db family are good for Denoising Signals
- Haar wavelet is computationally efficient and faster
- None of the features show periodicity, hence Db wavelet is also chosen for its unsymmetrical nature
- Symlet is near symmetrical and chosen for insufficient periodicity [127]

For each wavelet family chosen, we choose the first wavelet from their family as these require smaller support which means they are inexpensive in terms of computation and have simple implementation technique. Since we are choosing to use these on resource constrained environment, this becomes one of the important reasons to use each of the selected wavelets from its family.

4.5.3.1 Daubechies Wavelet (db):

The Daubechies (db) wavelet was first introduced by the author Ingrid Daubechies[128]. The member wavelets of 'db' family are represented by 'dbN' where 'N' represents the order of the wavelet[129]. The order for db families using N where N = 1, 2, 3, 4, 5, 6, 8, 9, 10. The first wavelet in db family i.e., db1 is the first order (basic) wavelet of the db family. This is also called the 'Haar' wavelet. Most of the db wavelets are unsymmetrical in nature and hence are used on non – periodic data to perform analysis. The Figure 33 below shows the scaling function and mother wavelet for Db2 wavelet.

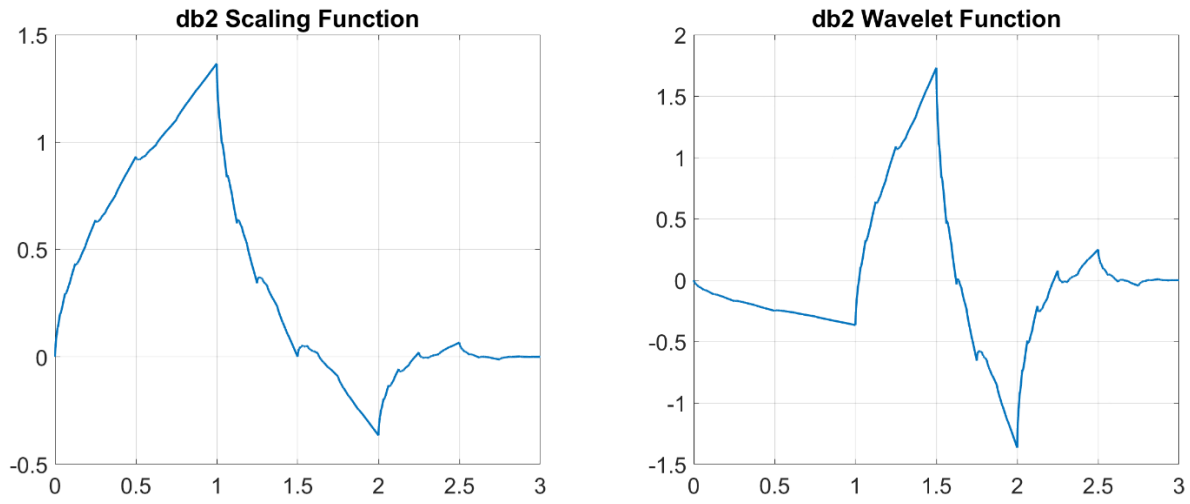


Figure 33 Graphical representation of Scaling and Mother wavelet (Db2) function

4.5.3.2 Symlet Wavelet (sym):

The Symlet wavelet is an extension of the Daubechies wavelet. The sym wavelet was invented due to the unsymmetrical properties of the 'db' wavelet, so the sym wavelet is least asymmetrical in nature and is denoted by 'symN' where N denotes the order of wavelet. The symlet family has seven members which are denoted by N where N = 2, 3, 4, 5, 6, 7, 8. These are compactly supported in time and are orthogonal in nature[129]. The Figure 34 below shows the scaling function and mother wavelet for Sym3 wavelet.

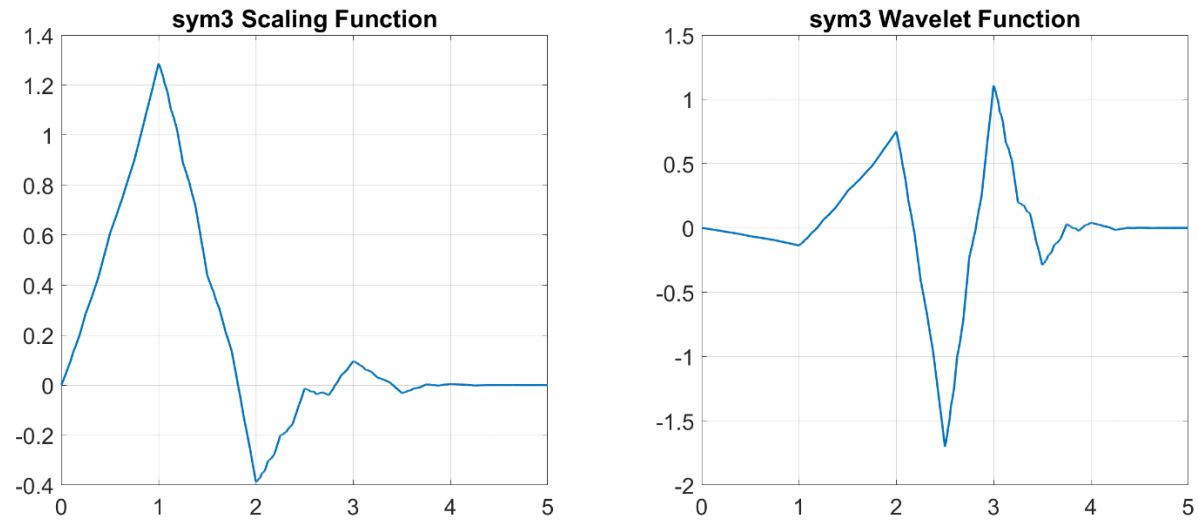


Figure 34 Graphical representation of Scaling and Mother wavelet (Sym3) function

4.5.3.3 Haar Wavelet (haar):

The simplest type of wavelet is the Haar wavelet and was presented by Alfred Haar[130]. It has a few advantages:

- The simplicity of the wavelet makes the processing faster
- Its simplicity makes them memory efficient

It has two functions i.e., scaling function and wavelet function. These are used for decomposing the signal into two sublevels, these are also known as the average sublevel and the difference sublevel. The Figure 35 below shows the scaling function and mother wavelet function of haar wavelet.

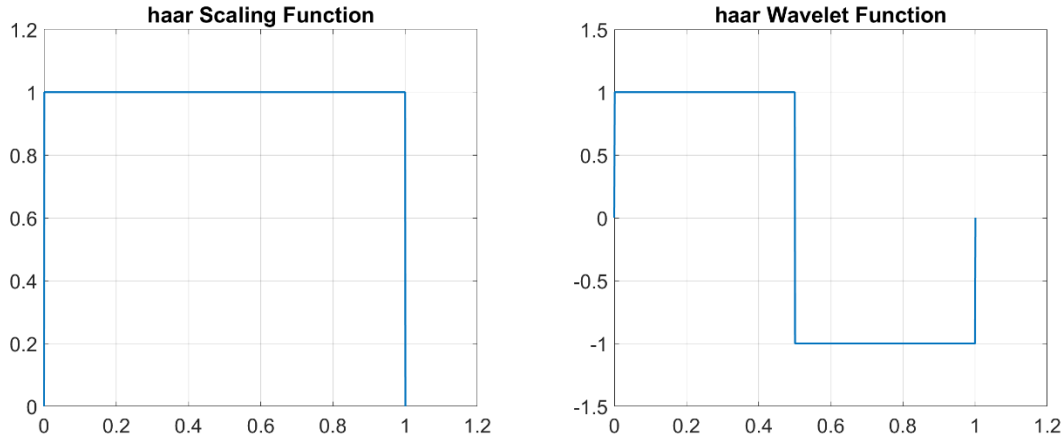


Figure 35 Graphical representation of Scaling and Mother wavelet (haar) function

4.5.4 Algorithm:

The algorithm description provides the details of how the data is processed before being used for classification. The following steps are followed in order when subjecting the data to DWT and use for classification:

1. Collect the data from Raspberry Pi, subject each feature for further analysis (described in previous chapter).
2. Feature analysis is followed by feature selection (criteria mentioned in chapter 3 in feature selection section)
3. Split the data into training and testing with 80:20 split respectively.

4. Apply Peak – Trough algorithm to detect the number of modes per feature using training data and separate the datasets into modal combinations (explained in section 3.4.1).
5. Data from each modal combination is then subjected to DWT using Haar wavelet and generate distribution for each modal combination.
    a. Each Sample Vector is subjected to DWT using Haar
    b. Each set of coefficients i.e., Approximate and Detail are used separately for classification. (Explained in detail in section 4.5.3 – Applying the wavelets)
6. Calculate the accuracy by using the training and testing data.
7. Lastly, use the standard classifiers (SVM, LR, GNB) to benchmark the results with respect to proposed classification method (MMVGD).
8. Repeat steps 5 to 8 for detail coefficients.
9. Repeat steps 4 to 8 for each selected wavelet i.e., Sym3 and Db2.

4.6 RESULTS

The classification is conducted after subjecting the raw features to DWT using three different wavelets to identify which wavelets work best for device identification. The features are subjected to DWT function to get first order approximate and detail coefficients. The experiment is performed using the features collected from 6 Raspberry Pi's. The data is collected in two different Pi usage scenarios:

Scenario 1: When the device is actively using its sensors and transmitting the data.

Scenario 2: When the device is actively deployed but only running the background processes.

The Table 7 below shows the device number and the scenario used to collect its data.

Device Number	Data Collection conditions
1	Scenario 1
2	Scenario 1
3	Scenario 1
4	Scenario 2
5	Scenario 2
6	Scenario 2

Table 7 Data collection scenario for each device

The classification metrics contains four parameters overall i.e., Accuracy, Precision, Recall, F1 score. The definitions of each of these are presented in 'Result' section in chapter 3.

The Table 8 below shows the classification metrics when wavelet – based features are subjected to the proposed classification technique and these results are benchmarked using the standard state of the art sklearn classifiers.

The table below shows the classification metrics achieved when the approximate coefficients of the three chosen wavelets i.e., sym3, haar and db2 are subjected to classification. Using the metrics, it is evident that MMVGD classification technique produces the highest results with each chosen wavelet. The Sym3 and Db2 produce highest results with 98.47% and 97.2% respectively, whereas haar wavelet produces the 83.4% accuracy. We also observe that the SVM classification technique produces comparable results with MMVGD. LR produces least accuracy for haar wavelet and show comparable results for sym3 and db2 wavelets. Sym3 achieves best results when using approximate coefficients but there is not much difference in results achieved by using MMVGD for db2 as well.

Sym3 Wavelet (Approximation)				
Classifier	Accuracy	Precision	Recall	F1 Score
MMVGD	98.47%	99.00%	98.00%	98.00%
LR	64.00%	59.00%	64.00%	60.00%
GNB	79.00%	79.00%	79.00%	79.00%
SVM	89.0%	89.00%	89.00%	89.00%
Haar Wavelet (Approximation)				
Classifier	Accuracy	Precision	Recall	F1 Score
MMVGD	83.40%	77.00%	83.00%	89.00%
LR	29.00%	15.00%	29.00%	18.00%
GNB	61.00%	61.00%	61.00%	59.00%
SVM	70.00%	64.00%	70.00%	66.00%
Db2 Wavelet (Approximation)				
Classifier	Accuracy	Precision	Recall	F1 Score
MMVGD	97.20%	97.00%	97.00%	97.00%
LR	64.00%	59.00%	64.00%	60.00%
GNB	84.00%	84.00%	84.00%	84.00%
SVM	89.00%	89.00%	89.00%	89.00%

Table 8 Classification metrics for wavelet - based features using approximate coefficients.

Similarly, we compare the results obtained using detail coefficients when subjected to classification. The Table 9 below shows the overall classification metrics to generate results. The results show that the MMVGD technique, that addresses the multimodal nature of the datasets yield best results with proposed classification technique. For detail coefficients, Db2 achieves best results compared to sym3 and haar. The LR classification technique achieves least accuracy, whereas GNB and SVM achieve comparable results with all the wavelets. Db2 wavelet achieves best result using detail coefficients.

Sym3 Wavelet (Detail)				
Classifier	Accuracy	Precision	Recall	F1 Score
MMVGD	93%	93%	93%	93%
LR	59%	51%	59%	54%
GNB	61%	69%	61%	59%
SVM	74%	72%	74%	71%
haar Wavelet (Detail)				
Classifier	Accuracy	Precision	Recall	F1 Score
MMVGD	91.33%	92%	91%	91%
LR	35%	20%	35%	25%
GNB	57%	60%	57%	54%
SVM	63%	65%	63%	59%
Db2 Wavelet (Detail)				
Classifier	Accuracy	Precision	Recall	F1 Score
MMVGD	98%	98%	98%	98%
LR	62%	56%	62%	58%
GNB	62%	63%	62%	59%
SVM	72%	71%	72%	69%

Table 9 Classification metrics for wavelet - based features using detail coefficients.

The technique described in this chapter provides an alternative to the one discussed in chapter 3. Hence, we compare the results obtained by using wavelet-based features and raw features, it is observed that using the raw features show better results. The bar graph below in Figure 36 shows the comparison of overall accuracy achieved using the wavelet features and raw features.

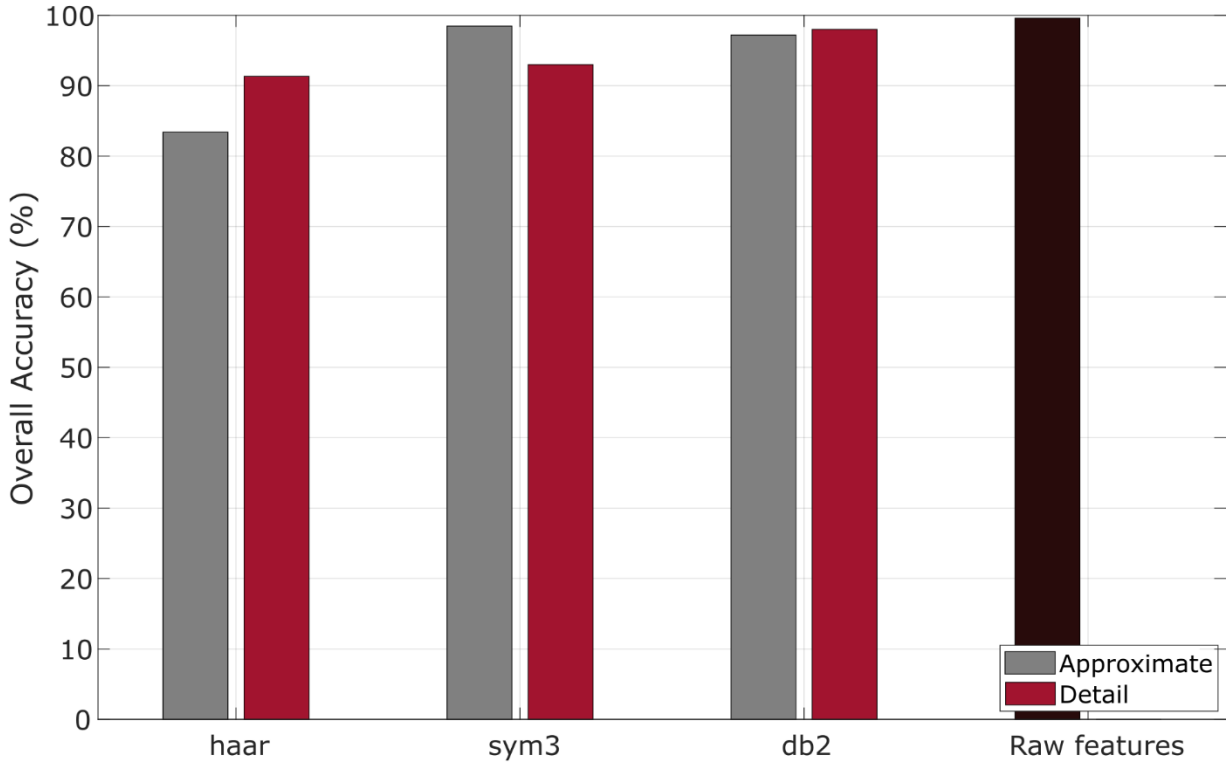


Figure 36 Comparison for overall accuracy obtained using wavelet features and raw features using MMVGD classifier.

Although, it is evident that the raw features produce better results, but it is also clear that there is not much difference between the results achieved.

4.7 SUMMARY

This chapter presents a novel method of using wavelet – based features to identify the devices. The initial steps before applying wavelets remain same as described in chapter 3 i.e., Data collection, Data Analysis and Data Selection process. After selecting the data, the features are analysed in multidimensional space to build modal combinations.

Each modal combination is subjected to Discrete Wavelet Transform per sample vector to observe its distribution in multi-dimensional space so each modal combination can be identified separately. When looking at each test sample, probability is calculated against each modal distribution to identify which distribution the test sample belongs to.

Three reasons to use wavelets are:

1. The importance of using wavelets across the sample vector to understand the dynamic relationship amongst the features.
2. Decreased computation time, as it approximates the data and reduces the dimension of the feature set.
    a. The approximate coefficients have smaller dimensions when compared with the raw feature dimensions, meaning the smaller number of features with lesser number of modal combinations observed when compared to modal combinations observed in raw features. This in – turn affects the time taken to process the data.
    b. The detail coefficients use the high frequency components of the data which converts the feature data into a smaller number of modes compared to raw features and approximate features. This in – turn affects the time taken to process the data.
    c. The reduced number of modal combinations means less computation power required to build the model.
3. Non – stationary nature of the data collected – shows no repeatable patterns visually.

When subjecting the wavelet – based features for classification, we observe that the proposed classification technique MMVGD achieves the highest results with about 97% which is higher than the other standard sklearn classifiers.

4.8 CONCLUSION

A novel method of using wavelets for identification is proposed in this chapter. By observing the visual representation of data, we observe that the data from the devices follow no definite pattern and from the KPSS test performed, in this chapter we also understand that the data collected from the devices is non – stationary in nature. One thing can be said with certainty is that the features are variably related to each other based on the state/working of the device. Hence, wavelet – based features are used as an alternative technique for identification of the devices.

Here the data collection, analysis and selection techniques remain the same. Then the peak – trough algorithm is used to bifurcate the data as per the modal combinations. These combinations might represent various working stages of the algorithm. Hence, the basic statistical parameters for each modal combination like mean, variance, covariance, and correlations amongst the features would be different. By using wavelets, we can observe the dynamicity amongst features, spatial representation using approximate/detail coefficients and generate distributions.

Wavelets are also used to reduce dimensions of the data which in turn reduces the computation time, but this can only be said certainly by repeating experiments with increasing number of devices using wavelet – based and raw features for comparison. By using the proposed classification method (MMVGD) and separating modal distributions we achieve a higher accuracy of 97% using approximate coefficients and 98% using detail coefficients. This is lower than the accuracy percentage achieved by applying MMVGD classifier while using raw features which was 99.6%. But the accuracy results are not vastly different for raw features and wavelet – based features.

The results from other classifiers used for benchmarking show slightly higher results with wavelet-based features when sym3 and db2 are used whereas show lower results with haar wavelet. This can be due to the number of resulting dimensions of the features after applying DWT. The resulting dimensions using sym3 and db2 remain same while for haar wavelet the dimensions decrease. Although these results can be interpreted in a way, but limitations of these interpretations are considered due to the limited availability of real datasets.

5 SYNTHETIC DATASET GENERATION – MULTIVARIATE MULTIMODAL DATA FOR VALIDATION

Synthetic Datasets are an important part of the world today. Datasets are an integral part of the Artificial Intelligence (AI) community and building effective Machine Learning (ML) models. Since the number of devices, available for data collection are small, the conclusions cannot be drawn without suspicion. This questions the authenticity of the algorithm developed to address multimodal datasets i.e., if the modal boundaries are appropriately recognised, the accuracy percentage of correctly classified data and if the classification results are a fluke. Hence the motivation to validate the output from the algorithm i.e., to observe and validate crucial steps of the algorithm which are building blocks of the classification model.

Numerous organisations acquire/require number of datasets, which are used for analysis, therefore used in ML and DL (Deep Learning) models to build and deploy state of the art technology to resolve challenging issues. Useful data is collected from variety of applications i.e., Medical, finance, Computer Vision or any IoT based application one can think of, but the primary challenge faced here is acquiring the real data from these applications as the data to be acquired is sensitive in nature and protected by privacy laws to protect individuals from identity fraud, fraudulent authentication or number of risks that come along with data leaks. Hence, the need to generate synthetic data that mimics real data and can be used in place of real data to build effective ML models as they use a significant amount of data for training models.

There are three categories of synthetic datasets that can be built:

- **Fully synthetic data:** The data generated in this category is solely synthetic in nature, i.e., no relational or any kind of similarity with original data. For purely synthetic data, the density per feature is identified which is used to estimate statistical parameters to draw the samples randomly from the distribution. This is used when the data is private and sensitive in nature. Hence, the protected features or all the features from the confidential data can remain safe. A full synthetic data is synthesised when the data is

supposed to be protected using privacy laws, so the data is replaced with synthetic data and used so the technique adheres strong privacy issues.

- **Partially synthetic data:** This technique is used when only a few features of the data are at higher risk of identity disclosure and are replaced with synthetic values. Hence, privacy laws are preserved, and a partial synthetic data is generated.
- **Hybrid synthetic data:** The hybrid data generation method uses the synthetic data and the real data together to generate a new dataset. For each real sample vector, a close representation of the real data is a generated synthetic sample vector and combined to form the hybrid dataset. It has advantages of both the synthetic and real data. This leads to privacy being preserved with higher efficiency but at the cost of processing time.

The key purpose of using synthetic data is to control the privacy of the data which is achieved by using one of the three methods described above and utilise the data for numerous applications.

There are various techniques to generate synthetic data, which can be categorised into three parts:

1. **Statistical Distribution based**: As the name suggests, the technique observes the statistical distributions of the real data and generate synthetic data samples using similar distribution. In a few use cases, sometime the data is simply not available or privacy laws in place do not allow the data to be shared. Any individual who understands the statistical distribution of the real data thoroughly, can recreate similar distribution by generating random samples from the distribution. This can be accomplished using probability distributions like lognormal, normal, exponential etc.

2. **Based on an agent to model:** The technique generates a model which explains the behaviour of the data which is observed in the real data and then generates random synthetic data using the same model. It is also recognized as using a known distribution to fit actual data. The Monte Carlo method for synthetic data generation is one of the popular methods for synthetic data generation. Some machine learning models like Decision Trees, are also used to generate data fitting a distribution. Using

Decision Trees can lead to data overfitting, so one must adhere caution while using these. When some of the real data is available, the corporations use hybrid synthetic data method to build datasets.

3. **Deep Learning based**: The deep learning technique either uses VAE (Variational Autoencoder) or GAN (Generative Adversarial Network) models or the Data Augmentation technique.

   The GAN models consist of two network models (neural network), one is the generator network, and the other is the discriminator network. The generator, generates the synthetic dataset and the discriminator, tries to determine if the dataset is fake by assessing it with respect to the dataset generated by the generator. When the fake dataset is identified, the generator is alerted which then leads to generator altering the next set of the data supplied to the discriminator. This improves the capabilities of the discriminator to identify the fake dataset. The GAN models are normally used in detecting fraudulent data and in healthcare for medical imaging.

   The VAE ML (Machine Learning) models are unsupervised type which are used to compress original data using the encoders and the decoders are used to perform analysis of the original data and generate parameters representing the actual data. The goal of VAE is to ensure the similarity between the input and output datasets.

   The Data Augmentation is an additional technique used by the researchers to add some new data to the current (real) data. This is commonly used to generate additional images (like the real images) with changed orientation, zoomed image, brightness etc. The technique can also be used to anonymise data to remove personal identification information. This technique cannot exactly be used to build synthetic data but can be used to anonymise it.

The next section describes the characteristics required in the synthetic data which were observed from the real data when collected from the Raspberry Pi devices.

5.1 DATA CHARACTERISTICS

The data analysis section in Chapter 2 shows the dynamic nature of the features collected. Using the Probability Density Function, we build the graphs to understand the visual distribution of the features. Hence, using synthetic data to derive those distributions to generate correlated data.

5.1.1 Characteristics Observed and Analytics:

To build the synthetic datasets that show the characteristics observed from the data of the real devices, we check the distributions of each selected feature of the selected feature set from all six devices. Each device yields similar distributions of the four features chosen. Most of the features from the data collected are highly dynamic in nature so we use only four features in the real data to prevent overfitting and to prevent building a model which is computationally expensive. Hence, we use the synthetic data generator to generate four features per dataset. Following are the four features we analyse for the distribution to model the synthetic data on:

- MemAvailable (F1 – Feature 1)
- Inactive(file) (F2)
- Slab (F3)
- SUnreclaim (F4)

The features MemAvailable and Inactive(file) are highly dynamic in nature since the change in working of the device reflects in the feature data, whereas features F3 and F4 are stable compared to F1 and F2. When the PDF (Probability Density Function) plot is generated, the features F1 and F2 are multimodal in nature but F3 and F4 are largely unimodal/bimodal. The Figure 37 below is the example of the MemAvailable (F1) feature for all 6 devices.

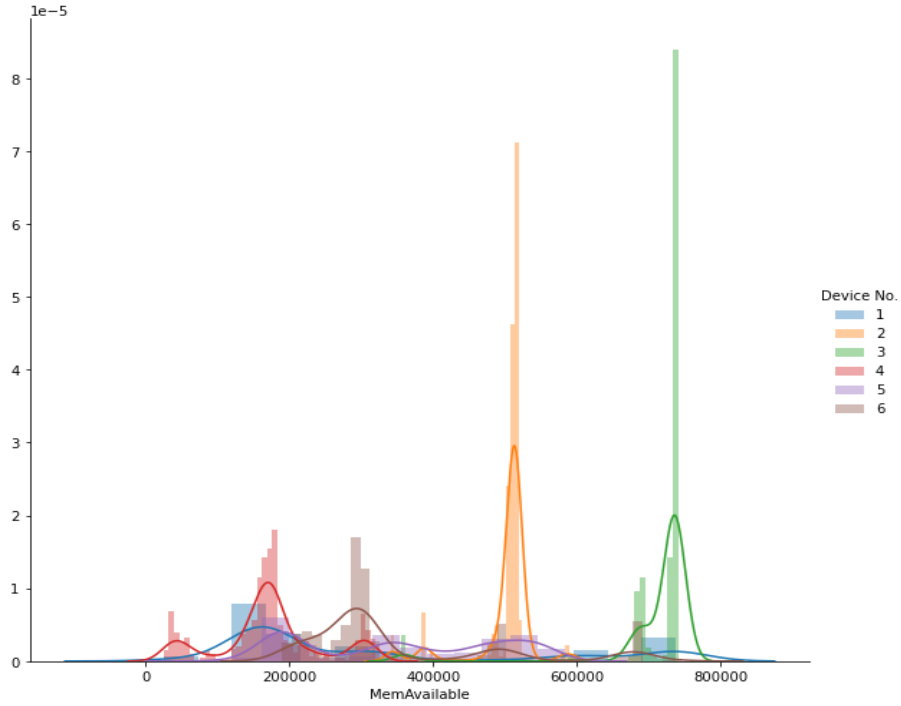


Figure 37 PDF graph for MemAvailable feature (All 6 devices)

The PDF graphs of the Inactive(file), Slab and SUnreclaim are showed in Figure 13, Figure 14, and Figure 15. These figures are used as base to understand the underlying distribution. Since the data is multimodal in nature, we cannot simply perform descriptive statistics (such as calculating mean, median, mode, correlation, or covariance) on the raw data directly. Hence, we use peak – trough algorithm to separate modes and then perform the statistical analysis as each feature mode can be related differently to other feature modes to address the multimodal features as explained in chapter 3. Hence, the next section reviews the previously used techniques to generate the synthetic data.

5.2 Related Work

The aim of this work is to generate the synthetic data distributions like the ones observed from the real data (collected using Raspberry Pi) as mentioned in previous chapter. Hence, we review the techniques used by researchers to understand and select the proper method to generate multivariate correlated data. This is an unconventional option, but effective to prevent the data breach.

There are various methods to generate synthetic dataset, paper [131] reviews four popular synthetic data generators namely, Data Synthesizer[132], Synthpop (non – parametric and

parametric)[133] and Synthetic Data Vault[134]. Authors of [131] evaluate the four data generators on a few empirical observations affect the utility of synthetic datasets:

1. Effect of data pre – processing
2. Effect of sharing primitive ML results
3. Effect of parameter tuning (and selecting features) – generating supervised models
4. Effect of propensity on classification accuracy

Each of the four data generators use different type of techniques to build datasets, the DataSynthesizer [132] uses Bayesian Network which uses correlation structure and greedy approach to construct the network that generated correlated variables. Then the samples are drawn to generate the categorical or non – categorical data. This model is generated using python and is publicly available. Paper [134] presents SVD (Synthetic Data Vault) built using Python, which estimates joint distribution of the data. This distribution is estimated using gaussian copulas[135]. The covariance function matrix is expressed as the gaussian copula which are the dependencies amongst the features. It draws the samples from the cumulative distribution of the population by specifying each feature's marginal distribution. The SDV produces data on the notion that all the attributes are numerical. When the data is non – categorical in nature, the values of the data are converted into numerical values between [0,1] using primitive pre – processing techniques based on the frequency of its occurrence.

The generator Synthpop (SP) was developed using R language[136] is built in two parts i.e., parametric and non – parametric. The Synthpop generates the dataset one feature at a time. It generates the first feature by estimating the marginal distribution and generate the synthetic records for the first feature. Then the generator generates next feature by using conditional distribution of F1 and F2 (Feature 1 and Feature 2 respectively). The next feature is generated using the conditional distribution and synthesized values of the first feature and generates next feature. This process is repeated for each subsequent feature is generated using the conditional distribution and synthesized values of all previous columns[131]. Using conditional distributions, SP presents two methods of generating synthetic data (as mentioned previously), the parametric method; this uses the logistic regression alongside linear regression whereas the non – parametric approach uses CART (Classification and Regression Trees) algorithm.

The empirical experiments performed by the authors of [131] to check the efficiency of the synthetic dataset. The propensity score is calculated for each synthetic data generator while performing the empirical experiments to understand the best method to generate datasets. The measure of best dataset yield (closest to the original dataset) is performed using propensity score. The propensity score is the measure of probabilities of data memberships, which is done by building the binary classification model to distinguish between the synthetic and original datasets. The propensity closer to zero shows the data is closer to the original dataset.

The first empirical experiment was performed to check the effect of pre – processing on original data, but the best synthetic datasets were generated with the least distinguishability while using the raw original data as the pre-processing of the dataset yields to overfitting. The second experiment was performed to understand the effect of sharing primitive ML results i.e., Real data tuning settings. The best results were generated when the features were selected, and hyper parameter tuning was performed and shared when generating the dataset using the synthetic generators. The third experiment about the hyper – parameter tuning settings yielded no conclusive results. The fourth experiment shows that the propensity scores are a good measure for prediction accuracy when the data is generated using Synthpop – p and DataSynthesizer, hence can be used for benchmarking.

Another unique synthetic data generation method was proposed by authors of [137]. The generation uses decision tree model to produce the data by using an XML based language called SDDL (Synthetic Data Definition Language) to generate the data model. PMML (Predictive Model Markup Language) is used to transfer and represent multiple data mining models. These models use the dictionary that describes the field, their datatypes and gives the kind of values that can be contained in each field. The authors convert the PMML tree model to SDDL by using a Parser, then the synthetic data is generated using the SDDL file and classification is performed. Even though the application of using decision trees for a novel architecture, does raises concerns about the privacy and since the decision trees are constructed using a particular attribute, the user cannot use a different characteristic for a class variable.

Similarly, in [138] authors use Random Forest classifier to generate a partial synthetic data by using the classifier modelled on the original. The model is then tuned to generate the partial

synthetic data to maintain the privacy and only generates the categorical data. Since Random Forest also uses tree structure it faces the similar problem while generating data using tree-based structure, the classifier is modelled using class variable selected. Thus, when the requirements are changed, the algorithm is adjusted and reused which makes the process time – consuming.

Paper [139] proposes a basis to generate a fully synthetic multivariate data. It uses the statistical distributions in a multidimensional space by using the dimensions provided by the user. The framework uses the PDF (Probability Density Functions) to generate the samples from the distributions. The generator only generates numerical (integer and float) samples. The generator uses a 1D, 2D and a 3D PDF to manipulate projections. 1D generator is used to handle each feature/dimension separately, 2D is used to control orthogonal projections of two components (defines conditional probabilities), whereas 3D object generator is also known as PDP (Probability Distribution Plane). The generation technique also adds noise to the data which is controlled by user. The overall parameters such as number of classes, number of features/dimensions, number of samples and type of distribution are to be specified. The focus of this system is to allow user to control the graphical characteristics of the dataset. The time taken to generate the data increases as dimensions of the dataset increase. Another method of generating synthetic data is proposed by the authors of [140] by using cardinality constraints. They use generalization of Linear Program (LP) is done to accommodate multiple attributes, which leads to exponentially enormous LP so, the authors reduce the size by using graphical models. The constraints are specified by user during the data generation. Although the algorithm can handle constraints injected by the user but has not been tested or modelled using any real dataset. Authors also mention the future development of their work could be in the direction of handling overlapping projections with the constraints.

Most of the synthetic data algorithms are used to generate data where privacy is a concern, data is inaccessible or only small dataset is available. These issues are mostly present while using some personal health-based data. So, even though the generation of synthetic data is the priority, protecting the data is the next on the list. The Researchers while generating the synthetic data, also take in account to check the similarities between generated dataset and original dataset create a risk of disclosure. Hence authors also look at the privacy concerns

when generating the data. To protect the user's privacy, a two stage method was proposed in paper [141]. Two stage approach is proposed based on the disclosure risk performed by the authors and their previous privacy-based work which determined that sharing large number of synthetic datasets can enhance the inferences. The multiple imputation technique causes large amount of missing information so, the generation of synthetic data is justified. The generated data should also match the security measures, so for each generated dataset the security measures are checked which becomes a tedious task. Hence the two-stage approach was proposed. So, stratified sampling is used to draw samples from the original data and some attribute values are imputed; second stage uses initially imputed values to impute the remaining attribute values. This is used to impute missing values and in generating partial and full synthetic dataset. The two – stage approach increases the execution time of the system. One of the popular approaches used to protect privacy is by using the Differential Privacy method which is used in [142] to generate a fully synthetic dataset where the noise is added after the generalisation of the data. The probabilistic method is used to generalize the raw data and combines the noise to ensure Differential Privacy. The authors also conclude the method to yield degrading results for a potentially large dataset.

A Context Aware Recommendation System (CARS) was proposed by authors of [143] to generate a full synthetic dataset generated based on users ratings or description or their associations. This is done via PDF functions defined by the user and contains a penalization function to adjust the data when the difference is too large for the context being used. Hence, taking in account the outliers of the data. The algorithm does not use real data and depends solely on user's specification. The conjunction of the optimum value for penalization function is required to increase accuracy which increases the time consumption and makes it susceptible to outliers.

Another Differential Privacy based method called 'PrivBayes' was proposed in paper [144] to generate a partial or full synthetic dataset, which is based on using Bayesian Network to model the correlation between data features to release a high dimensional data. It uses low dimensional marginal to approximate the data distribution. They then add the noise to the approximated data to comply with Differential Privacy and draw the samples. The dataset drawn is susceptible to errors because the samples are drawn after the noise addition. Generation of high dimensional data leads to the exponential growth of the Bayesian network

graph. Privacy is also a major issue when it comes to the database systems and their performance evaluation. The database performance depends upon the workload of the query and the data it contains. This data is usually unobtainable due to the privacy reasons. Paper [145] proposes a two stage model using a Differential Private technique to generate full synthetic data for release. The authors generate a query-based model of the data using its statistical distributions for the queried dataset. This model is then subjected to differential privacy method to generate the data releasable to the user. Each new query supplied by the user generates a new working model of the data which would be time consuming. Specially in the case when the data has large number of dimensions, as the repetitive method perturbation technique is used to get the best possible dataset for release. Authors of [146, 147] propose a synthetic data generation of categorical data using PeGS (Perturbed Gibbs Sampler). The method uses the data and builds the statistical blocks after its disintegration (using the l-diversity or Differential Privacy). These blocks represent the original data, to which then the noise is added, and the data is modified. Hence, data generated from altered statistical blocks of the original data. The generator only uses the categorical variables, so all the numerical type variables are converted to categorical. The hashing of data can be used with the perturbation technique to provide additional protection to the data but as concluded by the author, ε – Differential Privacy and *l* – diversity[3] are marginally better than multiple imputation perturbation technique. Overall processing time increases with the Perturbed Multiple Imputation (PMI) when dealing with a large dimensional data. The statistical model generated for the dataset is memory expensive as it stores every statistical information of the data.

Human behaviour data is another important application where synthetic data generation would be beneficial since it can protect the user's privacy. Paper [148] proposes a method that uses the Differential Privacy (DP) technique to protect the privacy and generate full synthetic dataset for human behaviour data. The DP technique uses the Laplace noise which is to be added after the generation of each feature's histogram to comply with the user's privacy. It uses the hubway trip history data to generate synthetic data which preserves individual's security. The model does not support personalisation during data generation and does not have a generalised model i.e., only works with trip data.

The approaches mentioned above are popularly used to generate the synthetic data, also by generating/variating one attribute at a time based on real samples or distributions provided by the users. Since our data is correlated multimodal multivariate data, we look at generating multivariate correlated dataset. The synthetic dataset generation is mainly performed for one of the two reasons, either to design a new data generation mechanism or to measure efficiency/validation for real life – scenarios.

The chapter presents a synthetic data generation technique to generate the distribution which is like the distribution observed from the real memory data collected from Raspberry Pi devices. From the analysis performed, the memory features are dynamic in nature, so we reproduce the distribution synthetically to verify and validate the modal distributions that our algorithm detects and compare the statistical data derived from the ground truth (values used during data generation). So, we look at the techniques used to generate multivariate correlated data. One of the earliest work was performed by [149] is based on generating uni-variate normal samples with defined mean and variance. Author uses a technique related to Box – Muller method to generate normal data. An extension of this is performed in paper [150] to generate multivariate normal samples using variance – covariance matrix and mean vector of the dataset, which can be applied to Monte – Carlo simulations. To generate the normal distribution, first the vectors are generated with zero mean and unit variance which are then multiplied by a matrix containing the vectors by Cholesky factor (U) of the covariance matrix. The Cholesky factor is the upper triangular matrix (where, U^T. U = Σ, covariance). Another early approach proposed by authors of paper [151] which uses a two stage approach for multivariate Random Number Generation. Stage one generates the samples individually for number of features in the dataset i.e., draws sample for each feature like drawing random samples for a univariate dataset from individual marginal probability distribution. Since the authors draw samples for one variable at a time, there is no correlation between generated variables. The second stage uses the heuristic combinatorial optimization approach to reorganize the samples with respect to each univariate feature to achieve desired correlation amongst features.

Generating random normal samples from the determined covariance matrix uses Cholesky Decomposition of covariance matrix which is the product of the lower triangular matrix and its conjugate transpose. The Cholesky Decomposition can only be used with the covariance

matrix which is positive definite. The other method of defining population covariance to define data distribution i.e., clusters or outliers to generate the data. Two other approaches were proposed by authors of [152] to generate a correlated dataset in place of Cholesky Decomposition by using covariance. The first technique, 'randnm' involves the data generation using complete covariance matrix by using three steps:

1. The first step uses Singular Value Decomposition (SVD) of covariance matrix Σ:

   $\Sigma = VDV^T$

   V – is the matrix of eigen vectors

   D – is the eigen value, Diagonal matrix
2. Using QR decomposition, generates a random valued matrix where Q is the orthogonal matrix and R is upper triangular matrix.
3. Resulting matrix is obtained using the equation $(N-1)^{1/2}QSV^T$ where N is the number of rows derived from step2 and S is the elemental square root of the Diagonal matrix D, such that $D = S^2$.

Whereas the second method uses the eigen values of the covariance matrix. A new algorithm called ADICOV (Approximation of a DIstribution for a given COVariance) was presented in paper [153] to approximate a dataset by using constricted covariance matrix from the original dataset. The ADICOV is an extended version of randnm technique to simulate the features. The first step generates the Eigen Decomposition (ED) of the covariance matrix as done in step one above while using SVD. The 'V' matrix in the ED is the eigen vectors of covariance matrix and 'D' is the diagonal matrix. Then Polar Decomposition is performed by performing SVD and column and row wise vectors are preserved. Then generate the data vectors by using these projections. This is a simplified version of ADICOV, whereas the original ADICOV algorithm uses a lot of projections. The paper [154] proposes a new approach of generating multivariate data as the previous methods namely, 'randmn', ADICOV and CD requires the covariance to be semi – definite matrix and since papers [152-154] use the data generation for chemometric applications, they do not require the data to have positive definite covariance matrix. Hence the author in paper [154] performs various simulations to compare the three algorithms to generate 100 replications of datasets using n * 100 dimension where n = (10, 100, 1000) using three criteria:

1. Computation time

2. SSE (Sum of Square Error) between covariance and one derived
3. SSE in data

To overcome the problem of Σ being semi – definite positive, the author proposes an algorithm *SimuleMV* where in the beginning the correlation level L is defined as 0 and covariance of new data is defined as random covariance matrix and number of observations are estimated by the generated model. They define the model (λ) using following equation:

$$\lambda = x_1 \cdot (x_2 - L) \cdot \left(\frac{\log(M)}{\log(x_3)}\right)^{x_4 e^{(-x_5 L)}}$$

The number of observations (N), number of dimensions in generated data(Y) and number of variables (M) are specified by the user. The algorithm performs ADICOV technique twice, initially its used to build a dataset (Z), where the correlation level (L) specified by the user influences the number of observations in the generated dataset. The second run of ADICOV uses the covariance of Z, number of observations specified by user (in the second ADICOV run) to generate Y.

To solve the error (which might be generated during initial ADICOV run), the model generated has its parameters encoded in the code, uses the common random generator to get the matrix $X_{\lambda,M}$ by using a random matrix as covariance for initial ADICOV run. Which then yields the desired correlation level and uses the yielded covariance for the second ADICOV run to get the data.

The techniques described above are application specific, these are built based on the requirements of the dataset collected/observed by the user or data scientist. Most of the algorithms designed are based on the real data distribution or features as described by the user. The synthetic data algorithm designed for our work is based on the data and distribution observed from the real data collected from Raspberry Pi devices. The major reason of using the synthetic data is to validate the inference generated by our algorithm. The next section provides the detailed description of the dataset with the characteristics observed in the original dataset and the validation criteria of synthetic data characteristics derived from algorithm and synthetic characteristics using which data was built.

Hence while generating multimodal synthetic data, we generate one mode at a time for all features. Since the data is multivariate and multimodal, we build a generator that generates one mode per feature for multivariate configuration as each multivariate modal combination would have unique statistical make – up. Each multivariate dataset generated with 'n' features, consists of a mean vector with 'n' means i.e., one mean per feature and respective using covariance matrix with 'n x n' dimensions. The generator is built using python so, the next section shows the algorithm used to build the generator. We have built two versions of the algorithm; the first version is an automated version of the generator which automatically decides number of modes per feature, covariance etc to build the dataset whereas the second version takes in the data (mean, number of modes per feature, covariance matrix) by user so they can generate data tailored to the statistical requirements of their need instead of randomly sampling mean vector and covariance matrix. The next section provides the constraints and requirements to be followed by the algorithm to generate data with appropriate properties.

5.3 Data Generation Requirements and Algorithm:

Every dataset exhibits a few properties, so while generating a synthetic dataset generator should be tailored to the requirement. The following are the requirements that we need the generator to follow:

- **Number of modes** – As observed, not more than 3 modes are observed at most in a multimodal feature. So, we randomly generate number of modes per feature in the generator. When generating multimodal configuration, if 'i' is the number of modes for a feature, then 'i' is the number of means generated per feature.
- **Covariance** – A covariance matrix is a symmetric semi definite matrix so; generator should generate a symmetric semi definite matrix.
- **Mean** – Mean vector is generated using the range provided so means can be randomly generated using random range function using random library in python.
- Number of samples – We have pre – determined the number of samples per dataset is 1000 samples. But we also must generate how many samples out of the overall 1000 samples will be assigned per mode.

- **Modal Combinations** – Number of modal combinations depend upon the number of modes generated per feature. So, we select a number randomly between 2 – 6 to use when modal combinations are more than 6. But when the number of modal combinations is less than 6 (let's say 'g' is the number of modal combinations where g < 6) we select a number between 1 – g. For example, if the generator selects 2 combinations to use, we use any two modal combinations from total number of combinations generated.

These are the few things that we must keep in mind while building the generator. Even though two versions of the generator are built (automated and manual), there are a few things that a generator needs from the user like number of devices the data is to be generated for and number of features these device datasets should have. So, we can build 'n' datasets in one go.

This algorithm is used to generate multimodal data. As the algorithm mentioned above in the literature account for various types of data i.e., numerical, categorical, non – categorical etc but does not mention the generation of numerical multimodally distributed correlated data. There are multimodal datasets available on a few open-source platforms, but these are images, video or audio used for image/video or audio processing respectively.

5.3.1 Algorithm:

Generate random number based synthetic datasets for one device, can be extended to 'n' devices for 'n' features.

1. *Input* – Number of devices (Datasets)
2. *Input* – Number of features per device (features per dataset)
3. Generate number of modes per feature – randint (1, 4) [no. of modes should be between 1 – 4 as observed in original dataset]
4. Generate mean values per feature:
    - For → Number of modes per feature
        - Generate mean values using randint (a, b)
    - Generate number of modal combinations – mcombos (described in chapter 2)
        - No. of combinations in dataset = len(mcombos)

- Select between 2 – 6 combinations to fill the data in. (The number 2 – 6 is chosen depending on the observations made using original datasets)
- Use random generator to generate covariances for each combination – symmetric semi definite.
- Generate number of samples per combination between 1 – 1000, where total number of samples per dataset should not exceed 1000 samples (For example, if number of modal combinations per dataset is 3; 1000 samples can be divided for each modal combination in any manner such as 500, 200, 300 or 211, 589, 200 etc, which is decided by the algorithm)
- Generate data for each of these selected combinations using numpy function in python – np.random.multivariate_normal(mean, covariance, no. of samples)

- Combine the data generated (via modal combinations method) for that device
- The resulting data is multimodal data (there can be a case where the data is not multimodal, this is purely random and decided by the algorithm.

5. Using steps 3 and 4 the next dataset is generated.
6. End

These steps are used to generate a correlated dataset which is random in nature and validation of the datasets could not be manipulated. The protocol described here assures that the data generated is randomised and built based on the datasets observed in real life scenarios. The key thing here is to generate the modes that are correlated to the modes observed in other features. This will represent various working scenarios of the device which are responsible in generating multimodal distribution for the key features. The algorithm can be extended to generate 'n' correlated features. The requirements mentioned above are tailored to the distribution observed in the real features of real devices. Mainly two types of observations were made in the modes of the features i.e., when the distance between modes of one feature is large and when the distance between modes is smaller. The next section shows the validation parameters generated for the synthetic data when subjected to the classification algorithm should generate similar statistical parameters. Hence, we generate various criteria to perform validation on the datasets.

5.4 Validation Parameters for Synthetic Data

Validation of synthetic datasets is necessary for a key reason which to compare the statistical configurations for when the dataset is generated, and statistical configurations yielded when the data is subjected to the classification algorithm. Following are the parameters to cross – verify i.e., parameters used during generation and parameters inferred using algorithm:

1. Number of Modes
2. Mean Vector for each modal combination
3. Number of Samples per modal combination
4. Covariance per modal combination

We only use one dataset now for parameter comparison for clarity. These will be discussed in depth in the Result section below. We also work out a few case scenarios where we use the manual version of the generator. The next section describes the methodology and reasons of using a few case scenarios to understand all the aspects of the generator and see how the data is perceived by the classification model described in chapter 2 for MMVGD (Multimodal Multivariate Gaussian Distribution).

5.5 Case Scenarios for Synthetic Data generation for classification model validation (MMVGD)

A few use cases were developed to understand how the generated data is perceived by the algorithm. These case scenarios are also used to check if the data is perceived correctly by the algorithm and if not, what are the reasons behind it. Case scenarios used are as followed:

1. Large distance between modal peaks
2. Small distance between modal peaks
3. Modal Peaks overlapping – Does it perceive as one?
4. Less number of samples per mode – perceived or not?

These are the four case scenarios we use to validate the classification algorithm if it perceives the data correctly. When the algorithm randomly generates the dataset, it can be any one of these scenarios since the data is generated randomly. The next section is the results of

synthetic data generation and validation results side by side for each case scenario by performing extensive tests.

5.6 RESULTS AND VALIDATION

The first result is generating dataset using the automated model where we only define number of datasets to generate and number of features they should have. The synthetic generator first generates a dataset where user has defined that a dataset that should have four features. The automated model chooses the number of modes per feature. The table below shows the comparison between the generated dataset and how the classification model perceives the data. We only compare one dataset at a time for clear understanding. The generated dataset has 4 features [F1, F2, F3, F4] where each feature has [4, 1, 3, 2] number of modes respectively. This makes total number of modal combinations for dataset as 24, out of which generator selected any 6 combinations at random (out of 24) to have data in. The table below will highlight how many combinations are interpreted.

	Generated Dataset (Ground Truth)	**Perceived Dataset (Derived Values)**
Number of Modes Data is in	6	5
Mean Vector for each modal combination and Number of Samples for each modal combo	(206621, 277580, 370231, 107402) – 257 (43997, 277580, 370231, 107402) – 422 (206621, 277580, 58988, 107402) – 127 (260664, 277580, 58988, 79289) – 3 (260664, 277580, 337512, 107402) – 180 (286698, 277580, 337512, 107402) – 11	(206521, 277506, 370121, 107542) – 257 (43863, 277675, 370150, 107322) – 422 (206366, 277899, 58859, 107604) – 127 (259971, 275311, 58468, 79527) – 3 (262443, 277615, 337494, 107483) – 191
Covariance of Modal combination with 257 samples	[[4538581 -2346723 488001 -1245348] [-2346723 2660749 678201 538020] [488001 678201 3760013 -1518600] [-1245348 538020 -1518600 2640767]]	[[4501343 -2371796 635391 -1282429] [-2371796 2495202 367511 746155] [635391 367511 3347347 -1232959] [-1282429 746155 -1232959 2414372]]
Covariance of Modal combination with 422 samples	[[4177166 739475 -1070975 669649] [739475 4111054 -3854039 -1772557] [-1070975 -3854039 3668575 1640956] [669649 -1772557 1640956 1428509]]	[[4.046048e+06 2.918279e+05 -6.524788e+05 8.800836e+05] [2.918279e+05 4.112616e+06 -3.824457e+06 -1.903911e+06] [-6.524788e+05 -3.824457e+06 3.615082e+06 1.756343e+06] [8.800836e+05 -1.903911e+06 1.756343e+06 1.618084e+06]]

Covariance of Modal combination with 127 samples	[[5857662 -3519380 3630152 -2216168] [-3519380 3450649 -1704758 2257909] [3630152 -1704758 3690196 -990558] [-2216168 2257909 -990558 1657585]]	[[6.182831e+06 -3.710140e+06 4.209746e+06 -2.220109e+06] [-3.710140e+06 3.366540e+06 -2.045030e+06 2.152654e+06] [4.209746e+06 -2.045030e+06 4.243277e+06 -1.143601e+06] [-2.220109e+06 2.152654e+06 -1.143601e+06 1.571024e+06]]
Covariance of Modal combination with 3 samples	[[1541881 577570 -361494 104914] [577570 2637264 136654 963520] [-361494 136654 3163721 349784] [104914 963520 349784 4281032]]	[[1.785456e+06 4.781095e+06 -2.644417e+06 3.071525e+06] [4.781095e+06 1.306518e+07 -7.696110e+06 8.935745e+06] [-2.644417e+06 -7.696110e+06 5.357690e+06 -6.215089e+06] [3.071525e+06 8.935745e+06 -6.215089e+06 7.209731e+06]]
Covariance of Modal combination with 180 samples	[[4265859 -78033 -580996 2423667] [-78033 1718670 1236577 -489885] [-580996 1236577 1844889 -417687] [2423667 -489885 -417687 3464898]]	[[3.893111e+07 -3.418815e+05 -4.578598e+04 2.093069e+06] [-3.418815e+05 1.858329e+06 1.241719e+06 -3.471786e+05] [-4.578598e+04 1.241719e+06 1.827332e+06 -3.510611e+05] [2.093069e+06 -3.471786e+05 -3.510611e+05 2.675851e+06]]
Covariance of Modal combination with 11 samples	[[2268265 1385768 -1117304 -806578] [1385768 2728525 -321932 1264820] [-1117304 -321932 2020109 468824] [-806578 1264820 468824 2181077]]	

Table 10 Statistical configuration of individual Modal Distribution

The table above shows the comparison of statistical distribution using each modal combination while data generation and data perception. We draw inference from comparing using above mentioned criteria. The comparison is drawn using ground truth (generator based) and perception based (algorithm derived):

1. The First Criteria is checking number of modes detected per feature. This can be different than what is provided by the generator as only a few modal combinations are filled with samples.
    - Generator – [F1, F2, F3, F4]: [4, 1, 3, 2]
    - Algorithm – [F1, F2, F3, F4]: [3, 1, 2, 2]
2. Secondly, we compare the number of modal combinations the data is in.
    - Generator – The generator in total has 24 combinations out of which the data is present in 6 modal combinations.
    - Algorithm – The algorithm in total has 12 combinations and intercepts the data in 5 modal combinations.

- The data in 5th and 6th combination from the generator has mean vector [(260664, 277580, 337512, 107402) – 180 samples] and [(286698, 277580, 337512, 107402) – 11 samples] respectively. As observed, these two mean vectors are same except the mean for F1, but they are closer to each other. Hence, these are perceived as one mode by the algorithm so, the 5th combination has in total 191 samples which is the sum of the number of samples in 5th and 6th combination and perceived as one by algorithm.

3. Mean vector comparison:

Sr. No.	Mean Vector, No. of samples – Generator (Ground Truth)	Mean Vector, No. of samples – Algorithm
1	(206621, 277580, 370231, 107402) – 257	(206521, 277506, 370121, 107542) – 257
2	(43997, 277580, 370231, 107402) – 422	(43863, 277675, 370150, 107322) – 422
3	(206621, 277580, 58988, 107402) – 127	(206366, 277899, 58859, 107604) – 127
4	(260664, 277580, 58988, 79289) – 3	(259971, 275311, 58468, 79527) – 3
5	(260664, 277580, 337512, 107402) – 180	(262443, 277615, 337494, 107483) – 191
6	(286698, 277580, 337512, 107402) – 11	

Table 11 Mean Vector for Individual Modal Combination

- As observed from Table 11, the mean vector for each modal combination when compared with the derived mean vector and ground truth (used while data generation) are remarkably close to each other. In theory, the mean derived from 5th modal combination should be in between the mean values in 5th and 6th combination while data generation. Since F2, F3 and F4 have same mean values in both the combinations, the F1 mean is the comparison to observe. The mean value derived for F1 using 5th combination is closer to the 5th combination used by the generator than 6th as it has larger number of samples.
- Since the last two modal combinations are close to each other during data generation, they are perceived as one mode by the proposed algorithm.

4. Covariance Matrix: The covariance derived for each modal combination is shown in table 12 side by side the ground truth (i.e., the covariance given during data generation). The covariance derived is not an exact match, but the difference among covariances is anticipated. The algorithm uses the default version of generating multivariate normal using numpy library in python. The default version uses SVD (Singular Value Decomposition) to generate correlated multivariate dataset.

5.6.1 Large distance between modal peaks:

The manual synthetic data generator is used to generate a dataset where the peaks are at a distance from each other. The code allows us to specify the mean and covariance of the data generation, which gives us the control over data statistics. The importance of the case scenario is to check if the thresholds for modes are detected correctly. Although the thresholds cannot be compared to the ground truth but can be observed and verified through the visualisation of the data. By generating pairplot of the dataset, we can observe the modal thresholds viewing the graphical representation of each feature individually and each feature with respect to the other.

The Table 12 below shows the comparison of the statistical parameters while data generation and data perception.

	Generated Dataset (Ground Truth)	Perceived Dataset (Derived Values)
Number of modes per feature	[F1, F2, F3, F4] = [2, 1, 2, 2]	[F1, F2, F3, F4] = [2, 1, 2, 1]
Number of Modes Data is in	2	2
Mean Vector for each modal combination and Number of Samples for each modal combo	(389232, 114246, 79542, 234468) – 342 (307606, 114246, 102739, 234468) – 658	(389292, 114319, 79562, 234446) – 342 (307587, 114257, 102692, 234504) – 658
Covariance of Modal combination with 342 samples	[[535062 224469 399338 30275] [224469 775518 107744 28023] [399338 107744 730844 -139990] [30275 28023 -139990 195330]]	[[513839 208489 386407 45800] [208489 856954 98671 30278] [386407 98671 739951 -131616] [45800 30278 -131616 188837]]
Covariance of Modal combination with 658 samples	[[324334 -48370 200336 143266] [-48370 540744 303716 -194292] [200336 303716 700129 -406073] [143266 -194292 -406073 535881]]	[[328143 -30667 210354 147304] [-30667 551643 307395 -171623] [210354 307395 689555 -380767] [147304 -171623 -380767 525230]]

Table 12 Statistical configuration of individual Modal Distribution

- Number of Modes: As observed from the table, when the distance between modes is greater, the algorithm easily identifies the number of modes the data is in and does not perceive two modes as one as they are far from each other.

- Mean Vector Comparison:

Sr. No.	Mean Vector, No. of samples – Generator	Mean Vector, No. of samples – Algorithm
1	(389232, 114246, 79542, 234468) – 342	(389292, 114319, 79562, 234446) – 342
2	(307606, 114246, 102739, 234468) – 658	(307587, 114257, 102692, 234504) – 658

Table 13 Mean Vector for Individual Modal Combination

As observed from Table 13, the mean vector derived per feature per modal combination are remarkably close to each other.

- The number of samples per modal combination are identified correctly.
- Covariance Matrix: The comparison of covariance matrix in the table shows difference between derived covariance and ground truth. Variation in the covariance is expected. Like the previous data generation, the default method SVD is used to generate correlated data.
- Modal Thresholds:

  F1 – [337669.570275, 391103.8024]

  F2 – [117501.7611]

  F3 – [87543.291625, 104857.6547]

  F4 – [236851.3704]

  The Figure 38 below shows the pairplots for each feature with respect to the other for all four features. The diagonal plots show the density estimation of each feature. The x – axis and y – axis are sample values. The numbers 0, 1, 2, 3 represent features F1, F2, F3, F4, respectively. Here, by observing the figure we can estimate the thresholds for each feature. The thresholds above show where each mode ends per feature. The figure also shows the correlation amongst features.

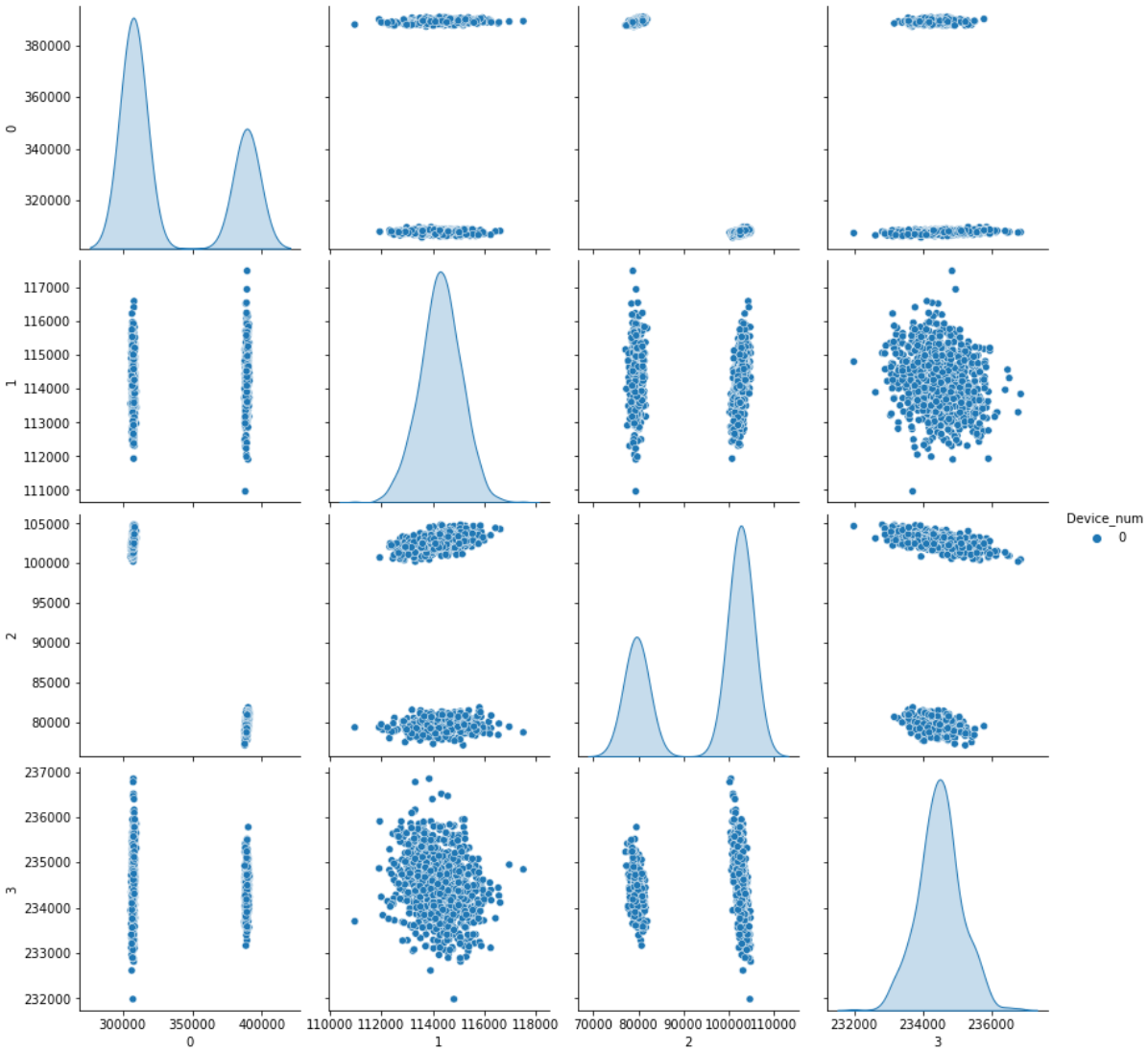


Figure 38 Pairplot of one dataset using all features.

5.6.2 Small Distance between modal peaks:

This case scenario uses smaller distance between modes to check when multiple modes are observed as one and make observation on data perception versus data generation. This case scenario again uses manual synthetic data generator as we can control the number of modes for a dataset and build modes closer to each other. A pair – plot graph of the dataset will also be generated to view the feature density and features with respect to each other.

The table 16 below shows the comparison between the statistical parameters during data generation and data perception.

	Generated Dataset (Ground Truth)	Perceived Dataset (Derived Values)
Number of modes per feature	[F1, F2, F3, F4] = [2, 1, 1, 3]	[F1, F2, F3, F4] = [2, 1, 1, 2]
Number of Modes Data is in	5	3
Mean Vector for each modal combination and Number of Samples for each modal combo	(44957, 44813, 40827, 36190)- 257 (44957, 44813, 40827, 35840)- 189 (30851, 44813, 40827, 36190)- 175 (30851, 44813, 40827, 35840)- 20 (30851, 44813, 40827, 44466)- 359	(44860, 44777, 40841, 36017) – 446 (30841, 44826, 40817, 36206) – 195 (30831, 44848, 40769, 44466) – 359
Covariance of Modal combination with 175 samples	[[272905 9897 -166953 21676] [9897 411585 -75649 356660] [-166953 -75649 666861 -196970] [21676 356660 -196970 454067]]	[[267497 -198 -64454 -6560] [-198 376192 -77044 289682] [-64454 -77044 650499 -160921] [-6560 289682 -160921 389368]]
Covariance of Modal combination with 20 samples	[[574858 250352 341283 5550] [250352 390298 -88022 -59915] [341283 -88022 513677 11590] [5550 -59915 11590 66353]]	
Covariance of Modal combination with 257 samples	[[1253902 1048296 -464702 539658] [1048296 1046637 -219985 426858] [-464702 -219985 462893 -205442] [539658 426858 -205442 249000]]	[[822105 513346 -249797 246629] [513346 688768 -162681 305874] [-249797-162681 305790 -150634] [246629 305874 -150634 304149]]
Covariance of Modal combination with 189 samples	[[312842 -165913 91074 -125733] [-165913 358166 -106338 208233] [91074 -106338 42008 -59052] [-125733 208233 -59052 367874]]	
Covariance of Modal combination with 359 samples	[[369200 156168 -223048 7768] [156168 337876 -540288 46720] [-223048 -540288 917240 -48552] [7768 46720 -48552 42152]]	[[333041 132181 -196816 -2103] [132181 356708 -579999 45968] [-196816 -579999 995459 -50556] [-2103 45968 -50556 41767]]

Table 14 Statistical configuration of individual Modal Distribution

- Number of Modes: As observed from the table, when the distance between modes is small, the algorithm groups the overlapping modes as one and perceive two modes as one. Even though they have small distance in theory but since they have large covariance, they overlap and are visible as one mode on graphs.

- Mean Vector Comparison:

Sr. No.	Mean Vector, No. of samples – Generator	Mean Vector, No. of samples – Algorithm
1	(44957, 44813, 40827, 36190)- 257	(44860, 44777, 40841, 36017) – 446
2	(44957, 44813, 40827, 35840)- 189	
3	(30851, 44813, 40827, 36190)- 175	(30841, 44826, 40817, 36206) – 195
4	(30851, 44813, 40827, 35840)- 20	
5	(30851, 44813, 40827, 44466)- 359	(30831, 44848, 40769, 44466) – 359

Table 15 Mean Vector for Individual Modal Combination

As observed from Table 15, the mean vector derived per feature per modal combination are remarkably close to each other.

- No. of Modes: There are 3 modal combinations detected by the algorithm where the data is present. Modes 1 and 2, 3 and 4 while data generation overlaps and forms one single mode for each pair. This can be concluded by observing the mean vector for the two combinations that overlap where mean values of F1, F2 and F3 are same. Even though the mean values for F4 is different in each combination, since they are close together the modes are observed as one.
- Covariance comparison: Since the modal combinations overlap, the covariance can be quite different than the one we initially used to generate the data.
- Modal Thresholds:

  F1 – [36194.20775875, 47670.25374]

  F2 – [47940.4058]

  F3 – [43715.39616]

  F4 – [39730.246620000005, 45123.47752]

  The modal thresholds can be verified using the pairplot representation of the features. The diagonal graphs show the density of the features, and all the other graphs represent each feature with respect to the other. Figure 39 shows the pairplot of the dataset.

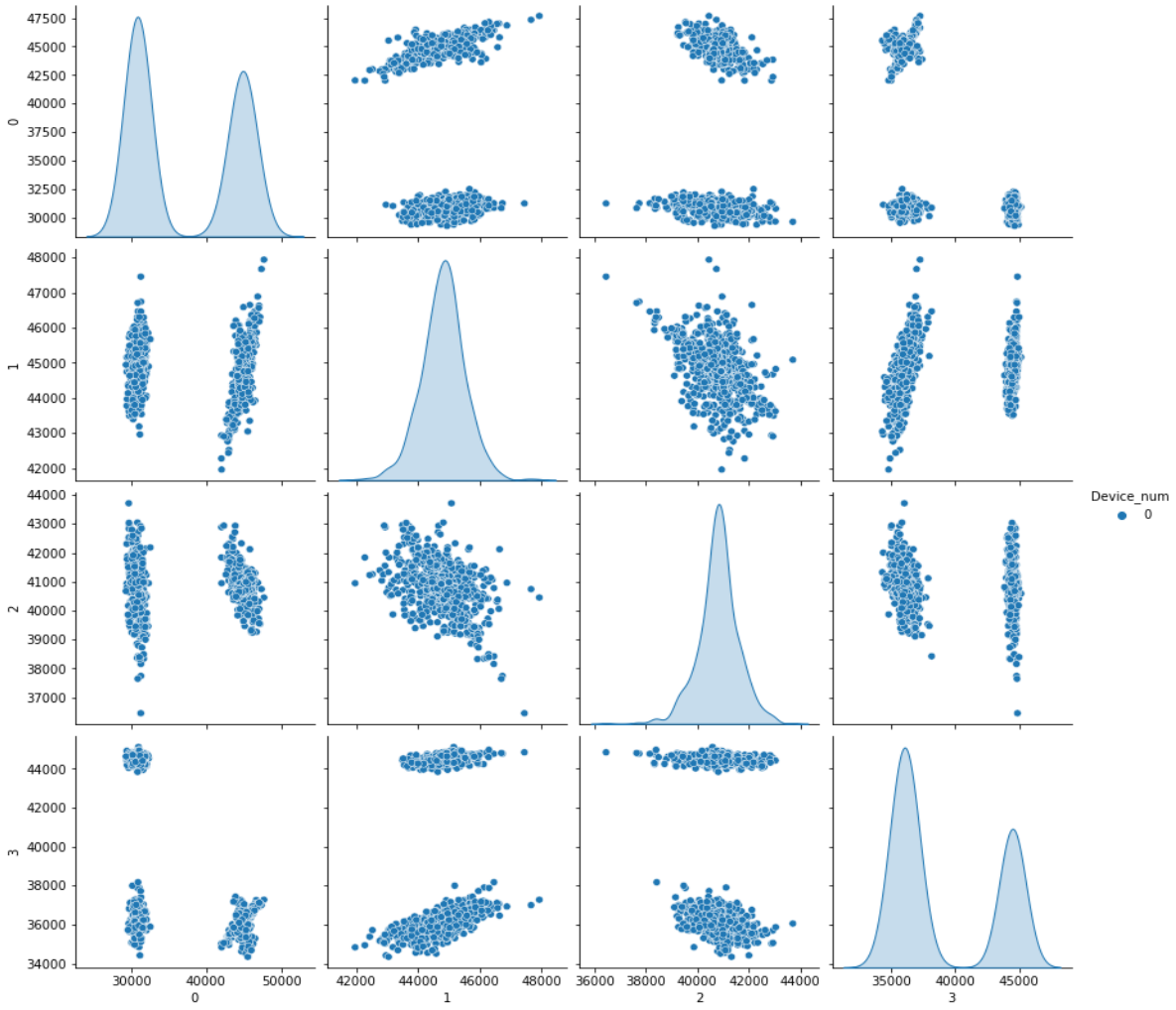


Figure 39 Pairplot of one dataset with closer modes using all features.

The next two case scenarios about modal overlaps and small number of samples have occurred in the 3 previous datasets generated. Hence, we will draw generalised conclusions for each case scenario and discuss the inference drawn for each use case. For each use case, all the aspects like statistical characteristics, number of modes, etc.

5.7 Results on Large Dataset:

The previous section shows the manual generation of the data to control the distances between modal peaks and its visualisations to understand when the algorithm stops differentiating between two modes of a feature. Since, the previous chapters clearly state that the number of datasets we have are limited, this section aims to use large number of devices to check the overall accuracy achieved by MMVGD and the rate of its decline.

This section presents the statistics when the classification technique built in chapter 3 i.e., which addresses the multimodal nature of the features, performs when the data consists of large number of devices.

The data generation algorithm mentioned in section 5.3.1 builds multimodal datasets, which are used to effectively identify devices even while using the unstable features. The previous section provides evidence of validating parameters i.e., if the classification algorithm clearly identifies the parameters set per modal combination during data generation.

Since the validity of the algorithm is proven, meaning it identifies number of modes, modal parameters accurately, there is a question of its performance on large number of devices. Hence, we have generated synthetic dataset for 300 devices using python and numpy library to generate correlated data for each modal combination.

The data is subjected to the proposed classification technique MMVGD, and these results are benchmarked using state – of – the – art classification techniques from sklearn library in python. The synthetic dataset is subjected in the interval of 20 (i.e., 20, 40, 80 up to 300 devices) to each of these classification techniques. Here, our aim is to use the automated data generation algorithm, so the algorithm randomly generates multimodal datasets and avoids the human interference to manipulate data in any way. The reasons to use the automated multimodal data building are to prevent prejudices brought due to human interference i.e., features could have too much overlapping or none at all. Here we generate datasets that exhibit multimodal nature for 300 devices with unknown amount of overlapping or correlations. These are then subjected to MMVGD technique to observe how the classification holds up or how accurately the samples are identified and if the number of datasets increase. Therefore, providing an insight on the data handling capabilities of the classifier. The Figure 40 Classification accuracy for large number of devices.

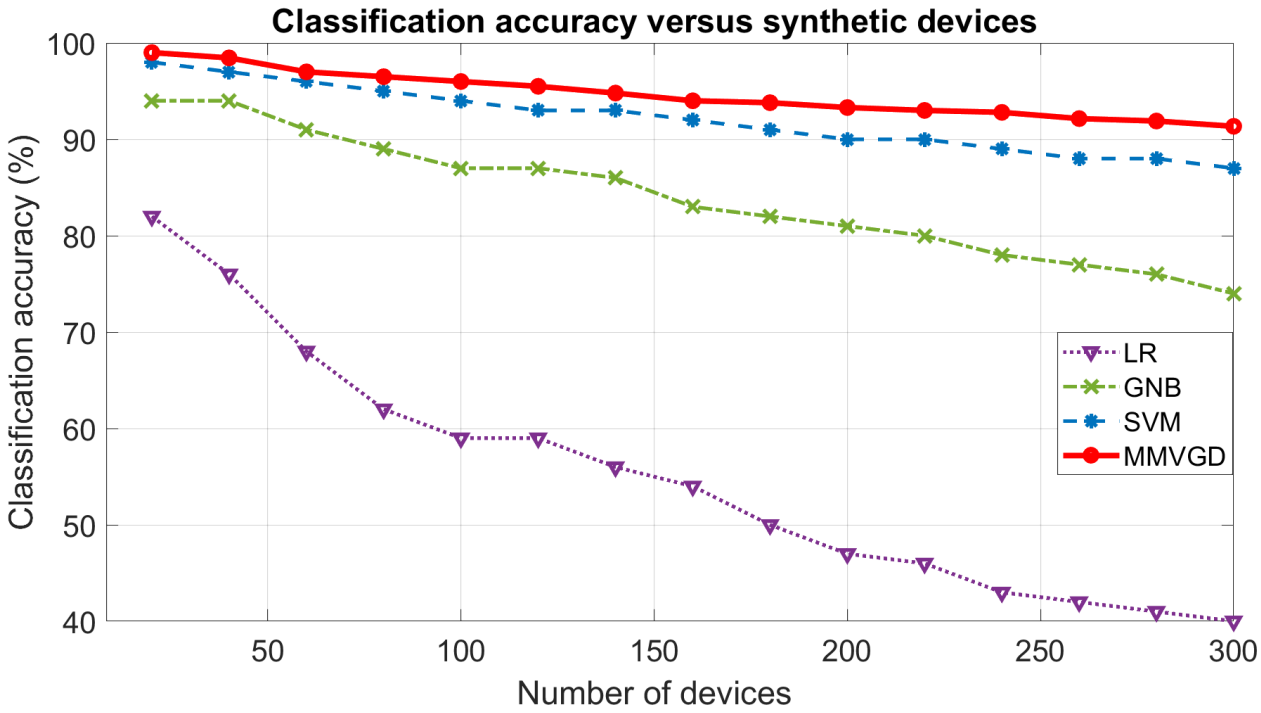


Figure 40 Classification accuracy for large number of devices

As observed from the graph, we can say that MMVGD technique gives the highest accuracy and can handle large number of devices. When the number of devices is increased, we observe a decline in the classification accuracy in all the classification algorithms. Each algorithm shows higher decline in accuracy than MMVGD (proposed classification technique) when number of devices are increased.

Even though the MMVGD shows highest accuracy rate, the classification results while addressing multimodal data also depends on other parameters. These parameters are discussed in the section below.

5.8 Conclusion and Discussion:

The Synthetic data generator is used to generate correlated dataset. The default method used is SVD method using numpy library to generate multivariate data which helps us investigate the authenticity of the classification algorithm and its capability to identify modes and their thresholds correctly, which are then verified by their density graphs. The four case scenarios mentioned in the section above are presented and each of them are used to draw conclusions. The following are the four cases used to draw conclusions:

1. Large distance between modal peaks

   The Large distance between modal peaks makes it easier for the algorithm to detect the modes for each feature. This is due to clear modal boundaries which are detected using peak-trough algorithm. The generator generates modal combinations and chooses at random the number of combinations and which combinations data is present in. When the modal boundaries are clear, the algorithm correctly identifies the number of combinations where the data is present. When statistical characteristics are compared, it is observed that mean vector derived using the algorithm remarkably close to ones used by data generators. When compared the covariance of the data, since the covariance is extremely high in the original datasets, we use the generator to generate higher covariances like observed in the original data. These higher covariances in turn shows higher difference between one used by generator during generation and one derived from generated dataset. The graphs generated are important for validation, this generates another proof of classification properly detecting the modes and their boundaries. The graphs can also show the correlation between features, since the distance between clusters is large, so correlation is not perceived from the graph but its present and the clusters appear to be in a straight line.

2. Small distance between modal peaks

   The Small distance between modal peaks makes it difficult to differentiate between two modal combinations since the covariance remains high, they blend in and are perceived as one mode. The case scenario is used to observe when different modes are perceived as one. While conducting the case scenario it was observed that modal combinations with closer mean vector are perceived as one. We can verify this from table 16 and 17 by checking and comparing the statistical parameters per modal combination side by side used by the generator during data generation and parameters derived from the generated data. The modal thresholds and number of modes per feature can be verified using the pariplot of the dataset. The graphs also show the correlation between the features from pairplots. Since, the distance between modal clusters is lesser than in the previous case scenario, the features are visibly correlated from the graphs.

3. Modal Peaks overlapping – Does it perceive as one?

   The third case scenario slightly coincides with the previous use case where the modes have smaller distance. As concluded from previous case, the close modes are perceived as one. When the overlapping occurs, the resulting mode has different statistical characteristics. The mean vector can be predicted as it will be somewhere between the two mean vectors used during data generation, where the actual mean values of the resulting mode will depend on number of samples each modal combination contains. The covariance and correlation of the data changes when the modes combine.

4. Less number of samples per mode – perceived or not?

   The number of samples are an important measure, to check if they are identified correctly with respect to their modal configurations. This case scenario is a subset of the previous case scenarios discussed. From the validation results, there are two conclusions that can be drawn. First, the modal combinations with a smaller number of samples are identified correctly if they are far from other modal combinations containing data. Secondly, if the modes with smaller number of samples are closer to other combinations, they are perceived as one. IF the number of samples per modal combination, perceived by the algorithm are less than 10, we eliminate the modal data used for training.

The random multivariate generator uses random seed to stabilise the distribution generation. Authors in paper [155] presents the methods to generate multivariate data using Cholesky and SVD method to check the reproducibility of the data. The authors use NUMPY library to sample multivariate datasets. The authors used the data generation methods for Thompson Sampling, but it gives a reasonable insight on reproducibility of data. Even while using random seed function, the distribution might remain same, but the samples produced are different when data is repeatedly generated using the same mean vector and covariance matrix. Hence, the synthetic data generated for validating the algorithm is provided with the thesis for others to validate or reproduce the results. Due to this limitation also observed in our work during synthetic data generation, we provide the 300 device synthetic data generated

using the algorithm proposed in this chapter if others want to reproduce the results obtained from this dataset.

6 CONCLUSION AND FUTURE WORK

6.1 INTRODUCTION

Security is one of the major concerns for the people in the IoT industry. These devices are low – cost and low – resource embedded devices that support large number of applications, support cloud connectivity and variety of sensors to easily integrate with the devices. One of these devices is Raspberry Pi, which is highly versatile in nature, therefore compatible with maximum number of sensors on market with compatibility to most communication protocols available. The literature review performed during the work of this thesis explores the vulnerabilities of Raspberry Pi, previous attacks on these devices and the extent of these attacks originating from IoT devices. It also explores literature on security protocols employed and tested on the devices to prevent these attacks but does not talk about how the resources used to employ these techniques would affect the working of application deployed on the device. Therefore, we look at lightweight techniques that are not computationally expensive to be employed on the resource constrained devices.

The work performed in this thesis provides the basis for a novel technique to identify the low – level hardware by using the foundation of ICMetrics technique which uses internal features of the device to build the identification. The dynamic operational features from the Raspberry Pi devices were collected. The novelty of this work lies in utilizing the dynamic multimodal features to build a novel classification technique to identify the devices. Since the number of available devices are limited, a novel synthetic data generation algorithm is also built to mimic the multimodal distributions observed from real datasets to validate the authenticity of MMVGD classification i.e., if the modal thresholds are recognised correctly, detects number of modes correctly and if the distribution of data detected by the algorithm matches the distribution parameters while generating the data. Secondly, the wavelets are used across the features to analyse the relationship amongst features using modal combinations and build distributions.

This chapter provides an overview of the work presented in this thesis, the limitations of the work performed, and results achieved.

6.2 RESEARCH FINDINGS AND LIMITATIONS

6.2.1 Original Contributions:

- Using Operational features which are multimodal in nature.
- Exploiting the relationship between the modes of the features to build a novel classification technique.
- Using wavelet – based features as an alternative method for non – stationary data for device identification.
- Build a novel synthetic dataset generator which accounts for building multimodal correlated datasets using the data observed in real world scenarios and verify modes identified by proposed classification technique is correct.

The research performed in this thesis is exclusively based on securing the low – cost devices like Raspberry Pi which are highly capable of supporting various IoT applications but have easily exploitable vulnerabilities. Based on the literature review on security for IoT devices, it is observed that the devices that are low in cost have security as a trade – off for lower cost.

The research questions identified during the literature review are presented in Chapter 1 of this thesis. The work performed in the thesis is an attempt to answer the questions presented in Chapter 1. These are answered based on the methodologies presented throughout the thesis and assessed based on the results obtained.

The first challenge was to collect the useful data from the Raspberry Pi devices, which would reflect the working of the devices and this data would reflect any changes in the working of devices. Hence, we collect the memory features which reflect the memory usage in the devices. Usually, the memory / operational features are deemed unusable due to their non – reliable nature and non – stability. But in the low – resource devices these features can be of great use as they reflect change in the internal working of the device.

The features collected are multimodal in nature and cannot be represented by simple statistical characteristics, so these are usually eliminated. Prior to this research the

multimodal features are usually encountered in image processing, biometrics where they have different orientations of the same individual's data. Hence, majority of the research is based on building a classification technique that addresses the multimodal nature of these features. By doing so, we observe extensive improvements in the results using MMVGD (proposed classification method) when benchmarked using the other standard classifiers using sklearn library in python. The results achieved by addressing the multimodality in features are extensively better. Even though the results showed improved accuracy, we recognise that we have limited number of devices, which raises a question about the limitations of the algorithm. So, we explore an alternative usage of features by subjecting the data to DWT and converting raw features to wavelet – based features.

The next chapter explores the wavelets as an alternative technique for device identification. Wavelets are explored as they have a property to enhance changes observed in signals and since no patterns are exhibited when the data is visually observed, we check for non – stationarity in the features to check the randomness depicted from the dataset. The test showed the data to be non – stationary. From the literature, wavelets are used extensively to identify an entity in non – stationary data. The work in this chapter also explores the multimodal nature of the features by converting them to modal combinations and applying wavelets the across features to exploit the changes in their relationship and observe them spatially. The approximate and detail coefficients are separately analysed to observe the best way to exploit their dynamic behaviour to our advantage. The issue again here is the data obtained were from limited number of devices. The wavelets selected are the initial ones from the family so and are used as they are believed to be computationally quicker but since we have limited number of devices, there is no evidence to back up the claim.

But the bigger question here is by addressing the multimodal features can the large number of devices be identified correctly?

To answer this, we look at synthetic data algorithms that can build large number of synthetic devices that exhibit multimodality. To test how the algorithm handles the large number of devices, how each parameter affects the identification, when the algorithm starts showing decline and benchmark these with the state-of-the-art standard classifiers to determine and compare the variation observed in accuracy results from each. From the literature for

synthetic data generation, a lot of different techniques are adopted but these do not specifically mention about adopting multimodal scenarios for numerical data (otherwise open – source platform does have multimodal data in the form of images for image processing applications). Therefore, we build an algorithm to generate 300 devices (with fully automated and manual versions), to test the proposed classification method MMVGD.

Out of all the four classification techniques, MMVGD shows least decline in the results compared to all the other standard classifiers. The chapter also discusses at length the parameters affecting the classification technique. One of the limitations of generating synthetic correlated data using python as observed by authors (based on previous work and work presented here) is even with the specified random seed and same parameters for a dataset, the samples generated are never an exact match. The function uses the SVD method to correlate the data, hence the covariance supplied during generation and the one derived are never an exact match.

Therefore, we also use visualisations to validate the modal parameters i.e., modal thresholds, number of modes and samples per modal combination derived using MMVGD classification technique.

By performing extensive research to address multimodal / dynamic features, and evidence from the synthetic data to use large number of devices to show the reliability of the MMVGD classifier when benchmarked with other classification methods.

6.3 Future Work

The work presented in the thesis forms a basis to identify Raspberry Pi devices but can be expanded to other low – resource embedded devices. At the moment the thesis utilizes two data collection scenarios, this can be expanded to include every possible scenario to understand the changes observed in the internal working of the device. Although, the research has limited number of devices, the algorithm can be updated for ‘n’ devices. Presently the system only utilises the raw internal characteristics of the device for identification, the future work can include stabilising the features, and use the stable features to generate keys for authentication or encryption.

Secondly, from the results above it is observed that non – wavelet features show better accuracy results by small amount. The future work can include building our own wavelets specifically for the purpose of identifying these devices where we can use the data observed from modal combinations of that device to generate its identity.

Lastly, the research can be expanded to simulate the attacks on a few devices to observe how the data deviates from its original relationships observed amongst features and how computationally intensive the attacks can be. If the changes can be detected, it can be used as a foundational step to detect attacks on simple devices without using computationally expensive security layer. The future work should expand on the approach that uses multi – modal data as these are unique and difficult to simulate.

7 REFERENCES:

[1] P. R. Khanna, G. Howells, and P. I. Lazaridis, "Design and Implementation of Low-Cost Real-Time Energy Logger for Industrial and Home Applications," *Wireless Personal Communications,* vol. 119, no. 3, pp. 2657-2674, 2021.

[2] S. Yadav, P. R. Khanna, and G. Howells, "Device Authentication Using Wavelet Based Features," 2022.

[3] P. R. Khanna and G. Howells, "A novel cost-effective Pressure Sensor based Smart Car park system," in *2020 XXXIIIrd General Assembly and Scientific Symposium of the International Union of Radio Science*, 2020: IEEE, pp. 1-4.

[4] P. R. Khanna and G. Howells, "Using Dynamic Operational features to Identify Embedded Devices," in *2021 XXXIVth General Assembly and Scientific Symposium of the International Union of Radio Science (URSI GASS)*, 2021: IEEE, pp. 1-4.

[5] S. Yadav, P. Khanna, and G. Howells, "Robust Device Authentication Using Non-Standard Classification Features," 2021.

[6] S. Yadav, P. R. Khanna, and G. Howells, "Device Identification Using Discrete Wavelet Transform," in *2021 International Conference on Engineering and Emerging Technologies (ICEET)*, 2021: IEEE, pp. 1-6.

[7] "Roundup Of Internet Of Things Forecasts And Market Estimates, 2016." https://www.forbes.com/sites/louiscolumbus/2016/11/27/roundup-of-internet-of-things-forecasts-and-market-estimates-2016/?sh=3dc90cf9292d (accessed 23/09/2022.

[8] "IoT devices to generate 79.4ZB of data in 2025, says IDC | ZDNET." https://www.zdnet.com/article/iot-devices-to-generate-79-4zb-of-data-in-2025-says-idc/ (accessed 23/09/2022.

[9] B. Kleyman. "Smart Things Everywhere: The Connected Future of 2025 • Data Center Frontier." https://datacenterfrontier.com/smart-things-everywhere-the-connected-future-of-2025/ (accessed 24/09/2022.

[10] A. Oracevic, S. Dilek, and S. Ozdemir, "Security in internet of things: A survey," in *2017 International Symposium on Networks, Computers and Communications (ISNCC)*, 2017: IEEE, pp. 1-6.

[11] C. Perera, Y. Qin, J. Estrella, S. Reiff-Marganiec, and A. Vasilakos, "Fog Computing for Sustainable Smart Cities: A Survey," *ACM Computing Surveys,* vol. 50, 03/21 2017, doi: 10.1145/3057266.

[12] H. Poston. "Many off-the-shelf IoT devices require no effort to hack, study shows | E&T Magazine." https://eandt.theiet.org/content/articles/2018/03/many-off-the-shelf-iot-devices-require-no-effort-to-hack-study-shows/ (accessed 24/09/2022.

[13] "Metrarc." http://www.metrarc.com/ (accessed 25/09/2022.

[14] W. Ahmed. "Researcher identifies popular swing VPN android app as ddos botnet." https://www.hackread.com/swing-vpn-android-app-ddos-botnet/ (accessed 10/07/2023.

[15] M. Silverio-Fernández, S. Renukappa, and S. Suresh, "What is a smart device?-a conceptualisation within the paradigm of the internet of things," *Visualization in Engineering,* vol. 6, no. 1, pp. 1-10, 2018.

[16] J. Bugeja, "On privacy and security in smart connected homes," 2021.

[17] A. Mosenia and N. K. Jha, "A comprehensive study of security of internet-of-things," *IEEE Transactions on emerging topics in computing,* vol. 5, no. 4, pp. 586-602, 2016.

[18] S. Duangphasuk, P. Duangphasuk, and C. Thammarat, "Review of internet of things (IoT): security issue and solution," in *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, 2020: IEEE, pp. 559-562.

[19] M. Burhanuddin, A. A.-J. Mohammed, R. Ismail, M. E. Hameed, A. N. Kareem, and H. Basiron, "A review on security challenges and features in wireless sensor networks: IoT perspective," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC),* vol. 10, no. 1-7, pp. 17-21, 2018.

[20] R. Sarma and F. A. Barbhuiya, "Internet of Things: Attacks and Defences," in *2019 7th International Conference on Smart Computing & Communications (ICSCC)*, 2019: IEEE, pp. 1-5.

[21] B. Jana, M. Chakraborty, T. Mandal, and M. Kule, "An Overview on Security Issues in Modern Cryptographic Techniques," in *Proceedings of 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT)*, 2018, pp. 26-27.

[22] Y.-H. Tung, S.-C. Lo, J.-F. Shih, and H.-F. Lin, "An integrated security testing framework for secure software development life cycle," in *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2016: IEEE, pp. 1-4.

[23] I. Ali, S. Sabir, and Z. Ullah, "Internet of things security, device authentication and access control: a review," *arXiv preprint arXiv:1901.07309,* 2019.

[24] A. Mukherjee, "Physical-layer security in the Internet of Things: Sensing and communication confidentiality under resource constraints," *Proceedings of the IEEE,* vol. 103, no. 10, pp. 1747-1761, 2015.

[25] Q. I. Sarhana, "Security attacks and countermeasures for wireless sensor networks: Survey," *International Journal of Current Engineering and Technology,* vol. 3, no. 2, pp. 628-635, 2013.

[26] J. Malan, J. Eager, E. Lale-Demoz, G. Ranghieri, and M. Brady, "Framing the nature and scale of cyber security vulnerabilities within the current consumer internet of things (IoT) landscape," *Centre for Strategy & Evaluation Services LLP,* pp. 1-102, 2020.

[27] I. Kizhvatov, "Physical security of cryptographic algorithm implementations," University of Luxembourg, Luxembourg, Luxembourg, 2011.

[28] D. Genkin, L. Pachmanov, I. Pipman, A. Shamir, and E. Tromer, "Physical key extraction attacks on PCs," *Communications of the ACM,* vol. 59, no. 6, pp. 70-79, 2016.

[29] J. J. Stapleton, *Security without obscurity: A guide to confidentiality, authentication, and integrity*. cRc Press, 2014.

[30] J.-S. Coron, "What is cryptography?," *IEEE security & privacy,* vol. 4, no. 1, pp. 70-73, 2006.

[31] A. Young and M. Yung, "The dark side of “black-box” cryptography or: Should we trust capstone?," in *Annual International Cryptology Conference*, 1996: Springer, pp. 89-103.

[32] N. Saini, N. Pandey, and A. P. Singh, "Enhancement of security using cryptographic techniques," in *2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)(Trends and Future Directions)*, 2015: IEEE, pp. 1-5.

[33] A. Rae and L. Wildman, "A taxonomy of attacks on secure devices," in *Proceedings of the Australia Information Warfare and Security Conference 2003*, 2003: York, pp. 251-264.

[34] G. Mustafa, R. Ashraf, M. A. Mirza, and A. Jamil, "A review of data security and cryptographic techniques in IoT based devices," in *Proceedings of the 2nd International Conference on Future Networks and Distributed Systems*, 2018, pp. 1-9.

[35] M. Tausif, J. Ferzund, S. Jabbar, and R. Shahzadi, "Towards designing efficient lightweight ciphers for internet of things," *KSII Transactions on Internet and Information Systems (TIIS),* vol. 11, no. 8, pp. 4006-4024, 2017.

[36] N. Joshi, J. Sundararajan, K. Wu, B. Yang, and R. Karri, "Tamper proofing by design using generalized involution-based concurrent error detection for involutional Substitution Permutation and Feistel Networks," *IEEE Transactions on Computers,* vol. 55, no. 10, pp. 1230-1239, 2006.

[37] L. Thulasimani and M. Madheswaran, "A single chip design and implementation of aes-128/192/256 encryption algorithms."

[38] D. Hong *et al.*, "HIGHT: A new block cipher suitable for low-resource device," in *International Workshop on Cryptographic Hardware and Embedded Systems*, 2006: Springer, pp. 46-59.

[39] A. Patil, G. Bansod, and N. Pisharoty, "Hybrid lightweight and robust encryption design for security in IoT," *International Journal of Security and Its Applications,* vol. 9, no. 12, pp. 85-98, 2015.

[40] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," in *Proceedings of the 52nd Annual Design Automation Conference*, 2015, pp. 1-6.

[41] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM,* vol. 21, no. 2, pp. 120-126, 1978.

[42] "An Overview of Cryptography." https://www.garykessler.net/library/crypto.html#rsamath (accessed 05/10/2022.

[43] M. El-Haii, M. Chamoun, A. Fadlallah, and A. Serhrouchni, "Analysis of Cryptographic Algorithms on IoT Hardware platforms," in *2018 2nd Cyber Security in Networking Conference (CSNet)*, 24-26 Oct. 2018 2018, pp. 1-5, doi: 10.1109/CSNET.2018.8602942.

[44] E. Fernando, D. Agustin, M. Irsan, D. F. Murad, H. Rohayani, and D. Sujana, "Performance Comparison of Symmetries Encryption Algorithm AES and DES With Raspberry Pi," in *2019 International Conference on Sustainable Information Engineering and Technology (SIET)*, 28-30 Sept. 2019 2019, pp. 353-357, doi: 10.1109/SIET48054.2019.8986122.

[45] S. Surendran, A. Nassef, and B. D. Beheshti, "A survey of cryptographic algorithms for IoT devices," in *2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, 2018: IEEE, pp. 1-8.

[46] T. Meng and W. Buchanan, *Lightweight Cryptographic Algorithms on Resource-Constrained Devices*. 2020.

[47] J. Duchatellier, T. Holmes, K. Suo, and Y. Shi, "An empirical study of thermal attacks on edge platforms," in *Proceedings of the 2021 ACM Southeast Conference*, 2021, pp. 175-179.

[48] C. I. King, "Stress-ng," *URL: http://kernel. ubuntu. com/git/cking/stressng. git/(visited on 28/03/2018),* 2017.

[49] S. Sontowski *et al.*, "Cyber attacks on smart farming infrastructure," in *2020 IEEE 6th International Conference on Collaboration and Internet Computing (CIC)*, 2020: IEEE, pp. 135-143.

[50] Y.-S. Won, J.-Y. Park, D.-G. Han, and S. Bhasin, "Practical Cold boot attack on IoT device-Case study on Raspberry Pi," in *2020 IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA)*, 2020: IEEE, pp. 1-4.

[51] E. D. Martin, J. Kargaard, and I. Sutherland, "Raspberry Pi malware: an analysis of cyberattacks towards IoT devices," in *2019 10th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, 2019: IEEE, pp. 161-166.

[52] "Cowrie SSH and Telnet Honeypot | Cowrie." https://www.cowrie.org/ (accessed.

[53] M. Antonakakis *et al.*, "Understanding the mirai botnet," in *26th {USENIX} security symposium ({USENIX} Security 17)*, 2017, pp. 1093-1110.

[54] C. Frank, C. Nance, S. Jarocki, and W. E. Pauli, "Protecting IoT from Mirai botnets; IoT device hardening," *Journal of Information Systems Applied Research,* vol. 11, no. 2, p. 33, 2018.

[55] "UNIX_PIMINE.A - Threat Encyclopedia." https://www.trendmicro.com/vinfo/us/threat-encyclopedia/malware/unix_pimine.a (accessed 05/10/2022.

[56] Y. N. Soe, Y. Feng, P. I. Santosa, R. Hartanto, and K. Sakurai, "Towards a lightweight detection system for cyber attacks in the IoT environment using corresponding features," *Electronics,* vol. 9, no. 1, p. 144, 2020.

[57] T. Zitta, M. Neruda, and L. Vojtech, "The security of RFID readers with IDS/IPS solution using Raspberry Pi," in *2017 18th International Carpathian Control Conference (ICCC)*, 2017: IEEE, pp. 316-320.

[58] Y. Fu, Z. Yan, J. Cao, O. Koné, and X. Cao, "An automata based intrusion detection method for internet of things," *Mobile Information Systems,* vol. 2017, 2017.

[59] A. K. Kyaw, Y. Chen, and J. Joseph, "Pi-IDS: evaluation of open-source intrusion detection systems on Raspberry Pi 2," in *2015 Second International Conference on Information Security and Cyber Forensics (InfoSec)*, 2015: IEEE, pp. 165-170.

[60] A. M. da Silva Cardoso, R. F. Lopes, A. S. Teles, and F. B. V. Magalhães, "Real-time DDoS detection based on complex event processing for IoT," in *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 2018: IEEE, pp. 273-274.

[61] T. L. von Sperling, F. L. de Caldas Filho, R. T. de Sousa, L. M. e Martins, and R. L. Rocha, "Tracking intruders in IoT networks by means of DNS traffic analysis," in *2017 Workshop on Communication Networks and Power Systems (WCNPS)*, 2017: IEEE, pp. 1-4.

[62] V. Visoottiviseth, G. Chutaporn, S. Kungvanruttana, and J. Paisarnduangjan, "PITI: Protecting Internet of Things via Intrusion Detection System on Raspberry Pi," in *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, 2020: IEEE, pp. 75-80.

[63] V. Visoottiviseth, P. Sakarin, J. Thongwilai, and T. Choobanjong, "Signature-based and Behavior-based Attack Detection with Machine Learning for Home IoT Devices," in *2020 IEEE REGION 10 CONFERENCE (TENCON)*, 2020: IEEE, pp. 829-834.

[64] A. Babaei and G. Schiele, "Physical Unclonable Functions in the Internet of Things: State of the Art and Open Challenges," *Sensors,* vol. 19, no. 14, p. 3208, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/14/3208.

[65] R. Pappu, B. Recht, J. Taylor, and N. A. Gershenfeld, "Physical One-Way Functions," *Science,* vol. 297, pp. 2026 - 2030, 2002.

[66] A. Al-Meer and S. Al-Kuwari, *Physical Unclonable Functions (PUF) for IoT Devices*. 2022.

[67] R. Maes and I. Verbauwhede, "Physically unclonable functions: A study on the state of the art and future research directions," in *Towards Hardware-Intrinsic Security*: Springer, 2010, pp. 3-37.

[68] K. B. Frikken, M. Blanton, and M. J. Atallah, "Robust authentication using physically unclonable functions," in *International Conference on Information Security*, 2009: Springer, pp. 262-277.

[69] H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh, "Mobile device identification via sensor fingerprinting," *arXiv preprint arXiv:1408.1416,* 2014.

[70] G.-J. Schrijen and V. Van Der Leest, "Comparative analysis of SRAM memories used as PUF primitives," in *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2012: IEEE, pp. 1319-1324.

[71] M. Deutschmann, *Cryptographic applications with physically unclonable functions*. Citeseer, 2010.

[72] R. Maes, A. Van Herrewege, and I. Verbauwhede, "PUFKY: A fully functional PUF-based cryptographic key generator," in *International Workshop on Cryptographic Hardware and Embedded Systems*, 2012: Springer, pp. 302-319.

[73] U. Rührmair *et al.*, "PUF Modeling Attacks on Simulated and Silicon Data," *IEEE Transactions on Information Forensics and Security,* vol. 8, no. 11, pp. 1876-1891, 2013, doi: 10.1109/TIFS.2013.2279798.

[74] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proceedings of the 17th ACM conference on Computer and communications security*, 2010, pp. 237-249.

[75] A. Babaei and G. Schiele, "Spatial reconfigurable physical unclonable functions for the Internet of Things," in *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*, 2017: Springer, pp. 312-321.

[76] Y. Gao *et al.*, "Obfuscated challenge-response: A secure lightweight authentication mechanism for PUF-based pervasive devices," in *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, 2016: IEEE, pp. 1-6.

[77] S. Tahir and I. Rashid, "ICMetric-Based Secure Communication," in *Innovative Solutions for Access Control Management*: IGI Global, 2016, pp. 263-293.

[78] R. Tahir, H. Tahir, and K. McDonald-Maier, "Securing health sensing using integrated circuit metric," *Sensors,* vol. 15, no. 10, pp. 26621-26642, 2015.

[79] Y. Kovalchuk, K. McDonald-Maier, and G. Howells, "Overview of ICmetrics Technology–Security Infrastructure for Autonomous and Intelligent Healthcare System," *International Journal of u-and e-Service, Science and Technology,* vol. 4, pp. 49-60, 2011.

[80] E. Papoutsis, G. Howells, A. Hopkins, and K. McDonald-Maier, "Integrating multi-modal circuit features within an efficient encryption system," in *Third International Symposium on Information Assurance and Security*, 2007: IEEE, pp. 83-88.

[81] H. Tahir, R. Tahir, and K. McDonald-Maier, "On the security of consumer wearable devices in the Internet of Things," *PloS one,* vol. 13, no. 4, p. e0195487, 2018.

[82] Y. Kovalchuk, H. Hu, D. Gu, K. McDonald-Maier, and G. Howells, "ICmetrics for low resource embedded systems," in *2012 Third International Conference on Emerging Security Technologies*, 2012: IEEE, pp. 121-126.

[83] Y. Kovalchuk *et al.*, "Investigation of properties of ICmetrics features," in *2012 Third International Conference on Emerging Security Technologies*, 2012: IEEE, pp. 115-120.

[84] E. Papoutsis, G. Howells, A. Hopkins, and K. McDonald-Maier, "Ensuring secure healthcare communications via icmetric based encryption on unseen devices," in *2009 Symposium on Bio-inspired Learning and Intelligent Systems for Security*, 2009: IEEE, pp. 113-117.

[85] E. Papoutsis, G. Howells, A. Hopkins, and K. McDonald-Maier, "Investigation of sample sizes and correlation in multi-cluster feature distributions for an efficient encryption system," in *2008 NASA/ESA Conference on Adaptive Hardware and Systems*, 2008: IEEE, pp. 409-416.

[86] A. B. Hopkins, K. D. McDonald-Maier, E. Papoutsis, and G. Howells, "Adaptive online profiling hardware for ICmetrics based security," in *2008 NASA/ESA Conference on Adaptive Hardware and Systems*, 2008: IEEE, pp. 498-504.

[87] E. Papoutsis, G. Howells, A. Hopkins, and K. McDonald-Maier, "Integrating feature values for key generation in an icmetric system," in *2009 NASA/ESA Conference on Adaptive Hardware and Systems*, 2009: IEEE, pp. 82-88.

[88] A. B. Hopkins, K. D. McDonald-Maier, E. Papoutsis, and W. G. J. Howells, "Ensuring data integrity via ICmetrics based security infrastructure," in *Second NASA/ESA Conference on Adaptive Hardware and Systems (AHS 2007)*, 2007: IEEE, pp. 75-81.

[89] K. Harmer, G. Howells, W. Sheng, M. Fairhurst, and F. Deravi, "A peak-trough detection algorithm based on momentum," in *2008 Congress on Image and Signal Processing*, 2008, vol. 4: IEEE, pp. 454-458.

[90] A. Kokosy *et al.*, "SYSIASS? an intelligent powered wheelchair," 2013.

[91] A. Jaimes and N. Sebe, "Multimodal human–computer interaction: A survey," *Computer vision and image understanding,* vol. 108, no. 1-2, pp. 116-134, 2007.

[92] L. Wei and H. Hu, "EMG and visual based HMI for hands-free control of an intelligent wheelchair," in *2010 8th World Congress on Intelligent Control and Automation*, 2010: IEEE, pp. 1027-1032.

[93] R. Tahir, H. Tahir, A. Sajjad, and K. McDonald-Maier, "A secure cloud framework for ICMetric based IoT health devices," in *Proceedings of the second international conference on internet of things, data and cloud computing*, 2017, pp. 1-10.

[94] J. Murphy, G. Howells, and K. D. McDonald-Maier, "Multi-factor authentication using accelerometers for the Internet-of-Things," in *2017 Seventh International Conference on Emerging Security Technologies (EST)*, 2017: IEEE, pp. 103-107.

[95] D. Al_Dosary, K. M. A. Alheeti, and S. S. Al-Rawi, "ICMetric Security System for Achieving Embedded Devices Security," in *Journal of Physics: Conference Series*, 2021, vol. 1804, no. 1: IOP Publishing, p. 012102.

[96] K. M. A. Alheeti, F. Khaled Alarfaj, M. Alreshoodi, N. Almusallam, and D. Al Dosary, "A hybrid security system for drones based on ICMetric technology," (in eng), *PLoS One,* vol. 18, no. 3, p. e0282567, 2023, doi: 10.1371/journal.pone.0282567.

[97] K. M. A. Alheeti and K. McDonald-Maier, "An intelligent intrusion detection scheme for self-driving vehicles based on magnetometer sensors," in *2016 International Conference for Students on Applied Engineering (ICSAE)*, 2016: IEEE, pp. 75-78.

[98] K. M. A. Alheeti and K. McDonald-Maier, "An intelligent security system for autonomous cars based on infrared sensors," in *2017 23rd International Conference on Automation and Computing (ICAC)*, 2017: IEEE, pp. 1-5.

[99] R. Tahir and K. McDonald-Maier, "Improving resilience against node capture attacks in wireless sensor networks using icmetrics," in *2012 Third International Conference on Emerging Security Technologies*, 2012: IEEE, pp. 127-130.

[100] J. Murphy, G. Howells, and K. D. McDonald-Maier, "A machine learning method for sensor authentication using hidden markov models," in *2019 Eighth International Conference on Emerging Security Technologies (EST)*, 2019: IEEE, pp. 1-5.

[101] M. Haciosman, B. Ye, and G. Howells, "Protecting and identifiying smartphone apps using ICmetrics," in *2014 Fifth International Conference on Emerging Security Technologies*, 2014: IEEE, pp. 94-98.

[102] C. Rosalie. "NASA Jet Propulsion Laboratory Network Was Hacked Through Raspberry Pi." https://www.businessinsider.com/nasa-jet-propulsion-laboratory-hack-raspberry-pi-2019-6?r=US&IR=T (accessed.

[103] "NASA Office of Inspector General Office of Audits CYBERSECURITY MANAGEMENT AND OVERSIGHT AT THE JET PROPULSION LABORATORY," 2019. [Online]. Available: https://oig.nasa.gov/hotline.html.

[104] "Node-RED." https://nodered.org/ (accessed 05/10/2022.

[105] C. A. Bouman, M. Shapiro, G. Cook, C. B. Atkins, and H. Cheng, "Cluster: An unsupervised algorithm for modeling Gaussian mixtures," ed, 1997.

[106] G. P. Nason, "A little introduction to wavelets," in *IEE Colloquium on Applied Statistical Pattern Recognition (Ref. No. 1999/063)*, 1999: IET, pp. 1/1-1/6.

[107] S. Ltf, T. Ahmet, B. L. Mohammad, and T. Mehmet, "Fundamentals and literature review of wavelet transform in power quality issues," *Journal of Electrical and Electronics Engineering Research,* vol. 5, no. 1, pp. 1-8, 2013.

[108] R. Yan, R. X. Gao, and X. Chen, "Wavelets for fault diagnosis of rotary machines: A review with applications," *Signal processing,* vol. 96, pp. 1-15, 2014.

[109] G. Facchini, L. Bernardini, S. Atek, and P. Gaudenzi, "Use of the wavelet packet transform for pattern recognition in a structural health monitoring application," *Journal of Intelligent Material Systems and Structures,* vol. 26, no. 12, pp. 1513-1529, 2015.

[110] M. Thill, W. Konen, and T. Bäck, "Online adaptable time series anomaly detection with discrete wavelet transforms and multivariate Gaussian distributions," *Archives of Data Sciences (AoDSA), Series A,* 2018.

[111] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE transactions on pattern analysis and machine intelligence,* vol. 11, no. 7, pp. 674-693, 1989.

[112] J. B. Tary, R. H. Herrera, and M. van der Baan, "Analysis of time-varying signals using continuous wavelet and synchrosqueezed transforms," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences,* vol. 376, no. 2126, p. 20170254, 2018.

[113] L. Daudet, P. Guillemain, R. Kronland-Martinet, and B. Torrésani, "Low bit-rate audio coding with hybrid representations," ed: Citeseer, 2000.

[114] S. Dhawan, "A review of image compression and comparison of its algorithms," *International Journal of Electronics & Communication Technology, IJECT,* vol. 2, no. 1, pp. 22-26, 2011.

[115] G. Othman and D. Q. Zeebaree, "The applications of discrete wavelet transform in image processing: A review," *Journal of Soft Computing and Data Mining,* vol. 1, no. 2, pp. 31-43, 2020.

[116] H. İ. Çelik, L. C. Dülger, and M. Topalbekiroğlu, "Development of a machine vision system: real-time fabric defect detection and classification with neural networks," *The Journal of The Textile Institute,* vol. 105, no. 6, pp. 575-585, 2014/06/03 2014, doi: 10.1080/00405000.2013.827393.

[117] C.-S. Cho, B.-M. Chung, and M.-J. Park, "Development of real-time vision-based fabric inspection system," *IEEE Transactions on Industrial Electronics,* vol. 52, no. 4, pp. 1073-1079, 2005.

[118] K. Mak and P. Peng, "Detecting defects in textile fabrics with optimal Gabor filters," *International Journal of Computer Science,* vol. 1, no. 4, pp. 274-282.

[119] K. Mak, P. Peng, and H. Lau, "A real-time computer vision system for detecting defects in textile fabrics," in *2005 IEEE International Conference on Industrial Technology*, 2005: IEEE, pp. 469-474.

[120] M. Khare and M. Jeon, "Towards discrete wavelet transform-based human activity recognition," in *Second International Workshop on Pattern Recognition*, 2017, vol. 10443: SPIE, pp. 35-39.

[121] H. Kutlu and E. Avcı, "A novel method for classifying liver and brain tumors using convolutional neural networks, discrete wavelet transform and long short-term memory networks," *Sensors,* vol. 19, no. 9, p. 1992, 2019.

[122] C. Feudjio, V. D. Noyum, Y. P. Mofendjou, and E. Fokoué, "A Novel Use of Discrete Wavelet Transform Features in the Prediction of Epileptic Seizures from EEG Data," *arXiv preprint arXiv:2102.01647,* 2021.

[123] N. Emanet, "ECG beat classification by using discrete wavelet transform and Random Forest algorithm," in *2009 Fifth International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control*, 2009: IEEE, pp. 1-4.

[124] E. B. Cherraqi, N. Oukrich, S. El Moumni, and A. Maach, "Discrete wavelet transform and classifiers for appliances recognition," in *Proceedings of the Mediterranean Symposium on Smart City applications*, 2017: Springer, pp. 223-232.

[125] Y. Shin and P. Schmidt, "The KPSS stationarity test as a unit root test," *Economics Letters,* vol. 38, no. 4, pp. 387-392, 1992.

[126] "Statsmodels documentation — DevDocs." https://devdocs.io/statsmodels/ (accessed.

[127] A. K. Karaev, O. S. Gorlova, V. V. Ponkratov, M. L. Sedova, N. S. Shmigol, and M. L. Vasyunina, "A Comparative Analysis of the Choice of Mother Wavelet Functions Affecting the Accuracy of Forecasts of Daily Balances in the Treasury Single Account," *Economies,* vol. 10, no. 9, p. 213, 2022.

[128] I. Daubechies, "Wavelets: a tool for time-frequency analysis," in *Sixth Multidimensional Signal Processing Workshop*, 1989: IEEE, p. 98.

[129] J. K. Sharma, Y. Gopal, D. Birla, and M. Lalwani, "An Algorithm for Selecting Compatible Wavelet Function in Electrical Signals to Detect and Localize Disturbances," *International Journal of Applied Engineering Research,* vol. 13, no. 14, pp. 11440-11447, 2018.

[130] H. Kanagaraj and V. Muneeswaran, "Image compression using HAAR discrete wavelet transform," in *2020 5th International Conference on Devices, Circuits and Systems (ICDCS)*, 2020: IEEE, pp. 271-274.

[131] F. K. Dankar and M. Ibrahim, "Fake it till you make it: Guidelines for effective synthetic data generation," *Applied Sciences,* vol. 11, no. 5, p. 2158, 2021.

[132] H. Ping, J. Stoyanovich, and B. Howe, "Datasynthesizer: Privacy-preserving synthetic datasets," in *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, 2017, pp. 1-5.

[133] G. M. Raab, B. Nowok, and C. Dibben, "Guidelines for producing useful synthetic data," *arXiv preprint arXiv:1712.04078,* 2017.

[134] N. Patki, R. Wedge, and K. Veeramachaneni, "The synthetic data vault," in *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2016: IEEE, pp. 399-410.

[135] I. Žežula, "On multivariate Gaussian copulas," *Journal of Statistical Planning and Inference,* vol. 139, no. 11, pp. 3942-3946, 2009.

[136] B. Nowok, G. M. Raab, and C. Dibben, "synthpop: Bespoke Creation of Synthetic Data in R," *Journal of Statistical Software,* vol. 74, no. 11, pp. 1 - 26, 10/28 2016, doi: 10.18637/jss.v074.i11.

[137] J. Eno and C. W. Thompson, "Generating synthetic data to match data mining patterns," *IEEE Internet Computing,* vol. 12, no. 3, pp. 78-82, 2008.

[138] G. Caiola and J. P. Reiter, "Random Forests for Generating Partially Synthetic, Categorical Data," *Trans. Data Priv.,* vol. 3, pp. 27-42, 2010.

[139] G. Albuquerque, T. Lowe, and M. Magnor, "Synthetic generation of high-dimensional datasets," *IEEE transactions on visualization and computer graphics,* vol. 17, no. 12, pp. 2317-2324, 2011.

[140] A. Arasu, R. Kaushik, and J. Li, "Data generation using declarative constraints," in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, 2011, pp. 685-696.

[141] J. P. Reiter and J. Drechsler, "Releasing multiply-imputed synthetic data generated in two stages to protect confidentiality," *Statistica Sinica,* pp. 405-421, 2010.

[142] N. Mohammed, X. Jiang, R. Chen, B. C. Fung, and L. Ohno-Machado, "Privacy-preserving heterogeneous health data sharing," *Journal of the American Medical Informatics Association,* vol. 20, no. 3, pp. 462-469, 2013.

[143] M. Pasinato, C. E. Mello, M.-A. Aufaure, and G. Zimbrao, "Generating synthetic data for context-aware recommender systems," in *2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence*, 2013: IEEE, pp. 563-567.

[144] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "PrivBayes: Private Data Release via Bayesian Networks," 2014.

[145] W. Lu, G. Miklau, and V. Gupta, "Generating private synthetic databases for untrusted system evaluation," in *2014 IEEE 30th International Conference on Data Engineering*, 2014: IEEE, pp. 652-663.

[146] Y. Park and J. Ghosh, "Perturbed Gibbs Samplers for synthetic data release," *arXiv preprint arXiv:1312.5370,* 2013.

[147] Y. Park and J. Ghosh, "PeGS: Perturbed Gibbs Samplers that Generate Privacy-Compliant Synthetic Data," *Trans. Data Priv.,* vol. 7, no. 3, pp. 253-282, 2014.

[148] H. Roy, M. Kantarcioglu, and L. Sweeney, "Practical differentially private modeling of human movement data," in *IFIP annual conference on data and applications security and privacy*, 2016: Springer, pp. 170-178.

[149] D. Pullin, "Generation of normal variates with given sample mean and variance," *Journal of Statistical Computation and Simulation,* vol. 9, no. 4, pp. 303-309, 1979.

[150] R. Cheng, "Generation of multivariate normal samples with given sample mean and covariance matrix," *Journal of Statistical Computation and Simulation,* vol. 21, no. 1, pp. 39-49, 1985.

[151] D. C. Charmpis and P. L. Panteli, "A heuristic approach for the generation of multivariate random samples with specified marginal distributions and correlation matrix," *Computational statistics,* vol. 19, no. 2, pp. 283-300, 2004.

[152] F. Arteaga and A. Ferrer, "How to simulate normal data sets with the desired correlation structure," *Chemometrics and Intelligent Laboratory Systems,* vol. 101, no. 1, pp. 38-42, 2010.

[153] J. Camacho, P. Padilla, J. Díaz-Verdejo, K. Smith, and D. Lovett, "Least-squares approximation of a space distribution for a given covariance and latent sub-space," *Chemometrics and Intelligent Laboratory Systems,* vol. 105, no. 2, pp. 171-180, 2011.

[154] J. Camacho, "On the generation of random multivariate data," *Chemometrics and Intelligent Laboratory Systems,* vol. 160, pp. 40-51, 2017.

[155] D. Kilitcioglu and S. Kadioglu, "Non-Deterministic Behavior of Thompson Sampling with Linear Payoffs and How to Avoid It."