UNIVERSITY OF TECHNOLOGY SYDNEY
Faculty of Engineering and Information Technology

CONSENSUS-BASED DATA MANAGEMENT WITHIN FOG COMPUTING FOR THE INTERNET OF THINGS

by

Firas Qais Mahammed Saleh Al-Doghman

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

Doctor of Philosophy

Sydney, Australia

2019

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Firas Qais Mohammed Saleh Al-Doghman declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Electrical and Data Engineering/Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise reference or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Production Note:
Signature: Signature removed prior to publication.

Date: 4/12/2019

ABSTRACT

CONSENSUS-BASED DATA MANAGEMENT WITHIN FOG COMPUTING FOR THE INTERNET OF THINGS

by

Firas Qais Mahammed Saleh Al-Doghman

The Internet of Things (IoT) infrastructure forms a gigantic network of interconnected and interacting devices. This infrastructure involves a new generation of service delivery models, more advanced data management and policy schemes, sophisticated data analytics tools, and effective decision making applications. IoT technology brings automation to a new level wherein nodes can communicate and make autonomous decisions in the absence of human interventions. IoT enabled solutions generate and process enormous volumes of heterogeneous data exchanged among billions of nodes. This results in Big Data congestion, data management, storage issues and various inefficiencies. Fog Computing aims at solving the issues with data management as it includes intelligent computational components and storage closer to the data sources. Often, an IoT-enabled infrastructure is shared among many users with various requirements. Sharing resources, sharing operational costs and collective decision making (consensus) among many stakeholders is frequently neglected. This research addresses an essential requirement for adaptive, autonomous and consensus-based Fog computational solutions which are able to support distributed and in-network schemes and policies. These network schemes and policies need to meet the requirements of many users. In this work, innovative consensus-based computational solutions are investigated. These proposed solutions aim to correlate and organise data for effective management and decision making in Fog. Instead of individual decision making, the algorithms aim to aggregate several decisions into a consensus decision representing a collective agreement, benefiting from the individuals variant knowledge and meeting multiple stakeholders require-

ments. In order to validate the proposed solutions, hybrid research methodology is involved that includes the design of a test-bed and the execution of several experiments. In order to investigate the effectiveness of the paradigm, three experiments were designed and validated. Real-life sensor data and synthetic statistical data was collected, processed and analysed. Bayesian Machine Learning models and Analytics were used to consolidate the design and evaluate the performance of the algorithms. In the Fog environment, the first scenario tests the Aggregation by Distribution algorithm. The solution contribute in achieving a notable efficiency of data delivery obtained with a minimal loss in precision. The second scenario validates the merits of the approach in predicting the activities of high mobility IoT applications. The third scenario tests the applications related to smart home IoT. All proposed Consensus algorithms use statistical analysis to support effective decision making in Fog and enable data aggregation for optimal storage, data transmission, processing and analytics. The final results of all experiments showed that all the implemented consensus approaches surpass the individual ones in different performance terms. Formal results also showed that the paradigm is a good fit in many IoT environments and can be suitable for different scenarios when applying data analysis to correlate data with the design. Finally, the design demonstrates that Fog Computing can compete with Cloud Computing in terms of accuracy with an added preference of locality.

Dissertation directed by Dr. Zenon Chaczko and Dr. Wayne Brooke
School of Electrical and Data Engineering

Dedication

I dedicate my dissertation work to my family, my supervisor and many friends. A special feeling of gratitude to my loving parents whose words of encouragement and push for tenacity ring in my ears and whose a look of happiness in their faces is my inspiration. In the same time, my beloved children who have never left my mind and who I work to make them be proud of me. My Supervisors, who guided and supported me along the way with a beautiful spirit and care. My friends, for always being there for me and help me with the best they know. Thank you all for the love, compassion and support.

Acknowledgements

I would like to thank my supervisors, Dr. Zenon Chaczko and Dr. Wayne Brooke, who guided and supported my work along the way and helped me get results of better quality. I am also grateful to my friend, Abdelwahed Khamiss, for his help and support in overcoming numerous obstacles I have been facing through my research. I am also grateful to my friends Mahmoud Gamal and Alina Rakhi who helped me in several occasions to improve my research directions and writing skills. I would like to thank my fellow doctoral students for their feedback, cooperation and of course friendship. I would like to thank my friends for accepting nothing less than excellence from me. Last but not the least, I would like to thank my family: my parents, my children and my sisters for supporting me spiritually throughout writing this thesis and my life in general.

Firas Al-Doghman

Sydney, Australia, 2019.

List of Publications

Journal Papers

J-2. **F. Al-Doghman**, Z. Chaczko, and W. Brooke "Data Science for the IoT Fog Computing" *IEEE Internet of Things Journal*, in communication.

Conference Papers

C-1. **F. Al-Doghman**, Z. Chaczko and W. Brooks 2018, "Adaptive Consensus-based Aggregation for Edge Computing *2018 26th International Conference on Systems Engineering (ICSEng)*.

C-2. A. Ajayan, **F. Al-Doghman** and Z. Chaczko 2018, "Visualizing Multimodal Big Data Anomaly Patterns in Higher-Order Feature Spaces *2018 26th International Conference on Systems Engineering (ICSEng)*.

C-3. **F. Al-Doghman**, Z. Chaczko and J. Jiang 2017, "A review of aggregation algorithms for the internet of things *2017 25th International Conference on Systems Engineering (ICSEng)*.

C-4. J. Jiang, Z. Chaczko, **F. Al-Doghman** and W. Narantaka 2017, "New LQR Protocols with Intrusion Detection Schemes for IOT Security *2017 25th International Conference on Systems Engineering (ICSEng)*.

C-5. **F. Al-Doghman**, Z. Chaczko, A. Ajayan and R. Klempous 2016, "A review on Fog Computing technology *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*.

C-6. **F. Al-Doghman**, Z. Chaczko, W. Brooks D., Hoang, and L. Carrion Gordon 2019, "Social Consensus-inspired Aggregation for Edge Computing *CSNet 2019 3rd Cyber Security in Networking conference*.

Book Chapter

C-1. **F. Al-Doghman**, Z. Chaczko, and A. Ajayan 2019, "Policy-based Consensus Data Aggregation for the Internet of Things *Smart Innovations in Engineering and Technology*, Topics in Intelligent Engineering and Informatics series.

C-2. A. Ajayan,**F. Al-Doghman** and Z. Chaczko 2019, "Tensor decompositions in multimodal Big Data: studying multiway behavioral patterns *Smart Innovations in Engineering and Technology*, Topics in Intelligent Engineering and Informatics series.

Contents

*parts of this section come from my paper "A review on Fog Computing technology"(2016)

†parts of this section come from my paper "A review of aggregation algorithms for the internet of things"(2017)

List of Figures

Abbreviation

IoT - Internet of Things

DM - Decision Making

WSN - Wireless sensor Network

DNN - Deep Neural Network

RNN - Recurrent Neural Network

CNN - Convolutional Neural Network

BNN - Bayesian Neural Network

BRNN - Bayesian Recurrent Neural Network

LSTM - Long Short-Term Memory

BS - Base Station

RPi - Raspberry Pi

AUC - Area Under the Curve

ROC - Receiver Operating Characteristics

L0 - Level zero: front end level in the proposed paradigm

L1 - Level one: Fog level in the proposed paradigm

L2 - Level two: BS or Gateway level in the proposed paradigm

Chapter 1

Introduction

1.1 Background

We are living in the era of the Internet of Things (IoT) where everything is, or about to be, connected to the Internet. From home facilities to industrial machineries throughout transportation vehicles, engines, exercise equipment and even pets and cattle, all be represented as 'things' in it. However, what have been accomplished till now is still far from expected, 50 billion objects by 2020 and 1 trillion by 2030 [22][23][24]. The Internet of Things (IoT) represents the upcoming stride in the direction of the digitisation of our society and economy, presenting fascinating opportunities across an amazing range of applications, where objects and people are interconnected through communication networks [25]. This number of devices that are connected to the Internet can produce a massive amount of a neuromas volume of data and create serious Big Data related problems [26] [27][28][29] [30]. Big data characterise with properties of having complexity, heterogeneity, autonomous, and distributed form of a continuously expanding dataset. The challenges emerge with the expansion of data that are extremely getting out of the scope of ordinarily used tools and software to deal with their management and analytics problems [31]. Big data in real-time have miscellaneous and autonomous exemplification bringing exceptionally unstructured and independent data based relationships in generating faulty and complex outcomes. The features of heterogeneous data are regarded as various data representations. In order to lessen the impact of heterogeneous and complex data; a computational operations are to be presented at localised scheme

taking into account that they are having improved computational power. Transforming data into a fusion of common forms of data can be a way of high compatibility for data linkage and proportionality indexing for acquiring better analytical results [32][33][34] [35][36].

The need to handle the growing concerns of being able to deal with big data generated and travelled within the networks, knowing it contains a prominent portion of redundancy, in real time and work within the limits of the available bandwidth leads to the emergence of Fog Computing which operate along the continuum from beyond the Cloud towards the front-end devices and supplies a distributed computing based network services and storage [37][38]. The interest in Fog Computing is motivated by the fact that it is crucial to deal with potential errors and intensive data flow, so it does not flood the whole system, at early stage [25]. Fog Computing represents a supplement to the Cloud paradigm which runs geo-distributed applications across the network. In opposition to the Cloud, the Fog not only carries out latency-sensitive applications near the network front-end, but also carries out latency-tolerant tasks effectively at nodes with powerful computational capability at the network intermediate level. Cloud Computing, at the top of the Fog layer, with data centres that are still a preferred technology for performing deep analytics tasks [39]. In the recent times, yet another important concept of Edge Computing was defined the context of the IoT. The concept relates to technology and solutions facilitating data processing and management that is done at the network edge and near the source of data generation [40] [41][42][43][44].

Usually, the extraction of all data, particularly in a real-time setting, is in most cases nonviable. Furthermore, presently applied procedures for dealing with big data are still not sufficient. Thus, a need exists for a platform holding the qualification of delivering real-time prediction reaction for data analysis [45]. Advanced sensing devices play a pivotal role in acquiring data generation, transition, and dis-

semination for the IoT. Particularly, the applications of IoT and the sensing devices intelligent systems, like Wireless Sensor Networks (WSNs), are jointly connected. Modern intelligent sensing systems collect mass volumes of sensing data, more than the processing capabilities of common techniques and tools. Therefore, collecting, managing, then processing big sensing data in IoT within a sufficient time duration is a challenge with a good potential for both industrial and research applications [46].

There is a need to manage the resources within IoT in order to mitigate Big Data problem and its redundancy related issues. Data Aggregation methods can be used as one of the techniques of Fog Computing by which only the requisite data is to be forwarded to the ascendant node and so forth up to the Cloud. Data aggregation involves the process of forwarding a synopsis of several data packets rather than the whole packets [47]. The accelerated use of IoT type of technologies (like sensors and actuators) would generate immense amount of data having enormous level of redundancy. To realise this proposition, it needs to be managed in a smart way that considers some policies by involving voting to produce a consensus aggregation that would gain resource efficiency. Consensus-based management is one of less explored techniques that uses voting mechanism amongst specific nodes (aggregators) according to a policy to reach agreement about what events or activities happens within the network environment as well as how to manage the resources. By accomplishing this, less data will travel across the network which will save bandwidth, energy, traffic control processes, computation, time, etc. It also will make the system more flexible to adapt or modify any computing rule or policy according to the system status. The proposed management methodology applied to the Fog platform within the IoT context has a two ways technique in which it can be implemented on data in an upstream direction as in deciding about combining sensed data and decisions related to them, then sending them up to the Base Station (BS). It also

applied on 'controls' in a downstream orientation for deciding how to integrate actuation commands from the base station down to the actuators where there may be a presentation of redundant commands and/or it is targeting specific actuators. The process accumulate data/controls in the aggregator node according to an agreed approach and propel the accord one to the ascendant node. Additionally, the proposed mechanism includes a Machine Learning process that leads the decision operation and deals with nodes reliability issue by discovering status change and faulty nodes within the network.

Fog network data analysis may be considered influential in increasing the efficiency of data aggregation in IoT as an information intermediary, informing nodes of a higher levels about network status. Their role involves conducting determination about data boundaries and patterns in different operational modes, and issuing decisions based on their knowledge (machine learning). Therefore, an opportunity exists to understand the importance of enhancing the quality of data reached to base station of the network as well as data analysis, by investigating different types of operational modes and the link between data patterns or boundaries and the decision to be made. In other words, this thesis addresses the need to understand what and how data patterns are learned by Fog network nodes to determine which data is of a value to the end users as well as the efficiency of network resources. An investigation of use of IoT Fog network by simulation and implementation of case studies will help achieve this. In addition, such an investigation provides an opportunity to examine the various factors driving the use of Data aggregation by the guidance of Group Decision Making techniques. The introduction of the Fog based platform in this work has macroscale impact and use within the IoT environments but is not limited to specific applications such as mobile networks.

The process of mitigating Big Data problem caused by the huge amount of IoT nodes generating a massive amount of data with a large amount of redundancy and

heterogeneity, pass through data organisation performed from the front-end so that only the valuable data is to be pushed to the upper levels. This requires knowing the boundaries of the desired data and what is of a value to the network and its users which will leads to the need of data analysis to specify the types and limits of the desired data as well as the quantities and cycling periods of delivering them to their destinations. After performing data analysis, the data needs to be coordinated in a way of introducing a degree of decision making within the Fog network levels. This ends up in a data that is considerably accurate, as opposed to raw data without decision making, at different levels which helps in saving packets (we don't have to transmit extra packets for a redundant information). In this direction, we will save energy and other network resources caused by network confusion, so the efficiency of the system will improve.

1.2 Research Objectives

The aim is to effectively manage data in Fog Computing architecture by using adaptive, collective and collaborative method to achieve autonomous decision making paradigm close to data sources within IoT framework.

This research is based on the following objectives:

i. Correlate and analyse data.

ii. Recognise event and activities within the network.

iii. Achieve distributed in-network Fog Computing.

iv. Develop an effective Consensus model.

v. Save network resources while maintaining accuracy levels.

1.3 Research Gap

There have been various attempts by researchers to perform practices of collecting and managing data in various networks. In most cases, data collection and management is performed using fixed criteria with no adaption according to network status. In addition, the researches, in most of times, did not consider the higher level application requirements and they did not give a clear description of the data collected. Also, projects aimed to improve the efficiency of data delivery have been undertaken by many researchers in the literatures. These initiatives include networks for sharing information and best practice, research collaboration, and the development of IoT and Fog Computing frameworks. However, those practices have not consider data analysis and intelligent data organisation at the network edge to extract what is of a value to the network and its users. Improvements to the guidelines and frameworks for Data management practices of IoT can only be made by having a better understanding of the use of intelligent data analysis and organisation near the front-end.

1.4 Research Problem and Motivations

The research problem is that within IoT, a huge mass of uncoordinated heterogeneous data with a large amount of redundancy will be generated and exchanged by billions of devices which will yield to Big Data issue. The existence of this issue causes traffic congestion and affects network efficiency. Most of the front end IoT infrastructure resources are shared among authorised users and software agents, still data generation and transmission is not well organised, i.e. there are no polices bounding what is of a value to the users among the whole measured data and no data analysis to anticipate it. This results in poor data quality generation and transmission since there is a lack of decision making at the front-end to specify the required data that suits users' needs. All of that leads to the question of how to

manage data in an intelligent manner to achieve the research objectives.

The Motivation of this work is the need to have a better ways of managing data and transferring them in an organised manner based on real-time data analysis at the network edge. There is an increase roll of front-end devices like intelligent gateways and access points which need to include smart processing.

1.5 Research Significance

This study will contribute in the Enhancement of data quality in the Internet of Things, not only at front-end but also in other part of the paradigm. The research will encourage the mitigation of Big Data issues like heterogeneity, incoordination, and redundancy and also consider the needs of network users. Additionally, the system will apply automation within Fog network nodes and adapt its performance to improve the efficiency of network resources.

The outcome to be considered consists of the following: the enhancement in all components of network metrics such as energy efficiency, overall packets savings, the reduction in network congestion, traffic control computation, data redundancy elimination, delay and bandwidth reduction, etc.; development of Fog network towards automation; increase in system flexibility to adapt or modify any computing rule or policy according to the system status to meet users demand; improve the reliability of the system due to its ability to detect the change in network status as well as discover (and possibly eliminate the effect of) faulty nodes.

1.6 Problem Statement

There is a need to autonomously handle the organisation of data generation and transmission within shared Fog network using consensus-based set of algorithms. Big Data is a serious problem within IoT. It is required to mitigate Big Data inherent heterogeneity, incoordination and redundancy as well as considering users needs.

Additionally, high levels of automation and fault tolerance is needed within IoT to cop up with the rapid growth of the IoT infrastructure and related technology. These infrastructures may potentially have billions of nodes exchanging data which makes it hard to process it at a point far from its source.

1.7 Research Hypothesis

In this work, it is hypothesised that **it is possible to design adaptive and fault-tolerant Consensus-based Machine Learning algorithms for data management in Fog Computing.**

1.8 Validation

In order to validate the proposed solutions, hybrid research methodology is involved that includes the design of a test-bed and the execution of several experiments. In order to investigate the effectiveness of the paradigm, three experiments were designed and validated. Real-life sensor data and synthetic statistical data was collected, processed and analysed. Bayesian Machine Learning models and Analytics were used to consolidate the design and evaluate the performance of the algorithms. In the Fog environment, the first scenario tests the Aggregation by Distribution algorithm. The solution contribute in achieving a notable efficiency of data delivery obtained with a minimal loss in precision. The second scenario validates the merits of the approach in predicting the activities of high mobility IoT applications. The third scenario tests the applications related to smart home IoT. All proposed Consensus algorithms use statistical analysis to support effective decision making in Fog and enable data aggregation for optimal storage, data transmission, processing and analytics.

1.9 Research Contribution

The main contribution of the research can be summarised as follows:

i. Introducing the accommodating Fog Consensus Management Paradigm that can suit a lot of IoT scenarios within it and can fit in many experiments and IOT environments.

ii. Developing the consensus decision making model which its accuracy surpasses individual decision making.

iii. propose the Likelihood Multiplication Consensus algorithm that can achieve markedly good performance when used to aggregate decisions resulted from Bayesian classifiers.

iv. Implementation of Fog Computing experiments in an IoT environment and demonstrate that its accuracy performance can compete with the Cloud.

v. Introducing Consensus Management which depends on voting mechanism in a way that leads to consensus about aggregation process which can be extended to self-management as an upcoming stage.

vi. Including the Machine learning mechanisms within the consensus algorithm as a data analysis tool to maintain the reliability of the system and contributed in fulfilling the big picture of reliable IoT.

1.10 Thesis Organization

This thesis is organised as follows: The first part gives a brief overview and introduction to the topic of this report. In general, the main aim of this research is to apply intelligent management to Fog Computing nodes within IoT. Other chapters of this thesis present the following:

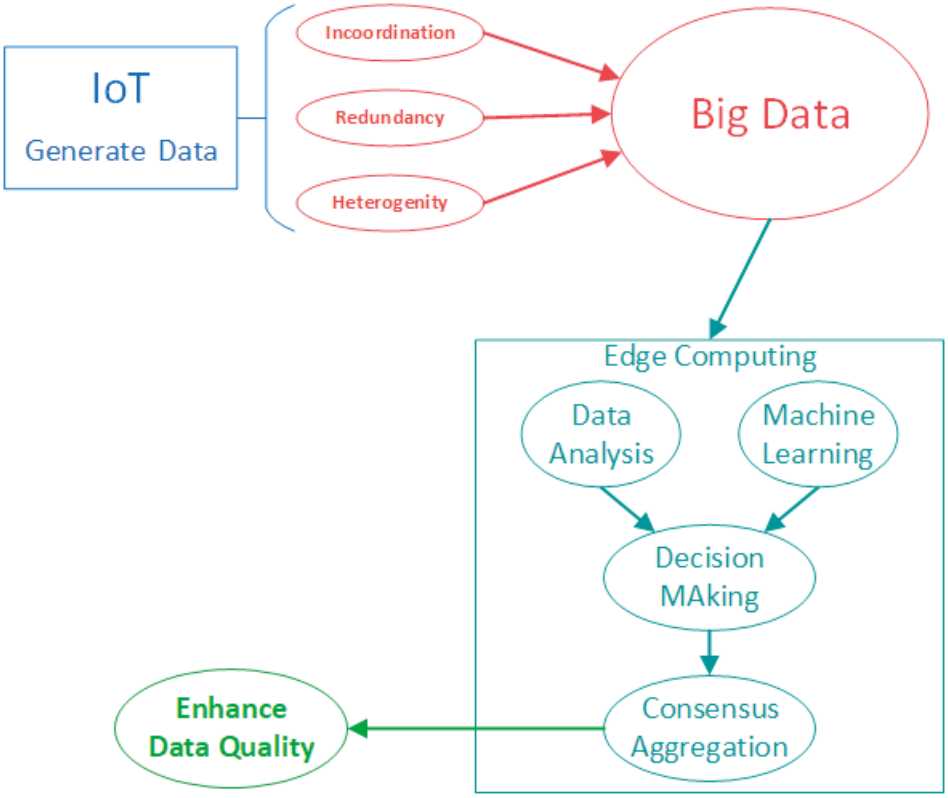


Figure 1.1 : Mind Map

- *Chapter 2*: This chapter presents a survey of different literature related to the research topic and directions.

- *Chapter 3*: The Hybrid methodology are derived in this chapter explaining the operation of the Consensus-based management set of algorithms.

- *Chapter 4*: This chapter presents an experiment design that tested to detect events based on environmental data generated by environmental and air quality sensors. The experiment tests the Aggregation by Distribution algorithm as well as implements the consensus-based algorithm using Bayesian Machine Learning models and Analytics.

- *Chapter 5*: This chapter presents another experiment which is used to consolidate the design and evaluate the performance of the proposed algorithms

predicting human activities based on mobile movements and position sensors. The proposed algorithm is executed using Bayesian Neural Networks.

- *Chapter 6*: This chapter presents a third experiment to test the applications related to smart home IoT activities based one fixed movement and air quality sensors. The Consensus-based operation is implemented with the help of Bayesian deep learning practices.

- *Chapter 7*: A brief summary and conclusion of the thesis contents are given in the final chapter. Recommendation for future works is given as well.

Chapter 2

Literature Survey

2.1 Introduction

Much research has been introduced to explain the significance of Fog Computing, aggregation processes, agreement mechanisms, fault tolerance approaches and autonomic computing which helped to build and support this research. The Fog Computing implies a number of characteristics that portray the Fog as a non-trivial extension of the Cloud while aggregation practices helped to reduce resource consumption in the networks it applied to. However, it's crucial to stratify reliable mechanisms for detecting and eliminating diverse threats, attack sources, and malicious nodes which is why the Byzantine Fault tolerance was examined.

The IoT is the network of multibillion interoperable devices (sensors) used for information or data acquisition, communication, analysing, decision making and reporting to intelligent systems, which are the part of such network. This information then can be used for manipulation and controlling the behaviour of physical devices in certain environmental settings. Due to this dynamism of IoT systems, both the academia and industries are getting attracted towards it.

2.2 The Internet of Things

The Internet of Things has been gaining a vast volume of attention since it was introduced and a large number studies have been proposed [48]. A scenario of a converged IoT based on a WSN and a smartphone was implemented in Tsitsigkos et al. (2012) to accomplish a monitoring service and object mobility tracking inside

a house [1].

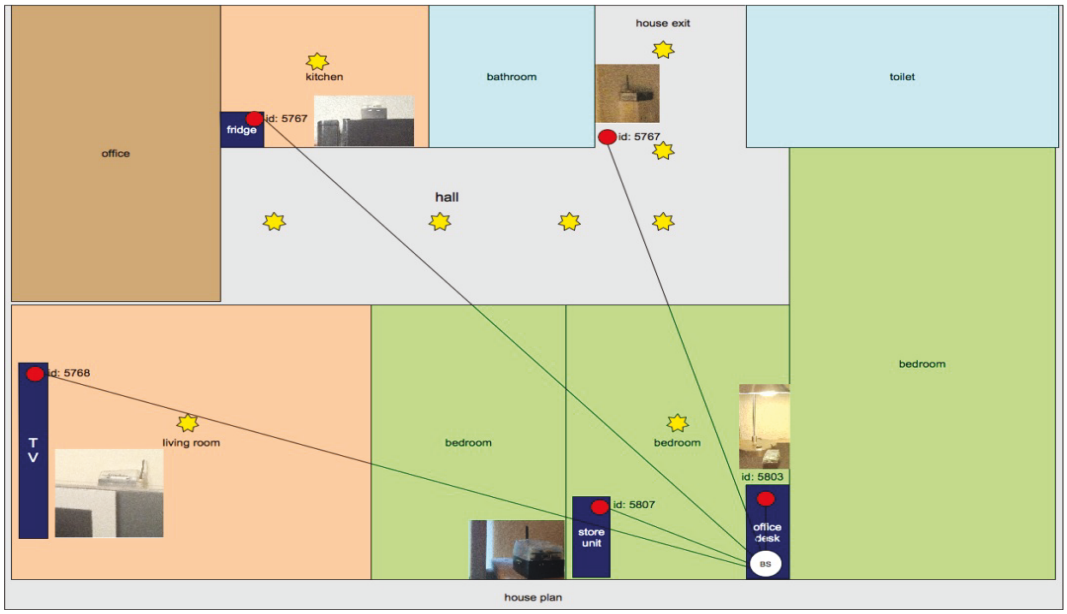


Figure 2.1 : A typical smart home IoT scenario [1]

The Internet of Things gained a numerous amount of attention since it was introduced and an extensive literature has been proposed. It demands real time response and optimum resource utilization in many applications and services, with huge data influx and bandwidth bounds. A one scenario (illustrated in Figure 2.2 as a part of capstone project) of keeping track of temperature in the set environment and notify the user if a temperature reading is out of the desired range was implemented in [25] to accomplish a monitoring service.

2.3 Fog Computing*

2.3.1 Characterisitcs

The Fog Computing contains a number of characteristics which portray it as a non-trivial expansion of Cloud Computing. In terms of IoT, Bonomi et al. (2012)

*parts of this section come from my paper "A review on Fog Computing technology"(2016)

monitoring System diagram.jpg

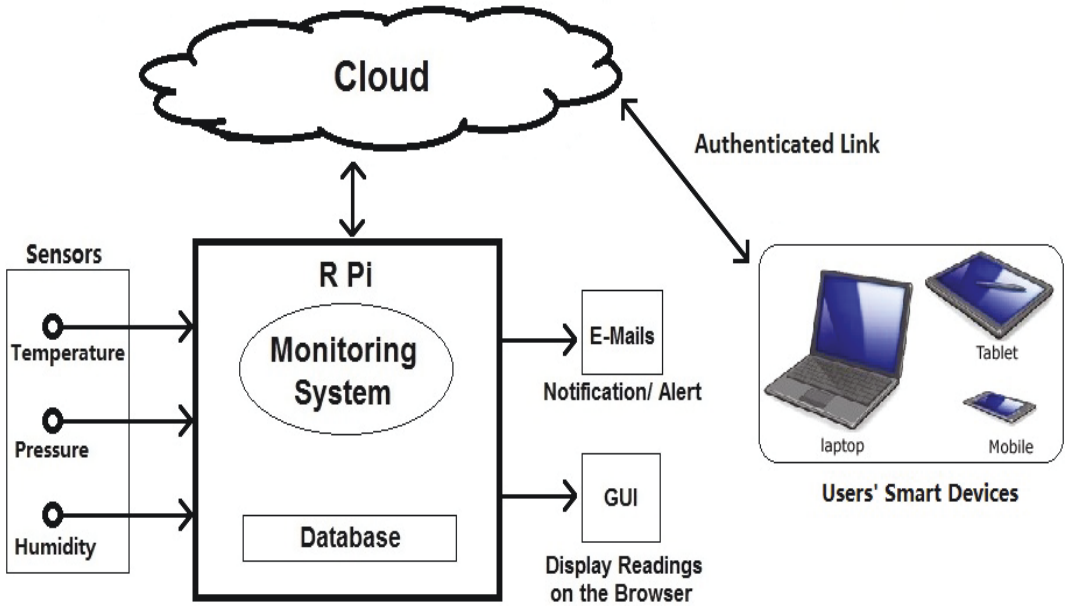


Figure 2.2 : Enhanced IoT Monitoring System Scenario

outline the vision of Fog Computing, define its main features and discuss that the characteristics of Fog Computing makes it the suitable architecture for numerous critical IoT practices and services. Fog nodes features are elaborated as low-latency and location awareness, wide-spread physical location, mobility, massive node numbers, pre-dominant part of wireless access, robust existence of streaming, heterogeneity and real time applications. There is an argument about the Fog model for it having three tiers including the one designed for M2M (Machine-to-Machine) reciprocal action. This is dedicated for data acquisition, processing and controlling of actuators, also it isolates the part of data for local use with rest of the data which it will push towards the top tiers. While HMI (Human-Machine-Interface) interactions are handled through the second and third tiers (visualization and reporting), as well as M2M (systems and processes).Further, Cloud provides global centralization which is used as data repository and business intelligence analytics base, while the Fog nodes offer the localization [49][50]. Yannuzzi et al. (2014) [51][52] mentions about the Fog Computing being the promising platform for IoT. This paper

observes some of the challenging and encouraging IoT scenarios, highlights the inevitable interaction of the Cloud and Fog in upcoming time and review some of the technologies that will necessitate substantial improvement so as to support the future IoT applications. Similarly Stojmenovic and Wen (2014) [53][54] reveal the new paradigm of Fog computing and the identical work under the same canopy and also have discussed about the attack of man-in-the-middle for the security domain in Fog Computing. Moreover, they have demonstrated the novel Fog Computing and have illustrated some applications and benefits of it in different areas including smart grids, sensor networks (WSNs), Internet of Things (IoTs) along with software defined networks (SDNs). Besides it, they have reported on some common issues like privacy and security in the Fog Computing. Fog Computing architecture considers a number of characteristics that portrays it as a non-trivial extension of Cloud Computing. The migration of trust and the services amongst Fog nodes or sensors and between the Cloud and Fog is commonly described in terms of IoT. However, Shi et al. (2015) discusses the Fog Computing characteristics and what services it can support. They introduce the discussion of the application of Fog Computing in the forthcoming healthcare systems, where its service support model has been discussed in Shi et al. (2015). The features of Fog Computing makes it a robust addition of the cloud by which it can minimize the overall interval by exchanging data within the local area network. Also, Fog Computing supports a small amount of data storage and by using different sorts of storage strategies it will consume reasonable computing power to attain precise analyses. Furthermore, it is able to do some altering and aggregation processing and filter out invalid or corrupted data before sending it to Cloud and should prioritise flow of what content to send, in which format and when to send it i.e. time. It has been said that Cloud is linked to servers, though Fog is connected by the smart devices in place and performs in distributed setting instead of centralised performance as in the case of Cloud. The Fog nodes process clusters

of data which are disintegrated with the huge computational power. It has been summarise that Fog is the mediator between the Cloud and the networked devices and facilitate them with pushing service to ingest the data and for updating the processed information on the Cloud for deep mining and long lasting storage added that data altering and aggregation need some rules, analysis (online) requires actual or associated data to calculate and realise the intelligent on-line analytics, and also mentioned that ephemeral and semi-permanent storage has to be supported by Fog [55].

2.3.2 Resource Management in Fog Model

Fog Computing is a paradigm that was introduced recently, so there has been no existing standard architecture as such regarding computing resource management. The Fog model for resources management is presented in Aazam and Huh (2015) [56]. They elaborate that for the IoT, it is very much essential to have an efficient and effective framework for management of resources. They further illustrated that the model should emphasize on prediction of resources, advance reservation, customer-based resource reservation and estimation and based on the characteristics of old or new customers, the model must focus on the pricing as well. Their dynamic and flexible model is executable in myriad settings by incorporating diverse scenarios and is capable of adjusting according to different scenarios. The implementation of the model was made using Java/NetBeans 8.0 and the evaluation was done by CloudSim toolkit. As per the needs of industrial process, Gazis et al. (2015) [4] describes the facilitating infrastructure of Fog Computing as an AOP (Adaptive Operations Platform) which provides end-to-end management. AOP is built on the service capabilities of the following layers: Fog Computing Infrastructure which involves networking equipment with specific Fog capacities and supplies Operational Support System (OSS) and end-to-end communication services which influence the

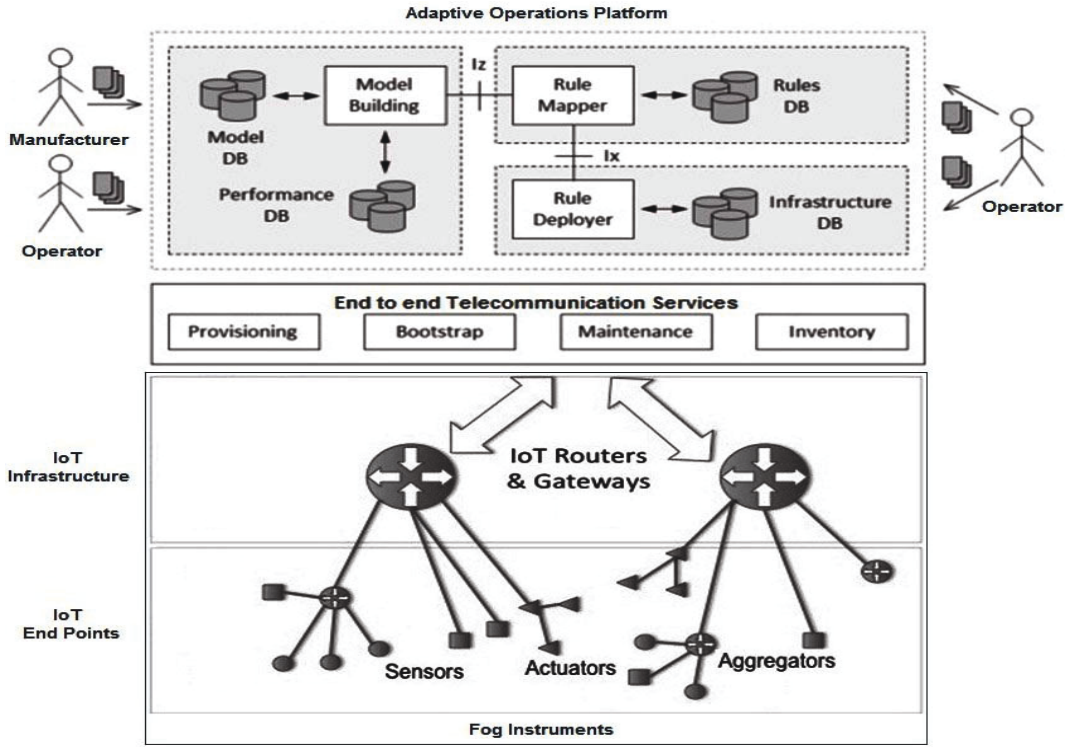


Figure 2.3 : Architecture of the Adaptive Operations Platform (AOP)

Fog Infrastructure to give the standard resource the executives and business support capacities (e.g., stock, upkeep, provisioning, and so forth.). To effectively use key highlights of the Fog Infrastructure, AOP incorporates a few useful components. The Model Building (MB) practical component consolidates static data about the disappointment models of the gear types found in the modern site alongside unique information gathered amid the last's task. The practical component of the Rule Mapper (RM) entrusted with mapping the intertwined model with a huge traffic dealing and with guidelines comprehended by the software defined networking foundation. The utilitarian i.e. Rule Deployer (RD) component having a massive traffic handling and with principles and portrayal of the capacities detected in SDN framework registers the sending intend for applying this arrangement of guidelines over the fitting components. This model can be altered as in Fig. 2.3 to describe a more complete IoT model.

Table 2.1 : Outcomes Comparison between Cloud and Fog

	Cloud Computing	Fog Computing
Prediction Latency	5 sec.	1.5 sec.
latency of Web page display	8 sec.	3 sec.
Internet Conjestion	75 Kbps	10Kbps
Hardware usage	Amazon Web Server	Raspberry Pi

The technique used by Krishnan et al. (2015) to advises a method for transferring the computation from the cloud to the network, Fog Computing, by presenting an android like appstore on the networking devices where in the user can select the data which is to be processed on the edge and that which will be processed on the Cloud. It includes labelling packets which are to be processed on the network device while unlabelled packets are sent straight to the Cloud without any transitional processing. The deployment of Arduino tool for connection to Wi-Fi and Raspberry Pi for measuring surrounding temperatures for each 5 seconds and then sending it to the router i.e. to RPi, which has remained their approach for implementing Fog. At the acquisition of temperature information, a python script executed by RPi throws temperature readings in separate file(s) readable by Arduino tool. The records are to be time-stamped and the outcomes need to be written into MySQL databank instance running over cloud and later web interface is to be established for displaying information by Could PHP instance readings. Finally they suggest incorporating Fog Computing in the Routers and using machine language techniques for deciding which packet should be processed by the Fog and which one to be done at the Cloud. The results (Table 2.1) showed that Fog based architecture has a better response time compared to the cloud architecture [57].

Fog Computing model has a significant layer of the Cloud as Madsen et al. (2013) [58][59] described and also have stated for examining the reliability of smart devices networks. They present the reliability challenges introduced by current computing paradigms and prolongs the argument towards reliable Fog platforms integrating smart devices networks that communicating among themselves as well as the Cloud and discuss whether reliable Fog Computing platform is practicable for real life projects or not. The issue of pairing or utility-based matching within the same domain of IoT nodes is to be implemented with Irvings matching algorithm in a Fog model as highlighted to be effective IoT node pairing scheme in Abedin et al. (2015) [2] The classical pairing algorithm being the modified proposition explodes Fog Computing with improved utility factor in the context of a facilitating and expert method of M2M communication and it can pledge collaborative pairing among nodes for instance, one node to multiple nodes. Simulation was used to study the effectiveness of proposed algorithm which is used for solving the stable roommate issue by considering one-to-one steady matching. The aim of tuning Irvings pairing algorithm is for supporting quota-based nodes' pairing, wherein every node has the ability to facilitate both one to multiple as well as one to one pairing. Figure 2.4 depicts the performance outcome of the proposed procedure within node-pairing approach where the comparative results demonstrate the Irvings matching algorithms efficiency for 5 pairs of nodes with quota. It also outperforms the Greedy algorithm in which the nodes are matched by considering the neighbouring nodes. The blend of quoted-based method and the highest utility based nodes' selection matching is the cause of this distinguished efficiency. Further, the overall utility of huge pairing between one to many nodes is described due the escalated collaboration between IoT nodes. Also, this scenario suggests that having a big number of nodes' quota based pairing and node set, the entire utility of node's domain set will be augmented.

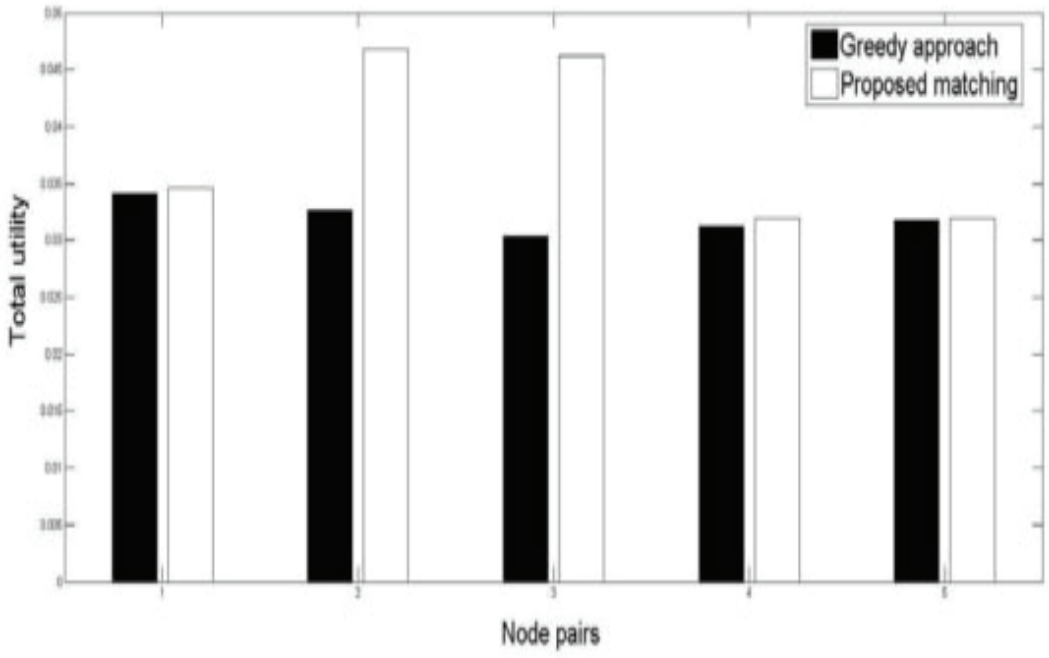


Figure 2.4 : Comparison outcome example illustrating the improvement of the entire utility of node pairs for the proposed matching procedure [2]

A shared parking model was constructed by Tran et al. (2015) [60] by considering parking problem from IoT perspective where Roadside Cloud and Fog Computing are employed for finding an available location. In the model, the Fog server machines function as a connection between parking lot sensor and Roadside Cloud, in which information regarding parking slot i.e. either vacant or reserved will be communicated with Fog server locally installed. Then the information of the managed free sport will be delivered from Fog servers to RUSs. The system can control all available parking data (RFPARK) and at each RSUs the (RFPARK) will bridge the Fog servers to advice drivers for an optimal spot based on matching theory method, where the parking slots association is formulated similar to many-to-one pairing game, wherein, a group of vehicles will be allocated a group of parking lots using preference concept to pattern the common and inconsistent regard. Stojmenovic and Wen (2014) [53] introduces a survey article which expands the Fog notion for

the distributed control of a small building and identifies the Cloudlets as a particular instance of Fog Computing and also associates it with the software defined network (SDN) scenarios. He discusses the scheduling of Cooperative information, the issues in SDN based vehicular networks adaptive traffic light, and the administration of demand response in micro grid based smart grids and macro station. He also claims that Fog Computing is becoming a significant class of cyber physical systems (CPS) and its role in several interesting scenarios was expounded: smart building control, vehicular networks, smart grids, and wireless sensor and actuator networks. There is strong need of reconsidering the licensing of software, privacy, business models and certain other problems in the context Fog Computing and Cloudlets.

2.3.3 Design and Architecture

The growth of IoT, according to its diverse and dynamic nature, shifted the paradigms many times in terms of the design of network architectures and many approaches have been presented and proposed. The crucial design and implementation of a new Fog node named as the IoT Hub was presented by Cirani et al. (2015) [3] as a mediating network entity which combined the tasks of application layer and the border function of router, targeting IoT applications located at multiple physical edge of networks for creating IP based IoT network that can be deployed as a structure of the growth of WoT (Web of Things). The aim for building IP-based IoT network, being utilised as a foundation of the organization of IoT implementations which can enhance the network abilities by implementing cross-proxy, border router, cache and directory of resource as functions. The integrated IoT Hub has the ability to completely cover the heterogeneous nature of smart objects which can relate to them by using uniform interfaces without the need of any preceding configuration. The interaction with smart objects through the IoT Hub occurs as shown in Fig.2.5.

The performance appraisal has shown that the IoT Hub, using limited process-

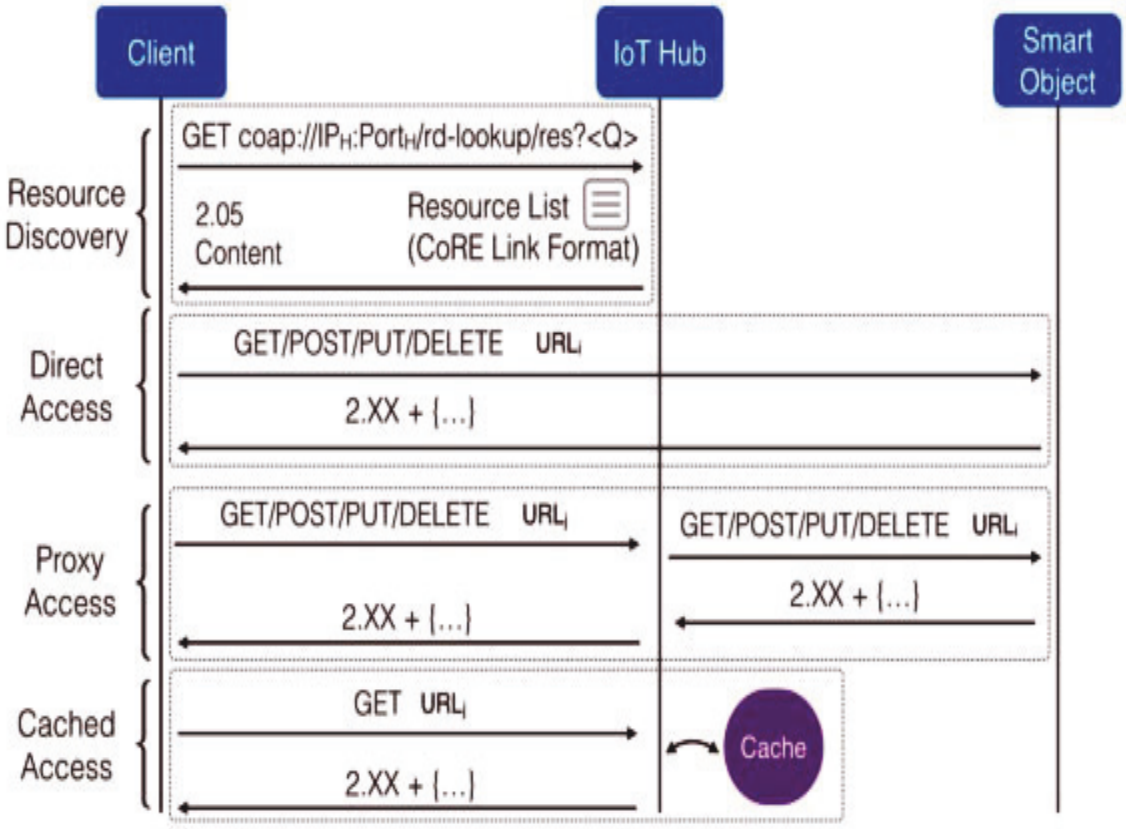
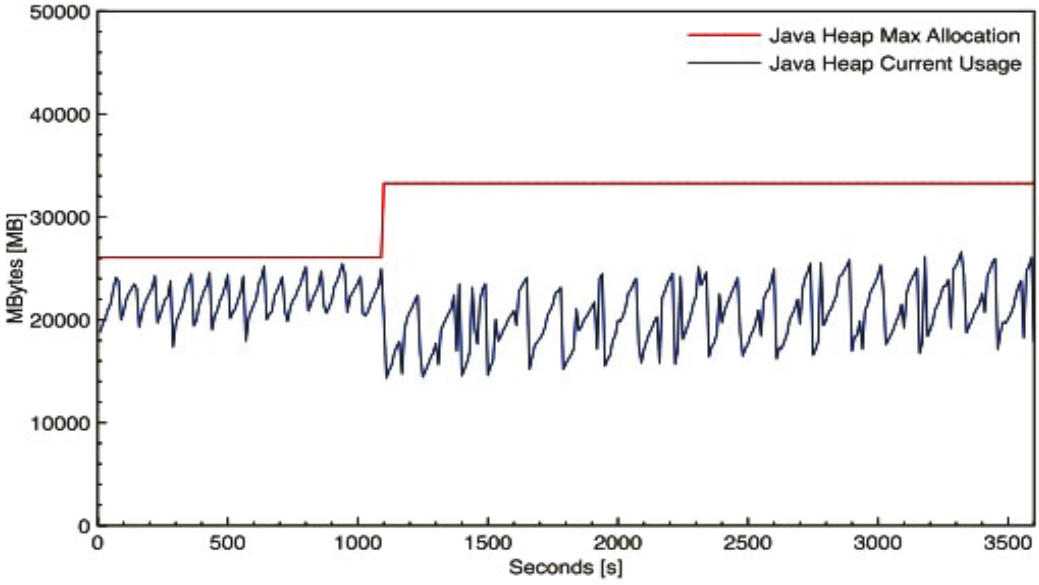


Figure 2.5 : Interaction between clients and heterogeneous smart objects with the mediation of the IoT Hub [3]

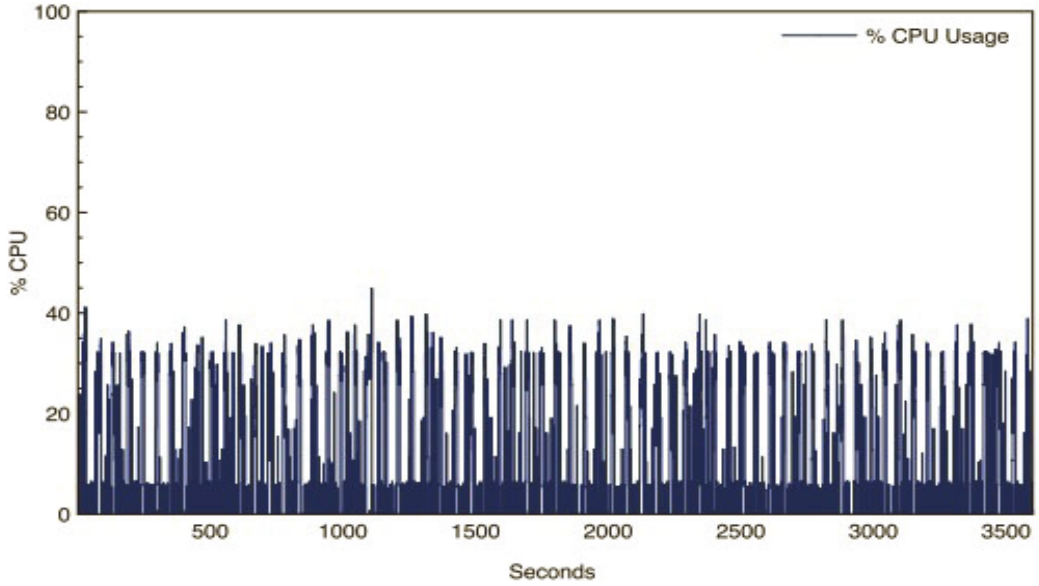
ing and memory resources, is able to administrate several heterogeneous physical networks which makes it possible to install it on low-end devices like the RPis. The method of implementing the IoT Hub was to use a Java based application about CoAP and the associated drafts deployed to an RPi and used to administrate resources installed in heterogeneous Smart Objects in a real world IoT infrastructure. The IoT Hub can decrease the handling load on smart objects while moving a portion of it to the network edge. The results are shown in Fig. 2.6.

In Datta and Christian (2015)[61] an architecture for connecting vehicles where the Fog platform is deployed at the Road Side Units (RSUs) and M2M gateways. Such an architecture enables customer-centric services such as M2M data analytics including semantic web technologies, connected vehicles management and IoT services discovery which empowered by Fog Computing distinctive features where the whole Fog architecture is combined into one M2M typical architecture to explore Fog Computing paradigm advantages. The goal of Sarkar et al. (2015) [62] was to develop a mathematical model computing model of Fog and examining the suitability of the model for IoT, particularly when its critical to match the necessities of latency-sensitive applications operating at the network edge within the structure of IoT. Additionally, a relative performance for assessment of Cloud Computing is executed with the Fog Computing for situation where high number of Internet-connected devices need real-time services. The model was done by mathematically characterizing the power consumption, CO2 emission, service latency, and cost of Fog Computing network, and performing performance evaluation in a high numbered devices, connected to internet environment with real-time service demand. The work focusing on analysing Fog suitability within the IoT framework for crucial encounter of the latency requirements for the operating applications at the network-edge had the aim to build a mathematical model of Fog Computing. The work eventually rationalise the Fog as an enhanced, eco-friendly platform for computing which can



(a)

(a)



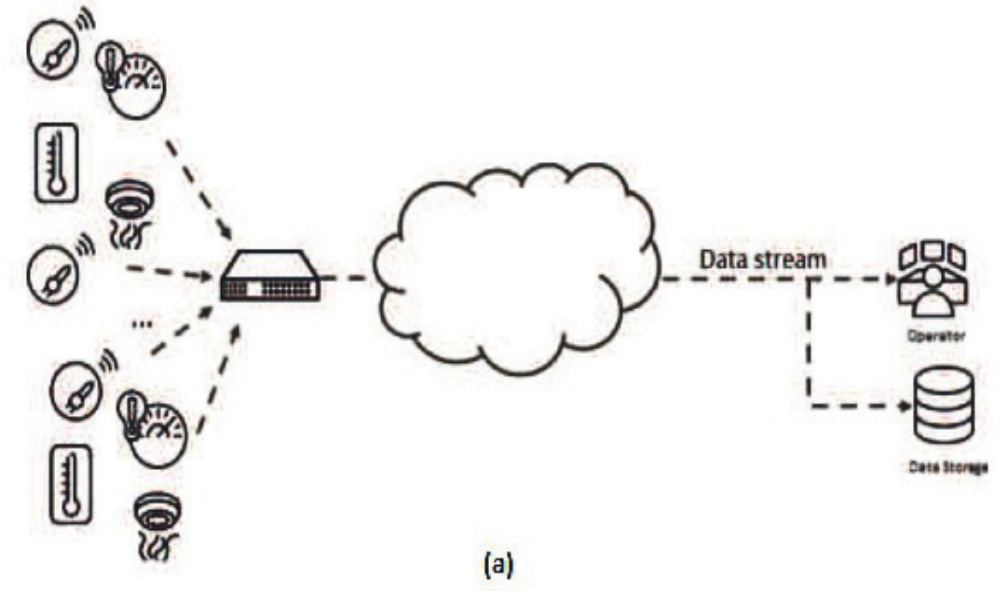
(b)

(b)

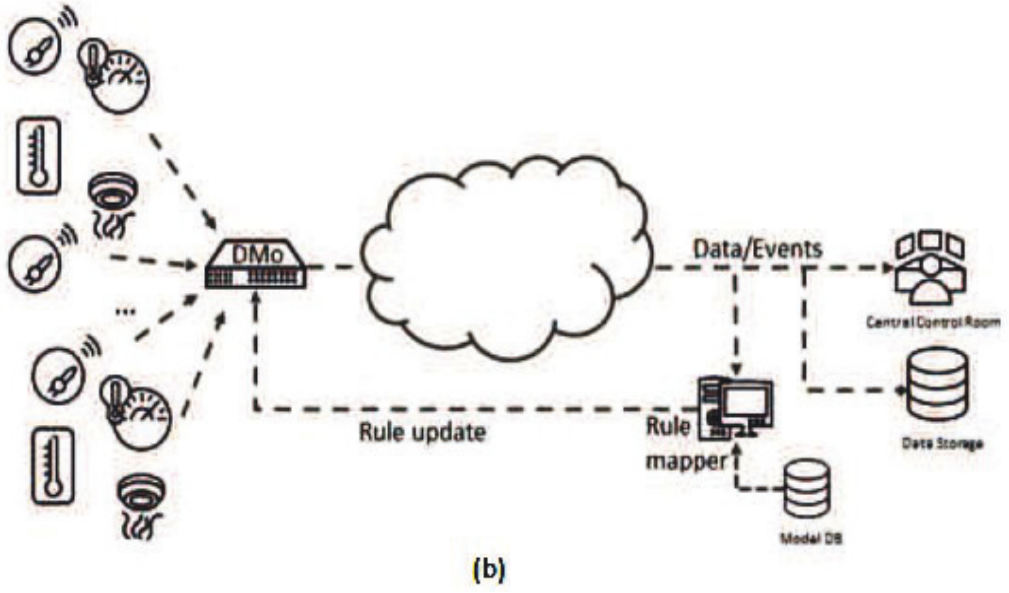
Figure 2.6 : Evaluating performance : a- Heap memory used (dimension in MB); b- CPU usage (dientional) [3]

better sustain the IoT as compared with the prevailing computing platform of Cloud. However, Yannuzzi et al. (2014)[51] examines some of the promising and challenging scenarios of IoT applicability and discusses the inevitable interplay of Fog and Cloud Computing in the near future, along with a review of some of the technologies that will require significant margins improvement so as to bolster the applications that the IoT market will requisite. A notable work Gazis et al. (2015)[4] reports on a novel scenario of using Adaptive Operations Platform (AOP) chased by evaluation scenarios for two specific cases and proposes an industrial-oriented solution for applications of inference maintenance utilising the Fog model and the technology of DMo. The setup is that of N interconnected sensors monitoring the status of the deployed machinery and aggregate the measurements to evaluate the performance of the Fog Components and report the captured values through a router to a central server, as depicted in Fig.2.7 a, further store the measurements in a database. This holds the objective of investigating potential machinery anomalies. Testing is done amidst a common router and a router or gateway (Fig.2.7 a and b) supporting DMo, for comparing the centralised scheme to the APO approach.

The first use case is the reduction of the amount of data received from the router supporting DMo, the generated traffic characterised by a Gaussian distribution, with most of the values within [50, 70] - the normal behaviour of the machine monitored, and potential anomalies falling outside it. In order to reduce the amount of data stored, it is possible to apply the AOP approach. Here, the server runs a Machine Learning (ML) algorithm (e.g., k-Means) trained to grasp the standard behaviour of the appliance, then and sets a rule to receive data outside the normal range representing anomalies and send to the operator. In this case, the average of the received values is estimated near 60, which corresponds to normal behaviour, updating a new rule to forward only the values representing anomalies. The range of values can differ according to thresholds, and the results are illustrated in Fig.2.8.



(a)



(b)

Figure 2.7 : Scenario using a normal router (a) and a router supporting DMo (b)[4]

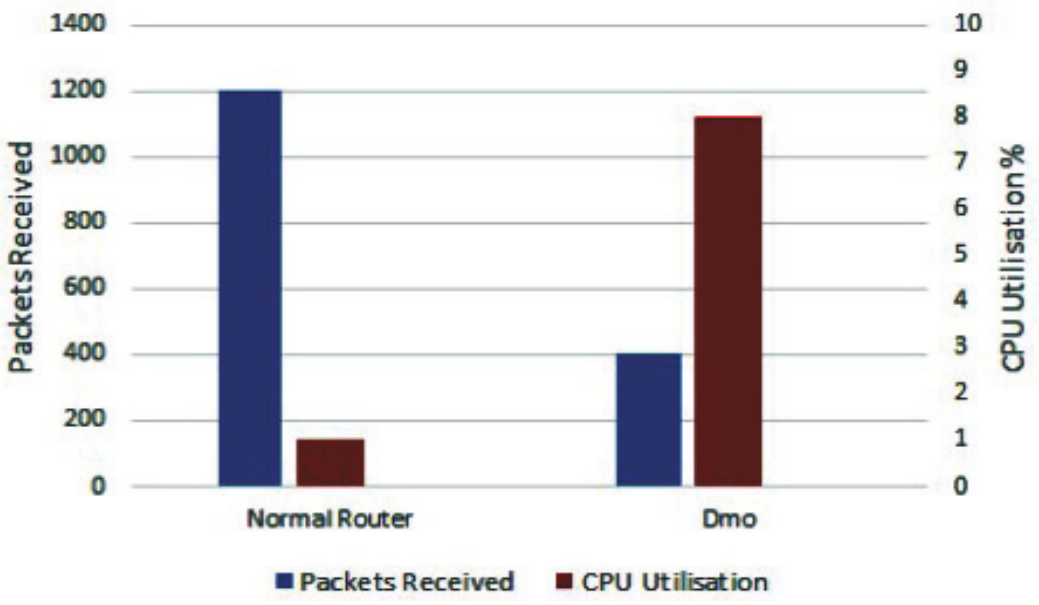


Figure 2.8 : Router CPU usage and the total number of packets received

As expected, the number of transmitted packets has decreased drastically and the DMo use claims only 8% of the CPU for a 1/3 reduction in the received traffic. A second use case targets demonstrating the proposed system capability to dynamically change the rules. Specifically, the temperature inside the rack with appliances is to be measured to alert the operator only when it exceeds a certain threshold (an anomaly). In order to save bandwidth and other resources in the server, a rule was created in the DMo router to send only measurements from the rack sensor. Analysing Fig. 8, an estimate is obtained that an abnormal event occurs inside the rack without designating which appliance has an anomaly. In this case, the server receives values above the threshold and it sends a new rule (created by the Rule Mapper) to the DMo for the router to forward the data from all the sensors. This produces a temporary increase in received packet number, helping identify the appliance with the anomaly. At this point, the ML identifies the sensor sending this surplus data above the threshold, then instructs the DMo router to transmit only from this sensor in addition to the rack sensor. This rule is applicable till the rack

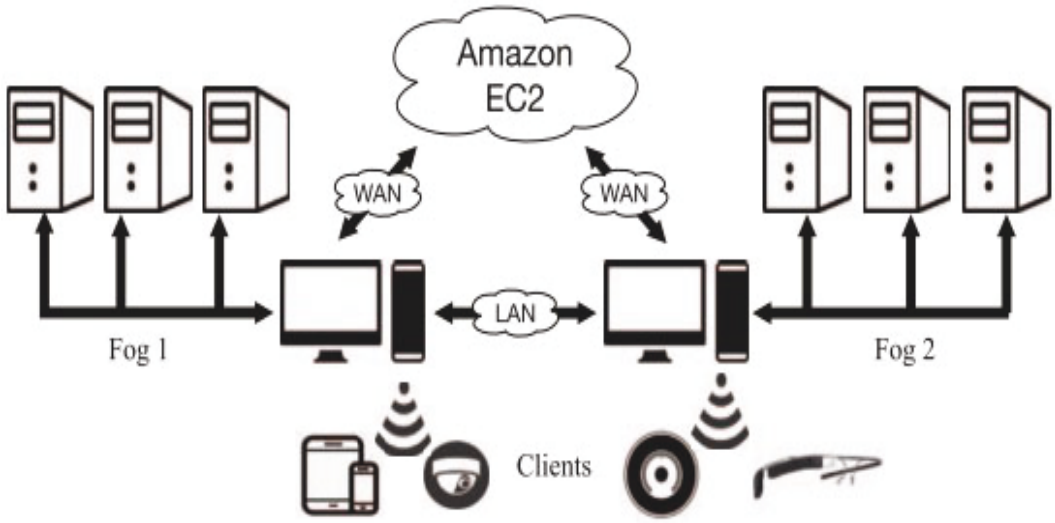


Figure 2.9 : Fog Computing Platform Yi et al. (2015)[5]

situation returns to normal conditions. After that, the operator will again receive the measurements of the rack sensor alone. According to Stojmenovic (2014) [63] Fog Computing function, being the main class of Cyber-Physical-System (CPS) has numerous applications in smart grids, networked vehicles, wireless sensor/actuator networks and control of smart buildings with Cloudlets as an important special case. Yi et al. (2015) [5] has built a proof of concept Fog platform, with two fog subsystems having 'OpenStack' installed in and possessing 1 router and 3 servers each. Those routers were inter-connected with through LAN, towards the Cloud (Amazon EC2) by WAN, and unified with Wireless AP function. Four OpenStack modules were installed: Keystone for security, Glance of Virtual Machine image management, Nova as a compute entity; and Cinder a block-level storage module. When comparing latency and bandwidth, Fog has stronger advantages over Cloudlets.

VM migration is vital in Fog Computing and its function was applied in two means: first, Fog 1 capturing a snapshot of the VM to be voyaged, compressing and later transmitting information to Fog 2., in which it decompresses and re-launches that VM; second, Both Fogs save a VM base snapshot, with only the incremental

part transmitted. The user will only remark Transmission time + Post-transmission time and the incremental method is better in this experiment. A running face recognition application across a smartphone and a Fog or a Cloud was implemented as well. Similar actions were run on Fog as well as on the Cloud (Amazon EC2). Nevertheless Shi et al. (2015) introduce an alternate option to Fog hierarchical view by enabling device clouds to reciprocate in a P2P way with smart device or sensor clouds by focusing on the use of the IoT protocol CoAP as a method of linking clouds of sensors & smart devices via mobile devices with users [64].

2.3.4 Mobile Edge Computing

Mobile Edge computing, also known as Multi-access edge computing (MEC), is a network structural design model, defined by ETSI Standards Developing Organization, that supports the capabilities of cloud computing and the services of an IT environment near the front end devices of a cellular network and at the edge of any network [65][66]. The main purpose of MEC is to reduce network congestion and allows a better application performance through running the applications and performing correlated processing duties nearer to the cellular customer. MEC technology is deliberated to be carried out at the edge nodes such as cellular base stations, and facilitates rapid and flexible distribution of newly added applications and services[67].

A vital upcoming target setting for MEC deployments is 5G networks based on the specifications of 3GPP 5G. The system specification of 5G and its architecture facilitate the service-based interlinkages among various network functions. Such specification allows the alignment of system operations with the Software Defined Networking and network virtualization models, which is similar to the features of MEC specifications. Moreover, the enablers of edge computing are described by 3GPP 5G system specifications, which allow 5G and MEC systems to cooperate in

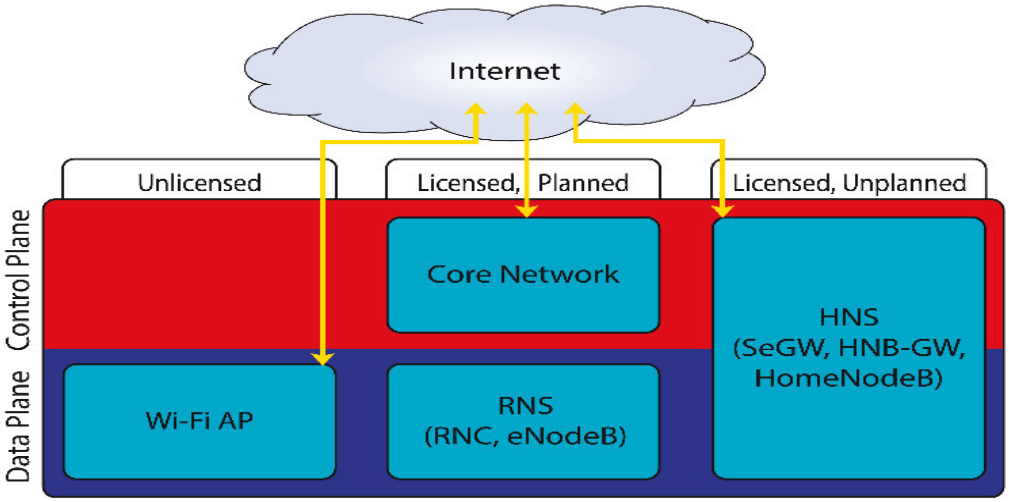


Figure 2.10 : Case study: Co-existence of heterogeneous networks may be managed in part by clients, Chiang(2016) [6]

policy control and traffic routing processes. Edge computing enablers are crucial in the support for integrating MEC in 5G networks creating a robust environment for edge computing [68][69].

Chiang (2016) [6] examine a client-based HetNets control, a use case implementing 3GPP standard, where management, control, inference and configuration of Fog control plane networks were illustraited.in the case study, a client can examine its local settings and choose a network to join. This local arrangements can converge globally towards a desirable configuration by performing randomisation and hysteresis.

Considering ETSIs MEC framework and reference architecture (defined in the Group Specifications (GS) MEC 003 [70]), which declares and clusters the systems high-level functional units [7], a respectable possibility for this research to suit within the framework can be found. In the proposed blueprint operational framework in figure 3.2, the front end devices can easily fit in the network level components of MEC framework. Likewise, the Fog-Decision level of the proposed blueprint can

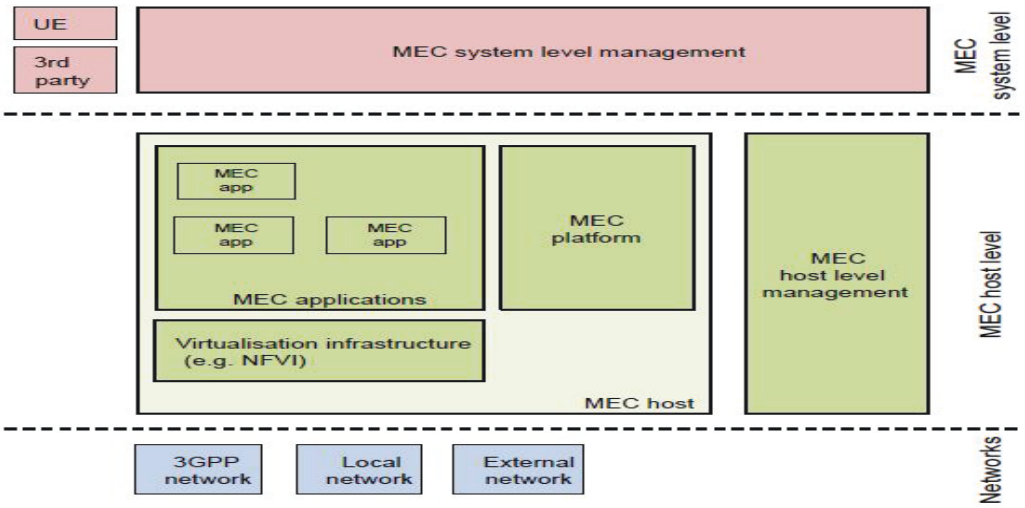


Figure 2.11 : ETSIs MEC framework and reference architecture [7]

be fitted in the MEC host level management entity, while the MEC system level is retaining the global view of the whole MEC system similar to the gate way level in the proposed blueprint.

2.4 Data Aggregation†

The objective of Data aggregation processes is to minimise the amount of exchanged data in a network and accordingly reduce the packet overhead and improve energy efficiency. Consensus aggregation is the process that use voting to reach an agreement about the way to collect information coming from different sources and push the results to the ascendant node. In this section, some literatures concerned with data aggregation methods are introduced to support the proposed approach.

†parts of this section come from my paper "A review of aggregation algorithms for the internet of things"(2017)

2.4.1 Data Aggregation Reviews

In order to understand the classifications and methodologies used to implement data aggregation, several survey and review papers were studied. Vodel and Hardt (2012) present data aggregation common methods, including the adapted communication process and analytically explain theory benefits and compare these theoretical advantages with measured real-world results. They have intentionally presented a roadmap for corresponding data aggregation method's advantages and disadvantages in a resource-limited situations and indicate the significance of the difference between theoretical data aggregation notions and experimental practices and that the use of elementary uncoordinated aggregation structures in collaborated sensor network setting. This negatively stimulates and effectiveness on communication behaviour specifically in terms of limited bandwidth architectures of embedded systems. Also propose practicable means for optimising data aggregation procedures and avoiding distributed sensor network architectures negative effects. Approaches like data compression, data fusion and data aggregation have to be performed to decrease the volume of communicating info so as to lessen the transmission function's relevant power consumption [71].

Chhabra and Singh (2015) [72] report on the practices of minimising the detected data using single node of a sensor and have surveyed the influence of the current aggregation protocol. Different aggregation methods can be implemented based on the resiliency in various applications. For saving the energy of WSN by reducing massive amount of transmission, the clustering protocols and data processing at a single nodes methods can be effective functions of data aggregation. Also, the data fusion helps in making decisions, which could not be possible by reading an individual sensor node and can be done irrespective of boosting the lifetime of network. The data fusion has been defined as the method of deploying data integration extracted from several sources and gathering such information for attaining

inferences, correlations and association, which are much effective and possibly more precise than if achieved by a single source. While, for reducing or eliminating the redundant data, the process of summarizing the data inputted from various SNs is done with data aggregation which in turn is a subgroup of data fusion. Also, the factors like, association among the sources, the level of abstraction and the link between input and output has been categorized as part of data fusion. For the relationship among sources there are three classes: Complementary, Redundant and Cooperative; whereas the level of abstraction contains: Signal, Pixel, Feature and Symbol levels. However, the relationship between input and output class includes: Data in - (Feature-out), Data (In) - Data (out), Feature in (Feature-out), Feature in (Decision-out) and Decision in (Decision-out) sun-classes. SNs are designed for building up a tree and classification of tree structure is classified for aggregation protocols. The hierarchical and planar protocols are the wider classification of the tree structure and the hierarchical algorithms are sub categorized as, cluster (structure), cluster (tree-structure) and cluster (grid-structure) algorithms. However, query (routing based), chain (routing based) and suboptimal aggregation tree algorithms are the classifications of planar algorithms.

Rajagopalan and Varshney (2006) [73][74][75] introduce a survey of data aggregation procedures in WSN, with contrasting and comparing various algorithms based on performance measures such as lifetime, data accuracy and latency. They argue that data aggregation algorithms mainly focus on efficient organization, routing and data aggregation tree construction. The main characteristics, the advantages and disadvantages of each algorithm were explained and the discussion of special characteristics of data aggregation such as source coding and security was extended. They argue that data aggregation algorithms mainly focus on proficient organization, routing and data accumulation tree construction. The main characteristics, the advantages and disadvantages of each algorithm were explained and the discussion

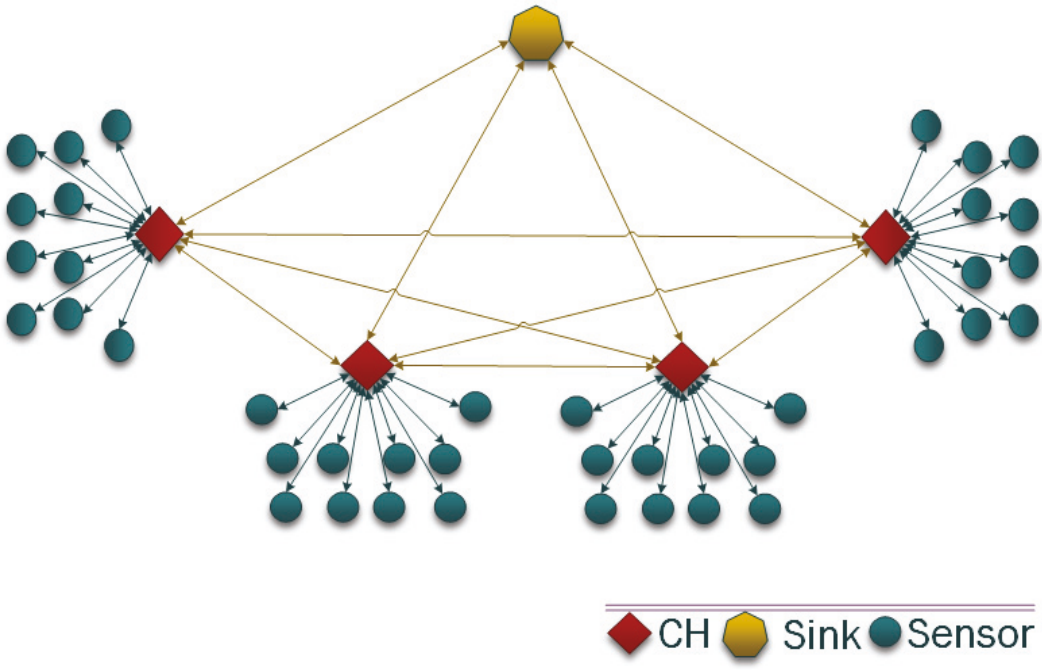


Figure 2.12 : PBFT example

of special characteristics of data aggregation such as source coding and security was extended. Also, the compromise of energy efficiency, latency and data precision has been emphasised and the protocol of data aggregation performance was combined with network bases. Data aggregation classifications explained in this paper were displayed as shown in Fig 2.14.

Shu et al. (2011) [76][77] Summarise a special problem that challenges the distributed intelligence and data fusion for sensor systems from the viewpoint of data aggregation and data storage, coding and channel allocation, security, routing, mobility and distributed services. How to enhance energy optimisation and power saving methods is the pivotal concern in WSN and therefore, data fusion and distributed intelligence can boost this method by making them smarter, flexible, adaptable, safe and scalable.

Stojmenovic (2014) [63] examines CPS beyond M2M model and considers futuris-

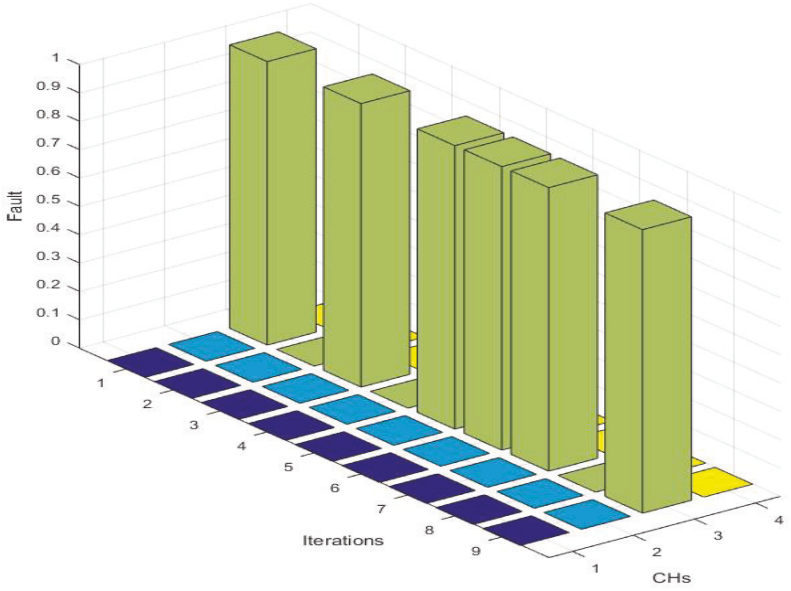


Figure 2.13 : PBFT example results

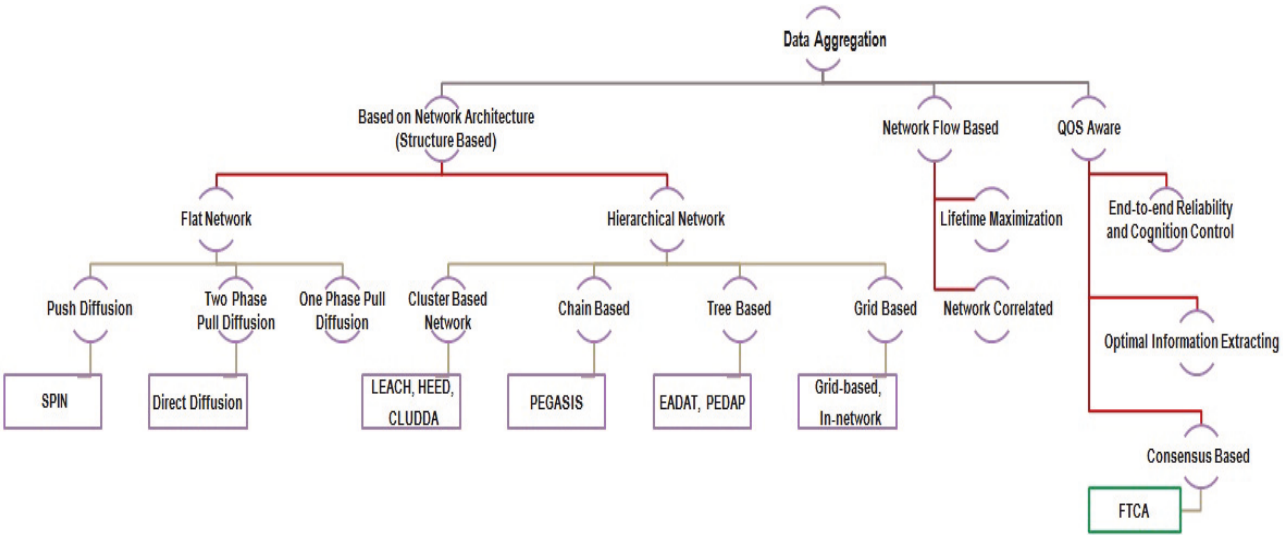


Figure 2.14 : Data aggregation classifications

tic applications and explain few particular scenarios that motivate the improvement of the M2M communication primitives fitted to large-scale CPS. None of the nodes in a cooperative network communicate directly with each other, but rather via gateway where the gathered information are then aggregated at possibly multiple layers of aggregation nodes. The aggregation function can decrease the amount of information retransmitted at each aggregation layer by filtering information based on relevance or by eliciting higher-level information from aggregated data. In order to enable a system of billion nodes, the data aggregation is deployed sot that devices i.e. (M2M) can be cost and energy efficient and have a limited operating field while post processing facilities and storage applications might be assisted by the Cloud. The presented communication and coordination paradigm can be exemplified as in Figure 2.15, assuming that the only criterion for choosing responding robot is the distance to the event for simplicity. The robots at distance 11 were reported about a fire event, so it initiates auction for perceiving nearest robot for the event by referring the neighbouring robots at mentioned 15, 10 and 5 distances. The one at distance15 evaluates which its neighbour at distance twenty, as well as all the other ones are more likely connected to it, will not be designated as the best answerers, so it will not ask it, in this case, it replies back (as shown in the green arrows) choosing itself as best offer. Then Robot at distance ten asks its neighbours (at distances twelve and eighteen) and the neighbour at distance twelve discoveries that its fellow neighbours (distances eighteen and nineteen) are not competitive enough, so it will not ask them. However, instead of that it suggests itself as the best service provider. The Robot at distance five negotiates its neighbour at distance eight, then suggests itself to robot at distance ten, which then answers the auctioneer robot about best option. Then the auctioneer robot requests the 'winner' at distance five (through yellow arrows) to appear at the event.

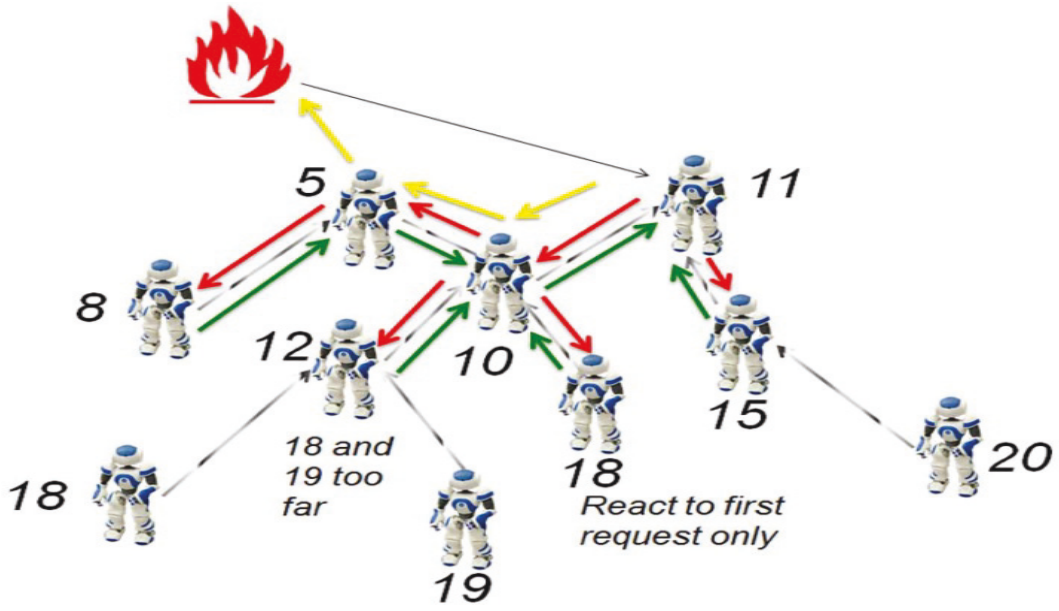


Figure 2.15 : Choosing responding Robot criteria. The robot at distance eleven holds an auction (red arrows used for contacting, green is for bids) in order to choose the nearest responder, which is the robot at distance five (through yellow arrows). Stojmenovic (2014)[8]

2.4.2 Data Aggregation Models

There have been a lot of researches done related to data aggregation models, algorithms, schemes and protocols in order to decrease the amount of data traveling through the network and improve the efficiency of its resources. Data aggregation involves the process of forwarding a synopsis of several data packets rather than the whole packets. One of the practices of the aggregation process is to produce a tree rooted at the final destination of the information whose leaves are the sources of the measured data. Snader et al. (2007)[78] introduce intelligent aggregation algorithm in WSN "Tethys", where the resolution to where and when to aggregate depends on cost and aggregation efficiency as well as the higher bound to transfer data from source to destination. This model was then used to create a lightweight, powerful, dynamic, distributed aggregation tree creation protocol. They also has explained

the issue of aggregating different kinds of data in WSN. While Hoang et al. (2012) [9] adopt an optimisation algorithm to make the optimal data aggregation trees in WSNs in which improvements to the basic IWD algorithm to enhance the structure of the tree by trying to boost the probability of choosing optimum aggregation nodes(see figure 2.16). Similarly, Commuri and Tadigotla (2007) [79] address the problem of applying dynamic data aggregation in WSNs through the proposal of reconfigurable cluster heads (RCHs) using FPGAs where various data aggregation algorithms can be professionally applied in run-time. When examined the model it showed the power use and processing periods of request, which quickly grows as the amount of aggregation operations rise. However, Chen et al. (2008) [10] suggest an 'adaptive data aggregation (ADA)' structure for clustered sensor networks in which the degree of timely-based aggregation is organised through reporting frequency at the nodes. In the same time cluster heads control the space-based aggregation by the aggregation ratio then they would be calculated by the current system state according to the reliability. The purpose of the ADA structure is primarily accomplished at the sink, leaving a tiny task at CHs and sensor nodes. The application of a single-hop clustered sensor network is considered concerning discovery of activity features depending on data collection of various sensors observing the activity. Assuming an activity occur at an area near the sensors, the sensors can sense the activity and send the data to the related CH which would perform the aggregation process and send the aggregated information to the sink as illustrated in Figure 2.17. The results of analysing and simulating the system converges the wanted reliability commencing from an arbitrary initial state. Bohm et al. (2010)[11] introduce a monitoring system performing a large-scale distributed computation setting as a first stride toward scalable system monitoring. The method based on categorizing all the collected monitoring entities relying on singular requirements and also on aggregating information that has modules of singular monitoring entities which use

the practice of network' tree-based overlay. Its prototype is capable of decrease the volume of collected monitoring info. The Figure in 2.18 shows the entire monitoring system architecture where the procedure of back-end are positioned on all of the nodes and execute monitoring with a classification of gathered entities so as to decrease the volume of the transmitted info. The medium procedure, located in a subset of the nodes, perform aggregation gathering entities in a tree style. The front procedures implements additional computations then saves the results in a database to be processed through extra tools. In addition, Park et al. (2008)[12] as well as Enam (2014) [80] examine the elimination of redundancy. While Park et al. (2008)[12] introduce a novel collaborative data reduction process for eliminating the redundancy coming from various sensors by using a tree-based model for data propagation to demonstrate the procedure of collaboration between many sensors. In the order to relieve time-delay issue during aggregation processes, the scheme separates the data aggregation process from the collaboration process, so that enough data to detect faults could be captured at the time of decreasing the info dimension as validated through the experimental outcomes. PCA and partial correlation were utilised to capture the linear redundancy while non-linear redundancy could be erased through applying the process in a kernel space. This scheme has 2 phases, collaboration and a data reduction. This is demonstrated in Figure 2.19. In the first phase, each sensor recognises the partition that its collaborative data reduces so as to eliminate the redundancy by using a tree-based data propagation information is important or non-redundant when it counts other sensors information. A middle device would take the information given by the leaf devices and execute the process of generalised data reduction. It results in a small dataset parts, denoted by an indices set and saved in the sensor memory till following collaboration phase, which need to be transferred. Data partitions are send over in the second phase, according to the indices. Enam (2014) [80] established an adap-

tive model of data aggregation that employs the space-based association between sensors. The main characteristic of the technique is that it reduces the redundancy transmission cost in the network. Also, it optimally exploits the obtainable space within packets at CHs. The simulation outcomes have revealed that the payload size requirement reduces to almost quarter of the non-compressed one and that the distortion percentage declines by 16% to 41% in comparison with the mean aggregation method. As related to clustering-based aggregation, Gionis et al.

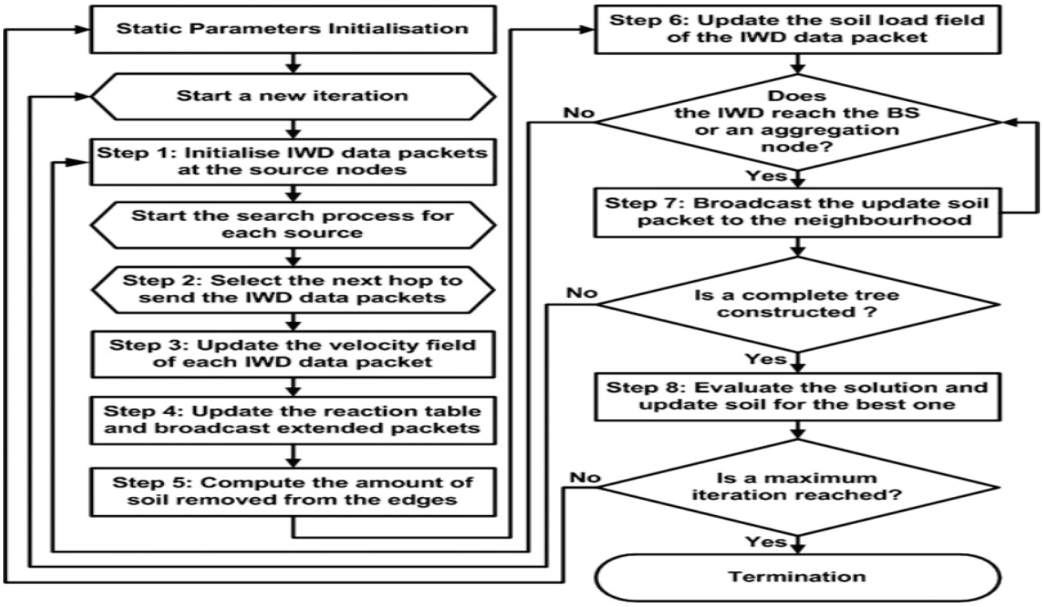


Figure 2.16 : Flowchart of the IWD algorithm for constructing data aggregation tree in WSNs Hoang et al. (2012)[9]

(2005) [81] proposed an approach that group a setting of objects within a cluster which has the ability to acknowledge with a an existing clustering as much as possible then explain many applications regarding to clustering aggregation containing clustering categorical data, handling heterogeneous data, detecting outliers, and enhancing clustering robustness. They also presented some procedures dealing with the problems of clustering correlation and aggregation taking into account a sampling procedure that can handle large amount of data without significantly losing

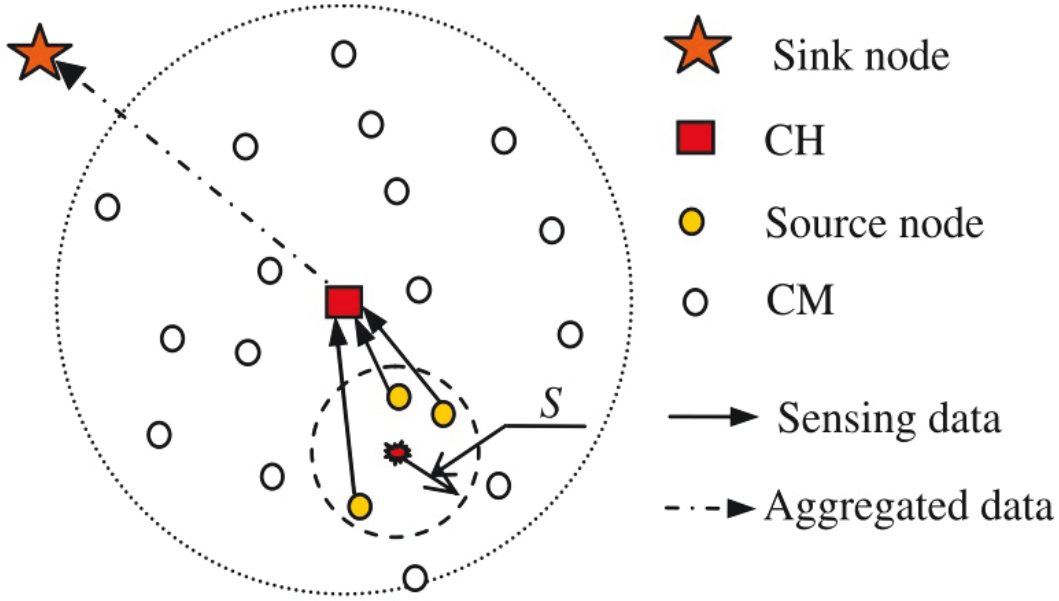


Figure 2.17 : Data aggregation process in single-hop clustered WSN Chen et al. (2008)[10]

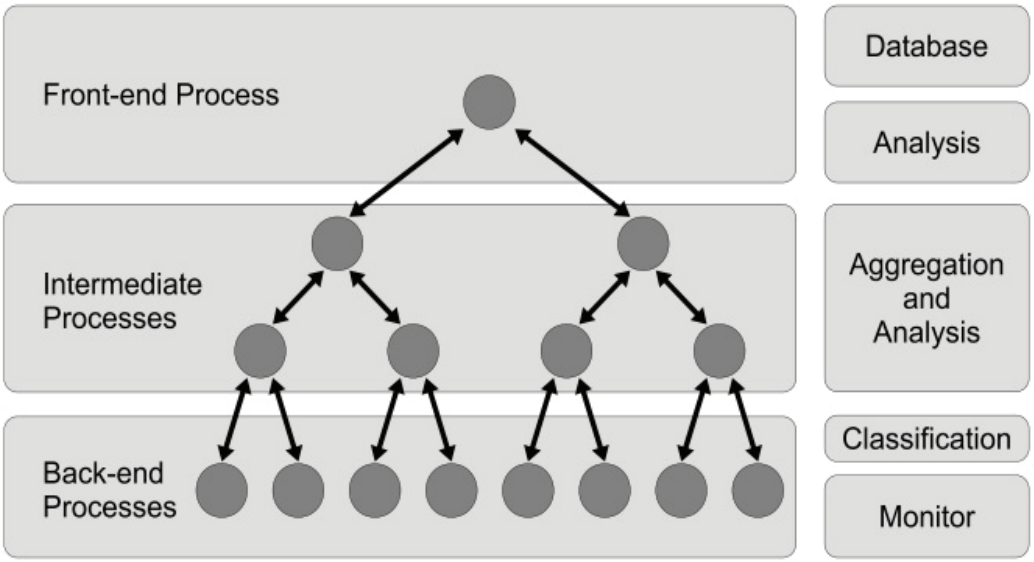


Figure 2.18 : Employing a tree based overlay network, like MRNet, as a scalable aggregation / analysis for real-time observation data. Bohm et al. (2010)[11]

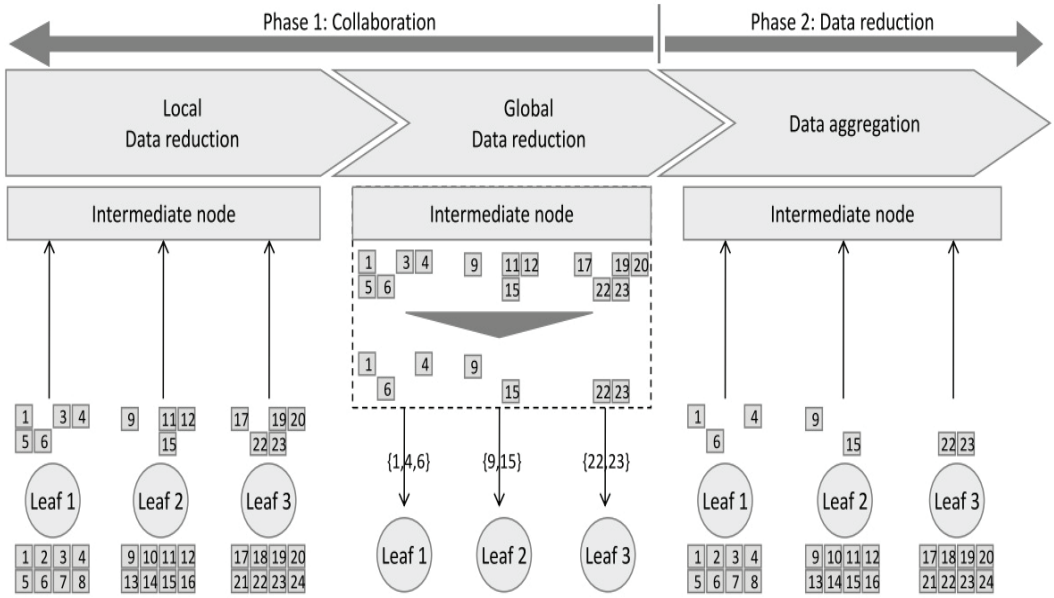


Figure 2.19 : Overall process of Collaborative data reduction method Park et al. (2008)[12]

the quality. Guo et al.(2014) [13] discuss the problem of reliable data aggregation route in WSNs which is a multi-objective and non-linear constrained optimization problem . Firstly, to enhance the energy efficiency in the WSN, a data aggregation adaptive route algorithm was proposed in that the construction process attained using 'discrete particle swarm optimization (DPSO)' so as to save the energy expenses and construct better routing tree taking into consideration communication and aggregation cost. Secondly, an adaptive route algorithm was offered to equilibrium the network load and launch a reliable network. The outcomes indicate that the proposed algorithm can effectively decrease energy consumption and trade off network lifetime in comparison with other tree routing algorithms. The network diagram is shown in Figure 2.20. Proposing a fault-tolerance techniques about data aggregation had gained a good attention. Iskander et al. (2012) [82] introduced and analyse a confidentiality preserving in-network aggregation protocol with fault-tolerant for placements of collaborative WSNs. The protocol permits the collection of data as

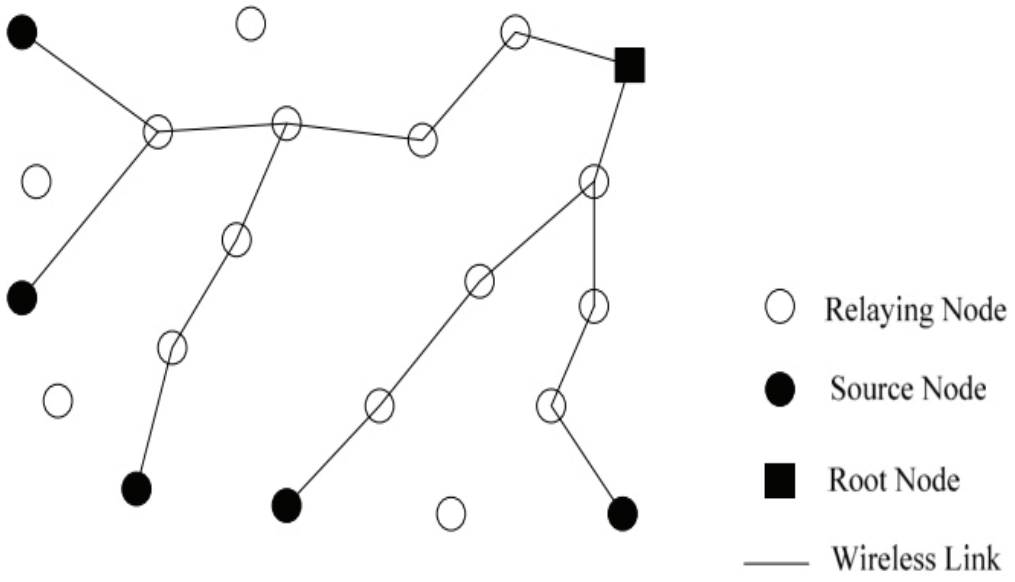


Figure 2.20 : Network model diagrams Guo et al. (2014)[13]

preserving end-to-end confidentiality for the aggregated outcome and the singular sensor readings. This protocol promises that with high probability all sensor readings will engage to the concluding aggregate through techniques of error detection and error correction. The presented scheme aims to achieve confidentiality where sensor readings and their aggregate values are only revealed to the sink rather than any external or internal attacker, fault tolerance in that lost sensor readings due to link errors are compensated through a parent node, exact aggregation outcome if there is no link failures instead of executing probabilistic query aggregate outcomes and if link failures exists, the concluding aggregate deviates by precisely the lost worth (not by some derivative of that worth), and low energy overhead on the size of packets transferred and amount of computation. A fault tolerant distributed method which can be made over topmost aggregation process and can generate correct results even in the existence of node failures was developed in Gansterer et al. (2013)[83] along with an aggregation algorithm for averaging or summing dispersed values, the push-flow algorithm that can reach higher flexibility characteristics in

connection to failures in comparison with present aggregation methods. Thus, a survey related to existing DDAAs and the study of their pros and cons in terms of fault tolerance was prepared with a focus on enhancing fault tolerance and explaining the new push-flow algorithm to examine the creation of proper distributed algorithms for matrix calculations constructed over the distributed data aggregation algorithm (DDAAs). Applebaum et al. (2010) [14] implements a privacy-preserving data aggregation (PDA) within a big quantity of members, where efficiency and scalability is attained via a partially-centralised design which splits accountability amongst a proxy which ignorantly covers the user incomes and database which sums data with keywords (blinded) and recognises such keywords whose values fulfil some assessment function. The scheme leverages a cryptographic protocol which provably guards the privacy of the keywords and the participants, as long as that proxy and database do not conspire, even if both of them may be independently malicious. The protocol contains five steps (as in Figure 2.21). The proxy interrelates with the participants, in the first two steps, to gather the blinded keys along with their related values encoded by the DBs public key, and deliver them to the DB. Then the DB aggregates the blinded keys, in the next two steps, with the related values in a table, then elects which rows should be exposed in accordance with a predefined function. Finally, the DB requests the proxy to unbind the consistent keys. Steps 4 and 5 are conditional statement additions to PDA, as well as extension input in step 2 (all shown in blue). Fs is a keyed hash function whose key s is identified only by the proxy. Bao and Lu (2015) [15] introduce a novel secure data aggregation model that can attain fault-tolerance as well as differential privacy in parallel 'DPAFT'. An artful constraint relation is assembled for data aggregation motivated by the key exchange protocol of DiffieHellman. With this constraint, DPAFT can provision fault tolerance for malfunctioning smart meters flexibly and efficiently and it can also improve to resist in opposite to differential attacks by enhancing

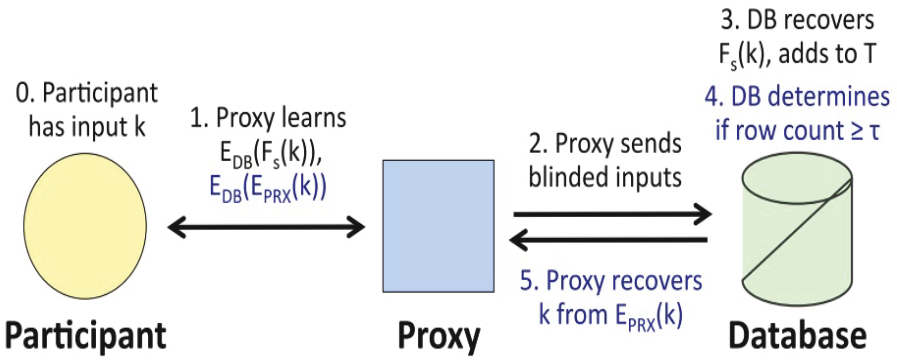


Figure 2.21 : High-level system architecture and protocol.Applebaum et al. (2010)[14]

the elementary cryptosystem of BonehGohNissim to be more appropriate to the practical circumstances. They conduct a comprehensive performance evaluations to show that their scheme outperforms the modern data aggregation models in considerations with storage cost, utility of differential privacy, computation complexity, the user addition and deletion efficiency, and robustness of fault tolerance. They study a classic smart grid communication design for residential users, that contains a residential gateway to aggregate data and forward it in a secure way, a trusted authority to organise the system, a large number of residential users in a residential area which supplied with a smart-meter as well as several smart-appliances to gather the real-time data and report them in a specific period, and a control centre that collects, processes, and analyses the real-time data as illustrated in Figure 2.22. A case study that gives an outline of the differences in resolution reduction, aggregation and perturbation of real-life energy consumption data in the Internet of Things (IoT) was proposed in Pohls et al. (2015) [84] where privacy, accuracy, computational overhead and compression-ratio of selected perturbation and aggregation methods was analysed. A real-life data set of in depth energy consumption logs of a particular family household was measured and privacy by simple, threshold-driven machine-learning algorithms was introduced which extract behaviour features. The

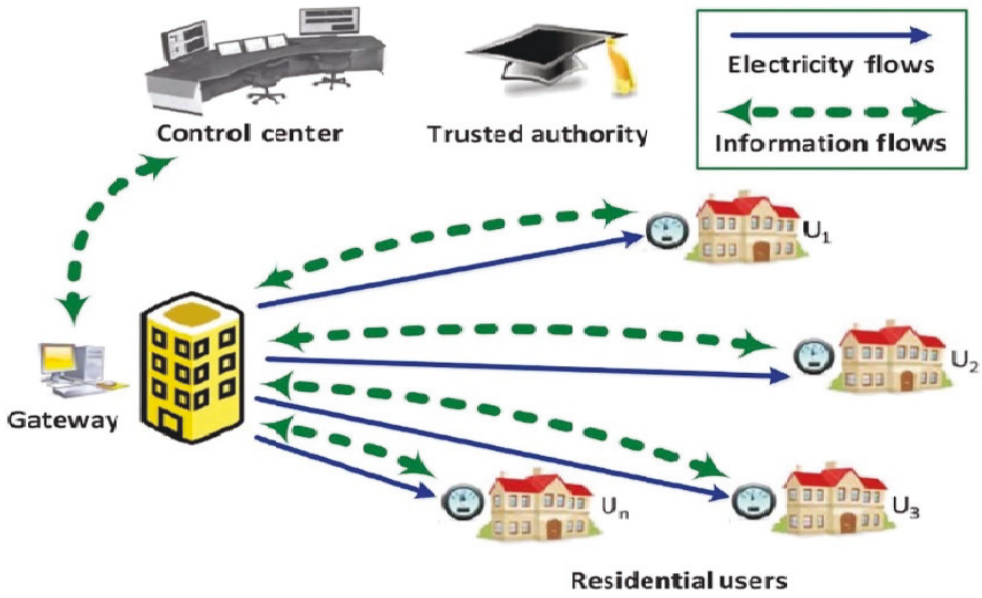


Figure 2.22 : System model under consideration Bao and Lu (2015)[15]

correctness of this extraction is used as privacy metric. The outcome is that a lot of detections for sensible estimates and intelligent responses are still promising with lower quality data and he damage in data quality can always be seen as a privacy gain.

2.5 group Decision Making

2.5.1 Group-based Recommendation Systems

In order to obtain consensus and amend the system policy, Recommendation Systems are used so that the system can grasp personalised demands and present customised services. Figure 2.23 shows the Consensus pattern which consists of one of the Recommendation System models that may include one of the aggregation strategies to obtain consensus values. There are five models of recommendation : the *general model* which avails crowd wisdom and advise the most popular items in one recommendation list for all users, the *personal model* that uses the standards Collaborative filtering algorithm to examine users individually and generates one

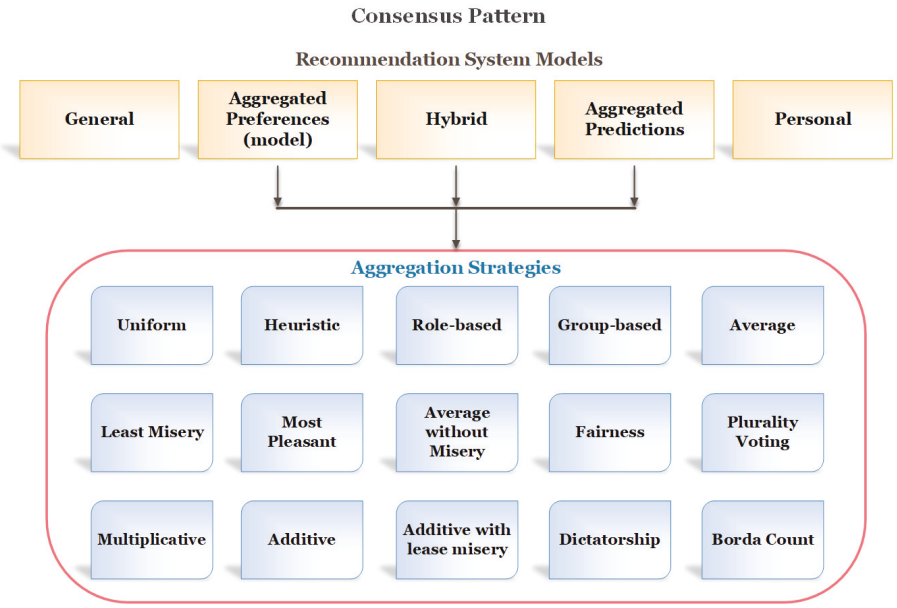


Figure 2.23 : Consensus Pattern

list per each user, while the *aggregated preferences model* and the *aggregated prediction model* exploit the group-based recommendation procedures which generate one list for each group. Many forms of hybridization were made to combine the group-based models forming the *hybrid model* such as the switching scheme that switches between the models according to system's condition. Maney efforts were made to investigate the GRS models and strategies. Berkovsky and Freyne (2010) examine the usage of several existing group recommendation models (Generic, Aggregated Models, Aggregated Predictions, and Personalised) and analyse the effect of switching model on the performance of the system [85]. While De Pessemier et al (2013), Carvalho and Macedo (2013), and Hu et al (2013) research the group-based models (Aggregated Models and Aggregated Predictions) along with a third model (combined model for De Pessemier et al (2013), generic model for Carvalho and Macedo (2013), while Hu et al (2013) propose a Deep-architecture model) [86][87] [88].

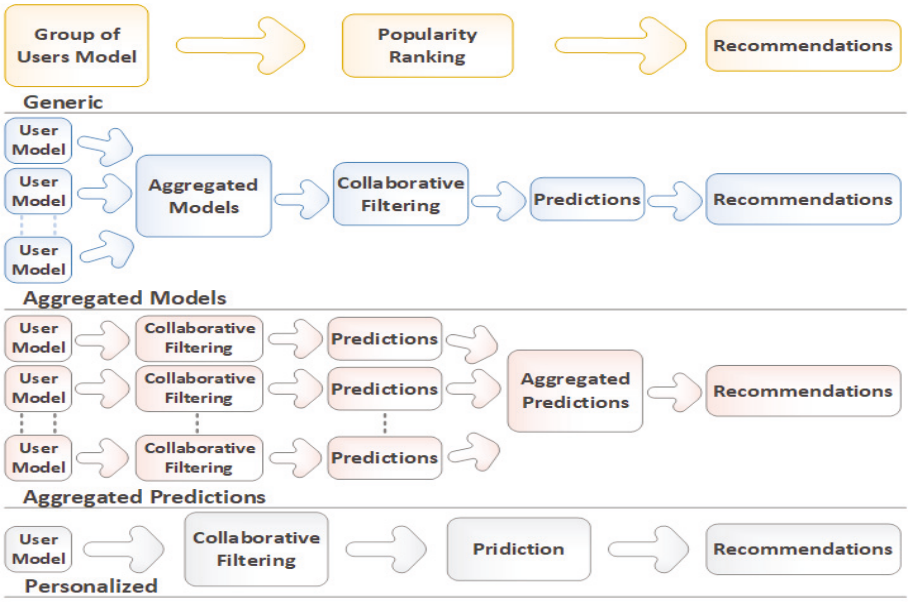


Figure 2.24 : Recommendation generation process

2.5.2 GDM model

For the GDM system to be implemented within the network as automation approach, let $A = \{a_1, a_2, \ldots, a_n\}$, be a finite set of feasible alternatives, $(n \geq 2)$, to be classified from best to worst by using the data provided by a set of participants or decision makers, $P = \{p_1, p_2, \ldots, p_m\}$, $(m \geq 2)$ whose weight vector is indicated as $G = \{g-1, g_2, \ldots, g_m\}^T$, where $g_k > 0$, $k = 1, 2, \ldots, m$, and $\sum_{k=1}^{m} g_k = 1$. Each participant presenting its preference relation on A_i as $a_{ik} \in S$, and S is an ordered finite set of labels $S = \{s_0, s_1, \ldots, s_t\}$, in which $s_i > s_j$ for $i > j$.

As presented in [89, 90, 91] and many others, there was an assumption that for each participant $p_k \in P$, the preferences over alternatives' set A could be represented as one of the four following ways:

- Alternatives'preference orderings . In this circumstance, $O^k = \{o^k(1), \ldots, o^k(n)\}$, where $o^k()$ is a permutation function over $\{1, \ldots, n\}$ index set for the participant p_k, outlining an alternatives ordered vector, from best to worst.

- Utility functions. For this circumstance, the participant pk provides its preferences as $U^k = u_1^k, \ldots, u_n^k$, $u_i^k \in [0,1]$, where u_i^k is the utility evaluation of p_k to the alternative a_i.

- Fuzzy preference relations. In this circumstance, the preference are expressed by a fuzzy preference relation $F^k \subset A \times A$, with a membership function, $\mu F^k \subset A \times A \to [0,1]$, where $\mu F^k(a_i, a_j) = f_{ij}^k$ denotes the preference degree of a_i over a_j.

- Multiplicative preference relations. For this, the preferences are ex-pressed by a positive preference relation $A^k \subset A \times A$, where the preference's intensity a_{ij}^k, is calculated by means of a ratio scale, precisely the $\frac{1}{9}$ to 9 measure.

Within this context, the resolution procedure of the GDM involves attaining a set of solution alternatives from the Participants preferences. As it assumed that the participants preferences are given in different ways, so the first step should deal with obtaining a uniform representation for the preferences. These ways can be transformed into the different representations by using different mathematical transformation functions. In this paper, we will consider multiplicative preference relation as the basis for information uniforming. When the uniform exemplification has been accomplished, we can apply the analytic hierarchy process (AHP), as it uses multiplicative preference relations, to gain the solution set of alternatives. This resolution process is represented in the next section.

2.6 Consensus

There has been many literatures introduced covering the area of voting/consensus aggregation. Adaptive Cumulative Voting-based Aggregation Algorithm (A-CVAA) was studied by Saeed and Salim (2013) to combine many clustering of chemical structures while Muravyov and Khudonogova (2015) suggest a preference aggrega-

tion method for multisensory accuracy improvement depends on interval voting. However, Yu et al. (2007)[92] and Farnoud et al. (2012)[93] consider the voting aggregation problem and procedures. The efficiency of clustering was assessed in Saeed and Salim (2013) relay on the capability of clustering to distinct the active from the inactive molecules in every cluster then the outcomes were related to Wards technique. Experiments propose that the 'adaptive cumulative voting-based consensus' scheme, which includes two main phases the partitions generation and combination using the consensus function, can efficiently enhance the effectiveness of merging various clustering of chemical structures. The approach in Muravyov and Khudonogova (2015) [16] permits to determine an amended value of a measured factor based on imprecise measurement data, obtained from neighbouring multi-sensors. Kemeny rule is used to find a resulting intervals to determine consensus relation that introduced as ranking which can contain a strict order relation and an correspondence relation. voting problem is counted as a consensus relation determination problem, where a group of participants rank a set of alternatives. Locate every interval along a real line of finite length as in Figure 2.25. Core steps of the preference aggregation procedure are illustrated in Figure 2.26 where in the first stage, real values range is determined while preference profile is formed in the second and the profile matrix is calculated in the third stage. The fourth stage engage with the recursive branch and bound algorithm which uses Kemeny rule for determining consensus relation for a given profile where The aim is to compute a consensus relation that would provide a combined estimation of all alternatives. In the fifth stage, if many consensus relations are obtained, they would be convoluted into single relation. With the election voting scheme in Yu et al. (2007)[92], a few elimination voting models, including Kemeny approaches, were analysed in a graph theoretic method as an addition to Borda: a classical voting rules. A novel heuristic elimination voting algorithm is introduced in Farnoud et al. (2012)[93] as Kemeny ranking problem is

NP-hard. To evaluate the voting procedures on rank aggregation, a few experimentations have been executed on TREC data and they show that these elimination algorithms have equivalent performance with Borda algorithms, and even outperform it in some cases. Similarly, consider the algorithmic characteristics related to the aggregation process for non-uniform vote aggregation. Two diverse aggregation approaches were presented for a new phase of weighted distance measures on votes. The first procedure used Spearmans foot rule distance to approximate the weighted distance measure, with guaranteeing verifiable constant approximation while the second one is based on a non-uniform Markov chain method motivated by PageRank, for which presently only heuristic guarantees are recognised. The performance of the suggested procedures on a number of distance measures was illustrated for which the optimum resolution might be straightforwardly computed. Kumar et al. (2004) explain a fully distributed data aggregation and consensus protocol for object position and tracking applications installed within WSN which can decreases the amount of data to be exchanged to generate consensus and decreases the state information needed to maintain the structure of the network. The protocol fulfils agreement, termination and validity properties. When an event is detected, a local reading is compared with a fixed threshold probability by the node and it would execute consensus if that local reading has value more than the probability. Ultimately one or more nodes will commence consensus and a PROPOSE message will be directed to the other nodes in the communication neighbourhood. An acknowledgment reply will be received by the initiating node from other participating nodes and later it would evaluate the readings utilizing a simple majority vote to find the right outcome. Finally, a DECIDE message will be broadcasted by the node which generated a consensus request. By reaching a consensus, only a single message referencing the detected event requires to be sent to the tracking application at the base station, resulting in a substantial savings in communication costs

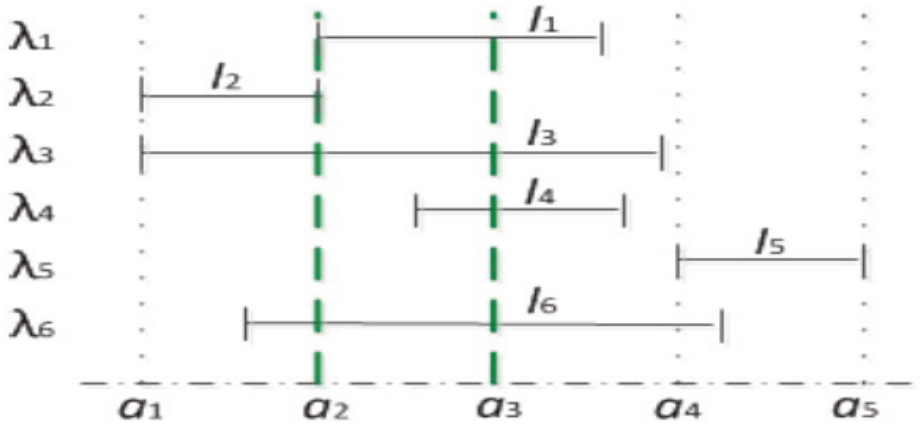


Figure 2.25 : Preference aggregation approach and Kemeny selection rule Muravyov and Khudonogova (2015)[16]

and extending network life [94]. Beliakov et al. (2014) Propos two consensus operators constructed from fuzzy implication operators and aggregation functions and explain the key properties with an adaptation to their definitions to the setting of inputs existed over the unit interval. Considering the consensus setting, both operators hold a fine semantic interpretation. They also disclosed how the choice of modules for the given consensus prototypes affects the fulfilment of these properties [95]. Ah-Pine and Corporation (2003) introduce data fusion using consensus aggregation functions where M rankings generated by M judges can be fused by a fusion system by calculating values of an aggregation function for items of the M rankings and building an aggregation ranking depends on the aggregation function values. Finally, informational content were outputted exemplifying the aggregation ranking[96].

Herrera-Viedma, et al. introduce a consensus model that use a comparison for the alternatives positions between the individual solutions and the collective solution. The model provides feedback proposing the way in how these experts should modify their preferences depend on the offset of individual solutions and the consensus

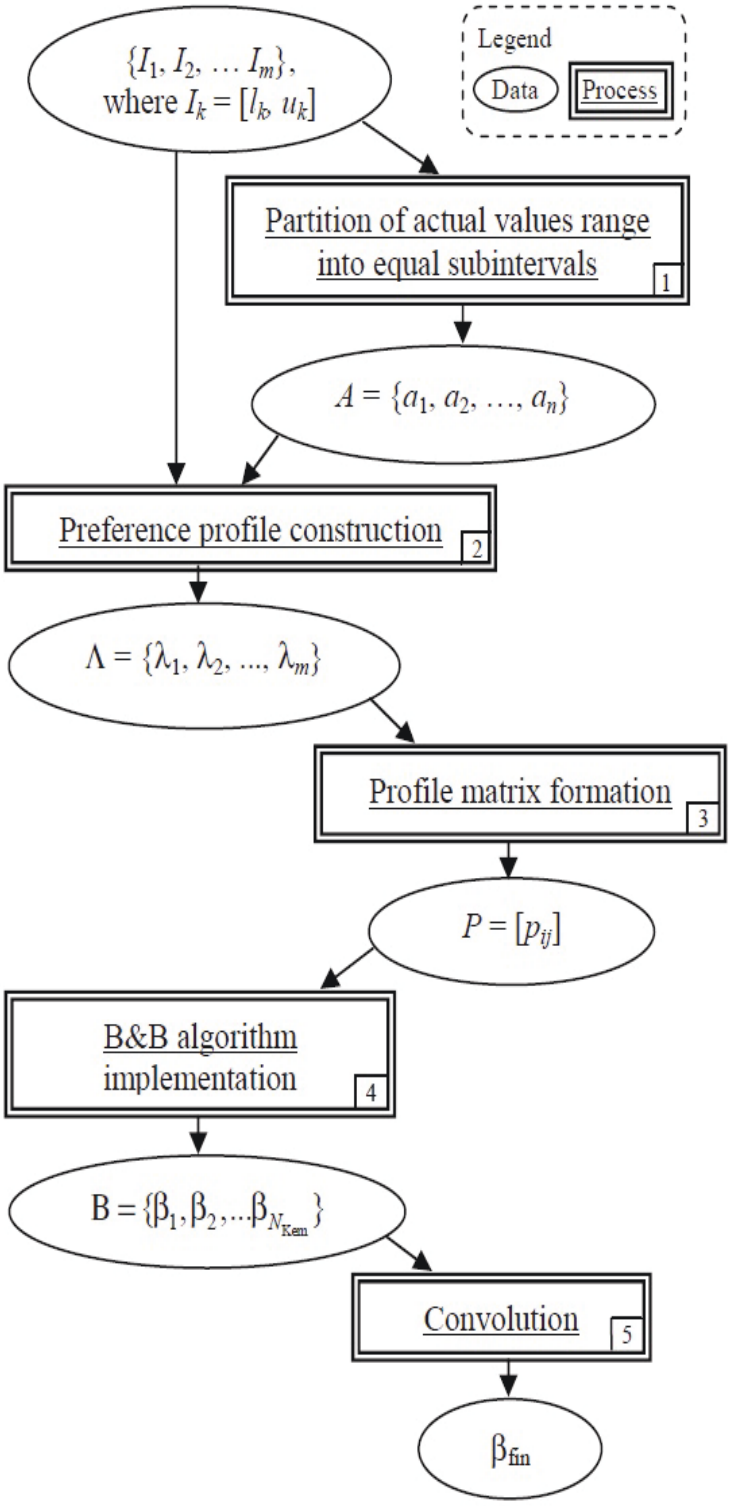


Figure 2.26 : Main stages of the preference aggregation procedure Muravyov and Khudonogova (2015)[16]

level. In result, the experts opinion is to be compromise for the sake of consensus. Additional shortcoming of this consensus measure is the absence of experts opinion weighting.

2.7 Byzantine Fault Tolerance

Reliable systems need to deal with faulty modules which provide contradictory information to various parts of the system. Software errors and malicious attacks are increasingly prevalent and may cause faulty nodes to reveal illogical behaviour. It is recognised that simple majority voting does not answer the problem of gaining interactive consistency when designing fault-tolerant distributed computing systems, especially when it comprises possibly malicious components. When requiring to achieve very high reliability for the distributed computing system, the Byzantine agreement procedures can afford a solution although it appears to be integrally expensive. Byzantine-fault-tolerant algorithms is important because they can permit systems to remain to function correctly even in the existence of software errors. Lamport et al. (1982) introduce the concept of Byzantine Generals Problem and proposed some solutions showing how they can be used to implement reliable systems. They express that a group of the Byzantine army generals camped around an enemy city with their troops. The generals have to agree upon a mutual battle plan by communicating only with messenger. However, one or more generals may be traitors trying to confuse other generals. The challenge is to construct an algorithm which ensures that the loyal ones will reach agreement. They express that if more than two-thirds of the generals are loyal, this problem can be solved otherwise it cannot[97][98] as shown in Figure 2.27. Patnaik and Balaji (1987)[99] objective at revising the concept of 'Byzantine resilient distributed computing systems', the related protocols, and their potential applications. They discuss the interactive consistency problem, the consensus problem, and the generals problem and summarise different

Byzantine generals problem agreement algorithms in terms of level of fault-tolerance and performance. The discussed Byzantine agreement algorithms classes were: the randomised, approximate, and deterministic agreement protocols. Finally, Byzantine agreement protocols application to clock synchronization was illustrated. In Castro and Liskov (1999) the practical Byzantine Fault tolerance was introduced to tolerate Byzantine faults, enhance the response time of earlier procedures and function in asynchronous environments. They applied a Byzantine-fault-tolerant NFS service by using their replication algorithm and measured its performance. The outcomes indicate that the service is only 3% slower than a typical non-replicated NFS. The algorithm is a procedure of state machine replication (as shown in Figure 2.28) which preserves the service state and carry out the service operations. The replicas transfer along a sequence of configurations called views in which one of them is the primary while the others are backups. When it seems that the primary has failed, a view change operation are executed. In approximation, the algorithm works as follows: A client directs a request to appeal a service operation to the primary which multicasts the request to the backups, then replicas perform the request and direct a reply to the client which waits for one replies from various replicas with the same result; which is the result of the operation. Two requirements were imposed on replicas: they must be deterministic and they must begin in the same state. With that, the algorithm guarantees the safety property by ensuring that all non- faulty replicas agree on an over-all order for the implementation of requests despite failures [100][101][102].

Oluwasanmi et al. (2010) propose a novel practical algorithm, based on a previous theoretical outcome showed that it is potential to solve the Byzantine agreement, universe reduction and leader election problems in the full information scheme. The fault model was reduce, to attain the algorithm, by permitting the adversary to administrate only a 1/8 part of the processors; and supposing the presence of a cryp-

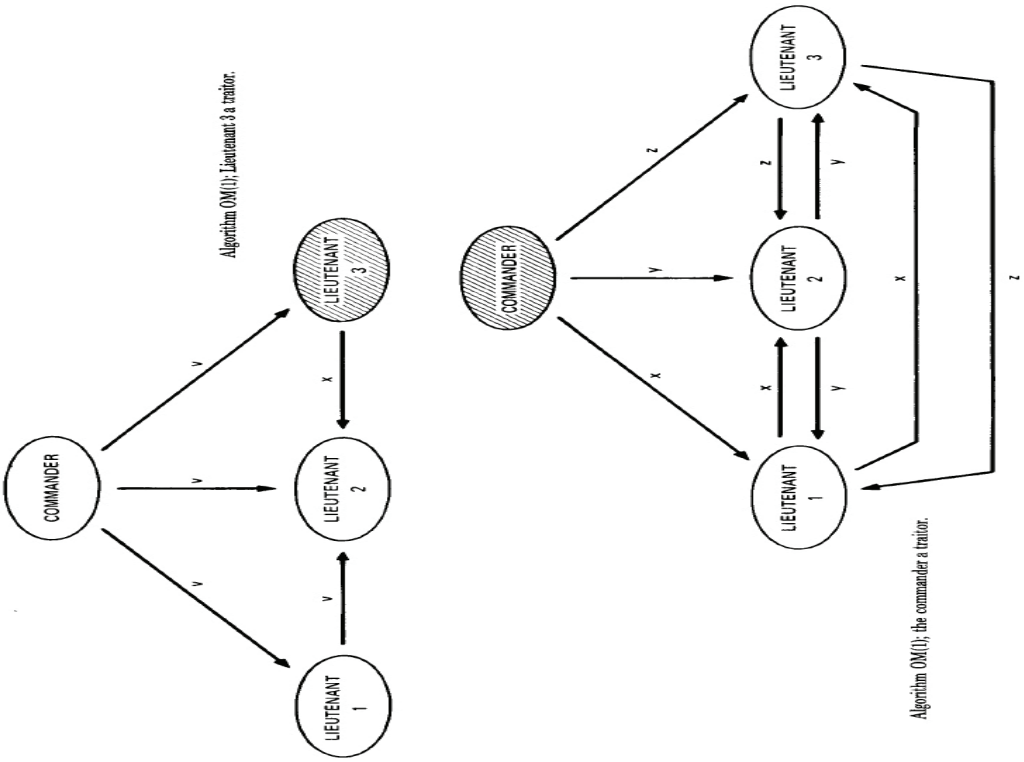


Figure 2.27 : Byzantine Generals Problem Algorithm Lamport et al. (1982)

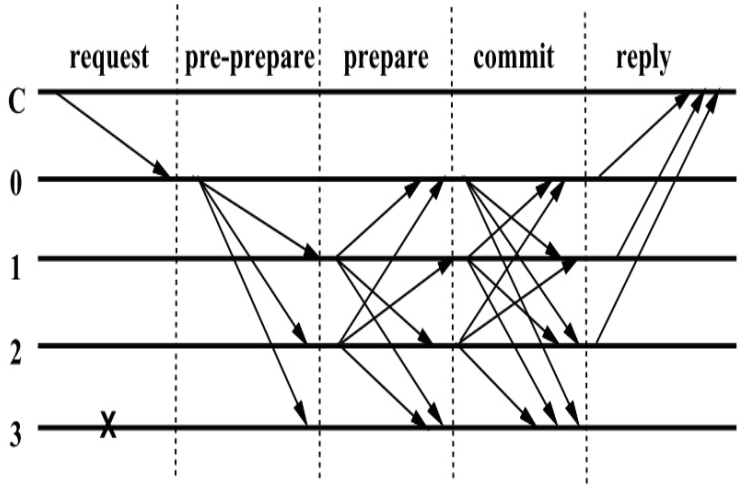


Figure 2.28 : Normal Case Operation of Practical Byzantine Algorithm Castro and Liskov (1999)

tographic bit commitment primitive. The procedure assumes a partly synchronous communication scheme and that the clock speeds of the non-faulty nodes are nearly the same though the clocks do not require to be synchronised. The algorithm needed less overall bits to be transferred for all networks and also less messages sent for large size networks (about 65000 or above). The outcomes propose that the algorithm could be an important step toward evolving Byzantine agreement algorithm for large networks [103][104]. Distler and Cachin (2016) introduce Resource-Efficient Byzantine Fault Tolerance (REBFT), a method which reduces the resource usage of a BFT system throughout normal-case operation approach, where the agreement procedure and implementation of client requirements ordered by the active influence of subgroup models. It will certainly be not required to remodel the system from start in this mode, instead using available protocols like Byzantine fault-tolerant for reaching the process of resource-saving in general can be helpful. In case of suspected or detected faults, the activation of passive replicas has remained reliable methods [105]. Klempous (2006) examine classic perceptions of Byzantine Failure Tolerance that can be used in fault-tolerant system design and propose a Byzantine algorithm that can help to decrease the impact of false alarms or a single disturbance in WSN so local distortion or disturbances issues can be resolved locally without comprising the whole infrastructure. The scheme is to validate suitability of alternatives of Byzantine Algorithms based on supplementary information flow for resolving such glitches locally. The procedure is done as follows: a sensor senses an event, false or true and then directs information to other sensors in a particular communication range. 'The sensor waits for some time T for response from other sensors and collects received information in a table (agreement matrix AM), if there is no response from a sensor it led to that there is no recognition of an event or object. A sensor also has information about the number of sensors in his Agreement Range (Ar) so after executing a majority function it can choose what to do. If the majority of sensors

senses an event the information will be directed to CH and then to the Sink. If not, then no data will be transferred through the network. In the design, an additional requirement was introduced that only the highest energy level sensors can transfer data to CH which reduces the communication load [17]. Chai and Zhao (2014) study how to provide an autonomic system built with the technology of event stream processing and explain how to plan such a scheme that is resilient to malicious attacks and hardware failures. A set of lightweight techniques was proposed which helps attain the event processing of Byzantine fault tolerant for autonomic computing based on a broad threat investigation of event stream processing. The techniques involve voting at the event consumers and a technique of on-demand state synchronization initiated when an event consumer fails to gather a minimum of matching decision messages. reliable ABFT matrix multiplication depends on linear codes that is suitable to P2P computing. The suggested protocol tolerates the potential unpredictable behaviour of the P2P network A mechanism of an evidence-based safe-guarding was also introduced which avoids a faulty event consumer from bringing unnecessary state synchronization rounds. By using the event stream processing scheme for autonomic computing it is likely to use event processing middleware that facilitate autonomic components design and implementation, as in Figure 2.29 as well as decouple the autonomic components, the sensors, and the scheme to be self-managed, which simplify upgrading and maintaining individual components [106]. In consideration of P2P, Fedotova and Veltri (2006) study and indicate the possibility of applying some classical resolutions for Byzantine Generals Problem and its mathematical model in aP2P environment to resolve some security problems based on the principles of Distributed Hash Tables (DHT) since it is very significant to detect reliable mechanisms of finding and eliminating various threats, malicious nodes, and attack sources [18][107], Figure 2.30. However, Roche et al. (2009) introduced a generalization of reliable ABFT matrix multiplication depends on linear codes

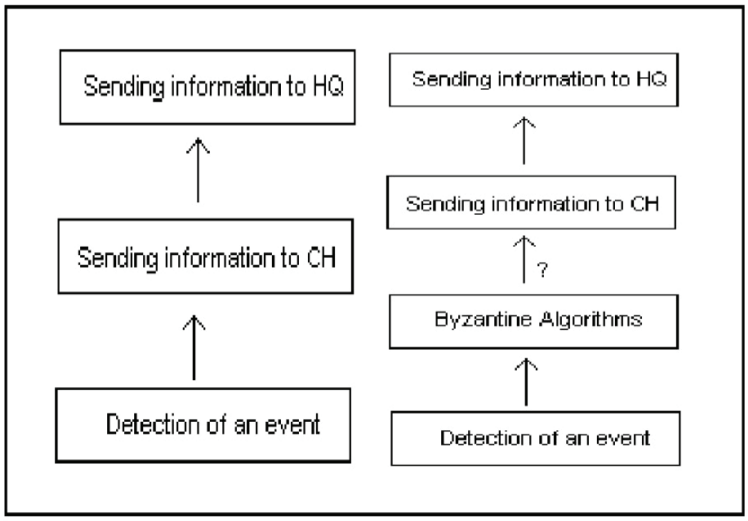


Figure 2.29 : Standard WSN vs. WSN with Byzantine Algorithms Klempous (2006)[17]

that is suitable to P2P computing. The suggested protocol tolerates the potential unpredictable behaviour of the P2P network; instead of limiting to 2D checksums which tolerate only a slight quantity of node failures, a suggestion to set up disk-less checkpointing on linear codes that possibly tolerate a big sum of faults. Then, we analyse and compare Low Density Parity Check (LDPC) usage to classical Reed-Solomon (RS) codes in relation with various fault models to fits P2P systems which offers efficient fault tolerance for peer disconnection as well as Byzantine errors (even involving malicious peers). The LDPC disk-less checkpointing technique is suitable when only node disconnections are examined, but cannot consider Byzantine peers while the RS disk-less checkpointing technique tolerates such byzantine errors, but is limited to precise finite field computations [108][109]. Different systems can be built with Byzantine fault tolerance including Cloud Computing systems and some literatures were made explaining such models. Reiser (2011) presents a position paper to outline the architecture of the CloudFIT project and analyse the impact of utilization in the cloud and explore to what degree the prevailing BFT algorithms

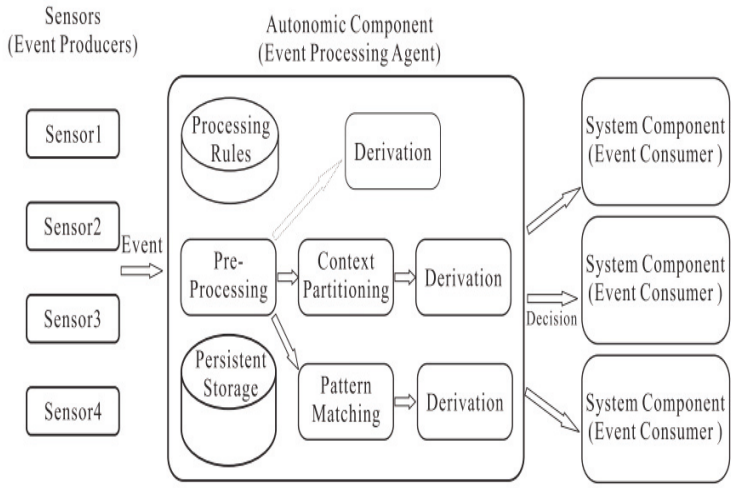


Figure 2.30 : Overview of a event stream processing system for autonomic computing Fedotova and Veltri (2006)[18]

could be executed for augmenting security and availability in the suggested architecture and what concerns need to be determined in the future [110][111]. Whereas AlZain et al. (2013) report for a practical paradigm for system building, comprising Byzantine fault tolerance in a multi-Cloud setting which depends on a new method which integrates Byzantine Agreement protocols with Shamirs sharing method of secret for sensing Byzantine failure within a multi-Cloud setting of computing and also endorsing the security of stored data in the Cloud. In the paradigm of BFT-MCDB, 2f+1 clouds interconnect with an entity of Cloud manager which takes the majority before outcomes are to be retrieved by client from the clouds. The 3 main model components include, cloud manager, the clouds side, and BFT communication protocol as shown in Figure 2.31. First, the cloud manager is in charge of submitting requests to the clouds from the clients and also executing secret sharing approach of Shamir on the trusted information, as well as directing regained outcomes to the client after voting them from the clouds. Second, the BFT of queries is presented to clouds and client by a communication protocol. Third, the side of cloud is in charge

of executing queries for the client on Shamirs before directing replies to the cloud manager. A series of client requests directed by the cloud manager relates to the input of proposed model, whereas the output is a series of the committed replies from the clouds [19][112]. Nevertheless, in the given framework of Zhang et al. (2011) of Byzantine fault tolerance for building optimal system in the environments of voluntary-resource cloud which ensures the robustness of system for conditions like, up to f of overall 3f +1 resource providers are defective, having random behaviour errors, crash errors, etc. The framework chooses voluntary devices depending on the performance of reliability and QoS features which adjust to that vastly dynamic environment representing voluntary resource Cloud Computing. The resources of the faulty voluntary was supposed to substitute with another appropriate resources the moment they were being recognised. Its experimental outcomes illustrates the efficiency of the method to assure the reliability of the system in Cloud Computing environment. Figure 2.32 displays the work procedures of BFTCloud model where the input is a series of queries with indicated QoS requirements (like preferences on price, bandwidth, capability, response latency, workload, failure probability, etc.) directed by the cloud component whereas its output is a series of committed replies corresponding to the queries. The model contains 5 phases explained as:

i- A primary selection: When accepting certain query from an element related to the cloud, a node is chosen as the primary by executing the algorithm of primary selection based on the requirements of QoS of the query.

ii- A Replica Selection: By applying a replica selection algorithm, nodes' set is chosen as replicas based on the requirements of QoS the query. The query is then forwarded by the primary to the entire set of replicas for implementation and the chosen replicas form a BFT group along with the primary.

iii- A Request Execution: The entire BFT group members apply the query locally

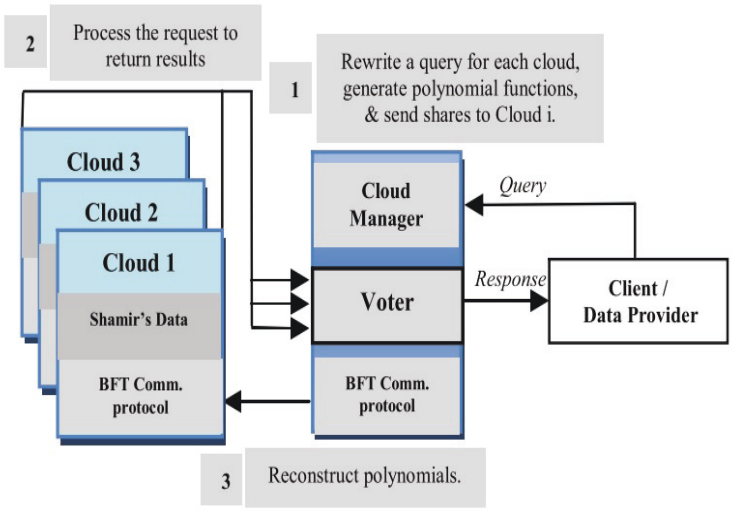


Figure 2.31 : BFT-MCDB Model Overview AlZain et al. (2013)[19]

and propel their replies back to the cloud element. After gathering replies from the group in a specific time period, the component of the Cloud is to evaluate the replies' consistency. when the group consistently replies, the present query is to be committed and the element of the Cloud is to direct the upcoming query but if reply is not consistent, then the procedure of fault tolerance would be triggered tolerating a maximum of f faulty nodes along with triggering the algorithm of primary updating and/or the algorithm of replica updating for updating the members of the group. If there were more faulty nodes than f, the query will be resent to the updated group and go in the phase of query execution once more.

iv- A Primary Updating: If there exist a faulty primary, it will be recognised in the group and substituted by a fresh chosen one.

vi- Replica Updating: Faulty replicas will be recognised in the group and it will be replaced in accordance with the data attained from the phase of request execution by applying the algorithm of replica updating to substitute the faulty one with another proper devices in the Cloud Computing [20][113].

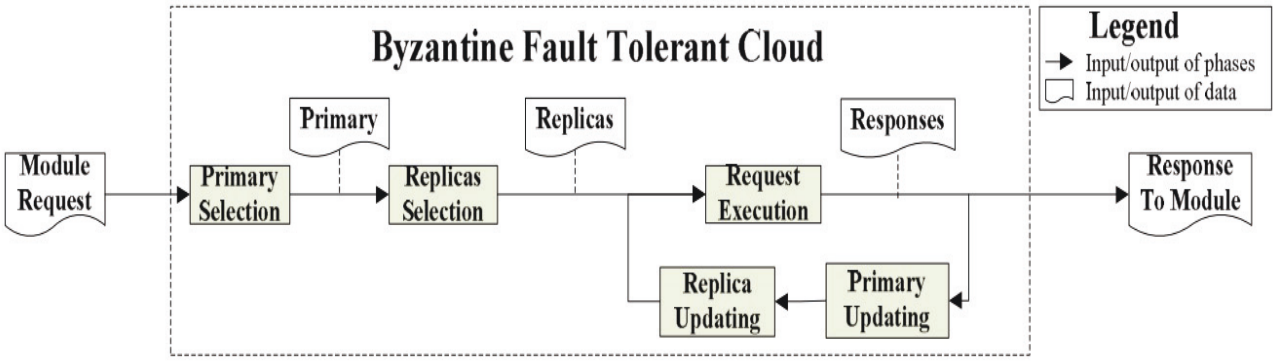


Figure 2.32 : Work Procedures of BFTCloud Zhang et al. (2011)[20]

Autonomic Computing has receive a lot of attention and many researches has been introduced in this promising field. Endo et al. (2011) [114] discuss about concerns to make self-adaptive systems for Autonomic Clouds, concentrating on infrastructure administration level and Buyya et al. (2012) recognise open problems in autonomic resource provisioning and introduce inventive administration methods for assisting SaaS applications located on Clouds and introduce a theoretical design and early outcomes proving the welfares of autonomic administration of Clouds [115]. While Yang et al. (2013) introduce Auto Solar Powered WSN, a new distributed framework to attain sustainable data gathering while improving the performance of end-to-end network for SP-WSNs. They also suggest a routing protocol SP-BCP, two protocols for self-adaptive network, and a rate control structure PEA-DLEX. The framework is an energy-aware provision module that offers consistent energy monitoring and prediction [116]. Nevertheless, Alaya et al. (2012) presented an involuntary system of computing which depend upon M2M standardised architecture and comprised of standard and extensible involuntary administrators and by experimenting a smart metering scenario to demonstrate the suggested solution. The solution differentiates three kinds of M2M machines including the M2M gateway, the M2M server and M2M device and permits for executing the job of administration classified in 3 levels i.e. service, application and communication. A Smart Metering

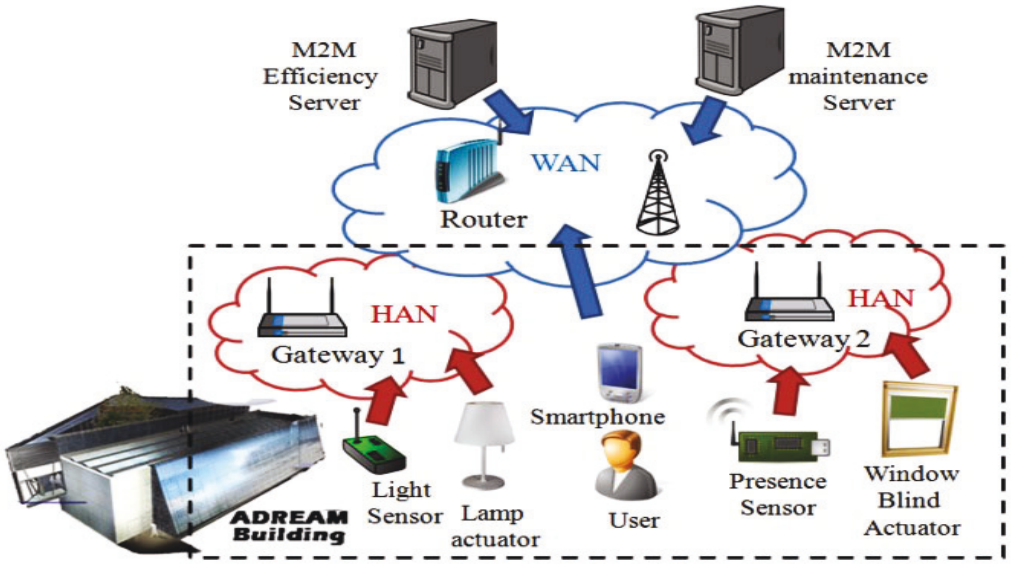


Figure 2.33 : Smart Metering automation scenario architecture Alaya et al. (2012)[21]

use case which can be considered as the autonomic Fog Computing environment that self-manages a room facility, depending on different actuators and sensors, in order to optimise the consumption of the energy. The network composed of M2M efficiency and maintenance servers and other components and devices as shown in Figure 2.33 [21][117].

All published events were collected from sensors by Gateways application managers in a specific format and the relevant ones like lighting glare, possession, status of lamp, and status of blind events were filtered and normalised. Afterwards, the knowledge model is updated by Server Application manager with received normalised events. Then the server incorporates the rules to deduce conditions for instance, lower or higher lighting, occupied or unoccupied space, On/Off lamp and open or close a window blind. Further, it uses specific directions for gathering new change request, for example minimise or maximise luminosity. The knowledge base was then checked by the planner to acquire the actual room status and on spot

behaviours and facts for making decisions and the optimum order of actions or behaviours which were determined to attain the aims for instance, opening and closing of lamp or window blind. The actions like, set-lamp / set-blind (String-Boolean) were sent back to the application managers for gateways to function as per plan, which was checked by executor which verifies the knowledge model to govern the details of web service analogous to each action. The network application manager could be capable of showing an alert memo having errors to the user to bring back the prior status of application layer or building up new action plans by considering recorded errors, all in case of the unsuccessful process of management. The subsequent control loop process occurs on the initialisation of the autonomic framework in the case of successful process of management. The significance of this proposition is that individual autonomic manager is autonomous.

2.8 Data Science

Data science is a systematic procedure aimed to extract insights from data. The usage of data science approaches can enhance the evaluation of many data-based projects to a great extent. They also allow evidence-based decisions to be made which can increase the confidence of the made decision that substantially raise their attractiveness. The objectives of data science are to obtain predictions and classifications based on data and to appoint a model that explains the data. Data science involves statistics, data analysis, machine learning and their related techniques. As shown in figure 2.34, data science procedure starts with collecting data from source nodes and pass it to be analysed to check the relationships between data variables or features. After that, the data is prepared for cleaning and unifying its complex sets through the processes of data wrangling. Following this step, hypothesis is to be presented about the best candidate models to perform the desired task. When choosing a model, data is to be divided into train subset and test subset

then building a model on the train data and predict the output using test data. The procedure's performance is evaluated by calculating the accuracy and/or the classification report to assess how accurate the used model is. With the case when the model does not reach the desired accuracy, another model is to be trained and tested till getting the best model to perform on the data. [118][119][120][121].

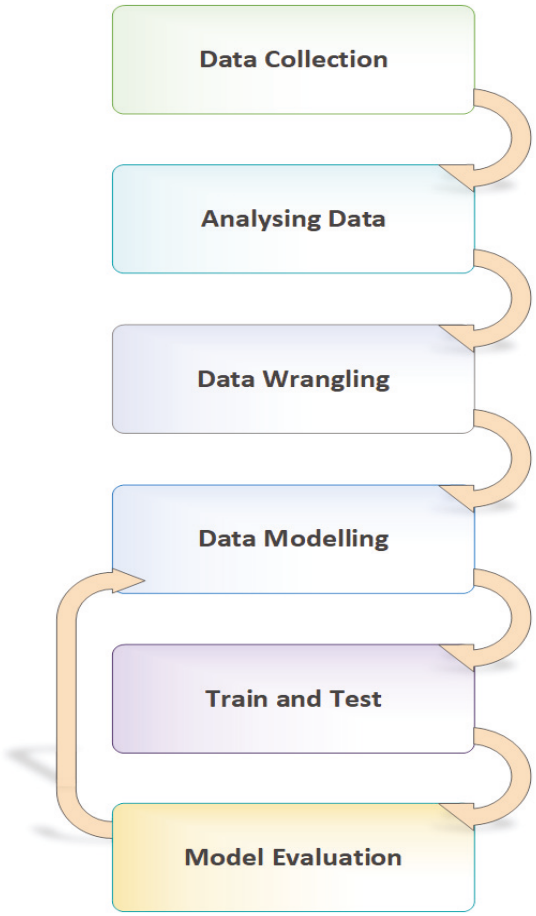


Figure 2.34 : Data Science main Procedure

2.8.1 Data Collection

Data collection is defined as the procedure of gathering, measuring and analysing insights in an endorsed systematic approach, which then enables a researcher to assess hypothesis, answer research questions, and evaluate the results. The aim behind

data collection is to record quality evidence leading to the construction of substantial and reliable explanations to the posed questions based on definite analysis [122][123].

2.8.2 Data Analysis

Data analytics refers to the science of raw data analysis so that conclusions can be drawn on that information. Many of its practices and procedures were automated into algorithms and mechanical processes which work to allow human consumption over raw data. Techniques in data analytics can expose patterns and indicators that would otherwise be lost in the information collection. This information may now be utilised to optimise processes to enhance a system or businesss total efficiency [124][125]. Data Analysis is performed through consistently applying statistical and/or logical methods to characterize, evaluate, clarify, outline, and abstract data. Its objective is to discover beneficial information, reach to a reasonable conclusion and support the process of decision making. There exists a wide range of techniques with many approaches to implement data analysis [126]. There are a lot of methods to analyse data, each dataset is analysed according to its nature and the purpose it is analysed for. Therefore, only some of those techniques will be discussed here as they are used when analysed the data in the presented experiments [127].

Factor analysis

Factor analysis is a useful statistical approach used to investigate variability among observed correlated variables that seeks to find underlying factors from a subsets of unobserved interpretable variables[128].

Data Integration

Data integration comprises linking information exists in different sets and providing a unified vision about them. data integration cover a wide variety of tasks including data mapping which means to create elements of data that associate one

data model to another. It also involves the process of transforming information between source and destination, data lineage analysis method that identifies data relationships, the discovery of sensitive hidden information, data consolidation and data elimination[129].

Feature engineering

In order for the machine learning model to be useful, the dataset has to be refined into features (Prediction variables) before it is fed into the model. Without appropriate features, the model cannot be trained accurately, despite how sophisticated the machine learning procedure is. Feature engineering is the process of utilising the datas domain knowledge to extract features from a raw dataset so as to empower the machine learning algorithm to function[130].

2.8.3 Data Wrangling

Data wrangling is the process for easing the access and modelling of data in a way to clean and unify disordered and complex data sets. It is a method to map and transform raw data from one format to another in order to make it more suitable and worthy for a wide range of downstream considerations including analytics. Some of the main practices of data wrangling include: data cleaning, data encoding, data modelling and data transforming. There are several practices to perform data wrangling, according to its aim and usage. Thus, only some of those practices will be introduced next as they are used in the presented experiments.

Data cleaning

Data cleaning is the procedure of reforming the outliers and other unfitting and undesirable parts of the information. It includes the processes of modifying, replacing, or deleting the dirty or coarse information such as anomaly values [131].

Feature Scaling

Feature scaling is the procedure of normalising a Data range of independent variables or features. This procedure is also known as data normalisation in the context of information processing and it is usually executed during the data pre-processing period [132].

One hot encoding

In order to suit the classification variables to be fitted machine learning operations, One hot encoding procedure is used. The objective from using this procedure is to enable the ML algorithms to ensure a better decision making prediction[133].

Botstrapping

Botstrapping is the process of recurrently drawing and resampling non-repeated large quantity of small samples from a population with replacement for the purpose of generating synthetic data[134].

Data Augmentation

When there is a lack of an adequate size of training data, increasing the effective volume of existing information is possible by the mean of data augmentation process. Data augmentation is a strategy that has the impact to effectively increase the dataset through perturbing a segment of existing data to produce new ones by creating multiple augmented variants of that segment. Data augmentation has contributed to considerably enhance the performance of deep neural networks. It assists the network in becoming more robust through driving it learning relevant features. Data augmentation is also a way to reduce overfitting, where size of training data is increased utilizing information exists in the training data itself[135]. This model can be used to solve the data imbalance issue through increasing the volume of the

small classes in the data.

2.8.4 Data Modelling

Data modelling is the method of building a data model designed for an information system via applying definite recognized practices. In order to manage data modelling as a resource, the practices and approaches are utilised to model data in a typical, predictable, reliable way. Basically, data models are constructed during the phases of a projects analysis and design in order to assure full understanding of a new applications requirements. Additionally, it can be invoked later within the lifecycle of data to justify data schemes that were initially built on an ad hoc basis [136][137][138]. A communal task in machine learning practices is algorithms study and construction which can learn from and make forecasts on data. Through constructing mathematical model from input data, those algorithms work to form data-driven predictions or decisions[139][140].

2.8.5 Train and Test

When creating the final mode, the data used to construct it normally originates from multiple data sets which usually utilised in different phases of model formation. In order to fit the model parameters, the model is to be apply on the training dataset whereas testing dataset is used to evaluate models performance [141][142]. Train data is used for model training and parameter tuning. The purpose behind using it is to learn patterns from the data yielding a model that can make near-expected predictions. Although validation data, which is a subset of train data that insights are iteratively taken from, is used for parameter tuning to understand model behaviour and generalisability on unseen data resulting in insights on how to tune your model. However, the use of test data aimed for understanding how the model would perform in real world scenario providing a complete unbiased estimate of model performance. [143][144][145].

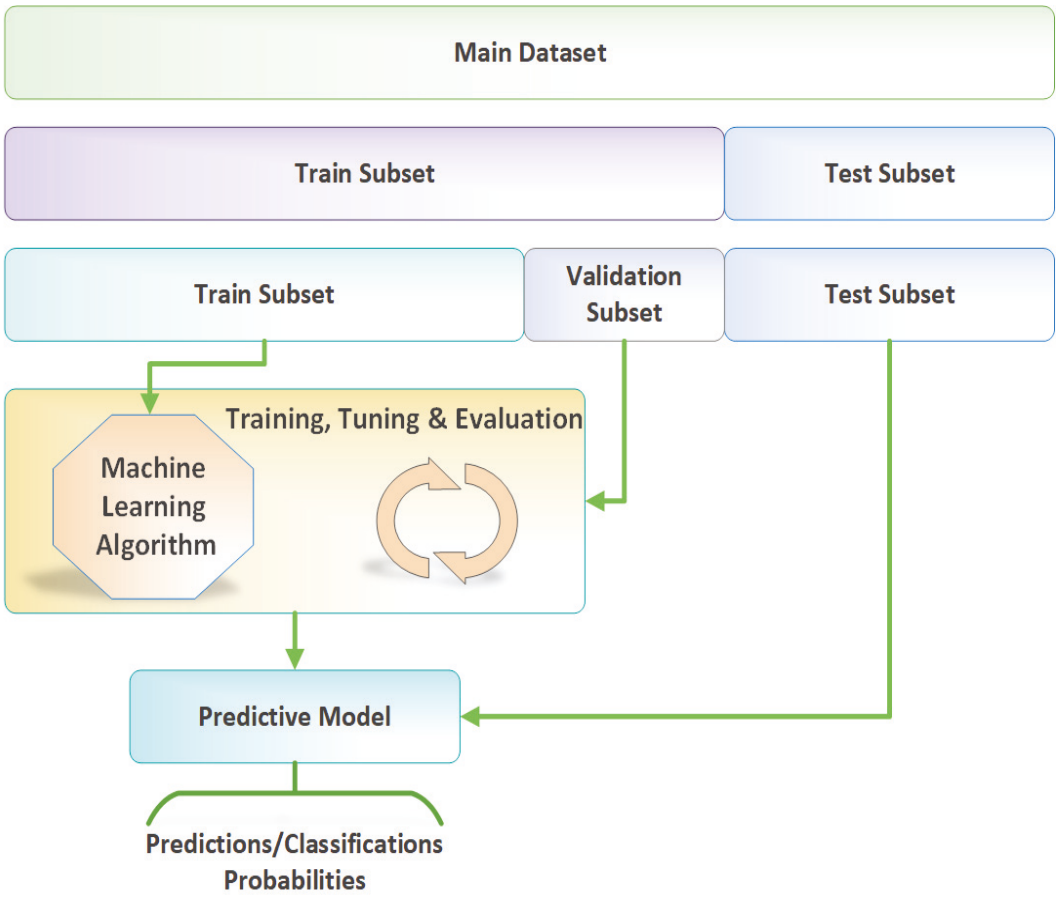


Figure 2.35 : Training validation and test dataset and its contribution in Model prediction

2.8.6 Model Evaluation

Although accuracy is the key metric to evaluate models, there are other metrics which are to be considered such as robustness, flexibility and scalability. Looking at the models full image, like realising data and making predictions, aids in understanding the model in-depth and assists in enhancing it [146][147].

2.9 Machine Learning

Machine learning (ML) is the technical study of procedures and statistical approaches used by computer systems so as to accomplish a particular task efficiently

employing data patterns to infer a specific decisions or outcomes without utilising explicit commands as in programming[139].

2.9.1 Deep Learning

Mainly, Deep Learning practices are presented to solve massive intricate optimization problems. Neural Networks are purely very complicated functions, comprising enormous amount of parameters, representing a mathematical solution to a problem. Through training neural networks, it basically aims to minimise a loss function in which the value of it gives a scale of how far is the performance of the network from optimum on a given data.

Consider a Neural Network (NN) with layers equal to L (having hidden layers equal to L-1 and a single output layer). Layer l weights and biases (its parameters) are denoted as: $W^{[l]}$, which is the (layer l size x layer l-i size) weight matrix, and $b^{[l]}$ represents the (layer l size x 1) bias vector[148] [149] [150]. Moreover, there are intermediate variables that computed during the training process as:

$$Z^{[l]} = W^{[l]} * A^{[l-1]} + b^{[l]} \tag{2.1}$$

$$A^{[l]} = g^{[l]}(Z^{[l]}) \tag{2.2}$$

wherein $Z^{[l]}$ is the linear activations at layer l, $g^{[l]}(.)$ represents the non-linear function, and $A^{[l]}$ is the non-linear activations and $A^{[0]}$ is the input X. In deep learning, to train a neural network, there are five main steps to be followed:

i. Initialise the parameters (weights and biases).

ii. Forward propagation: By the use of every layers input X, weights W and biases b, the linear and non-linear activation functions (Z and A) are to be computed. At the final layer, the activation function $f(A^(L-1))$ is computed which could be a sigmoid, softmax or linear function which would give the prediction y_hat.

iii. The loss function is to be computed: It represents the function of the actual and predicted labels (y and y_hat). It determines the actual difference between the prediction and the actual target. The objective of the whole process is to minimise this loss function.

iv. Backward Propagation: Herein, the gradients of the loss function $f(y, y_hat)$ in respect with the activation function A, weights W, and bias b are calculated (denoted as dA, dW and db). Using these gradients, the values of the parameters are updated from the last layer to the first one.

v. Finally, steps ii-iv are to be repeated for n iterations (called epochs) till reaching a minimised value for the loss function, without overfitting the trained dataset.

LSTM

LSTM is an artificial Recurrent Neural Network (RNN) architecture in deep learning field that can process an entire sequences of data. It has a feedback connections which makes it capable of simulating any computer algorithm logic [151]. The LSTM RNN have the ability to learn features from a sequenced dataset in an automatic way, support Multivariate data analysis, and produce a variable length sequences which might be utilised for multi-step forecasting. An RNN is a neural network wherein the output of a time step is delivered as an input for the subsequent time step. So, it enables the model to predict a decision based on the current time step input as well as a direct knowledge about the prior time step output. LSTM is considered to be successful algorithm in resulting stable models since it overcomes the challenges associated with training the RNN. It exploits the recurrent connection of the prior time step outputs as well as contains an internal memory operating as a local variable, giving it the ability of accumulating state over the input sequence.

Adam optimizer

Adam is considered as an adaptive learning rate optimization procedure that has been developed explicitly for training the DNNs [152]. It calculates distributive learning rates for various factors. It is important to choose an appropriate optimization algorithm for deep learning practice because it is directly related to speed up the learning process. When presenting the procedure, Kingma and Ba (2015) mention the benefits for employing Adam to be used on non-convex optimization issues. They express that it is implemented directly with computationally efficient, minimum memory requirements and typically need little tuning. It also a good fit in dealing with large issues in terms of data and/or parameters or those with very noisy or having sparse gradients[152]. Empirical outcomes from a lot of researches showed that Adam functions well in practice and compares favourably to other stochastic optimization algorithms. In the original paper [152], Adam was validated empirically to demonstrate that convergence meets the expected theoretical analysis. Being applied to the logistic regression algorithm, Adam was utilised on a Multilayer Perceptron procedure and Convolutional Neural Networks on an image recognition dataset. The conclusion was that Adam can efficiently mitigate practical deep learning issues when utilizing large systems and datasets. Roder(2016) introduce a thorough review abut recent gradient descent optimization procedures [153]. He mentioned that Adam marginally outperforms RMSprop when the optimization nearly approaches the end as gradients become thinner due to its bias-correction. He concluded that Adam could be the optimum overall decision. Karpathy et al. (2016) suggested to use Adam algorithm as a default optimization procedure for deep learning [154]. Additionally, Adam is being revised as a benchmarks in deep learning techniques. While Xu et al. (2015) used Adam on attention in image captioning[155], Gregor et al.(2016) utilised it on image generation [156].

Random Initialization for Neural Networks

The task of building a neural network can be confusing, upon that modifying the neural network so as to gain better outcome is very aggravating. The first step to consider when building a neural network is parameters initialisation which is a crucial practice for achieving optimisation in least period of time if done correctly[157].

In every neural network structure, there are weights between any two layers. The weights liner transformation of the current layer together with the rates of the former layers pass through a non-linear activation function in order to generate the weights for the next layer. This procedure takes place layer to layer throughout the forward propagation. As the back propagation operation occurs, the optimum rates of these weights could be detected so that it can generate accurate values given the input[158].

Thus far, randomly initialized weights have been utilised as an initialisation point for machine learning operations. Initial values of these weights play a significant role in concluding the cost functions global minimum of a deep neural network. There are many methods used for weights initialisation between the layers such as zero initialization, random initialization and He initialization[159].

Chapter 3

Strategy, Methodology and Computation Models

3.1 Introduction

We started our methodology with a question; is there a relationship between data organization and Decision Making techniques (consensus) when managing data within Fog network?. This question was derived from the main research question; is it possible to smartly organise Fog network data using consensus data organization? We had investigated that relationship and tried to perform smart data organisation and autonomous decision making as well as distributed computation among Fog network nodes. This investigation had led us to realise that the research has mixed qualitative and quantitative methods. As such, the decision is to be made based on the network status (qualitative), whereas it is authenticated by measuring the accuracy of that decision (quantitative). Our method focuses on measuring monitoring data in built environments where network nods are the entrants of the decision making process. A core practice in a decision making processes and data management determination is data analysis. Analysing sensor data is crucial in examining the patterns of each network status. Machine learning is one of the most effective methods in analysing data and performing autonomous decisions.

To achieve our objectives, we introduce consensus based data management and decision making set of algorithms which work together to perform data distribution and decision aggregation as well as carry out distributed computing within Fog nodes. The outcome of this would be the data distribution illustrating the status of data and a decision reporting the activities taking place within the network. Data

management techniques can take many forms as explained in chapter 2. From those, there was no calibration phase as explained in this research that investigate the status of each front-end device and data range in the normal operation mode in order to set up a data distribution. Also, they did not combine the decision aggregation together with data aggregation within their approaches. Data management takes two forms in the proposed model, one is by composing data distributions with reference to collected data and the other is through assembling decisions about activities that occur in the network then forwarding both of them to the ascendant node. Data distributions are deliberated by calculating the distribution parameters (Mean and the Standard Deviation (std)). For decision aggregation, Decision Making techniques such as statistical and Machine Learning methods (accompanied with the consensus methods explained later) to extract a decision about activities within the network. After the aggregation process, the node sends one packet of data (at most times) to the ascendant node containing data distribution parameters and the decision.

3.2 System Topology and Operational Platform

The proposed system topology and operational platform consists of three stages that would integrated together to address the distributed computation and high data quality through data analysis. IoT data needs to be analysed, due to its heterogeneity, and its patterns needs to be investigated from an early stage i.e. it requires a prediction which has three fundamental traits. The first is that a continuous quantity is required to be predicted (IoT data is continuously generated and transmitted). The second one is, predictions are made under uncertainty that's why a brief inspection is required to see how uncertain the predictions are. Third, that those predictions are contingent to some observation. Basically the process is not about obtaining a single answer, but in effect a range of answers to estimate each individual answer's probability. In summary: there exist a demand for a probability

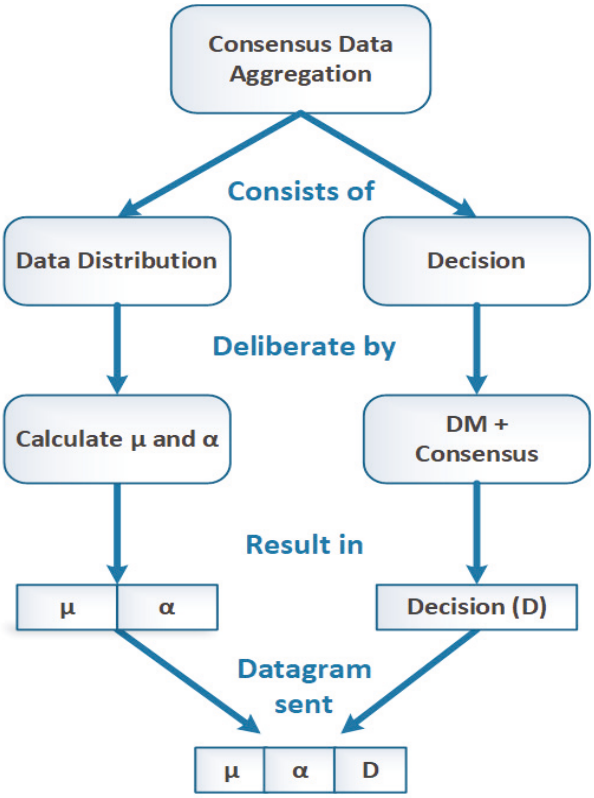


Figure 3.1 : Node Process diagram

distribution across a domain of answers given network data input so that to enhance the prediction process and may even perform a better-informed decisions for network administrator or even a user [139].

This Prediction process requires data analytics at network edge. Edge analytics is a method for information ingathering and evaluation wherein an automatic analytical computation is executed on records at a sensor, network switch or any other node rather than awaiting the data to be brought back to a centralised information store. The plan to analyse data include Bayesian analytics as the probability approach to perform predictions and construct network data distributions.

The topology and operational platform of the proposed system is shown in Fig. 3.2. The first level of it, Front-end Level (L0), is composed of front-end nodes (e.g. sensors), while the next level, Fog-decision Level (L1), consists of network nodes

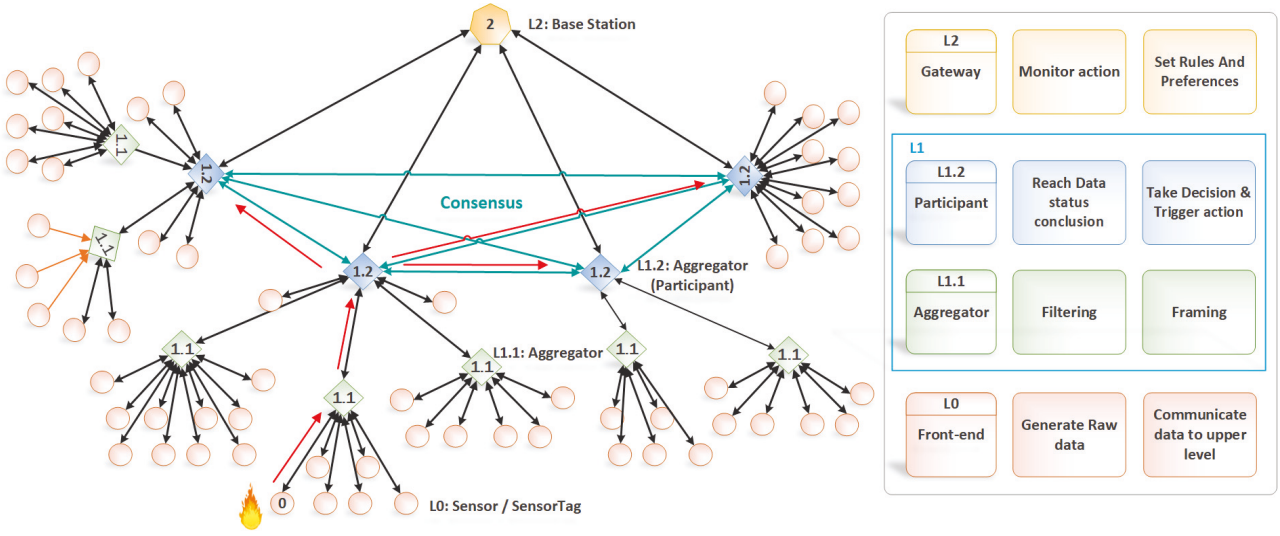


Figure 3.2 : IoT Fog Topology and Operational Framework

that has the ability to aggregate data (Aggregators) representing cluster heads of the subnetworks following them. This level can be made up of multi-sublayers based on the network topology with first level of Aggregators attached to the front-end nodes (as in L1.1 nodes in the figure) and top level aggregators represent the *"Participants"* of the consensus operation (L1.2 in the figure).If the network topology does not require multi-sublayers i.e. there is one aggregator node representing the Fog-decision level, the aggregator node would have a *"Participant"* entity to perform its operations. The last level (L2) points out the back end of the network or the Base Station (BS) which acts like a gateway that connects the network with the rest of the IoT world.

The main responsibilities and functions for each level of the operational platform, which is also represented by four levels, is demonstrated as follows:

L0:

In this stage, the devices have a low level access to the physical environment or phenomena and they generate the raw data, check its boundaries (set by Bayesian

Analytics representing a distribution suitable for the data), and communicate data to the upper level.

L1:

The purpose of this stage is to reach a fine detailed conclusion about data status, to make decision about the occurrence of any event or activity and to trigger the action required (as will be explained in the next section). As mentioned above, this stage can contain one or multiple sub-levels. The unimportant data is to be filtered out, according to Bayesian analytics, in the early sublevels by the Aggregators and then the rest of data is to be encapsulated in an abstract frame (data distribution) and forwarded to next sub-level. While in the last sub-level, the required actions depend on the communication between the responsible *Participant* node and its peers in a way that it would provide a reliable decision not only based on its local experience rather than based on the peer-participants as well. Bayesian analytics plays a core role in every step of the process.

L2:

Monitoring the action of the system, set the rules and preferences when required, and acts as a network Gateway.

As the methodology of our proposed system is based on Bayesian Analytics, Each *Participant* node would have a Bayesian Classifier that analyses the mode of operation for the system or parts of the system. The method starts with calibrating each sensor of L0 under close monitoring for each mode of operation so that its boundaries can be measured. Once that has been done, a distribution for each mode of each sensor is to be saved in the corresponding *Participant* node besides saving the normal mode distribution in the corresponding L1 node. After the calibration phase is finished, the system operates under the utilization of Consensus Management Procedure explained in section 4.

3.3 Operation Phases

The Operational framework of the suggested methodology lays in four phases. The first two are to adjust the network intity and the others are to be performed within the normal and abnormal modes.

3.3.1 Pre-Exploitation Modes

Before fully deploying the algorithm into the network and letting it operates independently, network components need to be set in order to handle the operation of the procedure within the network.

Sensor Calibration Phase

When monitoring an environment, the status of the atmosphere changes according to climate, objects movements, atmospheric size or intervention of external factors. The environment measures (temperature, humidity, light, movement, gyroscope ... etc.) are different between day and night time and among seasons while the available open space in any sphere affects their scope. Also, the movement of objects within the environment is the cause of changing the activities in it. However, abnormal changes in the environmental readings can take place according to the occurrence of an event like rapid temperature rise, fire ignition or flood emergence.

In order to detect the happening of any event or the change in any activity within the environment, we first need to measure the boundaries of the normal environmental mode and the basic activity patterns within it. In this case, any reading goes beyond them, it considered as an abnormal reading. We also need to define some abnormal modes and measure their boundaries so that the system can detect the mode and decide accordingly. This emphasises the importance of calibrating the readings of each sensor in each mode. It works as training for sensors such that each one can set the threshold boundaries for itself in each mode. The calibration

phase involves continuously and closely monitoring sensors measurements in each mode for a sufficient period of time so that we can set the range of values for each sensor (or a certain patterns) within that particular mode. The readings are then used to construct a mode probability distribution which will be used as a reference to that mode. Each distribution has its main parameters (mean μ and std σ) that the mode threshold boundaries will be set accordingly. As we know from statistics analysis that for Normal Probability distribution, for example, approximately 99.99 % of the data readings lie within $\mu \pm 5\sigma$ boundaries, we can set the first threshold value (th1) to $\mu \pm 6\sigma$ and the second (th2) to $\mu \pm 8\sigma$. The Normal Mode Distribution is to be stored in each sensor's memory as well as at the correspondent L1 Aggregator which constructs a general distribution for its network based on sensors distributions. Other Modes are to be stored at *Participant* based on the trained data obtained from the sensors at that mode and/or data gathered from an outside project or site that describes the operation of that mode like datasets obtained from fire or flood event or any recognised activity within the network.

Aggregator Training

The aggregators' classifiers are to be trained to recognise different operation modes and various activities within the network workspace. During the time of calibrating sensors ranges under a supervised monitoring, the aggregators of the network is to be trained as well. Data to be fed to the aggregator has to be annotated with the modes of operation and/ or the activity happening within the network by the network administrator. By having these annotations, the aggregator classifier entity can be trained to relate data to the mode or activity. The mean and variance of the data distribution will be constantly updated as more readings are received from sensors. In the same time, the aggregator Classifier is trained and updated using the distributions parameters and the annotated data.

3.3.2 Exploitation Modes

When the network settlement is done, the system starts its independent operation. During operation mode, a sequence of processes would be performed as illustrated in Fig.3.3. These processes will be explained from a different perspective by dividing the operation into two phases as in the next sub sections.

Environment Monitoring Phase (Normal Operation Mode)

The Environment is to be continuously monitored by sensors and as long as the readings are within the boundaries of the Normal Mode Distribution (NMD), the measures will be sent to the upper levels to train the aggregators nodes considering the management strategy as follows:

At Sensors Level (L0):

Data to be sensed and periodically forwarded to the upper level every certain time period designated as Normal Cycle Length (NCL) which is to be set according to the particular network demands. During this mode, each sensor forms a distribution of its readings each NCL and calculates its mean and variance as well as setting the thresholds as explained in calibration phase section. If the reading is beyond the set boundaries (thresholds), a Flag on will be passed indicating an abnormal mode of operation, otherwise a Flag off is to be moved onward.

At First Aggregator level (L1):

The L1 aggregator receives sensed data from its descending sensors each NCL time period, analyses and combines them according to their types (temperature, pressure, humidity, movement, etc.) and checks if there is any change in the activity. After that it abstracts a distribution representing the data within that period and predicts a decision about the current activity then sends the distribution mean and std along with the decision to the *Participant* node or entity. If there exist a change

in the activity within the network, the *Participant* is to send the last period's data to its peer *Participant*s requesting a consensus procedure to make a prediction about the current activity. The parameters are then used to calculate the *Participant*'s cluster Distribution, of which its parameters and the final decision are to be sent to the BS.

At Base Station (L2):

If the network is at normal mode, the BS receives a number of packets corresponding to the number of participants in the network each specified time (set according to network requirements or a request).

A huge savings in packets generated and transformed within the network as it requires only one packet to be forwarded from each level to its ascendant level in the normal mode of operation.

Event Handling Phase (Abnormal Operation Mode)

The network goes into this phase of operation when there will be sensor readings outside the boundaries of the normal mode indicating the occurrence of an event. In this case the network needs to automate its response rapidly and frequently in a much shorter cycle period called the Abnormal Cycle Length (ACL) that is present according to network specifications.

At L0:

In case the abnormal value is greater than th1 but less than th2, the sensor waits for another value in the same range within the same NCL. If a second value occurs within the range or the reading goes beyond th2 from the first time, the sensor changes NCL to ACL and starts sending the data to L1 switching the Flag in the packet to "On" state.

At L1:

When L1 aggregator receives sensed data with an "On" Flag, it requests immediate data from all other sensors within its network. Once it has received the responses, it checks whether the data lies within the normal mode boundaries related to its general Normal Mode distribution or not. In case of normal state, the Aggregator continues to work as in the normal phase, otherwise, it starts to forward the sensed data with a notification of Abnormal readings to the ascendant level. This prcedure is to be repeated at each ascendant level aggregator if there is multiple layers in the Fog-desicion Level until the data reaches the *Participant* aggregator node. In case of only one node representing L1, the aggregator forwards the sensed data with a notification of Abnormal readings to the *Participant* entity within the aggregator. If the *Participant* node or entity finds out that there is an abnormal readings, it starts Mode Recognition Process using Bayesian Analysis. This process involves matching the abnormal readings with all modes distributions stored in it using the Bayesian Classifier formula to obtain the probabilities of all modes of operations. At the same time, it pushes the abnormal data with a request for consensus to its fellow *Participant*s. The *Participant*s would check the data related to their own classifiers that were trained by separate data sets from every Peer-*Participant*'s particular network cluster. Each Peer-*Participant* would send back its own calculated probabilities for each mode. Finally the *Participant* would stratify the Consensus Process (like the Maximum A Posterior (MAP) estimation or ensemble learning or any other consensus process) in order to estimate consequent mode which depends on the all *Participant*s' decisions (consensus). The mode which will have the maximum probability is to be chosen to represent the current mode of operation and the network will act accordingly as in the network policy.

At L2: The decisions and data are to be sent to the BS for storage or any further check.

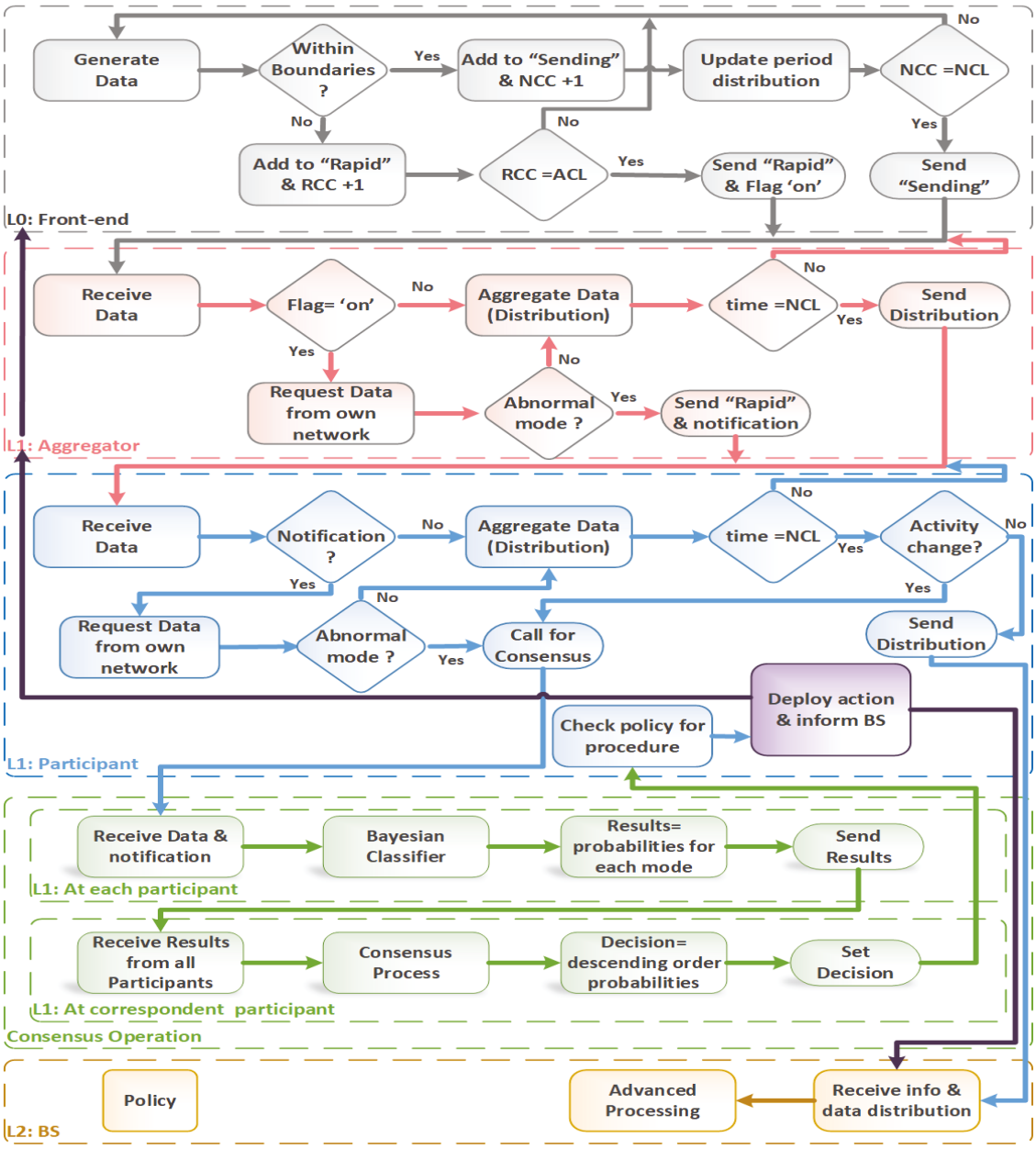


Figure 3.3 : Process diagram

3.4 Consensus Management Procedure

A consensus is general agreement among a group of participants to reach a solid decision about area of interest. However, our consensus model is built by bringing those techniques to decide about a network status or activity in real time learning from their previous own network data and updating at each certain period. The purpose is to improve the precision and efficiency of the system and to reach automation within the Fog Computing platform. Hence, combining the decisions of individual classifiers (*Participants*) may not indeed surpass the performance of the best *Participant*, but it definitely limits the overall risk of deciding a specific poor choice. Consensus algorithms are principally used to enhance the overall performance of a model (prediction, classification, function approximation, and many others), or minimise the probability of an individual model arriving at an unfortunate decision. Other applications of consensus algorithms consist of setting confidence to the choice made through the model, data fusion, nonstationary learning, deciding on optimum (or semi optimum) features, correcting errors, and incremental learning.

The ensemble based systems could be beneficial while handling large amount of information or lack of sufficient information. When the training information volume is very large and its hard for a single classifier to preform training, the information may be strategically subdivided into smaller subsections. Each subdivision could then be utilised to train a single classifier that could then be joined utilizing a proper management rule. If, alternatively, there is very little information, then bootstrapping may be used to train various classifiers using multiple data bootstrap samples, wherein every bootstrap sample is an arbitrary data sample drawn with a substitute and considered as if it was individualistically drawn from the original distribution.

Specific issues are sometimes hard to be solved by a given classifier. In reality,

the choice boundary that isolates information from distinctive classes may be very complex, or lie exterior to the space of capacities that can be performed by the selected classifier model. In logic, the classification framework follows a divide-and-conquer method by partitioning the information space into reduced and easier-to-learn segments, where each classifier learns only one of the simpler segments. The basic complex choice boundary can at that point be approximated by a suitable combination of various classifiers.

The consensus framework normally permits allocating a confidence to the choice made by such a framework. Reflect the case of having a consensus operation among some classifiers trained on a classification issue. When a large majority of the classifiers agree with their choices, such a result can be translated as though the consensus operation has a high level of confidence in its choice. In case, however, half the classifiers make one choice and the other half make a diverse choice, this could be translated as the consensus having a low level of confidence in its choice.

In our model, consensus plays a crucial part in reaching the final decision about the network mode or activities that lie within it. The consensus computation starts after each participant make a decision. However, the procedure starts long before that. When an out of boundaries sensor reading is detected, it is an indication of a change in the network mode or activities. From then, the series of steps described earlier would take place leading the model to make decisions by each participant and setting them ready to gain the consensus decision.

The system would start operating using the Consensus Management set of algorithms once the calibration and training phases finish. To achieve the methodology goal, we will explain in detail how to apply the consensus Management model to the system. The procedure to be performed within the network has four operational segments executing distributed computation. The first operational part would be

in front-end level where sensors pass their readings along with a flag to the corresponding aggregator at each certain time period. In the second stage of operation the aggregator receives the data and checks the flag, if an "on" flag status is detected (i.e. readings beyond thresholds), more data from other sensors will be requested and the sample rate will be heighten as well in order to check the network for an abnormal mode or activity change. If so, the readings with notification will be transferred to the *Participant* node or entity (whether there is a sublevel within the Egde-decision layer or not). For "off" flag situation (data within boundaries), the mean and variance (distribution parameters) will be calculated and moved forward. At the *Participant* node or entity, a comprehensive distribution is to be collected by combining distribution parameters then sending it to the Base station which will significantly enhance the quality of data delivery process. The third phase, operates in the *Participant* node or entity of Fog-decision level, a decision about the current mode of operation or current activities happening will be made by analysing data, appended in the same datagram as the data distribution and then sending it to the base station. Data analytics would be performed applying Machine Learning methods to train Bayesian Classifier in the *Participant* (during learning phase) exploiting all the data derived to it using Bayesian Classification rule since each sensor readings are independent from other sensors' readings and patterns. This would qualify the classifier to segregate different modes. So, when an event occurs in the network detecting a change in the patterns, the classifier would detect the change, initiate the consensus process and survey the current mode of operation. When a consensus decision is made, the corresponding *Participant* would check the policy to act accordingly. The last operational segment would be to deploy a new strategy on the network as exists in the policy procedure. Also, after each time period, set by system administrator, the classifier would be updated by training the measured data and the decisions made as result from the consensus operation within that period.

The consensus routine includes the operation of checking the most suitable mode or activity for the current network status depending on the readings, the process of negotiation, reaching consensus procedure and finally updating preferences based on historical experience of each *Participant*. Many Events can affect the network and the system planner/administrator have to introduce each type of event and activity describing the typical data distribution of each mode and activity within that event type and train the *Participants* to identify them.

3.4.1 Consensus Models

Various techniques are used to reach an agreement about an aim. To represent consensus, there are many techniques that can be utilised each of which has a different way of performing. Some of them are simple to compute like the algebraic combiners (minimum, maximum, sum, mean, product, median, etc.) or the voting based methods (majority voting or weighted majority voting or the proposed Likelihood Multiplication). There are also more sophisticated techniques. These incorporate Dempster-Schafer rule which calculates the plausibility based belief measures for each class; Borda count which considers the class support rankings; "decision templates" (Kuncheva 2001) which computes a resemblance degree between the present choice profile of the unidentified instance and the mean choice profiles of instances from every class; and behaviour knowledge space(Huang 1993) which employs a lookup table that records the foremost common accurate classes for each conceivable class combinations provided by the classifiers. However, there are many techniques that utilise machine or deep learning to perform consensus, called ensemble based systems. Ensemble learning is the technique through which numerous models, including classifiers or experts, are strategically generated and mixed to resolve a precise computational intelligence issue. An ensemble-based system is acquired through combining various models. Thus, such systems are also called mul-

tiple classifier systems, or simply ensemble systems. There are numerous situations in which utilizing a consensus based system makes statistical sense. In each case, the last selection is made through aggregating the individual choices of multiple expert participants. In doing so, the essential purpose is to limit the unfortunate choice of an unnecessary process or an insufficient learned node.

Commonly used ensemble learning algorithms including Bagging, Boosting, Stacking, and Bucket of models. Bagging, involves the use of bootstrapping to get replicas of the training data to obtain a variety of classifiers. To reinforce the variance of the model, every model in the ensemble is trained employing an arbitrarily drawn subset of the training set. A different same-type classifier is trained by each of these subsets. After that, the individual classifiers decisions are aggregated using a simple majority vote mechanism. At each instance, the ensemble decision represents the class selected by most classifiers. As the training datasets may overlap considerably, extra actions can be utilised to escalate diversity, like training each classifier utilizing a different subset of the training data. An example of Bagging is the random forest algorithm which combines bagging with random decision trees to attain a very accurate classification (Breiman 1996). Boosting generates classifiers ensemble by means of information resampling as well, then they are to be combined by a majority voting method. Boosting encompasses incrementally constructing an ensemble via passing every new model instance through training to give emphasis to the training instances which the former models mis-classified. Boosting has been revealed to gain superior precision than bagging at particular cases, though it inclines to over-fit the training information as well. Considerably, Adaboost is the most famous operation of boosting, even though some innovative procedures are conveyed to attain enhanced outcomes. In Stacking, bootstrapped samples of the training part of dataset is primarily utilised to train the classifiers ensemble, producing the first tier of classifiers. The next step would be to take their outcomes and train

a second classifiers tier (Wolpert 1992). The principal behind this is to test if the trained dataset has been appropriately learned. Stacking includes the training of a learning system in order to combine some other learning procedures predictions. It starts with training all the individual procedures employing the existing dataset, after that a combiner process is trained to produce a concluding decision using the predictions of all the individual procedures as an input. Generally, logistic regression model is used as a combiner. Stacking normally results in better performance than any of the individual trained models. The ensemble method of a bucket of models is an algorithm selection process which is used to pick the best trained algorithm for each case. Cross-validation Cross-Validation Selection is the most popularly used method for model-selection and Gating is a generalization of that method. It includes training additional learning methods to elect which of them is best-suited to resolve the issue. The gating model normally utilises perceptron to choose the best method, or it may be utilised to provide a linear weight to the predictions from every method in the bucket.

Bayesian parameter averaging (BPA) is a consensus method that aims to estimate the Bayes optimum classifier by sampling assumptions from the assumption space, and aggregating them utilizing Bayes' law. While Bayesian model combination (BMC) is an algorithmic modification to BPA. As a replacement of sampling individual models, it samples from the space of possible consensus models. This correction beats the inclination of BMA to converge toward offering the whole weight to only one model.

Likelihood Multiplication Consensus Models

This model is a proposed algorithm to achieve consensus. It is applied to the first experiment and compared with other methods. To model the Consensus operation within the network and to refer to Bayesian classifier, let $Y = \{y_1, y_2, \ldots, y_n\}$ be a

finite set of alternatives (classes), ($n \geq 2$), to be classified for network status (like Normal weather, Hot or Fire), and let the set of participants be $S = \{s_1, s_2, \ldots, s_m\}$, ($m \geq 2$) whose weight vector is marked as $G = \{g_1, g_2, \ldots, g_m\}^T$, where $g_k > 0$, $k = 1, 2, \ldots, m$, and $\sum_{k=1}^{m} g_k = 1$. Taking $X = \{x_1, x_2, \ldots, x_d\}$ as a set of features from the training data (which is the classifier's input data), Each participant $s_k \in S$ presenting its probabilities in relation to y_i ,where $i = 1, \ldots, n$, as $P^k(y_i|X)$ which is a finite set of probabilities for participant s_k and it's defined as:

$$P^k(Y|X) = \{P^k(y_1|X), P^k(y_2|X), \ldots, P^k(y_n|X)\} \tag{3.1}$$

where

$$\sum_{i=1}^{n} P^k(y_i|X) = 1 \tag{3.2}$$

After applying Bayesian analysis and obtaining the set of probabilities for each participant about all alternatives, the outcome of the Consensus process is calculated by first aggregating the corresponding probabilities of the same alternative from all participants with the weighted geometric mean according to the following equation [160]:

$$P(y_i|X) = \prod_{k=1}^{m} P^k(y_i|X)^{g_k} \tag{3.3}$$

Then collecting the overall probabilities of all alternatives provided by the participants $P^T(Y|X)$ as:

$$P^T(Y|X) = \{\prod_{k=1}^{m} P^k(y_1|X)^{g_k}, \ldots, \prod_{k=1}^{m} P^k(y_n|X)^{g_k}\} \tag{3.4}$$

At this point, the analyses of Bayes Classifier should be considered. Bayes approaches are a set of supervised learning algorithms established by applying Bayes theorem. Given y (class variable) and x_1 through x_n (dependent feature vector), Bayes theorem expresses the relationship:

$$P(y|x_1, ..., x_n) = \frac{P(y)P(x_1, ..., x_n|y)}{P(x_1, ..., x_n)}, \tag{3.5}$$

The inference is assumed to be independence, the observations' probability density given the parameters, in which it is factored over states in the training set (because of the independence assumption)[161] where

$$P(x_i|y, x_1, ..., x_i{-}_1, x_i{+}_1, ..., x_n) = P(x_i|y), \tag{3.6}$$

so for all i, the relationship is simplified as following

$$P(y|x_1, ..., x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1, ..., x_n)}, \tag{3.7}$$

Since given the input, P$(x_1, ..., x_n)$ is constant , the following classification rule can be used:

$$P(y|x_1, ..., x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y), \tag{3.8}$$

$$\hat{y} = arg\frac{max}{y} P(y)\prod \binom{n}{i=1} P(x_i|y) \tag{3.9}$$

and the Maximum A Posteriori (MAP) estimation can be used to estimate P(y) and P$(x_i|y)$; then the former is the training set's relative frequency of class y.

Bayes classifiers differ generally according to their assumptions regarding the probability distribution P$(x_i|y)$.

In order to continue the derivation of the Likelihood Multiplication model, equation 3.4 is to be substituted with (3.8) to have:

$$P^T(Y|X) \propto \{\prod_{k=1}^{m} P^k(y_1)\prod_{i=1}^{d} P^k(x_i|y_1)^{g_k}, \ldots, \tag{3.10}$$

$$\prod_{k=1}^{m} P^k(y_n)\prod_{i=1}^{d} P^k(x_i|y_n)^{g_k}\}$$

where d is the total number of samples of training data sets. The final decision about the mode resulting from the consensus $\hat{y}$ will be (as in (3.9)) :

$$\hat{y} = arg\ max \quad P^T(Y|X) \tag{3.11}$$

To avoid having the situation when one of the probabilities is zero from affecting the decision, we take the logarithm of the probability values and sum them to get the same decision, then equation (3.3) would be:

$$Z_i = \sum_{k=1}^{m} \zeta_i^k \tag{3.12}$$

Where Z_i=log $P(y_i|X)$ and $\zeta_i^k = \log P^k(y_i|X)^{g_k}$. Also eq 3.4 would alter to:

$$Z^T = \{Z_1, \ldots, Z_n\} \tag{3.13}$$

where $Z_T = \log P^T(Y|X)$ and finally the decision would be made upon the following formula:

$$\hat{y} = arg\ max \quad Z^T \tag{3.14}$$

Table 3.1 : Table of Notation

Notation	Description
i	index value for store locations
BS	Base Station or Network Gateway
NCL	Normal Cycle Length
ACL	Abnormal Cycle Length
NCC	Normal Cycle Counter
ACC	Abnormal Cycle Counter
SRD	Sensor Readings Distribution
t	Sensor Transmission period
τ	Aggegator Transmission period
T	Participant Transmission period
S_t	sensing rate
F	Flag($)
$Dist_\tau$	Aggregator Normal Mode Distribution (NMD)
$Dist_T$	Participant Normal Mode Distribution (NMD)
lhD	last hour Distribution($)
$th1$	First threshold ($\mu \pm 6\sigma$)
$th2$	Second threshold ($\mu \pm 8\sigma$)
thx	threshold update factor
N	number of Participants connected to BS
n	number of sensors/ nodes connected to the aggregator
agg	number aggregators/ nodes connected to the participant P^k
r'	number of received readings from a node
r"	number of received readings from an Aggregator
r'''	number of received readings from a Participant
nt	abnormal mode notification message
req	a request to obtain Modes Probabilities
C	Consensus resultant decision
NSP^k	Network Status Mode Probabilities for participant P^k
MAP	Maximum A Posterior Process
NSM	Network Status Mode
M	Current Network Mode

Decision Variables

v	=	$\{v_i$; for i=1, 2, ..., r; where r: number of readings in a time period $\}$
D_l	=	$\{(v,F)^l_j$; for j= 1,2, ...,r' and l=1, 2, ..., n$\}$
I_a	=	$\{(D=[Dist_\tau/D_l], nt/F)^a_b$; for b= 1,2, ...,agg and a=1, 2, ...,r" $\}$

Algorithm 1: Sensor level Operation

Input : Sensor reading values v_i

Output: v and F

```
while all v_i in v do
    if |v_i| < th1 then
        t = NCL
        append v_i to v
        F = 'off'
        transfer (v and F)at t;
    else
        if |v_i| > th2 or |v_i| > th1 in v then
            t = ACL
            v= v_i
            F = 'on'
            transfer (v and F)at t;
        else
            t = NCL
            append v_i to v
            F = 'off'
            transfer (v and F)at t;
        end
    end
    Update lhD
end
```

Algorithm 2: Aggregator level Operation

Input : v^l and F^l from each connected sensor l

Output: $\begin{cases} Dist_\tau, & \text{if Normal Mode} \\ D^l & \text{otherwise} \end{cases}$

for $l = 1, \ldots, n$ **do**
  if $F^l =' on'$ *and/or* $|v^l| > thd1$ **then**
    request all connected nodes for more real-time data;
    send all data v^l with nt at τ =ACL
  else
    append v^l to D^l
    Call *Distribution*
  end
end
Function `Distribution`(v^l)**:**
  Calculate distribution parameters (μ, σ) from D^l
  update $Dist_\tau$
  transfer $Dist_\tau$ at τ =NCL
  return;

Algorithm 3: Participant level Operation

Input : $\begin{cases} Dist_\tau, & \text{if Normal Mode} \\ D^l & \text{otherwise} \end{cases}$

Output: $Dist_T$

for $l = 1, \ldots, n$ **do**
  if $nt \neq' null'$ **then**
    request all connected nodes for more real-time data;
    Call *ConsensusProcedure* Send the Results to BS
  else
    append $Dist_\tau$ to I^a
    Call *T2Distribution*
  end
end
Function `T2Distribution`($Dist_\tau$)**:**
  Calculate distribution parameters (μ, σ) from I^a
  update $Dist_T$ after adding the new parameters
  transfer $Dist_T$ at T =NCL
  return;

Algorithm 4: BS level Operation

Input : $\begin{cases} R_k, & \text{if Normal Mode} \\ C & \text{otherwise} \end{cases}$

for $l = 1, \ldots, N$ **do**
  if $ntl \neq' null'$ **then**
    initiate the $AbnormalModeProcedure$
  else
    append R_k *to* network dataset
  end
end

Algorithm 5: Consensus Procedure

Input : D_l

Output: M

Add recieved data to I^c
Check I^c in relation to NMD
if $|I^c| < thd1$ **then**
  Return
else
  Send nt and I^c to all other participants with req
  Apply I^c to Bayesian Classifier
  Obtain the Probabilites of each Mode of Operation
  Receive NSP^k from all other Participants P^k Compare all NSP^k and etimate the consequent NSM using (MAP)
  if $NSM = M$ **then**
    Update threshoulds by thx Diploy the new thresholds on the affected branch nodes
  else
    Upadate $M(M = NSM)$
    Act according to $ModeChangeProcedure$ in the Participant's preferences
  end
end

Chapter 4

Environmental Event Detection by Consensus Bayesian Machine Learning

4.1 Introduction

This Chapter introduces an experiment representing data delivery using the consensus management within Fog Computing environments. The aim is to simulate and implement the IoT environment introduced in Figure 3.2. This experiment clarifies the role of Consensus Data Aggregation using Bayesian analysis in maintaining the quality of data delivery within IoT. The approach is to exploit the consensus style of management methods to handle the delivery of data from a large number of sources so that the whole process of delivering big data within Fog environment can be improved. The goal of the proposed method is to collect data in an intelligent manner reducing traffic and eliminating redundancy and possible errors as well, so only the essential and non-faulty data is to be pushed further. Each level of the proposed paradigm has a decision making capability. This is because the purpose of the proposed model is to achieve automation and distributed computation within Fog Computing.

In this experiment, environmental events are to be detected by Bayesian Classifiers utilizing machine learning approach. Then a consensus decision about network mode is to be obtained by a consensus procedure employing the outcome of the classifiers. Along with that, data would be aggregated utilizing the distributions of the data transmitted from the front end till the base station considering the conditions explained in chapter 3. Temperature-Fire events type are depicted as an

environmental events that can happen in any setting. This type basically has four modes: Normal, Hot-temperature, Fire-ignition and Fire. For the simplicity and limited resources of prototype description, one event type (Fire) will be considered as an abnormal event in this setting. Bayesian analysis is to be used to illustrate the status of each mode (Normal or Fire) and how to determine the appropriate mode.

4.2 Data Collection and Analysis

To implement the experiment as shown in figure 3.2, a pre-simulation as carried out to check the potential of the proposed model. Therefore, the network was simulated using a function that generates random numbers within boundaries around standard room temperature environment. Also, some out of range values were inserted among the numbers to test the effect of events on the network operation. The network consists of four main branches each of which contains five sub clusters and are headed by a L2 Aggregator (Participant). Each sub-cluster has a L1 aggregator serves 10 sensors. The Consensus Management set of algorithms were applied to the network then a comparison was made to differentiate the network situation before and after applying the aggregation operation in terms of the number of packets. The results showed a huge savings in the number of packets after applying the aggregation algorithm even with the existence of events. The indication from these results was to proceed with the implementation of the experiment to generate data.

On a second stage, one branch of the same topology was implemented using Raspberry Pi Zero (RPi0) representing L1.1 and L1.2 Aggregators and 10 CC2650 SensorTags connected to each L1.1 aggregator (RPi) via BLE (Bluetooth Low Energy) connection. The aim is to monitor a warehouse environment and act autonomously if an event occurs. Then network operation was designed to get through the pre-exploitation phases.

4.2.1 Pre-Expliotation Mode

The first phase to be started with after setting the network is the calibration phase of sensors. Data starts to be collected by the Rpi from each SensorTag under close observation ensuring a normal mode of operation. The aim was to build the Normal Mode Distribution (NMD) for each sensor by getting enough data collected for several days. This step represents orientation for sensor data to set the threshold boundaries based on the Distribution. Data is assumed to follow Gaussian distribution, the formation of this distribution will be used to identify the occurrence of an event if a deviation in the readings is to be indicated considering the mean and variance of the distribution. Data distribution parameters (mean and variance) will be constantly updated as more readings are received from sensors. After this task, the experiment had to transmit to simulation due to limitations in resources and space. In order to make four appliances to fulfil the design topology, Boosting method was used to generate three other datasets each representing a branch of the topology.

For measuring the distributions of other modes of operations, different data sets could be directly measured, downloaded from the web, or simulated. Fire Dynamics Simulator (FDS) was employed to simulate fire situation and measure environmental readings from several sensors before and during the happening of fire event. Data were collected from the simulator and then Fire Mode Distributions were constructed for each sensor. The simulation were made to four different areas. Till this point, both normal and fire datasets (four dataset for each mode) were ready. The last data analysis to be done was to annotate the data as 'normal' or 'fire' to be fed into the system.

In the second phase, aggregator training phase, the classifiers are to be trained utilizing the annotated datasets. The purpose is to recognise different operation

modes inside the network and to relate data records to that mode.

4.3 Experimental procedure

Bayesian analysis model was used to perform management for network's data. It was the utilised to classify the network mode of operation. In Bayesian analysis, each mode will have a prior distribution depending on information collected. Then the ongoing measured data will be used to calculate the likelihood function and finally the posterior distribution can be extracted from these measures. When a reading exceeds threshold, indicating out of normal mode distribution boundaries status, the reading is to be applied on a scale that contains the distributions of relevant modes. Then, the probability of that reading being within each distribution is to be calculated, and lastly the distribution which has maximum probability value will be selected and its mode will be chosen as the decision. The distribution parameters for both Normal and Fire modes would be saved in L1.1 and L1.2 aggregators. At L1.2 , each data record received are fed to Bayesian Classifier with a Normal Mode or Fire Mode notations indicating that this record represents "Normal Mode" or "Fire Mode" operation status respectively. In this case the classifier can be trained to designate the normal and fire modes. When the pre-exploitation phase finishes, the network is ready to perform and, as expected, the performance would be in either normal mode or abnormal mode (when an event occurs). At each level of the model, a different responsibility and decision making is performed to attain distributed computing and automation within the Fog.

4.3.1 Environment Monitoring (Normal Mode)

In this mode, all reading data are within the Normal mode boundaries.

At L0:

Periodically, data collected from sensors and sent to L1.1 aggregator according

to Normal Cycle Length (NCL). While within normal mode boundaries, the mean and variance of Normal Mode Distribution set in calibration mode is continuously being updated.

At L1:

The L1.1 aggregator receives sensed data in each NCL time period, analyses, combine them according to their types (temperature, pressure, humidity, etc.), then abstract a distribution representing the data within that period and send its mean and variance to L1.2 aggregator.

L1.2 aggregator gets the Normal mode distribution parameters (mean and std) from all aggregators of L1.1, analyses and aggregates them in a combined distribution of all its sub-clusters. Finally it sends the parameters belonging to it to L2 Base Station.

4.3.2 Event Handling (Abnormal Mode)

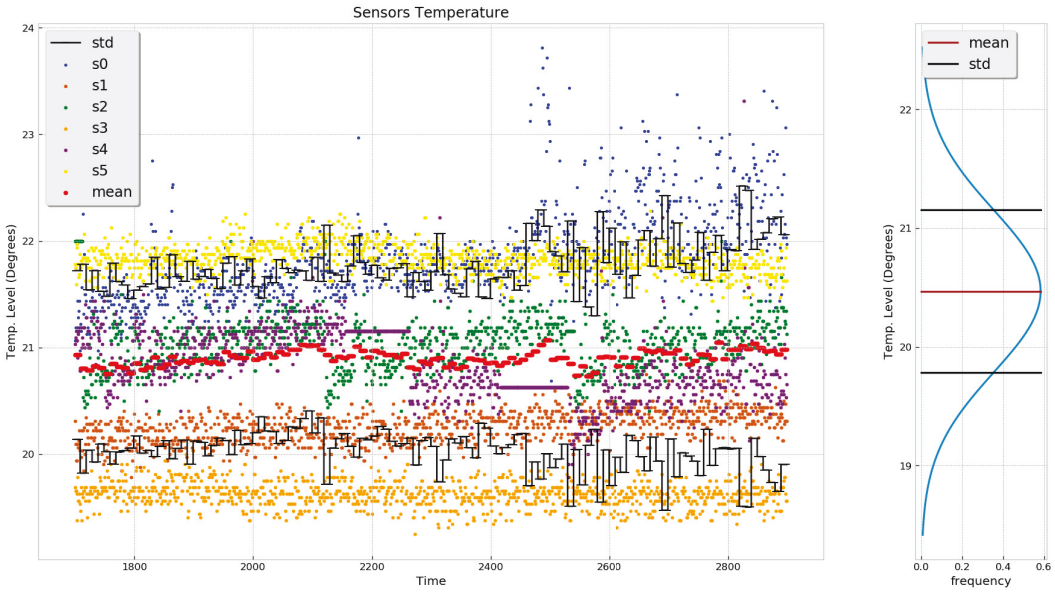
When a sensor node reading goes beyond the threshold indicating abnormal mode, the following will occur at different levels:

At L0:

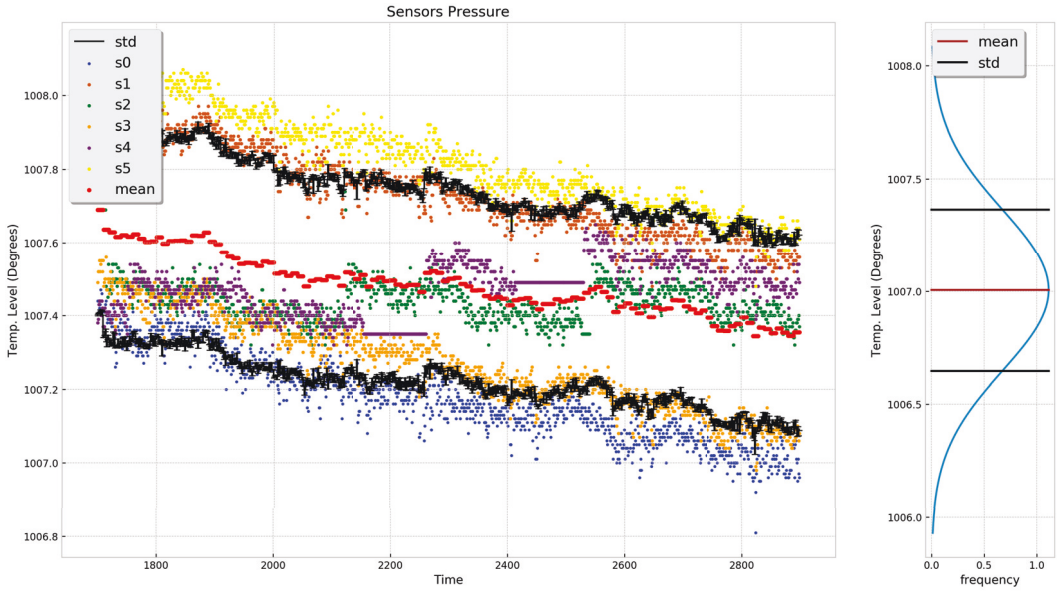
In the case that the abnormal value was greater than th1 but less than th2, the sensor waits for another value in the same range within the same NCL. If a second value occurs within the range or the reading goes beyond th2 from the first time, the sensor's decision would be to send a notification (set the flag bit to "on") and send data packets so that the aggregator would check it.

At L1:

The L1.1 aggregator receives packets with Flag on notification within packets of data in less than the NCL time period. It analyses them, sends requests to all its ascendant sensors requesting more data, then uses Bayesian analysis to indicate

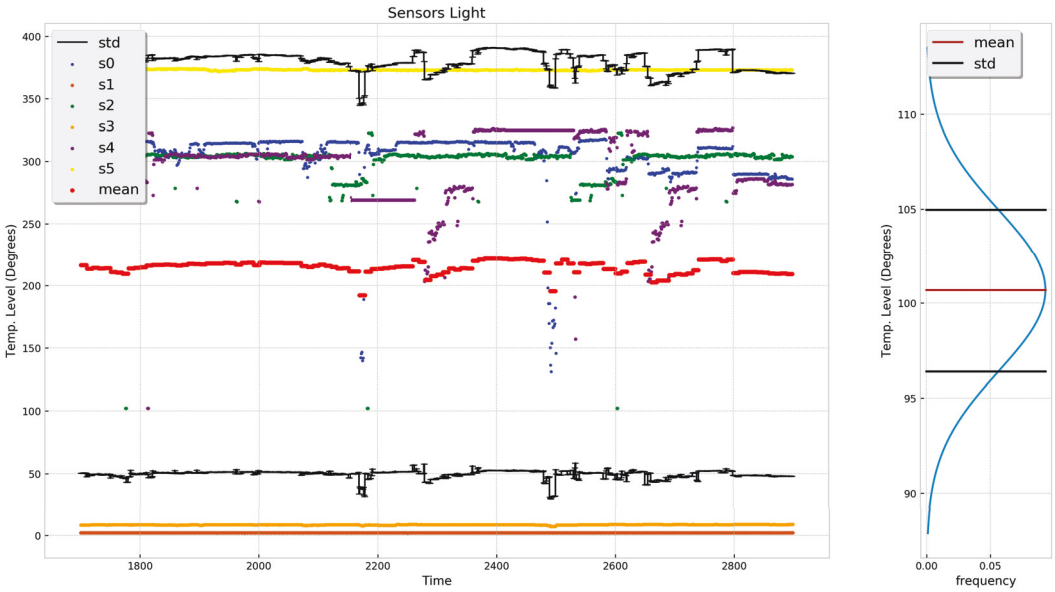


(a) Temperature

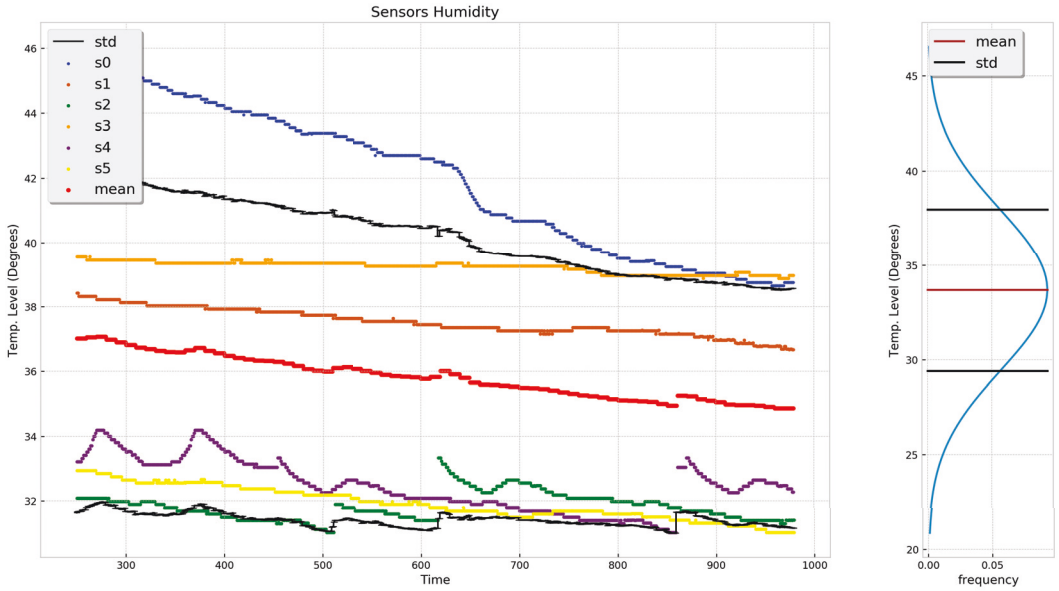


(b) Pressure

Figure 4.1 : Environmental Temperature and Pressure sensors data distribution parameters with its corresponding resultant normalised distribution after applying the aggregation



(a) Light



(b) Humidity

Figure 4.2 : Environmental Light and Humidity sensors data distribution parameters with its corresponding resultant normalised distribution after applying the aggregation

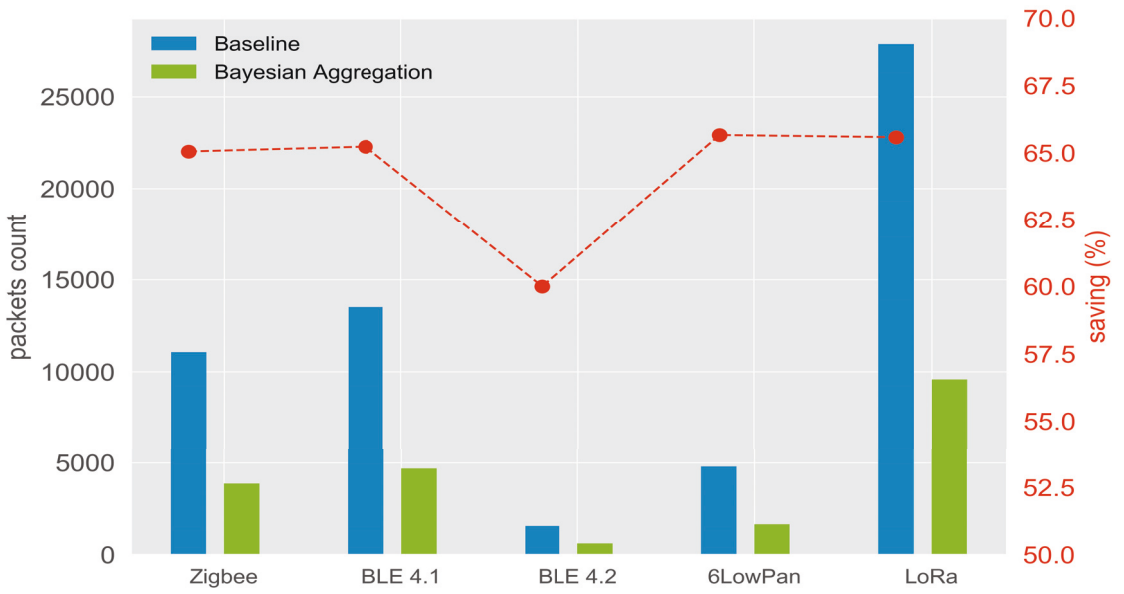


Figure 4.3 : The difference in Packets count between the Aggregated and non-aggregated (Baseline) cases with the savings in number of packets when using different communication techniques

whether its a Normal or an Abnormal situation. If the results from this process denote its a normal mode, the process ignores the notification coming from the sensor; if not, the decision would be to send these readings (not the distribution in this case) along with a notification to its ascendant L1.2 aggregator (Participant).

When receiving data and an abnormal mode notification from L1.1 Aggregator, the Participant tests the data in its Bayesian Classifier as well as sending the data and a consensus request to its fellow Participants. Each Participant would test the data based on its own classifier which trained using data generated in own network cluster and obtained the resultant probabilities for each operation mode. The result probabilities from all fellow participants for all modes (which is P(Normal — Data) for Normal Mode and P(Fire — Data) for Fire Mode) would be sent back to the original participant. Then the latter will apply the consensus estimation module to estimate the resultant mode based on all participant's experience.

The model simulation was run transferring packets that each carried readings from the datasets generated as mentioned in the data collection and analysis section. In the simulation, packets were aggregated as distribution parameters (meaning there will be only one packet representing the whole readings each NCL from each node) and transferred from the front-end nodes (L0 sensors) till the base station (L2 node) passing by all the ascendant nodes in the hierarchy model. When a packet carries an abnormal reading (notated as Fire Mode), the system would enter the abnormal phase. In this phase, the packets carrying readings instead of distribution parameters would be transferred. Also, the consensus operation between participants would take place including all the readings and notifications packets transfer as well as the results probabilities packets among participant. As normal mode is the dominant mode of operation in each network, the number of normal mode records in the dataset is much higher the fire mode records. A comparison was made to test the saving in packets number when applying the aggregation and in the case when no aggregation were applied. The number of packets were recorded for both cases.

The model was also tested by applying different communication means (BLE, Zigbee, 6LowPAN, LoRa) to show how the system responds regarding packets delivery amount and savings and communication rate/delivery delay. Considering the same network topology in figure 3.2, replacing each sensor with a SensorTag which has multiple sensors in one apparatus (like the CC2650 with ten sensors). Each sensor has different data length as in Table 4.1 which shows the size in Octets for each sensors group within the SensorTag (IR temperature: consisting of object and ambient sensors; movement: accelerometer, gyroscope and magnetometer; humidity: relative humidity and temperature; pressure: barometer pressure and temperature; optical: light intensity) as well as the IO service and Simple key services control octets. Adding two more octets containing a Flag bit for each sensor, the payload

(which is the part of transmitted data of is the actual intended message) required to represent each sensor reading within the sensortag is 60 octets. Since each sensortag sends the readings every NCL, the payload amount sent from the sensortag to the L1 aggregator is 60 multiplied by the number of sensing readings within NCL octets. The test will consider each communication mean at a time to deliver the same amount of data (same NCL) and count the number of packets that have traveled within the network for the case of baseline communication with no aggregation and then when employing our aggregation method throughout the system and thereafter calculating the savings in packets in accordance with aggregation. The objective is to have a clear idea about which communication technique better suits the proposed algorithm.

Assuming that the whole network utilises the same communication mean and that the system operates in Normal mode with NCL time cycle throughout all levels, packets number transferred over the entire network from L0 to L2 is to be calculated. Since each transmission technique has its own maximum payload size based on packet specifications standardised for that particular technique, a different number of packets would transmit across the network according to the applied communication mean. For a 6LoWPAN packet, as an example, the data units (PDUs) of the IEEE 802.15.4 protocol have various sizes based on the overhead it represents. Beginning with a maximum packet size of 127 octets in the physical layer with a 25 utmost frame overhead, the consequential maximum frame size is 102 octets at the media access control layer. Further overhead is imposed at Link-layer security, leaving only 81 available octets in the utmost case. Moreover, the length of the IPv6 header is 40 octets leaving only 41 octets for upper-layer protocols, like UDP which utilises 8 octets in the header. Hence, only 33 octets is left for application data. This space calculation demonstrations the worst-case state, and points out the need to perform header compression [162] [163]. The quantity of compressed octets from the IPv6

and UDP transport headers differs based on numerous factors like the used IPv6 addresses and 802.15.4 addressing modes, and the availability of network contexts.

The best compression case results in 6 octets from 48 [164] which concludes that the maximum payload to be 75 octets (81-6 is 75). Since ZigBee technique also depends on IEEE 802.15.4, it can consider the same payload size. For the ease of explanation and visualization, 6LoWPAN will be considered as the case with full header compression (75 octets) as its maximum payload size, while ZigBee would contain 33 octets for the rest of this discussion.

In the same direction, we can consider that maximum application layer payload for BLE version 4.1 is 27 octets and 251 octets for BLE version 4.2 (and above) [165], while LoRa would have 13 octets (its maximum payload size varies based on its transmission rate [166]). Figure 4.3 shows the packets count after applying the communication means for the baseline non aggregated transmission and with our Bayesian aggregation routing along with packets savings percentage for each transmission technique.

4.4 Results

The consensus operation is performed to get the best coherent decision among preferences, summing up the knowledge of all participants, and aiming for high level of accuracy. Therefore, a consensus operation was applied on a network that has three participants nodes and then another subnetwork (with different trained data sets) that has four participants nodes. The Consensus was performed using the Likelihood Multiplication method first, then by using the Majority Voting and the Weighted Average method to compare their performance. Figures 4.4 and 4.5 show the accuracy outcome of the individual participants decisions and the decisions of the three consensus methods for the first and second subnetworks respectively.

Table 4.1 : SensorTag Data Description

SensorTag Payload (Octets)				
Sensor Group	Data Size	Notification	Configuration	Period
IR Temp.	4	2	1	1
Movement	18	2	2	1
Humidity	4	2	1	1
Pressure	4	2	1	1
Optical	4	2	1	1
IO Service	1		1	
SK Service	1			
Total Octets				58

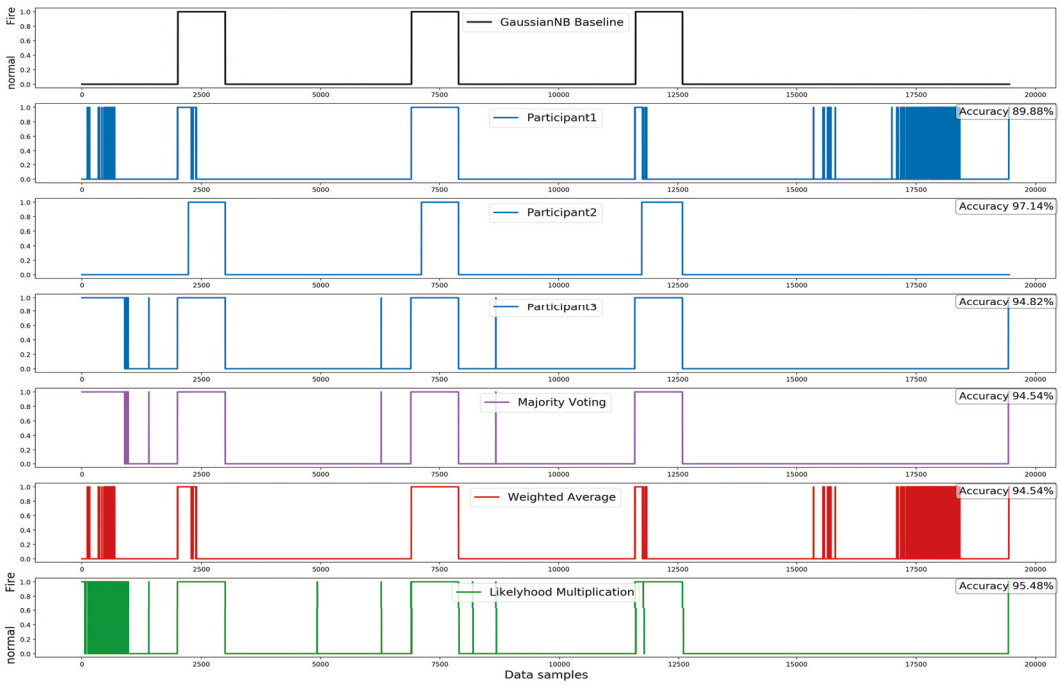


Figure 4.4 : The effectiveness of Bayesian Classifier and Consensus Procedure in mode detection accuracy (3 participants nodes)

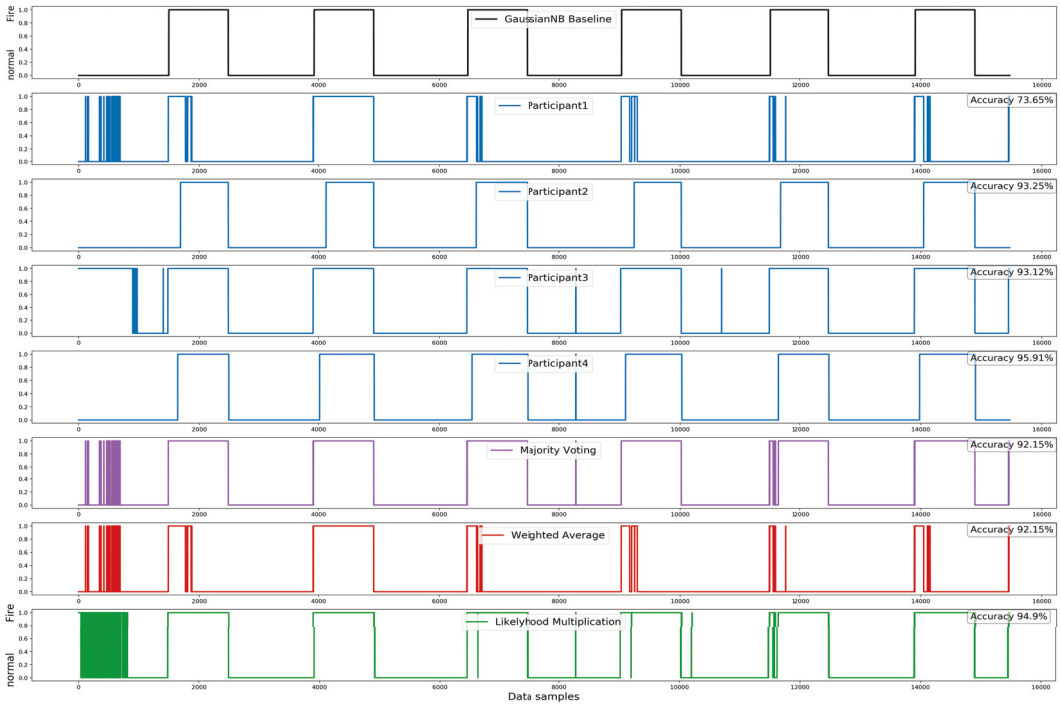


Figure 4.5 : The effectiveness of Bayesian Classifier and Consensus Procedure in mode detection accuracy (4 participants nodes)

To test the consistency of the superiority of our proposed method, the network with four participants nodes was run with the three consensus approaches respectively for a hundred times on different parts of the data as shown in Fig 4.6.

Another tests to the proposed model were made, in which the number of nodes within the network was gradually increased. Figure 4.7 shows the increment in packets savings in regards to the increase in Level 0 and Level 1 nodes in general. Also, by considering various communication means, each of which has different payload size for its packet. The results in figure 4.8 illustrate the savings that occur between the proposed Bayesian Aggregation (BA) and the Non Aggregated (NA) in different levels of the proposed model for each communication technique.

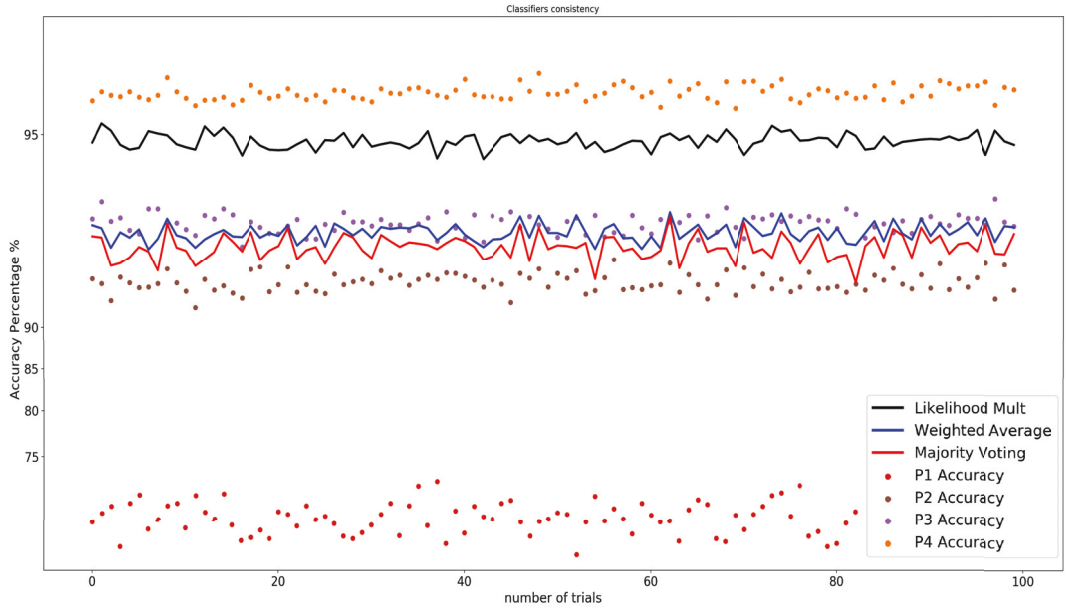


Figure 4.6 : Testing the consistency of the consensus approaches on detecting the mode of operation

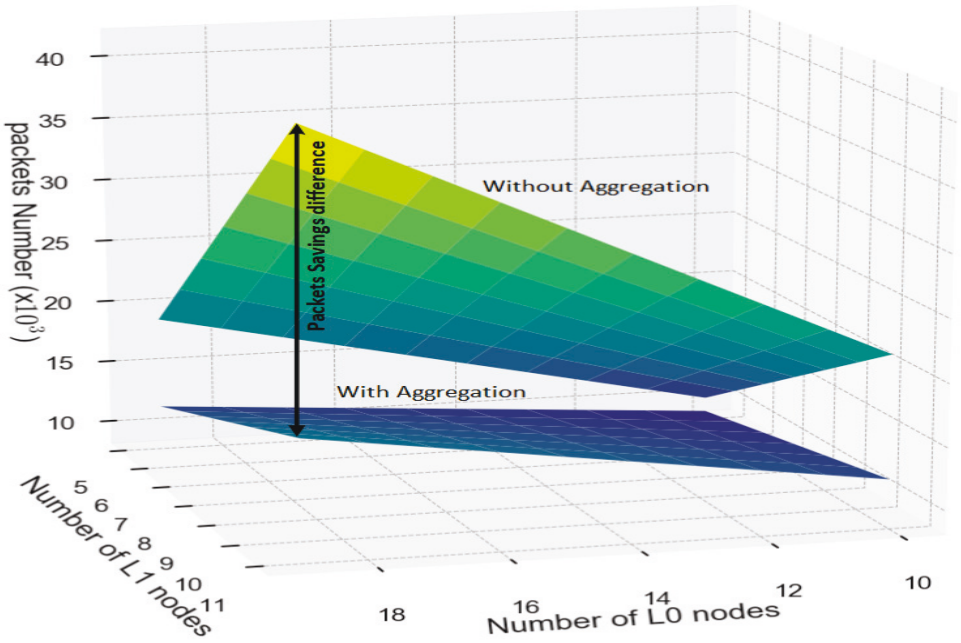


Figure 4.7 : Packets savings in Aggregated versus non aggregated cases with the increase number of nodes

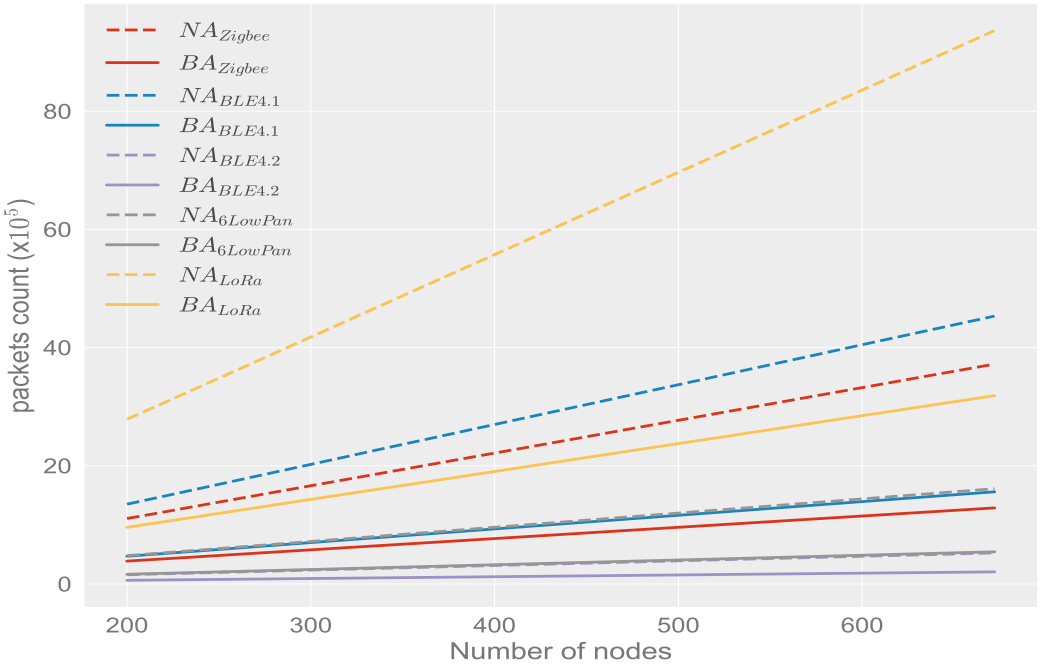


Figure 4.8 : Packets savings in Aggregated versus Non Aggregated cases with the increase number of nodes for each tested transmission technique

4.5 Evaluation

The aim behind our work is to gain high data quality that would increase the reliability of a system and implement autonomous distributed computation which can boost the efficiency of resources within Fog Computing using consensus based management techniques. For this purpose, the experiment was implemented utilizing the Bayesian Classifier and distribution along with its precision and efficiency will be evaluated in terms of mode detection as well as packet delivery. In order to test the accuracy percentage of the system model, and as mentioned that each participant's Classifier was trained utilises different data set (as each classifier would be trained by its own network data) as well as data distribution parameters of different modes (with the mode type for each record). Then a test data would be fed to those classifiers to check their ability in detecting the mode of operation of the network. In

this direction, the classifiers are tested with the data set so that its mode is known (but the mode was not fed to the classifier) and then the output was compared with the original data to inspect the correctness of the classifiers output. This process by which data separated into two parts is called cross validation, Training data to train the classifier and test data which is used to check how accurately the classifier was trained. The classifiers performance varied based on how relevantly close the trained data set was to the test data (in the real world, the classifiers decision is based on its own network data values).

When applying the proposed process on networks that has three then four participants nodes respectively, and as illustrated in the figures 4.44.5, is that the consensus technique has a very high percentage of accuracy. The conclusion gained from it is that it would be a reliable approach to achieve automation. This means that involving more than a participant (with its own experience) increase the efficiency of the model in distinguishing the mode of operation. Also, we can acknowledge that the confidence of the system (from accuracy levels) increases by nearly 2% when applying the Likelihood Multiplication Consensus method utilizing Bayesian estimation classifier than the Majority Voting and the Weighted Average methods. Also when consistently running the four participants nodes network with consensus for one hundred times, the Likelihood Multiplication Consensus algorithm shows the same attitude. It can be noticed that the likelihood Multiplication consensus method has a consistent best decision when using the Bayesian Classifier as it is close to the highest individual decision. However, the Majority Voting and the Weighted Average methods made their decisions near the mean of the individual participants' decisions.

In general, the consensus is consistently leading to improved detection accuracy. Aggregation ideally involves lowering the operation delay rate via summarising continuous sequence of values produced by sensors. Typically this saves the bandwidth.

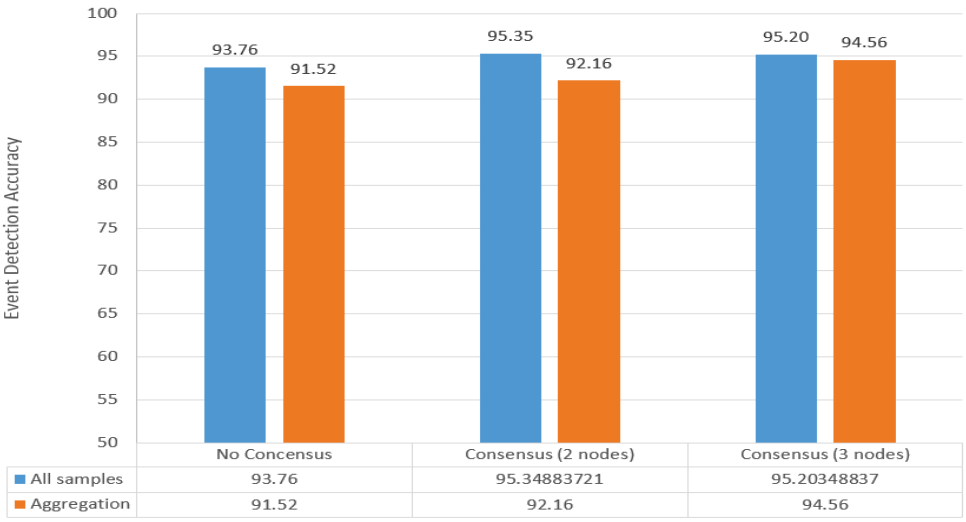


Figure 4.9 : consensus impact: the accuracy of the aggregated data samples increased as more nodes involved in the decision making (consensus) process

On the other hand, the summarisation may lead to accuracy loss as not the whole generated bulk of data would reach the Base station. The expected accuracy loss was quantified due to aggregation by comparing the case when no-aggregation applied ('all samples') vs when applying aggregation. On average, the event detection accuracy degrades by 2% when aggregation was applied considering one decision maker. When we use consensus (participating another node in the decision making process) the accuracy gap between the non-aggregated and the aggregated cases was decreased, which proves the effectiveness of the proposed scheme. Interestingly, the gap decreases as more nodes are included in the consensus process. This is evident in the three-node census where the aggregation introduces only 0.7% loss in event detection when using aggregation compared to considering all samples (no-aggregation case). Figure 4.9 summaries the impact of the two core traits of proposed framework on the event detection accuracy, namely the aggregation and consensus.

The aggregation by distribution approach used here significantly decreases the number of packets travelled from L0 to L2. Instead of sending all sensed data

(at least a packet for each SensorTag) readings within NCL all the way up to the Base Station, only one packet can be sent from each aggregator representing the distribution parameters of the entire sub-network. This leads to a huge save in network bandwidth, transmission time, number of collision and re-transmission data, memory and computation for higher levels, data scalability and energy consumption for the entire network. All that savings and operation happening at the front end (Fog Computing).

Another evaluation was made to check the effectiveness of the approach when enlarging network size. The outcome of this experiment showed that the percentage of savings in packets would increase by the increase of the network itself i.e. the more nodes/brunches the network has, the more savings in packets. This conclusion was obtained from observing figure 4.7. Similarly, when considering different communication means, we found that the savings occur in different levels according to the original payload size of each communication technique packet as shown in figure 4.8, the savings between the proposed Bayesian Aggregation (BA) and the Non Aggregated (NA). The smaller the payload size of a technique, the more savings we have when applying the approach. Considering figures 4.8 and 4.3, different levels of savings occurred with the use of different communication techniques for the same network. We can clearly notice that when performing with BLE 4.2, the saving percentage in packet numbers is a bit low in accordance with others. This is particularly due to the larger payload size when compared with other techniques.

The overall outcome of the system is benefited from the outcome of each level which is a result of the analysis being performed at that level to get only the desired data and push it forward. This leads the next level node to process the net data, leads to achieve optimal performance on the overall system. Getting the data abstract from the previous level nodes, where each level nodes eliminate redundancy, get abstract of data and send it to the next level, not just saves network resource

(such as packets, energy and delay time), but also directs the analysis and decision making towards the desired outcome. In this direction, level 0 outcome fed into level 1 is already within the desired boundaries so that the analysis at level 1 would be directed within controlled normal mode boundaries to aggregate that data and send it forward to the ascending node in the level (node 1.2). Then, that node would directly abstract the received data and send it again causing big savings in network resources and more controlled outcomes. Since level 1 abstracts data, if normal mode, coming from vast amount of sensors into one frame, most of the savings happening at that level.

4.6 Conclusion

Consensus Data Management is the method where certain aggregators agreed on a plan based on a set model. In this experiment, the complication of Big Data in the Internet of Things in terms of data quality and network efficiency is studied. Additionally, the use of Statistical Machine Learning is considered which would grant the system to acts as an autonomous entity. The experiment demonstrates by means of structured and operational framework that the proposed algorithms can attain markedly better performance than current solutions in practical cases. Especially the introductory of the Likelihood Multiplication process with Bayesian machine learning (classifier) criteria that eases the L1.2 Aggregators to make decisions based on the trained previous data which drives the algorithm to be a good candidate for deployment on all Fog devices because of its light computations. However, the model requires that the system first perform the calibration phase to extract the distributions of all the front-end nodes and to train the participants. The system is also required to send the data (not only the distribution) to the ascendant node whenever an abnormal mode is suspected (when the threshold goes beyond boundaries) in order to perform the consensus-based decision making process. It is shown

that bringing consensus using the probability approach of Bayesian analysis is persistently leading to enhance detection accuracy and acquire the system to be able to autonomously improve data quality. Additionally, the experiment results showed that packets savings percentage would increase by increasing more nodes/brunches in the network. Obtaining data distribution from each level nodes does not only save network resource, rather directs the decision making and analysis towards the desired output as well.

Chapter 5

Human Activity Recognition by Consensus Bayesian Deep Learning

5.1 Experiment description

Essentially, the experiment aims to test our proposed model over existing dataset to check its suitability in handling different datasets i.e. different environments and scenarios. In this experiment, we choose human activity recognition as a scenario to evaluate the outcome when applying the proposed model. Human activity recognition is concerned with the issue of forecasting what people are doing depending on records of their movements employing sensors. It categorises a series of accelerometer information recorded through specified harnesses or smart phones into a recognised well-defined set of movements and surrounding environment changes. Recognition can be achieved via exploiting the data coming from different origins including environmental or body-worn sensors. The concept is that when the subjects activity is identified and acknowledged, a smart computing system can then provide services. It is a challenging issue given that in general there is no obvious analytical process to link the sensor records to any particular activity. It is far too technically difficult due to the huge amount of aggregated sensor data (the enormous observations number generated every second), the observations' time-based nature, and the classical utilization of hand crafted heuristics and features from these records in maintaining predictive models. Sensor-based activity recognition pursues insightful knowledge regarding human actions from a mass of sensor records. In recent years, conventional pattern recognition techniques have achieved superb

improvements. Still, those techniques frequently strongly depend on hand-crafted heuristic feature extraction that can restrict their general performance. Lately, the latest development of machine and deep learning makes it viable to perform automated high-level feature extraction, for that reason accomplishes promising overall performance in lots of fields.

Principally, a consensus decision process was created over individual data sets, then the same on an aggregated set. Finally the study will compare the decision processes. The Aim is to investigate the liability of performing distributed Fog Computing through consensus procedure and its feasibility over Cloud Computing. It hypothesises that it is possible to use Bayesian Deep Learning approach to recognise users' activities and authenticate them through an ensemble approach.

5.2 Dataset

A typical dataset entitled Human Activity Recognition Using Smartphones Dataset was made available on the UCI Machine Learning Repository by Davide Anguita, et al. from the University of Genova, Italy in 2012. The dataset was described in their paper [167] while it was modelled with machine learning procedures in the paper [168]. The data was recorded from 30 people wearing a waist-mounted phone and performing six standard activities movements, then the movement records were labelled manually. The activities were: Walking, Walking Downstairs, Walking Upstairs, Standing, Sitting, and Laying (ladled as 1 to 6 in the processed data file and the outcome figures). The movement records were linear acceleration (x, y, and z accelerometer data) and angular velocity (gyroscopic data) from a Samsung Galaxy S II smart phone. Every person performed a series of actions twice, first with the phone on their left-hand-side and the second when it was on their right-hand side. Several frequency and time features commonly utilised in recognising human activity field were extracted from each window data using feature engineering. The outcome

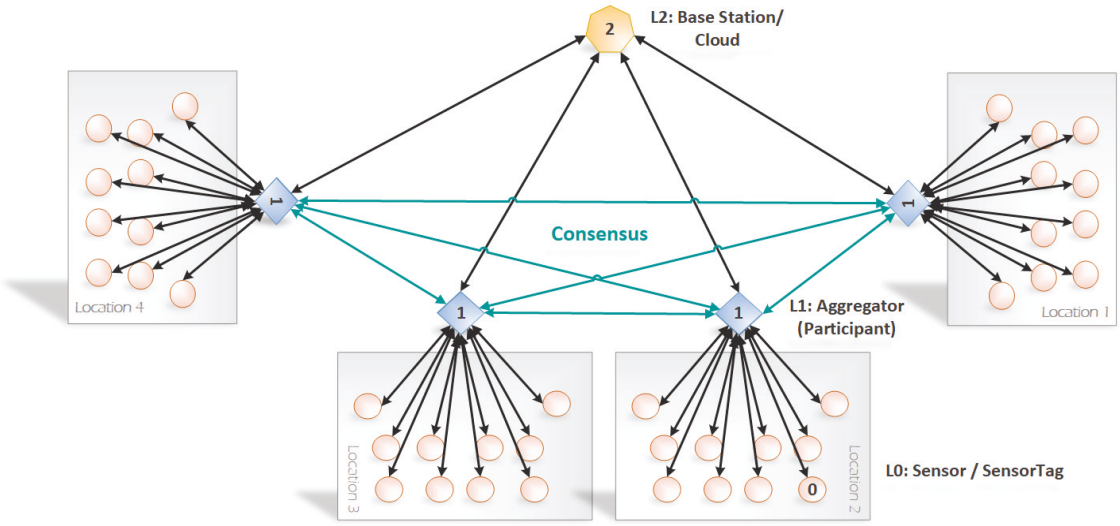


Figure 5.1 : Assumed Topology to suit the experiment dataset into the proposed paradigm

was a 561 element features vector.

5.3 Assumptions

To adapt the data to the proposed model, I assumed that the data was measured in four separate places and then aggregated by a node representing the cluster head of the network in that place. In this case, and according to the proposed design, the assumed topology of this layout would be to have L1 which contains one layer of aggregators each of which contains a cluster head and Participant entity. Thus, the whole training data was divided into four parts. Furthermore, each cluster here would represent the participant of the consensus operation. Thus, the data is partitioned into four separate training data sets then a statistical model with a Bayesian Neural Network was built for each of the individual four training data sets. In addition to having a testing dataset. The assumed topology would be as shown in figure 5.1.

5.4 Data Analysis

Neural Networks requires a large amount of data to perform deep learning and output good results. When first dividing the training data into four partitions, each of which has 25% of the total dataset, the accuracy level was not as high as expected especially when using Bayesian Deep Learning procedure. The results were as shown in table 5.1, therefore the human activity dataset used in this experiment required certain processing and analysis before cross validating it (training /testing).The aim behind this is to suit the model and to output the best possible outcome. The dataset has quite a few features (561 features per raw), but its overall length is not that long (only 7352 rows for the training set). Thus, partitioning the data into four sets would make each partition quite small to train each of the subsequent models. In this case, data was analysed using random bootstrap sampling before creating four training data partitions. Bootstrap sampling is the technique of repeatedly drawing a small sample from a single original sample by means of resampling lots of the same size small samples to create a more complex and dynamic records. As the length of each partition needs to be increased, Bootstrap sampling was implemented to resample more data from the original dataset so that the Neural Network is trained by employing more data. The resampling process were kept until each partition length reached 95% of the original dataset. An important note here is that each partition does not contain 95% of the original dataset rather having data length equals to 95% of the dataset length results from resampling multiple samples from a part of the original data.

5.5 Model Procedure

A Bayesian Recurrent Neural Network (BRNN) was utilised to develop a classification model for each training partition. The models were then used to categorise one of six labels in a 'test' data set. The process is then directed to obtain consensus

Table 5.1 : Accuracy levels after using 25% of the dataset

Entitiy	Prediction Accuracy
Participant 1	89.7%
Participant 2	87.1%
Participant 3	93.5%
Participant 4	93.7%
Weighted Average	89.7%
Majority Vote	89.7%
XGBoost	89.7%
Overall	96.93%

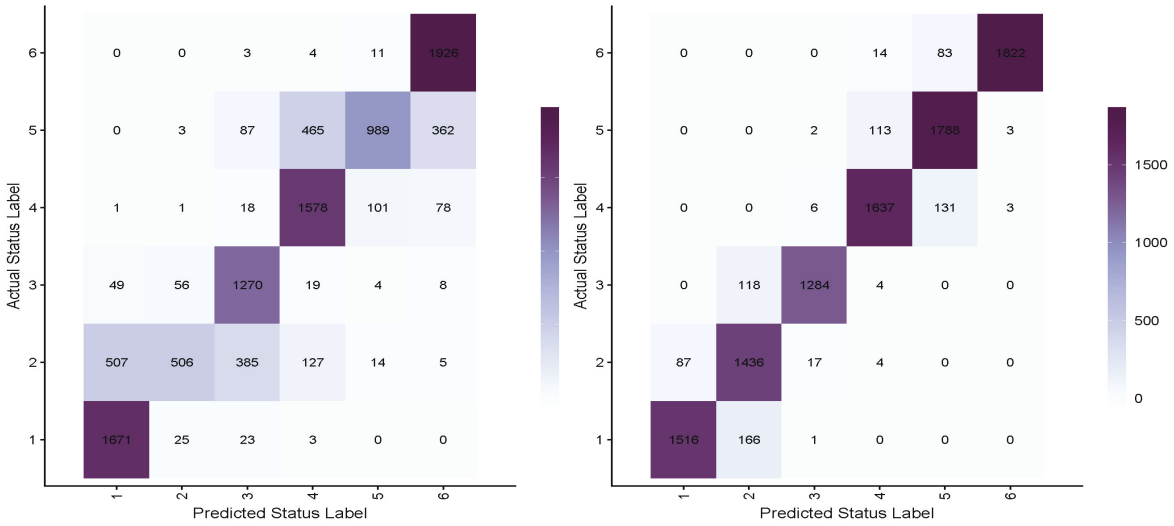
decision aimed to increase its accuracy. Three additional 'ensemble' models were developed with various methodologies. First, a 'Weighted Average' of the four training models was developed for classification. The 'Weighted Average' used the following weights for each model: participants 1 and 2: 25%, participant 3: 50%, participant 4: 0%. Next, an Extreme Gradient Boosting (XGBoost) ensemble method was developed using the training data. This method presents a model that makes collaborative prediction from models that have weak predictions via decision trees technique. This model is first developed in a stage-wise style same as other boosting approaches, then it infers them by permitting optimization of a random loss function. Finally, a 'Majority Vote' ensemble method was developed using the four training partitions. This combined the four model predictions for each of the six labels and selected the winning model. We made one more BRNN model as we want to test the advantage of the proposed distributed computing consensus process in relation to the central computing process. The model is to train the overall dataset, as in the case when the base station has an overall knowledge of all network data

while each participant node knows only about its own data partition. The purpose of this is to compare the accuracy of the overall model with that of each ensemble.

5.6 Outcome

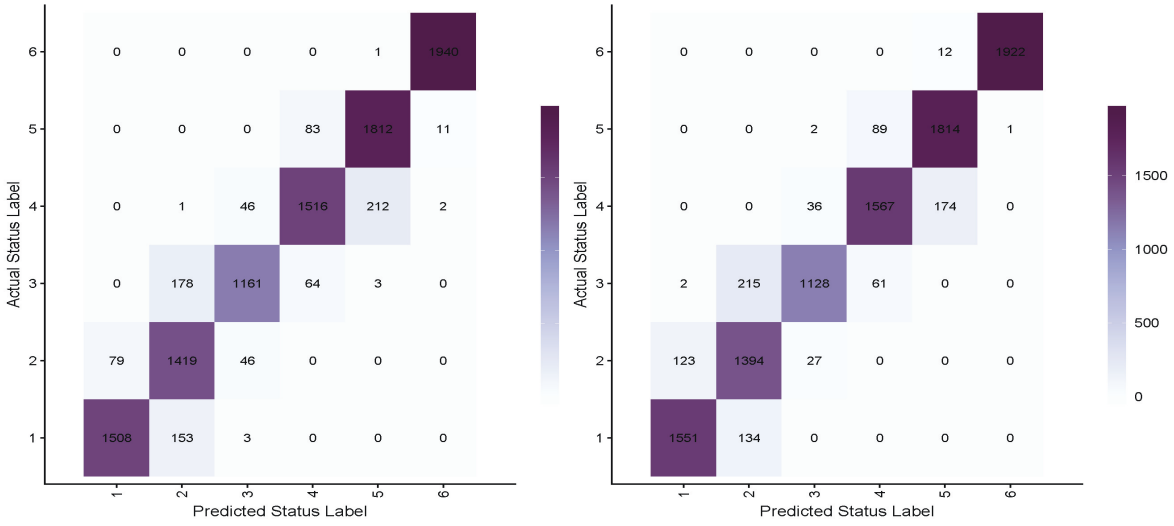
The BRNN models result in a set of predictive accuracy. The confusion plot in figure 5.2 shows the predicted outcome of the six activities' status in relation to the actual status recorded in the dataset. Where the diagonals show the number of correct predictions, the ones bellow the diagonals represent the incorrect predictions for each label. Subplots (a) to (d) output the BRNN prediction outcome for Participants 1 to 4 respectively, while in figure 5.3 the (a) to (c) subplots represents the ensembles prediction outputs vs the actual data. However, subplot (d) illustrates that relation for the dataset as a whole (as a central computing node would perform instead of the distributed nodes presented in our design).

In Machine Learning, when it is about classification issue, performance measurement is a crucial task. In this case, an AUC-ROC Curve can be used. An AUC-ROC curve is a classification task performance measurement at different threshold settings. When the performance of a multi-class classification task is needed to be checked or visualised, the AUC-ROC curve is used (AUC stands for Area Under The Curve while ROC is Receiver Operating Characteristics). It is a very significant evaluation metric when examining the performance of any classification model. ROC represents the probability curve while AUC is the measure or degree of separability. It illustrates the capability of a model to distinguish between classes The higher the AUC, the better the model is at predicting or distinguishing between labels [169][170].Figures 5.4 and 5.5 show the AUC-ROC performance curves of the system before performing the random bootstrap sampling on the data. The first figure 5.4 refers to each of the four data partitions (representing data at each participant) performance after performing the Bayesian Deep Learning (BDL) algorithm



(a) Participant 1 Confusion matrix

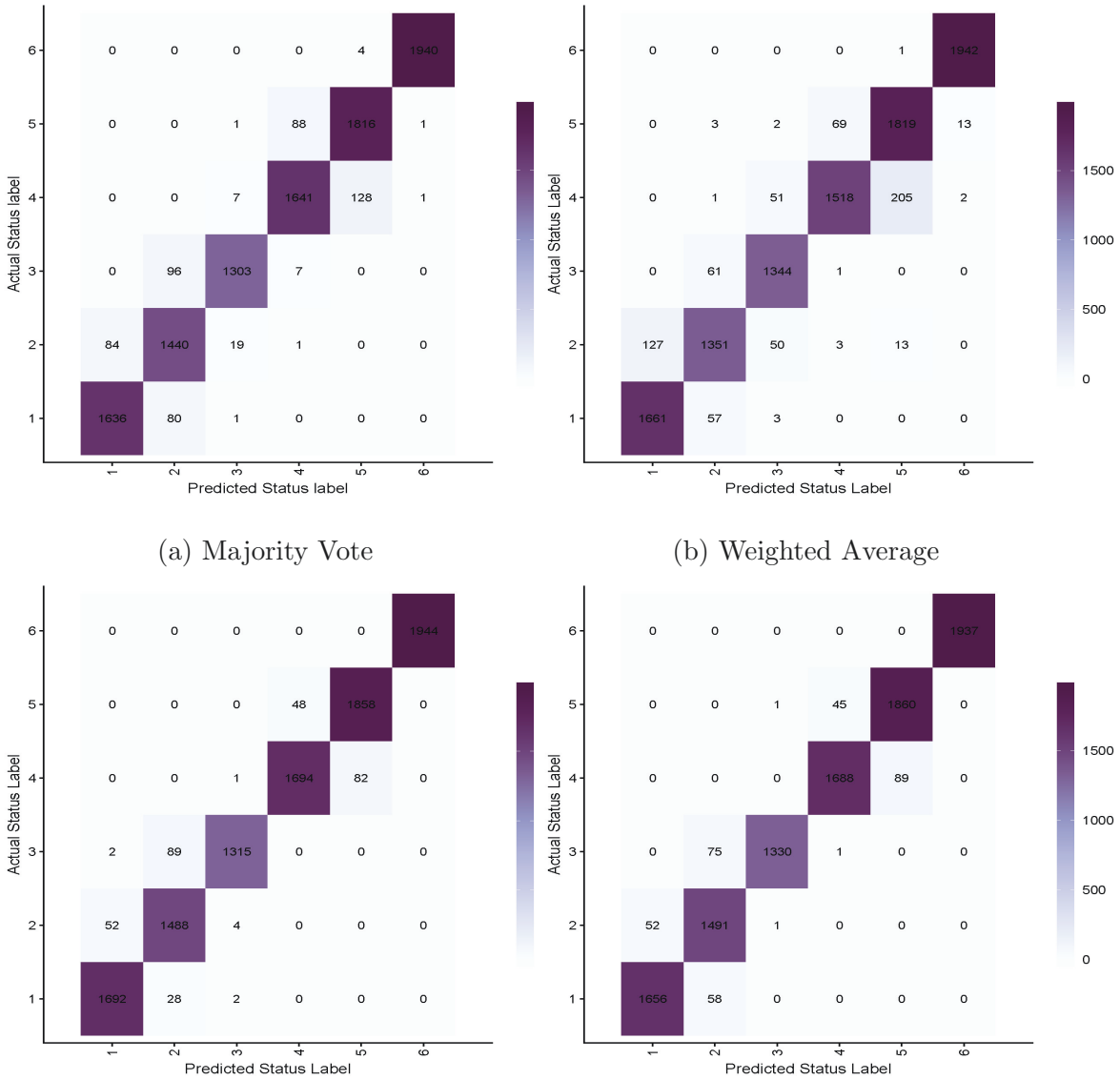
(b) Participant 2 Confusion matrix



(c) Participant 3 Confusion matrix

(d) Participant 4 Confusion matrix

Figure 5.2 : Confusion Plot of the Human Activity Recognition by Consensus Bayesian Deep Learning. a-d are the four participants.



(a) Majority Vote

(b) Weighted Average

(c) Adaboost

(d) Overall

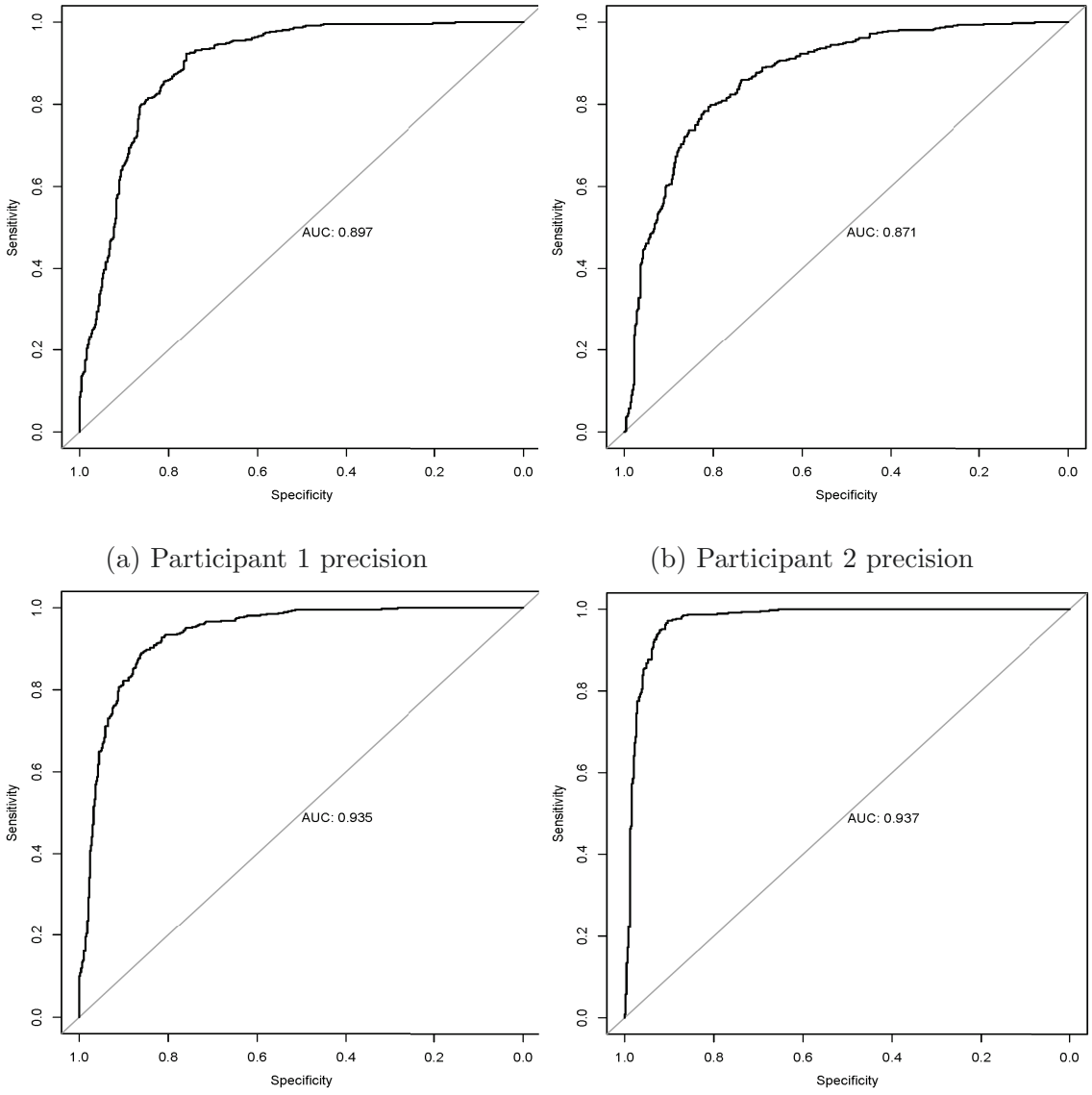
Figure 5.3 : Confusion Plot of the Human Activity Recognition by Consensus Bayesian Deep Learning. a-c are the consensus methods, while d is the overall as if the whole data reached to the BS

on each of them. The second figure 5.5 illustrates the consensus output of three ensembles (Majority Vote, Weighted Average, and Adaboost) as well as the performance of the whole data without partitioning (representing all the data received at the BS or Cloud). In contrast, the performance AUC-ROC curves for the data after applying the random bootstrap sampling is shown in figure 5.6 for the four new partitions. Additionally, figure 5.7 shows the AUC-ROC performance comparison after preforming the consensus Adaboost ensemble on the four partitions' outputs and when performing the BDL on the whole data.

5.7 Evaluation

The purpose of the second experiment is to test the suitability of the proposed paradigm in a different environmet setting. This experiment explores the scenario where sensors are mobile (attached to the people's mobile phones) and are recording the activities of the people in the environment. The setting has four locations to collect data. Each location includes one aggregator that aggregates all sensors' data in that particular location as well as contains a participant entity to predict decisions and perform consensus. Bayesian deep learning was employed as a decision making method to vary it from the previous experiment. Since deep learning is a very effective practice, the intention behind using it is to execute a procedure that combines the effects (output) of multiple deep learning appliances and investigates its outcome. In the experiment, ensemble techniques were used for performing consensus. The ensembles used were the Weighted Average, Majority Vote, and XGBoost.

At the beginning, the training dataset was divided into four partitions each of which contain 25% of the data. This resulted in making the BRNN models to have predictive accuracies of around 89% as shown in table 5.1 and figures 5.4 and 5.5 which contain the ROC plot of the participants, the ensembles as well as the prediction's accuracy of the overall dataset. The results of the individual



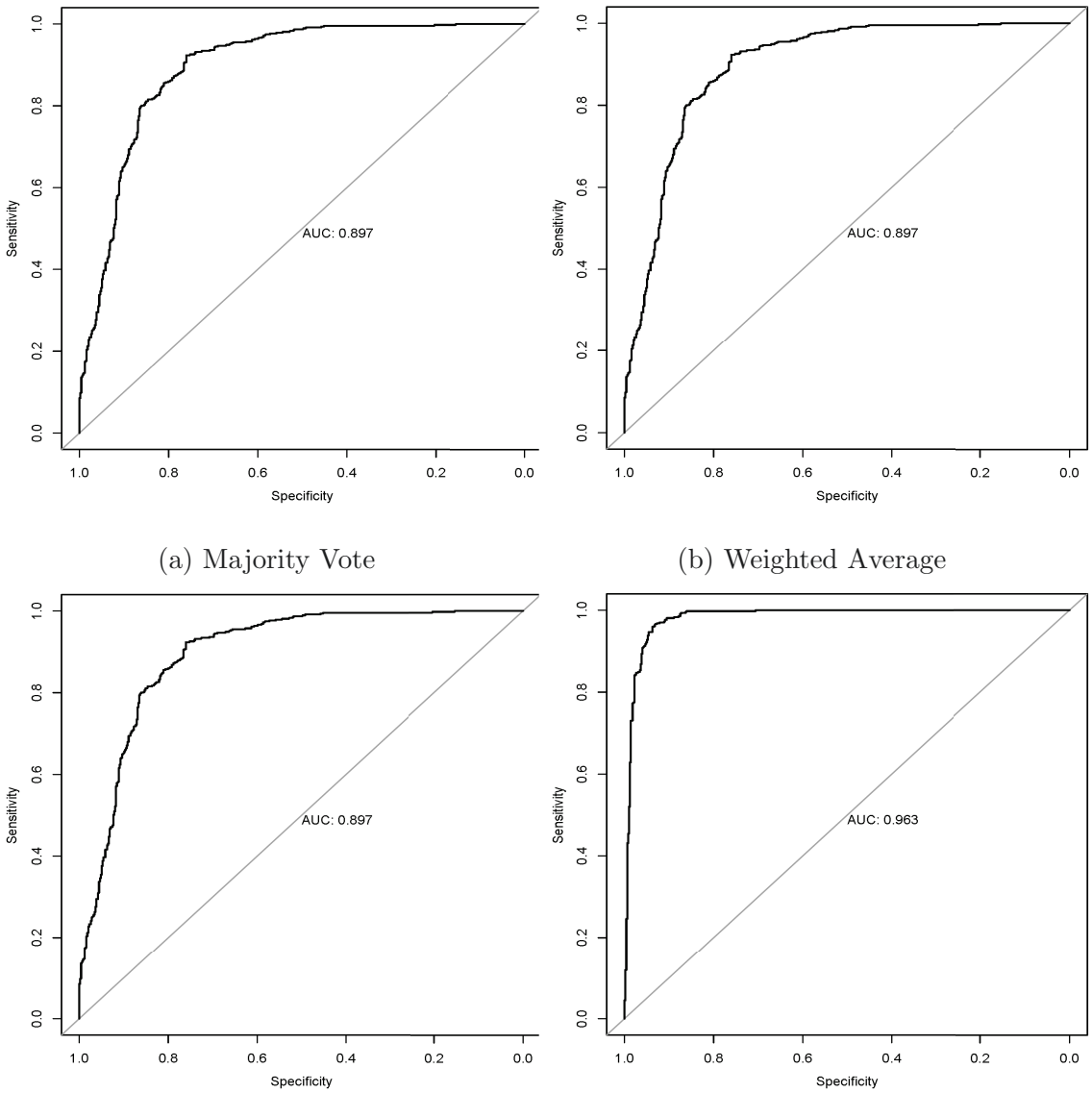
(a) Participant 1 precision

(b) Participant 2 precision

(c) Participant 3 precision

(d) Participant 4 precision

Figure 5.4 : AUC-ROC Plot of the Human Activity Recognition by Consensus Bayesian Deep Learning. a-d are the four participants



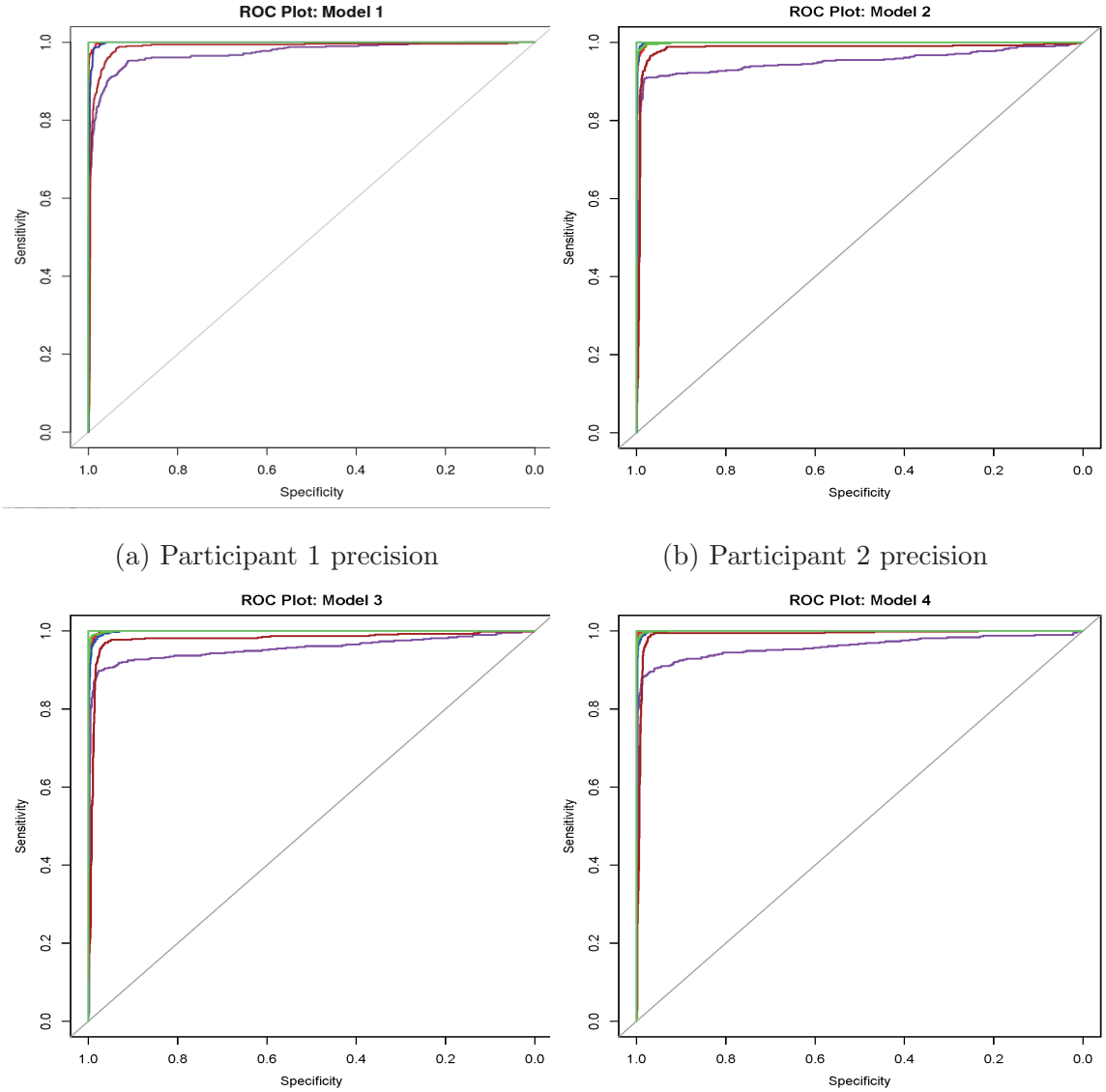
(a) Majority Vote

(b) Weighted Average

(c) Adaboost

(d) Overall

Figure 5.5 : AUC-ROC Plot of the Human Activity Recognition by Consensus Bayesian Deep Learning. a-c are the consensus methods, while d is the overall as if the whole data reached to the BS



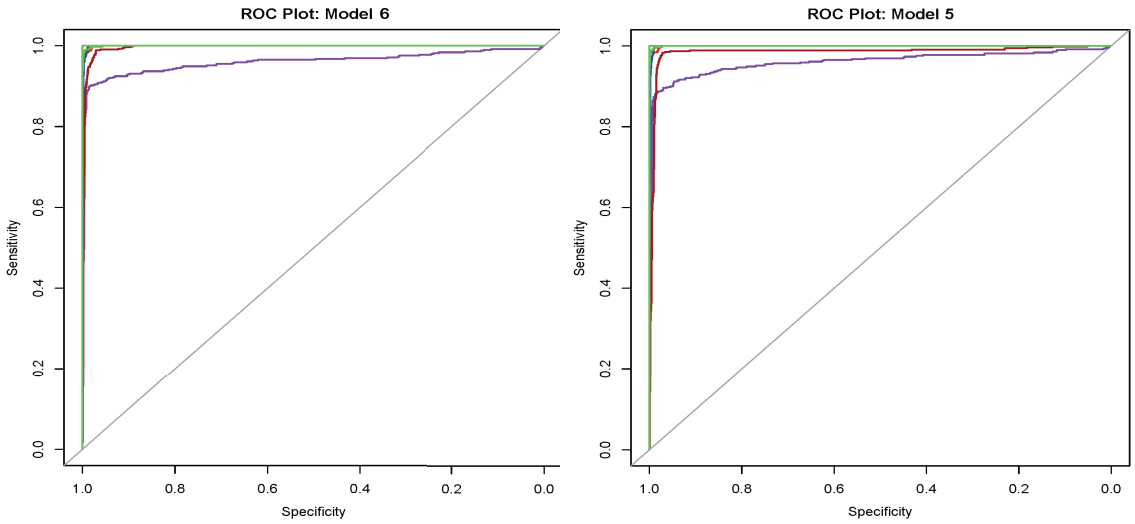
(a) Participant 1 precision

(b) Participant 2 precision

(c) Participant 3 precision

(d) Participant 4 precision

Figure 5.6 : AUC-ROC Plot showing the precision of the 6 activities' result by training each of the four participants after performing the random bootstrap sampling on the data



(a) Adaboost (b) Overall

Figure 5.7 : AUC-ROC Plot showing the precision of the 6 activities result from the Adaboost consensus ensemble after performing the random bootstrap sampling on the data vs. the overall model

participants and the consensus models are nearly close to the results obtained by Kolosnjaji and Eckert(2015)[171] as they tested the same dataset using a pre-trained neural networks with Dropout on the whole dataset. They compared the Dropout procedure with the random forest classifier and the outcomes illustrate that the Dropout had slightly better performance and that the training had given a precision of nearly 90%. While when the BRNN was applied on the overall data, the accuracy was much higher. This indicates the superiority of the BRNN over the other methods mentioned in [171]. It also demonstrates that even when the BRNN has 25% of the data to train on, it results in a nearly accuracy compared to those methods.

As deep learning practices require comprehensive amount of data to adequately train the Neural Networks, the partitioning percentage of the datasets increased for each dataset part. Random bootstrap sampling (which records random samples over and over again from the same group of data even on the occasions that it records the

same samples) was used for this purpose to obtain a size equal to 95% of the original overall dataset for each partition in order to increase prediction accuracy. For investigating whether the proposed paradigm fits in another setting, a mobile sensors data were picked to fulfil this purpose. The accuracy outcome for detecting correct activities were very high which points out the suitability of the paradigm when using mobile as well as fixed position sensors. The prediction accuracy increased for the participants as well as the ensemble to the accuracies shown in table 5.2. At the time when most of deep learning studies are focused on image processing and vision, this experiment shows the high degree of accuracy when using deep learning to predict activities based on sensors' data (i.e. its suitability with sensor data) with a requirement for certain data analysis. It is evident that the consensus models surpass the individual ones and that they reach the extent where they could overpass the overall prediction accuracy. The results show that when using the distributed computing with collaborative decision making (consensus), the accuracy trade-off is less than 1.5%. The importance of this observation is that the experiment highlights the possibility for the aggregate of decisions resulted from performing deep learning on subdivisions of a dataset, to outperform the decision of applying deep learning on the whole dataset.

5.8 Conclusion

The second experiment utilises deep learning techniques to perform the cross validation as well as the consensus operation (represented in the ensembles techniques). Patterns recognition is a crucial part in this experiment to predict the activities that occur within the environment. Data analysis (data cleaning, pre-processing, and data wrangling) plays a vital part in preparing and amending data to be fed into the system. The experiment shows that prediction accuracy increased for the participants and the consensus models and that the accuracy of consensus models

Table 5.2 : Models Prediction Accuracy

Entitiy	Prediction Accuracy
Participant 1	95.57%
Participant 2	96.26%
Participant 3	96.01%
Participant 4	94.02%
Weighted Average	96.42%
Majority Vote	96.93%
XGBoost	97%
Overall	96.93%

surpass the individuals. This demonstrates that the paradigm is appropriate for mobile and fixed location sensors. This Experiment also showed that after implementing data analysis upon the data then applying the BRNN method, the results shows a superiority over some other methods implemented by other research in [171] over the same dataset. It also illustrates that each of the consensus methods were able to surpass the accuracy of the overall model. The latter gives an indication that Fog Computing processes can compete with Cloud Computing in terms of accuracy with the added advantage of locality.

Chapter 6

Recognizing Smart Home Resident Activities by Consensus Bayesian Deep Learning

6.1 Experiment and Dataset Description

In smart homes, uncommon prospects offered by technologies like data mining and pervasive computing provide context-aware services, incorporating home health and wellbeing monitoring. Smart environment systems are required to identify and track activities in which residents ordinarily do as part of their everyday habits in order to provide such services. Nevertheless, recognising activities has normally involved aggregating and classifying large quantities of data in each location in order to learn the activities model in that location. Activity recognition is commonly utilised to measure the residents functional health. It also allows the smart home to react in a context-aware manner to requirements for attaining more security, wellbeing, and energy efficiency. It is challenging to learn the activities since the captured sensors data is rich in assembly and volume. Usually, each environmental circumstance has been addressed as a discrete setting wherein to perform the learning process. What can be researched in smart home is the capability to leverage knowledge of former states in new settings or with new individuals.

When someone looks at photos or video of people doing normal residential activities like sleeping or eating, he recognises such activities directly, even if he has not observed the environment or the residents before. This directs the concept of obtaining general models about activities from learning over particular environments and residents. In this experiment, the proposed blueprint was applied by exploring

Bayesian Neural Networks (BNN) algorithms to learn smart home activity models. Then, some consensus ensemble models were implemented to the output of the BNNs to evaluate their performance and their fit into the proposed blueprint utilising the ARUBA datasets from CASAS Smart Home project [172]. The focus in this experiment is on measuring the uncertainty, as it is one of the main reasons for employing Bayesian deep learning technique.

The dataset noted as Daily Life activities was made available on the CASAS Smart Home project developed at Washington State University. It was defined in the paper [173] while it was demonstrated utilizing supervised and semi-supervised machine learning procedures to explore setting-generalised activity models in the paper [172]. The dataset contains sensor data collected in a volunteer womans apartment with some relatives who visit regularly. The file contains raw annotated sensor activities generated from motion, door closure, and temperature sensors. The dataset marked eleven daily living activities, noted as: Eating, Bed to Toilet, Sleeping, Enter Home, Leave Home, Meal Preparation, Housekeeping, Relax, Respiration, Work, and Wash Dishes.

6.2 Assumptions

In order to fit the dataset into the proposed model, the data was assumed to be measured in several similar apartments (the first assumption was to make ten parts then it was changed to 4 as explained in section 6.3), so the training data was separated into ten partitions (the final partitions were four). The assumption involves aggregating each apartment data by a node representing the cluster head of the network in that apartment. And each cluster head would symbolise the participant in the consensus operation. Then, a Bayesian Neural Network was assembled for each of the partitions.

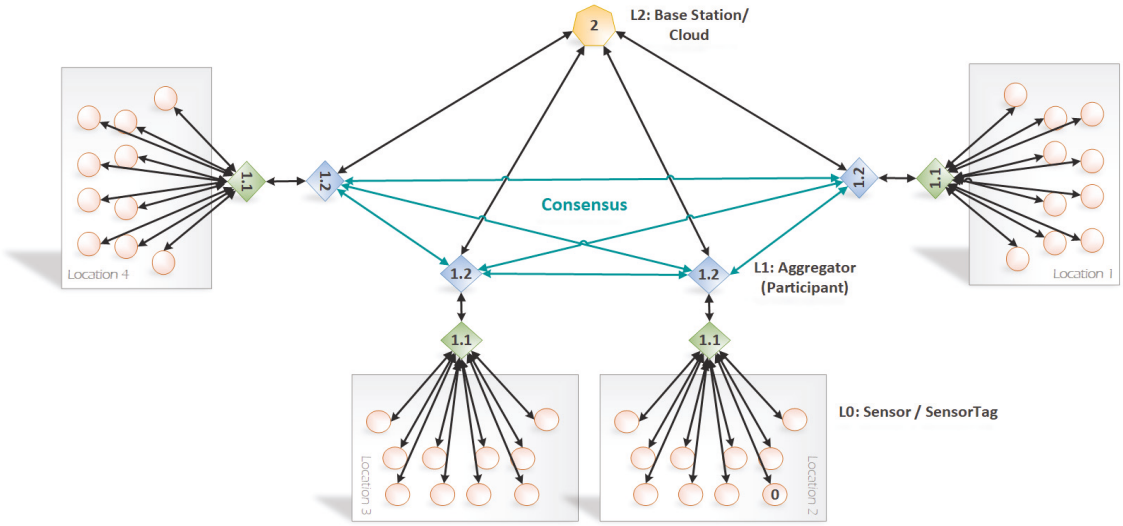


Figure 6.1 : Assumed Topology to suit the experiment dataset into the proposed paradigm

6.3 Data Analysis

Within the dataset, the periods of time between the Leave Home and the Enter Home activities is a period when no one is at home. So, a new activity was added, noted as outside. The system can use this information to enhance the energy consumption of the sensory infrastructure, as the installed sensors in the location might be deactivated while there is no one in it, except the door sensors, therefore reducing energy consumption without losing prediction accuracy. Also, there are spaces between the end of each activity and the beginning of the next one. These are not indicating any activities involved within the environment, so it was noted as "no activity" in the dataset.

The dataset needed to be analysed in order to be fitted and fed into the program. When analysing data a huge imbalance was noticed among the noted activities as in figure 6.2. Especially, the "no activity" class which has a dominance upon all other classes in the dataset that leads to misprediction (though the accuracy

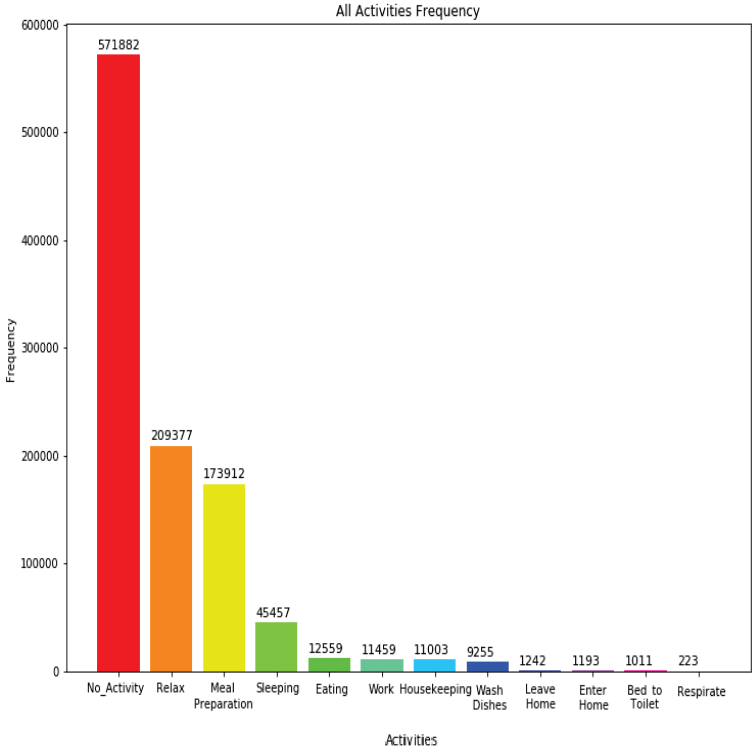


Figure 6.2 : Dataset Activies Frequencies showing lable inbalance

percentage would still be high because the classes with high occurrence would be predicted correctly). The Neural Network would tend to predict classes with higher existence in the data upon all other classes with little presence in the data whenever it is uncertain about the decision. This leads to the resolution of performing the model into two steps: separating the "no activity" class from all other classes (noted as "activity" class), then inferring one of the other classes as a second step. In such a case, an addition to the assumption were made that there are two levels of aggregators, the first one is used to classify the activities into "no activity" or "activity" shown in figure 6.3, while the other one predicts all other classes from the "activity" class. Then the assumed topology would be as shown in figure 6.1 where the L1.1 aggregator would classify the activity and no activity while L1.2 would predict other classes.

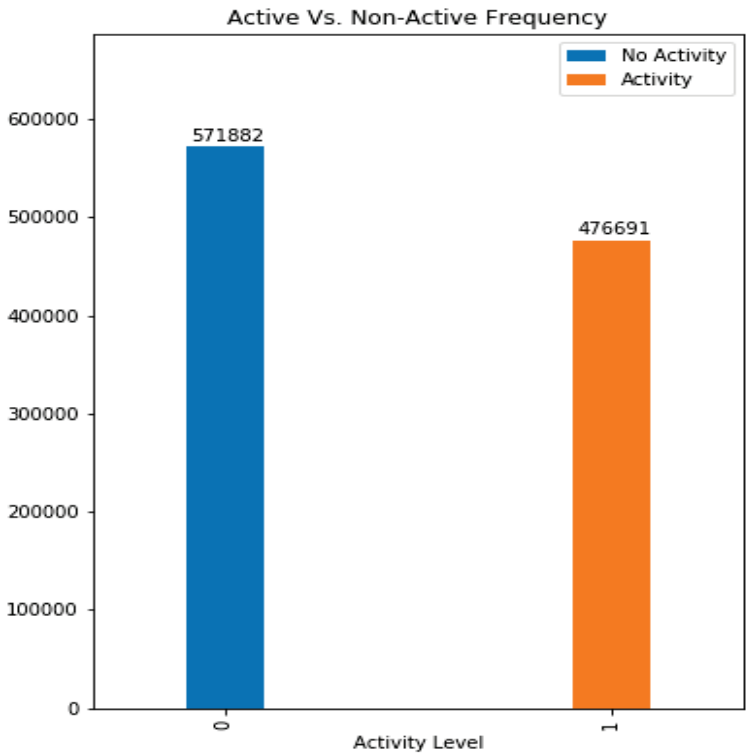


Figure 6.3 : dataset Active vs NonActive lables

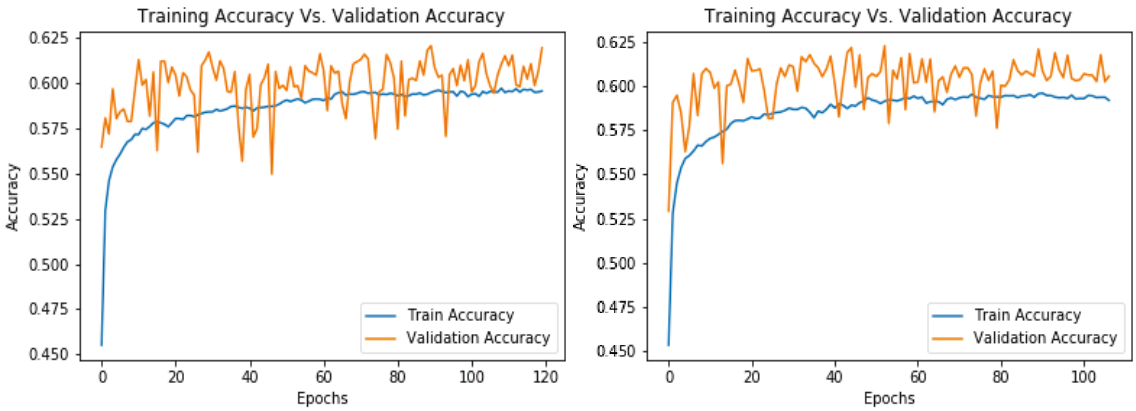
Another crucial part of data analysis is data scaling and normalization. The data has three types of sensors, motion sensors with "on" and "off" readings, door closure sensors with "open" and "close" status, and temperature sensors with values in degrees Celsius. For a model to well trained, the data fed into it has to scale within the same range and can be described in a single distribution [174]. For this reason, the data was normalised using data mapping techniques to scale sensors readings into a common range.

After performing the first phase of predicting the activity or no activity classes, the activity data need to be fixed in order to resolve the class imbalance among the activities. In deep neural networks, this is done by data augmentation which is the technique of producing artificial information so as to decrease the classifier's variance with the aim to decrease errors [175]. By applying data augmentation procedure

on the dataset, all the classes would have the same number of records within the dataset. In this case, the decision about the class in the test phase of the cross validation process would be fair [176]. Now the data is ready to be partitioned and its patterns is to be learned by the neural networks within the proposed blueprint.

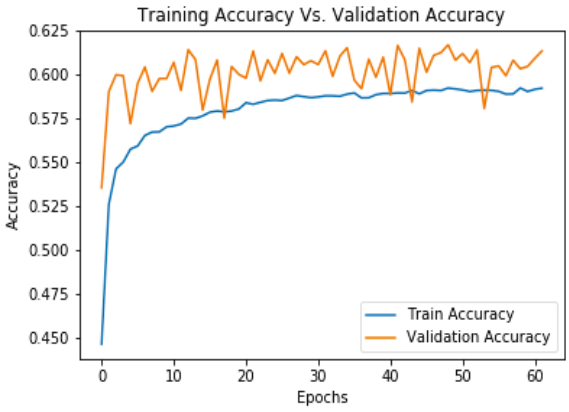
Initially, the data was separated into ten parts and the BNN was implemented upon each of these parts. The results from cross-validation show a low accuracy levels for each of the partitions as illustrated in figure 6.4. Nearly the same level of accuracy had been shown when executing a consensus ensembles (Bagging (Bootstrap aggregating), Weighted Average and Majority Vote) to the outcomes of the BNNs as shown in figure 6.5. To show how the data with one feature criterion can perform, a cross-validation was implemented using Machine Learning Random Forest and Logistic Regression classifiers. A comparison among them and the average of employing Bayesian Neural Network over the ten data chunks as well as their consensus ensemble is shown in figure 6.6.

A thorough investigation was made to discover the reason behind these results and a conclusion was made that the dataset itself needed to be modified. The original dataset has only one feature (sensors' readings) based on time series, thus more features needed to be extracted from the data in order to make it more distinguishable for the neural network to learn. Feature engineering is the technique utilizing domain knowledge of the dataset in order to transform raw data into features in a way which makes better representation to the decision making models, resulting in enhanced system accuracy [177]. This technique, feature engineering, was applied to the data then the dataset was divided into four training sets in addition to one extra set for testing by comparing the outcome of each technique.

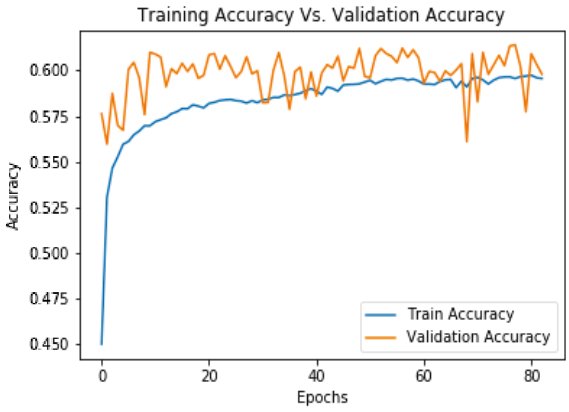


(a) 1st Chunk

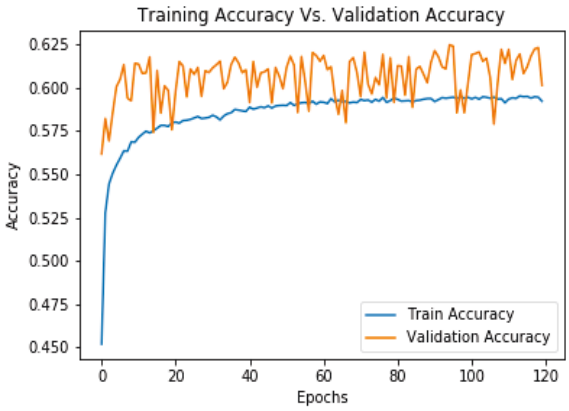
(b) 2nd Chunk



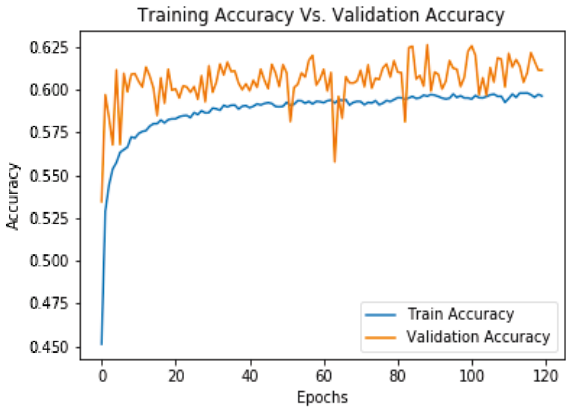
(c) 3rd Chunk



(d) 4th Chunk

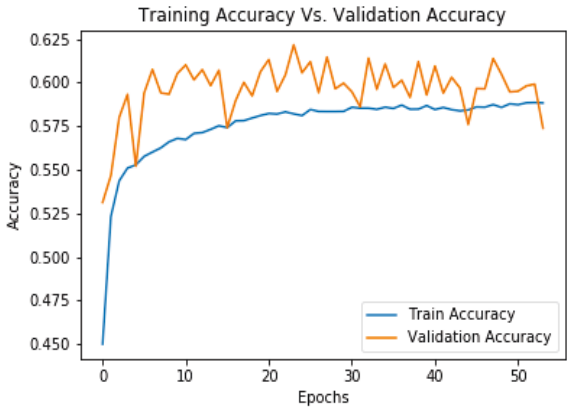


(e) 5th Chunk

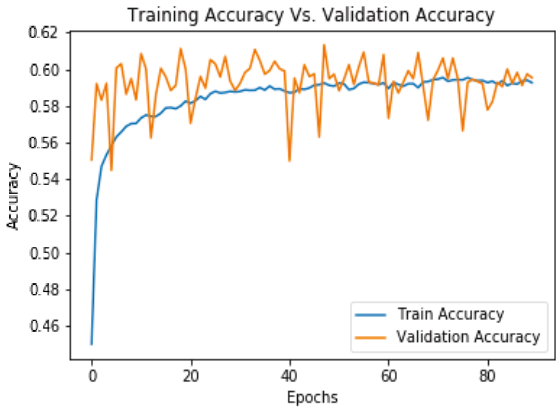


(f) 6th Chunk

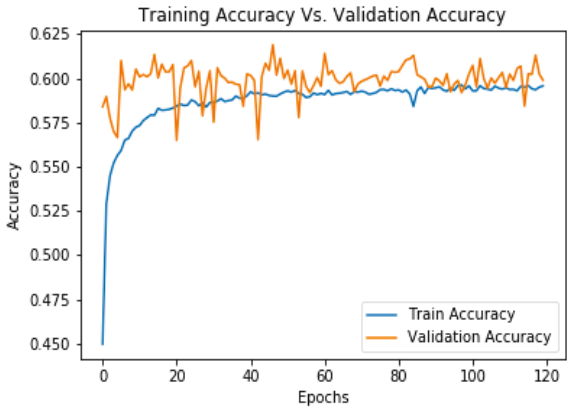
Figure 6.4 : Training accuracy vs. validation accuracy for the dataset parts after applying the second model befor performing extensive data analysis. a-f represent the first six parts of the data (participants).



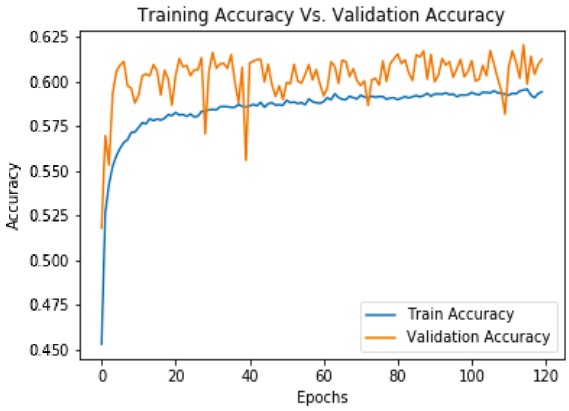
(a) 7th Chunk



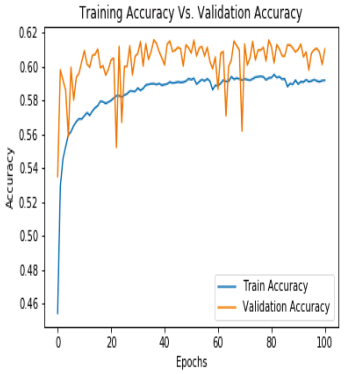
(b) 8th Chunk



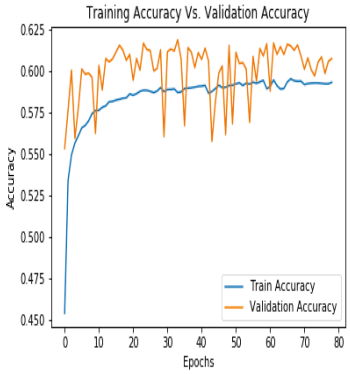
(c) 9th Chunk



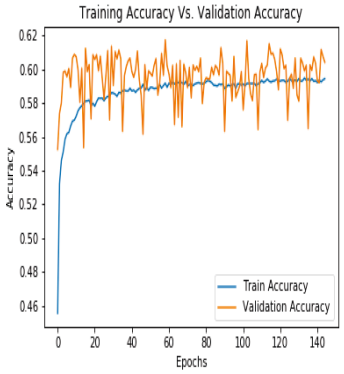
(d) 10th Chunk



(e) Bagging ensemble



(f) Weighted Average



(g) Majority Vote

Figure 6.5 : Accuracy for the last four parts of the dataset (participants) after applying the second model befor performing extensive data analysis (a-d). e-g are the ensembles Training vs. validation accuracies after applying the third model befor performing extensive data analysis (feature engineering)

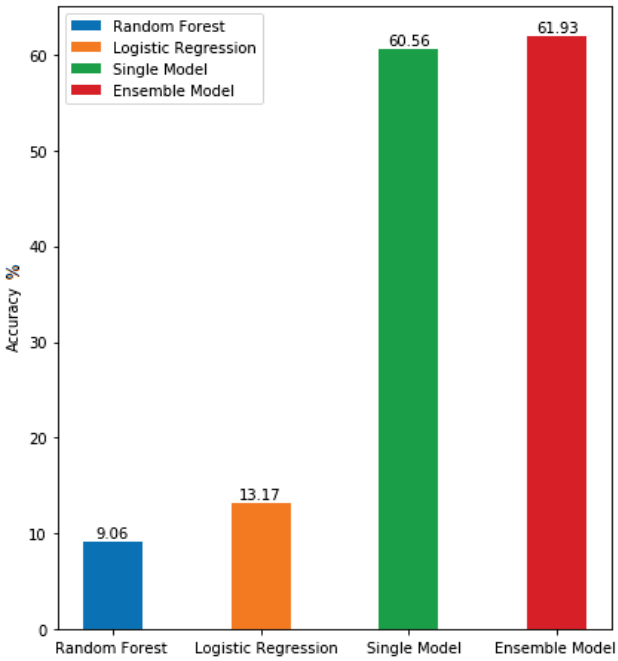


Figure 6.6 : comparison about prediction accuracy levels when cross-validating utilising Random forest, Logistic Regression and taking average of 10 data chuncks Bayesian Neural Network and when applying Bagging ensemble to the 10 BNN outputs for the data having one feature criterion

6.4 Model Procedure

The process is cascaded into two phases, predicting "no activity"/ "activity" phase and inferring the activity class in the other. One classification model was built to perform the first phase, the Long Short-Term Memory (LSTM). However, for acquiring the functionality of the second phase, two models were set up, Bayesian Neural Network for the individual data partitions (participants) and an ensemble process used for consensus.

LSTM is a time series based artificial Recurrent Neural Network (RNN) architecture in deep learning field that can process entire sequences of data. It has a feedback connection which makes it capable of simulating any computer algorithm

logic. The LSTM classification model used for separating the no activity from the activity classes uses the He initializer to initialise the weight matrix. Deep learning neural networks are probable to rapidly overfit a training dataset with few instances. In order to reduce overfitting and enhance generalization error in all kinds of deep neural networks, a dropout model is used. Dropout is a single model that can simulate a large number of various network architectures through arbitrarily dropping out nodes while training is in progress. This offers a very computationally inexpensive and remarkably efficient regularization scheme. For this experiment, the system is implemented in dropout layers as in [178]. Three dropout layers are used as well as three other batch normalization layers. Batch normalization is used to normalise the input layer by scaling and regulating the activations. It is beneficial so as to speed up learning, increase the stability of a neural network and permit the use of fewer dropouts in order not to lose a lot of information [174]. Additionally, two dense softmax units were employed in the model. Softmax is an activation function that is used to map the non-normalised output produced by a neuron to a probability distribution over predicted output classes. Softmax functionality is to determine the probabilities of every class over all potential target classes. Then, those probabilities would help in choosing the target class to the given inputs [179] [180]. In order to optimise the learning rate and update network weights iterative based in the trained dataset, Adam adaptive learning rate optimization algorithm is used with decaying learning rate of $1e^{-6}$ [181]. The selection of optimization procedure for deep learning technique can mean the difference between good outcomes in minutes, hours, or days. Finally, to avoid overfitting, an early stopping regularization is utilised. Such method updates the neural network to make it better fit the training data at every epoch [182]. In the assumed topology, this model is to be executed in L1.1 aggregator. The process of implementing this model is shown in the tensor board visualization figure 6.7. The same model settings were applied to two classifiers

perform employing machine learning, Random Forest and Logistic Regression, to illustrate the effectiveness of the LSTM within the design by comparing the outcome of each technique.

The second model, classifying the activity classes, is to take place in L1.2 aggregator nodes according to the assumed design. Ten (then finally four) identical models were trained on ten (then four) randomly split datasets (each of which is partitioned from the whole dataset).Bayesian deep learning algorithm was applied to each of the models. Each model has two Dense layers with rectified linear activation function and He uniform initialization and is composed of 200 and 220 units respectively. In addition, each model has one dense softmax layer with twelve units, two Dropout layers with 0.2 probability, and a total number of hidden layers equals to three. The models employ Adam optimizer with constant learning rate of 0.0005.

The third model performed to accomplish the objective of the project is the consensus model utilizing ensemble procedures. As the focus for this experiment is to appraise the uncertainty of the model, Bootstrap Aggregating (Bagging) ensemble technique is used. Bagging is a parallel ensemble i.e. it require each model to be built independently from the other and it is aimed to decrease variance. It decreases the prediction variance through generating extra training data from the original dataset utilizing combinations with repetitions to generate multi-sets of the same size as the original dataset. As measuring uncertainty includes the variance of the softmax probabilities and variation ratios [183], Bagging seems to be the best fit for the design. Two other well-known ensemble methods were implemented (Majority Vote and Weighted Average) to provide variety of techniques and confidence in the outcome.

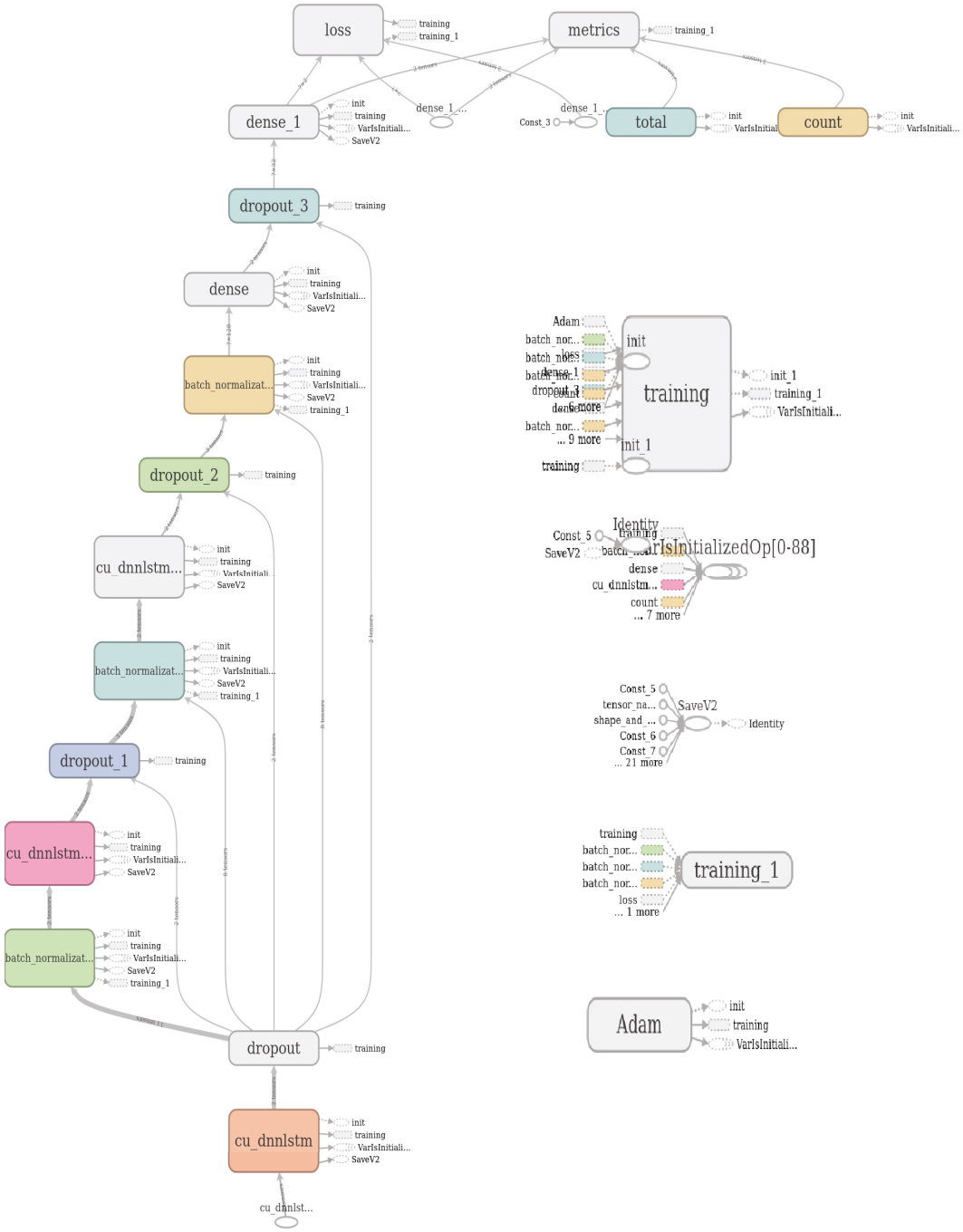


Figure 6.7 : Design procedure showing the the processes done to attain Bayesian deep learning

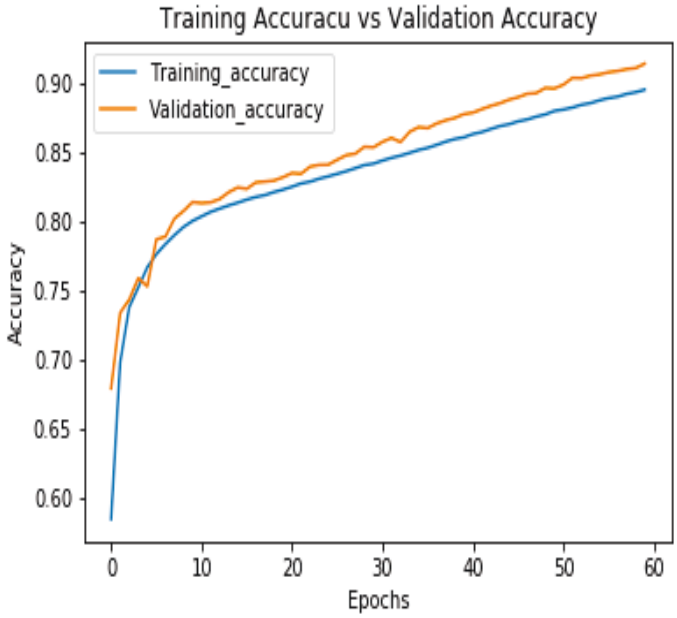


Figure 6.8 : prediction accuracy of the "no activity" or "activity" with the increas of epochs count

6.5 Outcomes

When performing the first classification model, involving the prediction of active and non-active labels, the accuracy of the prediction output is raising as the prediction model is learning(see figure 6.8). As the number of training epochs increases, the accuracy of model improves until it reaches 92.6% on the final epoch. Note that this model runs in 60 epochs. Additionally, the accuracy tests of the LSTM, Random Forest and Logistic Regression algorithms were performed as shown in figure 6.9.

When performing the second model, the uncertainty of each procedure (applied at one of the data partitions) at each epoch were calculated. Then after training every procedure, the uncertainty was recorded for that particular procedure. Also, during that operation, the Bagging ensemble method was run aggregating the out-

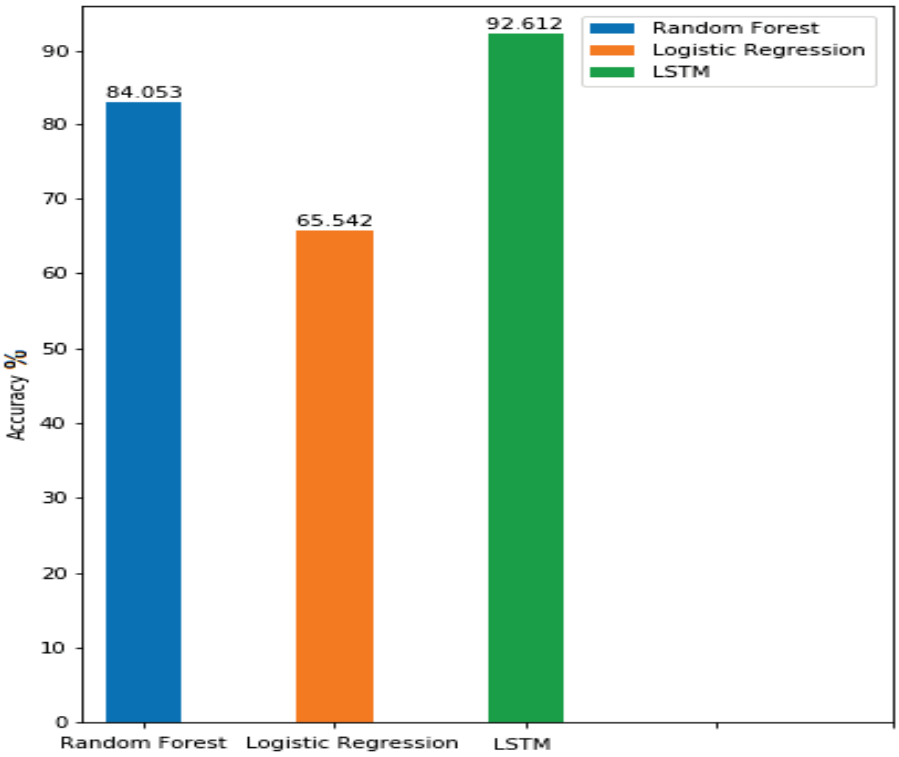


Figure 6.9 : The accuracy of the first stage of distinguishing between "Activities" classes and the "NO Activity" class

come of the models one by one. This means that the first ensemble method was run on the output of the first and second procedures, after that it was performed upon the first three procedures, then on the first four then kept doing so till it aggregates the ten procedures. While doing so, the certainty of each ensemble were measured. The uncertainty measures for each of the procedure (single ones) and the ensembles are shown in figure 6.10.

After dealing with data analysis and partitioning the data into four partitions (each one represents data of a sub-network headed by a participant node), each single partition was trained employing the second model to infer decisions about activities' classes. Afterward, decisions from the four participants were entered in the third model so as to proceed with the consensus procedure. The accuracies of each class at

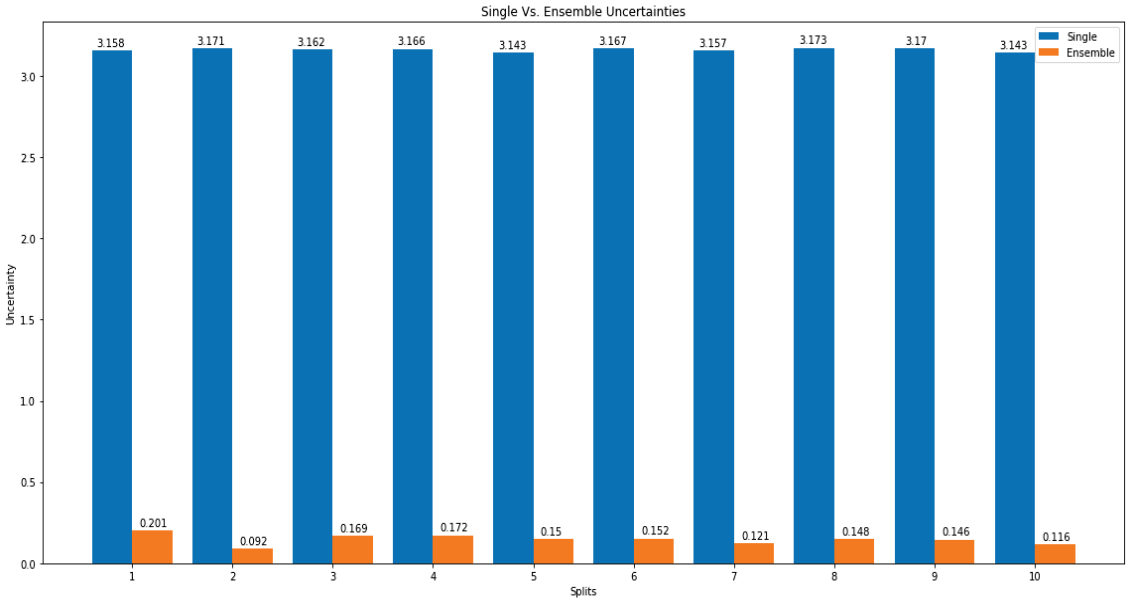
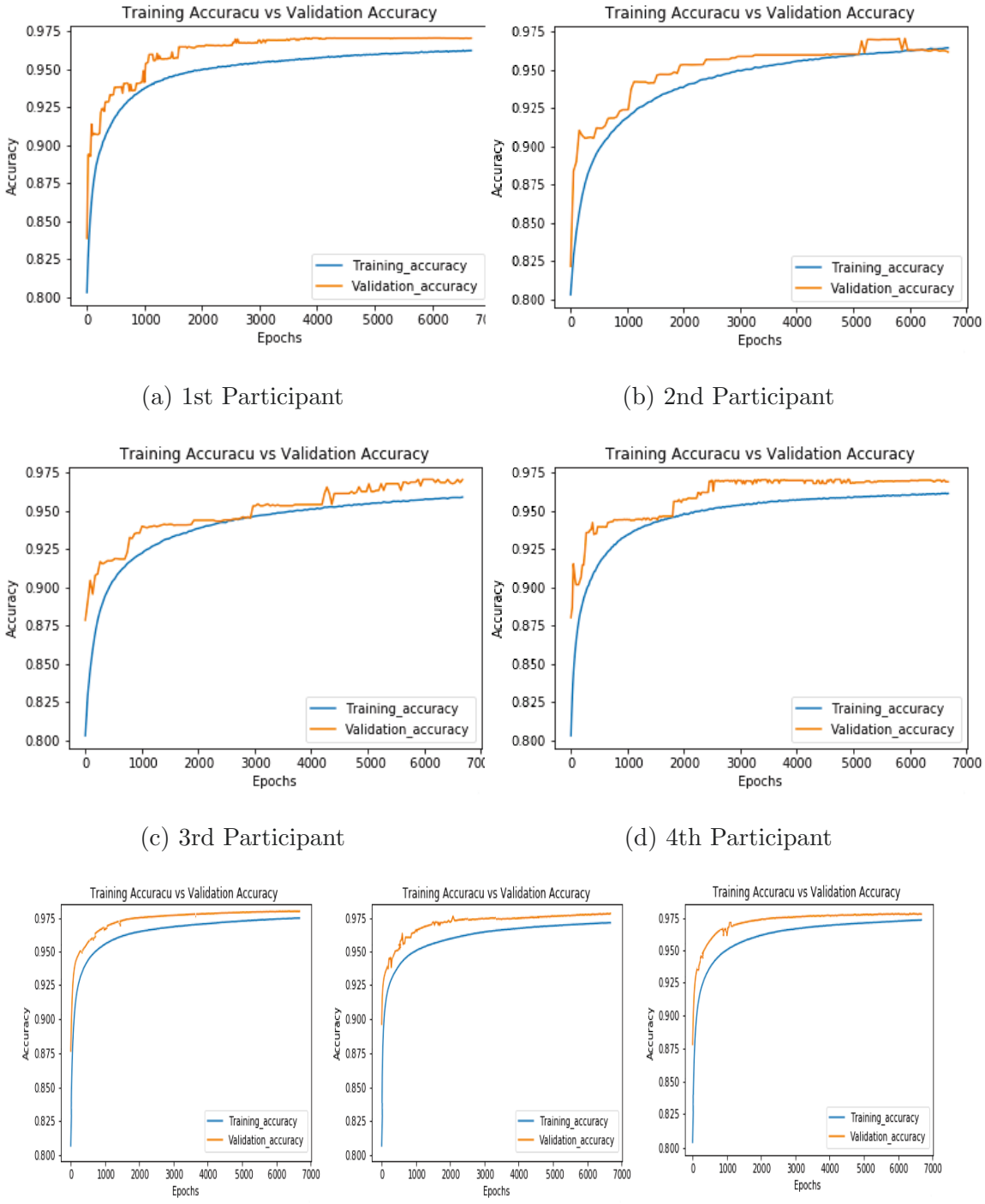


Figure 6.10 : uncertainity levels

each step were calculated to test the efficiency of the system. Figure 6.11 illustrates the training vs. validation accuracies after applying the extensive data analysis. Also, figures 6.12 and 6.13 show the confusion matrix with the accuracies of all activities' classes as a result of applying the second model to each participant's dataset and performing three consensus algorithms respectively. The consensus algorithms used are three ensembles: the Majority Vote, Weighted Average, and Bagging ensembles. The accuracies of all classes and the overall accuracy for each participant's BNN model as well as for every consensus model are shown in figure 6.14. The last column of this figure shows the final overall accuracy for applying each of the algorithms to the data. It shows that the precision of each BNN model is around 97.5% while it rise for each of the consensus models to be approximately 97.63% for Majority Vote ensemble, 97.7% for Weighted Average ensemble, and 98.1% for the Bagging ensemble.

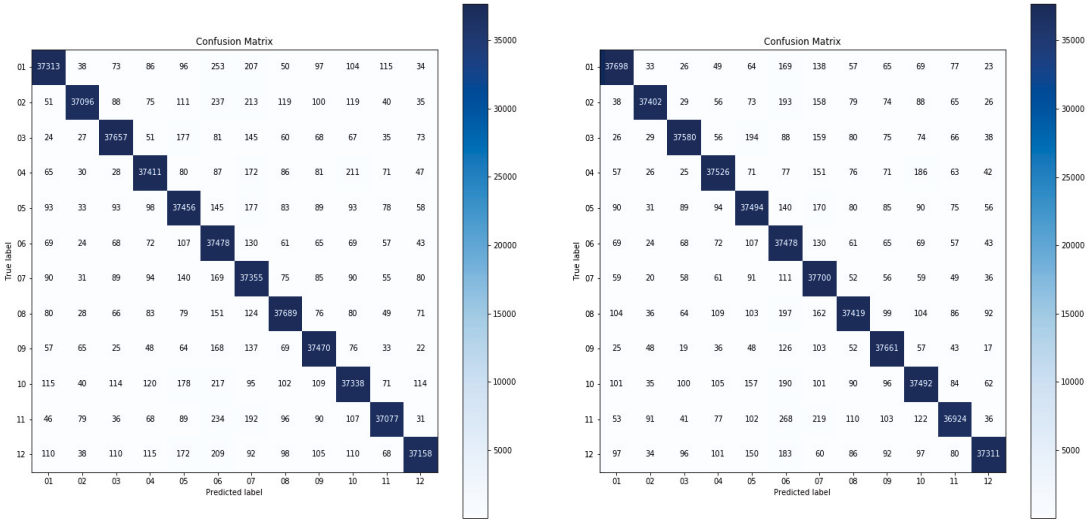


(a) 1st Participant (b) 2nd Participant

(c) 3rd Participant (d) 4th Participant

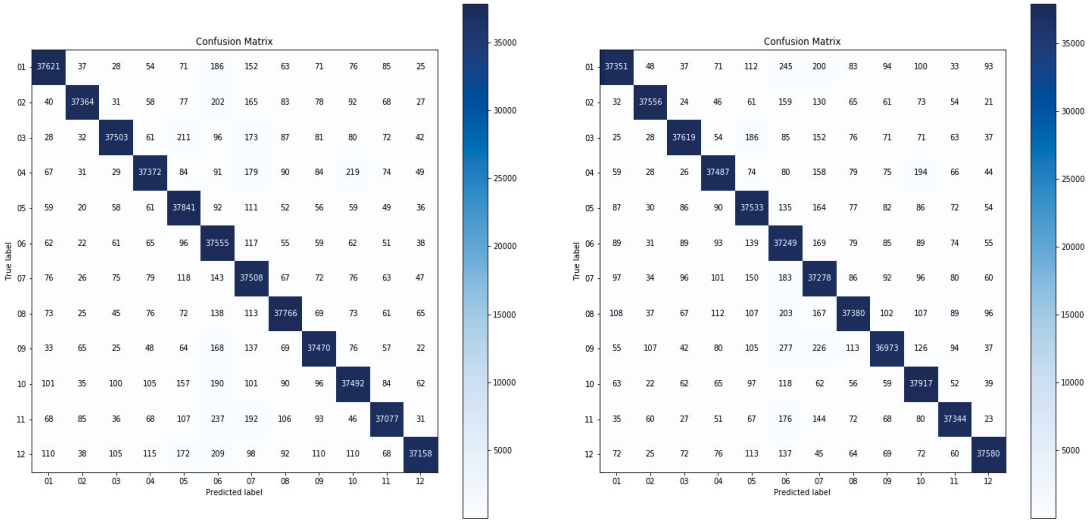
(e) Bagging ensemble (f) Weighted Average (g) Majority Vote

Figure 6.11 : The accuracy of participant's classifier for each of the four data subsets after applying the data analysis first and then generating the second model (a-d); e-g are the ensembles accuricies after applying the feature engineering to the dataset.



(a) Participant 1

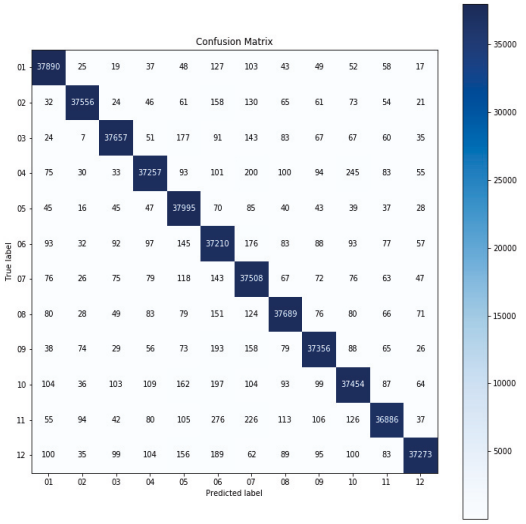
(b) Participant 2



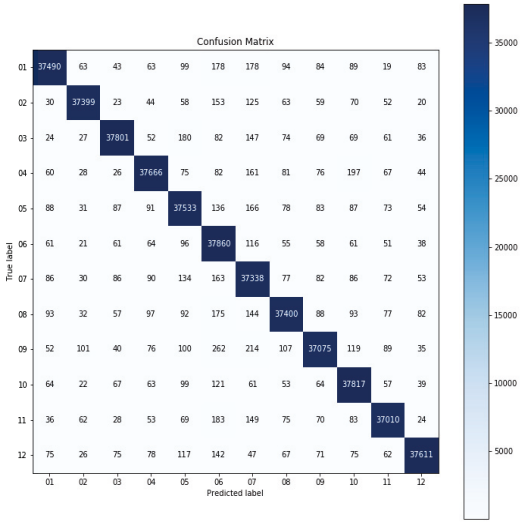
(c) Participant 3

(d) Participant 4

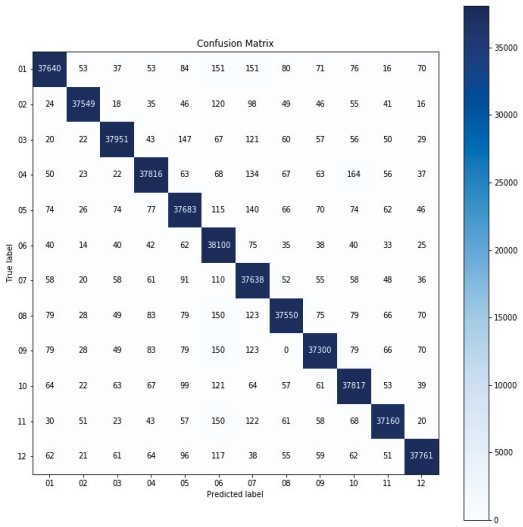
Figure 6.12 : Confusion matrix plot showing the precision of the 12 activities' result by training each of the four participants after data analysis data analysis and performing the BNN on partitions' data



(a) Majority Vote



(b) Weighted Average



(c) Bagging Ensemble

Figure 6.13 : Confusion matrix plot showing the precision of the 12 activities' result by performing three consensus models through three different ensemble algorithms

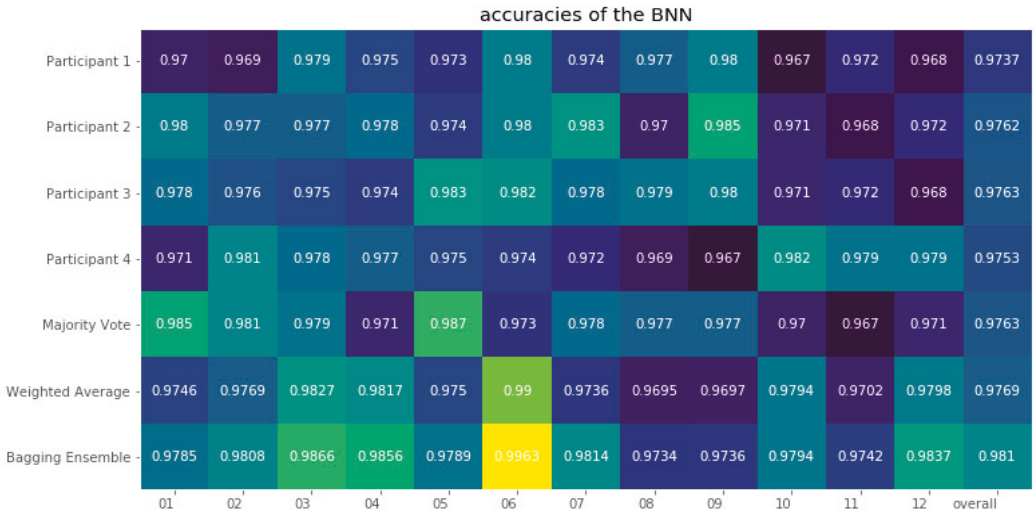


Figure 6.14 : Accuracies of all classes as well as the overall accuracies for the participants and ensemble algorithms after data processing and applying the BNN and consensus models

6.6 Evaluation

In this experiment, Bayesian Neural Networks were performed upon a smart home data generated by sensors installed in fixed spots of that setting. In order to examine the effectiveness of applying the proposed blueprint onto the produced dataset, the accuracies and uncertainty levels were tested for all classes in each of the models. As mentioned in 6.3 , the original data was not good enough to be trained by machine learning methods because it contains very few features and a huge class imbalance even though the dataset itself has a lot of records. Analysing the data and prepare it to suit the system was a very challenging task. The data

went into some processes involving: data cleaning, data mapping, and data scaling. At this point, the first model was trained by a part of the data (training) to infer the activity or no activity classes (employing the test data partition). The results shows an acceptable range of accuracies (reached to 92.6%) when using the LSTM depending only on the sensors readings and time stamp as illustrated in figures 6.8 and 6.9. This step demonstrates the effectiveness of the model when dealing with two classes.

After this step, the "activity" data was divided into ten partitions. After applying that form of data into the second and third models, it produced low accuracy as in the figures 6.4, 6.5 and 6.6 due to lack of enough features to enable the model to differentiate between several classes. What was surprising is even at that low percentage of the ensemble accuracy (about 62%), the uncertainty level changed vastly as the ensemble was used. The uncertainty levels after implementing the ensemble went down from approximately 3.16% to 0.116% in the last ensemble combining ten single models as shown in figure 6.10. Thats because the ensemble operation has a property of decreasing the variance of the overall model [184].

The same dataset was investigated by Yala, Fergani and Fleury(2015)[185]. They used the Support Vector Machine as well as three other proposed models to classify the classes. They also made a comparison between the activity and no activity classes and implemented two experiments, the first one without including the no activity class whereas the second one was with that class being included. The results showed an accuracy measures of around 87% when not including the no activity class, while it was around 67% when including it.

With the low accuracy range for all models, more data analysis was required to train the models on data which has more features. More data analysis techniques were applied to the data including: Bootstrapping, Factor analysis, Feature engi-

neering, data augmentation, class imbalance, label encoding for categorical data. Then the data was partitioned to four training datasets and one test dataset. The data was ready to be applied into the second model (then the results will be fed to the third model) and the inference results show a very high precision outcome as illustrated in figures 6.11, 6.12, 6.13 and 6.14. The improvement in the prediction accuracy (from the levels shown in figures 6.4 and 6.5 to that in figure 6.11), allows more credibility of observation and events detection for models when predicting activities. It is important because when precision is more accurate, it contribute to much more reliable models of activity prediction. This demonstration of the outcomes indicates the need for extensive data analysis when dealing with data with very few features. Additionally, the experiment exhibited some observations about the consensus operation. Consensus models provide stable, solid and overall better performance than using single models. It makes the models more prone to overfitting with a huge decrease in the uncertainty. The ensemble models decrease the variance of the overall model as well. Finally, even if the dataset changed, the consensus method has a stable level of accuracy even when one or more of the individual procedures produces low accuracy.

6.7 Conclusion

The goal of this third experiment is to investigate the efficiency of the proposed paradigm when applied on a data collected in a smart environment settings. The setting has fixed sensors producing motion and temperature measures from smart home to monitor the activities occurring in this setting. The aim of the project is to use different techniques in performing decision making and execute consensus models to investigate the effectiveness of the consensus operation upon different processes. In this experiment, the Bayesian Neural Networks were used to perform the cross validation process which would lead to making decisions about the activities

involved within the network settings. Three models were designed and implemented to fit the data generated in that particular setting into the design or the proposed paradigm. For that, the paradigm is assumed to have two aggregators' nodes each with different responsibilities and performed on a different forms of the same data. The first aggregator was assumed to perform the first model to classify between two classes, activity and no activity, and was performed on data applied to data cleaning, mapping and scaling. This step demonstrates the effectiveness of the model when dealing with two classes. The second aggregator (the participant) was assumed to perform the second model to classify between twelve classes. However, with this step there is a requirement for extensive data analysis techniques to be applied when dealing with data with very few features. The third model, the consensus model, was assumed to be performed by multiple participants, each perform the second model and get its outcome, then operate on all the resultant outcomes of the other participants to get the consensus decision. The output results demonstrated a very good gain in decreasing the uncertainty levels when utilizing the consensus methods even when the data was not good enough in terms of accuracy. It also showed that the consensus operation provides stable, solid and overall better performance than the individual models. Additionally, the consensus models are more resistant to overfitting and have better accuracies. This Experiment also illustrates that after implementing data analysis upon the data then applying the BNN algorithm, the outcome demonstrates a superiority over some other methods implemented by other research in [185] over the same dataset.

Chapter 7

Conclusion

7.1 Introduction

The recent expansion of the Internet of Things has created a tremendous amount of services, sophisticated data and shared resources that require automated systems to manage them. Big Data issues have evolved as a result of IoT which needs to be mitigated as the data is generated near its source. Fog Computing is the architecture which utilises the network edge devices to perform local computation, communication and storage. This research proposes a Consensus-base data management paradigm within the Fog Computing of the Internet of Things. The aim of the proposed paradigm is to manage the generation and transmission of IoT Fog data through consensus decision making practices. The paradigm also aims to recognise the events and activities which occur within the network environment. The design of the proposed paradigm encompasses distributed and in-network hybrid methodology. The design involves adaptive data aggregation management and autonomous consensus decision making. In order to investigate the effectiveness of the proposed paradigm upon different processes, three different experiments were carried out employing various techniques to perform decision making in order to execute the Consensus-based models. The design was tested to detect events based on environmental data which comes from environmental and air quality sensors in the first experiment. In the second experiment, human activities based on mobile movements and position sensors were investigated. Additionally, the third experiment examines smart home activities based on fixed movement and air quality

sensors. The execution of the three experiments shows that the paradigm is very suitable for many IoT environments and can be adapted into different scenarios. However, the data at each experiment's setting needs to be analysed first in order to fit it into the design of the proposed paradigm, especially when dealing with data with very few features. The output results of the experiments demonstrate that the consensus operation provides stability, solid efficiency, more resistance to overfitting and overall better performance and precision than the individual models. It also shows a very good gain in decreasing uncertainty levels when utilizing the consensus methods even when the data was not good enough in terms of accuracy. The first experiment demonstrates that the proposed Likelihood Multiplication algorithm using Bayesian machine learning can attain markedly better performance than current solutions in practical cases. In addition, the accuracy outcome of the second and third experiments demonstrates a superiority over some other methods implemented by other research over the same datasets. Additionally, the second experiment illustrates that the consensus decision obtained by aggregating decisions extracted from training subsets of data were able to surpass the decision accuracy obtained from training the overall data. This observation gives an indication that Fog Computing processes can compete with Cloud Computing in terms of accuracy, with the added advantage of locality.

7.2 Future Work

The scope of this research topic brings many directions for future work. One of these directions is that reinforcement learning methods can be applied to the nodes after training some models in a server or in more computationally powerful node. The trained models can be reinforced into the participants nodes to make them more intelligent and more capable to detect the changes in operational modes or activities within the network. Also, the consensus method can advance this process by an

exchange of the learned outcomes among the participants in that each participant would have the overall knowledge to make better decisions. Another future work direction is to expand the communication level between aggregators so that each participant, as the cluster head of its own network branch, can communicate with any sub-aggregator of another network branch to get any required data. The expansion can also involve performing the Consensus operation between the participant and its own sub-aggregators. This will help the participant in gaining more knowledge by training more data which will reflect positively on the DM performance for that participant and the network as a whole. In addition to these future directions, when using smart sensors which have the ability to perform sophisticated computations and maintain intelligent models within itself, a Consensus scheme can be applied among those sensors. The purpose of this Consensus operation is to decide on matters that can increase the efficiency of data generation and aggregation. An example of this is deciding what and when to sense and send as well as what to do when redundancy occurs.

Bibliography

[1] Alkiviadis Tsitsigkos, Fariborz Entezami, Tipu Ramrekha, Christos Politis, and Emmanouil Panaousis. A case study of Internet of Things (IoT) based on Wireless Sensor Networks (WSNs) and Smartphone (iPhone). 2012.

[2] Sarder Fakhrul Abedin, Golam Rabiul Alam, Nguyen H Tran, and Choong Seon Hong. A Fog based System Model for Cooperative IoT Node Pairing using Matching Theory. pages 309–314, 2015.

[3] Simone Cirani, Gianluigi Ferrari, Nicola Iotti, Marco Picone, Guglielmo Srl, and Reggio Emilia. The IoT Hub : a Fog Node for Seamless Management of Heterogeneous Connected Smart Objects. 2015.

[4] Vangelis Gazis, Alessandro Leonardi, and Kostas Mathioudakis. Components of Fog Computing in an Industrial Internet of Things Context. 2015.

[5] Jiazi Yi, Thomas Clausen, and Ulrich Herberg. Depth-First Forwarding for Unreliable Networks : Extensions and Applications. *Internet of Things Journal, IEEE*, 2(3):199–209, 2015.

[6] Mung Chiang. Fog networking: An overview on research opportunities. *arXiv preprint arXiv:1601.00835*, 2016.

[7] Fabio Giust, Xavier Costa-Perez, and Alex Reznik. Multi-access edge computing: An overview of etsi mec isg. *IEEE 5G Tech Focus*, 1(4), 2017.

[8] J a Stankovic. Research Directions for the Internet of Things. *Internet of Things Journal, IEEE*, 1(1):3–9, 2014.

[9] D. Chinh Hoang, R. Kumar, and S. Kumar Panda. Optimal data aggregation tree in wireless sensor networks based on intelligent water drops algorithm. *IET Wireless Sensor Systems*, 2(3):282, 2012.

[10] Huifang Chen, Hiroshi Mineno, and Tadanori Mizuno. Adaptive data aggregation scheme in clustered wireless sensor networks. *Computer Communications*, 31(15):3579–3585, 2008.

[11] S. Bohm, C. Engelmann, and S.L. Scott. Aggregation of Real-Time System Monitoring Data for Analyzing Large-Scale Parallel and Distributed Computing Environments. *High Performance Computing and Communications (HPCC), 2010 12th IEEE International Conference on*, pages 72–78, 2010.

[12] Chiwoo Park Chiwoo Park, Yu Ding Yu Ding, and Eunshin Byon Eunshin Byon. Collaborative data reduction for energy efficient sensor networks. *2008 IEEE International Conference on Automation Science and Engineering*, pages 442–447, 2008.

[13] Wenzhong Guo, Wei Hong, Bin Zhang, Yuzhong Chen, and Naixue Xiong. Reliable Adaptive Data Aggregation Route Strategy for a Trade-off between Energy and Lifetime in WSNs. pages 16972–16993, 2014.

[14] Benny Applebaum, Haakon Ringberg, Michael J. Freedman, Matthew Caesar, and Jennifer Rexford. Collaborative, privacy-preserving data aggregation at scale. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6205 LNCS:56–74, 2010.

[15] Haiyong Bao and Rongxing Lu. A New Differentially Private Data Aggregation With Fault Tolerance for Smart Grid Communications. *Internet of Things Journal, IEEE*, 2(3):248–258, 2015.

[16] Sergey V Muravyov and Liudmila I Khudonogova. Multisensor accuracy enhancement on the base of interval voting in form of preference aggregation in WSN for ecological monitoring. pages 293–297, 2015.

[17] Ryszard Klempous. Byzantine Algorithms in Wireless Sensors. 00, 2006.

[18] Natalya Fedotova and Luca Veltri. Byzantine Generals Problem in the Light of P2P Computing. *3rd Annual Conference on Mobile and Ubiquitous Systems*, pages 1–5, 2006.

[19] Mohammed a. AlZain, Ben Soh, and Eric Pardede. A Byzantine Fault Tolerance Model for a Multi-cloud Computing. *IEEE International Conference on Computational Science and Engineering (CSE)*, pages 130–137, 2013.

[20] Yilei Zhang, Zibin Zheng, and Michael R. Lyu. BFTCloud: A Byzantine Fault Tolerance framework for voluntary-resource cloud computing. *Proceedings - 2011 IEEE 4th International Conference on Cloud Computing, CLOUD 2011*, pages 444–451, 2011.

[21] Mahdi Ben Alaya, Salma Matoussi, Thierry Monteil, and Khalil Drira. Autonomic computing system for self-management of machine-to-machine networks. *Proceedings of the 2012 international workshop on Self-aware internet of things - Self-IoT '12*, page 25, 2012.

[22] Sudip Misra and Subhadeep Sarkar. Theoretical modelling of fog computing: a green computing paradigm to support IoT applications. *IET Networks*, 5(2):23–29, 2016.

[23] Zhiwei Yan, Sherali Zeadally, and Yong-jin Park. A Novel Vehicular Information Network Architecture Based on Named Data Networking. *IEEE Internet of Things Journal*, 1(6):525–532, 2014.

[24] Qihui Wu, Guoru Ding, Yuhua Xu, Shuo Feng, Zhiyong Du, Jinlong Wang, and Keping Long. Cognitive Internet of Things: A New Paradigm beyond Connection. *IEEE Internet of Things Journal*, 1(2):129 – 143, 2014.

[25] Firas Al-doghman, Zenon Chaczko, and Jianming Jiang. A Review of Aggregation Algorithms for the Internet of Things. In *25th International Conference On Systems Engineering*, 2017.

[26] Daniel Peralta, Sara del Río, Sergio Ramírez-Gallego, Isaac Triguero, Jose M Benitez, and Francisco Herrera. Evolutionary feature selection for big data classification: A mapreduce approach. *Mathematical Problems in Engineering*, 2015, 2015.

[27] Charith Perera, Rajiv Ranjan, Lizhe Wang, Samee U Khan, and Albert Y Zomaya. Big data privacy in the internet of things era. *IT Professional*, 17(3):32–39, 2015.

[28] Rabindra K Barik, Harishchandra Dubey, Chinmaya Misra, Debanjan Borthakur, Nicholas Constant, Sapana Ashok Sasane, Rakesh K Lenka, Bhabani Shankar Prasad Mishra, Himansu Das, and Kunal Mankodiya. Fog assisted cloud computing in era of big data and internet-of-things: systems, architectures, and applications. In *Cloud computing for optimization: foundations, applications, and challenges*, pages 367–394. Springer, 2018.

[29] Emre Yigitoglu. *Privacy-aware big data systems and services in the internet of things era*. PhD thesis, Georgia Institute of Technology, 2018.

[30] Nilanjan Dey, Aboul Ella Hassanien, Chintan Bhatt, Amira S Ashour, and Suresh Chandra Satapathy. *Internet of things and big data analytics toward next-generation intelligence*. Springer, 2018.

[31] Simone Cirani, Gianluigi Ferrari, Mirko Mancin, and Marco Picone. Virtual replication of iot hubs in the cloud: A flexible approach to smart object management. *Journal of Sensor and Actuator Networks*, 7(2):16, 2018.

[32] L Arockiam and Priya Mary. Issues and challenges hampering the evolution of iot big data analytics. *International Journal of Applied Engineering Research*, 10, 06 2018.

[33] Syed Husain, Andreas Kunz, Athul Prasad, Konstantinos Samdanis, and Jae-Seung Song. Mobile edge computing with network resource slicing for internet-of-things. In *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, pages 1–6. IEEE, 2018.

[34] Pawani Porambage, Jude Okwuibe, Madhusanka Liyanage, Mika Ylianttila, and Tarik Taleb. Survey on multi-access edge computing for internet of things realization. *IEEE Communications Surveys & Tutorials*, 20(4):2961–2991, 2018.

[35] Raja Lavanya. Fog computing and its role in the internet of things. In *Advancing Consumer-Centric Fog Computing Architectures*, pages 63–71. IGI Global, 2019.

[36] Muhammad Saad. Fog computing and its role in the internet of things: Concept, security and privacy issues. *International Journal of Computer Applications*, 975:8887, 2018.

[37] R. Al-Doghman, F., Chaczko, Z., Ajayan, A.R. and Klempous. A review on Fog Computing technology. In *2016 IEEE International Conference on Systems, Man, and Cybernetics' SMC*. 2016 IEEE International Conference on, pp. 001525-001530., 2016.

[38] Mohammad Aazam, Marc St-Hilaire, Chung-Horng Lung, Ioannis Lambadaris, and Eui-Nam Huh. Iot resource estimation challenges and modeling in fog. In *Fog Computing in the Internet of Things*, pages 17–31. Springer, 2018.

[39] D Madhu Kumar, Aindrila Ghosh, et al. Resource efficient routing in internet of things: Concept, challenges, and future directions. *International Journal of Computing and Digital Systems*, 8(6), 2019.

[40] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

[41] Rodrigo Roman, Javier Lopez, and Masahiro Mambo. Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges. *Future Generation Computer Systems*, 78:680–698, 2018.

[42] Tarik Taleb, Konstantinos Samdanis, Badr Mada, Hannu Flinck, Sunny Dutta, and Dario Sabella. On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration. *IEEE Communications Surveys & Tutorials*, 19(3):1657–1681, 2017.

[43] Righa Tandon and PK Gupta. Optimizing smart parking system by using fog computing. In *International Conference on Advances in Computing and Data Sciences*, pages 724–737. Springer, 2019.

[44] Kashif Munir. *Advancing Consumer-Centric Fog Computing Architectures*. IGI Global, 2018.

[45] Vighnesh Srinivasa Balaji. Fog computing and its challenges. In *The Rise of Fog Computing in the Digital Era*, pages 36–52. IGI Global, 2019.

[46] Saeed Mehrjoo and Farshad Khunjush. Optimal data aggregation tree in wireless sensor networks based on improved river formation dynamics. *Computational Intelligence*, 34(3):802–820, 2018.

[47] Frank Golatowski, Björn Butzin, Tim Brockmann, Thorsten Schulz, Martin Kasparick, Yuhong Li, Rahim Rahmani, Aviv Haber, Mustafa Sakalsız, and Özer Aydemir. Challenges and research directions for blockchains in the internet of things. In *2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*, pages 712–717. IEEE, 2019.

[48] Franco Cicirelli, Antonio Guerrieri, Carlo Mastroianni, Giandomenico Spezzano, and Andrea Vinci. *The Internet of Things for Smart Urban Ecosystems*. Springer, 2019.

[49] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog Computing and Its Role in the Internet of Things. *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16, 2012.

[50] Sander Soo, Chii Chang, Seng W Loke, and Satish Narayana Srirama. Proactive mobile fog computing using work stealing: Data processing at the edge. *International Journal of Mobile Computing and Multimedia Communications (IJMCMC)*, 8(4):1–19, 2017.

[51] M. Yannuzzi, R. Milito, R. Serral-Gracia, D. Montero, and M. Nemirovsky. Key ingredients in an IoT recipe: Fog Computing, Cloud computing, and more Fog Computing. *2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pages 325–329, 2014.

[52] Sanjay P Ahuja and Nathan Wheeler. Architecture of fog-enabled and cloud-enhanced internet of things applications. *International Journal of Cloud Ap-*

plications and Computing (IJCAC), 10(1):1–10, 2020.

[53] Ivan Stojmenovic and Sheng Wen. The Fog Computing Paradigm: Scenarios and Security Issues. 2:1–8, 2014.

[54] Antonio ATR Coutinho, Fabíola Greve, and Cássio Prazeres. An architecture for fog computing emulation. In *Anais do XV Workshop em Clouds e Aplicações (WCGA-SBRC 2017)*, volume 15. SBC, 2017.

[55] {Shi, Yingjuan and Ding, Gejian and Wang, Hui and Roman, H Eduardo and Lu, Si}. The Fog Computing Service for Healthcare. 2015.

[56] Mohammad Aazam and Eui-Nam Huh. Fog Computing Micro Datacenter Based Dynamic Resource Estimation and Pricing Model for IoT. *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, pages 687–694, 2015.

[57] Y Navaneeth Krishnan, Chandan N Bhagwat, and Aparajit P Utpat. Fog Computing- Network Based Cloud Computing. (Icecs):250–251, 2015.

[58] H. Madsen, G. Albeanu, Bernard Burtschy, and Fl Popentiu-Vladicescu. Reliability in the utility computing era: Towards reliable fog computing. *International Conference on Systems, Signals, and Image Processing*, pages 43–46, 2013.

[59] Pengfei Hu, Sahraoui Dhelim, Huansheng Ning, and Tie Qiu. Survey on fog computing: architecture, key technologies, applications and open issues. *Journal of network and computer applications*, 98:27–42, 2017.

[60] Oanh Tran, Thi Kim, Nguyen Dang Tri, Vandung Nguyen, Nguyen H Tran, and Choong Seon Hong. A Shared Parking Model in Vehicular Network Using Fog and Cloud Environment. pages 321–326, 2015.

[61] Soumya Datta and Bonnet Christian. Fog Computing Architecture to Enable Consumer Centric Internet of Things Services. 85:6–7, 2015.

[62] Subhadeep Sarkar, Subarna Chatterjee, and Sudip Misra. Assessment of the Suitability of Fog Computing in the Context of Internet of Things. *IEEE Transactions on Cloud Computing*, 7161(c):1–1, 2015.

[63] Ivan Stojmenovic. Fog computing: A cloud to the ground support for smart things and machine-to-machine networks. *2014 Australasian Telecommunication Networks and Applications Conference (ATNAC)*, pages 117–122, 2014.

[64] Heng Shi, Nan Chen, and Ralph Deters. Combining Mobile and Fog Computing: Using CoAP to Link Mobile Device Clouds with Fog Computing. *2015 IEEE International Conference on Data Science and Data Intensive Systems*, pages 564–571, 2015.

[65] Shu-Ching Wang, Wei-Shu Hsiung, and Chia-Fen Hsieh. Reliability enhancement of mobile edge computing for the iot with multiple damage communication. In *The International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, pages 825–832. Springer, 2019.

[66] Bart Garvelink. Mobile edge computing: a building block for 5g, 2018.

[67] Milan Patel, Brian Naughton, Caroline Chan, Nurit Sprecher, Sadayuki Abeta, Adrian Neal, et al. Mobile-edge computing introductory technical white paper. *White paper, mobile-edge computing (MEC) industry initiative*, pages 1089–7801, 2014.

[68] Fabio Giust, Gianluca Verin, Kiril Antevski, Joey Chou, Yonggang Fang, Walter Featherstone, Francisco Fontes, Danny Frydman, Alice Li, Antonio Manzalini, et al. Mec deployments in 4g and evolution towards 5g. *ETSI White paper*, 24:1–24, 2018.

[69] Sami Kekki, Walter Featherstone, Yonggang Fang, Pekka Kuure, Alice Li, Anurag Ranjan, Debashish Purkayastha, Feng Jiangping, Danny Frydman, Gianluca Verin, et al. Mec in 5g networks. *ETSI white paper*, 28:1–28, 2018.

[70] MECISG ETSI. Mobile edge computing (mec); framework and reference architecture. *ETSI, DGS MEC*, 3, 2016.

[71] Matthias Vodel and Wolfram Hardt. Data Aggregation and Data Fusion Techniques In WSN / SANET Topologies - A Critical Discussion -. 2012.

[72] Sakshi Chhabra and Dinesh Singh. Data Fusion and Data Aggregation / Summarization Techniques in WSNs : A Review. 121(19):21–30, 2015.

[73] Ramesh Rajagopalan and Pramod K. Varshney. Data-aggregation techniques in sensor networks: A survey. *IEEE Communications Surveys and Tutorials*, 8(4):48–63, 2006.

[74] Behrouz Pourghebleh and Nima Jafari Navimipour. Data aggregation mechanisms in the internet of things: A systematic review of the literature and recommendations for future research. *Journal of Network and Computer Applications*, 97:23–34, 2017.

[75] A Latha, S Prasanna, S Hemalatha, and B Sivakumar. A harmonized trust assisted energy efficient data aggregation scheme for distributed sensor networks. *Cognitive Systems Research*, 56:14–22, 2019.

[76] L. Shu, J. Lloret, J.J.P.C. Rodrigues, and M. Chen. Editorial: Distributed intelligence and data fusion for sensor systems. *IET Communications*, 5(12):1633, 2011.

[77] Wanjiun Liao, Jaime Lloret, and Lei Wang. Editorial. *Int. J. Sensor Networks*, 21(4):205, 2016.

[78] Robin Snader, Albert F. Harris III, and Robin Kravets. Tethys: A Distributed Algorithm for Intelligent Aggregation in Sensor Networks. *2007 IEEE Wireless Communications and Networking Conference*, pages 4117–4122, 2007.

[79] S. Commuri and V. Tadigotla. Dynamic Data Aggregation in Wireless Sensor Networks. *2007 IEEE 22nd International Symposium on Intelligent Control*, (October):1–3, 2007.

[80] Rabia Noor Enam. An Adaptive Data Aggregation Technique for Dynamic Cluster based Wireless Sensor Networks. 2014.

[81] Aristides Gionis, Heikki Mannila, and Panayiotis Tsaparas. Clustering Aggregation. (Icde), 2005.

[82] M K Iskander, a J Lee, and D Mosse. Confidentiality-preserving and fault-tolerant in-network aggregation for Collaborative WSNs. *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2012 8th International Conference on*, pages 107–116, 2012.

[83] Wilfried N. Gansterer, Gerhard Niederbrucker, Hana Straková, and Stefan Schulze Grotthoff. Scalable and fault tolerant orthogonalization based on randomized distributed data aggregation. *Journal of Computational Science*, 4(6):480–488, 2013.

[84] Henrich C. Pohls, Max Mossinger, Benedikt Petschkuhn, and Johannes Ruckert. Aggregation and perturbation in practice: Case-study of privacy, accuracy & performance. *2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, CAMAD 2014*, pages 183–187, 2015.

[85] Shlomo Berkovsky and Jill Freyne. Group-Based Recipe Recommendations : Analysis of Data Aggregation Strategies. *Proceedings of the fourth ACM*

conference on Recommender systems, pp. 111-118. ACM, pages 111–118, 2010.

[86] Toon De Pessemier. An Improved Data Aggregation Strategy for Group Recommendations. pages 36–39, 2013.

[87] Lucas A M C Carvalho, São Cristóvão, and Hendrik T Macedo. Users ' Satisfaction in Recommendation Systems for Groups : an Approach Based on Noncooperative Games. pages 951–958, 2013.

[88] Liang Hu, Jian Cao, Guandong Xu, Longbing Cao, Zhiping Gu, and Wei Cao. Deep Modeling of Group Preferences for Group-based Recommendation. 2013.

[89] Enrique Herrera-Viedma, Francisco Herrera, and Francisco Chiclana. A consensus model for multiperson decision making with different preference structures. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.*, 32(3):394–402, 2002.

[90] D Ben-Arieh and Chen Z. Linguistic-labels aggregation and consensus measures for autocratic decision making using group recommendations. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, Systems an(3):558568, 2006.

[91] Ignacio Javier Perez, Francisco Javier Cabrerizo, and Enrique Herrera-Viedma. A Mobile Decision Support System for Dynamic Group Decision-Making Problems. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 40(6):1244–1256, 2010.

[92] Y A O Yu, Z H U Shan-feng, and Chen Xin-meng. Rank Aggregation Algorithms Based on Voting Model for Metasearch. (22):214–216, 2007.

[93] Farzad Farnoud, Behrouz Touri, and Olgica Milenkovic. Nonuniform Vote Aggregation Algorithms. pages 1–10, 2012.

[94] Muhl Kumar, Loren Schwiebert, and Monica Brockmeyer. Efficient data aggregation middleware for wireless sensor networks. pages 579–581, 2004.

[95] Gleb Beliakov, Tomasa Calvo, and Simon James. Consensus measures constructed from aggregation functions and fuzzy implications. *Knowledge-Based Systems*, 55:1–8, 2014.

[96] Julien Ah-Pine and Xerox Corporation. Data Fusion Using Consensus Aggregation Functions. 1(12):0–4, 2003.

[97] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[98] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *Concurrency: The Works of Leslie Lamport*, page 203, 2019.

[99] L. Patnaik, Balaji, and S. Byzantine-resilient distributed computing systems. 11(October):81–91, 1987.

[100] Miguel Castro and Barbara Liskov. Practical Byzantine Fault Tolerance. *Proceedings of the Symposium on Operating System Design and Implementation*, (February):1–14, 1999.

[101] Dennis Wang, Nina Tai, and Yicheng An. Byzantine fault tolerant raft, 2018.

[102] Divya Mary. Improved practical byzantine fault tolerance for blockchains. 2019.

[103] Olumuyiwa Oluwasanmi, Jared Saia, and Valerie King. An Empirical Study of a Scalable Byzantine Agreement Algorithm. 2010.

[104] Shreya Agrawal and Khuzaima Daudjee. A performance comparison of algorithms for byzantine agreement in distributed systems. In *2016 12th European Dependable Computing Conference (EDCC)*, pages 249–260. IEEE, 2016.

[105] Tobias Distler and Christian Cachin. Resource-Efficient Byzantine Fault Tolerance. 65(9):2807–2819, 2016.

[106] Hua Chai and Wenbing Zhao. Byzantine fault tolerant event stream processing for autonomic computing. *Proceedings - 2014 World Ubiquitous Science Congress: 2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing, DASC 2014*, pages 109–114, 2014.

[107] Michele Amoretti and Francesco Zanichelli. Distributed reputation management for service-oriented peer-to-peer enterprise communities. *IJCSE*, 13(2):147–157, 2016.

[108] Thomas Roche, Mathieu Cunche, and Jean Louis Roch. Algorithm-based fault tolerance applied to P2P computing networks. *1st International Conference on Advances in P2P Systems, AP2PS 2009*, pages 144–149, 2009.

[109] Hodjatollah Hamidi. A new method for transformation techniques in secure information systems. 2016.

[110] Hans P Reiser. Byzantine Fault Tolerance for the Cloud. *Symposium A Quarterly Journal In Modern Foreign Literatures*, (November), 2011.

[111] Christian Berger and Hans P Reiser. Webbft: Byzantine fault tolerance for resilient interactive web applications. In *IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 1–17. Springer, 2018.

[112] Mohammed A AlZain, Alice S Li, Ben Soh, and Mehedi Masud. Byzantine fault-tolerant architecture in cloud data management. *International Journal of Knowledge Society Research (IJKSR)*, 7(3):86–98, 2016.

[113] Mehdi Nazari Cheraghlou, Ahmad Khadem-Zadeh, and Majid Haghparast. A survey of fault tolerance architecture in cloud computing. *Journal of Network and Computer Applications*, 61:81–92, 2016.

[114] Patricia Takako Endo, Djamel Sadok, and Judith Kelner. Autonomic Cloud Computing: Giving intelligence to simpleton nodes. *Proceedings - 2011 3rd IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2011*, pages 502–505, 2011.

[115] Rajkumar Buyya, Rodrigo N. Calheiros, and Xiaorong Li. Autonomic Cloud computing: Open challenges and architectural elements. *Proceedings - 2012 3rd International Conference on Emerging Applications of Information Technology, EAIT 2012*, pages 3–10, 2012.

[116] Shusen Yang, Xinyu Yang, Julie A. McCann, Tong Zhang, Guozheng Liu, and Zheng Liu. Distributed networking in autonomic solar powered wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 31(12):750–761, 2013.

[117] Admilson de Ribamar Lima Ribeiro, Fernando Mendonça de Almeida, Edward David Moreno, and Carlos AE Montesco. A management architectural pattern for adaptation system in internet of things. In *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 576–581. IEEE, 2016.

[118] Atia Javaid, Nadeem Javaid, and Muhammad Imran. *Ensuring analyzing and monetization of data using data science and blockchain in loT devices*. PhD thesis, MS thesis, COMSATS University Islamabad (CUI), Islamabad 44000, Pakistan, 2019.

[119] Joel Grus. *Data science from scratch: first principles with python*. O'Reilly

Media, 2019.

[120] David Ribes. Sts, meet data science, once again. *Science, Technology, & Human Values*, 44(3):514–539, 2019.

[121] Cristina Alonso-Fernández, Antonio Calvo-Morata, Manuel Freire, Iván Martínez-Ortiz, and Baltasar Fernández-Manjón. Applications of data science to game learning analytics data: A systematic literature review. *Computers & Education*, page 103612, 2019.

[122] Quan-Hoang Vuong, Viet-Phuong La, Thu-Trang Vuong, Manh-Toan Ho, Hong-Kong T Nguyen, Viet-Ha Nguyen, Hiep-Hung Pham, and Manh-Tung Ho. An open database of productivity in vietnam's social sciences and humanities for public use. *Scientific data*, 5:180188, 2018.

[123] Richard F Gunst. *Regression analysis and its application: a data-oriented approach*. Routledge, 2018.

[124] Yichuan Wang, LeeAnn Kung, and Terry Anthony Byrd. Big data analytics: Understanding its capabilities and potential benefits for healthcare organizations. *Technological Forecasting and Social Change*, 126:3–13, 2018.

[125] Deanne Larson and Victor Chang. A review and future direction of agile, business intelligence, analytics and data science. *International Journal of Information Management*, 36(5):700–710, 2016.

[126] Charles M Judd, Gary H McClelland, and Carey S Ryan. *Data analysis: A model comparison approach*. Routledge, 2011.

[127] Belle Selene Xia and Peng Gong. Review of business intelligence through data analysis. *Benchmarking: An International Journal*, 21(2):300–311, 2014.

[128] David J Bartholomew, Fiona Steele, Jane Galbraith, and Irini Moustaki. *Analysis of multivariate social science data*. Chapman and Hall/CRC, 2008.

[129] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246. ACM, 2002.

[130] Jason Brownlee. Discover feature engineering, how to engineer features and how to get good at it. *Machine Learning Process*, 2014.

[131] Shaomin Wu. A review on coarse warranty data and analysis. *Reliability Engineering & System Safety*, 114:1–11, 2013.

[132] Selim Aksoy and Robert M Haralick. Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern recognition letters*, 22(5):563–582, 2001.

[133] Maico Cassel and F Lima. Evaluating one-hot encoding finite state machines for seu reliability in sram-based fpgas. In *12th IEEE International On-Line Testing Symposium (IOLTS'06)*, pages 6–pp. IEEE, 2006.

[134] Ariel Kleiner, Ameet Talwalkar, Purnamrita Sarkar, and Michael I Jordan. A scalable bootstrap for massive data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(4):795–816, 2014.

[135] Jason Wang and Luis Perez. The effectiveness of data augmentation in image classification using deep learning. *Convolutional Neural Networks Vis. Recognit*, 2017.

[136] Christian Mancas. *Conceptual Data Modeling and Database Design: A Fully Algorithmic Approach, Volume 1: The Shortest Advisable Path*. Apple Academic Press, 2016.

[137] Graeme Simsion and Graham Witt. *Data modeling essentials*. Elsevier, 2004.

[138] Nantia Makrynioti, Nikolaos Vasiloglou, Emir Pasalic, and Vasilis Vassalos. Modelling machine learning algorithms on relational data with datalog. In *Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning*, page 5. ACM, 2018.

[139] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[140] Ron Kohavi. *Special issue on applications of machine learning and the knowledge discovery process*. Kluwer, 1998.

[141] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[142] Patrick St Pierre. Editorial commentary: Proficiency-based training: Are we ready for a new way to train and test orthopaedic surgeons?, 2018.

[143] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media, 2009.

[144] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4):611–629, 2018.

[145] Sudhir Varma and Richard Simon. Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics*, 7(1):91, 2006.

[146] Eitan Rothberg, Tingting Chen, and Hao Ji. Towards better accuracy and robustness with localized adversarial training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 10017–10018, 2019.

[147] Suhua Lei, Huan Zhang, Ke Wang, and Zhendong Su. How training data affect the accuracy and robustness of neural networks for image classification. 2018.

[148] Gang Leng, Thomas Martin McGinnity, and Girijesh Prasad. Design for self-organizing fuzzy neural networks based on genetic algorithms. *IEEE Transactions on Fuzzy Systems*, 14(6):755–766, 2006.

[149] Nazri Mohd Nawi, Meghana R Ransing, and Rajesh S Ransing. An improved learning algorithm based on the broyden-fletcher-goldfarb-shanno (bfgs) method for back propagation neural networks. In *Sixth International Conference on Intelligent Systems Design and Applications*, volume 1, pages 152–157. IEEE, 2006.

[150] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[151] Hava T Siegelmann and Eduardo D Sontag. On the computational power of neural nets. *Journal of computer and system sciences*, 50(1):132–150, 1995.

[152] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[153] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[154] Andrej Karpathy et al. Cs231n convolutional neural networks for visual recognition. *Neural networks*, 1, 2016.

[155] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[156] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Neural network-based user-independent physical activity recognition for mobile devices. *arXiv preprint arXiv:1502.04623*, 2015.

[157] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[158] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[159] Theertham Akilesh Sai and Hee-hyol Lee. Weight initialization on neural network for neuro pid controller-case study. In *2018 International Conference on Information and Communication Technology Robotics (ICT-ROBOT)*, pages 1–4. IEEE, 2018.

[160] Qingxing Dong and Thomas L. Saaty. An analytic hierarchy process model of group consensus. *Journal of Systems Science and Systems Engineering*, 23(3):362–374, 2014.

[161] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[162] Daniel Minoli. *Building the internet of things with IPv6 and MIPv6: The evolving world of M2M communications*. John Wiley & Sons, 2013.

[163] Gabriel Montenegro, Nandakishore Kushalnagar, Jonathan Hui, and David Culler. Transmission of ipv6 packets over ieee 802.15. 4 networks. Technical report, 2007.

[164] threadgroup. *Thread Stack Fundamentals (technocal white paper)*, 2015.

[165] Shahid Raza, Prasant Misra, Zhitao He, and Thiemo Voigt. Building the internet of things with bluetooth smart. *Ad Hoc Networks*, 57:19–31, 2017.

[166] Inc. LoRa Alliance. LoRaWAN 1.1 Specification. Technical report, LoRa Alliance, Inc., 2017.

[167] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *Esann*, 2013.

[168] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *International workshop on ambient assisted living*, pages 216–223. Springer, 2012.

[169] David J. Hand. Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine Learning*, 77(1):103–123, Oct 2009.

[170] Shijun Wang, Diana Li, Nicholas Petrick, Berkman Sahiner, Marius George Linguraru, and Ronald M Summers. Optimizing area under the roc curve using semi-supervised learning. *Pattern recognition*, 48(1):276–287, 2015.

[171] Eckert C. Kolosnjaji B. Neural network-based user-independent physical activity recognition for mobile devices. *Intelligent Data Engineering and Automated Learning IDEAL 2015*, 9375:378–386, 2015.

[172] Diane J Cook. Learning setting-generalized activity models for smart spaces. *IEEE intelligent systems*, 2010(99):1, 2010.

[173] Parisa Rashidi and Diane J Cook. Keeping the resident in the loop: Adapting the smart home to the user. *IEEE Trans. Systems, Man, and Cybernetics, Part A*, 39(5):949–959, 2009.

[174] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[175] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Data augmentation using synthetic data for time series classification with deep residual networks. *arXiv preprint arXiv:1808.02455*, 2018.

[176] Hwa-Yeon Kim, Yoon-Hyung Roh, and Young-Gil Kim. Data augmentation by data noising for open-vocabulary slots in spoken language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 97–102, 2019.

[177] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 7–10. ACM, 2016.

[178] Lewis Smith and Yarin Gal. Understanding measures of uncertainty for adversarial example detection. *arXiv preprint arXiv:1803.08533*, 2018.

[179] Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.

[180] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, 2012.

[181] Quoc V Le, Jiquan Ngiam, Adam Coates, Abhik Lahiri, Bobby Prochnow, and Andrew Y Ng. On optimization methods for deep learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 265–272. Omnipress, 2011.

[182] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

[183] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[184] Yarin Gal. *Uncertainty in deep learning*. PhD thesis, PhD thesis, University of Cambridge, 2016.

[185] Nawel Yala, Belkacem Fergani, and Anthony Fleury. Feature extraction for human activity recognition on streaming data. In *2015 International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, pages 1–6. IEEE, 2015.