DESIGN AND ANALYSIS OF MACHINE LEARNING APPROACHES IN IOT

Thesis Submitted for the Award of the Degree of

DOCTOR OF PHILOSOPHY

in

Computer science and engineering

By

Omar Farooq

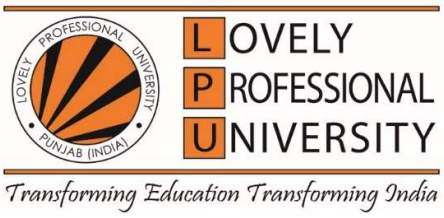
Registration Number: 11720104

Supervised By

Dr. Parminder Singh (16479)

Computer Science and Engineering (Professor)

Lovely Professional University



LOVELY PROFESSIONAL UNIVERSITY

PUNJAB

2023

DECLARATION

I, hereby declared that the presented work in the thesis entitled "Design and Analysis of Machine Learning Approaches in IoT" in fulfilment of degree of **Doctor of Philosophy (Ph. D.)** is outcome of research work carried out by me under the supervision Dr. Parminder Singh, working as Professor, in the department of Computer Science and Engineering of Lovely Professional University, Punjab, India. In keeping with general practice of reporting scientific observations, due acknowledgements have been made whenever work described here has been based on findings of other investigators. This work has not been submitted in part or full to any other University or Institute for the award of any degree.

Omar Farooq

11720104

Computer Science and Engineering

Lovely Professional University,

Punjab, India

CERTIFICATE

This is to certify that the work reported in the Ph. D. thesis entitled "Design and Analysis of Machine Learning Approaches in IoT" submitted in fulfillment of the requirement for the reward of degree of **Doctor of Philosophy (Ph.D.)** in the Computer Science and Engineering, is a research work carried out by Omar Farooq, 11720104, is bonafide record of his original work carried out under my supervision and that no part of thesis has been submitted for any other degree, diploma or equivalent course.

(Signature of Supervisor)

Dr. Parminder Singh

Professor

Computer Science and Engineering

Lovely Professional University,

Punjab, India

ABSTRACT

One of the most exciting emerging concepts nowadays is the Internet of Things (IoT). The rise of Machine Learning (ML), Big Data, IoT and Data Science has presented promising opportunities for research in modern times. However, the persistent deployment of IoT devices, sensors, and other data-gathering technologies is creating substantial strain on the current IoT infrastructure. As a result, managing the massive volume of data produced by these devices has become an increasingly pressing issue. The number of IoT devices is increasing exponentially, and presently we have more than 20000 million objects connected to the network. The amount of data and complexity circulating across networks is also growing exponentially. IoT plays a measure role in this growth rate of IoT data traffic, resulting in a significant rise in data traffic reaching the cloud or data center. The response time of IoT systems is affected by the growth of data traffic as this may not be appropriate for sensitive environments. In this IoT environment where resources are scarce, vast quantities of data are produced by the millions of IoT nodes distributed across devices at the network's edge. Given the constraints of the IoT network's resources, researchers are prioritizing data management as a critical area of focus. The data from diverse devices is both extensive and varied, making it imperative to select the right approach for classification and analysis. By doing so, the data can be optimized at the device, Edge, and Fog levels, which would result in improved network performance in the future. The IoT has also attracted industry-oriented researchers and has become a common platform for most IoT-based applications. The increasing number of IoT devices is pushing the boundaries of the existing IoT architecture. Therefore, alarming for a new or upgraded IoT framework. The IoT devices include sensing, storage, processing, and communication of the data collected from device level nodes and other nodes in the physical world to the local Fog/Edge. Various data management frameworks have been proposed at the IoT-edge

level to efficiently utilize the available resources in the IoT environment. The optimization of the resources in IoT demands the implementation of the machine learning approaches at different levels of the IoT-edge framework. These machine learning approaches must be designed to meet the expectations of the application for which the framework has to be designed. Since in most of the applications, we have challenges due to constrained resources. The resources are constrained due to the miniature and unattended IoT devices for using these applications. These applications experience pervasive limitations due to the extensive integration of IoT.

The foundation for the design and implementation of machine learning algorithms for resource optimization in IoT depends on the analysis and classification of the IoT-data at the IoT first/device level and second/edge level. There have to be different data management frameworks for Internet of things edge architecture for these resource-constrained IoT applications. The offloading of the IoT-data from the IoT devices through the IoT environment to the edge depends on the device's capability for processing, storage, communication, and how much delay the application can permit. The IoT-data sensed from these devices is in large amounts and is continuous. The limitations of storage, power, bandwidth, and memory are also adding to the problem. Therefore, data management in a new IoT framework is the solution that can bypass these limitations. The sensor data is captured in real-time and is implemented in proposed model. Explored various architectures for IoT, along with data types from diverse sensors, and generated insightful graphs based on an experiment conducted with a real-time dataset.

In order to achieve the objectives, Initially, an Internet of Things environment was established by incorporating six IoT devices. This is explained in Section 5.6.1. The classification of the IoT data on the basis of device configuration was the biggest challenge in this work. To achieve this, the dataset was categorized into two subsets. The first category of the dataset comes from the IoT sensors, and the second category is the primary data about the configuration of the IoT devices. This primary data helps us to create a regression model that sets the basis for data classification at the device level. With the help of this regression model, a data management framework was formulated, leveraging two algorithms one at the device level and the other at the edge level within the IoT environment to ensure seamless operations. The proposed model

is capable of deciding whether to push the data to the above level (edge) or to process the data at the first/device stage. As a result, the overall energy of the (the number of alive nodes) of the IoT network increased by 11.9 percent.

This thesis presents the two staged data management frameworks for the IoT with main emphases on machine learning-based modeling and IoT data classification. In this thesis, the focus is on classification of IoT-data at the first/device stage, second stage i.e., Edge/Fog level, and third stage i.e., cloud level using ML techniques and a framework a presented that defines the design/implementation of the machine learning algorithms for the resource-constrained IoT-edge-cloud architecture. The thesis also gives a comparative analysis of the proposed work with the existing approaches.

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my heartfelt gratitude to all those who have assisted me in various capacities throughout the research and report preparation process.

I am thrilled to extend my gratitude to my esteemed supervisor, Dr. Parminder Singh, for his unwavering support and guidance throughout the completion of this research. His exceptional expertise, sound judgment, and strong moral character have been invaluable.

I would like to acknowledge the Department of Computer Science and Engineering at Lovely Professional University for providing me with the necessary resources and financial support to pursue my doctoral degree. I am also thankful to the administrative staff at the Centre for Research Degree Programmes for their invaluable assistance throughout the application process.

I would also like to express my gratitude to my parents, wife, sisters, friends, and colleagues for their cooperation and support. I consider myself incredibly fortunate to have them in my life, and I cannot overstate the role their care and love have played in my accomplishments.

Omar Farooq

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

IOT	Internet of Things
ML	Machine Learning
AI	Artificial Intelligence
DS	Dataset
NB	Naive Bayes
KNN	K-Nearest Neighbor
PCA	Principle Component Analysis
SVM	Support Vector Machine
RF	Random Forest
NN	Neural Network
DL	Deep Learning
SL	Supervised Learning
UL	Unsupervised Learning
RL	Reinforcement Learning
SSL	Semi-Supervised Learning
ANN	Artificial Neural Network
LR	Linear Regression
PR	Polynomial Regression
BN	Bayesian Network
AN	Artificial Network
NLP	Natural Language Processing

LDA	Linear Discriminant Analysis
LSTM	Long Short-Term Memory
MLP	Multilayer Perceptron
SOM	Self-Organizing Map
SVD	Singular Value Decomposition
ICA	Independent Component Analysis
SARSA	State Action Reward State Action
LR	Logistic Regression
QDA	Quadratic Discriminant Analysis
GPS	Global Positioning System
MLADCF	Machine Learning Analytic Based Data Classification Framework
DNS	Domain Name System
NTP	Network time Protocol
VM	Virtual Machine
WWW	World Wide Web
QoS	Quality of Service
API	Application Programming Interface
HRCKNN	Hybrid Resource Constrained K- Nearest Neighbor
PE-WMOT	Power-Efficient Wireless Multimedia of Things
EDAS	Efficient Data Aggregation Scheme
ISFO	Improved Sunflower Optimization Algorithm
PBDMF	Parallelization Based Data Management Framework

CHAPTER 1

INTRODUCTION

In today's world, data is generated by the human being population using smart devices every minute. Researchers estimate that over 3 million new devices/nodes are added to the network connected each month. In every second, around 130 new nodes are added to the environment of Internet of Things. Moreover, within the upcoming four years, it is projected that the global count of interconnected devices will exceed 30 billion [1]. It was also predicted that by the year 2020, there would be nearly twenty to thirty billion electronic devices connected as part of the Internet of Things [2]. IoT is fetching a highly anticipated fuel for big data revolution by interconnecting smart devices in cities, i.e., connecting all the vehicles, appliances, industrial networks, health-care services, etc., to the internet [3]. To improve the quality of living style, the IoT is undoubtedly the solution and critical technology for the current world, having high future capabilities. However, at the same time, it also puts much pressure on the existing method for communication at the IoT network. The IoT has exceeded around 4.40 ZB amounting to 10 percent of the whole "digital universe," which was only 2 percent in 2012-2013. Furthermore, in the next 4 years i.e., by 2025, more than 75 billion IoT devices will be connected to the web. In communication technologies, a solution that tends to improve by itself for the future is a requirement because all the existing

infrastructures will not work for the future after looking at the IoT data's growth rate. All types of data that is generated by the IoT devices are assumed to be uploaded and stored in the cloud viz. IoT/cloud integration. To extract information, cloud-based data analysis is done, and results are generated for IoT data. This approach is not sufficient for most applications having a huge volume of redundancy in the collected IoT-data. As the IoT network is alarming because of the tremendous increase in the IoT data's growth rate, the existing ML algorithms are to be redefined in the resource-constrained network for an efficient and everlasting solution. Classification and Regression are the techniques of machine learning approaches, which fall under Supervised Learning. The ML algorithms can classify the data in a domain that helps predict the future response or helps in the decision support system. As in the IoT network case, which is flooded with the IoT data and lacks the resources at different levels, i.e., device level, Edge/Fog Level. Therefore, the classification technique is the best possible way of classifying the data at each level. The data is filtered at the first level, i.e., the device level. If the data is classified at the device level, then the resource constraints like memory, processing unit, communication unit, and power source will not be a barrier for the volume, velocity, and variety of the IoT data in the future.

The amount of data in an IoT network is so high that the existing framework and presently available algorithms are insufficient for processing the IoT data. Therefore, the present IoT network's improvement is a big Challenge and has high importance for IoT-data Analytics. Mohammad Saeid et al. have surveyed the use-case of different machine learning algorithms used for processing IoT data in an IoT network [4]. The data shows that the use cases like, smart environment, smart traffic, smart health etc., are processed at the edge, edge/cloud, and cloud, respectively. Smart air controlling, smart public places, and smart human activity have historical data and are processed at cloud/edge levels. The type of data for the smart cities and the location of its processing viz Edge or Cloud is different in the IoT Network. Most of the processing is done at the second/edge and the third/cloud level, and most of the researchers have shown the same type of Data characteristics, that further leads us towards the IoT data Analytics at all the levels, i.e., cloud, Edge/Fog, and device level (device-head cluster) [5]. The data captured by the device is forwarded to the above level, as the resources at the device level are limited; therefore, researchers are restricted for data processing at the device

level. In a Smart environment, data collection and the sensing are performed by the device level sensor nodes that deal with the IoT's communication. In a way, like gateways play a crucial role in connecting to other smart devices, the network communication nodes are responsible for Data aggregation, conversion, and transmission. The Typical IoT Node is shown below in Figure 1.1. The recent years have changed computer science's whole scenario by collaborating different subjects into its domain. Moreover, predictions using machine learning have introduced new strength in today's world, which we call Artificial Intelligence (AI), a combination of various fields like Expert Systems (ES), Natural Language Processing (NLP), Neural Network (NN), machine learning perception, etc. All these subfields are hungry for fuel, and that is data. Data is the fuel in today's world. The amount of data generated by devices is growing every day as we increasingly rely on them to connect to the internet. Hence, all these devices connected to each other and to internet form a network called the IoT.

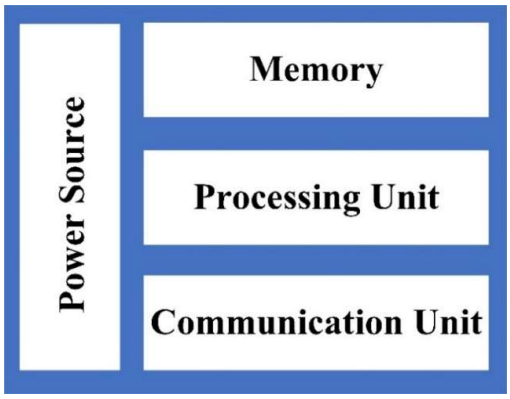


Figure 1.1: Resources of a Typical IoT Node.

The emergence of such an environment has led to a data explosion. The increasing population and the acceleration in the number of devices generating data at a higher rate are creating more significant problems in an IoT environment. The reason behind this is resource constraints. This resource constrained IoT environment has a specific limit, i.e., processing power, storage, battery power, bandwidth and memory. Figure 1.1 shows the resources of a typical IoT Node [3][6]. However, the data generated has nearly no threshold. Therefore, we have two options: either we must minimize the constraints or minimize the data. Managing hardware or a source is not a permanent

solution; therefore, managing data based on classification or feature extraction with machine learning can lead us towards a permanent solution. In this thesis, the data is classified, so that the data in this resource-constrained environment can flow according to the resource's capability.

1.2 IoT Architectures

IoT architecture comprises the following layers:

- The Perception Layer.
- Network Layer.
- Middle Layer.
- Application Layer and
- The Business Layer.

***Perception Layer*:** This layer has the capability of data collection and its responsibility includes collecting and identifying the data. After that, the data is analyzed and accessed by an IoT application to give service. In an IoT network, devices have mini-profiles fields that can be different or similar to the other IoT devices present in the IoT environment. Effectively managing shared data, both within and beyond cluster groups, will be greatly facilitated by this information.

***Networking Layer*:** Network Layer connects all entities and permits the sharing of data between different connected entities of IoT. It can aggregate data of existing IT infrastructures like business systems, transportation systems, power grids, tending systems, ICT systems, etc. Services provided by Layers of Architecture usually develop a heterogeneous network that is more complex. When we talk about the IoT networking layer, researchers must consider energy efficiency in the network, resource availability, QoS requirement, Data and Signal Processing, and most importantly, security and privacy [7].

***Middleware Layer*:** It has two essential tasks, i.e., management of Service, which is also called service management. The second task is storing the information present in the lower layers of the network. Figure 1.2 shows the middleware layer in the center of IoT architecture. This layer has more capabilities like decision-making based on

computational results, retrieving information, processing, and computing the information.

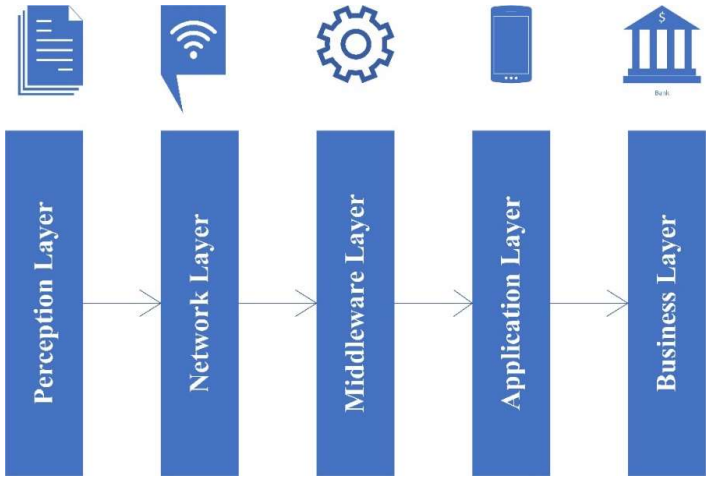


Figure 1.2 : Basic Architecture of IoT.

Application Layer: The application layer utilizes the information presented in the Middleware Layer to manage new applications. Examples of the IoT new applications are smart health, smart glasses, smart independent living, smart transportation, smart home, etc.

Business Layer: IoT applications and management services fall under this layer. It can make or describe Practical reality or business trends of the data with the help of flowcharts, graphs, models, etc. The processed data that it receives from the lower levels of architecture is processed effectively for data analytics, making it easier for the functional executives or managers to do further accurate analysis. The analysis is further required for important decisions or plans [7].

1.3 Functions of IoT Architecture

There are three essential functions of an IoT architecture given below

- Aggregation of data and sensing
- Utilizing data and
- Communicating data

The network communication nodes are accountable for data aggregation, conversion, and transmission. IoT applications utilize the sensed data to offer users a variety of

services. The continuous growth in the number of IoT nodes/devices that process, upload and collect/push the data to the higher level i.e., cloud, results in a significant volume of data being generated [8]. Nowadays IoT is considered a key technology, as the global aim is to enhance the quality of life, as well as promote financial development and employment opportunities. One of the main challenges for IoT to cloud integration is contemporary communication technologies. Internet of things cloud incorporation includes storing and uploading data generated by IoT nodes/devices; therefore, IoT data must be processed by cloud-based data analysis to retrieve helpful information. In most cases, it has been found that the collected data's density is very high; therefore, it becomes necessary to pre-process the data for analysis and storage purposes. Due to the growth of IoT, it becomes necessary to redesign IoT communications [9]. Figure 1.3 shows the essential functions of an IoT system. The basic architecture involves three functions: aggregation of data, sensing, utilizing data, and communicating the same for various services at the application level.

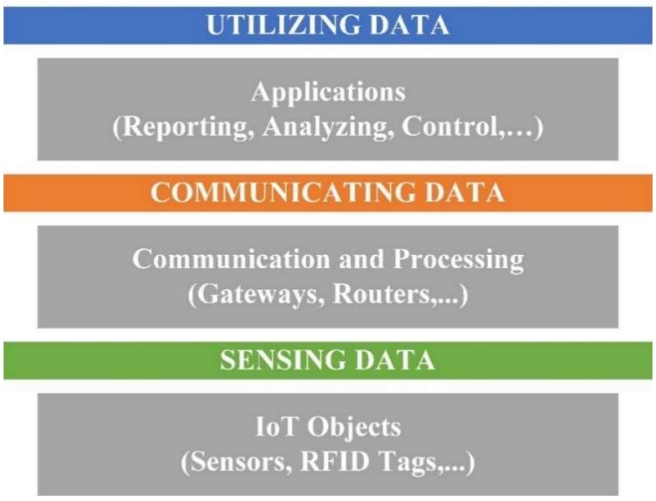


Figure 1.3: Basic Functions of IoT.

Physical IoT layer performs the data collection and sensing. The network communication nodes are responsible for data aggregation and transmission to other nodes or networks. The censored data is then accessed by IoT applications that can access a range of services provided by IoT applications. It includes a communication unit, processing unit, memory, and a power source. However, IoT networks comprise many complex heterogeneous components. The Typical IoT nodes have many resource

constraints; therefore, envisioning real-world IoT systems is intricate without including cloud platforms or powerful devices, including Smart gateways, Fog devices, etc. The limitations of resources should be addressed at both the hardware and software levels. Moreover, there are many constraints due to IoT nodes' characteristics, including the behavior of the network, constraint at the application level, etc. These constraints are primarily applications dependent and commonly found in pervasive applications of IoT [10].

Some of the applications of IoT are

- Remote Monitoring of Soil Parameters
- Environment monitoring
- Monitoring of green energy system.
- Water Monitoring.
- Disaster Monitoring.
- Remote Monitoring Patients etc.

1.3 Data explosion in device/edge Layer of IoT Environment

The data is the fuel for today's modern computer science technology, i.e., machine learning, decision support system, e-commerce, big data, data science, etc. And this fuel is not going to exhaust in the coming years because it is available in a considerable amount. e.g., Twitter generates 12 TB of data per day [11], also the New York Stock Exchange generates 1 TD of data every day. The number of videos uploaded per day on YouTube can be watched continuously for a year. Facebook, which is the most popular social networking website, generates 0.5 petabytes of data every day, including 40 million photos. Every 60 seconds, 98,000 plus tweets tweeted, 695000 status updates on Facebook, 11 million instant messages, 698445 Google searches, and above 168 million emails are being sent. The data generated in 60 seconds is 1820 TB [11].

The highest growth among all types of data is of IoT data. Today, people are so dependent on smart devices that almost all the smart home's electronic devices are connected to the internet. i.e., smart bulb, smart refrigerator, smart door-lock, smart air-condition, etc. The increasing growth of these devices is resulting in the explosion of data. It was predicted that by the year 2020-21, more than 25 billion devices would be connected to the Internet worldwide. More than 3 million devices are connected to the

internet every month. Moreover, in the next 4 years, 30 billion more devices are expected to be a part of the connected devices worldwide. The data generated has 3 characteristics: Variety, Velocity, and Volume. Also popularly known as the 3-V Theory. In simple words, the Variety describes the nature/type of the data. The data in a Smart environment can be image data, i.e., data captured by the camera, or it can be text, audio, temperature reading data, etc. The volume means that the amount of data generated is tremendous, hence we need more storage and processing power. The velocity refers to the processing of this data at a faster rate. As the flow of data is continuous and increasing, the processing of this data is also increasing day by day. The loophole is that data flows in a resource constrained IoT environment.

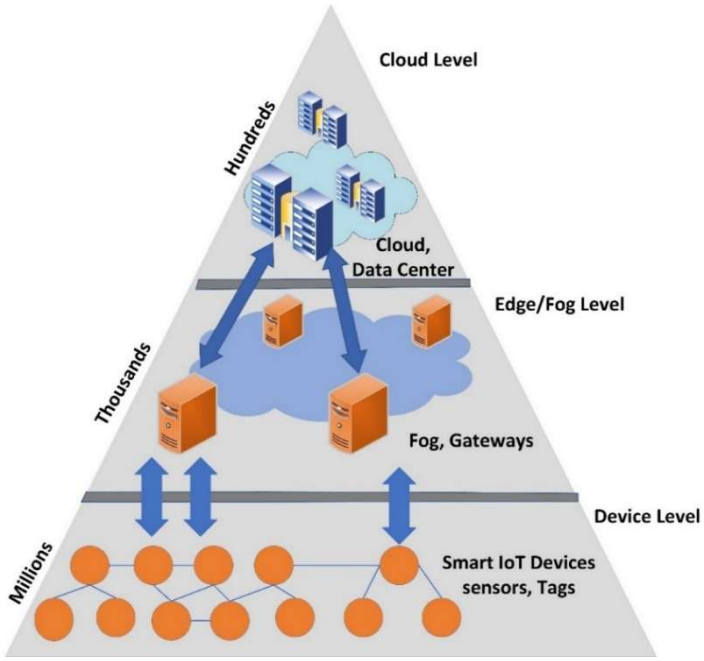


Figure 1.4: Three levels of IoT Environment.

This IoT Environment can be divided into three categories.

- Device Level
- Edge/Fog Level
- Cloud Level

The device-level is the weakest in terms of processing power. As shown in Figure 1.1, the device level has a limited power source, memory, processing power, and

communication unit. The device-level is responsible for capturing or monitoring the data. It records or senses the data and forwards it to the next level. The devices that can be at the device level can be of any type. For example, temperature sensors, image sensors, proximity sensors, accelerometer, pressure sensors, light sensors, smoke/gas sensors, IR sensors, ultrasonic sensors, etc., Figure 1.4 describes the three levels of an IoT environment. The data which is collected from these sensors is pushed forward to the cluster head. As the data is of variety and is in massive amounts, the device cannot process it. It can also process the data to a specific limit to classify the data into the device and edge level. The Edge/Fog level is the middle layer of the IoT Network. The resources in this layer are comparatively of a higher level than the device-level. The availability of the nodes such as Raspberry PI or routers with Cisco's IOx operating system has made the level capable of processing and storing the data virtually through virtual machines on infrastructure nodes. However, the Fog level cannot process a huge analytic task or multiple IoT applications competing for resources. This results in increased processing Latency. The cloud level is the top level of the IoT Network, which contains all the data and has all the resources available for the processing and data analysis to predict and respond to the user's input.

1.4 Challenges and Objectives

In the evolving landscape of the Internet of Things (IoT), the integration of machine learning has become a pivotal factor in shaping technological paradigms. Current technologies in IoT often rely on traditional rule-based systems, offering a deterministic approach to data analysis and decision-making. However, the emergence of machine learning has introduced a transformative shift. Unlike conventional methods, machine learning algorithms in IoT can adapt and learn from vast datasets, fostering a dynamic and predictive environment. Additionally, machine learning augments real-time processing, allowing for quicker and more accurate decision-making in diverse IoT applications. Moreover, traditional IoT technologies may struggle with the sheer volume and complexity of data generated by interconnected devices. Machine learning algorithms excel in handling such big data, extracting valuable insights, and optimizing resource allocation. While existing technologies offer a foundation for IoT functionality, the incorporation of machine learning introduces a paradigm shift,

unlocking unprecedented capabilities that propel IoT systems into a new era of efficiency, adaptability, and intelligent decision-making. The IoT data flows through a resource-constrained environment. The resources such as memory, battery, storage and bandwidth are limited in an IoT environment. The resources are weakest at the device level resulting in more restrictions for processing and storage. The biggest challenge is the pervasiveness of the IoT devices, therefore the upgradation of the resources at the device level becomes difficult. The data that flows through this IoT environment is vast and continuous resulting in the delay in the IoT network. As most devices are either pervasive or small and portable. Therefore, increasing the hardware capacity will increase the size of the IoT device. Hence the data management is the solution for the IoT data limitations. The raw data generated by the IoT devices is boundless, and if this data is well managed in an IoT environment, it can be useful for various IoT Solutions. Most IoT devices are lacking in processing power, battery/power, storage, and bandwidth. Therefore, it becomes difficult for an IoT device to process a huge amount of data. To implement the proposed framework, the work has been divided in to four objectives.

1.5 Objectives

1. Analysis and classification of IoT data at device level and Edge Level.
2. Data management framework for IoT Edge-Cloud architecture for Resource constrained IoT applications.
3. Design and Implementation of Machine Learning algorithms for resource optimization in IoT.
4. Performance Analysis and Comparison of the proposed work with the existing approaches.

1.6 Research Methodology

In IoT networks, efficient resource utilization is paramount. Analyzing data within the network plays a pivotal role in optimizing operations. By strategically minimizing data flow, each tier of the IoT architecture benefits. This process involves scrutinizing information at various levels, identifying patterns, and implementing data reduction techniques. Through meticulous analysis, the system can enhance resource allocation,

boost energy efficiency, and streamline overall performance. This approach ensures that the IoT network operates seamlessly, harnessing its potential to the fullest while minimizing unnecessary data transmission, thus contributing to a more sustainable and effective Internet of Things ecosystem.

IoT data Analysis in an IoT network for optimizing the resource utilization at each level of the IoT network. This can be achieved by minimizing the data flow in the network. Improving data management at each level designing efficient machine learning algorithms for each level. The implementation of the data analysis algorithms and machine learning algorithms will be done by using Python and MATLAB. The data for the analysis will be acquired in Realtime or data sets from the deployed IoT networks for different applications.

1. To achieve the 1st objective, an IoT environment is created to collect real-time data in various scenarios. A wireless sensor network (WSN) is created consisting of several sensors and a gateway as shown in the 5th chapter of this thesis. For this IoT environment, six sensors have been deployed in a particular area of agricultural land. This WSN comprises a gateway responsible for communication with the surrounding and distributed sensors using the Zigbee module. A total number of 6 sensors were deployed, and the sensors could cover the full agricultural field. The data stored in the form of csv file was first analysed and filtered. After pre-processing, the data is classified using the proposed mathematical model.
2. To achieve the 2nd objective, a hybrid resource constrained KNN algorithm and MLADCF framework is presented in a resource constrained environment. The framework is tested with real time data sets captured during first objective. The resources like memory, battery and storage were kept constant during the first scenario. The simulation is performed with Cloudsim toolkit by incorporating auto-scaling libraries.
3. To achieve the 3rd objective, three different algorithms have been developed for device level and Edge/Fog level. The output of the second objective is the input of the third objective. The overall energy, storage and number of alive nodes of the network is calculated and compared in three different scenarios by using MATLAB. Figure 1.5 shows the methodology in the form of a flow chart.

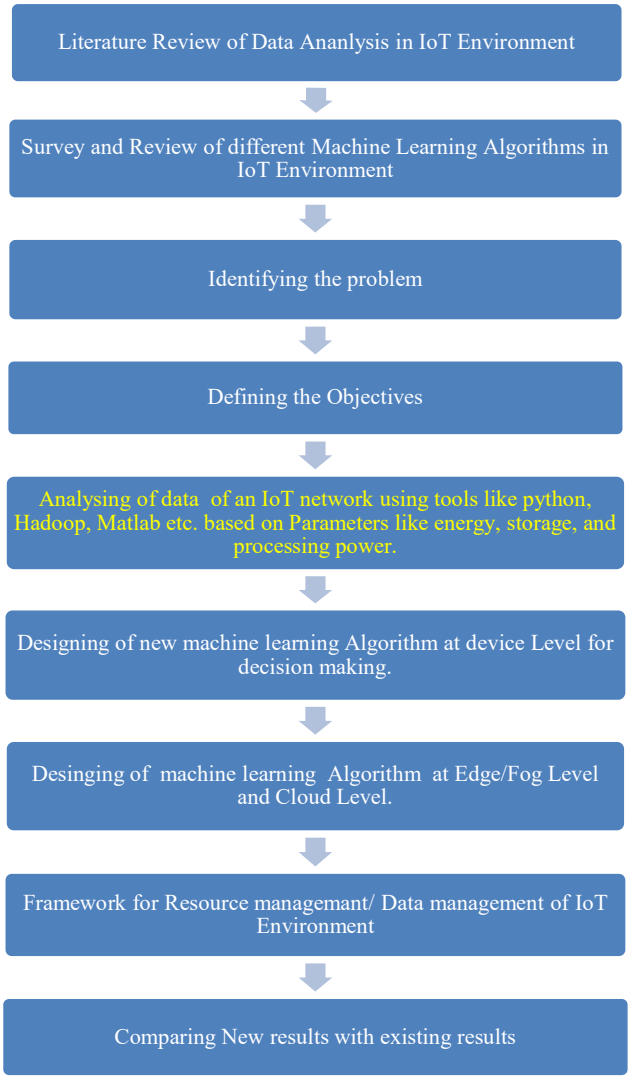


Figure 1.5: Flowchart of the Research Methodology.

4. To achieve the 4th objective, all the results of objective 2 are compared with the existing machine learning algorithms. And the results of the third objectives are compared with the existing approaches and are shown in the form of graphs and tables.

1.7 Thesis Organization

The thesis presents the data classification at different IoT-Edge architecture levels with main emphases machine learning based framework and classification of IoT-Data. The rest of the thesis is organized as shown in Figure 1.6.

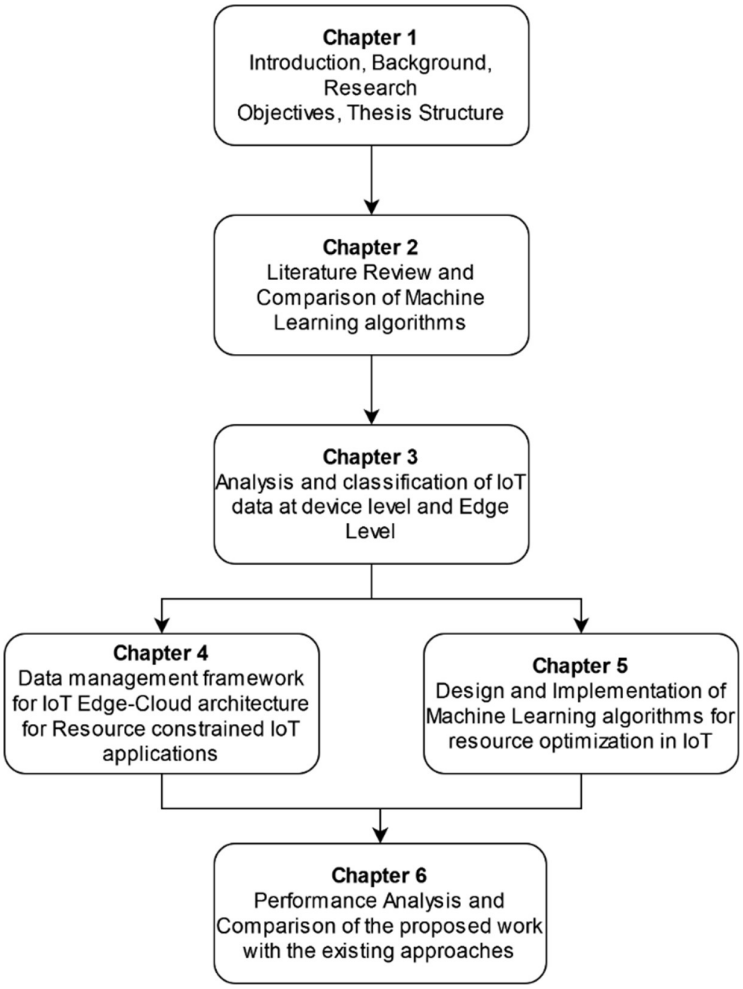


Figure 1.6: Organization of the Thesis.

Chapter 1 gives the introduction, research background, thesis structure and objectives; chapter 2 gives the literature survey of IoT environment, IoT architecture and different machine learning algorithms ; Chapter 3 Summarizes the three levels of an IoT environment and analysis the IoT data at device level and Edge Level; chapter 4 introduces a framework for data management in IoT Edge-Cloud architecture that is designed to support resource-constrained IoT applications; Chapter 5 presents the advancements in machine learning algorithms and their comparison based on advantages and disadvantage and implements the machine learning algorithms for device and edge level; In Chapter 6 a comparative analysis is presented between the proposed work and the existing approaches; Finally, Chapter 7 is concluding the thesis.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Internet of things (IoT) connects a huge number of things to the internet. It comprises complex environments having heterogeneous components. This IoT environment generates enormous data and therefore imposes a demand for storage, processing, and transmission. Since IoT provides many applications using other technologies such as Fog, Edge, and Cloud to help us in our day-to-day life applications, which are not limited to our daily lives, however, they include other more important sectors such as remote patient monitoring, precision agriculture, environmental monitoring, disaster mitigation, and other smart city applications. We expect these applications will increase day by day without any limit. The only limitation which we found is in the resources of IoT. The constraints in the resources of IoT pose many challenges before us at the network, hardware and software levels. Since the applications are increasing, resource management at the different levels of IoT systems becomes necessary. These resources include battery life, size, processing power, storage, and bandwidth. Due to the pervasiveness of some IoT applications, protocols and lightweight algorithms are employed to acquire, process, and store data. The general structure of the IoT environment comprises three stages/levels: the first/device level, the second/Fog/Edge level, and the third/Cloud Level. While most data is processed at the cloud level due to resource availability, Fog Computing technology enables some data to be processed at the Fog level. However, the device level has limited resources due to factors such as

size, pervasiveness, and wearables. Therefore, managing resources and data is crucial for a better IoT environment framework. Encryption and virtualization are added to the algorithms and protocols at the edge level. The development of more specific and lightweight protocols and algorithms is a key challenge in the resource constraints applications of IoT. The classification of the data at the first stage i.e., device level and proper resource allocation within the IoT network will help improve resource management. In recent years, technological advancements and the widespread availability of the internet have enabled us to become increasingly connected with the world around us. Incorporating mobile phones and smart devices into our daily routine has become an essential aspect of our lives, fostering a connection between individuals and every element present in our surroundings. This has led to the emergence of the IoT, which offers the potential to interconnect everyday objects such as appliances, light bulbs, traffic lights, and vehicles. As a result, IoT has become a reality, and the possibility of interconnecting various elements of our environment is now within reach [11-13].

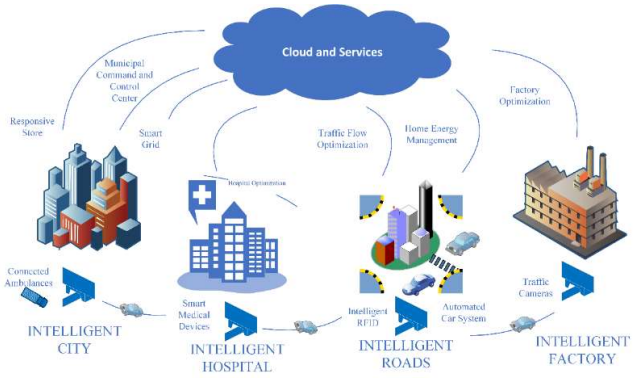


Figure 2.1: Application Scenarios for IoT.

It was projected that in 2021, the number of devices/nodes connected to the internet would exceed 30 billion, posing a challenge to the centralized infrastructures in place today. The widespread deployment of sensors, along with the rise of 4K video transmissions, augmented reality, and other advanced technologies, has led to a surge in internet traffic that is reaching data centers. This trend is only set to continue as IoT and other connected technologies become increasingly prevalent in our lives [14]. Sensor and IoT data is typically sent to cloud data centers for storage and processing,

which can create unacceptable latencies in environments that require real-time decision-making. At Massachusetts Institute of Technology (MIT), research was conducted, during which Kevin Ashton coined the term IoT (Internet of Things) on sensor technologies and Radio Frequency Identification networks (RFID). The concept was that if every object in our environment was equipped with this technology, computers could identify, observe, and comprehend the world [15]. An IoT device is characterized by a small electronic system equipped with a processor, sensors to measure the environment, actuators that allow it to perform certain actions in response to the data received, and communication modules that use network protocols. IoT's example is found in smart homes, in which sensors are installed in different areas of the house, connected to a central system allows optimizing the use of electricity, water, and energy consumption. The value for businesses lies in the insights obtained from the data, which can lead to process automation, resource optimization, and better decision-making, ultimately resulting in greater operational efficiency. Figure 2.1 illustrates the application scenarios for IoT. The size and variety of data circulating on today's networks are increasing exponentially, and IoT contributes significantly to this increase in volume. Cisco estimated that more than 30 billion devices would be connected to the network, implying an increase, a substantial amount of traffic circulating through the networks and reaching data centers in the cloud treatment [16].

2.2 Related Work

Cloud computing is a technology that has promoted IoT systems by offering the storage and computing capacities necessary to process and extract information from the data generated by said systems. The discussion revolves around architectures in which data gets transferred to centralized infrastructures for processing before being returned to the requestor. However, when storing, processing, and retrieving data from the cloud, the latency can be affected by the increase in traffic taking place on the networks, which can be unacceptable in environments that need to respond in Real-time. An autonomous vehicle cannot afford a delay between detecting a possible collision and acting accordingly (slowing down, stopping); industrial safety systems, such as fire alarms and smoke detectors, cannot afford data transmission delays. For this reason, organizations need solutions that allow data flows to be processed in the shortest

possible time to obtain immediate responses. Reducing latency means bringing processing closer to the end devices, the ones that generate and consume data. Bringing services closer to the network's perimeter has been challenging since the internet became a universal service and the widespread demand and consumption of content. Proof of this is content distribution networks, Content Delivery Network or CDN, whose objective is to prevail the native limitations of the internet in terms of QoS sensed by the user, providing services that enhance network performance by maximizing bandwidth enhancing accessibility. All this through a collaborative grouping of nodes in the network located in the vicinity of the clients [17].

2.3 Reference Model

The IoT model is designed as a balance among 7 levels or layers, detailing how each level should function to ensure simplicity, scalability, and compatibility among all sections of an internet of things. In this system, data moves bidirectionally between level 1 and level 7, as illustrated in Figure 2.2.

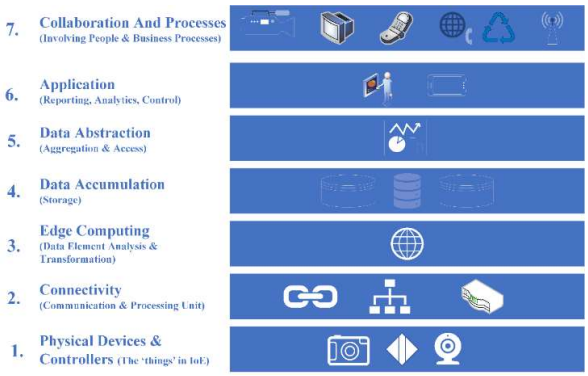


Figure 2.2 Reference Model for IoT.

Level 1: Sensors, actuators, and physical devices are effective in generating, transmitting, and receiving data through a network protocol.

Level 2: Level 2 of the IoT architecture involves the networking equipment that facilitates communication between level 1 devices and higher levels. These devices are typically located at the customer's end.

Level 3: The function of this layer is to conduct an initial analysis of the data flow. Level 3 of the IoT architecture can be regarded as an intermediary layer that resides

between the hardware and the cloud infrastructure. This layer is referred to as Edge or Fog Computing and should ideally be placed as close as possible to the data source, at the network's edge.

Level 4 to level 7: The services of Cloud, infrastructure, and applications are located in the remote network, where data is transformed into useful knowledge and information. The true potential of IoT lies in the combination of Cloud and edge Computing. One potential goal is to reduce the high level of centralization inherent in cloud services, presently possess is necessary to process data flow in real-time. This can be achieved by promoting analysis and knowledge generation closer to the node where data is generated. The Edge (Fog) Computing layer is thus critical to this process [17].

2.4. IoT Protocols

Communication technologies represent the channels through which things can communicate and thus allow heterogeneous devices to create services. For communication between network nodes and peripherals or sensors, lightweight protocols are required, efficient in battery, CPU, and bandwidth. As a result of the bibliographic review, some technologies are identified as candidates to be implemented.

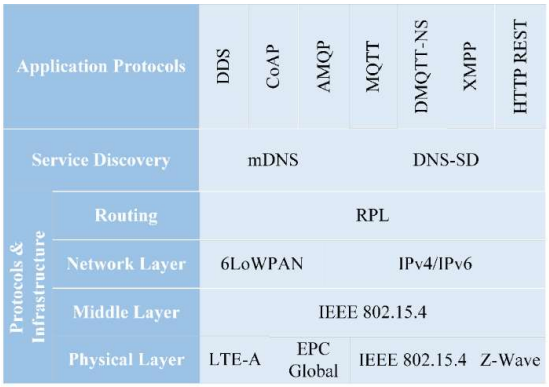


Figure 2.3: Communication Protocols.

The purpose of the standards is to facilitate and simplify the integration between applications, services, and other components of a technological solution. IoT is no exception: organizations such as the W3C, IETF, EPC Global, IEEE, and ETSI, among others, have advocated. Figure 2.3 summarizes the most used protocols based on the TCP / IP stack.

To carry out a diagnosis and evaluation of the protocols most used by IoT, together with their main differences specifically in the application layer of the OSI reference model. This evaluation was motivated by the fact that a unified IoT architecture has not yet been clearly defined. There is currently a lack of consensus regarding the definition of protocols and standards across all aspects of the IoT. According to the requirements survey results, it is possible to evaluate which ones are suitable for this implementation with the previous study of protocols. Computing power represents the brain of things by providing processing and storage capacity. Computing capacity is important in environments where access channels to the cloud are poor due to the lack of data network coverage. Here are some of the most popular and commonly used commercial alternatives for IoT product development [17].

2.5 Construction of the Architecture Prototype

As a representation model, suggest using 3-layer architecture, based on IoT, to manage the supply chain in which there is also a division at the process level. Similarly, mention some other emerging architectures, where the perception, network, and application layers of the 3-layer architecture can be appreciated selected as a model for this technological solution. In the proposed representation model, one of the main characteristics of IoT is evidenced, in which each physical object has a virtual representation through a good network and computing infrastructure that supports it, as shown in Figure 2.4.

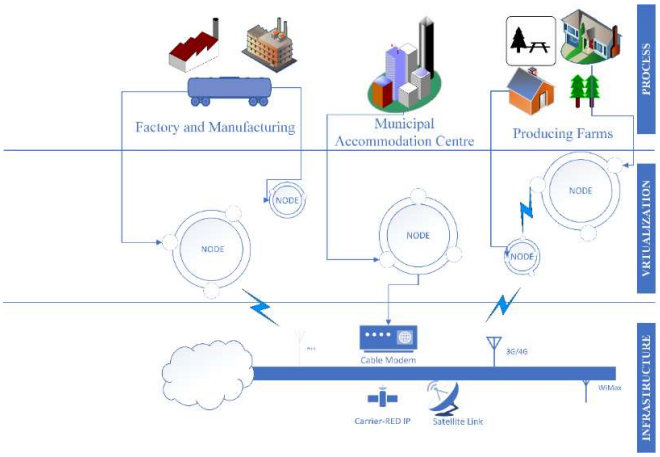


Figure 2.4: Representation Model.

By implementing a set of nodes, the data capture, transmission, processing, and subsequent presentation are carried out according to its context since the processing could be given before the transmission [18]. Next, the topology of the nodes developed according to the guidelines of the proposed architecture and the layer in which they are found is described. Figure 2.4 is used as a reference and follows an IoT EDGE or Internet of Things at the edge approach.

2.6 Node Topologies

The EDGE Gateway nodes require the use of interfaces defined under the IEEE 802.15.4 standard since they define the physical level, mentioning aspects of signaling, coding, and voltages. It also provides access to the medium addressing at the MAC2 and LLC3 layer levels, enabling traffic over IP for the MQTT and REST application protocols used to send data captured by the IoT nodes to the cloud. As SBC (in English, Single Board Computers), the reference Raspberry PI 3 has been selected because it represents the most attractive commercial alternative for processing and storage in embedded devices due to its easy acquisition, commercialization, configuration, and greater compatibility in the market. This reference supports the UART, i2c, and one wire protocols used by the GPS, NFC + ADXL345, and DHT22 modules.

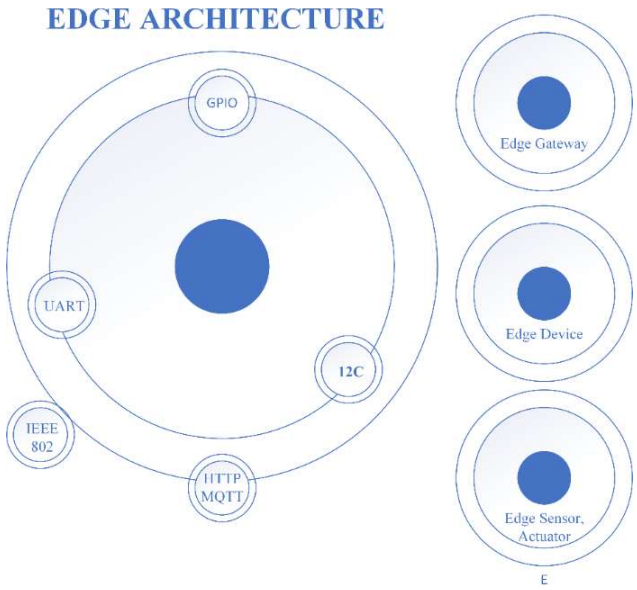


Figure 2.5: Topology of Nodes.

2.6.1. Sensing Layer

In the first layer (embedded systems and sensors), the Raspberry device digitally senses the readings of the variables through a set of sensors installed to capture raw data that will serve as input for the different services provided by the technological solution in the cloud. Following the IoT EDGE approach, the application protocols are configured to connect the device and the AWS, CARTO, and PUBNUB services since node-red supports the REST-full stack used by the device to deliver registers an adjustable frequency in JSON format, which contains the readings of the entire sensor [18]. The device also provides a web page4 to access from a tablet or Smartphone that acts as a base station of a Wi-Fi environment/network for the edge node. The following components act together on the sensing layer:

***Edge Devices*:** It is a general-purpose device that supports operating systems for embedded devices. As for the power source, it is not of great autonomy due to its high mobility. It makes decisions based on the calculations it performs on the input data taken from the sensors; it can execute commands on the actuators.

***Edge Sensor and Actuator*:** It includes devices of special or particular purpose; they do not support the execution of the operating system. They may be Environmental, Humidity and temperature sensor - DHT22, Accelerometer, and Gyroscope - ADXL345, Georeferencing, GPS Module - Adafruit Fona SIM808, Communications, Connection to Wi-Fi networks - Adapter 801.11, Connection to GSM networks - Adafruit Fona SIM808, Authentication and NFC - SL030, etc.

2.6.2. Network Layer

Given the heterogeneity, breadth, and depth of the proposed architecture, some nodes, depending on the availability of cell phone networks, will have the ability to use GSM modules to send and receive data on the internet without depending on other nodes. The gateway or gateway for the architecture model is assumed by any device that allows the option of creating Wi-Fi anchor points and that, through the respective ISP, gives the node an exit to the internet. Figure 2.5 shows the network topology compatible with the present technological solution. This topology uses different network connection mechanisms, from Wi-Fi cell phone networks to satellite links, to reach the services hosted in the cloud [18].

Edge Gateway: Includes devices with processing capacity, storage, memory, and high-capacity power supply, supporting the operating system. It can be any device that allows creating Wi-Fi anchor points.

2.6.3. Cloud Layer

This layer provides the microservices of the DSS, provisioned on Amazon infrastructure, the visualization and generation of indicators in the CARTO platform, and the generation of a dashboard or control panel. The microservices created by these platforms are accessed by the EDGE devices of the sensing layer to send the records stored in their local memories, taking advantage of the storage and processing capacities of these nodes. They are oriented to the location in PUBNUB-eon, connected to Mapbox using a publication and subscription scheme similar to that used by MQTT. The Initial State tool is used experimentally to build a history of records on a timeline.

2.6.4. Edge Computing

"Edge Computing" refers to a computational architecture where processing and analysis competencies are brought closer to the source of data. This approach can reduce latency and improve application response times by avoiding the need to transmit data to remote infrastructures for analysis. As a result, the volume of data sent to the network is reduced, making it an ideal solution for IoT environments. Virtualization and distributed computing are key factors within this decentralized architecture model that must scale horizontally. The term Edge device, in this context, refers to items with limited capabilities that have their own set of resources: CPU, memory, storage, and network. They can be smartphones, smart glasses, smartwatches, tablets, routers, autonomous vehicles, or any IoT device with processing capacity. Thus, Edge Computing refers to how part of the processes that are now carried out in cloud data centers are moved and executed in Edge devices or Edge nodes that can sometimes represent small data centers in the client's vicinity. Researchers have proposed various solutions to optimize the IoT environment. One solution that has been proposed is the use of Software Defined Networking (SDN) networks. However, there is currently no agreed-upon reference architecture for Edge Computing, as shown in Figure 2.6. The

concept of bringing processing to the data source through a collaborative process of nodes is not a new one, with studies and research dating back to the 1990s [19].

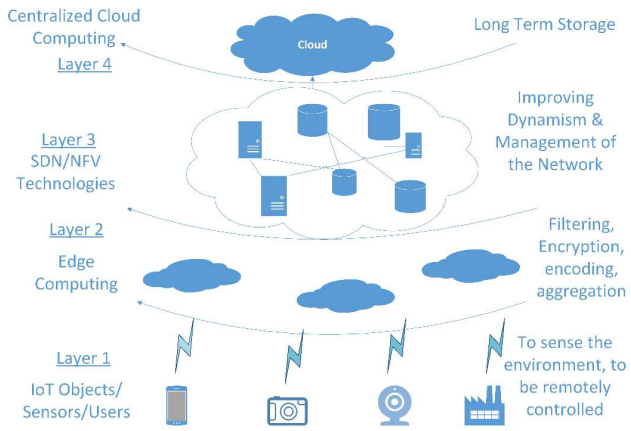


Figure 2.6: Architecture for Edge Computing.

The Cloud of Things concept provides mechanisms to bring data from IoT devices to the cloud. This environment consists of different layers/levels. The data has to pass through different layers i.e., device layer, Fog/Edge layer and finally cloud layer. As the data is increasing day by day therefore the use of machine learning approaches is very important in order to optimize the resources. Seamless intermediate network devices are used to communicate in an IoT environment, known as IoT gateway. The IoT Gateway provides information on this data that flows in both directions, and also it acts as a translator. Hence it contains two different protocols to communicate in day-to-day life, In the domain, a wide array of diverse sensors is encountered, each capable of acquiring distinct types of data. Every sensor has another purpose, and these sensors are enormous in numbers, so the amount of data is tremendously high. Managing a massive amount of data that comes from thousands of sensors is a challenging thing. The sensors such as optical sensor, infrared sensor, gas sensor, gyroscope, level sensor, pressure sensor, proximity sensor, temperature sensor, humidity sensor, accelerometer etc., collect different data types of different volume and variety.

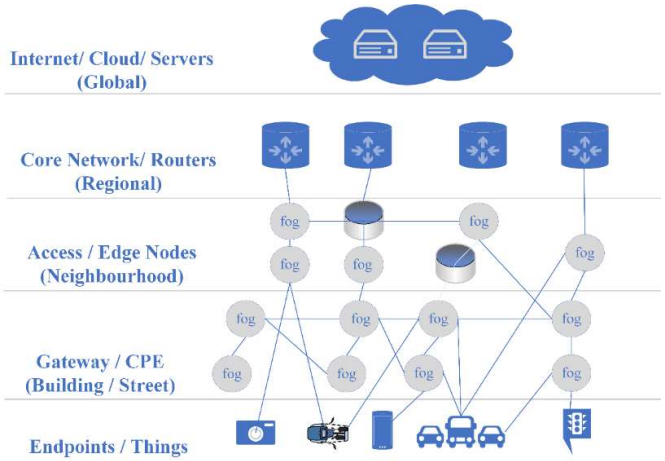


Figure 2.7: Overview of Fog Computing.

Fog computing offers significant benefits in managing the Big Data generated by IoT. Edge analytics, which involves processing data at the second level i.e., edge level of the environment, reduces the high data redundancy in IoT. Pre-processing tasks such as filtering, cleaning data, extraction of variables, comprehension, and reduction techniques of data dimensions at the edge have helped reduce the amount of data sent to the higher level i.e., cloud and subsequently stored [19]. Similarly, techniques to merge several data streams into one or send the data only when anomalies are detected reduce the frequency with which IoT data is generated. Additionally, handling interoperability technical, semantic, and syntactic in Fog nodes (gateway IoT) reduces IoT heterogeneity. In summary, Fog computing reduces latency, bandwidth consumption, and storage space. It provides essential advantages, such as improving QoS, supporting mobility, interoperability, and device location awareness, as depicted in Figure 2.7. This makes Edge Computing advantageous for systems with limited or intermittent connectivity, such as petroleum platforms, as data can be preprocessed at the source without the need for network connectivity.

2.7 Machine Learning

Machine learning algorithms has a critical role in the analytics of IoT data [19]. As a subset of Artificial Intelligence, it has great significance in the IoT network. With the increasing amount of data, day by day produced or generated by our own devices is of no use if not processed properly and utilized for data analytics like Big Data Hadoop, MATLAB, Bluemix, etc. Similarly, data of high volume is generated in the network of

IoT. This network is similar to the wireless network system in terms of architecture to accept a few things. The layers in the architecture require the machine learning algorithms for different levels separately. The standard machine learning algorithms that different researchers are presently utilizing are Naive Bayes (NB), k- Nearest Neighbor (KNN), K- means, Principle Component Analysis (PCA), Support Vector Machine (SVM), Random Forest (RF), etc. These algorithms have different functions and capabilities for accessing and processing the data. The k-means algorithms have significant existence among the group of algorithms present today. It is the oldest and commonly used technique. However, it has one big problem, i.e., the researcher should be confident about the value of the parameter k; otherwise, the algorithms will lead towards null or towards a terrible decision or output. Mohammad Saeid et al. have presented the algorithm for K-means where the cluster center S_k is learning to be optimal for the data and the assignment of the data πnk. K-nearest neighbor method is also one of the simplest methods that is useful to many data scientists. It is effortless to implement both in the classification and regression problems. The assumptions made by the KNN method are so close that the algorithm becomes much more helpful and gives approximate output to the researchers [19].

On the other hand, the Naive Bias method is based on the Bayesian's Theorem; it also proposes a predictive analysis and gives results based on probability which is also approximate to the accurate value. Another supervised ML algorithm, i.e., support vector machine, is the most popular among the data scientists and is mainly used when data visualization is highly required in more than one dimension. PCA finds data patterns where there is a high variation in the data sets. It eliminates the dimensions to find the pattern in the data. The most natural among the machine learning algorithms are neural networks that work on a different platform, i.e., artificial neuron, which is as same as the biological neuron. This algorithm's working falls under the category of Deep Learning (DL), which is a subset of artificial intelligence. The classification of these machine learning algorithms plays a vital role in the IoT network. The selection of algorithms can never be random. The design of the network has substantial dependability on the microcontroller that is incorporated in that IoT network. Table 2.1 below is a comparison of different machine learning algorithms based on their advantages and disadvantages. The best combination of the machine learning

algorithms will give accurate results in the output and accordance with the decision support system. The k-mean helps better in huge variables and produces clusters in a well-shaped manner. Likewise, random forest also is considered good in input variables [20][21]. Table 2.1 Compares different machine learning algorithms for their use case, advantages, and disadvantages.

Table 2.1: Machine learning algorithms based on their advantages and disadvantages.

Algorithm	Use Case for IoT	Advantages	Disadvantages
K-means [22],[23]	Traffic and Air Control monitoring, making the home as a smart home, so will be cities as Smart Cities.	More effective for handling large variables and results in more compact clusters.	Determining the appropriate K-value is challenging when dealing with data of varying densities.
Naïve Bayes [24],[25]	Agriculture data analysis, All citizens as smart citizens.	Easy to implement and requires minimal training data.	Assumption can be incorrect.
K-Nearest Neighbor [26]	Smart Citizen	It remains beneficial even when the training data is noisy.	The value of k must be precisely known.
Support Vector Machine (SVM) [27],[28]	Classify data, real-time Prediction.	Data can be visualized in dimensions exceeding two. The decision-making system is nearly impeccable.	Choosing an appropriate kernel value is a complex task, and when dealing with large datasets, training time increases significantly.
Principle Component Analysis [29-33]	Surveillance of public areas; valuable for detecting faults.	Low redundancy and reduced complexity in images.	Evaluating the covariance matrix is challenging, and even basic invariances might not be captured.
Neural Network [23]	Healthcare Analysis, Forecasting, Low energy consumption and useful with redundant data also.	It has the ability to model and learn Complex relationships. It can also generalize.	Interpreting parameters can be challenging, leading to complexities at times. Needs a large dataset.
Random Forest [23]	l-Banking, Stock Market, Medicine, and E-Commerce.	considered as good for taking thousands of input variables.	Noise in the datasets.

2.8 Classification of Machine Learning Algorithms

In earlier days, if someone would have thought about machines or algorithms that learn by themselves, then nobody would have believed. Today we have machine learning algorithms that learn by themselves with some parameters or training data. The most

important branch of artificial intelligence, i.e., machine learning, has advanced tremendously in recent years. Machine learning has four subsets, i.e., supervised learning, unsupervised Learning (USL), semi-supervised learning, and reinforcement learning. Supervised learning means that the algorithm has some labeled data used by the algorithm/machine to predict the future. Therefore, also known as a predictive model. Unsupervised learning has no labeled data, but its learning process entirely depends on selecting the parameters on which the comparison or the decision-making system works. Therefore, also known as a descriptive model. The reinforcement learning will use none of the above methods except the learning after the failure method. It detects the failure pattern and prevents the system from using it again. Table 2.2 below shows the recent advancements in machine learning algorithms by different researchers.

Table 2.2: Advancements in the machine learning algorithms by different researchers.

Author/name /year	**Algorithms /Techniques**	**Summary**
J. L. Berral-G arcia [34]	SVM, ANN, K-means, DBSCAN, Decision tree Algorithms.	Classification, prediction, and modeling survey was observed and completed using machine learning
Qui, Wu. Q, Ding. G, Feng [35]	Support vector machine, Regression, Neural Networks, Principal Component Analysis.	Using big data processing, A survey of regular and uncommon algorithms was done.
Bokhari, Zeyauddin, and Siddiqui [36]	SVM, ANN, PCA, NB.	They proposed a new model for storage and analysis for Big IoT data.
Wu, W. Cheng, Hoffman and Wang [37]	Principle Component Analysis, Regression	Healthcare analytics are introduced.
V Thool, C. Thool, R. Bendre, [38]	MapReduce, Linear regression	They developed a model using Regression and mapping and Reducing Technologies.

The conclusion from Table 2.1 and Table 2.2 suggests and can help any researcher in selecting an appropriate algorithm for his IoT environment. If k-mean is the choice, then the data should be of a vast number of variables. Similarly, for agriculture, the naïve bayes algorithm, based on the Bayesian theorem, is beneficial as it is elementary to implement and needs very little training data. On the other hand, neural network based on artificial neuron needs a considerable amount of training data and can take

much time. So, it is not suitable for agricultural purposes. The K-Mean has a drawback that almost every researcher has suffered because its value of k, i.e., the number of clusters, is challenging to identify or difficult to guess. Therefore, there is always a risk of bad results or zero percent accuracy.

The NB also has the same disadvantage of wrong assumptions. In K-Nearest Neighbor, the distance of the nearest neighbor is calculated for a cluster. It is helpful in Smart cities because it helps even if the data is noisy, but here, the value of k should be accurate. SVM can visualize the data in more than one dimension. If the data has no capability of classification and no inference line can be drawn, then SVM will ensure that the data is visualized in more than one dimension so that the classification is possible. However, the selection of a good kernel value is challenging. SVM is very good in Real-Time predictions. PCA is used for monitoring public places and for deducting faults. It has a lack of redundancy and decreased complexity in images. However, evaluating the covariance matrix is challenging, and capturing invariance can also be problematic. Random Forest is helpful in I-Banking and Stock Market because it is considered good in tasking, but it has noise in the datasets. Therefore, it is clear from the discussion that the correct technique is directly proportional to the type of data that a researcher needs to process.

The extraction of the information is crucial that even business decisions are also taken based on predictions. The advancement in the machine learning algorithm is significant for healthcare. The day is not far when doctors will also use machine learning techniques for deciding the treatment for a patient. The COVID-19 pracademic has shaken the whole world today. People have waited for almost one year for an antidote that can cure the COVID-19 disease. Machine Learning has the capability that can find patterns and can be an initiative in drawing the inference line that can lead to a better life.

Table 2.3: Machine Learning and Deep Learning Techniques in IoT Environment.

Ref. No.	Technique	Application	Focused Area	Evaluation Parameters	Experiment Setup/Data Set
[39]	Deep Learning		Physiological motion of human skeletons.	Accuracy ratio with MEMS, PHMM, NBDF.	Advanced Spatio-Temporal Extraction Model (ASTEM), Live sensor data of human skeleton.

[40]	TensorFlow Deep Learning Model and LoRa	Smart City	Waste Management	Gradients, weight, placeholders, bias, hyper-parameters, and metadata. And Type of waste metal, plastic or paper.	A prototype (Arduino Uno and Raspberry Pi.)
[41]	Deep Learning	Smart Environment	Water Quality Monitoring	Quality of multimodal data, Misinformation rate, Quality of multimodal data.	water quality monitoring in a coastal area of China.
[42]	Deep Learning	IoT infrastructure	Cellular Networks	Accuracy, EPOCH, Access Rate, No. of IoT devices.	dataset is obtained by using the (conventional) Hungarian algorithm.
[43]	Deep Learning	Natural Power Resources	Wind Power Generator	Power factor, velocity ratio.	_
[44]	Neural Network	Smart City	Waste Management	False Positive/Negative Rate, Precision, True Positive rate, Recall and Accuracy.	Images from the TrashNet dataset.
[45]	Deep Learning	Satellite communication	Feature Extraction	Accuracy, G-mean, etc.	Proxmox Virtual Environment, OpenSAND, OpenBACH, Selenium.
[46]	Deep Learning	Industrial IoT	Image Visualization	Precision, Recall, F-measure, Accuracy.	PythonTensorflow1.9, / Leopard Mobile dataset from IKM Laboratory.
[47]	Bayesian Network prediction	IoT Ecosystem	Cost Efficiency	Signal Strength, Network Type, Network Coverage, CQ, TE, OCS and FTSV.	ESM systems.
[48]	Reinforcement Learning	General Infrastructure	Bandwidth Optimization	Bandwidth, System Cost, Convergence, Weight Factor.	Deep Q-Network.
[49]	Neural Network	Image Processing	Feature Extraction	Distortion image detection, Low resolution image detection, Image detection Speed, Accuracy, CPU utilization and Memory Utilization.	Simulation Tests.
[50]	Artificial Neural Network	Smart Transportation	Passengers	precision recall f1-score lueto, (Walking, Bus, Train, and Bicycle).	Bad.App4 proprietary solution.

[51]	Artificial Neural Network	Smart Farming	Fuzzy Logic (Smart water pump Activation)	Relative humidity, Outside Air Temp., Solar radiation, Speed of Wind and Relative humidity of inside air. (Sum of squares Mean Square and F Value Probability).	ANOVA for validation.
[52]	Deep Learning	Smart City	Waste management	Types of Waste (Paper, Glass, Cup, Plastic, Cardboard,) Accuracy, no. of test images.	ResNet34(PYTORCH,) / Dataset of waste (GITHUB).
[53]	Deep Learning and NN.	Smart Restaurant	Food Data Accuracy	Food Dataset Classes, Accuracy,	Common Crawl, Scrapy, / Dataset of food Pictures.
[54]	Logistic regression, Decision Tree, Naïve Bayes, Multilayer perceptron, Random Forest and k-nearest neighbor.	Food Quality and traceability system.	RFID	Class label, Temperature, humidity (Speed of trolley, Frequency), gID, timestamp, RSS, and antenna ID.	Scikit-learn V0.19.1, Python V3.6.6 and XGBoost V0.81.
[55]	Neural Network	Smart Energy in IoT Environment	WSN	Response rate, delay, overhead, request failures, energy, lifetime, and live node count.	Network simulator.
[56]	RF, LR, KNN, ANN and NB	Smart Farming	Machine Learning	Rainfall, Avg. Rainfall, Avg. Temp., Year, Temperature, Year, Pressure, Avg. Pressure.	Python and Weka.
[57]	K-NN, SVM Navie Bayes (NB), Decision Trees (DT) and (RF)Random Forest and Logistic Regression (LR)	Smart Watches for athletes	Feature extraction and Accuracy	Forehand Drive, Chop, Flick, Backhand Control, Backhand Drive, Chop, Flick and Forehand attack.	Android Studio, SDK, and JDK / Smart watch data of 12 students playing Ping pong.
[58]	Deep Learning	Social Network	Sentiment Analysis	Accuracy, Precision, Recall, F Measure.	Live Streaming of Twitter Data (e.g., FLUME).
[59]	Natural Language Processing and ML methods	Smart City	Environmenta l Monitoring	temperature, humidity, light intensity, noise, pressure, wind strength.	SenSquare h.

[60]	Clustering (K-mean ++)	IoT Infrastructure	WSN	Energy Range, Computational Time, Accuracy, FNR estimation.	NS2 simulation tool.
[61]	Clustering (K-Mean)	General IoT Infrastructure	Cognitive decision Accuracy.	Error Rate, Repeat Times, no. of Neurons, No. of network Layers.	MATLAB 8.1.0.604 (R2013a).
[62]	Clustering, naive Bayes classifier	Smart Industries	Data Streaming	Accuracy, Sensitivity and Specificity, Gaussian parameters (λ, δ, φ and N) window size N.	Industrial Internet Consortium Testbed.
[63]	SVM and KNN	Disaster Management	Noise data sets	IDR, Building State,	HyperMesh, S-DYNA / non-noisy data set and noisy data set.
[64]	QR decomposition and parallel dual ascent.	Improved SVM Training	QR decomposition framework	stopping threshold, optimal step size, η∗, iterations, tc, γ and C.	LIBSVM datasets repository, covtype, Webspam and SUSY.
[65]	ANN, SVM, KNN	Smart City Disaster Management	Fault Tolerance	Presence of smoke, Presence of fire, Leakage of gas, Oil spill, zero fire in the building, zero gas leakage, zero oil spill, etc.	Netlogo 5.3.1 software, / online data repository of UCI, Kaggle, and Data world.
[66]	LDA, NB and SVM	Smart Industry	Manufacturing (Accuracy Rate.)	Accuracy, Frequency, Iteration, preceding level of warning, after level of caution, after level of float.	SEA dataset development technique.
[67]	Supervised Learning (SVM)	Smart Energy	renewable energy resources	Energy Distributed, Energy Saved.	Dataset source: Central Statistics Office Ministry of Statistics and Programme Implementation.

H. Zhang, Z. Fu, and K. Shu have proposed a technique for athletes using smartwatches for better accuracy [57]. They used techniques like K-NN, SVM, NB, LR, DT, and RF. To complete the experimental process, they collected data of smartwatches from the 12 players playing ping pong. The experiment setup was completed by using Android Studio, SDK, and JDK. The focus of the experiment was accuracy. Before that, in October 2018 [62], J. Diaz-Rozo, C. Bielza, and P. Larrañaga used Clustering Technique for better accuracy. The focus was on Industrial environments. To fulfill the

necessary data set requirement, they created the datasets by simulating random values from a Gaussian distributor mixture.

In today's world, machine learning techniques play a vital role in saving lives. As elaborated in paper [59], F. Montori, L. Bedogni, and L. Bononi have shown how ML techniques are useful in environmental monitoring. They have used NLP and other various ML Methods and have proposed a model for smart city.

Among all the techniques, Clustering is the most popular technique in the field of Artificial Intelligence. Paper [60-62] clustering techniques have been used for improving WSN network, IoT infrastructure, and Smart Industries data Streaming, respectively. However, in some instances where the researcher wants to visualize the data in more dimensions for better understanding, the SVM is the most reliable choice for any data scientist. SVM, in combination with other techniques, e.g., KNN, ANN, NB, etc., has performed better and has given more accurate results [63-67]. A. Ibrahim, A. Eltawil, Y. Na, and S. El-Tawil have used SVM and KNN for reduction of noise in the data sets that can be helpful in disaster management [63]. Similarly, Aljumah, A, Ahamed Ahanger, Bhatia, M, and A, Kaur have also worked on disaster management for the smart city, and they have also used ANN, SVM and KNN techniques [65]. Nowadays, the ML methods combination has been a better option for most researchers than using a single ML algorithm. Alfian et al. has proposed a concept of using RFID for food Quality and Traceability system. This experiment used logistic regression, decision tree, naïve bayes, Multilayer perceptron, k-nearest neighbor and random forest [54]. Similarly, Attia A. *et al.* have also used various ML methods for Smart Farming [56].

As a subset of machine learning, Deep Learning is also playing a vital role in data science. The parameters like Accuracy, precision, recall etc. have performed better when using Deep Learning [39], [42], [45], [46], [52], [53]. Similarly, Deep learning has been used in Waste Management, Water Quality Management, Natural Power Resources, Feature extraction, Smart Transportation, Smart Farming, Smart Energy, etc. T. J. Sheng et al. have discussed the challenges that are being currently faced by the City Management in keeping the city clean. In their research article, they have focused on waste management. They have worked on a deep learning model that can help smart dustbins and efficiently manage the waste around the smart city [40]. In

2020, deep learning and neural network concepts were frequently used for waste management. Alqahtani, F., Al-Makhadmeh, Z., Tolba, A. *et al.* have also contributed in the same area [44]. The advancement in the field of AI has driven us towards ML techniques and neural networks. Table 2.3 is an example of ML Methods and Deep Learning [41], [43]. The use of these methods for applications like smart city, smart transportation, smart energy, smart farming, IoT infrastructure, image processing, etc., is exemplified in papers [47], [48], [49], [50], [51], and [55].

Table 2.4: Taxonomy on Data Management/Classification Techniques in IoT Environment.

Ref. No.	**Technique**	**Application**	**Focused Area**	**Evaluation Parameters**	**Experiment Setup/Data Set**
[68]	Clustering & Classification (HLMCC - model)	IoT Infrastructure	Anomaly Detection	Device ID, Sensor Value and Delay Value.	LWSNDR and Landsat Satellite Dataset.
[69]	Classification	Social Network	Redundancy	No. of features, classification results and Total accuracy.	Java programming language with JDK 1.7.0 and GPS Trajectories, Indoor User Movement Prediction from RSS, Water Treatment Plant, Hepatitis and Twitter Dataset.
[70]	Clustering (Hybrid LBGLM)	IoT infrastructure (Classification Problem)	Minimization of labelled dataset	No. of clusters as 'k'. Rank loss, coverage, hamming loss and one error.	MLKNN implementation in MULAN package / CAL 500, emotions, mediamill, scene and yeast.
[71]	Unsupervised Learning Algorithm	Prediction (Predictive Analysis algorithm)	IoT data Classification	K value for data sets.	R-framework on an Intel Corei9-64GB_DDR4-2TB_SSDM2 machine.
[72]	Classification (Capsule network Model)	Smart City	IoT data Classification	Data Flow Length, Data Packet Length, Accuracy.	Ubuntu16.04OS, Python2.7, TensorFlow1.8.0, 4-core CPU and64G memory.
[73]	Classification (SVM)	IoT infrastructure	Feature Extraction (Based on Adaptive boosting.	SVM kernel function's RBF, *(σ) the* Gaussian width, *and* regularization parameters C.	(Simulation experiments) Performed with different values c ranging from 1 to 100.

			(Wireless signal classifiers)		
[74]	Classification (SVM and KNN)	Industrial Environments	Feature Selection	SVM kernel function, *(σ) the* Gaussian width and parameters C. *(σ) the* Gaussian width and parameters C.	(Simulation experiments) Performed with different values c ranging from 1 to 100.
[75]	Multi-stage machine learning based classification framework	Smart City	Classifying IoT Devices	distribution of volume/times during active/sleep periods), and signaling (e.g., domain names requested, server-side port numbers used and TLS handshake exchanges.	Apache server on a virtual machine, TP-Link Archer C7 v2, OpenWrt firmware release Chaos Calmer (15.05.1, r48532).
[76]	Binary Classification techniques	Smart-city cellular infrastructure	RACH-related sleeping cells	KPIs, AUC. False Positive Rate, ROC, Neighborhood Category.	LTE simulator.
[77]	Binary Neural Network	Smart Home (Devices)	Voice Commands Classification	Amplitude, number of voices, Classes, Frequency.	Data set with isolated voice commands (Brazilian Portuguese language) of 150 People.
[78]	Classification (CNN-G, CNN-G-F, Faster R-CNNG)	Smart Cities /Restaurant	Food Images	Accuracy Rate, Types of food, weight, Food Area.	Faster R-CNN / Dish-233 dataset.
[79]	Classification	IoT infrastructure	Quality of Service	appId, Latency L, Bandwidth B, MIPS. latency, energy consumption.	iFogSim.
[80]	Bayesian-based estimation method	IoT Infrastructure	Classification Accuracy	Uncertainty parameter, Accuracy, No. of Sampling Points.	Standard benchmark datasets from UCI machine learning repository and a real-world OTA dataset.

[81]	Classification (KNN, SVM, DT and LSTM)	General Infrastructure	Web News Data accuracy	Accuracy, Sensitivity, Specificity, Time and Space Complexity.	MATLAB fitctree.
[82]	SVM, nearest centroid and Naïve Bayes.	SDN (Software Defined Networks)	Classification of Data	Accuracy, Training Factor, Precision and Recall and F-Score.	tcpdump'a packet analyzer utility.
[83]	Optimization algorithm of IoT data	IoT Infrastructure	Data Storage	User Scale, No of Files created, File size, Fault tolerance. Downloading and uploading.	OPNET Modeler.
[84]	Big data Analysis	General Infrastructure	Data Management	Distance as a weighting parameter, Data Node, Data size and Node name.	SunJava6, Hadoop 1.03 and Ubuntu Linux 10.04 / online terminal analysis, OTA-selected training set.
[85]	Deep Learning (Tensor Train Approach)	General IoT infrastructure	Minimizing IoT data Traffic	SDN Control Traffic Control, Packet Size, Rate of Traffic, Ethernet port Speed, Service Rate, and Packet Send Interval.	Raspberry Pi 3 (1.2 GHz CPU,1 GB RAM.
[86]	Decision Tree, Logistic Regression and SVM	Social Network	Data Filtering	weighted value, frequency, idf.	Spark cluster, Spark YARN.
[87]	Neural Network	General IoT Infrastructure	Data Recovery	Kernel, Dilation, Stride, Output.	200,000 pieces of data from Kaggle.
[88]	Deep Learning	IoT Infrastructure, Smart Industry.	Data Compression	Power, Battery Performance, Accuracy, Decomposition Level, Error, Memory, bandwidth and Processor performance.	Datasets of UCI multivariate time series, UCR univariate time series and UEA multivariate time series.

Resource optimization is crucial in the IoT environment. IoT has limitations in terms of bandwidth, Processing Power, Energy, Memory, etc. This weakness of resource constraints in the IoT environment has put the IoT data flow in a state causing the delay. This problem cannot be solved by increasing the capacity of Hardware because we cannot increase the size of the IoT devices. Therefore, Data Management is the only solution to such a problem. Data management in the IoT environment can be done in various ways. Many researchers have used different techniques for managing data in an IoT environment. Table 2.4 represents the well-known research in the field. S. Lin, C. Chen and T. Lee have proposed a hybrid clustering technique, namely LBGLM, to minimize the labeled dataset in an IoT environment. They have proposed a method for the classification problem for the IoT infrastructure [70]. Classification is essential in data management and it can optimize the IoT resources at the first level in data management. H. Yao, P. Gao, J. Wang, P. Zhang, C. Jiang, and Z. Han have proposed a smart city model. This model can classify data with more accuracy [72]. Similarly, J. Huang, L. Zhu, Q. Liang, B. Fan, and S. Li [80] have also worked on the classification-accuracy using the Bayesian-based estimation method. They have used standard benchmark datasets from the UCI machine learning repository and a real-world OTA dataset. M. Raikar et al. have shown the importance of SVM, nearest centroid, and Naïve Bayes and have compared the Accuracy and have proposed a technique for Data Classification. For their experiment, they have used the tcpdump packet analyzer utility [82]. G. Casolla, S. Cuomo, V. S. d. Cola and F. Piccialli have proposed an Algorithm (Predictive Analysis algorithm) that can help predict the possible data set for Classification. They have used R-framework on an Intel Corei9-64GB_DDR4-2TB_SSDM2 machine for their experiment [71]. Data management is not limited to Classification, but it is also about data storage, data filtering, data compression, data Recovery, data Accuracy, etc. As Wang, M., & Zhang, Q has proposed an algorithm for optimization of IoT network. The focus of the experiment was data storage and was completed in OPNET modeler [83]. Hsu I., and Chang C, have used ML methods such as decision tree, logistic regression, and SVM for data filtering, which can be beneficial for social networking. Their work can be helpful if stopping the wrong information spread unnecessarily on social networks [86]. Azar J et al. have focused on the compression of IoT data. This paper has used deep learning and has used the datasets

of UCR univariate time series [88]. Similarly, [85] has also used deep learning to minimize the IoT data traffic. They used the Tensor Train Approach for the minimization of the IoT data.

The researchers have focused on data management rather than resource upgrade, as data management will work till the end. The data coming from the sensors is mostly the raw data and needs a lot of filtering because of the anomalies and redundancies. In the paper [68] and [69], the researchers have used different clustering and classification techniques for anomaly detection and removal of redundancy, respectively. Paper [68] has worked on the delay value and sensor value for the anomaly detection, and paper [69] has worked on the accuracy. Nowadays, people are mainly using the internet for news reading instead of radio and television, and there is a lot of wrong news information on the internet. To handle such a problem, Mulahuwaish et al. has classified data in terms of news data accuracy. They have used different ML methods such as kNN, SVM, DT, and LSTM and have used MATLAB fitctree for their experiment. This will help minimize the wrong information on the internet and increase the accuracy of Web news data [81]. Hou, R., Kong, Y., Cai, B. *et al.* have used Big Data analysis for data management in an IoT Environment [84]. Whenever we discuss data management, data recovery also plays a significant role in today's world. Shi, Y., Zhang, X., Hu, Q. *et al.* have worked on NN for data recovery. They have used 200,000 pieces of data from Kaggle for their experiment [87].

Similarly, [77] has also used neural network for smart devices in a smart home. They have worked on the voice commands classification. Moreover, they have used dataset with isolated voice commands (Brazilian Portuguese language) of 150 People.

Table 2.5: Machine Learning Technique for feature extraction and Smart Health.

Ref. No.	**Technique**	**Application**	**Focused Area**	**Evaluation Parameters**	**Experiment Setup/Data Set**
[89]	Decision Tree Based Partition	HealthCare	Heart Disease	Accuracy, Sensitivity and Precision. (Sex, Chest pain, B.P, Chol, Resting, Thali, Exang, Old Peak, Slope, CA, Thal and Num.)	UCI machine learning repository (Cleveland, Hungary, Switzerland, and the VA Long Beach).

[90]	Deep learning and Big data	Smart City	HealthCare	Latency, Bandwidth, Energy Efficiency, Reliability and Security.	ICanCloud, / ISPDSL-II, Waikato-VIII and WIDE-18.
[91]	Image Classification	HealthCare	(Image analysis) of the brain and chest.	Accuracy and Kappa coefficient.	12000 CT images of brain, chest and cervical spine.
[92]	Neural Network	Healthcare	Chronic Kidney Disease	weight and bias value.	CT images in renal cancer.
[93]	Random tree-based classifier	Healthcare	Diabetes Prediction	No. of neurons, No. of hidden layers, learning rate, epoch, activation function, neuron initializer, batch size, percentage of dropped neurons, loss function, the optimizer.	PIMA Indians Diabetes (PID) dataset of 768 female diabetic patients.
[94]	Deep Learning	Healthcare	Feature Extraction and Detection	No. of abnormal cells, abnormal cells, Precision, recall, support.	Pap smear images / Herlev dataset from Denmark Hospital.
[95]	Big Data (Spark)	Healthcare	Prediction	throughput and execution time, impurity, maxDepth and maxBins.	diabetic data from Kaggle, cleveland.data of heart disease.
[96]	ML (Support Vector Machine) DT, RF, and MLP	Healthcare	Emergency Response time	Accuracy, recall, precision and F-score.	Tenfold cross-validation / dataset for student health gathered from 1100 instances.
[97]	Neural Network	Healthcare	Perceptron	Accuracy, Average Performance, R-Value.	UCI machine learning repository obtained by the Garvan institute.
[98]	K-NN, SVM, Random Forest, Decision trees and MLP.	Healthcare	Data set accuracy	No. of patients, No. of disease types, Area Under curve using ML.	(WEKA) open-source tool / Online health data, datasets are downloaded from https://archive.ics.uci.edu/ml/datasets.html.
[99]	Support Vector Machine (SVM Classifier)	Healthcare	Feature Extraction	Detection, moving window integration, and pulse train. Filtering, squaring, and thresholding.	MATLAB classification learner app. / ECG acquisition system real time data.
[100]	Deep Learning	Healthcare	Prediction	Requested object identifier, request date/time, list of object's	Biology database, DNA Data Bank of Japan, GenBank and European Nucleotide Archive.

				constituents and RT name or geographical location.	
[101]	Deep Learning	Healthcare	Classification of skin cancer data	Recall, F1-score, Accuracy, Average accuracy, and Class Precision.	VGG19, Inception V3, SqueezeNet, and ResNet50 / (ISIC) image archive.
[102]	Probabilistic neural network (PNN)	Healthcare	Wearable Devices	Heart rate avg. R-R interval avg. Peak QRS avg. Peak T-wave avg. Duration of QRS avg.	Real time data from different players.
[103]	Bayesian Neural Network (BNN) and K-Nearest Neighbor) K-NN.	Healthcare	IoT-Fog Integration	Blood pressure, blood sugar, Body temperature, Heart Rate, True Positive Value, Throughput. Accuracy, Precision, Sensitivity and recall.	Real Time dataset of patients.
[104]	FUEHMF, 5G Technology	Smart City	Healthcare	Delay, No. of FoG servers, No. of Gateway Connected, Avg. Competition, Lost Rate data packet queue.	SSOAF, CBSF, Fog computing architecture for smart cities (FCASC) and FCRMFSC.
[105]	Fog Computing	Smart Home	Healthcare	Precision, Recall, F-measure, Specificity, Heart Rate, Blood Pressure, Respiration Rate, Gastro-intestinal tract, ECG.	UCI data repository, US EPA data repository, Health Related Datasets HRD, Datasets ERD, Datasets BRD.
[106]	Bayesian Classification	Healthcare	Air quality Forecast	Primary Pollutant, Pollution Level, PM 2.5.	Hadoop, HDFS, MapReduce.
[107]	Classification (SVM)	Healthcare	Feature Selection	mean-accuracies, mean-reduct-sizes, Accuracy.	MATLAB R2016a / BCI competition-II Dataset-III from Department of Medical Informatics.
[108]	Deep Learning	Healthcare	Predictive Model	cross-entropy loss and class balance. Survival rate and Accuracy.	clinical datasets from http://biogps.org.
[109]	Deep Neural Network	Healthcare	Heart Disease Data prediction	Accuracy, Sensitivity, FPR, Precision, Specificity, F-measure and G-mean.	PASCAL B-training dataset.

[110]	SOM and ORNN	Healthcare	Cancer Detection	Accuracy, Sensitivity, Specificity and low root mean square error.	Cloud, Coil100 and CIFAR-10 datasets.
[111]	Neural Network	Healthcare	Data Classification	Batch size, the learning rate, and the epoch size. Accuracy.	Herlev dataset.
[112]	Decision Tree, Naive bayes, Logistic Regression and Random Forest.	Healthcare	Prediction Model	Precision, F-Measure, Recall ROC and Accuracy.	Diabetes risk prediction dataset collected from a diabetes hospital.

Machine Learning is playing a vital role in the field of healthcare. Many researchers have proposed models and different algorithms to detect dangerous diseases like cancer in its earlier stages. Today the world is under threat due to the pandemic of Covid-19. Moreover, the day is not far when researchers will also use machine learning techniques to deal with all kinds of viruses. M. K. Hasan et al. and Hossain et al. have used ML methods to predict diabetes. They have used a random tree-based classifier and naive Bayes, logistic regression, decision tree, and random forest, respectively [93], [112]. In recent years researchers have worked on many healthcare applications for Smart Cities. As shown in Table 2.5, ML methods for improving the healthcare system have evolved from 2018 to 2020. T. Muhammed et al. have used deep learning and big data for smart cities [90]. Similarly, Paper [104] proposed a technique using a different method as FUEHMF, 5G Technology.

In the healthcare system, heart monitoring and heart disease are very important for IoT devices and smart health for Smart Cities. In the domain of Heart Monitoring, several ML methods have been exclusively applied in papers [89], [95], [102], [103], [105], and [109]. Notably, S. Mohan et al. have focused on heart diseases and utilized the decision tree-based partition technique [109]. The experiment collected a dataset from the UCI machine learning repository (Cleveland, Hungary, Switzerland, and the VA Long Beach) [89]. Similarly, Deperlioglu et al. has used the technique of deep neural network for heart disease data prediction and have used the PASCAL B-training dataset for their experiment [109]. In Table 2.5, some researchers have worked on the detection of cancer [92], [101], [110]. As G. Chen et al. have taken the CT images in

renal cancer and have used the neural network (Deep Convolution) method for detecting chronic kidney disease [92]. Similarly, A. Khamparia et al. have classified the skin cancer data by using deep learning [101]. Elhoseny et al. have also worked on cancer detection by using SOM and ORNN. For their experiment, they have used Coil100 and CIFAR-10 datasets [110].

Image classification has also played a vital role in today's healthcare monitoring system. H. Tang and Z. Hu have used 12000 CT images of the brain, chest, and cervical spine for their experiment of image analysis of the brain and chest [91]. Similarly, [94] has also used the Herlev dataset of Pap smear images from Denmark hospital for feature extraction and detection. Kesavan et al. has focused on the importance of IoT-Fog Integration using BNN and K-NN for the healthcare system by taking the real-time dataset of patients [103].

In day-to-day life, wearable devices have become our necessity. As it monitors our blood pressure, heart rate, and our calorie burn count in real-time. Some researchers have also focused on the area where wearable devices can be taken for a better healthcare monitoring system. Y. Atif et al. have taken real-time data from different players using IoT wearable devices and have used the Probabilistic Neural Network (PNN) technique [102].

Improving the quality of air is also part of our health. Y. Huang et al. worked on pollution parameters such as primary pollutants, pollution level, PM 2.5, etc. They have used the Bayesian classification technique for air quality forecast [106].

In case of emergency, the response-time has to be significantly more less. Souri et al. has used a dataset of student health gathered from 1100 instances and used an ML method such as SVM, DT, RF, and MLP in Tenfold cross-validation to improve Emergency response time [96]. Similarly, [98-99] and [107] have also used SVM for data accuracy and feature extraction for a better healthcare system in smart cities.

Table 2.6: Taxonomy on ML Approaches for Privacy and Security of IoT Infrastructure.

Ref. No.	Technique	Application	Focused Area	Evaluation Parameters	Experiment Setup/Data Set
[113]	Deep Learning (RNN)	General Infrastructure	Fog Security	DR and FAR	Keras on TensorFlow/ RPL-NIDS-2017 and N_BaIoT-2018.

[114]	Support Vector Machine and Block chain	Smart City	Privacy/Security	Gradient, size, class label, dimensions and the Euler phi-function.	Testbed/ Breast Cancer Wisconsin Data Set (BCWD) and Data Set of (HDD) heart disease.
[115]	Classification (NB)	Security	Spam email	Email percentage. Precision, F-Measure, Accuracy, True Positive, Negative, FN, TP and TN.	RStudio / Enron, PU1, UCU Spambase and Ling-spam.
[116]	MOEA (MOPSO-Lévy)	IoT infrastructure	Security	Acceleration constants, Inertia weight, inertia damping rate, No. of grids per dimension, Accuracy, (TPR), False alarm rate (FAR), (TNR) and Precision.	MATLAB 2016 / Datasets as Baby monitor, Danmini doorbell, Security camera PT737, Security camera PT838, Ecobee thermostat.
[117]	Supervised, Unsupervised and Reinforcement	IoT Infrastructure	Security	-	-
[118]	Neural Network	IoT Infrastructure Security	Botnet detection	Byte in Per Flow, false positive rate (FPR), Fβ, true positive rate (TPR) and Accuracy rate (ACC).	pkt2flow tool / ISCX-Bot-2014 data set, ISOT data set, ISCX 2012 IDS data set.
[119]	Support Vector Machine	IoT Infrastructure	Security	Detection accuracy rate, Average detection accuracy rate, Average, False alarm rate.	Floodlight, Mininet emulation. (Ubuntu).
[120]	Classification (MoE Neural Network)	IoT Infrastructure	Security	Comparison of Machine Learning Algorithms in terms of Accuracy (%).	VirusShare malware dataset.
[121]	Neural Network	General Infrastructure	Security	Precision, Accuracy, Recall and F1-score.	CICIDS2017 data set and Live Data Collection using different IoT devices, dpkt python library.
[122]	Deep Learning	General Infrastructure	Security	Mean Squared Error, Data times, Data Volume, Decision Times and Decision Error.	Keras, Python2.79, Matplotlib, Numpy, Hadoop Distributed File System (HDFS).

[123]	Deep Neural Network	Smart City	Security	Accuracy, Sensitivity, FPR, Precision, Specificity, F1-Score.	S2OS, NumPy framework and Pandas framework, Matplotlib, Keras framework and Scikit-learn framework / DS2OS data set.
[124]	K-NN Classification	Secure Algorithm	K Nearest Neighbor	Running time, Communication Cost, max. bucket length, cyclotomic polynomial, No. of nearest neighbors.	ESkNN system. PPEDP and OT, HElib, EMP toolkit.
[125]	Deep Learning DL-IDS	IoT Infrastructure	Security	Accuracy, Precision, Recall, F-1 score.	NSL-KDD benchmark data set, KDD'99 data set.
[126]	Random forest, K-nearest neighbor and naïve Bayes	IoT Security	Image Texture	Toxicity ratio τ, Correlation, ASM and IDM. Also, Precision, Recall, F-measure, Overall Accuracy in %Kappa (-1 to +1, Entropy and Contrast.	IoT malware dataset.
[127]	Support Vector Machine	IoT infrastructure Security	Data Mining	Accuracy rate, Number of messages, Correct Messages, Normal information and Spam information.	2200 real SMS provided by a mobile communication operator.
[128]	Bijective Soft Set and (NB, RF, DT and BN)	Smart City	Security	Accuracy Precision Recall TP Rate TTBM	Bot-IoT dataset
[129]	Deep Learning	IoT infrastructure	Fog Layer Security	Layer of (Convolutional, Pooling and Hidden), Nodes in First and 2nd Hidden Layer, Precision, Recall, Fall out, Accuracy and F-measure.	Apache Spark, / UNSW's Bot-IoT Dataset.

Machine Learning approach has already proved its importance by serving humanity in healthcare, smart city, smart farming, smart energy, smart home, etc. Security is the required field of the IoT environment. The fulfilment of confidentiality, integrity, and availability is the prime goal of every wireless system Protocol. Machine learning has also contributed to the fulfilment of these requirements. The researchers have used different ML methods for improving the security and privacy of IoT infrastructure. Table 2.6 shows the use of the ML techniques for IoT security. The researchers have

used SVM for improving security in the IoT environment [114], [119], [127]. Y. Chen et al. has worked on 2200 real SMSs provided by a mobile communication operator and have used the SVM technique for data mining to improve the IoT Infrastructure and Security [127].

Today we can see our mailbox full of spam data that are fraudsters attempting to take advantage of the infrastructure's security system's loopholes. Venkatraman et al. has proposed a technique and have used the Naive bias classification technique in the experiment to improve the classification of spam emails [115].

A. Samy, H. Yu, and H. Zhang have used Keras on TensorFlow for their research experiment and have worked on the technique Deep Learning (RNN) [113]. Similarly, Habib et al. have proposed a technique MOEA (MOPSO-Lévy) that will improve IoT security. Their experiment used MATLAB 2016 and collected the dataset as data of the Baby monitor, Danmini doorbell, Security camera PT737, Security camera PT838, Ecobee thermostat [116].

As a subset of the machine learning approach, deep learning has its significance and has evolved drastically, helping researchers in many experiments related to data science. [122], [123], [125], [129].

Reddy et al. has used the technique deep neural network for improving the security of Smart Cities. Their experiment used the S2OS, NumPy framework, and Panda's framework and used the DS2OS dataset [123].

Nowadays, many researchers are creating and proposing different hybrid ML algorithms for improving the security of the IoT environment. Some have proposed improving the security at the first level, that is, the Device level, while some researchers are proposing the scope of improvement at the Edge/Fog Level. M. Sun et al. has proposed a technique using KNN that will improve and make the algorithm more secure [124]. Similarly, Otoum et al. has proposed a Deep Learning DL-IDS method that will also improve the IoT infrastructure's security and privacy. Their experiment used the NSL-KDD benchmark dataset, namely the KDD'99 dataset [125]. Karanja et al. have used and compared many techniques like RF, KNN, and NB for measuring and comparing the toxicity ratio τ, correlation, ASM and IDM, precision, recall, F-measure, overall accuracy in kappa. Their experiment has collected an IoT malware dataset and has worked chiefly on image texture [126].

X. Dong et al. have used neural network for Botnet detection. The experiment was carried out using the pkt2flow tool, and the experiment was done using the ISCX-Bot-2014 dataset, ISOT dataset, ISCX 2012, and IDS dataset [118]. Similarly, [121] has also used neural network, and the experiment was done by using the CICIDS2017 dataset and live data collection using different IoT devices.

2.9 Summary

The resources of an IoT environment are limited. The nature of pervasiveness of the IoT devices has created a boundary for the researchers. Therefore, the data classification remains the best solution for this resource constrained IoT environment. In this study, a state-of-the-art survey has been carried out on IoT network and it was found that different machine learning algorithms have been used by the researchers for data classification. Machine Learning approaches has proved its importance by serving humanity in healthcare, smart city, smart farming, smart energy, smart home, etc. The advancements in the machine learning techniques have improved over the last several years, which results in the practicing the hybrid techniques in the field of artificial intelligence. From this chapter we can conclude that IoT data Classification needs a Hybrid data classification algorithm which can optimize the resources at the resource constrained IoT Network. The KNN has been used by most of the researchers for data classification. Some researchers have been combining the KNN with the deep learning. We will try to find out the best solution for the resource constrained problem of this IoT network.

In this chapter, a comparative analysis of machine learning techniques was proposed to classify IoT data for resource constrained IoT applications. For this purpose, data from different research papers was collected to create two datasets. The data sets are used for experiment and the results are shown in the fifth chapter of the thesis.

CHAPTER 3

ANALYSIS AND CLASSIFICATION OF IoT-DATA AT DEVICE LEVEL AND EDGE LEVEL

3.1 Introduction

In today's world, we are much more dependent on the internet than we were before. The machines' ease and dependability have led us towards a new scenario where everything connects to the Internet. Therefore, giving potential to the IoT has benefited us in numerous ways. The IoT emerges when different heterogeneous devices are connected and are communicating, which provides data of every kind. As the Internet of things is getting bigger since wireless communication started developing and getting importance. Today we can see every house in the metro cities is equipped with some IoT device. IoT devices are connected through a wireless router; therefore, these routers are used to transfer data and communicate with the environment cognitively. The most common example of these devices is Google Home, Alexa, smart light, etc. [130]. All these devices have different network policies for accessing and controlling the Internet. The environment in which this whole process is carried out is resource-constrained. The existing Framework for the IoT environment has certain constraints such as processing power, Storage, battery/energy, and bandwidth, hence causing the delay. Many heterogeneous end devices are interconnected, which collects data in almost every format is flooding the IoT environment [131-135].

In an IoT environment, seamless intermediate network devices are used to communicate, known as IoT gateway. The IoT Gateway provides information on this data that flows in both directions, and also it acts as a translator. Hence it contains two different protocols to communicate in day-to-day life, we come across so many various sensors that acquire different types of data. Every sensor has another purpose, and these sensors are enormous in numbers, so the amount of data is tremendously high. Managing a massive amount of data that comes from thousands of sensors is a challenging thing. The sensors such as optical sensor, infrared sensor, accelerometer, gas sensor, gyroscope, level sensor, pressure sensor, proximity sensor, temperature sensor, humidity sensor, etc., collect different data types of different volume and variety. To store and process, such a massive amount of data requires a flexible architecture/framework. The data captured by the sensors flows through different channels, i.e., device level, Fog/Edge level, and reaches the top-level, i.e., cloud level. Figure 3.1 describes the layers of an IoT architecture. An IoT architecture's main functions should be Sensing, Aggregation of data, Communication, and Utilization of data at the application level for providing service. Apart from the functions of an IoT Architecture, it should be capable of processing the ever-increasing data with its resource-constrained layers, i.e., device layer and Edge/Fog layer.

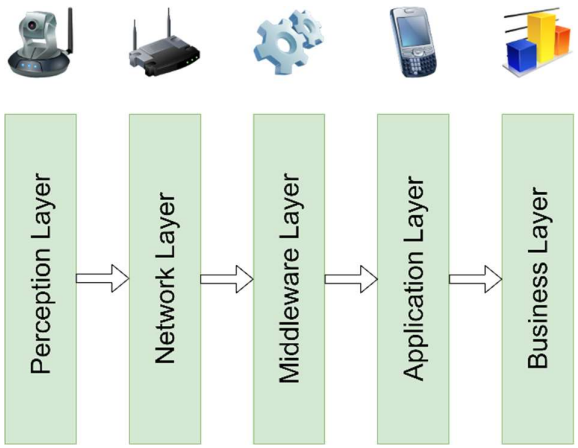


Figure 3.1: Layers of IoT Architecture.

3.2 Resource Constrained IoT Architecture

An IoT Network's environment consists of limited resources such as processing power, storage, battery/energy, and bandwidth that causes delay. In an IoT environment, their

limited resources are the main essential aspects of concern because of ever-increasing data [136]. With the increase in the data every day, the resources need to be upgraded with time. The limitations such as size, battery /energy, and the nature of pervasiveness restrict an IoT device. The three levels of an IoT architecture i.e., device level, Edge/Fog level and cloud level are shown in Figure 3.2.

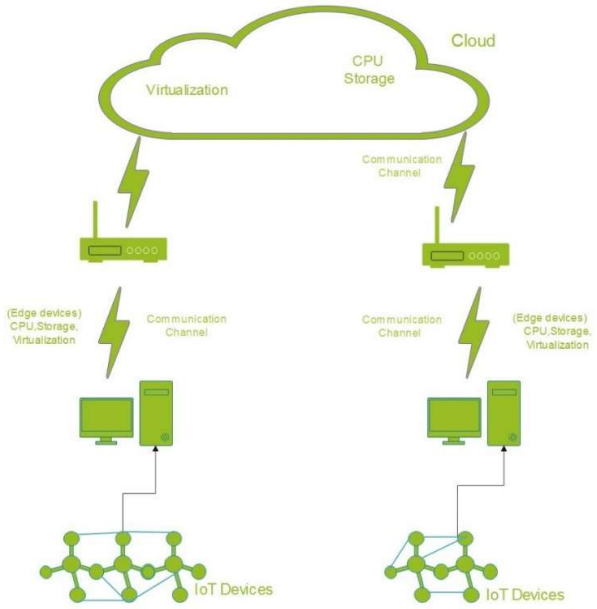


Figure 3.2: Three Tier IoT environment.

The resource constraints in an IoT environment are processing power, battery, energy, bandwidth, and storage. The constraint parameters remain the same in both the layers of an IoT environment, i.e., device/Sensor layer and Edge/Fog layer. An IoT device/Sensor receives enormous data from the surroundings. IoT devices need good processing power, storage, battery/energy, and bandwidth to process a massive amount of data. Due to the restrictions and demand for making small IoT devices, the battery's size becomes a big problem in a pervasive IoT device. Therefore, much work on data management has been done, and numerous energy-efficient algorithms are being employed to process, filter, store, and transfer data. The algorithms work based on the application's requirement; similarly, the massive amount of data demands good processor power for processing and storing the data. Suppose we cannot increase the size of an IoT device. In that case, we have only one Solution and, i.e., data management in an IoT environment to optimize the available resources [137-138]. IoT is changing

the manufacturing and edge innovation rules in the industrial sector; this has led the industrial sector transformation into digital factories and manufacturing intelligently. According to the 2018 analysis, IoT application manufacturing has the top priority in the world. The most popular organization, such as Microsoft and AWS, are among the top manufacturers of IoT applications. Smart manufacturing has become the goal of every leading industry across the globe. As its IoT application has a wide range of Connected devices both inside and outside the industry.

The IoT has numerous applications for various sectors such as transport, weather monitoring, water supply, Health Care homes, offices, agriculture, and straw. Etc. But all these applications share a common problem: limitations of resources 50 memory storage processing power and energy. The effect of resource-constrained in the environment of Internet of Things has made it harder to process the data at the device level [139-141].

The sensors and pervasive IoT devices are the building block of the IoT network. Where IoT device is sense collect and analyse the data via different IoT nodes. The purpose of this IoT network is to sense, facilitate and utilize numerous resources and provide service in real-time. This network is divided into three levels that is device-level, Fog level and the cloud level. During the last few years, much work has been done on the universal architecture for an IoT environment, as shown in Table 3.1.

Table 3.1: Research Work on System Architecture Framework.

Research Work	**Focused Area**	**Author & References**
System Architectures	Application Frameworks	H. Hsieh et al. [142], A. P. Castellani et al. [143], J. Kiljander et al. [144], J. Gubbi et al. [145].
	Cloud Centric Architectures	G. C. Fox et al. [146], A. P. Castellani et al. [143], Q. Wu et al. [147], J. Gubbi et al. [145], D. Mazza et al. [148], A. Munir et al. [149], A. Brogi et al. [150], A. Gupta et al. [153].
	Hardware Architectures	C. Sarkar et al. [154], Morgan [155], A. Al-Fuqaha et al. [157], H. Hada et al. [158], Miao Yun et al. [159], A. H. Ngu et al. [164], M. A. Razzaque et al. [165], S. K. Datta et al. [167], K. K. Karmakar et al. [168].
	Conceptual Models	M. Kim et al. [169], J. Stankovic et al. [170], D. Guinard [17], O. Elijah et al. [171], S. Chen et al. [172], A. Whitmore et al. [173], A. Athira et al.[174], M. Díaz et al. [176], A. Bassi et al. [177], A. H. Alhamedi et al. [182] A. Javed [183], O. Kaiwartya et al. [184] M. R. Palattella et al. [185].

	Process Architectures	Z. J. Muhsin et al. [187], F. Kawsar et al. [188], D. Mazza et al. [189], S. Sarkar et al. [190].
	Restful architectures	S. Hong et al [161], D. Guinard et al. [192], A. Rizzardi [193], I. Grønbæk [194].
	SOA Architectures	A. Athira et al. [174], A. P. Castellani et al. [143], B. Cheng et al. [195], P. Spiess et al [196], I. Chen et al. [197].

IoT came into existence in 1999; since then, it is growing with a fierce amount of velocity. The basic idea is to connect the day today's electronic equipment of households with the internet and provide a cognitive approach towards the development of artificial intelligence. The idea was also beneficial for many other sub-line economic attributes, for example, business marketing, decision-making, Healthcare, E-Commerce, agriculture, manufacturing, industries, social networking, and many more. The communication between user and cloud storage via a common platform gives rise to the data analytic and decision-making system. And it is attracting many business organizations and industries investing in IoT research. Add business organizations can find patterns and make important business decisions with the help of Big IoT Data Analytics [198].

The Big IoT environment is flooded with the data having the property of 6 V's that is, volume, velocity, variety, value, veracity, and variability; therefore, it is called big IoT data. The base of an IoT environment is like a wireless network that combines a large number of nodes (small sensors). These sensors are basically for sensing and taking the input, e.g., sensing pollution, humidity, monitoring bridge, etc. Most of the data that comes from these sensors are generally streaming data in the form of measurements or events happening at a particular time. Some sensors are deployed over that type of application which generates a large amount of data at high speed. This continuous flow of data coming from different types of sensors has emerged the necessity to a different and modern technique, framework, and tools to manage this huge amount of data which is mostly unstructured. This unstructured data is also increasing day by data and because of the increase in the smart devices usually has a sensor embedded in them. It is also believed that this data will increase with a high speed in the coming years [199].

3.3 Sensor Data Format

Sensor data consists of data collected from various sensors at the device level. This data comes from sensors like GPS, Accelerometer, Gyro, Magnetometer, Barometer, Thermometer, image sensors, Motion detector, smoke detector, audio sensors, etc. The data of a sensor is processed at different frequencies depending upon the type of sensor. For example, sensors like GPS, Barometer, and Thermometer will process at a frequency of 1Hz [200]. The data sensed by the sensor is recorded without any change. The data gets stored in the following way:

Table 3.2: Data Stored from The Sensor.

Data Type	Payload		Data Type	Payload
1 Byte	N Bytes	.....	1 Byte	N Bytes

As shown in Table 3.2, the first column contains the data type of 1 byte. The second includes the payload of N bytes. When data is transmitted over the internet, each unit has the header information and the actual data. The header recognizes the destination and the source of the data packet, and the original data is called as payload. The receiver on the other end only receives this payload. Following Table 3.3 represents data/sensor types currently defined along with corresponding descriptions:

Table 3.3: Sensor Datatype.

Sensors	**File Type**	**Description/Example**
Temperature	XML text (Float)	Ambient Temperature. Temperature inside of a housing in milli centigrade
Image	TIFF version 6 uncompressed (.tif) (Binary)	Grayscale or RGB combination.
GPS	vector and raster data (.gpx)	Packet gps-data structure containing relevant GPS data.
Video	MPEG-4 High Profile (.mp4) motion JPEG 2000 (.jp2) (Discrete)	Captures Continues images.
Light	Integer	Illuminance.

Accelerometer	Text	Acceleration in milli Gs for x, y and z axes, correspondingly.
Motion	discrete	Detection of the changed value.
Smoke	float	Alarm is generated because of difference in the atmosphere due to smoke.

Table 3.4 below shows the GPS data structure. The values are stored in little endian format. All integers are 4 bytes (total size 92 bytes) and all doubles are 8 bytes long. Also, some data may be undefined due to FPS status.

Table 3.4: Image Sensor Requirements For Processing.

Type	**Rate**	**Bandwidth**		**Size**
	(Hz)	**Byte/s**	**(Byte)**	**Bit/s**
JPEG Format	0.05	250k	~ 5M	2000k
RAW Data	0.05	650k	~13 M	5200k

3.4 Recent IoT Devices and Technologies

The heart of an IoT device is its microcontroller/processor, an (SoC) service on-chip responsible for data processing and storage. Microcontroller development boards are the best platforms to inherit these microcontrollers and provide various services that allow the user to program microcontrollers according to the application. Many microcontrollers have been developed, but very few are being used and are typical for the devices or sensors. Some of the standard SBCs single-board computers are Arduino Uno, Particle Electron, Espressif system ESP8266-01, ATMega328P Processor, etc. Table 3.4 shows all the standard microcontrollers that are being used for an IoT environment. Arduino is the most popular in an embedded (MCK) Microcontroller kit and is used to incorporate devices, for example, sensors that can collect information from the environment.

The selection of a suitable microcontroller for an IoT device depends on the type of sensors and number of sensors incorporated on that device. The microcontroller that could match our needs, should coordinate reading and should match the desired output. The data communication protocol also plays a vital role. It should be such that it is used

between intra-device communication. In the end, we have to keep in mind the communication of the device with the edge-level or cloud-level. Therefore, the hardware capable of communicating with the cloud should be given the highest priority. Table 3.5 shows the comparison of different Microcontrollers commonly used in an IoT device.

Table 3.5: Different Microcontrollers commonly used in IoT Devices.

Name	Processor	Processing Power	Memory	Power/ Battery	Ref. No.
Arduino Uno	ATMega328P	16KHz	32kb flash, 1kb EEPROM	500 mA	[201],[204]
Particle Electron	32-bit STM32F205 ARM Cortex M3	120 MHz	1 MB Flash, 128 kb RAM	3.9V-12V DC	[201],[204]
Espressif System ESP8266-01	32-Bit Tensilica L106	80 MHz	1 MB	300 mA	[201],[104]
Rasberry Pi4	ARM Cortex A72	1.5 GHz	1-4 GB	5V 3A	[202],[205]
BeagleBone Black	AM335X ARM Cortex A8	1 GHz	512 MB RAM, 4GB Flash	5V 12A – 2A	[201],[205]
Qualcomn DragonBoard 410c	ARM Cortex A53	1.2 GHz	1GB, 8GB Flash.	6.5 – 18v 2A	[201],[204]
PICI6C5X	8-Bit	40Mhz	2KB EEPROM	-	[132],[206]
MSP430	16-Bit	16 MHz	512 KB	-	[205],[206]
Mega AVR	8-Bit AVR	16-20 MHz	4-256KB	-	[201]
Particle Electron	32-Bit ARM Cortex M3	120 MHz	128 KB RAM, 32 KB Flash	-	[201]
Adafruit feather 32u4 FONA	32-Bit AT Mega 32u4	120 MHz	2KB RAM 32KB Flash	-	[202]
Hologram Dash	32-Bit ARM Cortex M4	120 MHz	128KB RAM, 1MB Flash	-	[204]
LinkIT One	32-Bit MT2502A, ARM7EJ-S	260 MHz	4MB RAM, 16 MB Flash.	-	[204]
GOBLIN 2	32-Bit AT Mega328P	16 MHz	2KB RAM, 32 KB Flash.	-	[204],[206]

PIC 18F4550	8- Bit	31 KHz to 48 MHz	256 bytes EEPROM, 2KB RAM	-	[201],[204]
8051 Microcontrollers	32-Bit		4KB ROM, 128 Bytes RAM	-	[202],[204]
MSP430 micro-c	16-Bit	16MHz	512B SRAM, 16 KB Flash	-	[202]
Infincon TRicore	32-Bit		-	-	[202]
Atmel AVR MC	8-Bit	16 MHz	2KB SRAM, 1024 EEPROM	-	[202],[206]

Above Table 3.5 is showing a detailed specification of different IoT devices used for different applications. An analysis of the table suggests the way and approach be used for data analysis at different levels of the IoT/Edge network. The analysis of different IoT devices is based on processing power, delay, battery/energy, bandwidth, and storage/memory. Based on these parameters, the machine learning classification can be utilized at different IoT-Edge network levels. By implementing classification of machine learning algorithms and analysis of the IoT devices based on processing power, storage/memory, bandwidth, battery/energy, and delay of the IoT edge network, the following mathematical model is proposed for data analysis of IoT-Edge network at different levels.

3.5 Mathematical Model

Let the matric Am*n denote the IoT devices and its Corresponding sensors as shown below:

$$A= \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} m*n \qquad (3.1)$$

Where ‘m’ is the number of IoT devices and ‘n’ is the number of sensors included.
Let I_k be the element a_{ij} of matrix A.

$$\text{If } a_{ij} = \begin{cases} 1 & Then\ go\ to\ vector\ S_\alpha \\ 0 & Sensor\ not\ Present \end{cases}$$

Let S_α be the vector from the matrix A, as shown below

$$S_\alpha = \begin{bmatrix} D_1 \\ D_2 \\ D_3 \\ . \\ . \\ . \\ D_\beta \end{bmatrix} \beta^* \tag{3.2}$$

Where β is the number of data Chunks/clusters generated by 'n' sensors.

Let d_j be the j[th] element of vector S_α and each d_j will be a column vector as

$$D_j = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ . \\ . \\ . \\ P_\gamma \end{bmatrix} \gamma^* \tag{3.3}$$

Where γ is the number of packets in the j[th] data cluster and $d_j \leq S_\alpha$ {where d_j is a subset of S_α }

Processing Step 1:

$$\text{Det} \sum_{i=1}^{\gamma} d_i \leq S_k \tag{3.4}$$

If Determinant of $\sum_{i=1}^{\gamma} d_i \leq S_k$then the data will be processed at the device.

If Det $\sum_{i=1}^{\gamma} d_i > S_k$, then the data will not be processed at the device level and hence will be offloaded and will be forwarded to the next level i.e., Edge Level.

Similarly, For the k number of sensors present in an IoT device, the processing step 1 can be written as:

$$\text{Det} \sum_{i=1}^{\gamma} d_i \leq S_{k+1} \tag{3.5}$$

If Det $\sum_{i=1}^{\gamma} d_i \leq S_{k+1}$,then the data will be processed at the k[th] sensor. Therefore, repeat the Steps for all IoT devices. Then the data will be processed by the k[th] sensor and, If Det $\sum_{i=1}^{\gamma} d_i > S_{k+1}$, then the data will not be processed at the device-level rather it will be pushed to the edge level.

3.5.1. Classification based on the Processing Power of each IoT Device

Let P_p is the Processing power of the IoT device and θ_p is the value of effectiveness (VoE) of the P[th] device in terms of Processing Power. And z is the constant value that can be obtained from the data set. As the processing power of any device is directly proportional to the value of effectiveness, we can write the below equation (3.6).

$$\theta_p \propto P_p \tag{3.6}$$

$$\text{And} \qquad \theta_p = ZP_p \tag{3.7}$$

3.5.2. Classification based on the Storage of each IoT Device

Let $S_s$be the storage capacity of the S[th] IoT device. The VoE of that device in terms of storage capacity is given as

$$\theta_s = X/S_s \tag{3.8}$$

Where X is the constant value derived from the dataset.

3.5.3. Value of Effectiveness (VoE)

VoE in terms of processing power and storage is given by

$$\theta_{sp} = t_o P_p/S_s \tag{3.9}$$

Where, $t_o$is the constant value.

3.5.4. Classification based on the Power/Energy of each IoT Device

If Power/Energy of IoT device is taken into consideration, then VoE is given by

$$\theta_e = YE_e \tag{3.10}$$

Where Y is the constant value.

The value of effectiveness (VoE) in terms of power/energy and Processing power is given by:

$$\theta_{ep} = t_1\ P_e\ E_e \tag{3.11}$$

Where t_1 is the constant.

3.5.5. Classification based on the Bandwidth of each IoT Device

The value of effectiveness in terms of Bandwidth is given by

$$\text{And} \qquad \theta_b = VB_b \tag{3.12}$$

Where V is the constant.

The value of effectiveness in terms of processing power and bandwidth is given by

$$\theta_{bp} = t_2 \, P_b \, B_b \tag{3.13}$$

3.5.6. Classification based on the Delay in sending the sensed information from IoT device to Edge of the Network

If delay is taken into consideration, then the value of effectiveness (VoE) is given by

$$\theta_d = \mathrm{K} * 1/\mathrm{D} \tag{3.14}$$

In terms of Processing Power and Delay, The VoE is given by

$$\theta_{dp} = t_3 * P_d/D_d \tag{3.15}$$

The final value of Effectiveness θ_f by combining the above all parameters is given by

$$\theta_f = \mu * P_i E_i B_i / S_i D_i \tag{3.16}$$

Where, μ is the constant value. More the value of θ_f better will be the option of choosing it for the data processing.

The IoT environment's general structure has three levels, i.e., device level, Fog/Edge level, and Cloud Level. As most of the data is processed on the cloud level because of resource availability, and some may be processed at the Fog level due to the technology called Fog Computing. The device-level has very few resources due to various factors such as size, nature of pervasiveness, wearables, etc. Therefore, resource management with data management is the only solution for a better IoT environment Framework. To utilize the device level's available resources and to optimize it for the data processing, a general equation/mathematical model has been proposed for the device level. This model can decide which data packet should be processed at the device's level and which

should be sent to the Fog or cloud for data processing. In the above equation (3.16), processing power, energy/battery, bandwidth is directly proportional to the (θ_f)Value of Effectiveness. Whereas Storage and delay become inversely proportional. More the value of θ_f, better will be the option of choosing it for the data processing.

3.6 Experiment and Results

The available sensors present at different locations around the globe produce a large amount of data. Among those sensors, some of the data has been captured and available at some authenticated websites. For the experiment, data has been captured in real-time from various IoT sensors, and a dataset available online has also been processed. The algorithm required for the classification at the device level needs a manual analysis of the data so that the classification based on the algorithm's various parameters can be done. Two datasets have been utilized for the experiment. The first data set is captured in real-time, and the other is taken from 'data.world', which is free and open to the public. The data set consists of the sensor data that are present around the globe at different locations. The graphs below were generated by using Python and R programming language in windows 10. Figure 3.3 to Figure 3.7 below shows the graph for the humidity level at different times. This data set is from a data sensor at Chicago for weather monitoring. The dataset parameters are water temperature, humidity, turbidity, wave height, wave period, and battery life.

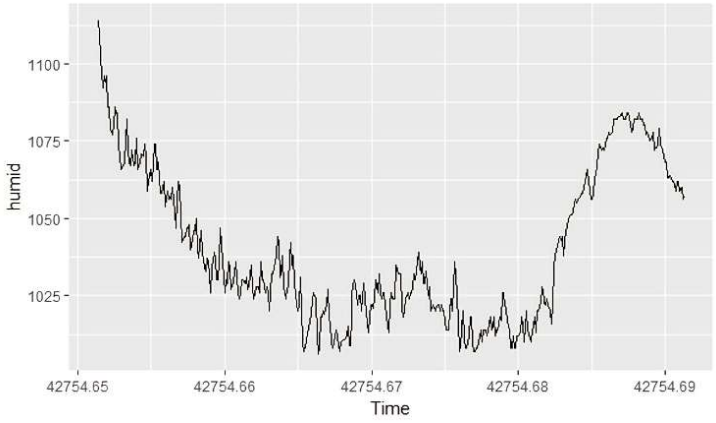


Figure 3.3: Data from Sensor-1.

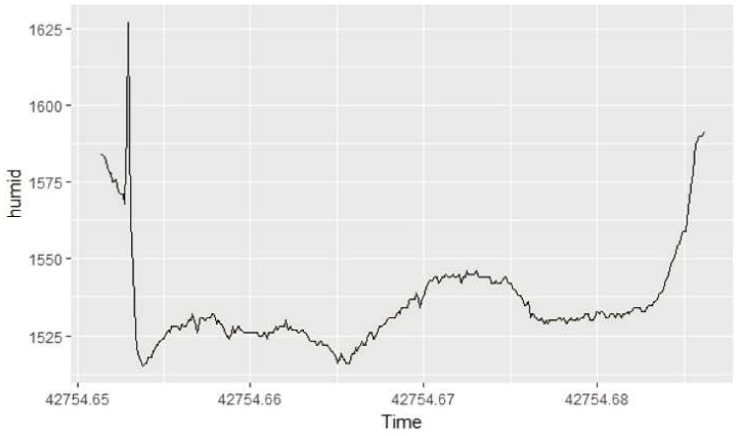


Figure 3.4: Data from Sensor-2.

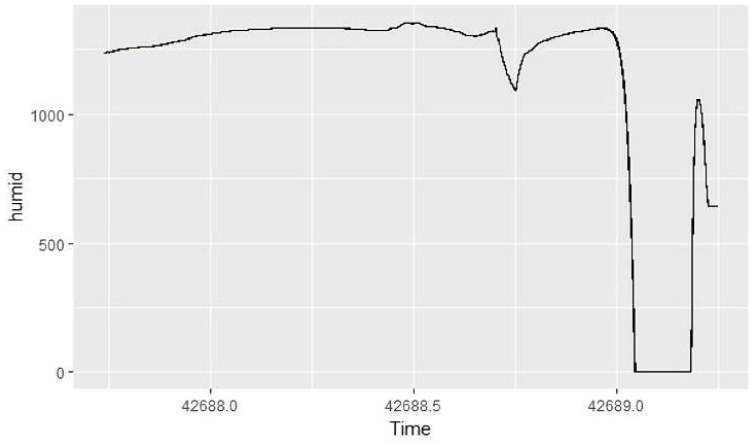


Figure 3.5: Data from Sensor-3.

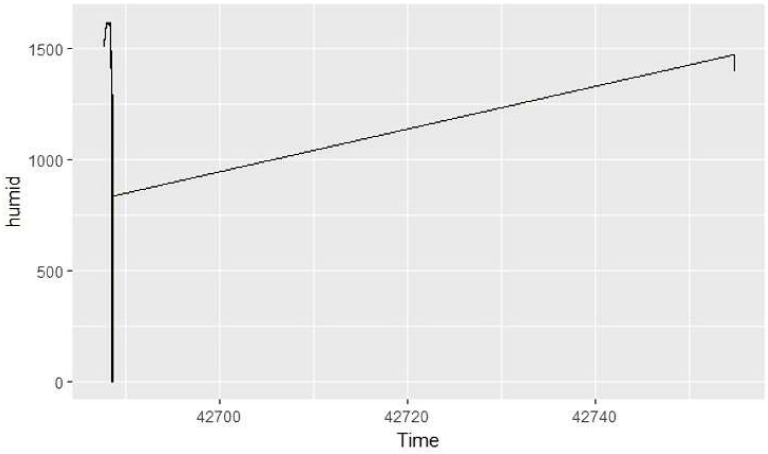


Figure 3.6: Data from Sensor-4.

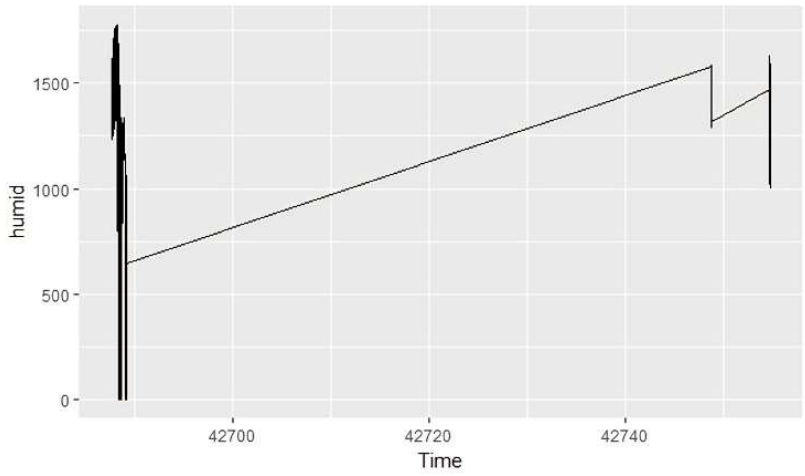


Figure 3.7: Data from Sensor-5.

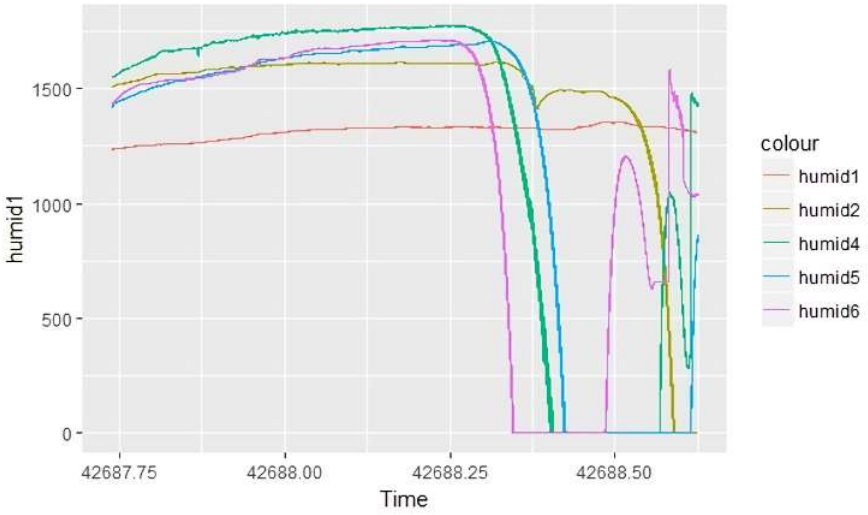


Figure 3.8: The combination of all the sensors from 1 to 5.

Figure 3.8 shows the combination of all the graphs that are generated from five different sensors. It can be seen that the humidity level is lesser for all the sensors at a particular time i.e., 42688.50. This particular time can be considered as the optimal for the IoT Applications such as smart farming in a smart city. Similarly, graphs were plotted for other parameters i.e., pressure, Voltage, light, accel., and temperature. Below are the graphs for some of the parameters.

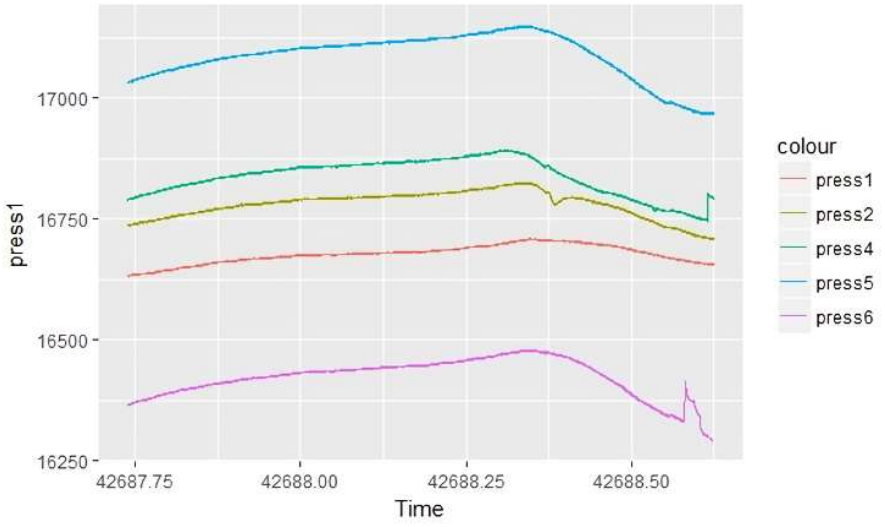


Figure 3.9: The combination of all the pressure sensors.

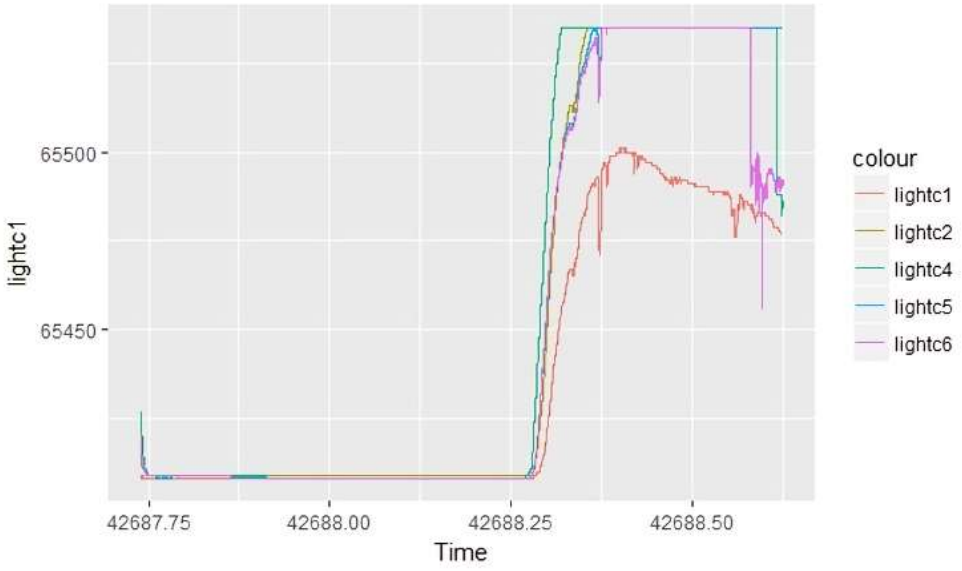


Figure 3.10: The combination of all the Light sensors.

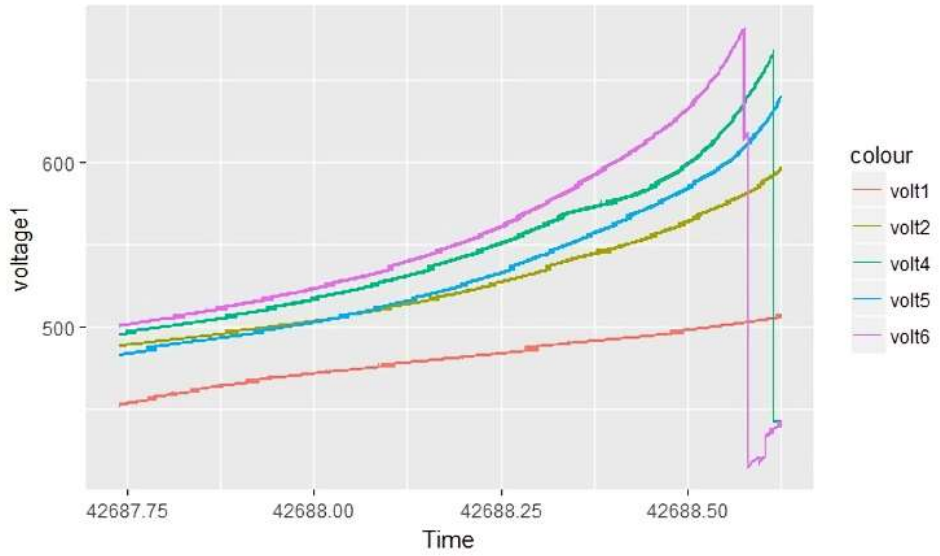


Figure 3.11: The combination of all the Voltage sensors.

3.7 Summary

The IoT data flows through a resource-constrained environment. As most devices are either pervasive or small and portable. Therefore, increasing the hardware capacity will increase the size of the IoT device. Hence the data management is the solution for the Big-IoT data limitations. This chapter proposes a mathematical model that classifies the data at the device-level Fog/Edge level. The data coming from the IoT devices is vast, and if this data is well managed in an IoT environment, it can be useful for various IoT applications. Most IoT devices are lacking in processing power, battery power, storage, and bandwidth. Therefore, it becomes difficult for an IoT device to process a huge amount of data. This problem results in delays caused by these IoT devices. These constraints underscore the intricate balance required for effective IoT operations. Processing power limitations impede the device's ability to swiftly analyze and respond to data, while restricted battery power mandates judicious energy consumption. Insufficient storage capacity hampers the device's ability to store and manage extensive datasets, limiting its potential functionality. Bandwidth limitations further compound the predicament, hindering seamless communication and data exchange between IoT devices and central systems. Consequently, the inherent constraints necessitate innovative approaches to data management, such as edge computing or distributed processing, enabling more efficient data handling closer to the source. Overcoming

these challenges is pivotal for unleashing the full potential of IoT applications, prompting ongoing research and development endeavors to enhance the capabilities of these devices. Addressing the deficiencies in processing power, battery life, storage, and bandwidth is integral to advancing the efficacy and ubiquity of IoT technologies in an increasingly interconnected world.

Graphs were generated for the data sets obtained from different IoT sensors, such as the Voltage sensor, Pressure sensor, Light sensor, and humidity sensor. The results show the difference in the data processing capability by using the proposed model, as it is optimal for saving energy and resources.

CHAPTER 4

DATA MANAGEMENT FRAMEWORK FOR IOT EDGE CLOUD ARCHITECTURE FOR RESOURCE CONSTRAINED IOT APPLICATIONS

4.1 Introduction

The Internet of Things (IoT) is a technology that strongly enters people's reality. All environments are involved, urban, industrial, office, or home. The speed of adoption of the technology has produced a certain disorder and informality in the process. As a consequence, important elements were left aside; one of the most relevant is that of security. In principle, IoT security does not have to be different from security in a typical computer network. In practice, however, there are environmental difficulties that further complicate the security problem. Many IoT devices are computationally limited, preventing the use of several known robust security mechanisms. The large number of devices that can be involved in an IoT network and the exponential increase in the number of interactions exacerbate the problem. The diversity of the equipment used, both in hardware and software, complicates the possibility of generalizing the proposed solutions. There is a wide variety of methods and tools that can be used to undertake the work. In the IoT environment, we have resource constraints in most of the scenarios. Therefore, including the security aspect in the IoT framework has to be in accordance with the processing power, battery life, communication range, and other characteristics of the resource constraints IoT framework.

The proposed Framework distributes the processing load to the last nodes of a digital network (sensors in the case of IoT). The use of computing type poses very attractive advantages for IoT solution providers. For example, they allow to minimize latency and preserve network bandwidth, operate reliably, speeding up decision-making, capture and protect a large number and types of data, and transfer the data to the most appropriate place for processing, with better analysis of local data. Edge computing technologies have been on the rise for several years, but the reach of IoT technology is accelerating its take-off process. As for the factors driving this change, two stand out: Falling prices for peripheral devices with increasing processing power. Centralized infrastructures support the increasing workload. Edge computing technology also arrives at artificial intelligence on devices much more feasible. It allows companies to leverage their data series in real-time rather than working with terabytes of data in central repositories in the world real-time cloud. In the next few years or decades, the technology may evolve to find a balance point between the cloud and more powerful distributed edge devices. Software vendors are developing specific, more robust, and secure infrastructures and security solutions. Providers will begin to incorporate security solutions for peripheral components into their current service offering to prevent data loss, provide network health diagnostics, and protect against threats.

This chapter also discusses different security protocols for the resource-constrained IoT framework. The rest of the chapter is organized as Literature Review, IoT Ecosystem, Secure Architecture, Machine Learning, Result and Analysis and Conclusion.

4.2 Background

In the ecosystem of IoT devices, these are largely unsafe, as they are small and energy-efficient devices; therefore, they also have limited computational resources. This last condition affects a lot when trying to include complex security schemes [207]. At an industrial level, several enterprises are applying IoT, for example, in intelligent transport and agriculture; however, one of the great current home and office automation goals is connected living. This objective requires important advances in the field of IoT, where it is necessary to provide answers to the problems related to the enormous increase in devices that must interact. A particular case in the IoT is that of smart homes

since many times the solutions implemented are ad-hoc by the users themselves, who usually try to reduce costs and efforts as much as possible, which is generally reflected in a minimal and probably non-existent security scheme.

The devices involved require interconnection in a many-to-many scheme. It is necessary to implement an identity management system to ensure the exchange of information that scales appropriately. In this sense, proposes a home system that combines Extensible authentication protocol and datagram transport layer security. Also concerned with identity management and access control, confirm the possibilities of OAuth and make an architectural proposal compatible with services.

Analyze the problem of IoT security [208-209] in the home and highlight how important it is to prevent sensors from indiscriminately capturing and distributing household data. As an example, they present the case of private conversations, which should not be published. Among the possible approaches, they mention a viable alternative that is oriented to services, to balance between centralization and distribution of control. Furthermore, the trend in software and distributed applications seems to be generally directed towards the use of microservices, delving along this line. They highlight the benefits that a microservices architecture, based on TLS / PKI, can have in IoT by lightening development and maintenance tasks, beneficial both for suppliers and distributors and for users while reinforcing interconnection security. Case studies such as that of confirm the possibilities of these techniques in a practical way.

Said work then analyses the possibilities of using SSH and highlights the advantages provided by the data compression issue included in the said protocol, which is especially advantageous when working over HTTP. Although most of the related work's SSL / TLS is the preferred security and encryption mechanism, what proposes is very interesting when analyzing the complications at a practical level in IoT with TLS. Contribute to the IoT environment with a model-managed approach and propose an OAuth-oriented model with a strong UML inclination.

Another interesting architectural and security proposal is presented by, which the SSL certification authorities; an approach of local certification authorities would be used, which more frequently, but also with a lighter process, would authenticate the IoT equipment. This proposal, through transformations, could be adapted to a specific architecture, offering the possibility of customizing it to the required environment. It

can be considered that a middle point between the two proposals would be that of, which uses traditional certificates, but with close authentication, rather than at the node level; they emphasize that this mechanism could be complemented with one of authorization, such as OAuth or similar.

Emphasize OAuth, but above all, with the particularity of concentrating its security architecture on the gateway equipment, where the base station or sink node resides, which is in charge of processing heavy of authenticating, authorizing, and establishing the links between clients and resources. This approach is very relevant when we consider how susceptible those edge devices are linked to edge computing, through which an entire IoT system can be compromised. One of the high-security points in edge systems in IoT is usually related to MQTT (or similar protocols). Several works, such as that of propose improvements over said protocol.

4.3 IoT Ecosystem

This work mainly considered that within the IoT ecosystem, it is necessary to segment the location, scope, and access of the equipment involved in two layers: local or edge and centralized. In the local layer, represented schematically in Figure 4.1, we have those elements that will invariably be installed in the smart home or office, such as sensors, actuators, edge processors such as gateways and brokers, and mobile user devices.

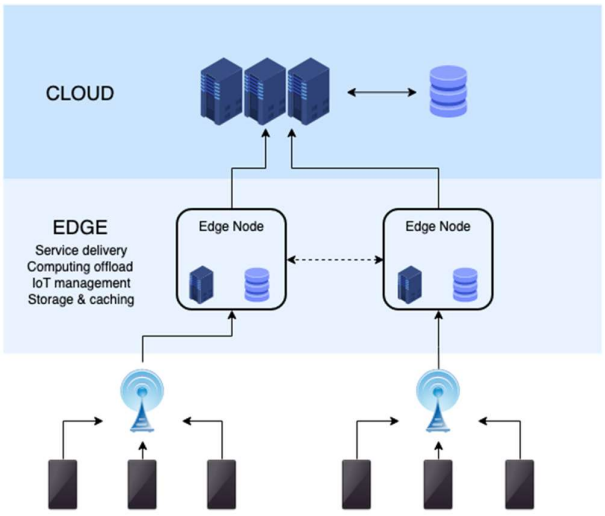


Figure 4.1: Components at the local or edge layer of the IoT system.

Sensors are all equipment capable of capturing physical phenomena, virtual events, or periodic signals. Those sensors with the necessary capacity can communicate directly with the central broker; otherwise, they will interact with equipment in the edge processing subsystem. The sensors will be static when they emit a constant signal that would generally be used by mobile equipment moving in the environment as Bluetooth beacons for positioning. The dynamic sensors will capture measurements of the environment, which will vary depending on the environmental conditions, such as luminescence, temperature, humidity, among others. The actuators will allow interaction with hardware or software generating events or actions. They will receive instructions either directly from the broker or a pre-processor. They are divided into premises located in the intelligent environment, such as light or temperature controllers. They can also be remote, such as those capable of sending instructions, probably over the network, to a distant computer, but controlled from the home or smart offices, such as when it is required to send an SMS, email, or tweet.

Finally, this layer contemplates the pre-processor equipment, which can also be called edge processors or brokers. These computers can be Raspberri Pi or small computers such as tablets. These capture raw data from the sensors to forward it to the centralized broker or actuator when the sensor is incapable.

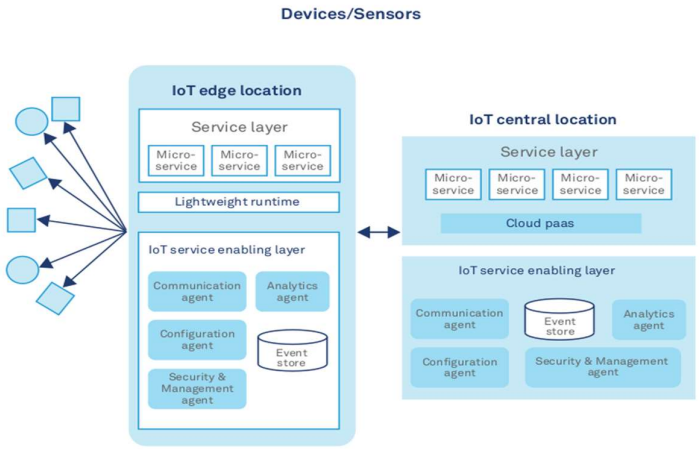


Figure 4.2: Components in the centralized layer of the IoT system.

The information can be sent as received from the sensor, or it can be pre-processed, and this result sent. In the centralized layer, presented in Figure 4.2, the teams in charge of

the general coordination of all the components are necessary, i.e., administration, processing, and persistence. These subsystems are linked to each other and also to the edge layer through a centralized broker [210].

The administration subsystem defines the parameters and configuration of the system presents the web interfaces for the administrator users to interact with the entire system. In the case where central processing is done with multiple machines, it also manages the resulting cluster. The main part of this subsystem is then monitoring, which will allow all types of users to review the relevant information, preferably through dashboards and statistical tables. For all heavy information processing, the corresponding subsystem takes the data collected from the broker and processes it as defined by the specific applications or needs of the IoT system. In general, the processing will be divided depending, above all, on the urgency of the processing, in real-time, which processes the data in a continuous flow, as they arrive from the sensors; in memory, which collects the information in the cluster's memory, depending on the needs, and processes it in small batches; and batch, which interacts in general with the storage system, for processes where the amount of data is greater than what fits in the system's live memory. The information generated by the IoT system is directed to the persistence module, which safeguards the data for later use either in the development of models or in the generation of reports. Several alternatives must be considered, depending on the size of the information and the way it will be accessed. Finally, the entire system, and more specifically the border and centralized layers, must connect and exchange information, which is achieved through a central broker, in charge of managing all the message queues, thus reducing the complexity of the interactions.

4.4 Secure Architecture

The resulting architectural design took into consideration, above all, the need to ensure the exchange of information of all the components of the system, taking care of the speed of calculation at all times. These elements require reconciling characteristics that are often incompatible. For example, more robust cryptography systems may require more computing power than many lightweight devices, such as sensors, provide. The final architecture designed, implemented and tested is the one outlined in Figure 4.3,

which will be described in detail below. In the first place, the components involved will be specified, and then the security functionality in general will be presented.

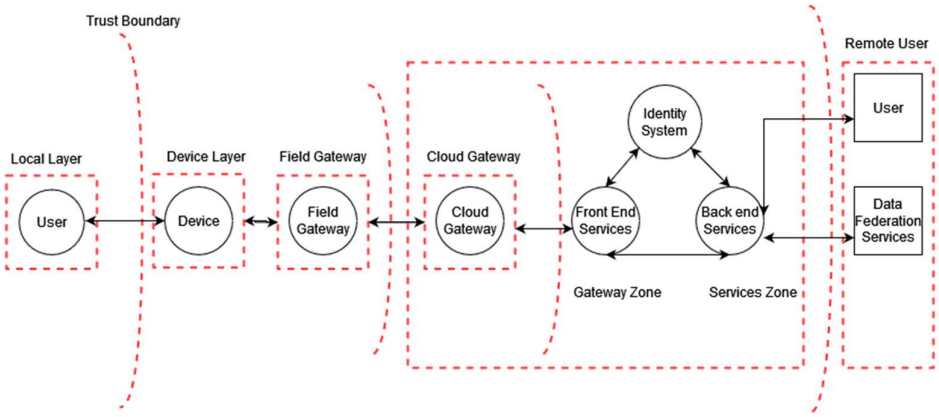


Figure 4.3: Secure Architecture.

4.5 Components

The organizational structure primarily comprises three key components: Registry Service (Registry) that is responsible for managing and maintaining the registry service, User Authentication and Authorization (UAA) that manages the equipment providing authentication services for both clients and users. Moreover, general services and client teams that encompasses all teams providing general services, as well as the various client teams. For the sake of simplicity, the components are referred to in the singular form. However, it is important to note that the architecture allows each category or type of component, especially services, to operate in clusters.

4.5.1 Registry Services

The main task of the registry (REG) is to allow services to register through IP and aliases (service name) and thus make them available to customers, who will connect to the REG to request the information with which they will finally connect to the services of interest. The REG also provides a load balancing service, by detecting that a service is registered in a cluster (multiple computers with the same service). The first point of contact for all other components of the system, whether these services or clients, is the REG, which requires a static IP; all the other components of the system, however, can work with dynamic IPs, via DNS.

4.5.2. User Authentication and Authorization (UAA)

The UAA takes its acronym from the English User Authentication and Authorization. Within the proposed architecture it basically provides us with the authentication service, which works under OAuth2. The UAA stores the data of all the clients in the system, including their roles; With this information, the UAA user services may or may not authorize the use of certain elements. Any component can connect with the UAA to, through its client credentials (user and password), request an access token. In the same way, any component of the system can request the UAA to validate a token received from a third party.

4.5.3. Generic Services and Client Computers

The last category of components accommodates all other services and all clients. In general, these components will interact with each other after having registered/ authenticated in the system with the help of the REG and UAA. The services can be very diverse and it is up to the system administrator to decide which ones to require. However, in the proposed architecture for IoT, there are some that are fundamental, for which they have been implemented in the test system, and they will be mentioned below. To allow interconnectivity and, at the same time, reduce its complexity, the messaging service was implemented, which in the methodology is represented by the central broker. The broker is able to receive and distribute all the messages circulating in the system and basically allows all services and clients to establish a single connection with the broker in general, to deposit messages and retrieve them from one or more queues. This broker can work with any communication protocol, or a combination of several. However, since in the world of IoT the most widespread protocol is the Message Queuing Telemetry Transport (MQTT), that is used in the implementation and presented. Another service implemented for the proof of concept of the architecture is the one related to persistence, as a necessary support to subsequently implement batch processing. For this, a transit service was implemented that takes the information from the broker and transfers it to a Hadoop cluster, where different types of tools of said ecosystem can be used to process the information. One of the cases that was worked on, given the nature of the IoT information, especially that

from the sensors, was that of time series. For this, two data series services were built, thus providing graphing and trend analysis, among others. Regarding customers, all the sensors are considered here, which provide information to the system, the actuators, which react with the environment thanks to the information from the system, and all those devices, mobile or desktop, that allow the user to enter to configure the system, collect processed information, or even act also as sensors and actuators [211].

4.5.4. Security Scheme

The system's security architecture comprises three fundamental scenarios: the basic one, to which all elements must adhere to in their transactions, unless otherwise specified; the lightweight scheme, generally used only when starting a worker process on the system; and the strengthened one, for relationships of trust between services.

4.5.5. Basic Scheme

This is the default scheme that the system components will use in their transactions. This scheme is represented in Figure 4.3 by the dotted line that encompasses the system, and uses a combination of one-way TLS plus OAuth2. Every service must provide its public security certificate (PKI) to clients, who can then validate it with the certificate authority (CA). Also, every client must provide the services with an OAuth access token so that they can validate it with the authentication service. The use of TLS, in the proposed scheme, is especially necessary to be able to encrypt the content of the information that is transmitted. It is used only on the services side to limit as much as possible the overhead that would imply, above all, at the administration level (but also of resources and processing), use it in all the components. The security breach that appears is compensated with the use of OAuth2, through which the clients are in turn validated by the services.

4.5.6. Light Outline

This is a scheme that could be considered insecure, which is why it is provided only for those cases where an access token cannot yet be obtained, or when it is considered redundant to request it. Two cases exist at the moment in the work environment, which implement this scheme. When components enter the system in order to initiate their

transactions, it is generally necessary to have the OAuth token, but since the UAA server IP may have changed, the first step is to contact REG to request the updated IP. For this connection, the client does not yet have the token, which is why it is not possible to work with the basic scheme. The second implementation of this scheme was applied to avoid unnecessary redundant connections and occurs when the service receives the token and must validate it with the UAA. It is for this type of case that the lightweight scheme comes into play. The service provides its PKI with which the communication is encrypted, but the client is not obliged to validate it (although it is recommended that they do so), the service provides an "insecure" access point for the client, which does not require the token. This is the scheme provided by REG exclusively to be able to deliver the UAA data.

4.5.7. Strengthened Scheme

Similar to the problem worked in the light scheme, sometimes two services require interconnection, but at least one of them (who acts as a client) is unable to obtain its access token. When dealing with services, it is not convenient to open an insecure channel as is done in the lightweight scheme. In order to maintain the security standard, then, it was decided to implement a two-way TLS scheme, which is possible, without incurring greater overhead, since, being services, they already have their PKI anyway. Additionally, in general, the services will be executed in equipment with greater processing capacity. This implementation also requires a dedicated channel to be able to execute this type of validation and the example is given by the communication between the REG and the UAA. The UAA is the one that provides the access tokens and therefore should validate itself which would generate a security hole. The REG, then, opens a dedicated channel so that a UAA service can register in this way at all times. By connecting the UAA with the REG, they exchange their respective PKIs, mutually validating each other over TLS, without reducing the system's security standard.

4.5.8 Functionality

Returning to Figure 4.3, the dotted line represents the scope of the basic security scheme, which encompasses the entire system. Internally, the numbers 1, 2 and 3 can

be seen, circled, which indicate the recommended starting order to guarantee the fluidity of the service. In practice, at least the services have in their base library the functionality to retry the connection when this starting order is not respected. However, this can be subject to unnecessary delay.

In the first place, the registration server, REG, is started, which will provide a central access point for the acquisition of contact information for the other services: every service will register in the REG its respective IP and its alias (service name) and every client will search here, by aliases, for the IP of the service required to be able to connect with it. REG offers three access points, each of which must handle a different security mode: the first, lightweight, allows any client to obtain the UAA's IP without any additional security; the second mode, strengthened, allows the connection of the UAA using two-way TLS; the last, basic one, which requires OAuth2, allows clients to request information from services, and from services to record their contact information. Second, an Authentication Server (UAA) is started, which will provide OAuth2 credentials to clients. The enhanced security mechanism, with two-way TLS validation, is used between the UAA and the REG. The UAA connects with the REG, as well as any other service, to give it its IP and aliases and thus be available to the entire system. Once these two services, REG and UAA, are online, all the rest of the components, services and clients can start their work. Finally, then, as point 3, any other component, be it this service or client, will proceed as follows: first, using the lightweight security scheme, they will connect with the REG to request the IP of the UAA; They establish the connection with the UAA and request the access token, using their client credentials. With the access token in hand, the basic security scheme can already be used and, in the case of services, they will be registered with the REG, delivering IP and aliases, to wait for client requests, or act as a client from another service, as needed. In the case of a client, the next step is to use a service, where the basic security scheme will be applied; it connects to the REG and by means of an alias it requests the IP of the service of interest, and then connects with said service.

4.6 Machine Learning

To perform the object detection, the approach involves utilizing the K-NN algorithm, which serves the purpose of using a database where data points are categorized into

different classes to predict the classification of a new sample point. It is an ideal choice for classification studies when little or no prior knowledge is available about the distribution of the data. The K-NN algorithm operates based on the similarity of characteristics. The extent to which the characteristics of a new sample point resemble our training set determines how we classify it. In the following section, the working of the algorithm will be presented. In Figure 4.4, the aim is to classify the test sample (represented by the green circle) into either the first class of blue squares or the second class of red triangles. The K-NN algorithm is used to make this determination. If we set the value of k to 3 (as shown by the solid line circle), the test sample is assigned to the second class since there are two triangles and only one square within the inner circle. Conversely, if we set k to 5 (as shown by the dotted line circle), the test sample would be assigned to the first class because there are three squares and only two triangles within the outer circle.

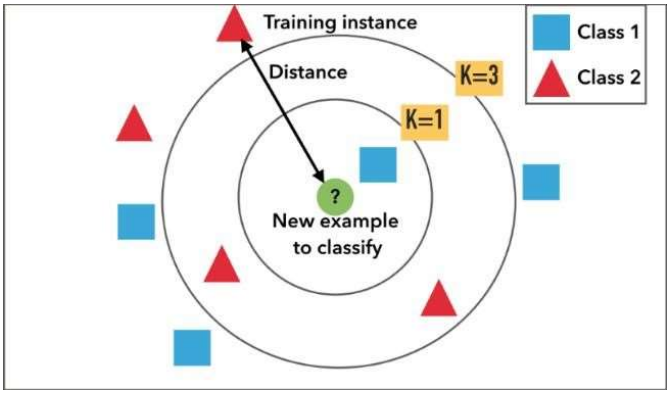


Figure 4.4: Operation of the Algorithm. [27]

The K-NN algorithm can be applied to both classification and regression tasks. In classification, K-NN predicts a class membership or a discrete value. To classify an object, the algorithm assigns it to the most common class among its k nearest neighbors, determined by majority vote. In regression, K-NN predicts continuous values. The algorithm calculates the output value as the average (or median) of the values of its k closest neighbors. Additional attributes of K-NN include:

- K-NN retains the entire training dataset, which it utilizes as a proxy.
- K-NN doesn't acquire any models.

- K-NN conducts real-time predictions by assessing the similarity between an input sample and each instance in the training dataset.

4.7 System Model

The MLADCF has three layers, i.e., Device Layer, Edge-Fog Layer, and Cloud Layer. The device layer is placed at the bottom of the MLADCF. The Edge-Fog layer is in the middle, and finally, the Cloud Layer is placed at the top of the MLADCF. The whole IoT environment is squeezed in the above 3-level framework as shown in Figure 4.5. The device layer at the bottom of the MLADCF is the densest layer among the three layers. Usually, millions of active nodes present are sensing data from the environment.

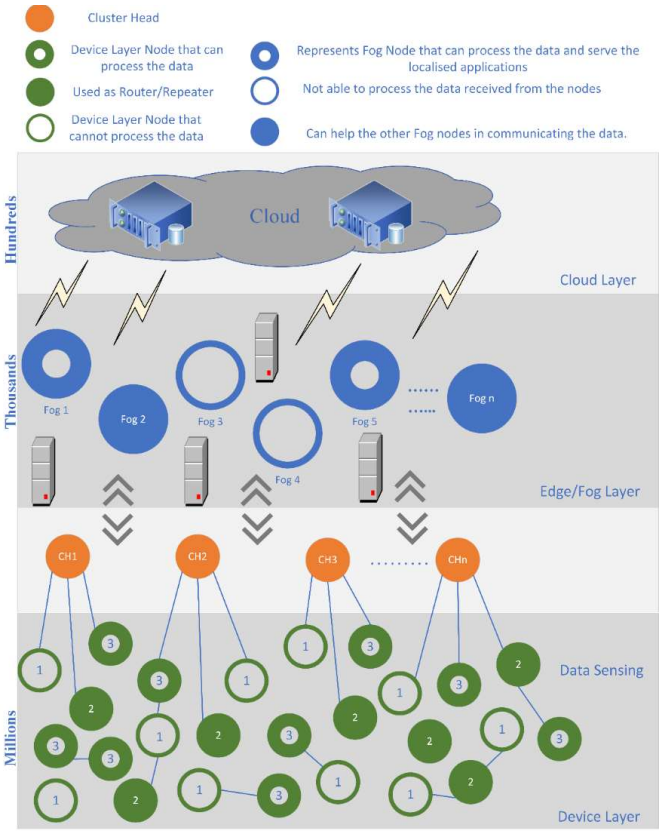


Figure 4.5: Machine Learning Analytics Based Data Classification Framework (MLADCF).

The nodes in this layer vary in different aspects. There are different sensors in today's world, such as video sensors, motion detectors, smoke detectors, humidity sensors, temperature sensors, proximity sensors, pressure sensors, accelerometers, level sensors,

infrared sensors, gas sensors, optical sensors etc. All these Nodes collect data of different data types. In order to perform well in an environment, every datatype needs additional data storage, processing power, bandwidth, power/energy, etc. These device layer nodes have been categorized into the following three categories.

- Device layer nodes have the capability of processing the data;
- Nodes that cannot process the data;
- Nodes used as routers/repeaters.

The nodes present at the device level collect the data from the environment. These nodes can be of different types, such as capturing video data, audio data, textual data, etc. The cluster heads connect the data from these nodes. Traditionally the sensed data is captured/sensed followed by the filtration or compression process and then forwarded to the next level. However, the proposed MLADCF is designed to classify the data at the device level based on the parameters such as storage, processing power, bandwidth, and battery/power. The MLADCF will classify the data as the device level so that if a particular data packet can be processed at the device node, it will not be pushed to the Edge/Fog level. Moreover, if the data packet cannot be processed at the device level, it will be pushed to the Edge/Fog level. The middle layer is known as the Fog/Edge layer. This layer contains the Fog nodes. These nodes are lesser in number in comparison to the device level. This layer has better resources in comparison to the device level. These nodes have sufficient battery/power, storage capacity and processing power for processing the data coming from the device level. However, these nodes also vary in terms of storage, processing power, battery, etc. Likewise, device level here, some nodes are capable of processing the data, and some are not capable of processing a huge amount of data coming from the millions of sensors at the device level. Finally, there is the cloud level; this layer has all the required resources for data processing, data analytics, big data, decision-making, etc. Cloud provides services to various levels of the IoT environment.

The Machine Learning Analytics Based Data Classification Framework (MLADCF) is a meticulously crafted system with a core objective: efficient data classification. In its intricate design, MLADCF orchestrates a dynamic process where data undergoes a meticulous sorting mechanism. The nodes within the system possess a specialized capability to process specific types of data packets. This targeted approach ensures that

each node, equipped with tailored processing capabilities, efficiently handles the data assigned to it. The underlying principle is to enhance processing speed and optimize resource utilization by assigning tasks based on node capabilities.

In the complex choreography of data processing, MLADCF acts as a conductor, directing the flow of information through the nodes. The synergy between machine learning algorithms and data processing capabilities results in a finely tuned orchestration. However, not all data fits the predefined processing criteria of the nodes. MLADCF accounts for this by intelligently diverting unprocessable data to the upper layer. This strategic decision prevents bottlenecks and ensures that the system maintains a fluid and adaptive functionality. In essence, MLADCF leverages its design intricacies to create a responsive and agile framework that optimizes data processing while gracefully handling the nuances of diverse data types.

The proposed MLADCF is designed so that the data will be classified so that the nodes capable of processing the data will process the data packets, and the rest of the data will be pushed to the upper layer. This method will be incorporated into the device level and the Edge/Fog layer. This data management method will minimize the delay and utilize the resources efficiently. Furthermore, it will optimize the resource utilization for the coming IoT infrastructure in the future. The final value of Effectiveness θ_f by combining the above all parameters is given by equation (4.1).

$$\theta_f = \mu * P_i E_i S_i \tag{4.1}$$

Where, μ is the constant value. More the value of θ_f better will be the option of choosing it for the data processing. This paradigm provides the means by allowing data to be obtained from billions of devices that can sense, send and make decisions for the problems identified in IoT environment. From the state of variables, the structured, unstructured, or semi-structured records can be generated, which have the potential to generate changes from the information and knowledge that can be obtained when the processing mention the challenge to achieving the above due to the heterogeneity and discovery of the sensors, for which they propose solutions such as the creation of a middleware, that allows the connection between the sensors and the cloud to be made transparently. The MLADCF distributes the processing load to the last nodes of a digital

network (sensors in the case of IoT). The use of computing type poses very attractive advantages for IoT solution providers. For example, they allow to minimize latency and preserve network bandwidth, operate reliably, speed up decision-making, capture and protect a large number and types of data, and transfer the data to the most appropriate place for processing, with better analysis of local data. Edge computing technologies have been on the rise for several years, but the reach of IoT technology is accelerating its take-off process. As for the factors driving this change, two stand out: Falling prices for peripheral devices with increasing processing power. Centralized infrastructures support the increasing workload. Edge computing technology also arrives at artificial intelligence on devices much more feasible. It allows companies to leverage their data series in real-time rather than working with terabytes of data in central repositories in the world real-time cloud. In the next few years or decades, the technology may evolve to find a balance point between the cloud and more powerful distributed edge devices. Software vendors develop specific, more robust, and secure infrastructures and security solutions. Providers will begin to incorporate security solutions for peripheral components into their current service offering to prevent data loss, provide network health diagnostics, and protect against threats.

4.8 Deployment and Testing

The implementation decision was mainly that each of the components can be executed on a variety of computers and with the least interdependence, for which we proceeded to work on a microservices architecture that allows their deployment either as independent processes, or within a containerization structure, such as Docker. For desktop services and clients Java was used with Spring Boot in general. For the central messaging service, it was decided to work using the MQTT protocol and for the development of the broker the Moquette library was taken as a base, to which it was modified to add support for OAuth2 mainly. An HDFS cluster was used as a basis for persistence services time series services were built on the local network, which, according to the current interests of the thesis, were the most suitable for processing the data coming from the sensors.

4.9 Algorithm for Proposed Work

Suppose we have n data points, denoted as (X_i, C_i), where i ranges from 1 to n. Here, X_i represents the feature values, while C_i represents the corresponding labels for X_i. Assuming that the number of available classes is c, the label C_i can take on values from 1 to c for all values of i. Let x be a data point for which the label is unknown, and we want to determine its label class using the k-nearest neighbor algorithm.

- Calculate the Euclidean distance "d (x, x_i)" between x and each training instance x_i, where i ranges from 1 to n.
- Arrange the n Euclidean distances in ascending order.
- Choose a positive integer k and select the first k distances from the sorted list.
- Find the corresponding k points for these k distances.
- Let ki denote the number of points belonging to the i^{th} class among the k points, where i ranges from 1 to c.
- If $k_i > k_j$ for all j ≠ i, then assign x to the class i.

4.10 Setup

It interacted with OpenTSDB and Prometheus as for customers, it was decided to build generic libraries for different types of systems, which facilitate the process of developing specific applications. A Java client library was developed, one for Android mobiles, one for Arduino MKR and another for ESP32, the latter three based on Eclipse Paho. The Arduino libraries were intended exclusively for use in sensor and actuator controllers. System tests were conducted in a controlled office environment. The main equipment was an RPi 3B + that served as a Wi-Fi Gateway, providing DNS and NTP services, among others. MKR 1010 and ESP32 controllers were used simultaneously, which permanently received information from temperature, humidity and noise sensors, such as the DHT11 and the KY038. The RPi3B + also hosted the REG, UAA and messaging broker MQTT services. Persistence services over HDFS as well as TSDB were installed on Linux on an i5-4210U with 8 GB of RAM. In this last equipment, generic services were also installed for sending and receiving messages, with which the fluidity of the interaction was verified.

As a mobile client, a N9005 was used, which had two functions: a dummy sensor, sending a large number of random numbers to the system, and a light actuator, warning

each time that the measurements of the real sensors exceeded certain levels parameterized in the application. Figure 4.6 is presents one of the setup of the testing system.

In the experimentation related to the security tests, it was decided to assume that a possible attacker was already connected to the local network and that he was, potentially, capable of making any request and capturing all the traffic. In the first test, Zap was used to perform both active and passive scanning, and it was verified that security was maintained (encrypted traffic), except in the case (documented in the architecture) of the lightweight scheme.

Figure 4.6: Setup.

In Figure 4.6 from left to right, a cluster of 3 Raspberry Pi is observed running all the services on Docker; an ESP8266 monitoring noise level with a KY038; a MKR1010 monitoring room temperature with a DHT11; an N9005 injecting random messages, and a desktop monitoring all services.

4.11 Results and Analysis

The process of including complex security schemes in lightweight IoT devices is not trivial as stated by Khan and in this work this statement is agreed. Two notable cases were that the handling of TLS with auto-generated certificates for MKR 1010 required regenerating all the firmware to include the CA, and that it could not be carried out on ESP8266 controllers due to the unavailability, in practice, of open-source libraries. This is sufficiently complete to guarantee the expected level of security. The approach adopted generally takes up the warnings made by Lin and Bergmann tries to provide a

sufficiently complete and simple system so that with minimal technical support it could be implemented at home, and then managed directly by the user.

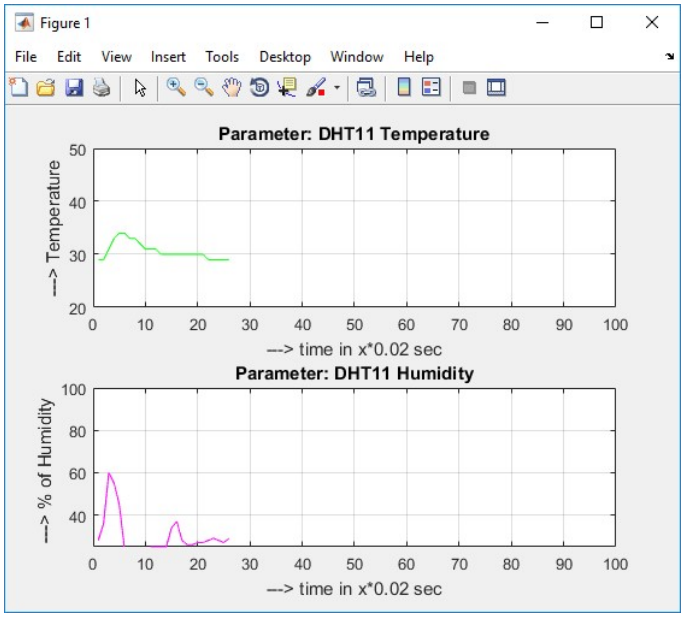


Figure 4.7: Measurement of Effect of change in Temp/ Humidity in DTH11 Sensor.

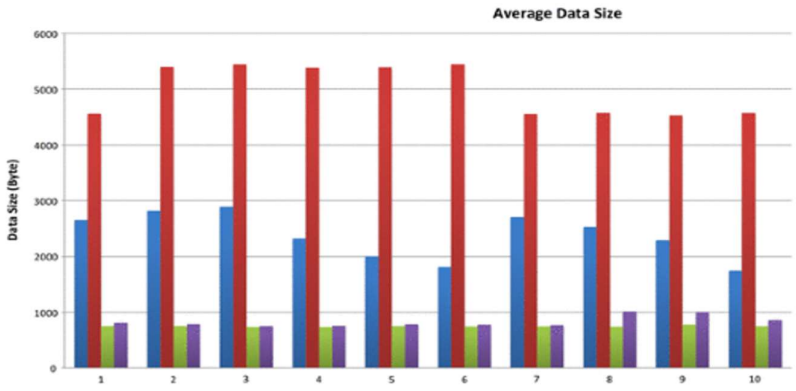


Figure 4.8: Measurement of Effect of change in Temp/ Humidity in DTH11 Sensor.

This, of course, has limitations given by the great variety of types of users that may wish to be included in the IoT. However, the tests carried out suggest that the heart of the prototype would allow providing this, if some facilities at the user interface, equipment and installer's level can be included. This work confirms in relation to the problem of identity scaling and the benefit that can be obtained both with OAuth and

with a service-based approach. Delegating authentication to a single point, then relying on temporary credentials, as OAuth2 allows, keeps the identity infrastructure light.

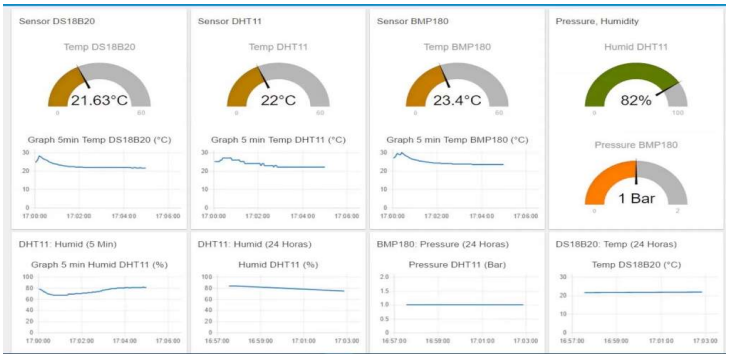


Figure 4.9: Temp/ Humidity in DTH11 Sensor comparative Analysis.

It is precisely this approach to services / microservices that allows working with TLS to not be too demanding at the maintenance level, as highlighted by Díaz-Sánchez et al. This section recognizes the importance that at the security level should be given to the edge equipment; it especially reinforces the gateway equipment in order to later implement improvements in the MQTT protocol, in the style of what was done by Singh et al. but mainly including the use of OAuth2 and TLS.

4.12 Summary

This chapter focusses on the current need for data management in resource constrained IoT applications, especially those linked to a home context, and to do so without impairing the user's freedom of access to collect information and to modify the configurations of their system. The proposed Framework is centered around the management of data at various levels of the IoT architecture. In this chapter, the MLADCF (Machine Learning Analytic Based Data Classification Framework) has been proposed. The main contributions of this work include the implementation of a data management architecture for agriculture. In this regard, IoT sensors were utilized for data collection, enabling the creation of a real-time IoT network. As discussed at various sections of this chapter, the response time of the IoT device is mainly dependent on data generation, resource availability and data management at different levels of the framework. This need, as it turned out, starts among other things from the relative

informality of the IoT, especially in a smart home environment. It began with a methodological conception that stratifies the environment in layers: the one most closely related to the interaction with the space by collecting information and executing actions to modify the microenvironment, and the layer of centralized processing and analysis of the information. The device layer and edge layer were interrelated with a centralized connection that unifies them while also maintaining their light coupling.

CHAPTER 5

DESIGN AND IMPLEMENTATION OF MACHINE LEARNING ALGORITHMS FOR RESOURCE OPTIMIZATION IN IOT

5.1 Introduction

The Internet is one of the tools of the 21st century that has advanced the most over time. IoT is considered a base technology for implementing Smart city projects in the world. The growth in the number of connected devices is exponential. In 2015, the number of connected devices was five billion; in 2017, it exceeded eight billion. We currently find ourselves with a forecast of between thirty and forty billion for 2023. This gives us an idea of the large number of data to be processed by these devices and companies' involvement in obtaining the greatest benefit. The Internet of Things has been developed with greater force in factories under the name of Industrial Internet of Things (IIoT, Industrial Internet of Things), compared to the Consumer IoT developed outside of them. The IoT is based on RFID (Radio Frequency Identification) technology. It allows each product or device to be assigned a code that serves as a unique identifier. The objects connected to the IoT usually carry sensors capable of detecting real-world conditions and actuators with which they can execute actions. In short, the IoT focuses information and decision-making on each device and then shares that data in the network through the Internet. Moreover, the combination of IoT and Artificial intelligence has also been important for decision-making, forecasting, smart healthcare, smart city, smart agriculture, smart industry, and much more [211].

Machine Learning is used to apply algorithms, which, thanks to statistical techniques, allows machines to learn from experience. Machine Learning has four main variants. The machine is trained with labeled data as an example in supervised learning, thus providing a learning guide. On the other hand, in unsupervised learning, the data is not labeled, so the machine looks for similarities and forms groups based on them. Reinforcement learning is based on trial and error. The machine learns to carry out a task according to the consequences of its past decisions. Finally, Deep Learning is inspired by the functioning of the human brain to extract capabilities such as vision, pattern recognition, or motor-sensory control through artificial neural networks. People's lives are easier thanks to Artificial Intelligence. It frees up repetitive or complex tasks quickly, improves the quality of transport and safety, enables market forecasts, and pursues a good user experience through chatbots and recommendation systems, among others. In this chapter, the technical aspects of the Machine Learning tool and its applications, such as Smart Cities, are explained. A smart city's development is based on sustainability and technology to achieve efficient, innovative, safe, and high-quality public services. Thus, an urban complex is qualified as intelligent based on the investments made in education, energy infrastructures, communication, and transport technologies, ensuring a high quality of life, sustainable economic-environmental development, and good use of the citizens' time. Tokyo, London, Singapore, Barcelona, Amsterdam, Oslo, and New York are the main examples of smart cities for carrying out numerous smart projects. However, smaller cities such as Srinagar are also putting into practice proposals to boost tourism through innovation applied to public and ornamental lighting. From improving traffic flow to constructing houses equipped with energy-saving systems through efficient lighting using solar energy [212].

5.1.1 Cloud of Things

The Cloud of Things concept provides mechanisms to bring data from IoT devices to the cloud. This integration provides scalability, reliability, flexibility, and security to store IoT data securely, efficiently, and economically. Various technological proposals from companies' leaders in cloud computing, open-source projects, and projects research promoted by the European Commission were developed to enhance this

integration. Tech companies have sought to exploit the infrastructure they have currently to support the inclusion of IoT in the cloud. Companies such as Amazon, Microsoft, Google, and IBM promote device support intelligence in their cloud computing infrastructure through platforms and different services to meet the needs of connecting the physical world with the digital world. Amazon allows the connection of IoT devices with your cloud using the Amazon Web Services (AWS) IoT Core, which provides a secure connection of devices IoT, routing and processing, control, and interactivity with IoT devices. The messages received by the AWS IoT Core can be routed to other platforms of the Amazon product portfolio such as AWS S3, QuickSight etc. [213].

Cloud computing is a mature technology that offers increased complex, large-scale computing capabilities via the Internet. Cloud computing provides easy access and secure computing resources to individuals and businesses on demand. In addition, this technology provides technical advantages such as energy efficiency, resource optimization, elasticity, or isolation of environments execution. These benefits have been possible due to the technological pillars fundamentals on which cloud computing is based, virtualization, and distributed computing. As a result, cloud computing introduces flexibility, scalability, high availability, and security features, converting it into this technology with more interest for handling the Big Data generated by IoT.

Virtualization is a technology that allows creating computational resources virtual or logical based on the abstraction of physical resources. As a result, computational resources are more efficient and flexible. The most widely used virtualization techniques are virtual machines and containers. The Virtualization is at the hardware level managed by a hypervisor. It is the operating system capable of being installed. It runs using resources virtual computational, i.e., memory, processor, hard drive, etc. provided by the operating system. The hypervisor is in charge of abstracting physical resources, creating a set of customizable computational resources, and presenting these resources for virtual machines to use. As a result, each virtual machine is isolated and independent of the others to provide some level of security and run the applications necessary to its operation without affecting the other virtual machines. Examples of hypervisors are Xen, VMware, and Kernel-based Virtual Machine (KVM).

Container is a lightweight operating system executed using the physical computational resources provided by the system operating (host). Virtualization is at the operating system level managed by a virtualization layer (container engine). The virtualization layer partitions host resources to provide virtual network interfaces and independent spaces for container processes. In this way, each container can run applications isolated from the other. Examples of virtualization layers are Docker, Linux Container (LXC), and OpenVZ.

Infrastructure as a Service (IaaS) provides infrastructure resources through virtual machines. User can configure system operating, software, and some network and security settings. Platform as a Service (PaaS): provides platform layer resources giving support to operating systems. The configurations allowed for the user are more restrictive and limited to the configurations and deployment of the applications.

Software as a Service (SaaS) provides applications to end users. The consumer is only able to configure the parameters that allow the application. IoT has found in cloud computing an opportunity to solve the limited characteristics of their devices to store, process and analyze the data. The integration between IoT and cloud computing was imminent because of its many advantages. This integration has led to the generation of new business models such as Sensing as a Service (SenaaS). The SenaaS model proposes that the data collected by sensors be stored, sold or traded by the owner of the data. For example, a temperature sensor in a house can collect data and store this information in the cloud, and said information from the temperature sensor is published so that a manufacturer of thermostat equipment can improve the performance of your products. In addition, this integration gave rise to the creation of the concept of Cloud of Things [214].

5.1.2 Limitations of cloud computing

Despite the many advantages of managing data in the cloud computing infrastructure, this has some limitations that must be faced for the optimal management of Big Data generated by IoT.

Latency: is the most critical of the Big Data analysis limitations. End-to-end latency is critical in delay-sensitive applications, such as remote health monitoring of critically ill patients. Despite that the cloud can provide sufficient resources for optimal

performance in real-time processing, latency depends on other factors such as communication links and devices
network intermediates.

Bandwidth occupation: large volumes of data that must be transmitted to the cloud can cause saturation at the level network. Video and audio data generated by IoT devices during monitoring mainly cause this saturation. Although there are compression mechanisms for reduce the amount of data of the type video and audio in IoT, which is still considered a problem.

Geographic Centralization: Computational resources are located in a centralized location that can be different from the geographic location of IoT devices. As a result, IoT devices can be geographically located at large distances with the location of the servers that make up cloud computing which can increase latency and decrease performance QoS.

Low QoS: Geographical distribution of data centers that make up cloud computing can cause low QoS and low perception of service or quality of experience. Furthermore, the communication between IoT devices and the cloud depends on an unreliable network, which can fail, causing a reduction in the QoS.

5.1.3 Fog Computing

The concepts of Fog Computing and Edge Computing emerge as proposals for efficient technological solutions to face the limitations of cloud computing and efficiently handle the Big Data generated by IoT. After the operation of both, the main idea is to bring the storage capacities closer and process to the source that generates the data (IoT devices, mobile phones, cameras, etc.). However, the two paradigms are usually seen as analogs, these present small characteristics that differentiate them each. Below are the characteristics of each of these paradigms are detailed:

5.1.4 Edge Computing

It is also known as Mobile Edge Computing (MEC) and has its origin in mobile communications. MEC appears under the necessity of smart mobile devices (smartphones) users to improve their access to applications and services on the mobile network. At MEC, part of the Computational resources of cloud computing is brought

to the brink of mobile networks to meet these needs through servers, micro data centers, and cloudlets. The cloudlets were a previous proposal to MEC and are framed in what is known, such as Mobile Cloud Computing. From an IoT point of view, edge computing refers to devices with interconnection capabilities of the network and some computational capabilities at the edge of the IoT network. The edge of the network in an IoT architecture comprises IoT gateways, smartphones, and some embedded devices with limited computing features.

Unlike edge computing, this has its origin in IoT to cover IoT devices' storage and processing needs. For this reason, Fog Computing is more associated with IoT than edge computing associated more with mobile communications. Fog Computing comprises nodes Fog between IoT devices and the cloud. The Fog nodes are devices that offer computing capabilities, storage, and network services such as IoT gateways, routers, and smartphones. The wide range of devices for Fog Computing makes it possible to create geographically distributed services. It is known as a highly distributed and heterogeneous technology. In current literature, Fog Computing is considered an extension of cloud computing. Because it has the objective of extending data processing, storage, and analysis capacities. This extension is known as Cloud Continuum or IoT-Fog-Cloud Continuum, including Edge-Cloud Continuum. Since some devices used as Fog Nodes are at the edge of the network, edge computing can be a case particular to Fog Computing.

5.1.5 IoT Architecture with Fog Computing

The Internet of Things World Forum (IoTWF) 2014 proposed an extension of the reference architecture of 3-layer IoT in a 7-layer architecture, which includes Fog Computing. Fog nodes have limited computational resources and a capacity that varies, so it is a huge challenge to develop a platform and a service model in this layer. Currently, lightweight virtualization technologies are being used for their efficiency in the use of available resources. Mainly, the technology based on service containers emerge as the best option to be adapted at the Fog nodes. This technology offers virtualization and isolation of applications or services at the operating system level without the need for virtualization of hardware and drivers.

Fog computing provides important benefits to manage the Big Data generated by IoT. The processing performed at the edge of the network, also known as edge analytics, reduces the high data redundancy IoT. Pre-processing data tasks such as filtering, cleaning data, extraction of variables, comprehension, and reduction techniques of data dimensions at the edge have reduced the volume of data sent to the cloud and subsequently stored. In the same way, techniques to merge several data streams into one or send the data only when anomalies are detected reduce the frequency with which IoT data is generated. In addition, handling interoperability technical, semantic, and syntactic in Fog nodes (gateway IoT) reduces IoT heterogeneity. In short, Fog Computing reduces latency, bandwidth consumption, and storage space. It provides important advantages such as improving QoS, supporting mobility, interoperability, and device location awareness [215].

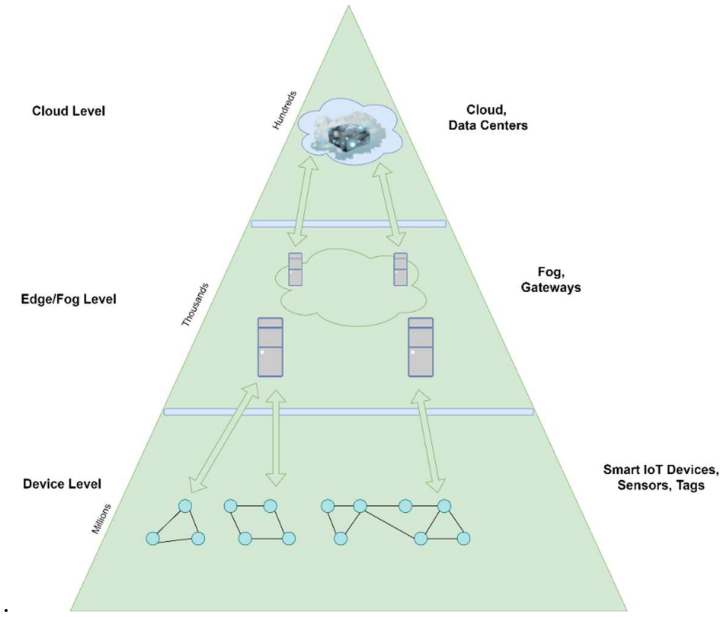


Figure 5.1: IoT Architecture with Fog Computing.

With this vision of using light virtualization technologies, platforms such as ParaDrop or Apache Edgent have been developed to provide data processing at the edge. Apache Edgent is a platform designed for real-time data processing on devices situated at the edge. It is primarily utilized for integration with the IBM Watson IoT platform in the Cloud. However, the development of more platforms adapted for use in Fog Computing. A clear example of using technology containers is presented, where a

gateway is implemented IoT using micro-services based on container virtualization Docker to optimize resources [215].

5.2 Background

Human beings rely on pattern recognition to comprehend their surroundings and anticipate behavior. The surge in user-system interactions generates vast amounts of data, presenting an opportunity to extract patterns that enable accurate predictions. Although often used interchangeably, data science, data mining, machine learning, and artificial intelligence differ from one another. This chapter elucidates these distinctions and provides an introduction to machine learning fundamentals. Moreover, it explores various learning types, each suitable for specific problem-solving scenarios, along with examples of algorithms employed in each case. Machine learning is an automated approach to identifying significant patterns within datasets. It has become ubiquitous in tasks involving extensive data analysis, permeating our daily lives through technologies like email filters, recommendation systems, facial recognition, smartphone speech detection, weather forecasting, and traffic inquiries. Its applications extend to diverse domains, including medicine, marketing, logistics, and industrial equipment maintenance. Due to the complexity of these applications, it is impractical for humans to manually program specific instructions for each task. Instead, computers must possess the ability to learn from experience and adapt to novel situations. Different forms of learning exist, such as supervised learning, unsupervised learning, reinforcement learning, and deep learning, each characterized by unique attributes and algorithms that are employed based on the specific problem at hand.

5.2.1 The Machine Learning

Learning in a Machine Learning system consists of adjusting the parameters of a model based on the data received. This data set is called a data set, which contains both independent and dependent variables. The independent variables (features) are those columns of the data set used by the algorithm to generate a model that best predicts the dependent variables. On the other hand, the dependent variables (labels) are the columns of the data set resulting from a correlation between independent variables, so they must be predicted by the implemented model. Classical programming is

understood as a set of rules designed by a person to obtain responses that meet these rules from input data. On the other hand, Machine Learning is in charge of obtaining the most effective rules that relate the input data with the answers we hope to obtain through learning. The advantage of Machine Learning is that these rules can be applied to different input data to produce answers that have been automatically generated by what the system learned and not by instructions generated by a human. For example, if we wanted to design a Machine Learning algorithm to predict the price of a house in a certain city, the dependent variable would be the price of the house. In contrast, the independent variables would be all those factors that influence the price of housing, such as the surface area, the number of rooms, the distance to the city center, etc.

The model must be sufficiently adjusted to the input data, but it must also have enough consistency to give a good result when different data are introduced. To do this, the data set is divided into two data subsets: the training data, which corresponds to approximately 80% of the data set; and the test data, which corresponds to the remaining 20% and will be used to measure the quality of the model after training.

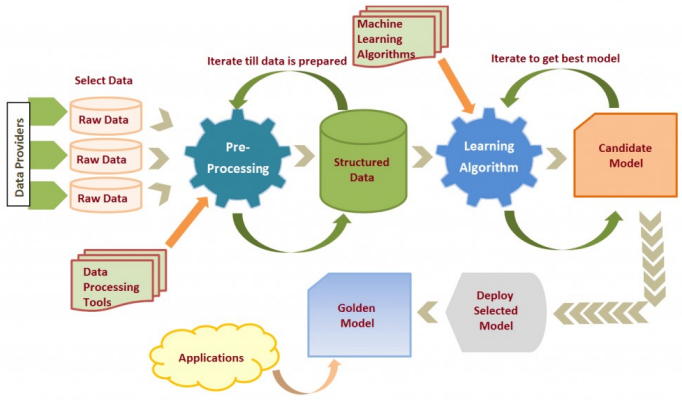


Figure 5.2: Machine Learning Process.

Once we have the data, we need to establish a hypothesis: find an equation that best approximates the real behavior of the modeled phenomenon. This equation is the one that relates the input data and model parameters to the output. What is not measured cannot be improved, so the next step is to find the error in the prediction and try to minimize it. To minimize a function, set its derivative equal to zero. The cost function is responsible for compiling the error between the dependent variable to be determined and the hypothesis based on the model parameters. In some cases, it is easy to find the

formula that introduces the input and output data and provides the best parameter value to reduce the error to a minimum. However, with other models and other cost functions, it will not always be possible to find the minimum of the cost function analytically. For this reason, it is usual to use iterative methods that allow the error to be minimized little by little, such as the Gradient Descent method. The diversity of models and cost functions in Machine Learning makes it necessary to find solutions for non-convex functions, that is, for those with more than one minimum. The Gradient Descent method takes advantage of the derivative calculation to find the local minima since the derivative indicates the value of the slope at a given point. Since what is desired is to reach the minimum point, the logical thing is to advance in the direction in which the slope is high. Hence, the steps to be followed by this method are as follows: locate the largest slope in the current position, move in that direction a certain distance and stop in that new position. The process is repeated iteratively until convergence.

5.2.2 Supervised Learning

In supervised learning, the agent looks at sample input and output data pairs to learn a function that models the output based on the input. Therefore, the data used to build the model is the information to be predicted. There are two types of problems in supervised learning: regression problems and classification problems. A regression model predicts a continuous quantity, while a classification model predicts a label.

Some of the most commonly used regression and classification algorithms are briefly explained below. Among the regression techniques, linear and polynomial regression stand out; and logistic regression, K-Nearest Neighbors and Support Vector Machine, stand out among the classification algorithms. Decision Trees, Random Forest, and Naïve Bayes classification are other known algorithms [216].

5.2.3 Linear Regression vs Polynomial Regression

Linear Regression is a widely used technique in the field of Supervised Learning due to its simplicity and great utility. It consists of predicting a dependent variable y based on one or several independent variables x by drawing the straight line that best fits the data set. Equation (5.1) represents the hypothesis of this model.

$$h(x)= b_0+b_1x_1+b_2x_2+b_3x_3\ldots\ldots\ldots \quad (5.1)$$

The bi coefficients are the model parameters adjusted in the training stage as more data is entered. The model's objective is to minimize the cost function, that is, to make the difference between the actual output value (y) and the prediction value (h) minimal. The cost function is the Mean Square Error, which measures the squared distance between each point and the vertical that joins it with the regression line (Equation 5.2).

$$J\ (a,\ b)\ =12m\ \Sigma(h(x(i))\ -y(i))2 \quad (5.2)$$

The Gradient Descent algorithm is used to minimize this cost function, which is an iterative process that gradually reduces the error until a minimum is found in the cost function. Linear Regression can be simple if there is only one independent variable or multiple if there are more than one. This model is fast and robust, but there must be a certain linear relationship between input and output.

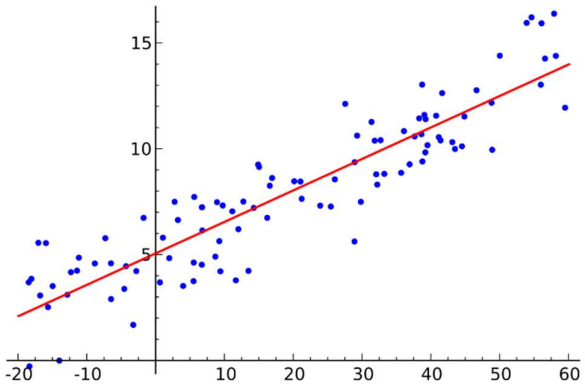


Figure 5.3: Linear Regression. [27]

An example of simple Linear Regression shown in Figure 5.3 is to predict the number of umbrellas sold based on the amount of rain according to the history of the previous year. An example of Multiple Linear Regression is predicting the sales of a product based on the money invested in TV advertising and radio advertising. In the latter case, since there are two independent variables, it is possible to represent them by adjusting a plane instead of a straight line [216].

Polynomial Regression tries to find a polynomial of degree n that fits the data distribution using a curve. It is useful when the Linear Regression cannot fit the data sufficiently due to some non-linearity between them. Equation (5.3) represents the

hypothesis for a one-variable third-degree Polynomial Regression, while Equation (5.4) represents the hypothesis for a two-variable second-degree Polynomial Regression.

$$h(x)=b0+b1x1+b2x12+b3x13 \quad (5.3)$$

$$h(x)=b0+b1x1+b2x12+b3x2+b4x22+b5x1x2 \quad (5.4)$$

Polynomial models gain much flexibility for linear ones since more or less adjusted curves are obtained depending on the polynomial degree. However, increasing the degree of the polynomial too much brings with it the problem of overfitting since, although the error is greatly reduced for the initial data, the model loses its ability to generalize to new input data. For example, the hourly energy demand as a function of the outside temperature follows a non-linear behavior adjusted with a polynomial of degree four.

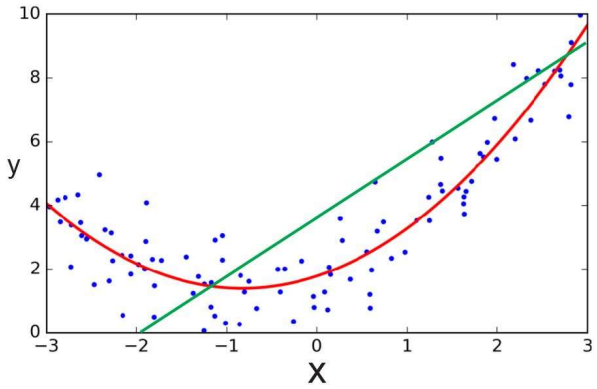


Figure 5.4: Linear Regression Vs Polynomial Regression. [27]

5.2.4 Unsupervised learning

In unsupervised learning, the agent detects existing patterns in the input data without observing the output. Therefore, the objective is to extract meaningful information from the input data, which lacks a label and whose structure is unknown. There are two types of problems in unsupervised learning: clustering and dimensional reduction. Grouping is responsible for creating sets of objects with similar characteristics. At the same time, dimensional reduction looks for redundant information in the data to reduce the number of variables and thus improve computational performance [216].

5.2.5 K-Means

The K-Means algorithm is the most popular clustering algorithm. It is an iterative multi-stage algorithm. The first is to define the variable K, the number of clusters. Then K data are randomly chosen from the set called centroids. Each datum is assigned the nearest centroid, thus obtaining K classes. Moreover, the centroid is moved to the point corresponding to the mean of the distances between each datum and its centroid. The assignment between the centroid and the closest points is made again in this new position.

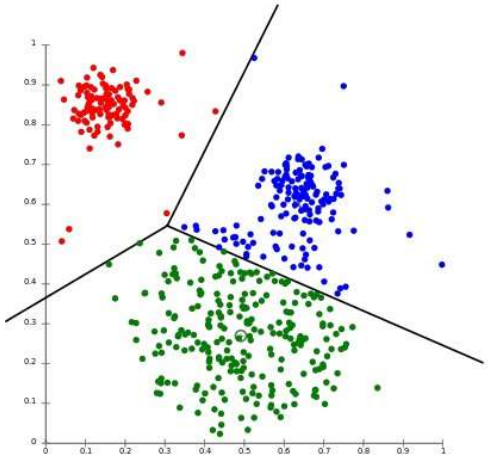


Figure 5.5: K-Means Clustering. [28]

The iterative process ends when the centroid barely moves between one iteration and another, thus reaching convergence. A representation of the process is shown in Figure 5.5.

Sometimes, looking at the data distribution, it is easy to guess the number of clusters. However, it is common to find a data set that is loosely clustered at a glance or even high-dimensional. Therefore, the Elbow Method is usually used to choose an appropriate K value. This method consists of representing the value of the cost function for each value of K and choosing the elbow of the function as the optimal K. The cost function that determines how good this algorithm is called the Distortion cost function, and it corresponds to the average distance between each datum and the assigned centroid. This algorithm is fast, robust, and simple when the data is very distinct or can be linearly separated. On the other hand, it is very sensitive to the value of K and can be ineffective when the data is superimposed or contains a lot of noise. Some examples

of the K-Means algorithm application can be market segmentation or the grouping of words with similar definitions to improve search engines [217].

5.2.6 Principal Component Analysis

Given a data set, Principal Component Analysis (PCA) tries to find the most significant independent variables that represent said data set. When the number of variables and data is very high, the performance of the models decreases due to the computational cost of dealing with some data that does not provide relevant information. For this reason, dimensional reduction algorithms are needed.

Mathematically, PCA reduces the initial n dimensions to k dimensions by finding k vectors into which to project the data, minimizing the projection error. The minimization of the projection error retains the data with the highest variance since they are the ones that play an important role in determining the output label. On the contrary, a low variance indicates that the value of that variable does not influence the label that is trying to be predicted. This does not mean the starting data is ordered from highest to lowest variance, and those with the lowest variance are eliminated. A large amount of information would be lost, but rather that the known vectors must contain the maximum possible variance. The original set of n variables is transformed into another smaller set of known uncorrelated variables called "principal components".

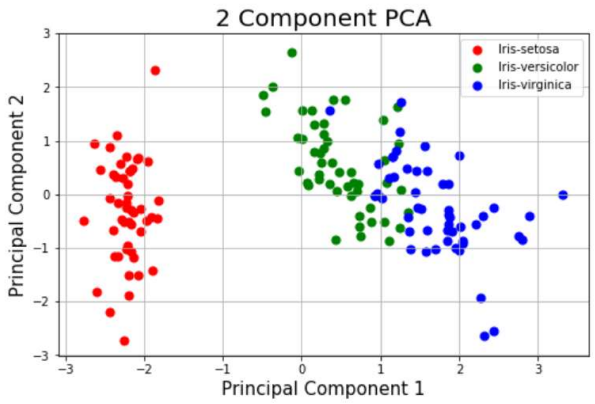


Figure 5.6: Principal Component Analysis. [27]

The computing time of a neural network that works with images is reduced thanks to PCA as a preprocessing stage. This algorithm is very useful in image compression,

where each pixel corresponds to a variable. In Figure 5.6, the objective is to retain the information of the pixels that contain the label to be predicted, so all the pixels that do not contain cats are useless [218][219].

5.2.7 Singular Value Decomposition

In linear algebra, Singular Value Decomposition (SVD) is a method of factoring a real or complex matrix used for dimension reduction. It is based on the principle of decomposition of vectors in their orthogonal axes as shown in Figure 5.7, so that any vector a can be expressed by two variables: the unit vector that indicates the direction of projection ($v1$) and the length of the projection ($sa1$). In SVD, this conclusion is extended to many vectors and in all dimensions.

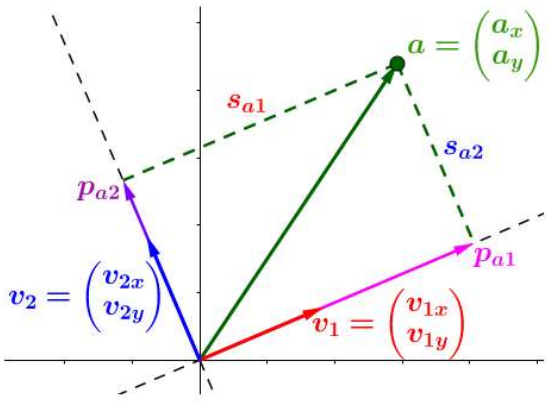


Figure 5.7: Singular Value Decomposition. [28]

The' projection of the vector 'a' in the direction of the unit vectors gives the length of the projections, according to equation (5.5).

$$a^T.\ V_i = (a_x\ a_x).(\frac{v_{ix}}{v_{iy}}) = S_{ai} \tag{5.5}$$

Using matrices, equation (5.6) is reached, where A is the point matrix, V is the orthogonal axis decomposition matrix, and S is the projection length matrix.

$$A = SV^{-1} = SV^T \tag{5.6}$$

In equation (5.7), the matrix S is decomposed into the matrices U and Σ.

$$S = U\ \Sigma \tag{5.7}$$

Where U is the matrix of projection unit lengths and Σ includes the values of σi, which increase if the points approach the axis of the corresponding σi, as indicated by equation (5.8).

$$\sigma_i = \sqrt{((S_{ai})^2)} + (S_{bi})^2 \quad (5.8)$$

Finally, the expression of the SVD method is the one shown in equation (5.9).

$$A = U \Sigma V^T \quad (5.9)$$

The main application of this method is to choose some unit vectors (matrix V) that coincide with the main components of the data set (matrix A). Thus, if the line of maximum variance is known, the points can be projected onto it.

5.2.8 Independent Component Analysis

Independent Component Analysis (ICA) is a computational and statistical technique used to reveal hidden factors within variables, measures, or signals. The variables found by ICA are called "independent components." The objective is to decompose a multivariable signal into additive subcomponents assuming that they are non-Gaussian and statistically independent signals. This dimensional reduction technique, an extension of PCA, is much more powerful and works well in cases where classical methods fail. PCA can only impose independence up to the second order and thus define orthogonal directions. Tt is assumed that random variables follow Gaussian distributions, preventing ICA application. The ICA is a particular type of blind source separation, as it obtains the signals from n sources from the mixtures collected by m sensors. The ICA handles the blind recovery of said sources, assuming that the original signals are independent. A common example of its application in audio is the cocktail party problem, which occurs when several speakers speak simultaneously. The idea is to focus solely on one speaker to extract their voice and remove background noise or other conversations. This process can be modeled as a linear mix followed by source filtering [220].

5.2.9 Reinforced Learning

Reinforcement learning is a form of machine learning based on a system of rewards and punishments in which an agent seeks the optimal decisions to obtain the maximum

reward both in the short and long term. The reward measures the contribution of the action to the final goal. This agent needs to know if its decision is good or bad. However, instead of having a supervisor as in supervised learning, it learns from its own experience from interactions with the environment. The framework that defines the interaction between the agent and the environment is a Markov Decision Process (MDP). Each state depends only on the previous state. Figure 5.8 schematically represents the three types of learning: supervised, unsupervised, and reinforcement. Unlike unsupervised learning, reinforcement learning tries to maximize the reward function instead of finding certain hidden patterns in an unlabeled data set. In addition, these rewards do not have to be offered instantly but may arrive delayed.

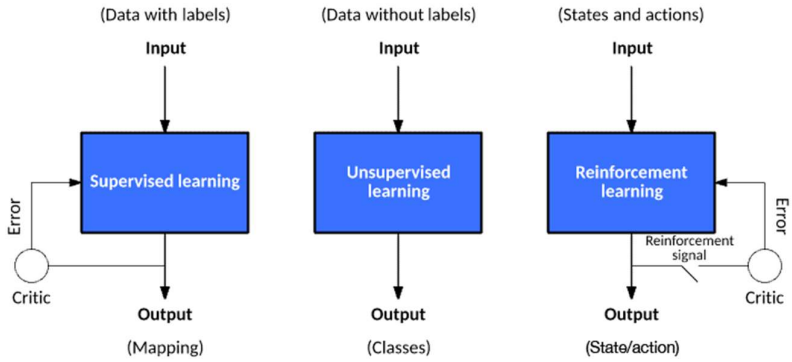


Figure 5.8: Types of Learning.

The main challenge of reinforcement learning is to balance exploration and exploitation (exploration and exploitation dilemma). To obtain high rewards, the agent tends to repeat those actions that have turned out to be positive. However, to discover these actions, it is necessary to test those that have not been previously tested. Therefore, the agent exploits current knowledge while exploring different options.

Each reinforcement learning algorithm must follow a policy (policy) to decide what decision to take based on the state (state) in which it is found. However, this policy may not be followed in the learning stage. Those algorithms whose update rule performs the action that will bring the maximum benefit, although the current policy restricts said action, is called off-policy algorithms. On the contrary, on-policy algorithms are those that strictly follow the policy. When an agent tries to optimize its policy, it can do so from two different approaches. With a Model-Free approach, the agent obtains

information without prior knowledge of the environment through trial and error. Following a Model-Based approach, the agent calculates all states' success probabilities before acting based on these probabilities and thus models the environment [220].

5.2.10 Q-Learning

The Q in Q-learning stands for quality, which in this case, measures how good action has been to earn a future reward. The goal is to learn the policy that maximizes the total reward. The algorithm creates a Q-table or matrix representing states versus actions. This matrix is initialized with all values to zero, and with each episode, the Q-values of all possible state-action pairs are updated. The matrix becomes a reference for the agent to select the best action based on the Q-values. At this point, the agent can interact with the environment in two ways. The first consists of exploiting the acquired knowledge, so the next action will be the one with a maximum Q-value. The second is exploration, where the agent acts randomly to add new states to the table. The parameter ε balances the percentage of exploitation and exploration actions. Equation (5.10) represents the update rule of this algorithm.

$$\mathrm{Q}(\mathrm{s}_t, \mathrm{a}_t) \leftarrow \mathrm{Q}(\mathrm{s}_t, \mathrm{a}_t) + \mathrm{a}\,[\,\mathrm{r}_{t+1}, + \xi \max Q\,(\mathrm{s}_{t+1}, \mathrm{a}) - \mathrm{Q}\,(\mathrm{s}_t, \mathrm{a}_t)] \qquad (5.10)$$

This rule adjusts the Q-values based on the difference between the new and old values. The new values are discounted with the discount factor (γ), and the step size is adjusted with the learning rate (α). The reward (r) is the value received after completing an action in a given state. Therefore, the next action is chosen to maximize the Q-value of the next state instead of following the current policy. A more complex and popular version is the Deep Q-Network, which replaces the state-action matrix with a neural network to work with massive amounts of data and perform more complex tasks. One of the applications of these algorithms is to make computers learn to play video games even better than humans [221].

5.2.11 SARSA

SARSA (State-Action-Reward-State-Action) is a reinforcement learning algorithm that learns a policy based on a Markov Decision Process, just like Q-learning. The

difference is that SARSA is an on-policy algorithm; the learned Q-values always follow the current policy. Equation (5.11) represents the update rule of this algorithm.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + a\,[\, r_{t+1}, + \xi\, Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \quad (5.11)$$

This quintuple (s, a, r, s', a') gives the algorithm its name. The following action means that it must store the information for a long time before updating the values. This rule depends on the current state, the current action, the reward obtained, the next state, and the next action.

The convergence of the method depends on the nature of the policy, its dependence on Q, and the learning rate. SARSA converges with probability 1 to an optimal policy if all state-action pairs are visited infinite times. SARSA has faster convergence but is more likely to get stuck in a local minimum. On the other hand, Q-learning has better final performance, but it needs more learning time. A combination of both is Backward Q-learning, which increases learning speed while improving final performance [221].

5.2.12 Deep Learning

In recent times, a distinct realm within Machine Learning has surfaced, known as Deep Learning. This domain revolves around learning algorithms operating across multiple tiers of representation and abstraction, enabling the modeling of intricate relationships within data. These tiers signify varying degrees of concepts, with the upper echelons being hierarchically informed by the lower tiers. The notion of sequential representations achieved through these stratified levels lends the term "deep" to this form of learning, with the model's depth corresponding to the number of layers it encompasses. Within the realm of Deep Learning, these stratified representations coalesce into a Neural Network—a meticulously organized assembly of neurons. This nomenclature is rooted in neurobiology, drawing inspiration from the workings of the brain. Layers within this network comprise a specific quantity of neurons, each neuron gathering input from the preceding layer via external stimuli transmitted through its input connections. Internally, neurons conduct computations, yielding an output value channeled to neurons within the ensuing layer. This output serves as the sought-after prediction.

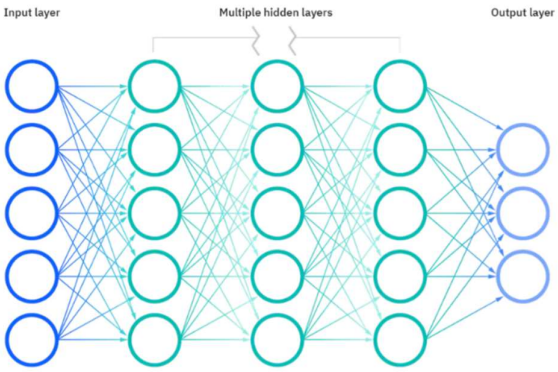


Figure 5.9: Deep Neural Network.

Nevertheless, in order to achieve accurate predictions, the network must curtail the disparity between the prediction and the anticipated outcome through the automatic adjustment of model parameters. Within this context, each neuron allocates weights (w) to its individual input variables (x), thereby gauging the extent to which each variable influences the output, as depicted in equation (5.12).

$$w_0x_0 + w_1x_1 + w_2x_2 + \ldots .. + w_nx_n \tag{5.12}$$

Following the computation of the linear amalgamation of weights and inputs, an activation function (∅) is employed. The purpose of this function is to transform the output (y), imparting a non-linear character to it. This non-linearity enables the network to address problems of a non-linear nature, as delineated in equation (5.13).

$$y = \varphi\ (\ w_0x_0 + w_1x_1 + w_2x_2 + \ldots .. + w_nx_n\ ) \tag{5.13}$$

Backpropagation, also known as backward propagation, serves as the learning mechanism for neural networks. It involves transmitting error information from the last layers to the initial ones, thereby adjusting the parameters of each neuron. However, a challenge arises with this process: as the error propagates through layers, it diminishes, potentially causing only the final layers to receive significant training attention in very deep networks. Consequently, the neural network predominantly understands the weight or impact of each neuron on the erroneous outcome. The advancement in deep learning lies in its capacity to facilitate concurrent learning across all layers, circumventing the need for a sequential approach. Adjusting a parameter prompts

automatic adaptations in the interconnected parameters, streamlining the process under a unified feedback signal.

Deep Learning possesses several vital attributes that mark it as a revolution within artificial intelligence: simplicity, scalability, and versatility. Its simplicity arises from the automation of feature engineering, which historically demanded substantial time and effort before model development. Scalability empowers the manipulation of datasets of varying sizes, while versatility enables the utilization of pre-trained models for diverse applications such as text generation, language processing, object identification in images and videos, and more. Deep Learning finds application in numerous domains, including client prospect identification for companies, disease prediction via image and medical data analysis, real-time threat detection and prevention in cybersecurity, and beyond. Various types of neural networks cater to specific purposes and degrees of complexity. The choice of network type hinges on the nature of the data at hand. Nevertheless, their overarching objective is to discern patterns within data to execute specific tasks like classification, clustering, or predictive analysis. In the subsequent sections, three neural network types are elucidated: Convolutional Neural Networks, Recurrent Neural Networks, and Generative Adversarial Networks [222-225].

5.3 Machine Learning Algorithms for Data Classification in IoT

The Scikit-learn package offers an extensive array of algorithms capable of processing and manipulating datasets provided as input. Among these algorithms, many are proficient in forecasting forthcoming continuous values over time or categorizing data, drawing from both historical and novel observations. These algorithms span diverse categories contingent on the specific classification tasks they undertake [226]. The most frequently used and significant algorithms that are commonly employed to address various classification problems will be highlighted and analyzed in this work. Table 5.1 presents a list of commonly used Machine Learning Algorithms for classifying IoT data. The primary emphasis lies in the domain of classification, particularly in the context of supervised learning using preexisting datasets. In this scenario, the objective is to determine the category to which a novel object belongs, even if we haven't

encountered it before, drawing insights from our past observations. Thus, we can find the following within these categories according to their nature:

Table 5.1: Machine Learning Algorithms for IoT data Classification.

Algorithm Family	Algorithm
Linear Models	• Logistic Regression • Perceptron
Neural Network	• Multi-Layer Perceptron
Discriminant Analysis	• Linear Discriminant Analysis • Quadratic Discriminant Analysis
Support Vector Machines	• Linear Support Vector Machines • Polynomial Support Vector Machines
Neighbors	• K-neighbors
Naïve Bayes	• Bernoulli NB • Gaussian NB • Multinomial NB
Trees	• Decision Tree
Ensemble	• AdaBoost • Random Forest

5.3.1 Perceptron:

The perceptron is a model of a simple neuron. In 1958 the psychologist Frank Rosenblat developed this model based on McCulloch and Pitts and a learning rule based on error correction; he called this model Perceptron. What was most striking about this model is its ability to learn to recognize patterns. The perceptron is based on a series of sensors or inputs from which it receives the data to classify or recognize and where we also have an output neuron to tell us if it belongs to one class or another, activating or not depending on whether the output is 1 or 0. The simple perceptron algorithm is based on fundamental functions that enable data classification using the concept of a simple perceptron. Let us start by assuming that we have the function f of Rn in {-1, 1}, to which we can apply an input pattern x = (x_1, x_2, x_n) T ∈ Rn and where we will have an output desired z ∈ {-1, 1}, or what is the same, f(x)= z. We will consider that input pattern as each characteristic that our data set has and to which we are going to pass said function. Since we have several input patterns to carry out the classification, we

will have the following relationship: $\{x_1, z_1\}$, $\{x_2, z_2\}$ .... $\{x_p, z_p\}$, where x_i is the input pattern i ∈ R_n yz = f (x_i) [227]. What the function performs is a partition of the set of inputs into two spaces, on the one hand we would have the input patterns whose output is +1 and on the other the patterns whose output is -1, so we can say that this function is capable of distinguishing between two classes. Since our problem deals with the classification of more than two classes, the simple perceptron supports the one-vs-all classification technique since the outputs we obtain are continuous and numerical, a technique explained earlier in this chapter. Moreover, how can we build a model that fulfills that function? For this, we will start from a bipolar process unit that fulfills the following function:

$$\frac{\mathbf{1}\ s_i w_1 x_1 + w_2 x_2 + \cdots + w_n x_n \geq \theta}{-\mathbf{1}\ s_i w_1 x_1 + w_2 x_2 + \cdots + w_n x_n \leq \theta} \tag{5.14}$$

Where we have that the parameters w_i are the so-called synaptic weights. These weights will assume the importance or weight we give to each input value. On the other hand, we have the weighted sum that we call synaptic potential and the threshold □. When the output of the said function is 1, it is said to be active, and otherwise, its output will be -1 or inactive.

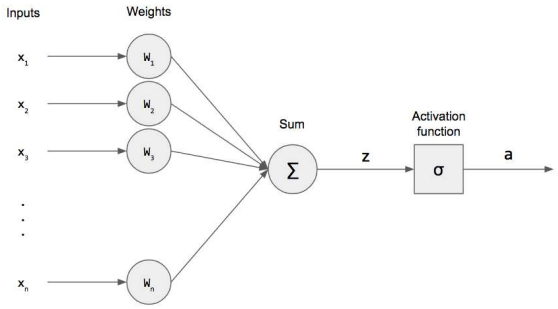


Figure 5. 10: Perceptron.

5.3.2 Logistic Regression

Logistic Regression is a classification algorithm used to predict the outcome of a binary categorical variable based on the independent variables. The objectives of this model are: to determine the existence or absence of a relationship between one or more independent variables and a dichotomous dependent variable to measure the sign of a

said relationship and predict the probability that the event Y=1 will occur based on the values of the independent variables [228].

The logistic function is the one that, for each individual, finds the probability (P) that the effect in question occurs. To understand the origin of this function, it is necessary to explain the concept of Odd, which is the ratio between the probability that an event occurs and that this event does not occur (Equation 5.15).

$$\text{Odd} = \text{P} / 1 - \text{P} \tag{5.15}$$

The Logit function shown in equation (5.16) is a logarithmic transformation on the Odd that converts probability values in the range [0,1] into values within the range [-∞, ∞].

$$\textit{Logit}(\text{P}) = \textit{ln}\ \text{P} / 1 - \text{P} \tag{5.16}$$

The Logit function can be represented linearly as a function similar to the one used in Multiple Linear Regression shown in equation (5.17).

$$Z = w^T x = w_0 + w_1 x_1 + \ldots + w_m x_m \tag{5.17}$$

In this way, it is obtained that the inverse of the Logit function is the logistic function sought (hypothesis function), called the sigmoid function represented in Figure 5.11.

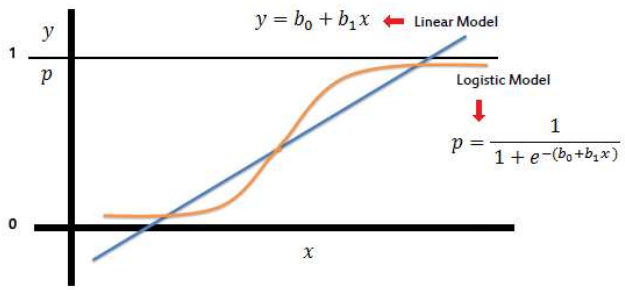


Figure 5.11: Logistic Regression.

This function is responsible for taking real values and transforming them into a value in the range [0,1], which indicates the probability that a sample belongs to class 1 given the characteristics x parameterized by the weights w. A quantizer translates the probability into a binary output as shown in Equation (5.18).

$$Y = \begin{cases} 1 & if\ z \geq 0 \\ 0\ for\ the\ rest \end{cases} \quad (5.18)$$

This often happens in medicine since it tries to answer questions formulated based on the presence or absence of a certain characteristic that is not quantifiable but rather represents the existence or not of an effect of interest. For example, Logistic Regression in medicine is used to predict whether a tumor is benign or malignant based on the patient's age and size. The decision limit is marked by $z=wTx$, so the degree of each term is varied to obtain the curve that best fits the data.

5.3.3 Neural Network

The Multilayer Perceptron is a generalization of the Simple Perceptron and arose due to its limitations when classifying data sets that were not linearly separable. Minsky and Papert were able to show in 1969 that combining the use of several simple Perceptron. Let us consider them as hidden layers, would solve the classification problem for problems that are not linear. However, for this solution, there was no definition of how to adapt the synaptic weights for each perceptron in the hidden layer since the simple Perceptron rule cannot be applied to this problem. Despite this, combining several simple Perceptron served as a help to the studies carried out by Rumelhart, Hinton, and Williams in 1986. They presented a way of committing a backpropagating error and adapting synaptic weights through a generalized delta or backpropagation rule.

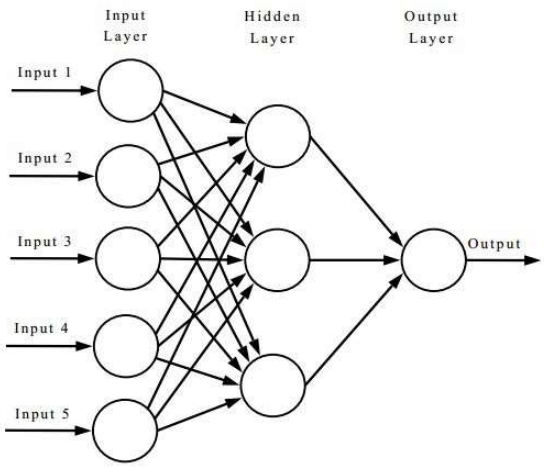


Figure 5.12: Neural Network.

The architecture of Multilayer Perceptron, as its name indicates, is based on the structuring of its neurons in various levels or layers. This architecture is a feedforward network where we can find an input layer, which we call sensors or our data set's characteristics. Another output layer will be the different possible types to classify and a certain number of intermediate layers of process units, which we can also call hidden as they have no connection with the outside. The role played by the hidden or intermediate layer is that of a projection of the input patterns onto a cube whose dimension is given by the number of units in the hidden layer. With this, what is intended is to make a projection in which the input patterns are linearly separable so that the value we get in the output layer is as correct as possible. The output units are connected only to the last hidden layer. In this type of network, what is intended is to establish a relationship between an input set and an output set. Thus, we have the following relationship: $(x_1, x_2, x_3 \ldots x_n) \in R_n \dashrightarrow (y_1, y_2, y_3 \ldots y_m) \in R_m$. In this way, we start from a set p of training patterns where we know that the pattern input (xk_1, xk_2, xk_n) corresponds to the output (yk_1, yk_2, yk_m) with k =1, 2, p. In Figure 5.12, we have shown a representation of this type of neural network. To understand this relationship, we indicate that the input layer has as many neurons as there are variables or characteristics in our data set. On the other hand, in the output layer, we will have as many process units as desired outputs in our system. The determination of the number of hidden layers is not established concretely. Using a greater number of hidden layers does not ensure that we obtain a better precision when finding the desired output since the methods used to train this type of network can be more expensive with a greater number of layers and can take time. Too long to find the best values for our training [229-231].

5.3.4 Support Vector Machines

The support vector algorithm (SVM) is a discriminative classifier whose objective is to define a hyperplane in N-dimensional space (N is the number of independent variables) that maximizes the distance between data of both classes. The hyperplane is the decision boundary that helps classify the data, so each side of the hyperplane is assigned a class. Support vectors are those closest to the hyperplane, thus influencing its position and orientation. Finding a hyperplane that separates the data is usually not a one-size-

fits-all solution. However, the optimal solution is the one that maximizes the distance between two parallel lines located symmetrically on each side of the decision boundary such that the support vectors are contained in them. This distance is known as margin. Those models with a large margin reduce the generalization error, while models with a small margin tend to be less prone to overfitting. Figure 5.13 represents the hyperplane chosen from among all the possible hyperplanes, as well as the different elements that have been commented on.

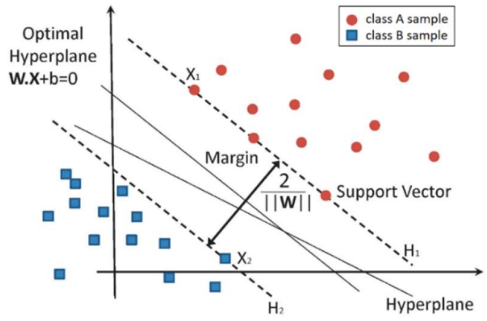


Figure 5.13: Classification of Data by Support Vector Machine.

In Logistic Regression, the linear output value is transformed into a value in the range [0,1] through the sigmoid function, where for values greater than 0.5, class 1 is assigned, and for smaller values, class 0. In the algorithm of support vectors, if the linear output is greater than 1, it is identified with one class, and if it is less than -1, it is identified with the other. Therefore, the range that acts as a margin is [-1,1]. For nonlinear classification problems, mapping and kernelization techniques are used. These techniques reorganize the data through mathematical functions, either in the same plane or in planes of higher dimensions, to adopt a configuration that makes them separable. This is one of the great advantages of support vectors over other models and their significant accuracy at the cost of low computing power. Furthermore, these principles can be used to solve regression problems [232].

5.3.5 Linear Support Vector Machine:

Linear algebra is used to find the hyperplane that separates the two data sets. One way to look at this type of algorithm is to use the inner product of two particular observations, the dot product of two vectors, instead of the observations themselves. This means that the dot product of two vectors is the sum of the multiplication of each

pair of input values of each vector. The core of the Linear SVM algorithm is responsible for defining the distance between the vectors and the new data. The more complex the kernel (such as Polynomial Kernel or Radial Kernel), the better the classification as they allow for curved lines that can better fit the data distribution. For this type of algorithm, the kernel is what we call the scalar product and which we represent as follows: K (x, x_i) = sum(x*x_i), where x is the data vector, we are looking at xi is the support vector *i*.

5.3.6 Polynomial Support Vector Machine

Like Linear, a polynomial can be used as a kernel; in this case, we will have K (x, x_i) = 1+ sum(x*x_i) d, where d indicates the degree of the polynomial. If d=1, the kernel would be the same as the Linear type. As mentioned previously, to a greater degree, it will allow the use of a more curved vector adjustment, but this also implies a higher computational cost.

5.3.7 Discriminant Analysis

Linear Discriminant Analysis, this type of algorithm, is easy to handle in those cases where the frequencies within each class are unequal, and the performance has been evaluated based on randomly generated data. This method tries to maximize the variance ratio that exists for both the data within each class and the variance that exists between the different classes in any particular data set. This ensures that the maximum separation between classes is achieved by drawing a separation region between the given classes. The formulas to carry out these calculations derive from basic probabilistic models in charge of extracting a model from the probable class conditioned through the data P(X|y=k) for each class k. The predictions can be obtained through the Bayes rule, and we select the class k that maximizes this conditional probability. If we want to use this model as a classifier, it is necessary to estimate from the training set the a priori class to which it belongs and the covariance matrices. The classification carried out in this type of classifier is creating a probability model for each class following a Gaussian distribution. In this case, we have that the Gaussian distributions of each class are the same and therefore share the same type of covariance matrix. In this way, we will achieve linear decision surfaces between the classes. As an

example of this type of algorithm, we can find it for automatic face recognition or document classification. Linear Discriminant Analysis is responsible for extracting a set of features that gives us the most relevant information to carry out the classification. This is done by analyzing the eigenvectors of scattering matrices to maximize the variations between classes and minimize the variations within the same class. In this way, the instances can be discriminated against to know if they belong to one class [233][234].

5.3.8 Quadratic Discriminant Analysis

The classification using this method is very similar to the previous Linear version with the difference that this time the Gaussian distributions of the classes are not assumed. Therefore, the posterior distributions are used to estimate the class given a test example. Giving rise to decision surfaces of a quadratic form. These Gaussian parameters for each class can be obtained from the training set with a maximum likelihood estimate.

5.3.9 K-Nearest Neighbors

The K-Nearest Neighbors (K-NN) algorithm is a classification algorithm that labels each data item based on the label of the data closest to it. For this, the variable k is defined, which corresponds to the number of closest neighbors chosen to carry out the classification. Depending on this variable, we have some predictions or others. In the example shown in Figure 5.14, for k=1, the algorithm classifies the element as white; for k=2, the algorithm needs a criterion to classify since there is a neighbor of each color; and for k=3, the algorithm classifies the element as black, since two of the three nearby elements are black.

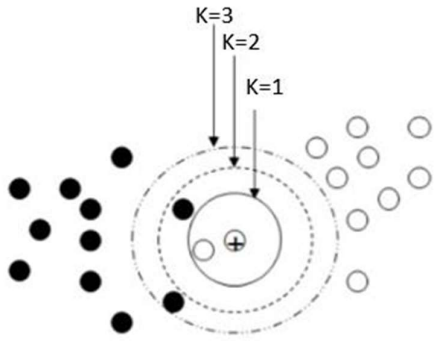


Figure 5.14: K-NN.

Unlike other supervised learning algorithms, K-NN does not learn with the training data, but rather the learning occurs with the test data once the training data has been memorized. These algorithms are known as lazy algorithms, and they allow several problems to be solved simultaneously since the objective function is approximated locally for each element. The algorithm does not learn from a model but instead uses the data to generate an answer only when a prediction is requested. As a disadvantage, the computational cost is very high due to all the training data storage [235].

5.3.10 Naive Bayes

The Naive Bayes classifier is widely used and gives great results for text analysis and classification. To understand how this classifier works, it is necessary to understand the Bayes theorem. This theorem works using conditional probability, or the probability of something happening, given what has happened previously. So, we can calculate the probability with which something will happen based on the conditional probability. The formula to calculate this probability would be:

$$P(h|D) = P(D|h)\ P(h)P(D) \tag{5.19}$$

Where:

- P(h) is the prior probability of hypothesis h.
- P(D) is the probability when looking at the training set D.
- P(D|h) is the probability of observing the training set D in a universe where the hypothesis 'h' is verified.
- P(h|D) is the posterior probability of h when the training set D has been observed.

Let us suppose the following example extracted from the following reference to understand this theorem. "Suppose an engineer is searching for water on a piece of land. A priori, it is known that the probability of water in the said farm is 60%. However, the engineer wants to make sure better and decides to carry out a test that allows detecting the presence or not of water. This test has a reliability of 90%; if there is water, it detects it in 90% of cases. Also, when there is no water, the test predicts no water in 90% of cases. Therefore, using the said test, what is more likely, whether there is water or not? Here, we can observe that we have a 60% probability that there may be water and that

it is conditioned by the 90% probability that the test is correct. Thus, Thomas Bayes gave us the solution to this problem with his theorem and the formula previously mentioned. For the selection of the hypothesis that uses the Maximum a Posterior technique, also known as MAP and which is as follows: hMAP= argmaxh∉HP(h|D) = argmaxh∉H$P(D|h)$ $P(h)P(D)$ = argmaxh∉HP(D|h) P(h). since many times the hypotheses are comparable, it is not necessary to multiply by P(h). Therefore, we are left with hML= argmaxh∉HP(D|h), called maximum likelihood. Considering the previous problem of the search for water, the one with the highest probability will be the solution to the problem above, that is, knowing if we will find water in said land. The Bayesian classifier will extract the most probable classification for a new example given a training set [236]. There are different types of Bayesian classification depending on the type of problem and data we are dealing with and how they are distributed.

5.3.11 Bernouilli NB

This type is used for data distributed according to Bernoulli's variegated distribution, i.e., there may be multiple attributes in our data. However, each one is assumed to have a binary variable, true or false, 1 or 0. Therefore this algorithm works very well when the attributes have binary values.

GaussianNB: If the values of the attributes are continuous, then it is assumed that the values of the different classes are distributed according to a Gaussian distribution, that is, a Normal Distribution. Therefore, this type would be the most appropriate if we have this distribution of data.

MultinomialNB: If the values we have are distributed according to different values, then it is preferable to use Naive Bayes Multinomial, being one of the standard classification algorithms used, for example, for text classification or categorization. For example, each event in the text classification represents the occurrence of a word in a document.

5.3.12 Decision Tree

The use of decision trees can be found in several fields, such as text extraction and classification (cancer detection, heart problems, or detection of different diseases such as Parkinson's or if a patient must be hospitalized having dengue. If there is something that we can highlight about this type of algorithm and the visualization of its

effectiveness, it is because of its easily understandable classification rules for humans. The idea comes through the well-known structure of trees where we can find a root and some nodes, where we will see their respective branches and leaves. A decision tree starts at the root node and spans down two or more branches from left to right. The node where the chain ends is what we will call the leaf node. The interpretation of a decision tree would be as follows: we will consider each intermediate node of the tree as an attribute or characteristic of the data to be processed, the root node being the most relevant attribute, and each leaf will correspond to a class or hashtag. In decision trees, a grouping of the data is performed based on the values of the attributes of the data we have. A class division is carried out according to the attribute that best distinguishes between some others. This process is applied recursively until all the data of a subset being treated belongs to the same class. The pseudocode of a decision tree would be as follows: *1.* We place in the root node the attribute that is most relevant to the decision. *2*. Divide the training set into subsets where we have data that contain the same value for the attribute chosen in steps 1. *3*. Repeat steps 1 and 2 until we have found all the leaves on each tree branch [237].

5.3.13 Ensemble

The objective of this type of method is to combine the predictions of several classifiers with building a new one that is more robust and accurate. We can find two types depending on the technique used to find a new classifier and improve accuracy.

Reinforcement methods: Classifiers are created sequentially, trying to correct the direction of the previous one, taking into account the error made in its rules. The idea is to build a more accurate classifier starting from several with lower precision or fit. The AdaBoost method (Adaptive Boosting) will be studied in this analysis.

Averaging methods: Different independent classifiers are created, and their predictions are averaged. Using the average is better than predicting only one since the variance is reduced. The focus of this study will be on exploring the Random Forest method.

5.3.14 AdaBoost

This algorithm created by Freund and Schapire is one of the best options for classification binary decision trees. It is used in numerous fields such as network

intrusion detection, object detection or for, text classification, and also for the detection of different heart diseases. This method is responsible for analyzing several weak algorithms, which will be small decision trees of depth 1, which have little precision through the modified training set in each iteration. Sequentially, it will form a new, more precise one with the combination of all of them, assigning a weight in the final vote for the choice of the best rules for each one. The data modification in each reinforcement iteration consists of applying weights to each of the examples of the training set. These weights are all equal, and in the first iteration, data classification is performed. In each iteration, the weights of each data are modified, and a classification is made again with these new weights. The weight that was not modified properly will be increased. The weight will be increased oppositely for those that were not, so those data that are more difficult to classify are given more importance. The next simple classifier puts more interest in them—those who are heavier. To know how this algorithm works best, we can expand this information in the reference [238].

5.3.15 Random Forest

In this type of ensemble algorithm, a search for a more accurate classifier is performed using decision trees. Its use can be found in classifying the type of land cover. Each decision tree created is made from a random sequence of a subset of data from the training set. The difference with a decision tree is that the attribute with the highest gain or Gini index is not taken at the node where we have to divide the data. However, the best division between a random subset of the entire set of characteristics is taken. This will cause a greater deviation of the tree, but the variance decreases when using the mean of several trees. This algorithm classifies as follows: given an example with its respective characteristics, it is placed in each of the decision trees that exist in the forest and that the algorithm has created [239]. Each tree will give a ranking, and that tree gives a vote to that class that we have predicted. The forest chooses the classification through the class that has received the most votes. Scikit-learn's implementation is to combine several classifiers based on their probabilistic prediction, rather than using the vote each classifier gives to the class of the classified example.

5.4 Feature Extraction

Data dimensionality and the need for its reduction are critical considerations in contemporary storage and processing, as high dimensionality slows down training. Nevertheless, numerous attributes can assist algorithms in identifying optimal solutions. Various techniques enable us to extract and project data onto new datasets with fewer attributes, yielding metrics comparable to the original dataset but with reduced computational and storage costs. However, it is crucial to acknowledge that these techniques may compromise data quality, potentially resulting in decreased precision, despite faster training in certain cases. Consequently, it becomes necessary to investigate how these techniques impact different algorithms to determine whether attribute reduction is worthwhile. Moreover, these techniques enhance speed, reduce storage requirements, and facilitate improved data visualization. For instance, by reducing dimensions to 2 or 3, we can graphically represent the data, enhancing our understanding of its distribution. One such technique is Feature Extraction, which aims to decrease data dimensionality while preserving attributes that contain the most significant information. Within these techniques, we will study a well-known one used and implemented in the sci-kit-learn package: PCA (Principal Component Analysis). This linear data transformation technique helps identify patterns based on the correlation between attributes. PCA is responsible for finding the hyperplane closest to the data and projecting them into a new space smaller than the original. The orthogonal axes, which are the main components where these new data are projected, of the new sub-space can be interpreted as the directions of the maximum variance, knowing that the new set of features is orthogonal to each other. We can see it in the following image.

Vector PCA wants to project a vector with one dimension onto another with a lower dimension. Thus, in the new vector of the searched subspace, the first component will be that attribute with the greatest possible variance, taking into account that there is no correlation between them; that is, those components that remain in the new are orthogonal to each other. It is appropriate to indicate that PCA is sensitive to the distribution of the data and the scale at which they are found. We will take this into account when we analyze how it is possible to reduce the dimension of the data set that we are dealing with. We first have to have all the data on the same scale to give all the

features the same importance and then apply this reduction technique. This is something that we will take later chapters, we will study how this technique affects our data set, and we will analyze whether or not it affects the different metrics that we can obtain from the different algorithms.

5.5 Proposed Methodology

The proposed Framework (MLADCF) distributes the processing load to the last nodes of a digital network (sensors in the case of IoT). The use of computing type poses very attractive advantages for IoT solution providers. For example, they allow to minimize latency and preserve network bandwidth, operate reliably, speed up decision-making, capture and protect a large number and types of data, and transfer the data to the most appropriate place for processing, with better analysis of local data. Edge computing technologies have been on the rise for several years, but the reach of IoT technology is accelerating its take-off process. As for the factors driving this change, two stand out: Falling prices for peripheral devices with increasing processing power. Centralized infrastructures support the increasing workload. The MLADCF combines with hybrid resource constrained KNN (HRCKNN) model to give the system a complete learning approach for future IoT environment.

5.5.1. Machine Learning Analytics based Data Classification Framework for IoT (MLADCF)

To implement the proposed framework let us first consider n number of IoT devices in an IoT environment. Let the matric A_{m*n} denote the IoT devices and its Corresponding sensors as shown below:

$$A= \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \text{m*n} \tag{5.20}$$

Where ‘m’ is the number of IoT devices and ‘n’ is the number of sensors included. Let I_k be the element a_{ij} of matrix A.

$$\text{If } a_{ij} = \begin{cases} 1 & Then\ go\ to\ vector\ S_\alpha \\ 0 & Sensor\ not\ Present \end{cases} \tag{5.21}$$

The S_α below denotes the vector from the matrix A

$$S_\alpha = \begin{bmatrix} D_1 \\ D_2 \\ D_3 \\ . \\ . \\ . \\ D_\beta \end{bmatrix} \beta^* \tag{5.22}$$

Where β is the number of data Chunks/clusters generated by 'n' sensors. Let d_j be the j^{th} element of vector S_α and each d_j will be a column vector as

$$D_j = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ . \\ . \\ . \\ P_\gamma \end{bmatrix} \gamma^* \tag{5.23}$$

Where γ is the number of packets in the j^{th} data cluster and $d_j \leq S_\alpha$ {where d_j is a subset of S_α}.

The equation will allow us to optimize the resources and to classify the data packet for the edge node. Edge computing technology also arrives at artificial intelligence on devices much more feasible. It allows companies to leverage their data series in real-time rather than working with terabytes of data in central repositories in the world real-time cloud. In the next few years or decades, the technology may evolve to find a balance point between the cloud and more powerful distributed edge devices. Software vendors develop specific, more robust, and secure infrastructures and security solutions. Providers will begin to incorporate security solutions for peripheral components into their current service offering to prevent data loss, provide network health diagnostics, and protect against threats. From the above eq. (5.23), If Determinant of $\sum_{i=1}^{\gamma} d_i \leq S_k$ then the data will be processed at the device. This can be written as

$$\text{Det} \sum_{i=1}^{\gamma} d_i \leq S_k \tag{5.24}$$

If Det $\sum_{i=1}^{\gamma} d_i > S_k$, that means the device is not capable of processing the data, therefore data will be offloaded and will be pushed to the higher level. Similarly, equation (5.24) can also be written as

$$\text{Det} \sum_{i=1}^{\gamma} d_i \leq S_{k+1} \tag{5.25}$$

If Det $\sum_{i=1}^{\gamma} d_i \leq S_{k+1}$, that means device is fully capable of processing the data at the k^{th} sensor and will not be forwarded for processing.

And if Det $\sum_{i=1}^{\gamma} d_i > S_{k+1}$, that means device is not capable of processing the data and data will be forwarded to the edge level.

The resulting mathematical design considered, above all, the need to ensure the exchange of information of all the components of the system, always taking care of the speed of calculation. These elements require reconciling characteristics that are often incompatible. For example, more robust cryptography systems may require more computing power than many lightweight devices, such as sensors, provide. The final architecture designed, implemented, and tested is outlined in the form of equation (5.25). In the first place, the components involved will be specified, and then the security functionality, in general, will be presented. IoT results from the convergence and evolution of ubiquitous or pervasive computing, internet protocols, sensing technologies, and embedded systems. These technologies form an ecosystem where the real and digital worlds are in continuous symbolic interaction.

5.5.2. Hybrid Resource Constrained K-Nearest Neighbor Classifier (HRCKNN)

The K-Nearest Neighbors (K-NN) algorithm is a classification algorithm that labels each data item based on the label of the data closest to it. For this, the variable k is defined, corresponding to the number of closest neighbors chosen to carry out the classification. Depending on this variable, we have some predictions or others. Unlike other supervised learning algorithms, K-NN does not learn with the training data, but rather the learning occurs with the test data once the training data has been memorized. These algorithms are known as lazy algorithms, and they allow several problems to be solved simultaneously since the objective function is approximated locally for each element. The algorithm does not learn from a model but instead uses the data to generate

an answer only when a prediction is requested. As a disadvantage, the computational cost is very high due to all the training data storage.

In this thesis, a hybrid K-NN Classification approach has been proposed, namely HRCKNN and MLADCF, which work simultaneously in an IoT environment. This double approach is a solution to the resource constrained IoT environment. HRCKNN works with the proposed mathematical model MLADCF to tackle the problem of resource constrained IoT networks. The HRCKNN is the beginning of the proposed model, as the data has been further classified at the Edge/Fog level in the process. The overview of the HRCKNN is shown in Figure 5.15.

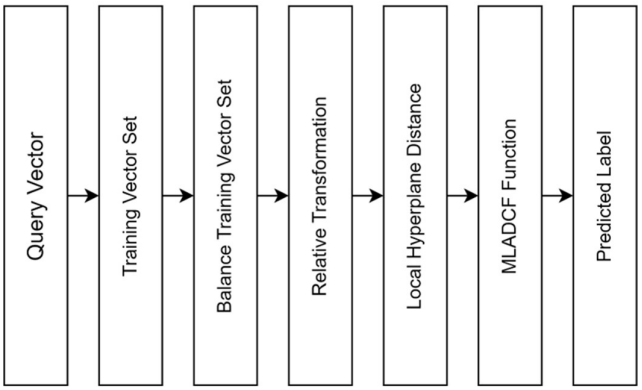


Figure 5.15: HRCKNN Classifier Overview.

HRCKNN works in two phases. HRCKNN first trains with the training data and helps the network tackle the problem of resource constraints. The first phase identifies the IoT devices at the device level, which can process the data. This helps the HRCKNN inform the groups of the IoT nodes.

To understand HRCKNN, let us consider the following assumptions. Let 'v' is the query vector. F is the training feature vector where F= ($f, f_2, f_3, f_4 \ldots\ldots\ldots\ldots f_n$), X is the set of labels with respect to F. is the class where j ∈ {1,2,3……. k}. The size of the neighborhood is denoted by λ. Φ denotes the Euclidean distance ED. Moreover, is the Euclidean distance between the query vector and nearest neighbor. In the first step, a local environment is created by HRCKNN.

$$F' = \mathrm{U_j}\ \mathrm{F_j}(\lambda, v) \tag{5.26}$$

$$F_j(\lambda, v) = \{f_i \in C_j \mid \Phi(f_I, v) \leq \Phi_j^{\lambda} \quad (5.27)$$

In equation (5.26), the inclusion of the training set can be written as

$$F'' = F' \cup \{v\} \quad (5.28)$$

The HRCKNN is capable of calculating the distance between f and F''.

$$\{\Phi(f_1, f) \ldots. \Phi(f_{n-1}, f), \Phi(v, f)\} \quad (5.29)$$

Where $n' = \lambda k + 1$ is the new training set F''.The second step of HRCKNN calculates the distances for all the classes i.e., k-local hyperplane. The relative transformation is also adopted to construct the relative space in equation (5.28). In the final step, the HRCKNN calculated the distance of k local hyperplane. A local hyperplane is constructed as follows

$$\Phi_j^{\lambda}(v) = = \{h | h = f' + \sum_{i=1}^{F} \alpha_t (f_t - f'), \alpha_t \in \xi\} \quad (5.30)$$

$$f' = \frac{1}{\lambda} \sum_{t=1}^{\lambda} f_t \quad (5.31)$$

The HRCKNN allows the query feature vector to calculate the distance of local hyperplane with respect to the nth local hyperplane. The formula can be written as follows

$$\Phi(H_j^{\lambda}(v), v) = \min_{\alpha_t} || v - f' - \sum_{i=1}^{\lambda} \alpha_t (f_t - f') || \quad (5.32)$$

Where $\Phi(H_j^{\lambda}(v), v)$ is the local hyperplane distance, and the solution of the below equation (5.33) will give the value of α_t.

$$(U. U') . A = U'. (v - f') \quad (5.33)$$

The above equation (5.33) can also be written as follows

$$\Phi(H_j^{\lambda}(v), v) = \min_{\alpha_t} \{ || v - f' - \sum_{t=1}^{\lambda} \alpha_t (f_t - f') || + \beta \sum_{t=1}^{\lambda} \lambda_t^2 \quad (5.34)$$

Where A= ($\alpha_1, \alpha_2, \alpha_3$ ... ... α_λ) and the composed matrix of vector $f_t - f'$ is U which is an n × λ matrix. The HRCKNN adoptrs the mathematical model to summarize the local hyperplane distances. Hence creating the groups of the nodes that equally fall in the category of resources constrained and vice versa.

5.5.3. Training Data

The training data, also known as "training data set," is the data utilized to train a machine learning model. The effectiveness and accuracy of our machine learning model are directly influenced by the quality of the training data. Consequently, data scientists invest a substantial amount of time in tasks such as data cleaning, debugging, and data wrangling. Training data is an indispensable component in artificial intelligence and machine learning, enabling the development of accurate, efficient, and fully functional machine learning models. Training data plays a crucial role in machine learning algorithms, serving as the foundation for model development. Without training data, machines would lack the necessary understanding and knowledge to perform tasks effectively. Similar to how individuals require specific training for their respective jobs, machines necessitate a substantial body of information to fulfil their designated purposes and yield appropriate outcomes.

5.5.4 Stages in MLADCF

The proposed framework consists of 5 stages as shown in Figure 5.16. The first stage has n number of sensor nodes present. These nodes senses data from the environment and stores the raw data in the device storage. The cluster head namely CH1 as show in Figure 5.16 receives the data from these sensors. The working of the proposed algorithm starts from this stage 1. The proposed algorithm decides whether to push the data to the stage 2 or should process the data at stage 1 i.e., the device layer.

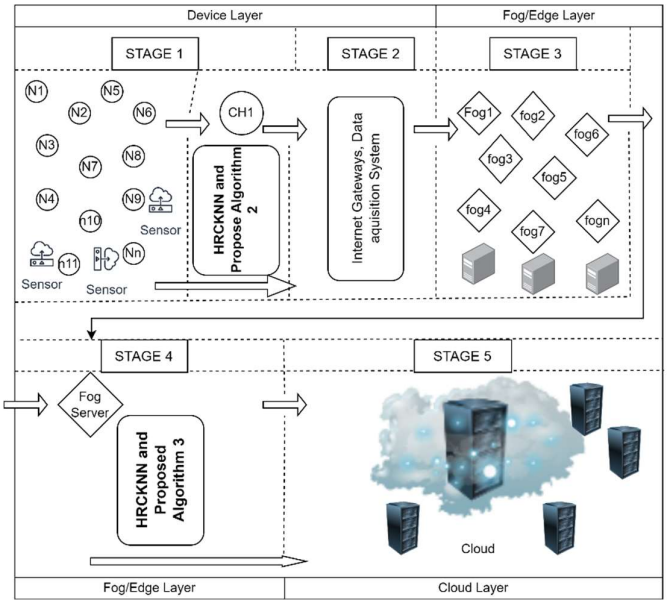


Figure 5.16: Stages of MLADCF.

As the data is pushed to the stage 3 which is the Fog/Edge layer. The Fog/Edge layer consisted of n number of Fog nodes. The Proposed algorithm 2 works of the Fog server node and decides whether to push the data to the cloud or to process in this stage.

5.5.5 Proposed Algorithm 1

Algorithm 1

Require: Sensor node and cluster head.

- *Start*
- *Calculate the value for* d_i
- *If Det* $\sum_{i=1}^{\gamma} d_i > S_k$ *, then push the data to Fog 1*
  Else,
- *Go to algorithm 2*
- *Stop*

5.5.6 Proposed Algorithm 2

Algorithm 2

Require: Source node and edge Node.

- Node N_1, senses data from time t_1 *to* t_2

- Evaluate value of effectiveness θ_f / d_i of the IoT device
- Evaluate the size of the packets in the J^{th} data cluster
- The size of the data packet reaches a maximum size of the S_k
- Node N_1, acts as a Source node, SRC_i with Id, SRC_{id}

for *each source node, SRC_i* **do**

> *Every SRC_i there is a cluster of available nearby nodes, N_1 to act as service nodes, S_{in}*
> *SRC_i broadcasts a request for nodes where, $d_i \leq S_k$*
> *N nodes accepts the request of SRC_i to act as slave nodes, $S_{i1}, S_{i2} \ldots . S_{in}$*
> *S_{in} selected by SRC_i based on Availability quotient, A where: A=f (E_i, SRC_i)*
> *Divide the data of data size, X into data chunks X_{i1}, X_{i2}, X_{i3},...,X_{in}*
> *SRC_i ends X_{i1}, X_{i2}, X_{i3},...,X_{in} to the selected slave nodes, S_{i1}, S_{i2}, S_{i3},...,S_{in}*
> *Each of N selected Sin sends the data to the edge, T_1*
> *Data is collected by E_1 and acknowledgement is sent to SRC_i*

end for

Data is deleted on SRC_i and S_{in}
The further processing of data is performed on the T_1
The chunks of data received from S_{in} are accumulated on the T_1
Data is stored for each SRC_i and maintained on the edge node, T_1

5.5.7 Proposed Algorithm 3

Algorithm 3

Require: Source node and cloud.

- Fog F_1, receives data from form Ch1 over a time period of t
- Evaluate value of effectiveness θ_f/d_i of the Fog Node
- Evaluate the size of the packets in the J^{th} data cluster
- The size of the data packet reaches a maximum size of the S_k
- Fog F_1, acts as a Source node, SRC_i with Id, SRC_{id}

for *each source node, SRC_i* **do**

> *Every SRC_i there is a cluster of available nearby Fog nodes, F_1 to act as service nodes, S_{in}*
> *SRC_i broadcasts a request for nodes where, $d_i \leq S_k$*
> *N nodes accepts the request of SRC_i to act as slave nodes, $S_{i1}, S_{i2} \ldots . S_{in}$*
> *S_{in} selected by SRC_i based on Availability quotient, A where: A=f (E_{nr}, SRC_i)*
> *Divide the data of data size, X into data chunks X_{i1}, X_{i2}, X_{i3},...,X_{in}*

SRC_i ends X_{i1}, X_{i2}, X_{i3},...,X_{in} to the selected slave nodes, S_{i1}, S_{i2}, S_{i3},...,S_{in}
Each of N selected Sin sends the data to the cloud
Data is collected by cloud and acknowledgement is sent to SRC_i

end for

Data is deleted on SRC_i and S_{in}
The further processing of data is performed on the cloud.

5.5.8 Simulation Parameters

Table 5.2: Simulation Parameters.

S. No.	Parameters	Values
1	No. of Service Nodes	500
2	No. of SRC Nodes	100
3	No. of Edge Nodes	5
4	Initial energy of an IoT Service Node	300 mAh
5	Initial energy of Source IoT Node	300 mAh
6	Transmission Range of Service IoT node	40 mtr
7	Transmission Range of Source IoT Node	40 mtr
8	Block Size	256

5.5.9 Performance Evaluation

The comparative analysis is carried out through the following parameters:

• ***Round*****:** The completion of a process is called round. It starts with the sensing of the data and ends till the data is pushed to the edge level.

• ***Energy***: The difference between the total energy of an IoT node and the energy consumed in one round.

$$E_{nr} = E_{nt} - E_{nc} \tag{5.35}$$

where E_{nr} is the remaining energy, E_{nt} is the total energy before the round, and E_{nc} is the energy left after the round. Equation (5.35) can also be written as

$$E_n = \Sigma E_{nr}/\text{nr} \tag{5.36}$$

This is called the average energy of the IoT system where nr is the active nodes.

• ***Storage***: Let S_o is the storage occupied in an IoT node after the round. It can be calculated by the following equation (5.37).

$$S_o = S_t - S_r \tag{5.37}$$

where S_t is the total storage of a node before the round and S_r is storage remaining after the round.

• ***Processing Time***: Processing time can be calculated by adding the time taken by the node in sensing, offloading, aggregating, and storing.

The average energy of the system will be calculated by subtracting the average energy of MLADCF and the average energy of the traditional method.

5.6 Experimental

The heart of an IoT Device is its microcontroller/processor, an (SoC) service on-chip that is responsible for data processing and storage. Some of the common SBCs single board computers are Arduino Uno, Particle Electron, Espressif system ESP8266-01, Rasberry Pi4, Beaglebone black, PICI6C5X, MSP430, Mega AVR, Adafruit feather FONA, Holagram Dash, LinkIT One, Goblin2, PIC 18F 4550, 8051 Microcontrollers, MSP 430, Infincon TRicore, Atmel AVR MC, ATMega328P Processor, etc. It has been observed that the cost of the IoT Nodes is increasing with the increase in resources, i.e., Processing power, Memory, etc. Figure 5.17 and Figure 5.18 below shows the increasing cost while choosing the hardware of higher configuration.

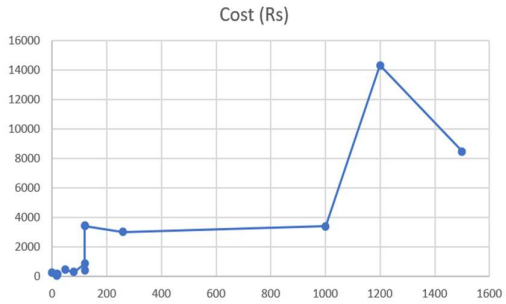


Figure 5.17: Cost Vs Processing Power.

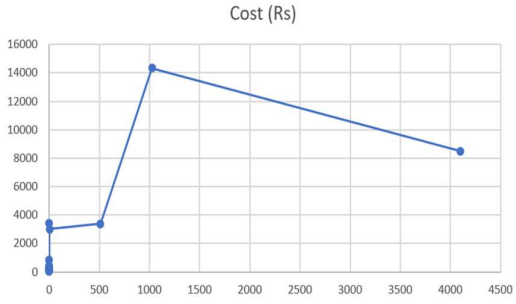


Figure 5.18: Cost Vs Memory.

The essential aspect considered by companies today is the balance between income and costs and the result of the utilities. The cost of the product/device increases with its hardware capability. How should one evaluate or determine the cost of an IoT device and its impact on the network? For this, cost accounting is required, especially the costs related to the device-level hardware. However, it can be affirmed that there is no uniform vision of cost and what should be included under this term. In recent years, ideas about cost and performance have evolved rapidly to become concepts or part of accounts. Costs are an integral part of the production cost, so they cannot be separated.

The selection of the right microcontroller for an IoT device depends on the types and number of sensors incorporated on that device. Figure 5.17 shows the comparison and cost of different Microcontrollers that are commonly used in an IoT device, and Figure 5.18 shows a comparison of the cost of different IoT devices used for different applications.

5.6.1 Setup

The first step involves setting up an IoT environment to collect real-time data in various scenarios. A wireless sensor network comprising multiple sensors and a gateway was created, as depicted in Figure 5.19, Figure 5.20 and Figure 5.21. Within this IoT environment, six sensors have been deployed in a specific area of agricultural land. Each note has an MTS 420 censor board. This WSN comprises a gateway responsible for communication with the surrounding and distributed sensors using the Zigbee module. The IRIS is programmed according to the censor board. MIB520 is used for the Gateway. The communication with the computer is done by the IRIS, which is programmed for the Gateway to act as a communicator. Six sensors have been successfully deployed, providing full coverage of the agricultural field. Mote-config is an application used to program the IRIS and MID 520.

Figure 5.19: IoT Environment.

Figure 5.20: IoT Motes and Gateway.

Figure 5.21: IoT Mote.

This mote-config works in the Windows operating system and configures the sensors. It also provides a user-friendly interface and allows the user to configure the Node ID, RF power, RF channel, and group ID. The nodes can further be enabled over the setup feature on all X mesh-based firmware. High power and low power X mesh applications are available for each sensor board. To upload firmware, a program tab is used by using a gateway. The setup of the hardware is as per the below protocol.

- An ethernet port or USB should connect the Gateway and computer and be powered.
- The Gateway should be attached to the IoT motes.
- The programming. Next should be done while keeping the motes in power off mode.

For all other motes, the XMESH file must be uploaded over it. Moreover, for MDS 420 sensor node, the IRS needs to be programmed for the censor board. After programming all the sensors, the sensors were deployed carefully into the agriculture field and connected to the Gateway successfully. The topology of the sensors is discussed in section 3.11.2 in Figure 5.22, Figure 5.23, and Figure 5.24. The advantage of the scenarios is that it covers all the possibilities that could happen during data collection in the smart agricultural activity. The scenarios would cover all the possibilities and failures of the motes if any of the modes was disconnected due to the battery drain. Therefore, the WSN deployed is highly reliable and scalable. The IoT environment is shown in Figure 5.19, Figure 5.20, and Figure 5.21 has the following sensors in each IoT node that is a temperature sensor, humidity sensor, light sensor, voltage, press petition, node ID, and location X, Y. The data rate of an IoT node shown in Figure 5.21 is 250 KBPS and has a range of 500 meters: the range and data rate were enough for experimenting in this agriculture field. Postgress database was used for logging data into the sensors' database. To analyse this data and find a prediction model, this data was extracted into a.CSV file. The next step involves the cleaning of data in this.CSV file. The. CSV file contains anomalous values and redundant values. Therefore, data pre-processing is needed to clean the data. The. The CSV file also contains high pitch values and extra columns, which can fail to find the patterns and need to be cleaned.

5.6.2 Scenario I, II & III

During the implementation process, three distinct scenarios were encountered. The first scenario, as shown in Figure 5.22, is the scenario I, where data is less in the sense that the node is capable of processing and able to communicate the data without any resource constraints. As shown in Figure 5.22, node5 and node8 have sensed the data and can process the data, hence forwarding the data to node7 and node9, respectively. Fog 1, the edge node, receives the sensed data from its nearest nodes, i.e., node2 and node10.

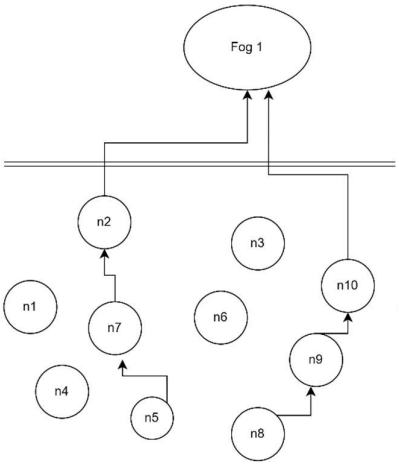


Figure 5.22: Scenario I.

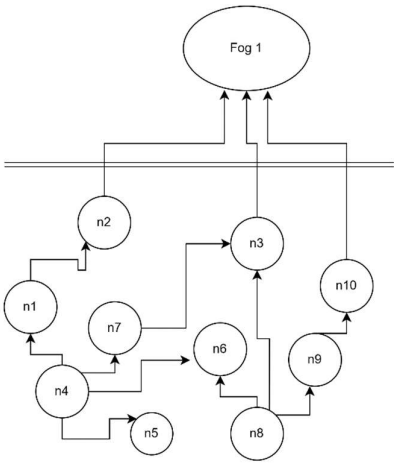


Figure 5.23: Scenario II.

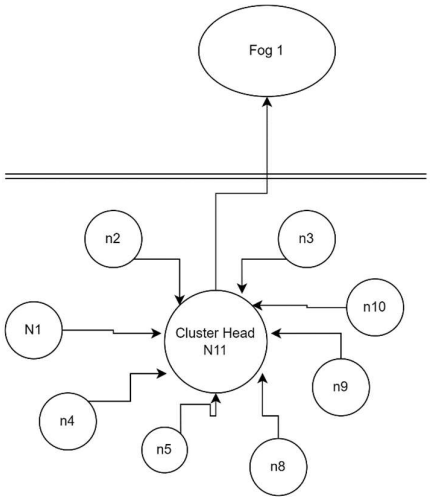


Figure 5.24: Scenario III.

Fog 1, the edge node, receives the data from its nearby slave nodes. Figure 5.23 is an example of scenario II where data is more than the scenario I, and the nodes are not capable of processing the data; therefore, the nodes will break the data into data chunks and push the data for processing the data at nearby nodes and from those nodes the data will be pushed to the Fog from the slave-nodes. As shown in Figure 5.23, node4 and node8 are sensing the data but are not capable of processing the data, dividing the data into data chunks, and sending the data chunks to nearby nodes node1, node7 node5, node6 and node9, node3 respectively.

Scenario II leads us toward scenarios III, which has two different nodes, i.e., homogeneous nodes and heterogeneous nodes. Homogeneous Nodes Have the Same

Capability in processing power, battery, memory, etc. In heterogeneous nodes, a cluster Head has Higher Capability than Sensing Nodes. To add practicality to this experiment, we considered heterogeneous nodes. Scenario III leads to data classification, which, in turn guides us to the proposed framework. The adaptive Machine Learning algorithm is depicted in Figure 5.24 and plays a crucial role in the process. Classification of data is the base for the adaptive ML algorithm, where the IoT nodes will lead us for the IoT network, how to process, state, and communicate the data so that it can reach up to the Fog level.

5.7 Summary

This chapter discusses the importance of machine learning algorithms and the implementation of the proposed algorithms for resource-constrained IoT environment. The Cloud of Things concept provides mechanisms to bring data from IoT devices to the cloud. This environment consists of different layers/levels. The data has to pass through different layers i.e., device layer, Fog/Edge layer and finally cloud layer. As the data is increasing day by day therefore the use of machine learning approaches is very important in order to optimize the resources. The K-Nearest Neighbors (K-NN) algorithm is a classification algorithm that labels each data item based on the label of the data closest to it. This approach is commonly used for IoT data classification. To address the resource constraints, we have devised HRCKNN and incorporated it into the proposed framework as a proposed solution.

CHAPTER 6

PERFORMANCE ANALYSIS AND COMPARISON OF THE PROPOSED WORK WITH THE EXISTING APPROACHES

6.1 Metrics for Evaluation of Model Performance

After adjusting the learning algorithm to perform the task, we have to measure its efficiency, that is, try to extract some measure that informs us of how well (or poorly) it is doing. As in the cases of supervised and unsupervised learning, the objectives sought are very different; the efficiency of some or other algorithms is also usually defined in very different ways.

The case of supervised learning is the most natural and usual. Let us remember that, in this case, we have a set of initial examples on which we perform the learning and from which we know the desired result that the proposed algorithm must return. We want to see if the machine is able, from the trained examples, to generalize the learned behavior so that it is good enough on data not seen a priori, and if so, we say that the machine (model, algorithm) generalizes correctly. Since supervised learning algorithms learn from this data to adjust their internal parameters and return the correct answer, there is little point in measuring the machine's efficiency by bypassing the same data back to it since the information it would give us would be misleadingly optimistic.

6.1.1 Accuracy

It can be defined as the percentage of correct predictions made by the classification model. It is a good metric to use when classes are balanced; that is, the proportion of instances of all classes is similar. However, it is not a reliable metric for data sets that have a class imbalance; that is, the total number of instances of one data class is much less than the total number of instances of another data class.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TN+TP+FN+FP}} \tag{6.1}$$

Accuracy measures the percentage of cases that the model has got right. This is one of the most used and favorite metrics that should be avoided. The problem with accuracy is that it can be misleading; it can make a bad model.

6.1.2 Precision

Indicates, of all the positive predictions, how many are positive. It is defined as the ratio of correct positive predictions to overall positive predictions

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{6.2}$$

Precision is the ratio of correctly predicted positive values to total predicted positive values. This metric highlights the correct positive predictions out of all positive predictions. High precision indicates a low false-positive rate.

6.1.3 Sensitivity/TPR/Recall

Indicates how many are predicted to be positive of all the truly positive values. It is the ratio of correct positive predictions to the total number of positive cases in the data set.

$$\text{TPR} = \text{Sensitivity} = \frac{\text{TP}}{\text{TP+FN}} \tag{6.3}$$

The completeness metric will inform us about the amount that the machine learning model is capable of identifying.

6.1.4 F1 Score

The F1 value combines the precision and recall measurements into a single value. This is handy because it makes comparing the combined accuracy and recall performance between various solutions easier. When avoiding both false positives and false negatives is equally important to the problem, a balance between Precision and Sensitivity is needed. In this case, the metric F1 can be used, which is defined as the harmonic mean between these values.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision+Rec}} \tag{6.4}$$

A machine learning classification model can be used to predict the actual class of the data directly or, much more interestingly, predict its probability of belonging to different classes. The latter gives more control over the output, and a custom threshold can be used to interpret the classifier output, which is often more prudent than building a completely new model if the last one has failed.

6.2 Data Sets

The performance of the proposed model was evaluated using four real-time data sets obtained by creating an IoT environment. The first data set was captured during the daytime with the full battery capacity of the IoT devices. The IoT nodes/sensors were placed in an orchard area for a day. The topology of the sensors is shown in Figure 5.22 in the scenario I. The second data set was captured by keeping the same IoT sensors. The battery was unchanged during the process, and the storage was also intact. While capturing the data, some of the IoT notes ran out of battery and stopped working. Hence this gives us important information about the practicality of the IoT sensors, as these types of situations may arise in an IoT environment. The 3rd dataset falls under the category of scenario II. In this scenario, the batteries of the IoT sensors were replaced with new ones. In this scenario, the IoT sensors were strategically placed to ensure communication between all the sensors. The four data sets were captured by keeping the sensors in scenario III, as shown in Figure 5.24. In this scenario, all the sensors communicate with a cluster head, giving us the new data set for the experiment.

6.3 Performance Comparison of the Proposed Hybrid Model

The selection of the algorithms for comparing the proposed algorithm or model is very important. Therefore, the algorithms commonly used for data classification in an IoT environment have been selected. Consequently, the algorithms chosen are Logistic regression (LR), Naive bias (NB), K-Nearest Neighbor (KNN), Decision trees (DT), Random Forest (RF), and Support vector machines (SVM). The model development was carried out using Spyder IDE and DL libraries. Additionally, Keras, TensorFlow, and scikit-learn packages were utilized to leverage devices for the implementation. The experiments were conducted on a Windows-based operating system, utilizing Python on a system equipped with an Intel i7 processor, 16 GB of RAM, and 1 TB of secondary storage. The study involved four distinct data sets. The results for all the datasets are shown below from Table 6.1 to Table 6.4. Furthermore, the graphs are plotted in Figures 6.1 to Figure 6.8. Four data sets were employed to evaluate the performance of the proposed model. The results for the first data set are shown in Table 6.1. During the first experiment of DS1 (Data set first), it was noted that the Naive Bayes (NB) algorithm exhibited the best execution time; however, it lacked in terms of accuracy. The HRCKNN was average in execution time, but the Accuracy of HRCKNN was the best among all the Machine learning algorithms. The Accuracy was 85 percent, followed by the SVM, which showed the same Accuracy. Nevertheless, the SVM lacks execution time badly, i.e., 9.9 seconds which was the worst among all the algorithms.

The second data set comprises the data for a lesser no of nodes in comparison with dataset 1. As the nodes ran out of power, the rest of the nodes continued to sense the data from the environment. The results of the second dataset are shown in Table 6.2. The execution time was least in the proposed model at 0.004 secs, and it was seen as worst in the case of SVM. The KNN was comparatively better in terms of Precision, Recall, F1 score, and Accuracy. However, its execution time was not good in comparison with HRCKNN. HRCKNN and KNN produced an accuracy of 98 percent. Table 6.3 used data set 3, and Navie Bias was the fastest among all the algorithms but was worst in the Accuracy. Similarly, SVM produces an accuracy of 85 percent but was the slowest and took 9.9 secs in execution. However, HRCKNN proved to be the best in terms of Accuracy, i.e., 85 percent, and was near the best execution time, i.e., 0.036

secs. Data set 4 was generated by scenario III, also called the hybrid Scenario, as this data set was generated when all the sensors were communicating with a common cluster head. For this data set, the results are shown in Table 6.4. In this table, Navie bias has the best execution time but lacks the Accuracy of the data, i.e., only 14 percent. Here HRCKNN has the best Accuracy among all the algorithms. The graphs between the Execution time and Accuracy are plotted and are shown in Figures 6.1 to Figure 6.4. Moreover, the graphs between execution time and sensitivity are plotted in Figures 6.5 to Figure 6.8.

Table 6.1: Performance comparison of machine learning algorithms for DS1.

Algorithm	Execution Time (Sec)	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (avg) (%)
KNN	0.005	83	83	83	80
SVM	9.9	80	80	80	**85**
RF	0.72	88	88	88	79
DT	0.067	80	79	79	76
NB	**0.003**	54	51	48	53
LR	0.68	63	62	62	63
HRCKNN	0.036	**87**	**87**	**87**	**85**

Table 6.2: Performance comparison of machine learning algorithms for DS2.

Algorithm	Execution Time (Sec)	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (avg) (%)
KNN	0.07	**98**	**98**	**98**	**98**
SVM	2.6	96	98	97	97
RF	1.1	**98**	**98**	**98**	**98**
DT	0.06	97	97	97	98
NB	0.005	78	78	78	83
LR	0.85	79	85	81	84
HRCKNN	**0.004**	97	97	97	**98**

Table 6.3: Performance comparison of machine learning algorithms for DS3.

Algorithm	Execution Time (Sec)	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (avg) (%)
KNN	3.24	84	80	82	92
SVM	9.9	30	30	30	48
RF	12.1	85	79	81	92
DT	4.8	83	78	80	91

NB	**0.30**	31	41	29	60
LR	17.11	23	22	17	51
HRCKNN	0.40	82	**84**	**84**	**92**

Table 6.4: Performance comparison of machine learning algorithms for DS4.

Algorithm	Execution Time (Sec)	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (avg) (%)
KNN	0.009	67	69	68	66
SVM	18.12	63	69	65	67
RF	1.03	70	71	70	72
DT	0.07	70	71	71	70
NB	**0.005**	21	23	11	14
LR	3.04	25	28	26	40
HRCKNN	1.2	**77**	**76**	**76**	**76**

Figure 6.1: Accuracy Vs Execution Time for DS1.

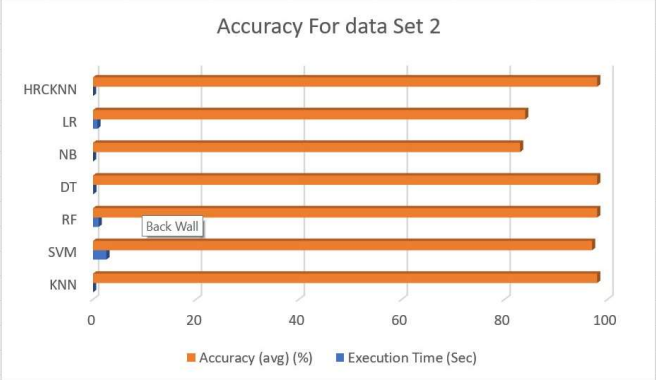


Figure 6.2: Accuracy Vs Execution Time for DS2.

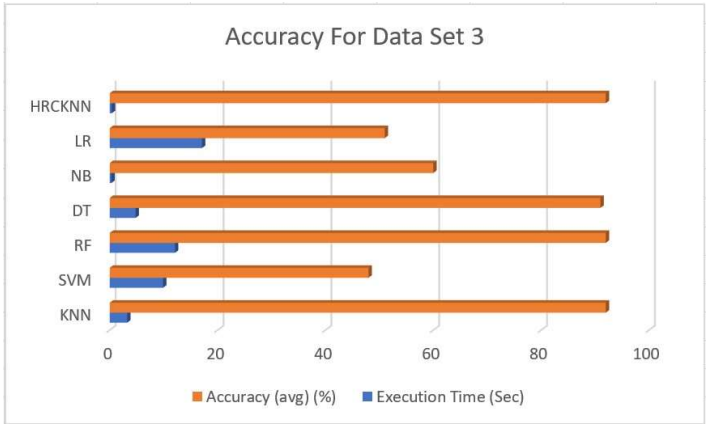


Figure 6.3: Accuracy Vs Execution Time for DS3.

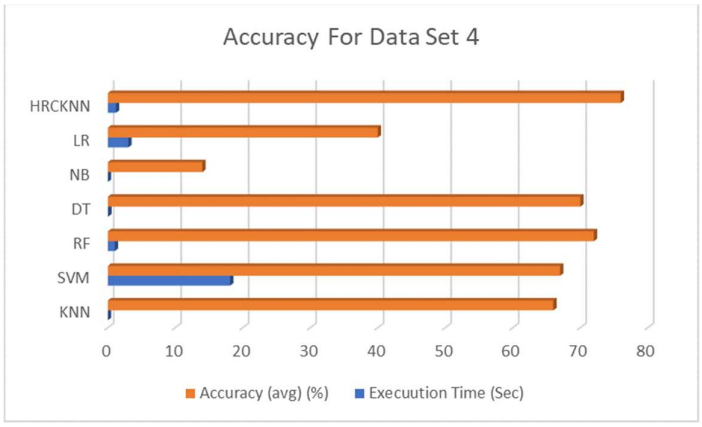


Figure 6.4: Accuracy Vs Execution Time for DS4.

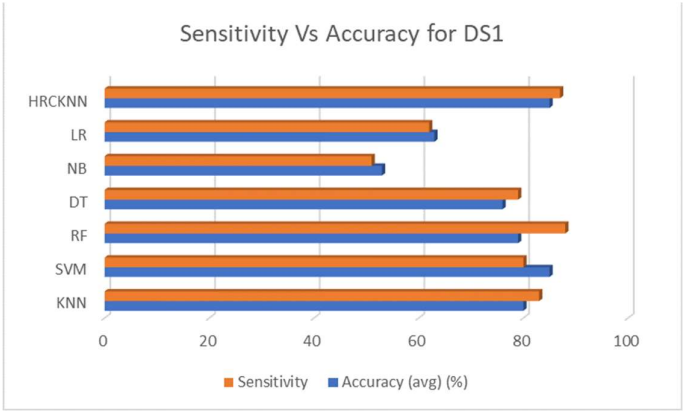


Figure 6.5: Sensitivity Vs Execution Time for DS1.

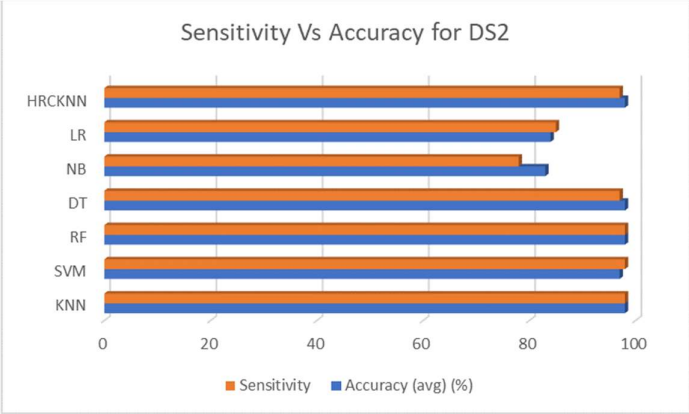


Figure 6.6: Sensitivity Vs Execution Time for DS2.

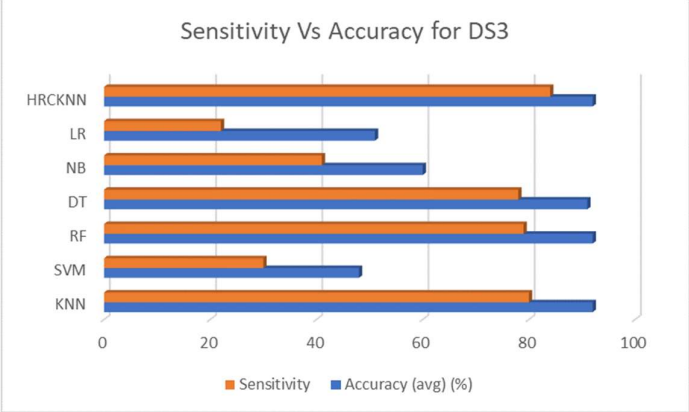


Figure 6.7: Sensitivity Vs Execution Time for DS3.

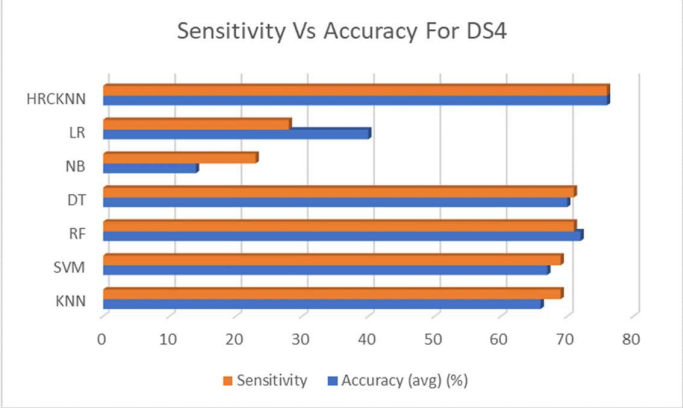


Figure 6.8: Sensitivity Vs Execution Time for DS4.

6.4 MLADCF Results

The MLADCF was tested through simulation to assess its performance and effectiveness, deploying IoT sensors, and creating an IoT environment for live data capturing. The performance evaluation of MLADCF included metrics such as energy consumption, alive nodes, processing time, and storage. The results were presented in the form of graphs. The node's energy during the simulation process is shown in Figure 6.9. The energy of the node is calculated by equation (5.35) and equation (5.36). The results show that the node's energy is better in the proposed MLADCF. In the case of a traditional IoT environment, it was observed that the energy was getting exhausted during the process in comparison to the MLADCF. The results shown in Figure 6.9 show an improvement in the node's energy by 11.9%. All scenarios in the IoT environment were considered, and a comparison was made to identify the factors causing energy drop in an IoT node. In scenarios where nodes engage in frequent communication rather than just transmitting data, we observed an 80% energy drop in the IoT nodes.

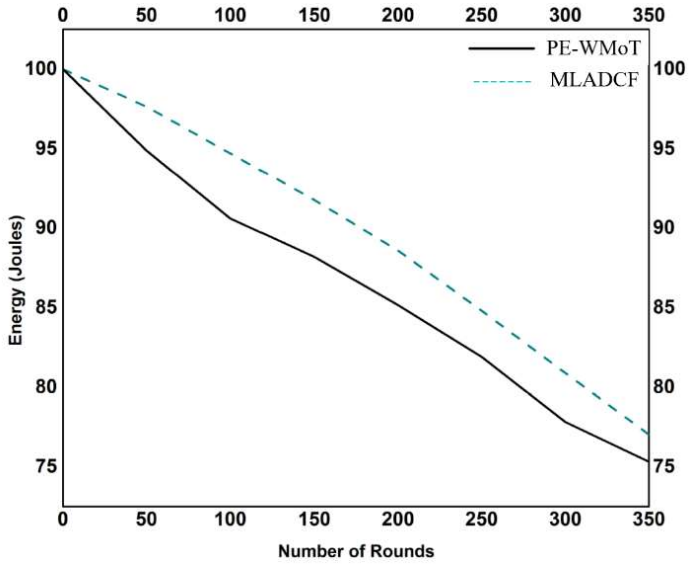


Figure 6.9: No. of Rounds Vs Energy.

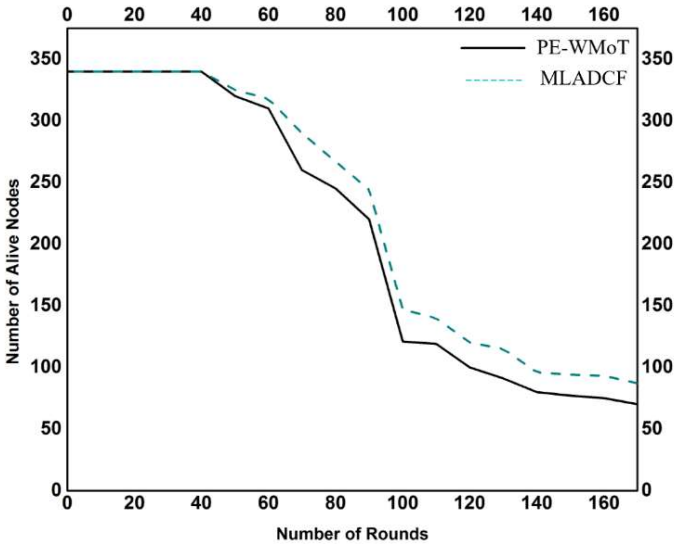


Figure 6.10: No. of Rounds Vs No. of Alive Nodes.

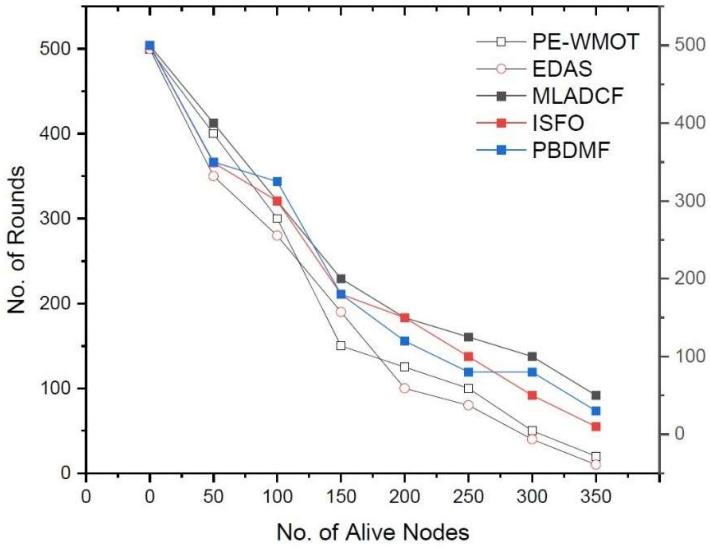


Figure 6.11: No. of Rounds Vs No. of Alive Nodes.

As every node has a major impact on the overall IoT environment, each node is responsible for the life of the network. If a node is overloaded with the incoming data traffic, its energy affects the overall network. The observation from Figure 6.10 and Figure 6.11 leads us towards the results of the MLADCF. In Figure 6.10, we can see that the number of alive nodes is more at all the points than the traditional framework,

hence improving the network's life. Moreover, the improvement was calculated to be 24 percent.

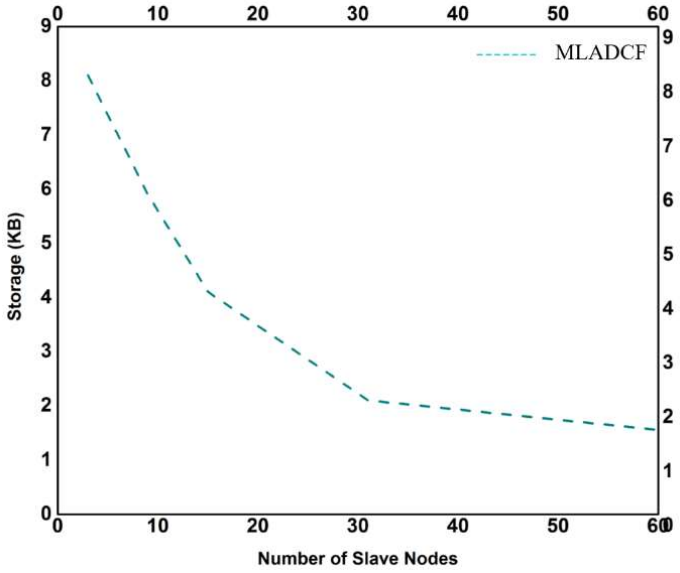


Figure 6.12: No. of Slave Nodes Vs Storage.

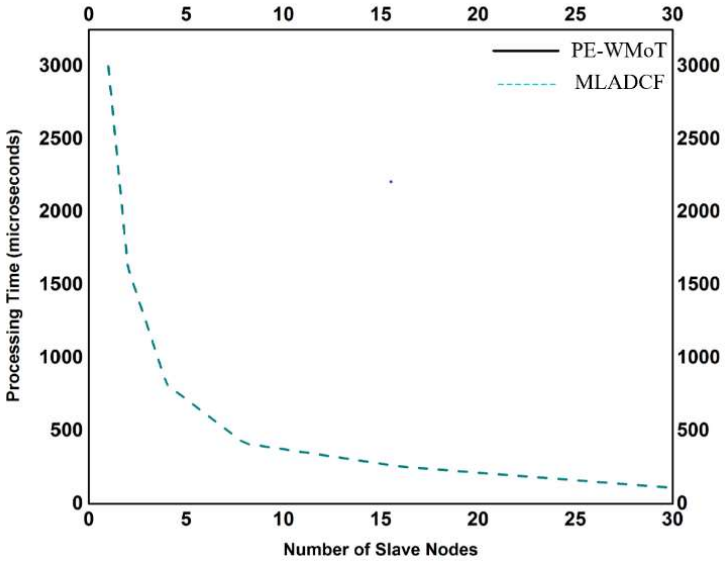


Figure 6.13: No. of Slave Nodes Vs Processing Time.

The increased data traffic from the millions of sensors required more storage capacity. In the proposed framework MLADCF, the storage problem has been taken into account, and it is observed from Figure 6.11 as the nodes start capturing data, the storage used is further reduced in the MLADCF. The number of secondary nodes for the data

processing is increased. The results for the processing time are shown in Figure 6.12. It is observed from the graph that the processing time decreases if the number of nodes is increased.

6.5 Summary

In this chapter, the proposed model is implemented for the resource-constrained IoT environment. The results were compared with existing machine learning approaches. The comparison revealed that the proposed model outperformed all the existing approaches for IoT data classification, making it the optimal choice for the resource-constrained IoT environment. Comprehensive experiments were conducted as part of the study to evaluate the performance of the proposed model in comparison to the existing machine learning algorithms i.e., KNN, SVM, RF, DT, NB, and LR used for data classification. The primary difference between this thesis and other researchers' experiments lies in the specific focus on optimizing the overall energy consumption of the IoT network. Therefore, resulting in the improvement of the IoT network's energy and the number of alive nodes remining in the IoT network. Two distinct experiments were conducted, as illustrated in Chapter 5 and Chapter 6. The first experiment was conducted to evaluate the mathematical model and proposed hybrid KNN algorithm. The output of the first experiment serves as the input for the second experiment, where two separate algorithms were proposed for device level and edge level, respectively. The end result is shown in the form of graph i.e., No. of rounds vs energy, no. of rounds vs no. of alive nodes, no. of slave nodes vs storage, and no. of slave nodes vs processing time. For the experiment, four data sets were utilized, all of which were captured in real-time by creating an IoT environment, as illustrated in Figure 5.22, Figure 5.23, and Figure 5.24.

CHAPTER 7

CONCLUSION AND FUTURE DIRECTIONS

7.1 Conclusion

The optimization of the resources is important in resource-constrained IoT applications. Implementation of machine learning approaches to meet the expectations of the application is required. Offloading data from IoT devices through the IoT network to the nearest Fog node is used in many applications, but the data offloading depends on many factors to optimize resources. The optimization is possible through mechanisms at different layers of the IoT environment. This Thesis discusses the layered approach for data classification. Based on the device capability, it is decided where to process, store, and communicate the sensed data within the permissible limits of delay in real-time IoT applications. In the proposed framework for the optimization of the resources, the machine learning algorithm is used to decide at which layer of the network and device-level within the IoT network, i.e., IoT device, node, cluster node or Fog node, the processing, storage, communication of the data is to be taken. The data classification helps to optimize resource utilization within the IoT framework.

This thesis has provided valuable insights into the challenges encountered by IoT systems and how emerging technologies like Edge and Fog Computing aim to address these challenges. By reducing network latencies and cloud service costs, these technologies seek to enhance the efficiency and performance of IoT systems. However,

it is important to note that deploying such systems is not a straightforward process, and determining where to begin can be daunting due to the heterogeneity of platforms, operating systems, hardware, cloud solutions, and communication protocols involved. Thus, there is a need for interoperable solutions that can integrate these diverse elements seamlessly. The study also observed how different IoT scenarios can contribute to standardization, making it easier for these systems to be adopted widely. Overall, the proposed methodology has successfully achieved its intended objectives.

7.2 Future Directions

Machine learning plays a pivotal role in advancing organizations along the business intelligence maturity curve by shifting from retrospective descriptive analysis to forward-thinking, autonomous decision-making. Although the technology has been around for decades, recent innovations and product advancements have sparked renewed interest among companies. Machine learning-based analytics solutions operate in real time, introducing a new dimension to business intelligence. While these models continue to provide valuable insights and reports to senior decision-makers, real-time analytics empower frontline employees to enhance performance on an hourly basis.

As a subset of artificial intelligence, machine learning involves training systems with specialized algorithms to analyze, learn from, and generate predictions and recommendations based on extensive datasets. These predictive models can adapt to new data without human intervention, continuously improving accuracy and consistency in their results and decisions. This iterative process enhances system intelligence, enabling the discovery of hidden perspectives, historical relationships, trends, and new opportunities across various domains, from customer preferences to supply chain optimization and even oil exploration. Importantly, machine learning empowers organizations to leverage big data effectively and integrate capabilities like IoT analytics. Machine learning is a robust analytical technology already available to businesses. A wide range of commercial and open-source machine learning solutions, accompanied by a thriving developer ecosystem, is readily accessible. It is likely that your organization is already leveraging machine learning in some capacity, such as for spam filtering. Embracing machine learning and analytics more broadly empowers

faster responses to dynamic situations and unlocks greater value from rapidly expanding data repositories.

As discussed in previous chapters, the development of current societies will depend to a certain extent on implementing the Internet model of the things in device generation. However, despite its advantages, the IoT has some drawbacks that lead to rethinking the processes and design of objects and their interconnectivity to adapt to certain daily needs and enable the expected interaction effectively. This situation implies that, soon, the creation of the devices and services based on the development of the IoT model should change, focusing on the paradigms that define the consciousness of the text as well as the application of the ubiquitous computing, thereby allowing the design of resources according to situations individuals in different social spheres. Identifying the aspects that determine the interaction process of a device with a user or a device with another through the Internet gives the possibility of generating new designs that circumscribe the concept of the Internet of Things.

"MLADCF" the proposed model is based on a completely new framework" in which the need to know the environment or context, transcending a "Great concept Design" to "Design in Context." This context-awareness can be controlled by open hardware such as Arduino, Netduino, Galileo, Raspberry, or similar. Implementing communication that allows managing data generated in the Cloud closes the object-user-awareness system of the context-Internet of Things. Finally, it is worth clarifying the use of technology in design within this model. Design for the IoT framework must involve ML technology during the process. However, it should not be considered use of this as the end but as the means; that is, it should be designed for the technology and not with technology. It has been seen that the increase in interaction between people and objects at any time is empowering more and more the development of current societies. In this sense, the Internet of things is already a reality present worldwide in all areas of social development. Nevertheless, being an emerging technology, the IoT model still has certain shortcomings that make it not entirely applicable. This implies a clear evolution of the Internet and changes in the conceptualization of everyday life and its context, which forces us to prepare for future interaction design and devices. A paradigm shift is necessary for object design and services based on the development of the IoT model, which should focus fundamentally on the approaches that define the

awareness of the context and the application of ubiquitous computing, thereby allowing the design of resources according to particular situations. That situation sets the tone for Carrying out work and research that will benefit the development of the Internet of Things and societies.

In Chapter 6, it was observed that the HRCKNN algorithm exhibited inconsistency when applied to different data sets. The execution time, precision, and accuracy were additional and were not best for all the parameters. However, the hybrid quality of the model manages to outplay all other approaches. But there is a scope for improvement in the accuracy of the data set, which has a lower recall value. Similarly, for the framework MLADCF, the overall energy of the system can be improved by using an enhanced version of the approach.

REFERENCES

[1] M. Marjani, Fariza. Nasaruddin, Abdullah. Gani, A. Karm, I.Abaker, A. Siddiqa, "Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges," in *IEEE Access*, vol. 5, pp. 5247-5261, 2017.

[2] M. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, & A. Sheth, "Machine learning for internet of things data analysis: a survey", *Digital Communications and Networks*, vol. 4, pp. 161-175. 2018.

[3] S. Zahoor and R. Mir, "Resource management in pervasive Internet of Things: A survey", *Journal of King Saud University - Computer and Information Sciences*, vol. 33, no. 8, pp. 921-935, 2021.

[4] A Souza and J Amazonas, "An Outlier Detect Algorithm using Big Data Processing and Internet of Things Architecture", *Procedia Computer Science*, vol. 52, pp. 1010-1015, 2015.

[5] V. Agneeswaran, P. Tonpay and J. Tiwary, "Paradigms for Realizing Machine Learning Algorithms", *Big Data*, vol. 1, no. 4, pp. 207-214, 2013.

[6] T. Cover and P. Hart, "Nearest neighbor pattern classification", *IEEE Transactions on Information Theory*, vol. 13, pp. 21-27, 1967.

[7] S. Li, L. Xu and S. Zhao, "5G Internet of Things: A survey", *Journal of Industrial Information Integration*, vol. 10, pp. 1-9, 2018.

[8] A. Čolaković and M. Hadžialić, "Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues", *Computer Networks*, vol. 144, pp. 17-39, 2018.

[9] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep Learning for IoT Big Data and Streaming Analytics: A Survey", *IEEE Communications Surveys & Tutorials*, vol. 20, pp. 2923-2960, 2018.

[10] "AWS IoT| Industrial, Consumer, Commercial, Automotive | Amazon Web Services", *Amazon Web Services, Inc.*, 2022. [Online]. Available: https://aws.amazon.com/iot/. [Accessed: 11- May- 2022].

[11] "Introduction to IoT", *CISCO*, 2022. [Online]. Available: https://www.netacad.com/courses/iot/introduction- iot. [Accessed: 11- May- 2022].

[12] García C. Cortes and V. Vapnik, "Support-vector networks", *Machine Learning*, vol. 20, no. 3, pp. 273, 1995.

[13] C. Rudin, KL. Wagstaff, "Machine learning for science and society", *Machine Learning*, vol. 95, pp. 1–9, 2014.

[14] S. N. Swamy and S. R. Kota, "An Empirical Study on System Level Aspects of Internet of Things (IoT)," *IEEE Access*, vol. 8, pp. 188082-188134, 2020.

[15] O. farooq, A. sharma and P. Mishra, "Big data in mobile and pervasive computing", in *International Conference on Intelligent Sustainable Systems*, 2017, pp. 1068-1072.

[16] J. Hegde and B. Rokseth, "Applications of machine learning methods for engineering risk assessment – A review", *Safety Science*, vol. 122, pp. 104492, 2020.

[17] D. Guinard, V. Trifa, S. Karnouskos, P. Spiess, and D. Savio, "Interacting with the SOA-Based Internet of Things: Discovery, Query, Selection, and On-Demand Provisioning of Web Services", *IEEE Transactions on Services Computing*, vol. 3, pp. 223-235, 2010.

[18] D. Vukobratovic, D. Jakovetic, V. Skachek, D. Bajovic, D. Sejdinovic, and G. Karabulut Kurt, "CONDENSE: A Reconfigurable Knowledge Acquisition Architecture for Future 5G IoT", *IEEE Access*, vol. 4, pp. 3360-3378, 2016.

[19] V. Gunjan, J. Zurada, B. Raman, and G. Gangadharan, "Modern Approaches in Machine Learning and Cognitive Science: A Walkthrough", *Springer International Publishing*, vol. 885, pp. VIII, 245, 2020.

[20] X. Zhang, Y. Zhao and W. Liu, "A Method for Mapping Sensor Data to SSN Ontology", *International Journal Of U- And E- Service, Science and Technology (IJUNESST)*, vol. 8, pp. 303-316, 2015.

[21] "Machine Learning on Edge Brings AI to IoT", *IoT Times*, 2022. [Online]. Available: https://iot.eetimes.com/machine-learning-on-edge-brings-ai-to-iot/. [Accessed: 11- May- 2022].

[22] A. Likas, N. Vlassis, and J. Verbeek, "The global k-means clustering algorithm", *Pattern Recognition*, vol. 36, pp. 451-461, 2003.

[23] D. Singh and C. Reddy, "A survey on platforms for big data analytics"," *Journal of Big Data*, vol. 2, p. 8, 2014.

[24] R. Bro, and A. Smilde, "Principal component analysis", *Anal. Methods*, vol. 6, pp. 2812-2831, 2014.

[25] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *in Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.

[26] Y. Qin, Q. Sheng, N. Falkner, S. Dustdar, H. Wang, & A. Vasilakos, "When things matter: A survey on data-centric internet of things", *Journal of Network and Computer Applications*, vol. 64, pp. 137-153, 2016.

[27] T. R. N and R. Gupta, "A Survey on Machine Learning Approaches and Its Techniques," *IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, Bhopal, India, 2020, pp. 1-6.

[28] V. Shende, "A Survey on Different Machine Learning Approaches", *International Journal for Research in Applied Science and Engineering Technology (IJRASET)*, vol. 7, pp. 1620-1624, 2019.

[29] P. Ni, C. Zhang and Y. Ji, "A hybrid method for short-term sensor data forecasting in internet of things", *11th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, vol. 2, pp. 369-373, 2014.

[30] X. Ma, Y. Wu, Y. Wang, F. Chen, and J. Liu, "Mining smart card data for transit riders' travel patterns", Transportation Research Part C: *Emerging Technologies*, vol. 36, pp. 1-12, 2013.

[31] W. Derguech, E. Bruke and E. Curry, "An autonomic approach to real-time predictive analytics using open data and internet of things", *Ubiquitous Intelligence and Computing IEEE 11th Intl Conf.*, vol. 10, pp. 204–211, 2014.

[32] R. Luss, and A. D'Aspremont, "Predicting abnormal returns from news using text classification", *Quantitative Finance*, vol. 15, pp. 999-1012, 2012.

[33] W. Han, Y. Gu, Y. Zhang and L. Zheng, "Data driven quantitative trust model for the Internet of Agricultural Things," *2014 International Conference on the Internet of Things (IOT),* Cambridge, MA, USA, 2014, pp. 31-36.

[34] J. L. Berral-García, "A quick view on current techniques and machine learning algorithms for big data analytics," *2016 18th International Conference on Transparent Optical Networks (ICTON)*, Trento, Italy, 2016, pp. 1-4.

[35] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "Erratum to: A survey of machine learning for big data processing," *EURASIP J. Adv. Signal Process.,* vol. 67, no. 1, 2016.

[36] S. Athmaja, M. Hanumanthappa and V. Kavitha, "A survey of machine learning algorithms for big data analytics," *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS),* Coimbatore, India, 2017, pp. 1-4.

[37] P. Y. Wu, C. -W. Cheng, C. D. Kaddi, J. Venugopalan, R. Hoffman and M. D. Wang, "Omic and Electronic Health Record Big Data Analytics for Precision Medicine," *in IEEE Transactions on Biomedical Engineering,* vol. 64, no. 2, pp. 263-273, Feb. 2017.

[38] P.Sreeja and M. Mondal, "A Survey on the overview of Internet of Things (IoT)", *International Journal of Recent Trends in Engineering and Research (IJRTER)*, vol. 4, pp. 251-257, 2018.

[39] H. Cui and C. Chang, "Deep Learning Based Advanced Spatio-Temporal Extraction Model in Medical Sports Rehabilitation for Motion Analysis and Data Processing", *IEEE Access*, vol. 8, pp. 115848-115856, 2020.

[40] T. J. Sheng, M. S. Islam, N. Misran, M. H. Baharuddin, H. Arshad, M. R. Islam, M. E. H. Chowdhury, H. Rmili and M. T. Islam, "An Internet of Things Based Smart Waste Management System Using LoRa and Tensorflow Deep Learning Model", *IEEE Access*, vol. 8, pp. 148793-148811, 2020.

[41] L. Jiang and C. Wu, "A Massive Multi-Modal Perception Data Classification Method Using Deep Learning Based on Internet of Things", *Int J Wireless Inf Networks*, vol. 27, pp. 226–233, 2020.

[42] B. M. ElHalawany, K. Wu, and A. B. Zaky, “Deep learning-based resources allocation for internet-of-things deployment underlaying cellular networks,” *Mob. Netw. Appl*., vol. 25, no. 5, pp. 1833–1841, 2020.

[43] F. Chen, Z. Fu and Z. Yang, “Wind power generation fault diagnosis based on deep learning model in internet of things (IoT) with clusters”, *Cluster Comp*., vol. 22, pp. 14013–14025, 2019.

[44] F. Alqahtani, Z Al-Makhadmeh, A. Tolba and wael said, “Internet of things-based urban waste management system for smart cities using a Cuckoo Search Algorithm”, *Cluster Comp*., vol. 23, pp. 1769–1780, 2020.

[45] F. Pacheco, E. Exposito, and M. Gineste, “A framework to classify heterogeneous Internet traffic with Machine Learning and Deep Learning techniques for satellite communications,” *Comput. netw*., vol. 173, no. 107213, p. 107213, 2020.

[46] H. Naeem, Farhanullah, M. Naeem, S. Khalid, D. Vasan, S. Jabbar, S. Saeed, “Malware detection in industrial internet of things based on hybrid image visualization and deep learning model,” *Ad Hoc Netw*., vol. 105, no. 102154, p. 102154, 2020.

[47] J. Siryani, B. Tanju and T. J. Eveleigh, “A Machine Learning Decision-Support System Improves the Internet of Things Smart Meter Operations”, *IEEE Internet of Things Journal (IoT)*, vol. 4, pp. 1056-1066, 2017.

[48] R. Zhao, X. Wang, J. Xia, and L. Fan, “Deep reinforcement learning based mobile edge computing for intelligent Internet of Things,” *Phys. commun*., vol. 43, no. 101184, p. 101184, 2020.

[49] C. Mo and W. Sun, “Point-by-point feature extraction of artificial intelligence images based on the Internet of Things,” *Comput. Commun*., vol. 159, pp. 1–8, 2020.

[50] M. Mastalerz, A. Malinowski, S. Kwiatkowski, A. Śniegula, and B. Wieczorek, “Passenger BIBO detection with IoT support and machine learning techniques for intelligent transport systems”, *Procedia Computer Science*, vol. 176, pp. 3780–3793, 2020.

[51] A. Castañeda-Miranda and V. M. Castaño-Meneses, “Internet of things for smart farming and frost intelligent control in greenhouses,” *Comput. Electron. Agric*., vol. 176, no. 105614, pp. 105614, 2020.

[52] M. W. Rahman, R. Islam, A. Hasan, N. I. Bithi, M. M. Hasan, and M. M. Rahman, “Intelligent waste management system using deep learning with IoT,” *J. King Saud Univ. - Comput. Inf. Sci*., 2020.

[53] Z. Shen, A. Shehzad, S. Chen, H. Sun, and J. Liu, “Machine Learning Based Approach on Food Recognition and Nutrition Estimation”, *Procedia Computer Science*, vol. 174, pp. 448–453, 2020.

[54] G. Alfian, M. Syafrudin, U. Farooq, M. Maarif, M. Alex, N. Latif, J. Lee, J. Rhee, “Improving efficiency of RFID-based traceability system for perishable food by utilizing IoT sensors and machine learning model,” *Food Control*, vol. 110, no. 107016, p. 107016, 2020.

[55] A. Alarifi and A. Tolba, “Optimizing the network energy of cloud assisted internet of things by using the adaptive neural learning approach in wireless sensor networks”, *Computers in Industry*, vol. 106, pp. 133–141, 2019.

[56] N. G. Rezk, E. E.-D. Hemdan, A.-F. Attia, A. El-Sayed, and M. A. El-Rashidy, “An efficient IoT based smart farming system using machine learning algorithms,” *Multimed. Tools Appl*., vol. 80, no. 1, pp. 773–797, 2021.

[57] H. Zhang, Z. Fu and K. Shu, “Recognizing Ping-Pong Motions Using Inertial Data Based on Machine Learning Classification Algorithms”, *IEEE Access*, vol. 7, pp. 167055-167064, 2019.

[58] D. Goularas and S. Kamis, "Evaluation of Deep Learning Techniques in Sentiment Analysis from Twitter Data," *2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML)*, 2019, pp. 12-17.

[59] F. Montori, L. Bedogni and L. Bononi, "A Collaborative Internet of Things Architecture for Smart Cities and Environmental Monitoring”, *IEEE Internet of Things Journal (IoT)*, vol. 5, pp. 592-605, 2018.

[60] S. K. Mydhili, S. Periyanayagi, S. Baskar, P. M. Shakeel, and P. R. Hariharan, “Machine learning based multi scale parallel K-means++ clustering for cloud

assisted internet of things," *Peer Peer Netw. Appl*., vol. 13, no. 6, pp. 2023–2035, 2020.

[61] H. Jin, "Data processing model and performance analysis of cognitive computing based on machine learning in Internet environment", *Soft Comput*., vol. 23, pp. 9141–9151, 2019.

[62] J. Diaz Rozo, C. Bielza and P. Larrañaga, "Clustering of Data Streams with Dynamic Gaussian Mixture Models: An IoT Application in Industrial Processes", *IEEE Internet of Things Journal (IoT)*, Vol. 5, pp. 3533-3547, 2018.

[63] A. Ibrahim, A. Eltawil, Y. Na and S. El-Tawil, "A Machine Learning Approach for Structural Health Monitoring Using Noisy Data Sets", *IEEE Transactions on Automation Science and Engineering*, vol. 17, pp. 900-908, 2020.

[64] J. Dass, V. Sarin and R. N. Mahapatra, "Fast and Communication-Efficient Algorithm for Distributed Support Vector Machine Training", *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, pp. 1065-1076, 2019.

[65] A. Aljumah, A. Kaur, M. Bhatia, and T. Ahamed Ahanger, "Internet of things-fog computing-based framework for smart disaster management," *Trans. emerg. telecommun. technol. (ETT)*, 2020.

[66] J. Li, H. Tao, L. Shuhong, S. Salih, J. Zain, L. Yankun, G. Vivekananda, M. Thanjaivadel, "Internet of things assisted condition-based support for smart manufacturing industry using learning technique," *Comput. Intell*., vol. 36, no. 4, pp. 1737–1754, 2020.

[67] P. Sharmila, J. Baskaran, C. Nayanatara, and R. Maheswari, "A hybrid technique of machine learning and data analytics for optimized distribution of renewable energy resources targeting smart energy management", *Procedia Computer Science*, vol. 165, pp. 278–284, 2019.

[68] N. Alghanmi, R. Alotaibi and S. M. Buhari, "HLMCC: A Hybrid Learning Anomaly Detection Model for Unlabeled Data in Internet of Things", *IEEE Access*, vol. 7, pp. 179492-179504, 2019.

[69] S. K. Lakshmanaprabu, K. Shankar, A. Khanna, D. Gupta, J. P. C. Rodrigues, P. R. Pinheiro, V. H. Albuquerque, "Effective Features to Classify Big Data Using Social Internet of Things, *IEEE Access*, vol. 6, pp. 24196-24204, 2018.

[70] S. Lin, C. Chen and T. Lee, "A Multi-Label Classification with Hybrid Label-Based Meta-Learning Method in Internet of Things", *IEEE Access*, vol. 8, pp. 42261-42269, 2020.

[71] G. Casolla, S. Cuomo, V. S. d Cola and F. Piccialli, "Exploring Unsupervised Learning Techniques for the Internet of Things", *IEEE Transactions on Industrial Informatics*, Vol. 16, pp. 2621-2628, 2020.

[72] H. Yao, P. Gao, J. Wang, P. Zhang, C. Jiang and Z. Han, "Capsule Network Assisted IoT Traffic Classification Mechanism for Smart Cities", *IEEE Internet of Things Journal (IoT)*, vol. 6, pp. 7515-7525, 2019.

[73] A. H. Wahla, L. Chen, Y. Wang, R. Chen and F. Wu, "Automatic Wireless Signal Classification in Multimedia Internet of Things: An Adaptive Boosting Enabled Approach", *IEEE Access*, vol. 7, pp. 160334-160344, 2019.

[74] S. Egea, A. Rego Mañez, B. Carro, A Sánchez-Esguevillas and J Lloret, "Intelligent IoT Traffic Classification Using Novel Search Strategy for Fast-Based-Correlation Feature Selection in Industrial Environments", *IEEE Internet of Things Journal (IoT)*, vol. 5, pp. 1616-1624, 2018.

[75] A. Sivanathan, H. H. Gharakheli, F. Loi, A. Radford, C. Wijenayake, A, Vishwanath, V. Sivaraman, "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics", *IEEE Transactions on Mobile Computing*, vol. 18, pp. 1745-1759, 2019.

[76] O. G. Manzanilla-Salazar, F. Malandra, H. Mellah, C. Wetté and B. Sansò, "A Machine Learning Framework for Sleeping Cell Detection in a Smart-City IoT Telecommunications Infrastructure", *IEEE Access*, vol. 8, pp. 61213-61225, 2020.

[77] F. C. Ribeiro, R. T. S. Carvalho, P. C. Cortez, V. H. C. De Albuquerque and P. P. R. Filho, "Binary Neural Networks for Classification of Voice Commands from Throat Microphone", *IEEE Access*, vol. 6, pp. 70130-70144, 2018.

[78] P. Wei and B. Wang, "Food image classification and image retrieval based on visual features and machine learning," *Multimed. Syst*., 2020.

[79] A. Ksentini, M. Jebalia, and S. Tabbane, "IoT/cloud-enabled smart services: A review on QoS requirements in fog environment and a proposed approach based

on priority classification technique," *International Journal of Communication Systems*, vol. 34, no. 2, 2021.

[80] J. Huang, L. Zhu, Q. Liang, B. Fan and S. Li, "Efficient Classification of Distribution-Based Data for Internet of Things," *in IEEE Access*, vol. 6, pp. 69279-69287, 2018.

[81] A. Mulahuwaish, K. Gyorick, K. Z. Ghafoor, H. S. Maghdid, and D. B. Rawat, "Efficient classification model of web news documents using machine learning algorithms for accurate information," *Computers and Security*, vol. 98, no. 102006, p. 102006, 2020.

[82] M. Raikar, M. Mulla, M. S. Meena, S. Shetti and M. Karanandi, "Data Traffic Classification in Software Defined Networks (SDN) using supervised-learning", *Procedia Computer Science*, vol. 171, pp. 2750–2759, 2020.

[83] M. Wang and Q. Zhang, "Optimized data storage algorithm of IoT based on cloud computing in distributed system", *Computer Communications*, vol. 157, pp. 124–131, 2020.

[84] R. Hou, Y. Kong, B. Cai and H. Liu, "Unstructured big data analysis algorithm and simulation of Internet of Things based on machine learning", *Neural Comput & Applic*, vol. 32, pp. 5399–5407, 2020.

[85] A. Singh, G. S. Aujla, S. Garg, G. Kaddoum and G. Singh, "Deep-Learning-Based SDN Model for Internet of Things: An Incremental Tensor Train Approach", *IEEE Internet of Things Journal (IoT)*, vol. 7, pp. 6302-6311, 2020.

[86] I.C. Hsu and C.-C. Chang, "Integrating machine learning and open data into social Chatbot for filtering information rumor," *J. Ambient Intell. Humaniz. Comput*., vol. 12, no. 1, pp. 1023–1037, 2021.

[87] Y. Shi, X. Zhang, Q. Hu and H. Cheng, "Data recovery algorithm based on generative adversarial networks in crowd sensing Internet of Things", *Personal and Ubiquitous Computing*, vol. 16, no. 5, pp. 819–844, 2020.

[88] J. Azar, A. Makhoul, R. Couturier, and J. Demerjian, "Robust IoT time series classification with data compression and deep learning," *Neurocomputing*, vol. 398, pp. 222–234, 2020.

[89] S. Mohan, C. Thirumalai and G. Srivastava, “Effective Heart Disease Prediction Using Hybrid Machine Learning Techniques”, *IEEE Access*, vol. 7, pp. 81542-81554, 2019.

[90] T. Muhammed, R. Mehmood, A. Albeshri and I. Katib, “UbeHealth: A Personalized Ubiquitous Cloud and Edge-Enabled Networked Healthcare System for Smart Cities”, *IEEE Access*, vol. 6, pp. 32258-32285, 2018.

[91] H. Tang and Z. Hu, “Research on Medical Image Classification Based on Machine Learning”, *IEEE Access*, vol. 8, pp. 93145-93154, 2020.

[92] G. Chen, C. Ding, Y. Li, X. Hu, X. Li, L. Ren, X. Ding, P. Tian and W. Xue, “Prediction of Chronic Kidney Disease Using Adaptive Hybridized Deep Convolutional Neural Network on the Internet of Medical Things Platform”, *IEEE Access*, vol. 8, pp. 100497-100508, 2020.

[93] M. K. Hasan, M. A. Alam, D. Das, E. Hossain and M. Hasan, “Diabetes Prediction Using Ensembling of Different Machine Learning Classifiers”, *IEEE Access*, vol. 8, pp. 76516-76531, 2020.

[94] A. Khamparia, D. Gupta, de Albuquerque, A. Kumar and R. H. Jhaveri, “Internet of health things-driven deep learning system for detection and classification of cervical cells using transfer learning”, *The Journal of Supercomputing*, vol. 76, pp. 8590–8608, 2020.

[95] A. Ed-daoudy and K. Maalmi, “A new Internet of Things architecture for real-time prediction of various diseases using machine learning on big data environment,” *Journal of Big Data*, vol. 6, no. 1, 2019.

[96] A. Souri, M. Y. Ghafour, A. M. Ahmed, F. Safara, A. Yamini, and M. Hoseyninezhad, “A new machine learning-based healthcare monitoring model for student’s condition diagnosis in Internet of Things environment,” *Soft Comput*., vol. 24, no. 22, pp. 17111–17121, 2020.

[97] M. Hosseinzadeh et al., "A multiple multilayer perceptron neural network with an adaptive learning algorithm for thyroid disease diagnosis in the internet of medical things", *The Journal of Supercomputing*, vol. 77, no. 4, pp. 3616-3637, 2020.

[98] P. Kaur, R. Kumar and M. Kumar, “A healthcare monitoring system using random forest and internet of things (IoT)”, *Multimed Tools Appl*., vol. 78, pp. 19905–19916, 2019.

[99] L. Devi and V. Kalaivani, “Machine learning and IoT-based cardiac arrhythmia diagnosis using statistical and dynamic features of ECG”, *The Journal of Supercomputing*. vol. 76, pp. 6533–6544, 2020.

[100] O. Irshad, M. U. G. Khan, R. Iqbal, S. Basheer and A. K. Bashir, “Performance optimization of IoT based biological systems using deep learning”, *Computer Communications*, vol. 155, pp. 24–31, 2020.

[101] A. Khamparia, D. Gupta, V. de Albuquerque, A. Sangaiah and R. Jhaveri, "Internet of health things-driven deep learning system for detection and classification of cervical cells using transfer learning", *The Journal of Supercomputing*, vol. 76, no. 11, pp. 8590-8608, 2020.

[102] Z. Wang and Z. Gao, “Analysis of real-time heartbeat monitoring using wearable device Internet of Things system in sports environment,” *Comput. Intell*., vol. 12337, pp. 1-18, 2020.

[103] R. Kesavan and S. Arumugam, “Adaptive deep convolutional neural network-based secure integration of fog to cloud supported Internet of Things for health monitoring system,” *Trans. emerg. telecommun. technol. (ETT)*, vol. 31, no. 10, 2020.

[104] J. Wu and L. Patrono, “A fog-based ubiquitous exercise healthcare monitoring framework for smart cities,” *Internet Technol. Lett*., vol. 4, no. 1, 2021.

[105] P. Verma and S. K. Sood, “Fog Assisted-IoT Enabled Patient Health Monitoring in Smart Homes”,IEEE*, Internet of Things Journal (IoT)*, vol. 5, pp. 1789-1796, 2018.

[106] Y. Huang, Q. Zhao, Q. Zhou and W. Jiang, “Air Quality Forecast Monitoring and Its Impact on Brain Health Based on Big Data and the Internet of Things”, *IEEE Access*, Vol. 6, pp. 78678-78688, 2018.

[107] R. Chatterjee, T. Maitra, S. K. Hafizul Islam, M. M. Hassan, A. Alamri, and G. Fortino, “A novel machine learning based feature selection for motor imagery EEG signal classification in Internet of medical things environment,” *Future Gener. Comput. Syst. (FGCS)*, vol. 98, pp. 419–434, 2019.

[108] W. Xuan and G. You, “Detection and diagnosis of pancreatic tumor using deep learning-based hierarchical convolutional neural network on the internet of medical things platform,” *Future Gener. Comput. Syst (FGCS)*., vol. 111, pp. 132–142, 2020.

[109] O. Deperlioglu, U. Kose, D. Gupta, A. Khanna, and A. K. Sangaiah, “Diagnosis of heart diseases by a secure Internet of Health Things system based on Autoencoder Deep Neural Network,” *Comput. Commun*., vol. 162, pp. 31–50, 2020.

[110] M. Elhoseny, G.-B. Bian, S. K. Lakshmanaprabu, K. Shankar, A. K. Singh, and W. Wu, “Effective features to classify ovarian cancer data in internet of medical things,” *Computer networks*, vol. 159, pp. 147–156, 2019.

[111] A. Ghoneim, G. Muhammad, and M. S. Hossain, “Cervical cancer classification using convolutional neural networks and extreme learning machines,” *Future Gener. Comput. Syst. (FGCS)*, vol. 102, pp. 643–649, 2020.

[112] M. A. Hossain, R. Ferdousi, and M. F. Alhamid, “Knowledge-driven machine learning based framework for early-stage disease risk prediction in edge environment,” *J. Parallel Distrib. Comput*., vol. 146, pp. 25–34, 2020.

[113] A. Samy, H. Yu and H. Zhang, “Fog-Based Attack Detection Framework for Internet of Things Using Deep Learning”, *IEEE Access*, vol. 8, pp. 74571-74585, 2020.

[114] M. Shen, X. Tang, L. Zhu, X. Du and M. Guizani, “Privacy Preserving Support Vector Machine Training Over Blockchain Based Encrypted IoT Data in Smart Cities”, *IEEE Internet of Things Journal (IoT)*, vol. 6, pp. 7702-7712, 2019.

[115] S. Venkatraman, B. Surendiran. and Arun Raj Kumar, “Spam e-mail classification for the Internet of Things environment using semantic similarity approach”, *The Journal of Supercomputing*, Vol. 76, pp. 756–776, 2020.

[116] M. Habib, I. Aljarah and H. Faris, “A Modified Multi-Objective Particle Swarm Optimizer-Based Lévy Flight: An Approach Toward Intrusion Detection in Internet of Things”, *Arab J Sci Eng*., vol. 45, pp. 6081–6108, 2020.

[117] S. Zeadally and M. Tsikerdekis, “Securing Internet of Things (IoT) with machine learning,” *Int. J. Commun. Syst*., vol. 33, no. 1, p. e4169, 2020.

[118] X. Dong, C. Dong, Z. Chen, Y. Cheng, and B. Chen, “BotDetector: An extreme learning machine-based Internet of Things botnet detection model,” *Trans. emerg. telecommun. Technol. (ETT)*, 2020.

[119] A. Mubarakali, K. Srinivasan, R. Mukhalid, S. C. B. Jaganathan, and N. Marina, “Security challenges in internet of things: Distributed denial of service attack detection using support vector machine-based expert systems,” *Comput. Intell*., vol. 36, no. 4, pp. 1580–1592, 2020.

[120] F. Wang, S. Yang, Q. Li, and C. Wang, “An internet of things malware classification method based on mixture of experts neural network,” *Trans. emerg. telecommun. technol. (ETT)*, 2020.

[121] O. Salman, I. H. Elhajj, A. Chehab, and A. Kayssi, “A machine learning based framework for IoT device identification and abnormal traffic detection,” *Trans. emerg. telecommun. technol. (ETT)*, 2019.

[122] W. Ma, “Analysis of anomaly detection method for Internet of things based on deep learning,” *Trans. emerg. telecommun. technol. (ETT)*, vol. 31, no. 12, 2020.

[123] D. Reddy, H. Behera, J. Nayak, P. Vijayakumar, B. Naik and P. Singh, "Deep neural network-based anomaly detection in Internet of Things network traffic tracking for the applications of future smart cities", *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 7, p. 4121, 2020.

[124] M. Sun and R. Yang, “An efficient secure k nearest neighbor classification protocol with high-dimensional features”, *Int J Intell Syst*., vol. 35, pp. 1791-1813, 2020.

[125] Y. Otoum, D. Liu, and A. Nayak, “DL-IDS: a deep learning–based intrusion detection framework for securing IoT,” *Trans. emerg. telecommun. technol. (ETT)*, 2019.

[126] E. M. Karanja, S. Masupe, and M. G. Jeffrey, “Analysis of internet of things malware using image texture features and machine learning techniques,” *Internet of Things*, vol. 9, no. 100153, p. 100153, 2020.

[127] Y. Chen, “Mining of instant messaging data in the Internet of Things based on support vector machine”, *Computer Communications*, vol. 154, pp. 278–287, 2020.

[128] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, “Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city,” *Future Gener. Comput. Syst. (FGCS)*, vol. 107, pp. 433–442, 2020.

[129] B. Amma and Selvakumar, “Anomaly detection framework for Internet of things traffic using vector convolutional deep learning approach in fog environment,” *Future Gener. Comput. Syst. (FGCS)*, vol. 113, pp. 255–265, 2020.

[130] S. Trilles, A. González-Pérez, and J. Huerta, “A comprehensive IoT node proposal using open hardware. A smart farming use case to monitor vineyards,” *Electronics (Basel)*, vol. 7, no. 12, p. 419, 2018.

[131] S. N. Swamy and S. R. Kota, "An Empirical Study on System Level Aspects of Internet of Things (IoT)," in *IEEE Access*, vol. 8, pp. 188082-188134, 2020.

[132] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum and N. Ghani, "Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations," in *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2702-2733, third quarter 2019.

[133] Y. Li *et al*., "Toward Location-Enabled IoT (LE-IoT): IoT Positioning Techniques, Error Sources, and Error Mitigation," in *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4035-4062, 15 March15, 2021.

[134] X. Hu, X. Li, E. C. H. Ngai, J. Zhao, V. C. M. Leung and P. Nasiopoulos, "Health Drive: Mobile Healthcare Onboard Vehicles to Promote Safe Driving," 2015 *48th Hawaii International Conference on System Sciences*, 2015, pp. 3074-3083.

[135] Mohamed, Adham, Mohamed Mostafa M. Fouad, Esraa Elhariri, Nashwa El-Bendary, Hossam M. Zawbaa, Mohamed Tahoun, and Aboul Ella Hassanien. “RoadMonitor: An Intelligent Road Surface Condition Monitoring System.” *In Advances in Intelligent Systems and Computing*, 2015, pp. 377–87. Cham: Springer International Publishing.

[136] T. Gabber and A. E. Hassanien, “An Overview of Self-Protection and Self-Healing in Wireless Sensor Networks,” in *Bio-inspiring Cyber Security and Cloud Services: Trends and Innovations*, Springer, 2014, pp. 185–202.

[137] F. M.M and H. A.E, "Key Pre-distribution Techniques for WSN Security Services", 70th ed., Berlin: Springer, 2014, pp. 265–283.

[138] S. Madden and M. J Franklin, "Fjording the Stream: An Architecture for Queries Over Streaming Sensor Data", in *Proceedings 18th International Conference on Data Engineering*, 2002, p. 0555.

[139] A. Mahmood, K. Shi, S. Khatoon and M. Xiao, "Data Mining Techniques for Wireless Sensor Networks: A Survey", *International Journal of Distributed Sensor Networks*, vol. 9, no. 7, p. 406316, 2013.

[140] D. Yao, C. Yu, H. Jin and Q. Ding, "Human mobility synthesis using matrix and tensor factorizations", *Information Fusion*, vol. 23, pp. 25-32, 2015.

[141] S. Chen, H. Xu, D. Liu, B. Hu and H. Wang, "A Vision of IoT: Applications, Challenges, and Opportunities with China Perspective", *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 349-359, 2014.

[142] H. Hsieh, K. Chang, L. Wang, J. Chen and H. Chao, "ScriptIoT: A Script Framework for and Internet-of-Things Applications", *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 628-636, 2016.

[143] A. P. Castellani, M. Dissegna, N. Bui and M. Zorzi, "WebIoT: A web application framework for the internet of things," *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2012, pp. 202-207.

[144] J. Kiljander et al., "Semantic Interoperability Architecture for Pervasive Computing and Internet of Things", *IEEE Access*, vol. 2, pp. 856-873, 2014.

[145] J. Gubbi, R. Buyya, S. Marusic and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions", *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645-1660, 2013.

[146] G. C. Fox, S. Kamburugamuve and R. D. Hartman, "Architecture and measured characteristics of a cloud-based internet of things," *2012 International Conference on Collaboration Technologies and Systems (CTS)*, 2012, pp. 6-12.

[147] Q. Wu, G. Ding, Z. Du, Y. Sun, M. Jo and A. Vasilakos, "A Cloud-Based Architecture for the Internet of Spectrum Devices Over Future Wireless Networks", *IEEE Access*, vol. 4, pp. 2854-2862, 2016.

[148] D. Mazza, D. Tarchi and G. Corazza, "A Unified Urban Mobile Cloud Computing Offloading Mechanism for Smart Cities", *IEEE Communications Magazine*, vol. 55, no. 3, pp. 30-37, 2017.

[149] A. Munir, P. Kansakar and S. Khan, "IFCIoT: Integrated Fog Cloud IoT: A novel architectural paradigm for the future Internet of Things.", *IEEE Consumer Electronics Magazine*, vol. 6, no. 3, pp. 74-82, 2017.

[150] A. Brogi and S. Forti, "QoS-Aware Deployment of IoT Applications Through the Fog", *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1185-1192, 2017.

[151] Li, Wei, Igor Santos, Flavia C. Delicato, Paulo F. Pires, Luci Pirmez, Wei Wei, Houbing Song, Albert Zomaya, and Samee Khan, "System modelling and performance evaluation of a three-tier Cloud of Things", *Future Generation Computer Systems*, vol. 70, pp. 104-125, 2017.

[152] S. Sarkar and S. Misra, "Theoretical modelling of fog computing: a green computing paradigm to support IoT applications", *IET Networks*, vol. 5, no. 2, pp. 23-29, 2016.

[153] A. Gupta and R. Jha, "A Survey of 5G Network: Architecture and Emerging Technologies", *IEEE Access*, vol. 3, pp. 1206-1232, 2015.

[154] C. Sarkar, A. Nambi S. N., R. Prasad, A. Rahim, R. Neisse and G. Baldini, "DIAT: A Scalable Distributed Architecture for IoT", *IEEE Internet of Things Journal*, vol. 2, no. 3, pp. 230-239, 2015.

[155] Morgan, Cindy. "RFC 7452: Architectural Considerations in Smart Object Networking | Internet Architecture Board", *Iab.org*, 2022. [Online]. Available: https://www.iab.org/2015/03/20/rfc-7452-architectural-considerations-in-smart-object-networking/. [Accessed: 11- May- 2022].

[156] *Cocoa.ethz.ch*, 2022. [Online].Available:https://cocoa.ethz.ch/downloads/2014/01/1524_D1.3_Architectural_Reference_Model_update.pdf.[Accessed:11-May- 2022].

[157] A. Al-Fuqaha, A. Khreishah, M. Guizani, A. Rayes and M. Mohammadi, "Toward better horizontal integration among IoT services", *IEEE Communications Magazine*, vol. 53, no. 9, pp. 72-79, 2015.

[158] H. Hada and J. Mitsugi, "EPC based internet of things architecture," *2011 IEEE International Conference on RFID-Technologies and Applications*, 2011, pp. 527-532.

[159] Miao Yun and Bu Yuxin, "Research on the architecture and key technology of Internet of Things (IoT) applied on smart grid," *2010 International Conference on Advances in Energy Engineering*, 2010, pp. 69-72.

[160] A. P. Castellani, N. Bui, P. Casari, M. Rossi, Z. Shelby and M. Zorzi, "Architecture and protocols for the Internet of Things: A case study," *2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops),* 2010, pp. 678-683.

[161] S. Hong et al., "SNAIL: an IP-based wireless sensor network approach to the internet of things", *IEEE Wireless Communications*, vol. 17, no. 6, pp. 34-42, 2010.

[162] F. Andreini, F. Crisciani, C. Cicconetti and R. Mambrini, "Context-aware location in the Internet of Things," *2010 IEEE Globecom Workshops*, 2010, pp. 300-304.

[163] G. Pujolle, "An Autonomic-oriented Architecture for the Internet of Things," *IEEE John Vincent Atanasoff 2006 International Symposium on Modern Computing (JVA'06*), 2006, pp. 163-168.

[164] A. H. Ngu, M. Gutierrez, V. Metsis, S. Nepal and Q. Z. Sheng, "IoT Middleware: A Survey on Issues and Enabling Technologies," *in IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 1-20, Feb. 2017.

[165] M. A. Razzaque, M. Milojevic-Jevric, A. Palade and S. Clarke, "Middleware for Internet of Things: A Survey*," in IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 70-95, Feb. 2016.

[166] "Research Services, Open Innovation and Breakthrough Technology", *PARC*, 2022. [Online]. Available: https://www.parc.com/. [Accessed: 11- May- 2022].

[167] S. K. Datta, C. Bonnet and N. Nikaein, "An IoT gateway centric architecture to provide novel M2M services," 2014 IEEE World Forum on Internet of Things (WF-IoT), 2014, pp. 514-519.

[168] K. K. Karmakar, V. Varadharajan, S. Nepal and U. Tupakula, "SDN-Enabled Secure IoT Architecture," *in IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6549-6564, 15 April15, 2021

[169] M. Kim, H. Ahn and K. Kim, "Process-Aware Internet of Things: A Conceptual Extension of the Internet of Things Framework and Architecture", *KSII Transactions on Internet and Information Systems*, vol. 10, no. 8, 2016.

[170] J. Stankovic, "Research Directions for the Internet of Things", *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 3-9, 2014.

[171] O. Elijah, T. A. Rahman, I. Orikumhi, C. Y. Leow and M. N. Hindia, "An Overview of Internet of Things (IoT) and Data Analytics in Agriculture: Benefits and Challenges," *in IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3758-3773, Oct. 2018.

[172] S. Chen, H. Xu, D. Liu, B. Hu and H. Wang, "A Vision of IoT: Applications, Challenges, and Opportunities with China Perspective," *in IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 349-359, Aug. 2014.

[173] A. Whitmore, A. Agarwal and L. Da Xu, "The Internet of Things—A survey of topics and trends", *Information Systems Frontiers*, vol. 17, no. 2, pp. 261-274, 2014.

[174] A. Athira, T. D. Devika, K. R. Varsha and S. S. Bose S., "Design and Development of IOT Based Multi-Parameter Patient Monitoring System," *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2020, pp. 862-866.

[175] *Supporting Internet of Things Activities on Innovation Ecosystems*. UNIFY-IoT, 2022, pp. 1-64.

[176] M. Díaz, C. Martín and B. Rubio, "State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing", *Journal of Network and Computer Applications*, vol. 67, pp. 99-117, 2016.

[177] A. Bassi et al., *Enabling Things to Talk*, 1st ed. Berlin: Springer, 2013, pp. 10-349.

[178] "The Global Standard for Machine-to-Machine Communication in SMT Assembly", *Engineering 360*, 2019. [Online]. Available:

https://standards.globalspec.com/std/13365327/ipc-hermes-9852. [Accessed: 11- May- 2022].

[179] "Machine-to-machine (m2m) device and methods for 3gpp and etsi m2m interworking - Patent WO-2014008065-A1 - PubChem", *Pubchem.ncbi.nlm.nih.gov*, 2022. [Online]. Available: https://pubchem.ncbi.nlm.nih.gov/patent/WO-2014008065-A1. [Accessed: 11- May- 2022].

[180] "IEEE Standard for an Architectural Framework for the Internet of Things (IoT)," in IEEE Std 2413-2019, vol., no., pp.1-269, 10 March 2020.

[181] "Global ICT Standardisation Forum", *Gisfi.org*, 2022. [Online]. Available: https://www.gisfi.org. [Accessed: 11- May- 2022].

[182] A. H. Alhamedi, V. Snasel, H. M. Aldosari and A. Abraham, "Internet of things communication reference model," *2014 6th International Conference on Computational Aspects of Social Networks*, 2014, pp. 61-66.

[183] A. Javed, A. Malhi, T. Kinnunen and K. Främling, "Scalable IoT Platform for Heterogeneous Devices in Smart Environments," *in IEEE Access*, vol. 8, pp. 211973-211985, 2020.

[184] O. Kaiwartya et al., "Internet of Vehicles: Motivation, Layered Architecture, Network Model, Challenges, and Future Aspects," *in IEEE Access*, vol. 4, pp. 5356-5373, 2016.

[185] M. R. Palattella et al., "Internet of Things in the 5G Era: Enablers, Architecture, and Business Models," *in IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 510-527, 2016.

[186] "Process-Aware Internet of Things: A Conceptual Extension of the Internet of Things Framework and Architecture", *KSII Transactions on Internet and Information Systems*, vol. 10, no. 8, pp. 4008-4022, 2016.

[187] Z. J. Muhsin, A. Al-Taee, M. A. Al-Taee, W. Al-Nuaimy and A. Al-Ataby, "Mobile workflow management system based on the Internet of Things," *13th International Multi-Conference on Systems, Signals & Devices (SSD)*, 2016, pp. 436-441.

[188] F. Kawsar, G. Kortuem and B. Altakrouri, "Supporting interaction with the Internet of Things across objects, time and space," *Internet of Things (IOT)*, 2010, pp. 1-8.

[189] D. Mazza, D. Tarchi and G. E. Corazza, "A Unified Urban Mobile Cloud Computing Offloading Mechanism for Smart Cities," *in IEEE Communications Magazine*, vol. 55, no. 3, pp. 30-37, March 2017.

[190] S. Sarkar and S. Misra, "Theoretical modelling of fog computing: a green computing paradigm to support IoT applications", *IET Networks*, vol. 5, no. 2, pp. 23-29, 2016.

[191] "CoRE Working Group", *IETF CoRE WG*, 2022. [Online]. Available: https://core-wg.github.io/. [Accessed: 12- May- 2022].

[192] D. Guinard, V. Trifa and E. Wilde, "A resource-oriented architecture for the Web of Things," *Internet of Things (IOT)*, 2010, pp. 1-8.

[193] A. Rizzardi, S. Sicari, D. Miorandi and A. Coen-Porisini, "AUPS: An Open Source AUthenticated Publish/Subscribe system for the Internet of Things", *Information Systems*, vol. 62, pp. 29-41, 2016.

[194] I. Grønbæk, "Architecture for the Internet of Things (IoT): API and Interconnect,"*Second International Conference on Sensor Technologies and Applications (sensorcomm 2008)*, 2008, pp. 802-807.

[195] B. Cheng, D. Zhu, S. Zhao and J. Chen, "Situation-Aware IoT Service Coordination Using the Event-Driven SOA Paradigm," *in IEEE Transactions on Network and Service Management*, vol. 13, no. 2, pp. 349-361, 2016.

[196] P. Spiess et al., "SOA-Based Integration of the Internet of Things in Enterprise Services," *IEEE International Conference on Web Services*, 2009, pp. 968-975.

[197] I. Chen, J. Guo and F. Bao, "Trust Management for SOA-Based IoT and Its Application to Service Composition", *IEEE Transactions on Services Computing*, vol. 9, no. 3, pp. 482-495, 2016.

[198] X. Liu, Q. Sun, W. Lu, C. Wu and H. Ding, "Big-Data-Based Intelligent Spectrum Sensing for Heterogeneous Spectrum Communications in 5G", *IEEE Wireless Communications*, vol. 27, no. 5, pp. 67-73, 2020.

[199] C. Philip Chen and C. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data", *Information Sciences*, vol. 275, pp. 314-347, 2014. Available: 10.1016/j.ins.2014.01.015.

[200] M. Chen, S. Mao, Y. Zhang and V. Leung, "Big Data Generation and Acquisition", Springer, 2014, pp. 19-32.

[201] S. Fosso Wamba, S. Akter, A. Edwards, G. Chopin and D. Gnanzou, "How 'big data' can make big impact: Findings from a systematic review and a longitudinal case study", *International Journal of Production Economics*, vol. 165, pp. 234-246, 2015. Available: 10.1016/j.ijpe.2014.12.031.

[202] A. Rehman, A. Abbasi, N. Islam and Z. Shaikh, "A review of wireless sensors and networks' applications in agriculture", *Computer Standards & Interfaces*, vol. 36, no. 2, pp. 263-270, 2014. Available: 10.1016/j.csi.2011.03.004.

[203] "Choosing the best hardware for your next IoT project", *Developer.ibm.com*, 2022. [Online]. Available: https://developer.ibm.com/articles/iot-lp101-best-hardware-devices-iot-project/. [Accessed: 12- May- 2022].

[204] "IoT Sensors & Actuators | 2019 List of Sensor and Actuator Types and Manufacturers", *Postscapes*, 2022. [Online]. Available: https://www.postscapes.com/iot-sensors-actuators/. [Accessed: 12- May- 2022].

[205] A. Morris and R. Langari, "Sensor Technologies", in *Measurement and Instrumentation*, Elsevier, 2012, pp. 317-345.

[206] F. Meneghello, M. Calore, D. Zucchetto, M. Polese and A. Zanella, "IoT: Internet of Threats? A Survey of Practical Security Vulnerabilities in Real IoT Devices," *in IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8182-8201, Oct. 2019.

[207] O. Said, Z. Al-Makhadmeh and A. Tolba, "EMS: An Energy Management Scheme for Green IoT Environments," *in IEEE Access*, vol. 8, pp. 44983-44998, 2020.

[208] M. Frustaci, P. Pace, G. Aloi and G. Fortino, "Evaluating Critical Security Issues of the IoT World: Present and Future Challenges," in *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2483-2495, 2018.

[209] J. Hwang, A. Aziz, N. Sung, A. Ahmad, F. Le Gall and J. Song, "AUTOCON-IoT: Automated and Scalable Online Conformance Testing for IoT Applications," in *IEEE Access*, vol. 8, pp. 43111-43121, 2020.

[210] L. Gutiérrez-Madroñal, L. La Blunda, M. F. Wagner and I. Medina-Bulo, "Test Event Generation for a Fall-Detection IoT System," in *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6642-6651, Aug. 2019.

[211] O. Said, Y. Albagory, M. Nofal and F. Al Raddady, "IoT-RTP and IoT-RTCP: Adaptive Protocols for Multimedia Transmission over Internet of Things Environments," in *IEEE Access*, vol. 5, pp. 16757-16773, 2017.

[212] M. G. Samaila, J. B. F. Sequeiros, T. Simões, M. M. Freire and P. R. M. Inácio, "IoT-HarPSecA: A Framework and Roadmap for Secure Design and Development of Devices and Applications in the IoT Space," in *IEEE Access*, vol. 8, pp. 16462-16494, 2020.

[213] A. M. Zarca, J. B. Bernabe, A. Skarmeta and J. M. Alcaraz Calero, "Virtual IoT HoneyNets to Mitigate Cyberattacks in SDN/NFV-Enabled IoT Networks," in *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1262-1277, 2020

[214] J. Bhayo, S. Hameed and S. A. Shah, "An Efficient Counter-Based DDoS Attack Detection Framework Leveraging Software Defined IoT (SD-IoT)," in *IEEE Access*, vol. 8, pp. 221612-221631, 2020.

[215] R. Muñoz *et al*., "Integration of IoT, Transport SDN, and Edge/Cloud Computing for Dynamic Distribution of IoT Analytics and Efficient Use of Network Resources," in *Journal of Lightwave Technology*, vol. 36, no. 7, pp. 1420-1428, 2018.

[216] M. N. Khan, A. Rao and S. Camtepe, "Lightweight Cryptographic Protocols for IoT-Constrained Devices: A Survey," in *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4132-4156, 2021.

[217] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood and A. Anwar, "TON_IoT Telemetry Dataset: A New Generation Dataset of IoT and IIoT for Data-Driven Intrusion Detection Systems," in *IEEE Access*, vol. 8, pp. 165130-165150, 2020.

[218] M. Yi, X. Xu and L. Xu, "An Intelligent Communication Warning Vulnerability Detection Algorithm Based on IoT Technology," in *IEEE Access*, vol. 7, pp. 164803-164814, 2019.

[219] M. W. Condry and C. B. Nelson, "Using Smart Edge IoT Devices for Safer, Rapid Response with Industry IoT Control Operations," in *Proceedings of the IEEE*, vol. 104, no. 5, pp. 938-946, 2016.

[220] S. Sathyadevan, K. Achuthan, R. Doss and L. Pan, "Protean Authentication Scheme – A Time-Bound Dynamic KeyGen Authentication Technique for IoT Edge Nodes in Outdoor Deployments," in *IEEE Access*, vol. 7, pp. 92419-92435, 2019.

[221] M. Shafiq, Z. Tian, A. K. Bashir, X. Du and M. Guizani, "CorrAUC: A Malicious Bot-IoT Traffic Detection Method in IoT Network Using Machine-Learning Techniques," in *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3242-3254, 2021.

[222] A. A. Simiscuka, T. M. Markande and G. -M. Muntean, "Real-Virtual World Device Synchronization in a Cloud-Enabled Social Virtual Reality IoT Network," in *IEEE Access*, vol. 7, pp. 106588-106599, 2019.

[223] J. An *et al*., "Toward Global IoT-Enabled Smart Cities Interworking Using Adaptive Semantic Adapter," in *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5753-5765, 2019.

[224] Y. Cheng, Y. Xu, H. Zhong and Y. Liu, "Leveraging Semisupervised Hierarchical Stacking Temporal Convolutional Network for Anomaly Detection in IoT Communication," in *IEEE Internet of Things Journal*, vol. 8, no. 1, pp. 144-155, 2021.

[225] A. H. Ngu, M. Gutierrez, V. Metsis, S. Nepal and Q. Z. Sheng, "IoT Middleware: A Survey on Issues and Enabling Technologies," in *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 1-20, 2017.

[226] S. Chen, H. Xu, D. Liu, B. Hu and H. Wang, "A Vision of IoT: Applications, Challenges, and Opportunities with China Perspective," in *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 349-359, 2014.

[227] D. Xu and H. Zhu, "Secure Transmission for SWIPT IoT Systems with Full-Duplex IoT Devices," in *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10915-10933, 2019.

[228] P. K. Mallapragada, R. Jin, A. K. Jain and Y. Liu, "SemiBoost: Boosting for Semi-Supervised Learning," *in IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 11, pp. 2000-2014, 2009.

[229] A. Guezzaz, Y. Asimi, M. Azrour and A. Asimi, "Mathematical validation of proposed machine learning classifier for heterogeneous traffic and anomaly detection," *in Big Data Mining and Analytics*, vol. 4, no. 1, pp. 18-24, 2021.

[230] Y. Chen, E. Yao and A. Basu, "A 128-Channel Extreme Learning Machine-Based Neural Decoder for Brain Machine Interfaces," *in IEEE Transactions on Biomedical Circuits and Systems*, vol. 10, no. 3, pp. 679-692, 2016.

[231] F. Cui, Q. Cui and Y. Song, "A Survey on Learning-Based Approaches for Modeling and Classification of Human–Machine Dialog Systems," *in IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 4, pp. 1418-1432, 2021.

[232] S. Mirzaei, T. Sidi, C. Keasar and S. Crivelli, "Purely Structural Protein Scoring Functions Using Support Vector Machine and Ensemble Learning," *in ACM Transactions on Computational Biology and Bioinformatics*, vol. 16, no. 5, pp. 1515-1523, 2019.

[233] Q. Xiao, C. Li, Y. Tang and X. Chen, "Energy Efficiency Modeling for Configuration-Dependent Machining via Machine Learning: A Comparative Study," *in IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 2, pp. 717-730, 2021.

[234] N. Reamaroon, M. W. Sjoding, K. Lin, T. J. Iwashyna and K. Najarian, "Accounting for Label Uncertainty in Machine Learning for Detection of Acute Respiratory Distress Syndrome*," in IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 1, pp. 407-415, 2019.

[235] X. Deng, T. Sun, F. Liu and D. Li, "SignGD with error feedback meets lazily aggregated technique: Communication-efficient algorithms for distributed learning," *in Tsinghua Science and Technology*, vol. 27, no. 1, pp. 174-185, 2022.

[236] S. Hussein, P. Kandel, C. W. Bolan, M. B. Wallace and U. Bagci, "Lung and Pancreatic Tumor Characterization in the Deep Learning Era: Novel Supervised and Unsupervised Learning Approaches," *in IEEE Transactions on Medical Imaging*, vol. 38, no. 8, pp. 1777-1787, 2019.

[237] S. Ertekin, L. Bottou and C. L. Giles, "Nonconvex Online Support Vector Machines," *in IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 2, pp. 368-381, 2011.

[238] S. M. Aslam, A. K. Jilani, J. Sultana and L. Almutairi, "Feature Evaluation of Emerging E-Learning Systems Using Machine Learning: An Extensive Survey," *in IEEE Access*, vol. 9, pp. 69573-69587, 2021.

[239] C. Mendes, A. Barcelos and S. José Rigo, "MLtool: A Tool to Automate the Construction, Evaluation, and Selection of Machine Learning Models," *in IEEE Latin America Transactions*, vol. 17, no. 07, pp. 1163-1170, 2019.

[240] S. Zahoor and R. N. Mir, "A parallelization-based data management framework for pervasive IOT applications," *Scalable Computing: Practice and Experience*, vol. 21, no. 3, pp. 463–477, 2020.

[241] M. Salah ud din, M. A. Ur Rehman, R. Ullah, C.-W. Park, D. H. Kim, and B.-seo Kim, "Improving resource-constrained IOT device lifetimes by mitigating redundant transmissions across heterogeneous wireless multimedia of things," *Digital Communications and Networks*, vol. 8, no. 5, pp. 778–790, 2022.

[242] A. F. Raslan, A. F. Ali, A. Darwish, and H. M. El-Sherbiny, "An improved sunflower optimization algorithm for cluster head selection in the internet of things," *IEEE Access*, vol. 9, pp. 156171–156186, 2021.

[243] V. S. Badiger and T. S. Ganashree, "Data Aggregation Scheme for IOT based wireless sensor network through Optimal Clustering Method," *Measurement: Sensors*, vol. 24, p. 100538, 2022.

LIST OF PUBLICATIONS

1. O. Farooq, P. Singh, M. Hedabou, W. Boulila, and B. Benjdira, “Machine learning analytic-based two-staged data management framework for internet of things,” *Sensors*, vol. 23, no. 5, p. 2427, 2023. Available: doi:10.3390/s23052427. [Impact Factor**: 3.847**, SCIE, SCOPUS].

2. O. Farooq and P. Singh, "Data Analytics and Modeling in IoT-Fog Environment for Resource-Constrained IoT-Applications: A Review", *Recent Advances in Computer Science and Communications*, vol. 15, no. 7, pp. 977-1000, 2022. Available: 10.2174/2666255814666210715161630. [SCOPUS].

3. O. Farooq and P. Singh, "Data Management Framework for Resource Constrained IoT Applications", in *Applications of Machine Intelligence in Engineering*, 1st ed., Boca Raton: taylor and francis, 2022, p. 14.

4. O. Farooq and P. Singh, "Machine Learning Approaches for IoT-Data Classification", in *International Conference on Innovative Computing & Communications (ICICC) 2020*, New Delhi, 2020.

5. O. Farooq and P. Singh, “A secure Architecture for Resource Constrained IoT Environment”, in *Journal of University of Shanghai for Science and Technology,* vol. 23, no. 9, pp. 248-264, 2021. Available: https://jusst.org/a-secure-architecture-for-resource-constrained-iot-environment.