Towards an efficient data analytics architecture for the Internet of Things

Thesis submitted for the degree of

Doctor of Philosophy

at the University of Leicester

by

Badraddin Alturki

School of Informatics

University of Leicester

October 2020

Declaration

I declare that this thesis has been produced from my own work, which has not submitted earlier to any degree in any University. In addition, all the sources that have been used are acknowledged as academic references. During my PhD I have published several papers including ([16], [17], [77]) and some of the contents that are used in this thesis are from my papers.

Badraddin Alturki

October 2020,

Leicester

Abstract

In the Internet of Things (IoT), the traditional architecture aims to process the data in the cloud. This creates several challenges such as high communication latency between the end devices and the cloud while making the network busy by sending all the raw data continuously. In this thesis, we propose an alternative architecture for the IoT which processes part of the data in the fog to avoid all raw data to be sent to the cloud. However, the cloud processes intensive data analytics. We conduct a trade-off analysis to show the advantages of applying data fusion closer to the data source and then processing the intensive data analytics algorithms in the cloud. We explore the effectiveness of the available architectures including centralised, decentralised, and distributed architecture to propose the most effective data analytics architecture for the IoT. The trade-off analysis shows the effectiveness of various service decomposition strategies leading to an understanding the various balances between Fog and IoT processing and their effectiveness in data communications reduction and result accuracy allowing achievements of 70% data communication reduction while still achieving approximately 90% accuracy. We propose a service distribution strategy called Most Efficient IoT Node (MEIN), which aims to distribute the services to either cloud nodes or fog nodes based on their capabilities while maintaining the usage of resource in IoT architecture. This strategy selects the best nodes and distributes the services on nodes based on the demands of services and capabilities of nodes.

Acknowledgements

First of all, I praise Allah for gifting health and patience, as without him I could not tackle all the challenges through my PhD and life. Then, I would like to thank to several people for their support through my studies and my life.

I would like to thank Prof Stephan Reiff-Marganiec and Dr Fer-Jan de Vries for their supervision through my PhD work. Also, thanks to Dr Charith Perera. Their support, advice and motivation have been significant for the progress of my thesis. I appreciate their valuable guidance and their experience for giving me valuable input.

I want to thank my sponsor King Abdul Aziz University for funding my studies and the Saudi Arabian Cultural Bureau in London who gave me constant support.

Special thanks to my parents, without their moral support and help this project could not be finished. I owe a debt of gratitude to my lovely family; wife, daughter (Alaa) and son (Ali). I am so thankful for their patience and their time to support me and encourage me as this journey was not short and easy.

To my friends and colleagues, they were good listeners, generous in sharing information, experienced in giving advice and with them this long journey became memorable. Thanks especially to Marco, Jakob and Bello.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recently, the IoT has become one of the most trending topics among researchers in academia and business. The Cluster of European Research Projects on the Internet of Things (CERP-IoT) defined the IoT as allowing "*people and things to be connected Anytime, Anyplace, with Anything and Anyone, ideally using Any network and Any service*" [98]. The IoT has extended the Internet's vision by connecting physical objects in our environment to the internet. The vision of IoT includes cyber-physical functionalities, data gathering, data analytics, volatility, and heterogeneity. In the IoT, the objects can be any objects that can have an internet connection, also can collect and share data over the network without the intervention of human that allows us to produce insights and get useful information.

Sensors embedded in many everyday physical objects around us play a key role in the IoT. These sensor embedded objects include vast sensing capabilities [70] and can then send the data through the network to decision points typically in the cloud. After collecting the data, there is a need to analyse them to gain insights to help, automate and speed up decision making [75]. In other words, the idea is to enrich the objects in our daily life with sensors that collect relevant information about the state of these objects. Our aim is to analyse all this local data near to the data source, then only communicate relevant insights to the cloud. In a data driven economy, the data and insights are considered as the main goods [67].

The IoT devices have been created with limited computational capabilities and constraints which are part of their characteristics. In addition to the constraints of devices, the services have constraints which make the process of distributing the services to the devices difficult. This allows the approaches that are built-in software engineering area to focus on the cloud platforms that have significantly more capabilities than the IoT devices. Cloud computing uses remote servers that are available on the Internet for several purposes including data storing, management and processing. This is an effective way to have extremely high computation power from various sources and provide the fastest service to the users. However, in some cases, the data needs to be processed and analysed locally to have increased privacy, fast responses to the users and to have reduced usage of network and storage resources.

There are several projections regarding the amount of internet-connected devices and how the IoT will affect on the internet and economy during the next decade from business and academia. For example, Cisco projects that "*there will be 29.3 billion networked devices by 2023, up from 18.4 billion in 2018*" [7]. Besides, the number of M2M connected devices will be increased from 6.1 billion in 2018 to 14.7 billion by 2023. However, Gartner expects that the amount of internet-connected will be around 20 billion devices by 2020 [48]. Furthermore, the IoT will have a financial impact on the worldwide economy between \$3 to \$11 trillion by 2025 as estimated by McKinsey Global Institute [58]. It is clear that these projections are educated guesses in numbers, but all of them point at the effect of IoT on the internet and the economy.

Hypothesis of this thesis is that there is a need for distributing data processing over the network by pushing the data processing near to the data source to find the best place for services to process them. In this regard, fog computing has been proposed to collect, process and analyse the sensors' data locally in the IoT devices. Fog computing can be considered as a distributed cloud computing that is deployed on the edge of the network, but with limited capabilities. Moving the data processing near to the data source is challenging, but also, determining which services should be distributed to which devices need to be considered as one of the main challenges.

There has been a recent interest in moving away from centralised data processing centres to a more distributed fog computing paradigm to bring computing to the edge of the network, closer to user devices [16, 19, 46]. Fog computing is defined as a hierarchically distributed computing paradigm that bridges cloud data centres and IoT devices [23]. The combination of IoT devices and fog computing enables smart environments that can respond to real-time events by combining services offered by multiple heterogeneous devices. IoT based on fog computing and distributed data processing is still under research and different challenges remain open. Though there have been advances in fog computing with proposals for reference architectures [38], practical realisations need to tackle the challenge of resource management [102]. A recent study [102] identifies that local resource management has to consider the following problems: Provisioning the fog nodes to execute the workloads that are distributed from the cloud, resource management of fog nodes and deployment of workloads to fog nodes.

The IoT data life cycle starts from producing data, collecting sensors, filtering, fusing and pre-processing, to storage and archiving, finally to querying and analysis [12]. In the IoT system, there are different nodes in the network including IoT nodes, fog nodes and cloud nodes. There are two levels in the network namely fog level and cloud level, the fog level has IoT nodes and fog nodes which are constrained devices and the cloud level has an unlimited number of devices with high computational power. A device can be considered as a constrained device when the characteristics of the device have constraints on ROM, RAM, processing power and energy [24].

All of these require knowledge and an awareness of the resources available on fog devices as well as constraints related to the services that run on such nodes. Existing approaches do not focus on service constraints or the data computation capabilities of local devices [103]. Moreover, the range of possible data computation capabilities in the IoT devices also needs to be taken into account when distributing service processes among the nodes.

So, in this IoT data processing, we are going to focus on the distribution of computation across the entire space. For example, the fog level should be responsible for collecting, filtering, fusing and pre-processing the data. In addition, based on IoT and fog nodes' capabilities analysing and processing the IoT data will be done. Otherwise, the final data analytics is performed globally based on the data resulting from the local preprocessing of local data. This means that data analytics can be done at both levels of fog and cloud. These levels play an important role in different stages.

The remainder of this chapter will discuss research problems and challenges, contributions followed by the thesis statement. Then, it will give an overview of the whole remaining parts of the thesis.

1.1 Research problems and challenges

As we mentioned earlier, the (IoT) objects will produce a significant amount of data when they are connected to the internet and communicate over the network. In a centralised architecture, the aim is to process the data in a single point of decisions possible in the cloud. As a result, a significant amount of data needs to be communicated to the cloud. This architecture creates a number of challenges such as high data communication over the network. This means that there is a need to have an alternative architecture to counter these weaknesses.

Constraint awareness is an important aspect of the design of the IoT architecture as it will connect a large number of devices with varying computational capabilities, storage, battery power and Internet connectivity. Further, there will be a variety of services with different requirements (e.g. resource requirements, data requirements, latency requirements). These services will run on IoT nodes which are constrained devices, and services will use bandwidth. This means there is a strong relationship between the services and nodes in the IoT. The main challenge is to determine which services should be run on either fog devices, the cloud devices or both fog and cloud devices in a given IoT architecture, by considering both overall efficiency and feasibility.

Besides, the management of resources at the edge of a network is crucial for evaluating the potential of fog computing. Proposing an efficient architecture for the IoT brings several challenges. Personal data store [73] is an example of where addressing these challenges becomes a necessity.

As the IoT system becomes big and complex, the deployment of services in a distributed architecture will be complex which will require a way to decompose the big size services into smaller services, then distribute the services to the node/s effectively, which will lead to finding the ideal architecture. This can lead to many challenges, we are going to address the following research challenges (RC) and answer the following research questions.

RC1. Most of the research and existing work in the field of the IoT relies on cloud computing, because of the offered power in terms of processing and storage. The common way to process the data is to send all data to the cloud and return results after analysis. In addition to the significant power available in the cloud, the processing in the cloud means that as complete a collection of data is available to analysis as can be obtained.

However, processing all streaming raw data in the cloud negatively affects several aspects, such as increased network traffic, latency (to get actions back to the user), energy consumption and privacy. As the IoT grows the need to tackle these issues grows. This means that there is a need to explore the possible architectures, then apply experiments in the architectures to check which architecture is the most efficient. Then, we need to use the best architecture to evaluate and validate the architecture's efficiency. The following issues are the key issues to address:

1) How to process data in the IoT in an efficient way?

- Where to process the data in the network?
- How to move the computation near to the edge of the network?

- How to evaluate the efficiency of an architecture?
- How to handle large data that will be sent over the network?

RC2. Exploring different architectures is important to check the efficiency of each architecture. This can be a challenging process, because finding the most effective architecture needs a trade-off analysis to demonstrate the advantages and disadvantages of each architecture. Moreover, IoT devices have limited capabilities, which means that not every device is capable of processing all the services. Thus, there is a need to decompose services into smaller services, then distribute them among the IoT devices, also we cannot ignore the power of the cloud in cases where IoT devices cannot handle the services.

Hence, we need to explore and demonstrate the effectiveness of service decomposition and distribution by applying experiments and using different types of datasets. After applying experiments, the results will indicate the efficient and inefficient architecture with the given configuration and setup. It is not possible to say that one of the architectures is the best for all types of data and applications. The following issues are the key issues to address:

2) How to identify the most effective data analytics architecture for the IoT?

- What are the strategies to use while exploring the architectures?
- What aspects can affect the architecture's efficiency?
- How to select an effective architecture?
- How to evaluate and validate the effectiveness of an architecture?

RC3. The IoT and fog nodes are constrained devices because of their limitations in processing power. Therefore, it is not possible to process large workloads on fog nodes. Moreover, it is difficult to make a decision about the amount of computation load that can be assigned to a fog node.

Also, distributing the intelligence across the fog nodes is challenging since most of the neural network, artificial intelligence and machine learning algorithms require high processing power. Therefore, we need to know and have full knowledge of the nodes and services. Then we need to do optimisation in the process of distribution of services on the nodes while maintaining the resource usage in an efficient way. The following issues are the key issues to address:

3) How to distribute services to the nodes according to their resources?

- What are the capabilities of nodes and the requirements of services?
- How to distribute the services to the nodes in the network?
- How important is service distribution strategy and optimising the resource usage?
- How to evaluate the efficiency of the distribution strategy?

The research challenges RC1 and RC2 are the architectural approaches, and the use of RC1, which is the use of the concept of understanding what to keep on the IoT device and what to ship out is all aimed to create the partitioning process. In other words, the idea is to understand where to process the data in the network. Then, in RC2, understanding whether to send the big services or to decompose them into smaller services before sending them. This is the key outcome of the work, RC1 and RC2 contribute towards the partitioning process. Then, RC3 is the use of the concept of mapping the services to the nodes in the network. The three research challenges RC1, RC2 and RC3 are integrated challenges to this particular outcome.

1.2 Methodology

In order to address these challenges, we use experimental methodology to explore and evaluate possible solutions for the challenges. We start with understanding the balance between processing data in one place and distributing them while considering the accuracy and performance. After that evaluating that balance by running experiments several times and the results that indicate what went good. After getting the insights from experimental approach, we decide to design the new architecture that will allow us to do this. We propose an architecture and understand which data we need to keep and process in the IoT nodes and what to distribute to other nodes (fog and cloud) in the network. Then, evaluating the accuracy, data communication and the performance show that the proposed approach is good with the given dataset and setting. Also, we compared the proposal architecture with other architectures to validate the architecture's performance. We realised that there are big size services that are not difficult to process them with constrained devices and distributing them costs high data communication. Therefore, we propose the strategy where we can decompose the big services into smaller services by using the same architecture and approach that made earlier and distribute the services to the other nodes by experimenting. This partitioning process is important as IoT and fog devices are constrained devices, this will let them process the small services effectively. Then, we propose a service distribution strategy to map the services based on their technical requirements with the nodes based on their capabilities. In this part we have used assumptions and data about services and resources. Also, we use an optimisation method to optimise the use of nodes capabilities.

1.2.1 Optimisation problem and the criteria for the success

Our problem is multi criteria optimisation problem and the objective function is multi objective function as we want to minimise data communication (min-data), minimise execution time (min-exec), maximise privacy (max-privacy), minimise energy usage (min-energy), maximise usage of nodes (max-node) and maximise number of distributed services to fog nodes (max-number). These are also can be considered as the success criteria for the proposed approach.

Objective Function

$minimize/maximize\ f_m(x)$

$$m = \{\text{min}-\text{data}, \text{min}-\text{exec}, \text{max}-\text{privacy}, \text{max}-\text{node}, \text{max}-\text{number}\}$$

f is the objective function, m is the set of objectives and x is a vector of variables.

The elements of objectives m are explained as follows. Firstly, min$-$data means that minimising the cost of the data communication over the network. Secondly, min$-$exec means that minimising the execution time of processing the services over the network. Thirdly, max−privacy means that maximising privacy by processing the data as much as possible locally to have control of the data. Fourthly, max−node means that maximise the usage of nodes by using as much as possible the full capabilities of nodes. Finally, max−number means that maximise the number of distributed services to fog nodes as much as possible.

In later chapters the experiments are designed to compare different scenarios against that objective function and different set up experiments will allow us to see what works best.

There are set of approaches that can be used to solve the optimization problems as follow. Bin Packing is "*NP-hard [40] and heuristics have been developed to approximate the minimum number of bins*" [51]. Another "*NP-hard*, knapsack is "to fill a given multi-dimensional capacity-limited knapsack with a subset of items in order to get the maximum benefit associated with the profit of each selected item" [39]. Additionally, genetic algorithm (GA) is "a search technique; it is based on Darwin's theory of evolution and selection of biological systems" [78]. The Bin Packing approach will be used in chapter 5.

1.3 Thesis statement

The high increase in data generated from various sources and the demand for processing and managing these data in an efficient and effective way is a significant challenge in the Internet of Things. We propose an efficient approach to process these data closer to the source where feasible. In addition, we determine a strategy for distributing services to nodes based on the capabilities of nodes and the technical requirements of services.

1.4 Research scope and contribution

The overall aim of my research is proposing an efficient data analytics architecture to process the data with less bandwidth usage, faster response time, optimised resource usage and identify an ideal way to process the data in large scale. Also, evaluating and testing the proposed approach by using real experiments and possibly simulation is one of the objectives. The main research contributions (C) of this thesis are:

C1. In response to the first research challenge, we propose an efficient architecture for the IoT which moves the computation near to the fog side of the network. Then, we preprocess and fuse the data in WISDM dataset [55] locally, finally we apply five machine learning algorithms on the data that is preprocessed locally. Additionally, we evaluate the architecture by comparing it with the traditional centralised architecture and decentralised architecture to validate the efficiency of the proposed architecture (Chapter 3).

C2. To address the second research challenge, we explore the effectiveness and ineffectiveness of the architectures to find the most effective one among them. Then, we conduct a range of experiments on four architectures to see how different types of data modalities including numerical, text and images using different Machine Learning algorithms, perform in each architecture to highlight the efficiency of architecture and the importance of efficiently selecting which services should run on which node (Chapter 4).

C3. To address the third research challenge, we explore to which nodes services can be best distributed to the right nodes while maintaining the usage of resources in the IoT architecture. Also, we propose a distribution strategy called Most Efficient IoT Node (MEIN) for distributing the services on the nodes based on their capabilities. Also, it will be used for optimising the usage of resources. We will use the bin packing algorithm as a baseline, then we extended this algorithm in a way to fit in Fog and IoT environment. This strategy selects the best nodes and distributes the services on nodes based on the demands of services and capabilities of nodes. We conduct a range of experiments to find the most efficient combination of capabilities for fog nodes which will allow us to distribute the services as much as possible to the fog nodes (Chapter 5).

In summary, (C1) we start by proposing an architecture and understand which data we need to keep in the IoT nodes and what to distribute to other nodes in the network. After understanding the architecture, (C2) we decompose the services into smaller services and the dataset to smaller data, the partitioning process is important as IoT and fog devices are constrained devices, this will let them process the small services effectively. Then, (C3) we distribute the services to the nodes based on the services demand and the node capabilities, which is the mapping and optimisation process.

1.5 Thesis overview and summary

This chapter has discussed the introduction of the thesis by providing a general review of contributions. Also, we presented the research challenges for the IoT data analytics architecture that aims to move the computation near to the data source. We have provided a thesis statement that concentrated on efficient data processing in fog computing within IoT environments.

The rest of this thesis is organised as follows:

- In chapter 2, we discuss the details of the background and related works of this project. We provide an overview of the Internet of Things, IoT architectures and Fog Computing. Then, we present data analytics in the Internet of Things and an overview of data fusion. Also, providing the available data analytics architectures. Followed by summarising the main related works in the area of Fog Computing in the Internet of Things. Then, we discuss the evaluation of the IoT system and listing the available datasets.

- In chapter 3, we introduce the proposal of efficient data analytics architecture for the Internet of Things. We evaluate the proposed architecture by using a publicly available dataset and five data analytics techniques and how feasible the IoT devices are when processing these techniques.

- In chapter 4, we explore four different architectures for the Internet of Things to select the most effective architecture. Also, we explore service decomposition strategy in a distributed architecture for the Internet of Things. Then, we evaluate the four architectures by using three types of datasets and apply trade-off analysis to show the efficiency of each architecture.

- In chapter 5, we explore the importance of distributing services to the nodes in an optimised way. Then, we propose a distribution strategy that can distribute the services to the nodes based on their capabilities, also the strategy keeps the resources usage optimised and maintained. We evaluate the strategy by conducting 15 experiments on randomly generated data to show the efficiency of the strategy.

- In chapter 6, we discuss and evaluate the key elements in this work by presenting the observation about the results of real testbed from the experiments introduced in chapter 3 and 4, highlight the insights from the comparison between the proposed architecture and other architectures in a more detailed manner.

- In chapter 7, we summarise this thesis and highlight the main contributions and conclusions. We outline a number of potential future research topics.

Chapter 2

Research background and related work

2.1 The Internet of Things

Recently, the IoT has become one of the most popular topics in both academia and business. It is accepted that there is no single standard definition for the IoT until now. In the IoT, the vision of the internet has been expanded to add all disconnected things to the network. As we mentioned earlier in the Introduction, the IoT will connect the human with objects at any time and place by communicating over the network. This means that the IoT will have heterogeneous data which will make the IoT project more complex to manage and to process data that are produced from various sources.

Moreover, the IoT can be considered as "*a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies*" [85]. The IoT bundles different technologies namely sensors, actuators, cloud computing, modelling of data, storage, semantic, data analytics, communication and computation. These technologies are coupled together based on the needs of the IoT. The IoT is designed to be a smart world in which smart things can communicate autonomously.

The IoT has characteristics like connectivity, sensing, energy, interoperability, dynamism, autonomous, intelligence, privacy and security that are extracted from the description supplied by CERP-IoT [91]. All these characteristics are relevant for the successful execution of tasks by the IoT.

Autonomy is an important IoT characteristic because it is important to reduce user intervention as much as possible. Automating the configuration of all parts of the system is needed to accomplish the aim. Engineering the IoT software is a difficult task because the IoT data are generated by various sources. Therefore, there is a need to follow the common aspects when dealing with heterogeneity like configuring the hardware and networking related things. Also, translating the message from a source to a different source using standard formats such as JSON and XML is important in this context. There are several IoT platforms that provide cloud services such as Xively, Thingworx, OpenIoT and Carriots.

The life-cycle of IoT data starts with producing data and then moves to data aggregation, then transferring the data before optional filtering and processing followed by data storing and archiving, finally, making analysis on the data to get useful insights. [26]. Considering the overall IoT, we always have devices at the edge as well as in the network and the cloud in a central position. Data processing can be at the in-network (so edge and devices in the network) and the cloud level [12, 100] – and these levels play a role in the various stages.

Advances in the sensing and data processing capabilities of devices, coupled with that of communication networks are leading to the maturity of the Internet of Things (IoT) paradigm. A number of application domains, such as smart healthcare, smart homes and buildings, now rely on devices as varied as user smartphones, sensor network gateways and network routers for their realisation. Most of these applications use devices for sensing and data pre-processing tasks such as aggregation and filtering, with the majority of the data analysis done in centralised cloud infrastructures [102]. One of the main cloud providers is Microsoft with its Azure Stack [106] which offers a hybrid cloud that allows companies to transfer benefits from their servers while keeping the management of servers for new types of cloud (hybrid cloud). In addition, they provide gateway devices in the cloud and data analytics. Similarly, IBM has an online web analytics system with IBM Digital Analytics. This service provides tracking and analysing of behaviours from visitors. The data analytics uses high power servers inside IBM. The IBM PureData system promises fast data analytics and warehouse that combine warehouse, data centres and analytics [33]. With the predicted increase in the number of devices (28 billion connected devices by 2021 [116], [3]), which can participate in existing and emergent applications, geographically centralised cloud data centres will find it difficult to support the highly distributed IoT devices without suffering a loss in quality of service. The resultant massive flow and exchange of data from a large number of connected devices impacts on electricity costs and carbon emissions [54], with achieving energy efficiency a serious challenge. As typical IoT applications being highly context-dependent, the resultant short but high-frequency data communication pattern from participating devices will pose a challenge to the bandwidth of the communication and cloud frameworks [79]. The existing computing and communication infrastructure is likely to cause an unacceptably high latency in service delivery and network congestion [57], as recent studies showing that cloud servers geographically situated far from user devices affects latency more negatively than those geographically closer [102]. The IoT applications that collect sensitive data such as users' private information or location face the challenging decision of whether to store it locally or communicate it to the cloud, since securing the data will incur overheads and subsequently affect performance [9].

Sustainability is also important for the IoT applications when deploying real-world systems because of factors such as computation strategy, energy consumption, computation workload and data distribution strategy. The authors of [72] discussed ten crucial characteristics including data analytics, security and privacy, context awareness, mobility and other features to develop sustainable fog computing architectures. A sustainable system aims to optimise trade-offs when selecting the computation strategy, energy consumption and data communication usage. Thus, the proposed infrastructure can help develop sustainable computing architectures as it can enable handling of more computation workload at the network edge by distributing the data computation efficiently.

2.2 Architectures

The Internet Architecture Board (IAB) described the most common 3 communication patterns for smart objects in the IoT [94] that depend on the individual characteristics of the smart objects. These characteristics of devices bring constraints when using software architecture in the Internet of

Things. For example, when the system requires big data analytics function which cannot be handled by IoT devices because of the computational constraints, then the system should ask for the processing from much more powerful devices than the constrained devices. The communication patterns that are discussed in IAB addressed the interoperability and connectivity aspects. We will discuss these 3 communication patterns now.

Device-to-Device. The devices are connected to each other directly as shown in Figure 2.1 This architecture usually does not require an internet connection as the devices are connected to each other in a proximity based way such devices that are installed in the same building or the same area. The network technologies that are used in this architecture are cheap in terms of communication cost and they can be RFID, WIFI, Bluetooth, Zigbee and Z-wave.

This pattern can be either a client/server model or decentralization architecture model. In the client/server model, the server will be the single point for having the major services and data analytics. However, the client will be computers that are connected to the server which means they are highly dependent on the server for services and resources. The drawback is that it has a single controlling unit: if it breaks down the whole system breaks down.

In a decentralised architecture, there is no single point for data analytics and decision making. This means that each device has its capabilities to make a decision and process the data when it is possible depending on the characteristics of the device. Then, the system behaviour will be aggregated from the individual device's decisions. This architecture does not have a single controlling unit which means if a node is down, then other nodes will carry on processing the services.

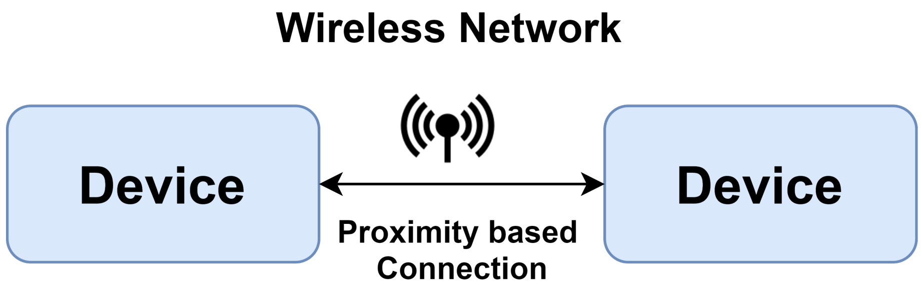


Figure 2.1: Device-to-Device architecture

Device-to-Cloud. In this architecture, the devices will be connected directly to the cloud via an internet based connection by using standard protocol including IP protocols and transports like TCP or UDP, also constrained protocols like COAP as presented in Figure 2.2. The IoT devices have a direct connection to the cloud for accessing the services and data. We observe a software and data analytics architecture approach for the Internet of Things as follows in a new paragraph:

The traditional cloud computing architecture which is placing all the main functionalities in one single control point which is a cloud platform that provides services. In this architecture, the devices send raw data to the cloud which has unlimited capabilities to collect, process and store data from many services and applications. This means that the cloud computing platform has full control of all the data. There are advantages and disadvantages of this architecture as every architecture has. The advantages of using device-to-cloud computing architecture are as follows. Firstly, it is easy to reach and get the required service without requiring specific hardware requirements and the availability of the service provider. Secondly, the maintenance of the application is easy as the cloud is a single point of control

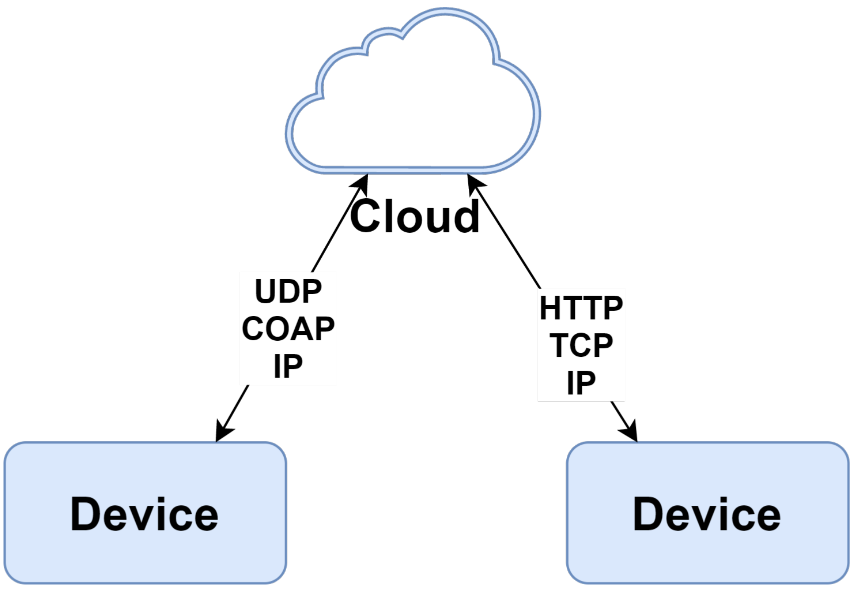


Figure 2.2: Device-to-Cloud architecture

and centralised, the system will do maintenance on one side and it will be done by all sides. Thirdly, the powerful super computing devices that reside in the cloud are capable of doing data analytics easily and efficiently. These powerful devices are too expensive to install at many different locations.

However, the architecture has several disadvantages in terms of processing the data as follows. The first drawback is that all raw data will be sent to the cloud from the IoT devices which introduce several issues such as increasing the cost of data communication over the network. This from a user perspective is losing the control of data raises privacy concerns like who will see the data and will the data be kept forever and how safe is the data. Also, unused and unnecessary data will be transmitted with raw data which will consume storage and slow the data preprocessing stage by eliminating them. For example, in a car parking area, one of the parking slots is free for approximately 20 hours and the system will send the information of free slot is available for 20 hours to the cloud, which consumes the network by sending the same data.

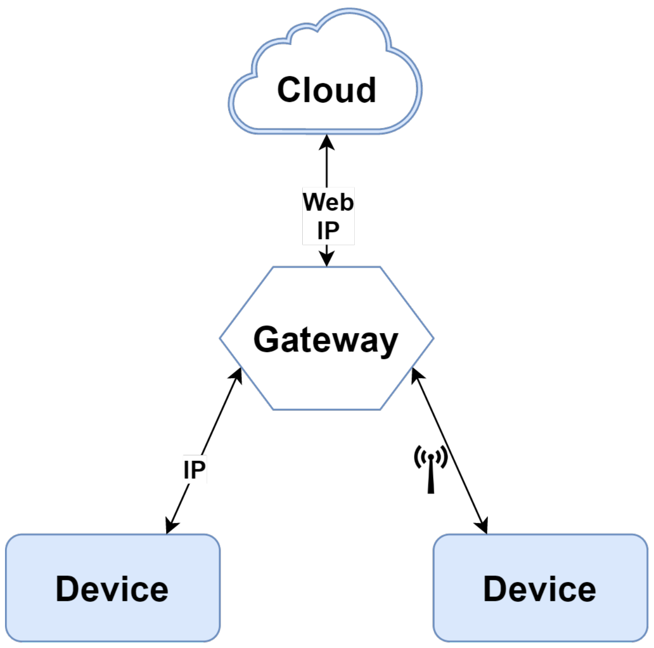


Figure 2.3: Device-Gateway-Cloud architecture

Device-Gateway-Cloud. The main purpose of this architecture is to provide an internet connections to IoT devices which are not embedded or have internet connection. Figure 2.3 shows the connections among the IoT devices, gateway and the cloud.

The devices of the gateway can be either personal computer, mobile phone or laptop. The role of the gateway can shape the architecture. The gateway might only transfer data from device to cloud which is like in the device-to-cloud architecture or it may have additional functionalities. Gateway indeed supports the IoT devices, but from an IoT solution perspective if the gateway is used only for transferring the data between the IoT devices and the cloud, then it might come up with the conclusion that which architecture should be chosen device to cloud or device to gateway to cloud. It is not an easy question to answer as it depends on the solution and IoT environments. For example, if some of the IoT devices do not have direct internet connection, then obviously the gateway will be helpful.

2.3 Fog Computing

Cisco is the earliest company that introduced the term fog computing [23]. Since then, many researchers have defined fog computing from various perspectives. One of the general definitions was was given in [110], Fog Computing is "*a geographically distributed computing architecture with a resource pool which consists of one or more ubiquitously connected heterogeneous devices (including edge devices) at the edge of the network and not exclusively seamlessly backed by cloud services, to collaboratively provide elastic computation, storage and communication (and many other new services and tasks) in isolated environments to a large scale of clients in proximity*". Another definition by the authors of [95] of fog computing is "*a scenario where a huge number of heterogeneous (wireless and sometimes autonomous) ubiquitous and decentralised devices communicate and potentially cooperate among them and with the network to perform storage and processing tasks without the intervention of third parties. These tasks can be for supporting basic network functions or new services and applications that run in a sandboxed environment. Users leasing part of their devices to host these services get incentives for doing so*". Also, fog computing has been defined by the Open Fog Consortium [32] as "*a system-level horizontal architecture that distributes resources and services of computing, storage, control and networking anywhere along the continuum from Cloud to Things*". Furthermore, a definition by [23] "*Fog computing is a highly virtualized platform that provides compute, storage, and networking services between IoT devices* *and traditional cloud computing data centres, typically, but not exclusively located at the edge of network*".

In this thesis, we consider Fog computing as an expansion of cloud computing located closer to the IoT data source. Fog computing stands between end devices and the cloud and it brings most of the services like data processing, services of network and storage as shown in Figure 2.4. The fog devices can be distributed to any place in the network and it can be any type of device that has computing power, internet connection and storage [92, 97]. The characteristics of fog computing are extracted from the following papers [14, 110] namely awareness of location, low latency, distributed geographically, scalability, mobility, real time response, heterogeneity, interoperability, and data analytics.

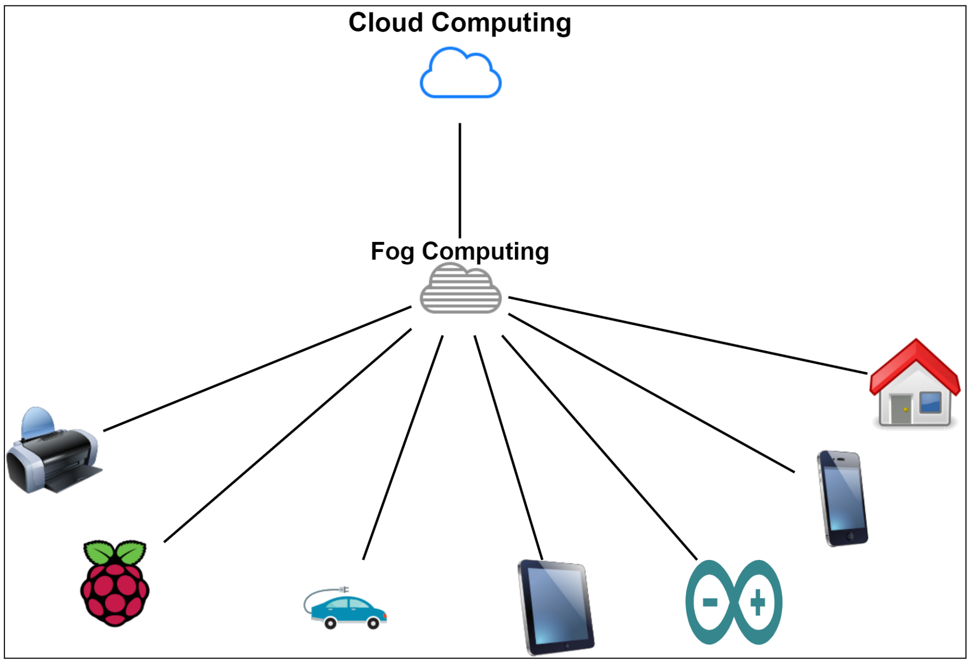


Figure 2.4: Fog computing

Fog computing brings some of the services of the cloud service provider near to the data source. The fog devices have limited computational power, storage and services and they are located in the middle of the IoT devices and the cloud. The major aim of fog computing is to reduce the latency for real-time applications in the IoT [88]. The fog computing architecture has six layers namely from the top to the bottom "*transport, security, temporary storage, pre-processing, monitoring, physical and virtualisation layer*" [8, 63, 65].

Fog computing is an infrastructure that involves distributed computing techniques that allow for remote control of different application services. There are a number of application services that are controlled remotely in the cloud.

Fog computing has a data plane that permits services to be located near to the data source contrary to servers in centres of data [4]. IoT concept can be supported by fog computing by connecting the devices to each other devices that are used by people like mobile and wearable devices.

Fog computing can help to analyse the data and produce insights from the data near to the data source. In addition, the fog computing takes applications, elements of data and computational power near to the data source possibly to the edge of the network where it is far from the cloud.

2.3.1 Fog and Edge Computing

Much of the literature mentions both fog computing and edge computing interchangeably [9, 31, 43], with some papers stating that edge is a synonym of fog computing [111]. Additionally, the author of [5] stated "*The term Fog computing or Edge Computing means that rather than hosting and working from a centralised cloud, Fog systems operate on network ends. It is a term for placing some processes and resources at the edge of the cloud,* *instead of establishing channels for cloud storage and utilisation*". Both fog and edge paradigms agree on the concept of moving the computation as much possible from a centralised level to distributed or decentralised levels. However, authors of [64] stated that they differ in terms of the radio access network, with fog computing involving Wireless LAN (WLAN) or cellular networks, but edge computing is usually cellular.

2.4 IoT Data Analytics

Connecting all smart objects directly to the internet has become unfeasible by the sheer amount of such objects and the amount of data that they collect.

All the sensors that are distributed over the world will generate a huge number of data that will be gathered, processed and analysed to produce insights and useful information. As a result of improvement in the technology of sensors, they are more powerful, inexpensive and smaller [74]. The emergence of the IoT concept is after the emergence of sensor technology, but the use of sensors has been limited. The IoT has extended the use of sensors by several processes including data gathering, data transmission and data processing.

Processing IoT data means to add value to the raw data by extracting important aspects and creating meaningful information – an essential element of the IoT [107], [18] identifies five steps to follow when processing IoT data, namely data collection, data pre-processing, transformation of data, mining and evaluation. Considering that "*data fusion and mining present an efficient way to manipulate, integrate, manage and preserve mass data collected from various things*" [109]. Data Fusion is a useful method to get richer knowledge about the environment and generate more precise data by aggregating and integrating sensor data. In IoT, data fusion will be vital because there will be many sensors that are available. Due to the many different sources that exist to give the same information. In addition, data filtering is a beneficial method to avoid the great number of data that are going to be transmitted over the network via filtering the sensor data. The five steps (Collection, Collation, Evaluation, Decide, and Act) are important to reach the goal of data fusion.

- The Collection step: "*collects raw data from sensors and other IoT data sources (Social media, smart city infrastructure, mobile devices etc.)*" [101].
- The Collation step: "*analyse, compare and correlate the collected data*" [101].
- The Evaluation step: "*fuses the data in order to understand and provide a full view of the environment*" [101].
- The Decide step: "*decides the actions that need to be taken*" [101].
- The Act step : "*simply applies the actions decided at the previous step. The Act step includes actuator control as well as sensor calibration and re-configuration*" [101].

Data fusion: Data Fusion is defined as "*the theory, techniques and tools which are used for combining sensor data, or data derived from sensory data, into a common representational format*" [60].

We are specifically interested in data fusion, which fits into the area of data pre-processing and transformation and allows us to reduce the volume of data but increase its value. Data fusion is referred to by other 'synonyms' such as information fusion, decision fusion, data combination, multi-sensor data fusion, sensor fusion and data aggregation. While there is no general agreement on these terms, there are some differences that can be observed: in some cases data fusion is applied on raw sensor data while information fusion is used to determine analysed data, meaning that the latter has a higher semantic grade than data fusion[27]. Similarly, data fusion and data aggregation are utilised mutually, however, they are diverse in some points. Similarly, data fusion techniques are used to integrate data from a variety of sources to produce more meaningful and effective inferences and associations, whereas data aggregation can be considered as a sub component of data fusion which summarises the sensor data to remove data redundancy [10]. The most common definitions by researchers are as follows:

- Data fusion is defined by the Joint Directors of Laboratories (JDL) workshop [105] as "*a multi-level process dealing with the association, correlation, combination of data and information from single and multiple sources to achieve refined position, identify estimates and complete and timely assessments of situations, threats and their significance.*"

- Hall and Llinas [44] say that "*data fusion techniques combine data from multiple sensors and related information from associated databases to achieve improved accuracy and more specific inferences than could be achieved by the use of a single sensor alone*".

Informally, our working definition of data fusion is that it aggregates and integrates all sensor data to allow obtaining accurate and meaningful data while eliminating redundant and uninformative data. Data fusion can be classified depending on a variety of attributes as shown in Figure 2.5 [30]. These attributes are discussed in detail in [10] and generally capture the idea that there are different dimensions such as the abstraction level or the relation between the data items from one or multiple sensors.

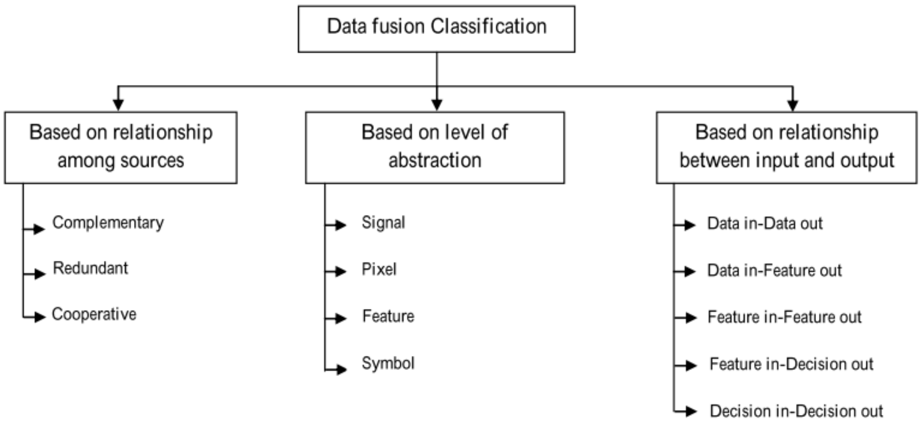


Figure 2.5: Data fusion classification

[27] has divided the data fusion techniques into several criteria as follows.

- Classification based on relations among data inputs, as proposed by [37]. They can be determined as complementary, redundant and co-operative data.

- Classification based on the data input and output as proposed by Dasarathy [34]. Dasarathy's data fusion classification system formalises the attributes just discussed and can be considered as one of the most common approaches [34]. Dasarathy's classification focuses on details of input and output based on the abstraction level. The classification contains five classes as follows [27]):

Data In-Data Out (DAI-DAO) is the primary method of data fusion in the classification model. It processes the raw data that is collected directly from sensors resulting in more accurate data. In addition, image and signal processing algorithms can be used at this stage.

Data In-Feature Out (DAI-FEO) processes the raw data to produce features which can depict a structure about the environment.

Feature In-Feature Out (FEI-FEO) processes a collection of features to get more effective feature results.

Feature In-Decision out (FEI-DEO) processes the features to acquire a collection of decisions.

Decision In-Decision Out (DEI-DEO) processes the decisions to extract more efficient decisions.

- Classification based on "*an abstraction level of the employed data including raw measurement, signals and characteristics or decisions*". Furthermore, [27] mentioned that Another data fusion classification based on abstraction level can be considered which contains "*low level fusion, medium level fusion, high level fusion and multiple level fusion*".

- Classification based on various levels of data fusion. In addition, the are five processing levels in the process of data fusion [105] including "*Source preprocessing, object refinement, situation assessment, threat assessment and process refinement*". This classification is described in detail in [27].

- Classification based on the architecture types.

Features are defined as the single measurements that are used to create the training model. In other words, they are the columns of data that are created for the training set [45]. In addition, data fusion can provide the required knowledge that is essential in a decision-making process, therefore, the amount of the available knowledge/data can affect the final decision at any stage. Many techniques use symbolic information and the data fusion process to determine the uncertainties and restrictions that are part of / effect the decision-making process [27]. In other words, the decision can be captured depending on the knowledge of the events that are collected from variety of sources by fusing them.

Data fusion is an active area in research and business. There are several data fusion techniques that focus on reducing the consumption of energy in [10, 36]. They have used a variety of methods including fuzzy set theory and neural networks. They succeeded in terms of removing redundancies while fusing the data. However, they did not focus on the resource constraints of devices that embed the sensors. In contrast, they assume that these devices work efficiently without a need to pay attention to their limitations. More importantly, these mechanisms send all the data to centralised computation systems, which affects the data communication cost, privacy and energy as well. As we could see in our experiments in chapter **??**, sending the raw data to the cloud is not efficient in terms of data communication over the network. This means that the effectiveness of data processing depends on the selected architecture which will be discussed below.

2.4.1 Data analytics architectures

Understanding what data fusion can achieve, one also needs to consider the architectural aspect of where data fusion should be done within the overall architecture of the IoT. The author of [27] considers three options including a centralised, decentralised or distributed architecture as follows:

- Centralised architecture: all the collected data from sensors will be sent to the cloud for the processing which means that everything is held in one single server. It is known that the cloud is capable of processing very large amounts of data effectively. However, in real time scenarios, data consumption over the network will be high, which will make the cloud not sufficient for an effective fusion of the data. This architecture is also very problematic if the data consists of images such as earth observation imagery. The reason is that there will be more delays in terms of data arrival time and this will impact badly on the output of data. Additionally, privacy will be one of the main issues because this architecture receives all the raw data without applying any reduction or aggregation previously. Finally, energy consumption has been important in the IoT because transferring raw data all time from devices using any network such as 3G and WIFI will consume significant amounts of energy.

- Decentralised architecture: there are several nodes in the network and each of them has their specific computation capabilities, so there is no single server like a centralised system. Every node applies data aggregation autonomously to its local data and data received from peers. One of the major limitations of this architecture is the high communication cost between peers. In this case, if we increase the number of nodes, then there might be a lack of scalability.

- Distributed architecture: sensor readings are processed at the source level before applying data aggregation in a specific node that is capable of data fusion. This can overcome various issues of the centralised architecture and can reduce communication costs over the decentralised architecture.

It is not possible to say that one of these architectures is the best, as it often depends on specific requirements and technology. Both decentralised and distributed architectures are quite similar to each other in many ways. However, they differ in terms of the place for pre-processing the data. In decentralised architectures the whole data aggregation happens in every node which produces comprehensive output. Whereas, in distributed architectures the raw data is firstly pre-processed at source to extract features, and then these features are fused. The main advantages of the distributed architecture over the centralised one are reducing the processing and communication costs because it pre-processes the data in a distributed manner before fusing data [27].

It is generally accepted that increasing accuracy and reducing energy usage are major aspects of data fusion [30], so any architecture that is presented needs to consider these aspects.

As there are obvious trade-offs between the different architectures it seems desirable to formulate solutions which combine the different ideas in ways that reduce the disadvantages and benefit from the advantages of each. Our method presented below attempts to achieve this.

There are two types of data processing the first one is in-network side data processing and the second one is cloud side data processing [101]. It is expected that the number of sensors that are distributed will be increased as well as the number of internet-connected devices over the next decade.

When there are multiple data sources and there is a need to integrate them, then collecting and analysing data in a centralised way is important. In the IoT, optimising the cost and performance requires an examination of connectivity, infrastructure and data analytics [76].

In IoT, the number of internet-connected devices increases which means that a significant amount of data will be produced that leads to IoT big data. Many of these data will be produced by video cameras that are becoming much more popular among people [81]. In 2013 in the UK, there was approximately a surveillance camera for nearly 11 people [22]. For this reason, data processing including data fusion and filtering have become an interesting topic in the research area. big data was defined in 2010 by Apache Hadoop as "*datasets which could not be captured, managed, and processed by general computers within an acceptable scope*". Big data is not a new term in computer science [113], it has been created by big technological companies like Yahoo, Microsoft and Google. Recently, big data concept has become the most challenging issue in computer science, particularly in the IoT. Despite the popularity of big data, there are many challenges that people in both academia and industry face such as privacy, scalability, heterogeneity and timeliness of data [96]. In addition, there 3 characteristics of big data including volume variety and velocity [113].

- Volume: "*Volume relates to the size of the data such as terabytes (TB), petabytes (PB), zettabytes (ZB), etc*" [113].

- Variety: "*Variety means the types of data. In addition, different* *sources can produce big data such as sensors, devices, social networks, the web, mobile phones, etc. There- fore, data could be web logs, RFID sensor readings, unstructured social networking data, streamed video and audio, and so on*" [113].

- Velocity: "*This means how frequently the data is generated, for example, every millisecond, second, minute, hour, day, week, month, or year. Processing frequency may also differ with user requirements. Some data needs to be processed in real-time and some may only be processed when needed. Typically, we can identify three main categories: occasional, frequent, and real-time*" [113].

This big data will be valuable when it becomes understandable to make a decision. This means there is a necessity to process large data to extract meaningful information for creating insights [96]. To have insight from processing data there are two main phases including big data management (collecting, data analysis preparation and saving) and big data analytics (analysis on data and obtaining reasoning) [96].

It can be noted that there is a need for IoT middleware solutions to help with solving these problems. According to [89], an IoT middleware can have capabilities namely context-aware, interoperability, control data volumes, device management, security and privacy. In addition, a recent study by [80] stated that IoT middleware should provide "*data management services to applications, including data acquisition, data processing (including preprocessing), and data storage. Preprocessing may include data filtering, data compression, and data aggregation*". Several most common projects are discussed in [35]. Most of these projects are focused on REST, OWL, Web services and SOA which are web technologies.

Most of the research until now focused on processing the data on the cloud. There is no doubt that the cloud has an important role in enabling the IoT since it has high power processing and storage [114]. However, one of the challenges that the cloud faces the accumulation of data that comes from many heterogeneous data particularly from cameras data [84]. It is notable that most of the use cases in the IoT such as smart transportation and smart cities are distributed naturally. This means a fourth characteristic namely geo-distribution should be added to big data's three characteristics (Volume, Velocity and Variety) [20]. Therefore, a smart distributed IoT platform is required that can manage a distributed system at the edge [22]. More importantly, data processing could be improved by using data management IoT middleware in in-network level (edge and fog computing). Edge analytics in real time on data can help with improving the time and value of the collected data. This view is supported by [13] who writes that moving the computation and storage close to data producing sources at the edge level of networks. Edge analytics can be supported by fog computing which can analyse, process and minimise data volume before sending to the cloud, therefore there will be less delay and bandwidth usage [110]. Data fusion is an active area in research and business particularly with a view to optimised data analytics. There are several data fusion techniques that focus on reducing the consumption of energy in [11, 36]. They have used a variety of methods including fuzzy set theory and neural networks. They succeeded in terms of removing redundancies while fusing the data. However, they did not focus on the resource constraints of devices that embed the sensors. In contrast, they assume that these devices work efficiently without a need to pay attention to their limitations. More importantly, these mechanisms send all the data to centralised computation systems, which affects the data communication cost, privacy and energy as well. As we could see in our experiments sending the raw data to the cloud is not efficient in terms of data communication over the network.

As discussed earlier the IoT devices will generate a great number of data which will create problems from several perspectives such as consuming the network with sending raw data to the cloud. In addition, energy is another factor that we need to consider while sending data to the cloud for processing. Moreover, the privacy and quality of data or accuracy are crucial to control private data and produce valuable information. It is true that relying more on local analytics or more on the cloud can cause problems. In this regard, the balance could be significant in terms of energy and data consumption, privacy and quality of data. The present study is motivated by the need to take into consideration the data processing locally before being sent to the cloud. There are many advantages of this approach such as it reduces the data transmission cost over the network, reduces energy consumption and increases privacy by processing and storing data locally.

2.5 Analysis of existing approaches for Fog Computing

This section reviews the current state of the art from the two aspects relevant to the problem definition: First, we review various computation distribution strategies employed in fog computing and the associated architecture implementations. Second, service decomposition as has been researched in the wider web services paradigm.

The architectural aspect related to the location of the data processing is important. Data processing can be applied in various architectures including centralised, decentralised and distributed.

In recent years, there has been an increase in the amount of literature on distributed architecture in the IoT. One of the attempted proposals is by [93]; they proposed distributed architecture for fog computing for analysing big data in a smart city. They distribute the smartness to the devices in edge and computation at every layer which executes applications that have latency awareness. A fog computing based face resolution framework is proposed in [47] which obtains the information by analysing a facial image. There are several features of this framework, including reduced data communication over the network and the response time of resolution, also efficiently solving the issues with bandwidth. A proposal in the distributed analysis by [112] which is a model that combines processing power which is at network level including edge and data centres to process and analyse the data from the collection point to a destination. Furthermore, in [86], personal modal training method has been proposed in which the data processing, particularly machine learning is applied to private data in devices that have constraints and raspberry pi was used to test the feasibility of the IoT device in the implementation of such methods. Authors of [102] proposed a framework for managing edge nodes which is called Edge NODE Resource Management (ENORM). In addition, they proposed several techniques that provision edge node resources. They used an online game called PokeMon Go-like to check the feasibility of their framework. Their results show that by using ENROM the application latency is reduced between 20 - 80%. In [57] the authors proposed an approach (latency aware) to place application modules on fog nodes to make sure that the service delivery satisfies the deadline for diverse applications. They modelled and evaluated their policy in Fog environment that is simulated in iFogSim [42]. In resource allocation for fog computing, authors of [68] proposed an effective resource allocation approach depending on Priced Timed Petri Nets (PTPN) for fog computing influenced by online shopping sales. Furthermore, users can select the required resources dynamically from already allocated resources. Also, they showed an algorithm that predicts the cost of time and price for finishing jobs relies on PTPN structure.

Table 2.1 shows a summary of the related work on fog computing, which are reviewed along with the following aspects:

- Data Modality: Format and modality of the data being used for implementation and validation.
- Fog Node: Types of fog nodes that are considered (user phones, low power embedded devices, general purpose computers, high-performance computers etc.)
- Fog node functionality: Functions that the fog nodes perform such as data sensing and pre-processing tasks, computation offloaded from the cloud.
- Distribution Strategy: Strategies used to distribute services, workload or computation to fog nodes from other nodes or the cloud.
- Application: Context aspects considered in the computation/service distribution.

It can be seen that several data modalities and formats are used, ranging over numerical, text and image data. Different types of devices are used as fog devices to perform different functions. The distribution strategy used is typically tied to the application scope and focus, ranging from data analysis parallelization and computation offloading to different optimisation strategies.

In web service composition, there is one proposal that focuses on autonomous web service composition by taking care of service constraints automatically [103]. It is important to be aware of the constraints on services, but in the IoT, there are nodes which have various constraints. This means that there is a need to have a constraint awareness approach for both services and nodes. In addition, [90] investigated the service composition's requirements and the way to obtain a composite service using the transport domain as an example. They provided several scenario based approaches to service composition and discussed these. Additionally, authors of [28] proposed a comprehensive device collaboration model which has four layers, namely device, device-oriented web service (doWS), resource and process. This model shows the possibility of integration between devices and web services, and also the devices can be considered as active actors because the data is not sent immediately to servers. The authors of [99] proposed a service based model in requirements decomposition. Their model process starts with user requirements which are defined as goals, service discovery and discovered services employed to check the feasibility of the preceding decomposition. They decomposed the requirements of three web service composition cases to validate their approach. Another proposal in service decomposition domain by [21] who proposed a greedy algorithm that decomposes interface into interfaces depending on cohesion. This approach mainly focused on improving the cohesion and it was successful in that

Table 2.1: Summary of related fog computing work

Work	Data Modality	Fog Node	Fog Node Functionality	Distribution Strategy	Application
B. Tang, et al. [93]	Real time Temperature sensing data	Parallelized small computing nodes	Data sensing, Pre-processing and data analysis	The workload of Data analysis parallelized between edge nodes, parallel computing mechanism.	Latency aware and location aware
P. Hu, et al. [47]	Three public face databases are used including Georgia Tech, Caltech and BioID	Personal computer and Laptop	Face Detection, Pre-processing, data analysis and computation offloaded from cloud	part of the computation tasks is offloaded from cloud to fog nodes.	Latency aware and network transmission sensitive
A. R. Zamani et al. [112]	simulated data	Virtual Machines with resource limitations	Data Sampling and Camera Aggregator.	Resource federation model. Workload distribution based on value of data. Job Scheduling optimization strategy.	Quality of Service, minimizing the computation cost and time of processing the job. Reduce data transfer time.
Servia-Rodriguez et al. [86]	WISDM dataset [55] (Numerical data), Wikipedia [2] and NIPS [6] datasets (Text data)	Raspberry pi and Personal devices	Supervised and Unsupervised Learning.	Share model (Training model) is distributed from cloud to personal devices to create personal model	Privacy preserving, improving accuracy and maintaining Efficiency.
Ni, Lina, et al. [68]	Simulated data	Linux based Computers	Fog can select satisfying resource from previously allocated resources	resource allocation strategy based on Priced Timed Petri Nets (PTPN)	Aware of task completion price and time.
N. Wang, et al. [102]	Open dataset and iPokeMon game [1].	Resource Allocation, request managing, communication latency monitoring, allocate or deallocate resources to containers.	Linux based Containers	Offloading workloads from cloud to fog.	Latency Aware
R. Mahmud et al. [57]	Simulated data	Heterogenous nodes in modelled environment	Intermediate layer between cloud and IoT devices	Optimization of resources	Latency Aware
D. G. Roy et al. [82]	Experimental data	Laptop, Personal Computer and mobiles.	Cloudlet agent, offloading tasks to mobile phones.	Offloading applications from cloud to multi-cloudlet.	Latency Aware and power consumption is reduced

regard, but aspects like coupling between interfaces are not considered .

Furthermore, authors of [52] proposed quality of service aware and energy centred service selection algorithm for service composition in the Internet of Things. The idea of the algorithm is that it is possible to save energy by decreasing the degree of quality of service while maintaining the expectation of the user. The algorithm has two phases, the first one performs pre-selecting the services by proposing a quality of service degree which meets the user's needs. In the second phase, in order to select the best option among selected services, a relative dominance relation is used for the process of service composition. Additionally, [81] proposed a method to perform data fusion via a service composition model of DOHA (Dynamic Open Home-Automation) which is SOA-based middleware in a distributed manner. Every service is liable to get data from outer services using composite processes to control, fuse or create new information. In the implementation, they used DPWS (Device Profile Web Service) which is a framework that develops lightweight service for constrained devices. This framework put restrictions on web service specifications that allow the web services to run on resource-constrained devices, for example, the size of messages. In [15] authors investigated the possibility of building complex services in the IoT environment. They showed the SYNAISTHISI IoT platform that is able to combine services, devices and people with systems. The services in this platform are enriched semantically using ontologies. They developed a context ontology model for smart meeting spaces and presented the way that the developer can create a service that defines the number of people inside an intelligent room. Authors of [87] examined the problems of microservices granularity and how it affects the latency. They simulated the deployment of microservices with two approaches including microservices in one container and microservices divided into several containers. They observed a slight increase in latency for several containers over single container deployments.

Our review of the existing literature shows that most of the proposed models, approaches and architectures do not take into account the resource constraints of the IoT devices. We acknowledge that many works have been done on data processing in resources constraint devices, although none of them have proposed service decomposition as a viable solution. The proposals in the service computing domain usually do not focus on the deployment of services on nodes by considering the constraints of devices; instead mostly focusing on the quality of services and constraint requirements of services. However, in the case of IoT systems, there will be many constrained devices distributed across the network. This means it is important to decompose the resource-heavy services into smaller micro-linked services which can be distributed to and handled by constrained devices.

2.6 Evaluating IoT Systems

The Internet of Things aims to digitalise everyday physical objects by connecting them to the internet. As a result, cyber-physical environments of multiple sizes emerge, imposing new requirements on applications and software systems in regards to heterogeneity and volatility. A challenging stage in the engineering of these systems is the validation. The validation of such systems is complex because of three key differential characteristics of the IoT environments: The heterogeneity, volatility and the size variety. In order to validate software solutions, researchers use one or a variety of techniques including real-world and simulations.

Validation in real-world testbeds is always preferred as it enables the replication of the cyber-physical conditions that are present in production environments. For example, the testbed used in [5] incorporates a ZigBee network with workstations, coordinators, readers and up to 200 objects with active RFID tags. The major drawback of this type of setting is the cost. Despite the low cost of hardware platforms for IoT devices, not all researchers have access to a real testbed with a sufficient number of IoT devices. Besides, the configuration, management and running of the platforms, supporting these testbeds, is also time consuming. Therefore, usually, the real-world settings are constrained and scenarios lack some or all of the key differential characteristics of IoT, namely: Heterogeneity, volatility and size variety. In the last years, a number of IoT experimental research facilities with support for medium/large numbers of devices have appeared, offering an infrastructure for evaluating solutions atop of the offered services. These platforms offer services that reduce the effort required to evaluate a particular solution, however, the price is that application developers must conform to a particular development and operation model. e.g. IoT devices are merely data feeders and cloud infrastructure concentrates data storage and processing. In case of evaluating decentralised architectures that require edge processing the usefulness of these platforms is reduced. This is a problem since the IoT research agenda includes the development of decentralised solutions that should run in a combination of fog/edge and cloud contexts, therefore environments for evaluating these solutions are required. Besides, simulations enable validation under multiple conditions, defining a model that offers a partial representation of the real-world. One of the most common uses of simulations is to validate systems' scalability to medium/large number of IoT devices, however other characteristics of heterogeneity of IoT devices and volatility.

2.7 Datasets

To evaluate a system, there is a need to either build a personal dataset or use the existing publicly available datasets to apply the experiments of the proposed solutions. We have selected a number of public datasets that are different in terms of data types including sensors data, numerical data, text data and image data. In IoT environments, there will be devices that collect video and image data like drones and surveillance cameras. Additionally, there will be devices that collect data from sensors either in numerical or text format. The datasets that we have used in this research are representative datasets for IoT environments. We have selected different types of data modalities to explore how an IoT approach will handle each type of datasets. The collected data will be analyzed by using different data analytics techniques which will indicate the trade-offs among the datasets in IoT environments.

The datasets that are used in this research are as follows:

- The WISDM dataset [55] is a set of accelerometer data on mobiles (particularly Android-based) from 36 users who are doing 6 activities (walking, jogging, climbing upstairs, descending downstairs, sitting and standing). These users carried their mobiles while they were performing these activities for a fixed time. We divided the data into 10 second chunks. In addition, 43 features are created depending on 200 readings, where each reading has three acceleration values (x, y and z), within the specified chunks. The transformed data contain 5418 accelerometer traces from the 36 users, with average 150.50 traces per user and a standard deviation of 44.73.

- The dataset is called Twenty Newsgroups [56] is a dataset of nearly 20,000 newsgroup files. This dataset is collected for a different purpose, but it has become a standard dataset for text analysis in machine learning environments. In addition, the data are divided into 20 newsgroups and each group has a different topic.
- Dogs vs. Cats dataset [49] which was a competition from Kaggle, it is a collection of pictures of both dogs and cats. In addition, the purpose is to have the classification of cat and dog, so we can know if the picture has a dog or cat. The dataset is divided into two parts including training and testing data. The total number of images in the dataset is 25000 which is equal to 570 MB.

2.8 Summary

This chapter discussed the concepts of the background including the IoT, architectures, fog computing, data analytics used in this research. We reviewed existing approaches in the field of IoT and fog computing, also data analytics architectures. Besides, evaluating an IoT system is one the topics that are covered in this chapter, as evaluation of a system is important to reach effective results. Finally, a number of publicly available datasets are presented and explained, and some of the discussed datasets will be used in the later experiments.

Chapter 3

An Alternative Architecture to Data Analytics for The IoT

3.1 Introduction

In this chapter, we develop a method in which fog level and cloud level processing can work together to build an effective IoT data analytics in order to overcome their respective weaknesses and to use their specific strengths. Specifically, we will collect raw data locally and extract features by applying data fusion techniques on the data of resource constrained devices to reduce the data and then send the extracted features to the cloud for processing. We will evaluate the accuracy and data consumption over the network and thus show that it is feasible to increase privacy and maintain accuracy while reducing data communication demands.

The rest of this chapter is organised as follows: section 2 identifies the challenges and requirements, followed by the main contribution in section 3. Then, we describe the solution space in section 4 while section 5 evaluates our solution. Before concluding the chapter, we present the results of the experiment and the evaluation.

Parts of this chapter have been published in [16].

3.2 Contribution

We propose a hybrid approach that moves some processing off the cloud and leads to savings in data transfer and changes to accuracy. In the proposed work, we are fusing and filtering data close to the source and then send meaningful higher level data rather than raw data to the cloud. As fewer details are being transmitted some privacy protection is already taking place – however further work in studying the privacy angle needs to be undertaken.

This data fusion technique will directly influence the collation and evaluation steps and hence the crucial question arising is: How can we fuse sensors data locally without harming the accuracy of the overall decision? A secondary question is considering the feasibility of distributing the processing considering that many network and edge devices have less processing power.

The novel contributions of this chapter are:

- Choosing the location of processing the data in the network is important to maintain the usage of the network and the efficiency of data processing. Therefore, we propose an efficient approach, which moves the computation as much as possible to the fog/ edge side of the network.

- Moving part of the data processing near to the data source is crucial as the IoT devices are constrained and processing some of the services might require high computational power based on the experiments. As a result, we explore the feasibility of applying data fusion techniques via resource constrained devices, like the Raspberry Pi 3 Model B.

- We note that applying data fusion on data in IoT devices near to the data source is important when dealing with large data, so fusing the large data before sending it to the cloud can maintain the network usage by sending only more meaningful data.

- We extensively evaluate the approach using the WISDM dataset [55] and five popular data analytics techniques. Also, we evaluate the efficiency of our proposed architecture with the efficiency of the traditional centralised architecture.

3.3 A hybrid approach for data analytics for the IoT

3.3.1 Overview

We propose a hybrid approach that moves the computation as much as possible from the cloud to the fog/edge level. The overview of this approach is demonstrated in Figure 3.1. We begin with applying data fusion techniques on sensors data to minimise the number of data points and extract features in the IoT devices. Then, we extract features from this data and send it to the edge/fog node. This step is important because it is widely accepted that raw time-series data cannot be efficiently analysed by algorithms for classification. After that, the features will be sent to the cloud for training purposes and to create inferences.

We used an experimental approach because it is about getting insights and understanding whether to place the data on IoT, fog or Cloud resources with executing the data multiple times on each of these approaches. We have used benchmarking to make data communication and performance comparisons based on latency and execution times that are used in the evaluation of the three approaches fog, cloud and hybrid.

Objective Function

$minimize/maximize \ f_m(x)$

$m = \{$ min−data, min−exec, max−privacy $\}$

f is the objective function, m is the set of objectives and x is a vector of variables.

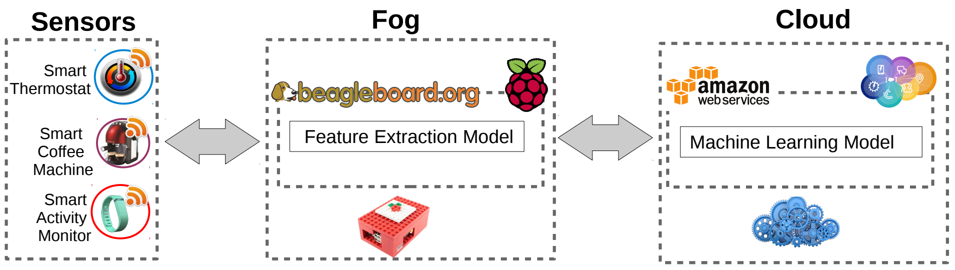


Figure 3.1: Distributed processing in the Internet of Things

3.3.2 Architecture

The architecture of our proposed solution consists of two main parts: First, the cloud level has the responsibility of data training and inference creation. Second, the fog level aggregates the sensor data to reduce data transmission cost over the network. The aim is reduce the data communication over the network, reduce Execution time, increase privacy and maintain accuracy as much as possible by analysing and processing data locally.

The communication between the nodes or peers can be undertaken in different ways as usual (such as WIFI, 3G and any other solutions). Figure 3.1 shows the architecture of the system. The distributed processing architecture contains three types of node including the IoT Devices, fog, and the cloud as follows:

- A **sensor node** is at the lowest level of the system and is typically embedded in physical objects. Sensor nodes are small and cheap in terms of price to make the process of deploying sensors to objects easy and inexpensive. It senses real-world inputs such as motion detection, temperature and so on. These sensor nodes are connected to Fog nodes via wireless or wired communication.

- A **fog node** resides next to the sensors or along the communication path to the cloud and collects sensors data (or data received from a 'downstream' fog node) and applies data fusion techniques to extract features. For our architecture they form the main component. Obviously a fog node has less power and a less global data view than a cloud node and hence it can apply less sophisticated data aggregation algorithms. It sends the transformed and fused data to the cloud for further processing and storage if required (ideally the fog node can make the ultimate decision). In an investigation into energy limitation, the authors of [25] found that these devices have restricted energy for particular tasks.

- **Cloud node**s reside in the cloud and provide the final processing mechanism, obtaining the transformed data from fog nodes. They mainly apply machine learning algorithms and store the data. It is clear that the processing power and storage capability of the cloud is high. This power can be used even more effectively by using the presented approach. According to [25], there is "no direct quantitative limitations to available energy" in the devices which are in the cloud as they are Mains-powered.

3.3.3 Activity recognition using accelerometer traces

To validate our architecture we have used the WISDM [55] data set which is a set of accelerometer data on mobiles (particularly Android based) from 36 users who are doing 6 activities (walking, jogging, climbing upstairs, descending downstairs, sitting and standing). These users carried their mobiles while they were performing these activities for a fixed time.

We divided the data into 10 seconds chunks because we realised that this duration example ensured enough time for capturing the recurrences of movements in the six activities that we mentioned earlier. However, for determination of the optimum value of duration we did not do experiments, but we have done comparison of the results among 10 seconds, 20 seconds and 30 seconds, then the we found that the 10 second duration was marginally better than others. Also, we have checked with other researchers in [59, 69, 104] and 10 second actually turned out to be probably the best on balance. It is what other researchers are using, so it is almost the most common in this domain. In addition, 43 features are created depending on 200 readings within the specified chunks. The transformed data contains 5418 accelerometer traces from the 36 users, with an average 150.50 traces per user and a standard deviation of 44.73.

We conducted 3 sets of experiments: Firstly, we apply analytical algorithms to the transformed data in the cloud to calculate the accuracy of each algorithm and the execution time as a baseline. Secondly, we apply analytical algorithms to the transformed data in a fog gateway to calculate the execution time and to check the feasibility of the resource constraint devices while processing the data. Finally, we apply data aggregation algorithms on the raw data to extract features. The final approach minimises the data as much possible in the fog then sends the transformed data to the cloud for analysis. We measure the accuracy and execution time as well as the data amount sent to the cloud.

We hypothesise that a similar accuracy can be achieved with the third approach without increasing processing time and with significantly reducing network data transmissions.

3.3.4 Assumptions

- In our experiments, we assumed that fog computing nodes have hard constraint on energy, CPU, memory, and storage, however, cloud nodes have soft constraints on energy, CPU, memory, and storage. In other words, fog nodes have limited capabilities, but cloud nodes have unlimited capabilities. Also, the fog nodes can be battery powered, but the cloud nodes are mains-powered.
- The technical requirements of services are identified using experimental method
- The communication among fog nodes will be wireless connection, but the communication between fog nodes and cloud nodes will be 3G, 4g, or LTE. In the experiments we have used fixed upload speed 1 Mbps to mimic the real-world environment.

- The speed of processing a data by a fog or cloud nodes relies on the data size and the types of nodes' configuration. This means that powerful nodes in terms of configuration will be able to process the low size data quicker.

Limitations

- Resources of nodes like energy or battery, memory, CPU and storage, and their usage and allocation were designed to mimic the real existing nodes, but in a simplified way.
- The cloud nodes are considered as unlimited nodes with unlimited capabilities and the energy costs are not considered. The reason is that real-world configurations are complicated when considering various factors, which goes out of the scope of this project and to simplify the algorithms.

Engineering process

- The process starts with preparing the raw datasets to make it ready for further analysis. In this step, the features will be extracted to apply the classification algorithms.
- The capabilities of fog and cloud nodes are preconfigured, and the capabilities of fog nodes are less powerful when compare it with the cloud.
- The technical requirements of classification algorithms and other services are identified using an experimental approach (benchmarking),

depending on the requirements the data is distributed to the appropriate nodes in terms of having the required capabilities to process the services. Also, the capabilities of fog nodes are adjusted after some iterations of the evaluation to fulfil the requirements of the algorithms.

- The feature extraction, data distribution and machine learning algorithms are installed on both fog and cloud nodes to handle the data in both sides.

- Set up a number of different scenarios, and compare them to the results that are gained from them. We look at how the data would process in different architectures with a view to finding out what performance gains can make. Then there was accuracy that we want to see what the impact besides on the accuracy of the insights.

3.3.5 Experimental set up

As mentioned earlier it is not possible to apply classification algorithms on raw data which is time series data. Therefore, there is a need to transform raw data into features [55]. In our experiment, we used a Raspberry Pi 3 model as an example of a low power fog gateway. The used Raspberry Pi has 1GB RAM and runs Raspbian Jessie with Pixel installed as an operating system. In addition, to simulate the cloud device we used a 16GB RAM Linux System. We used the Weka tool on both sides and we adjusted the heap size in both cloud and fog. In the fog the heap size was 650 MB, in the cloud we allowed 8GB RAM for our experiment. Moreover, we used the same data aggregation methods that were used to extract features in [55] to allow for comparability. We run data aggregation methods in the

Raspberry Pi to generate meaningful features depending on 200 readings where each has x,y and z acceleration information.

When we used the statistical measurements that are used in [55] we created 43 features including the average of each axis, the standard deviation of each axis, the average absolute difference of each axis, average resultant acceleration for all axis, the time between peaks of each axis and binned distribution for every axis (10 equal sized bins and totally 30 bins).

After the data is prepared we applied five classification methods from the Weka data mining and machine learning tools. The methods include decision tree (J48), logistic regression, multilayer perceptron, and naive Bayesian. Throughout our experiment we have used 10 fold cross validation.

3.3.6 Results

Figure 3.2 (*a*) shows the accuracy results of the 5 analysis algorithms that we applied on the transformed data. It is clear from the results that the multilayer perceptron has the highest accuracy percentage.

Figure 3.2 (*b*) shows the data communication time over the network from fog (Raspberry Pi) to the cloud. There are two bars visible: one for raw data and the other for transformed data. While applying this experiment the upload speed of the internet was 1 Mbps. It is clear that the fog only device has no data communication cost because the processing happens in the device and no communication to the cloud has to take place, therefore, there is no visible bar in the figure for fog. However, in the cloud approach the amount of raw data communication over the network from fog to cloud is high, whereas in the hybrid approach the transformed data communication over the network from fog to cloud is low. This is not a surprising result that confirms that we can save significantly on data communications by aggregating and pre-processing data early in the chain.

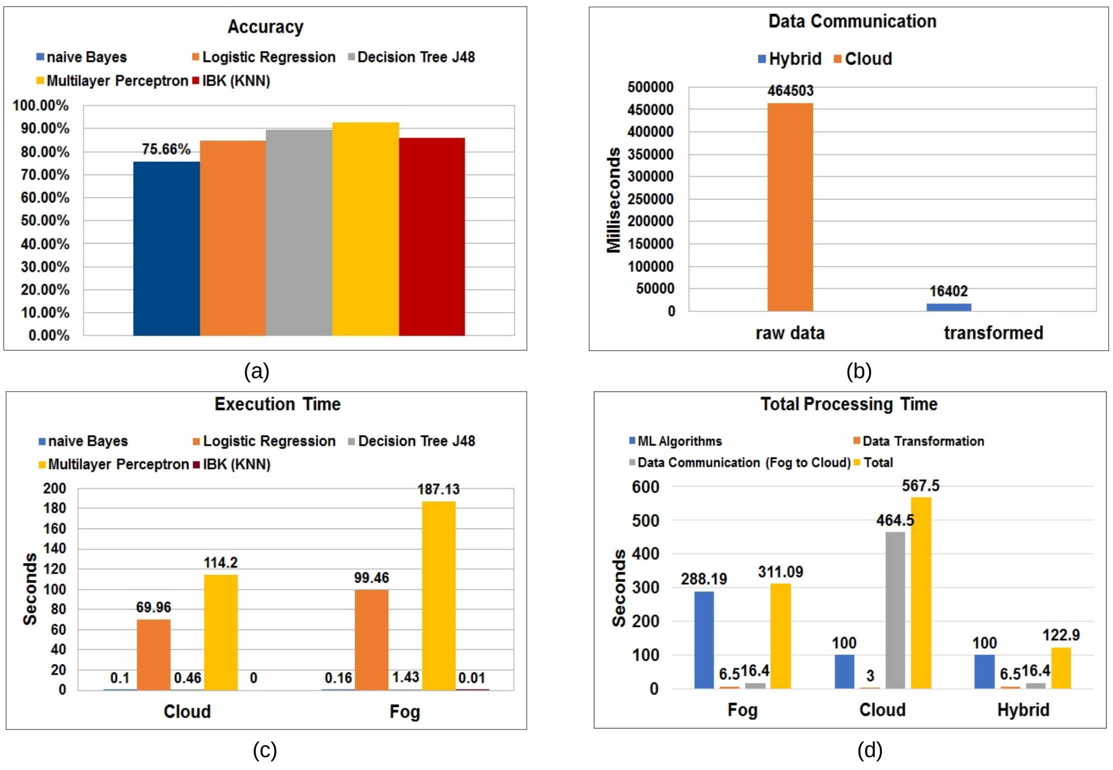


Figure 3.2: Data processing results (a - d)

Figure 3.2 (*c*) illustrates the execution time of the 5 analytics algorithms in both the cloud and fog device. The results show us that two algorithms (logistic regression and multilayer perceptron) have significant differences between the two sides. Obviously, the IoT device takes more time than the cloud to execute analytics algorithms because of its resource constraints.

Figure 3.2 (*d*) demonstrates the total processing time for the three architectures. There are three measurements for each architecture including the execution time of analytics (ML) algorithms, the execution time of the data transformation process and the data communication time between local device and cloud. This graph needs a bit more explanation as the results are more interesting, so the details are as follows:

- Fog (Raspberry PI): The data transformation process is conducted locally and it is clear that the processing time is higher than in the cloud. The analytics algorithms have been processed locally and they took much more time than cloud because of the processing power. However, data communication (the time to send data to the cloud) is very low as only aggregated data is being sent for storage. So, the overall processing time is in the middle of the measured approaches.

- Cloud: Data Communication is the time that the raw data takes from the IoT device to the cloud, which is clearly high as all raw data is being transmitted. The data transformation process was done in the cloud and due to the available resource ran quickly. Also, the analytics algorithms have been processed in the cloud and they took much less time than the fog because of the processing power. However, overall, due to the significant amount of transmission time of the cloud is the slowest in the given setting.

- Hybrid: Here the data transformation process is done locally (in the fog) on the Raspberry Pi, with the usual observation. Data Communication is the time the transformed data takes from the IoT device to cloud which as before is low. Finally, the analytics algorithms have been processed in the cloud on the transformed data they took much less time than locally because of the processing power. Overall by combining the various strengths this leads to a good execution time.

As follows from Figure 3.2 (*d*) the main difference between the fog and hybrid approach is that they differ in terms of the place for applying the machine learning algorithms on the transformed data. In the fog approach the whole processing (data fusion and machine learning algorithms) happens in the node itself which can be considered as decentralised architecture. Whereas, in the hybrid approach the raw data is firstly fused in the fog node to extract features, and then these features are sent to the cloud as input for the machine learning algorithms. The major benefit of the hybrid approach over the fog one is using the power of the cloud for the processing of the processing intensive machine learning algorithms. Therefore, this step helps in reducing the processing time as a contributor to the overall data processing time.

3.3.7 Approach Based on Preferences

The users can identify preferences like energy consumption, data communication, execution time, accuracy, privacy and total processing time. As we mentioned earlier that there are three approaches namely fog, cloud and hybrid. The approach selection can be varied based on the users' preferences. The following examples of selections are made based on the results that are obtained from the experiments.

(Cloud approach) If the user prefers to reduce energy, execution time and have full accuracy of data, then the raw data will be transferred to the cloud. As the execution time will be less due to the powerful devices located in the cloud, which can reduce as well as the use of energy. However, the energy costs can be more when comparing to the battery-powered fog nodes as running servers can cost more.

(Fog approach) If the user prefers to increase privacy, then data will be processed locally. As the data will be processed locally, the user will have full control of the data and data will not be shared with anyone outside the local network. This results in no data communication costs over the network as well.

(Hybrid approach)If the user prefers to reduce the data communication over the network, total processing time (Execution time and data communication), also maintain privacy and accuracy, the data will be processed in fog and cloud.

Based on the user's kind of preference, the user might actually end up with a different approach. So if the user wants to reduce energy on the local devices, then that will push towards a cloud approach rather than other approaches. Also, if the user is very interested in privacy, and doesn't want the data to go very far, then obviously this pushes towards processing more in the fog, even at the expense of energy and loss of accuracy or whatever other criteria that might not be quite so optimal. Whereas if the user is happy to push the data to the cloud, but the user wants to protect the local devices, then maybe almost moving the user towards more of a cloud approach. The default approach is hybrid, but based on the user preference it is possible to shift into one of the other approaches.

3.4 Summary

We have presented a hybrid approach in which data is fused in the fog before being sent to the cloud to reduce data communication over the network. The results show that this architecture is successful in terms of reducing data communication cost over the network without significantly reducing the accuracy of later decision making. We presented the proposed approach and its relevant methods. In addition, we used the WISDM dataset [55] to val-

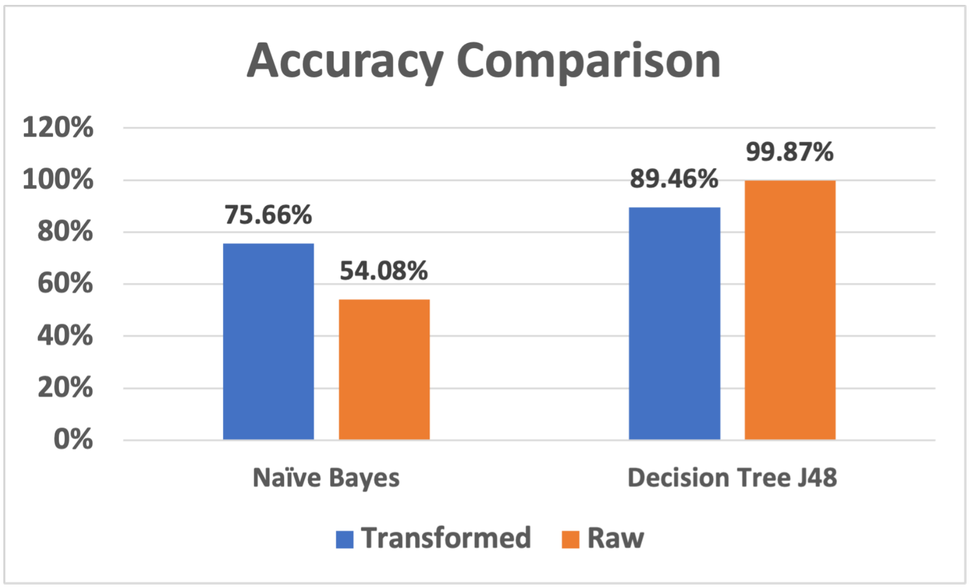


Figure 3.3: Accuracy Comparison

idate our architecture. We did a comparison between the accuracy of raw data and the transformed data as shown in Figure 3.3. In the comparison, we have used Naive Bayes and decision tree algorithms. The results of the comparison show that good accuracy was obtained with the decision tree algorithm in both raw and transformed data, but there was a little loss in the accuracy with transformed data. However, the accuracy of transformed data in Naive Bayes was better than the raw data. It is clear that there is a trade-off between speed and accuracy. Our approach is still performing well with both algorithms in both raw and transformed data. The decision of choosing whether accuracy or speed depends on the context, preferences and field. If the preference is to have a fast response without significantly affecting the accuracy, then our approach using the fog is a good choice. However, if the preference is to have a more accurate response without giving attention to the speed, then the fog can be considered as an extra node in the network. In our context the speed was more important than the accuracy, as our aim was to reduce the data communication over the network without significantly reducing the accuracy. Based on the objective function, we wanted to minimise data communication (min-data), minimise execution time (min-exec), maximise privacy (max-privacy). It is clear from the results that data communication is minimised, execution time is minimised, also the privacy is maximised by doing the processing as much as possible locally.

Chapter 4

Exploring the effectiveness of service decomposition in Fog Computing architecture

4.1 Introduction

In this chapter, we propose to decompose services to create *linked-microservices* (LMS). *Linked-microservices* are services that run on multiple nodes but closely linked to their linked-partners. *Linked-microservices* allow distributing the computation across different computing nodes in the IoT architecture. Using four different types of architectures namely cloud, fog, hybrid and fog+cloud, we explore and demonstrate the effectiveness of service decomposition by applying four experiments to three different types of datasets. Evaluation of the four architectures shows that decomposing services into nodes reduce the data consumption over the network by 10% - 70%. Overall, these results indicate that the importance of decomposing services in the context of fog computing for enhancing the quality of service.

The combination of IoT devices and fog computing enables smart environments that can respond to real-time events by combining services offered by multiple, heterogeneous devices. This can be achieved by decomposing the services into linked-microservices which can distribute the data processing as close as possible to the data source. Microservices are defined as independent, tiny autonomous services which function together to complete a task [66]. It is worth noting that if the microservices are linked to each other, then the distribution of processes among fog nodes will be admissible in an IoT architecture. In other words, moving the computation from centralised approaches to more distributed ones will be possible, leading to a reduction in data communication cost over the network and reduced data frequency between fog devices and the cloud [50, 102]. Most of such microservices can implement typical machine learning (ML) tasks that can deal with the volatility and heterogeneity of data produced in the IoT environments. Data processing using ML techniques in typical IoT-fog applications consists of well-defined steps such as feature extraction, pre-processing and applying relevant algorithms.

Thus, in this chapter, we focus on the research problem of arriving at the best service computation distribution strategy that is cognisant of node constraints and can deliver a reduction in data communication cost for different types of data modalities. For this, we conduct a range of experiments to see how the traditional machine learning algorithms perform in the fog computing domain to highlight the importance of efficiently selecting which services should run on which node.

We have used fog and edge computing interchangeably because of their similarity in moving the computation from centralised clouds to the edge of the network. The proposed linked-microservices decomposition strategy can be extended and applied to a range of edge devices, such as switches and routers if their device and data computation capability information can be obtained, as has been demonstrated in [108].

Our objective is to demonstrate the practical validation of the proposed approach. Our evaluation strategy is similar to the work that has been done in [83].

We have used several machine learning algorithms including Naive Bayes, Logic Regression, K Nearest Neighbours (KNN), Decision Trees, Multi-layer perceptron (MLP) and Support Vector Machine (SVM) to explore and test which algorithm is the best fit for the given data modality. Based on the results, there is no single optimum technique for all types of datasets. Every algorithm has different training time, with some requiring less time or less storage such as Naive Bayes, KNN and Decision trees. However, in terms of executing the persistent features and multi-dimensional datasets, the SVM and MLP algorithms can perform them effectively. The authors of [53] have reviewed many machine learning algorithms and they stated that it is not possible for an algorithm to perform better than others for all given datasets. This chapter has surveyed the well known techniques with the focus being to find the key concepts. Therefore, our selection of machine learning algorithms was based on their efficiency in different datasets and being well known techniques.

4.2 Contributions

The range of fog nodes and data modalities (i.e. numerical, text and image) considered in our experiments are drawn from representative IoT-enabled fog computing applications such as crowd surveillance [61, 62], service provision for massive ad-hoc crowds (e.g. 10 million Hajj pilgrims [79]), optimised computation distribution [102], augmented brain-computer interaction game [42], a visitor-identification system in smart homes [57] etc. The main contributions of this chapter are as follows:

- Identifying the most efficient architecture among the IoT architectures can be done by exploring some strategies and possibly applying them to identify the efficiency of the architecture. Therefore, we explore the importance of decomposing services into linked-microservices in a distributed architecture in fog computing domain for enhancing the QoS and meeting the low latency requirements of IoT-fog applications.

- The efficiency of data analytics architecture in the IoT can be affected when processing machine learning algorithms. As a result, we explore how different machine learning problems can be efficiently dealt with using service decomposition.

- Selecting the most effective data analytics architecture is a challenging task as it requires to apply trade-off analysis to present and evaluate the effectiveness and ineffectiveness of each architecture. Consequently, we propose an efficient approach, which decomposes the services and deploys them as close as possible to the edge of the network. We conduct a trade-off analysis to demonstrate the usefulness of different service decomposition strategies, with the evaluation of the four possible strategies showing that decomposing service computation over the fog nodes reduces the data consumption over the network by 10% (for text data) - 70% (for numerical data). This reduction in the data flow in turn implies less energy and bandwidth costs for the network, while also enabling reduced overheads for securing the condensed data features.

- We use different types of data modalities including numerical, text and images using different ML algorithms, since these are the most common modalities of IoT data sources, as shown from our analysis of a variety of IoT-fog applications above. Thus, we believe that this is generalizable for most of the cases in this problem domain.

In this chapter our objective is to show that splitting the services or decomposing the services into microservices should help in executing the services effectively, we propose as our future work to build a systematic way to divide the workload. For example, we tried different decomposition techniques and the results were different for each decomposition technique depending on the specific use case; for different kinds of datasets the results will be different based on the results that we have achieved.

The remainder of this chapter is organised as follows: First, we present a motivating scenario that is representative of the problem domain, second, we present our methodology through the different datasets and ML algorithms considered for the experiments. Third, we show and evaluate the outcomes of the experiments, finally the conclusion is drawn.

Parts of this chapter have been published in [17].

4.2.1 The process of splitting services

We have two decomposition one is decomposing is based on the data, so data decomposition by splitting the dataset. The second way is decomposing by functional decomposition, which can lead to smaller services. The Assumptions and the process that are considered in 3.3.4 are used in this chapter, but in this chapter, we have added a process of splitting services as follows The process of splitting services:

- Splitting services into smaller services are undertaken experimentally. This is a partitioning of data and partitioning of functions.
- To explain the process, assume there is an activity recognition service, and this service is compact has smaller services (functions) such as finding the average, standard deviation and so on. In our process, we spilt this compact service into separated smaller services.
- The services are converted to executable format, which helps to make the process of distribution and running the services in different nodes easy.

4.3 Problem analysis

IoT devices have constraints on resources like RAM, CPU and storage. The services that execute on these devices have restrictions expressed in terms of the same resources. Additionally, service execution and distribution need to take into account the data computation capabilities (i.e. in terms of the installed library support) of the devices. Therefore, we need to model the data about services to get knowledge about their restrictions before distributing them across the nodes. For example, the image recognition service needs at least 500MB to be executed, so the IoT device's capability should be powerful enough to execute this service. Therefore, if the device has a limitation in processing power, then it is not possible to execute the service on it. In this case, service decomposition is an important aspect of the IoT architecture due to the involvement of devices with limited hardware capabilities and varying data computation availability, which cannot handle resource intensive tasks.

In addition, decomposing services into linked-microservices is an important aspect in terms of service composition in the IoT architecture. This is crucial for the effective distribution of services to the nodes in the IoT. The main challenge is to determine which services should be executed on which node in a given IoT architecture, by considering both overall efficiency and feasibility. This is similar to the Job shop problem which is one of the most known problems in combinatorial problems [41]. Basically, the idea of the 'Job shop' problem, is that there is a group of machines with varying levels of computational power (i.e., given a specific job, j, it could be the case that there are two machines that complete j in different amounts of time, with the quicker one having higher computational power).

The problem then asks for an algorithm which produces an optimal assignment of jobs to machines, such that the overall amount of time it takes for all jobs to be completed is minimal. The following scenario illustrates the problem by using a real use case. The scenario is drawn from a recent UAV (Unmanned Aerial Vehicle) crowd surveillance study [61], which looked at energy efficiency achieved by offloading the facial recognition operation to a Mobile Edge-Computing node rather than processing it locally on the UAV (using a Raspberry-Pi for computation). We extend this in our case to include multiple fog nodes (devices) with varying hardware and computation abilities. The scenario demonstrates the significance of deploying the right service on the right node.

4.4 Motivating scenario: An event in a city

We will present a motivating scenario about an event in a city. There is a major event in a city where people are taking videos and photos. The law enforcement agencies are interested in re-utilising the captured images to identify criminals (or persons of interest) among the crowds in order to anticipate crimes. In Figure 4.1, there are four types of Nodes (N_i) including mobile phones (N_1), drones (N_2), streetlights (N_3) and cloud (N_4). Every node has a different combination of resources (CPU, RAM, energy, storage and network bandwidth) and each node can execute several different services (S_i). Each service requires a specific combination of resources to be executed on a given node. Additionally, the facial recognition service comprises a variety of linked-microservices corresponding to ML tasks, including facial feature extraction, data fusion, data filtering and face detection algorithms.

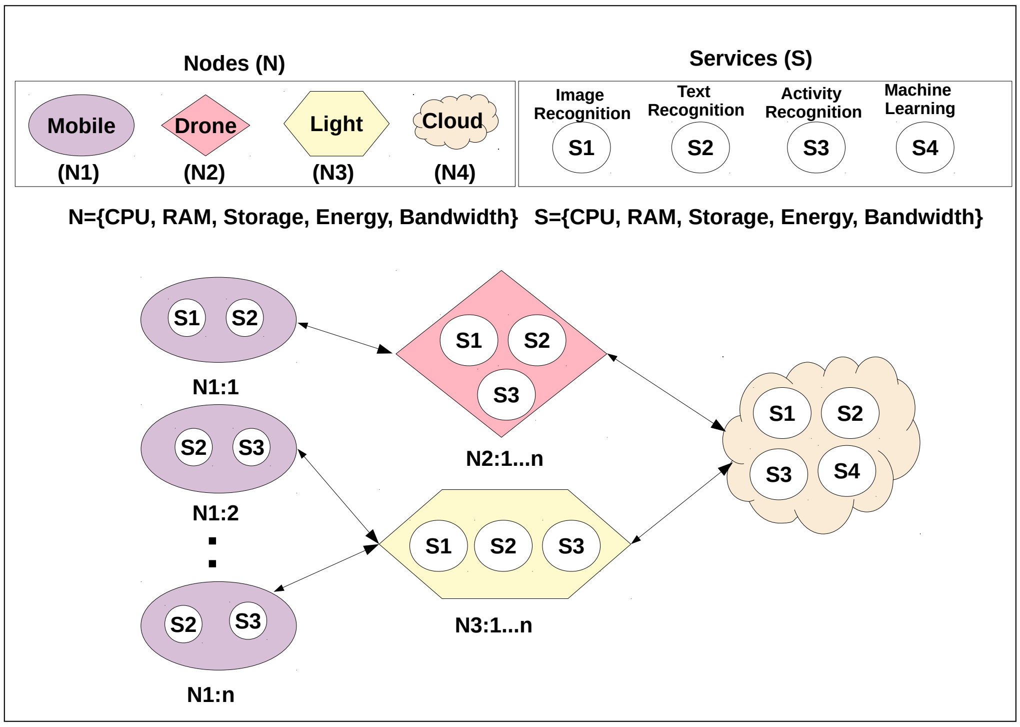


Figure 4.1: Scenario: An event in a city

The event goers are taking photos and videos of the event primarily for their pleasure, not for helping the law enforcement agencies. However, they may like to help the law enforcement agencies to maintain safety as long as the primary function of their devices in this scenario is not compromised. The system should ask the users for their consent to use their data and devices to avoid ethical issues, so permission is required from the users before using their devices.

The sustainability comes from better energy consumption, less communication means longer duration and more devices can be connected together. The overall architecture should not consume too much energy or communication bandwidth from users' devices. The simplest case of service distribution will send all the raw data to the cloud from individual user devices even though it consumes a lot of network bandwidth. However, if we can do the data transformation in a smartphone, then it consumes less bandwidth and sends only processed data to the cloud. The services associated with face recognition ML tasks are deployed dynamically in smartphones. It is crucial to consider which responsibilities should be assigned to smartphones.

In addition, the nearest lamp or drone will have more computing power than the phones and can handle the more computation intensive tasks than user devices. This reduces the communication cost in two ways: First, the images will be transformed into feature vectors in the mobile, and secondly, further processing will be applied to data when they are sent to the lamps or drones which are temporarily deployed because of the event and connected with the mobiles via Bluetooth which is cheap in terms of communication. Then the transformed data will be sent to the cloud via 3G for further analysis.

This scenario introduces a number of challenges since it involves deploying dynamically composed services during an event in the city. The event happens on a particular day and people are likely to move around while taking photos which introduces unpredictability in their location. The service orchestrator needs to be aware of the resources available on the users' phones when sending a request for service computation to them. It is crucial to consider how to compose services like sending data to the drones as mediator. Additionally, what services should be deployed to the drone and mobile is also an important aspect. This scenario illustrates the problems involved in service provisioning on fog nodes (taking into account varying hardware and data processing capabilities) and deploying the service taking into account data communication costs.

There can be different criteria or preferences to have the optimal architecture for certain scenario or application. The chosen criteria or preference might influence the choice of architecture as this multi criteria optimisation problem. As we mentioned earlier in 3.3.7 regarding the preferences of users can influence selecting the approach to be followed. It is similar here, but there are more criteria like we may have constraints because of legal requirements, or ethical considerations that has certain aspects that we cannot collect and process certain data which forces us to use different architecture. These are constraints that restrict the approaches that we select. Based on criteria being chosen we might end up with a different architecture. Involving this multi criteria problem is important as it is not possible to optimise everything in one approach, so there are some architectures can be optimised for accuracy and others can be optimised for performance.

4.5 Methodology

As we discussed in Chapter 2, there are clear trade-offs among the three architectures (centralised, decentralised and distributed). It is possible to find solutions that can maximise the advantages and minimise the disadvantages of each architecture. Our proposed method endeavours to fulfil this, which is presented below. We propose an efficient approach which aims to move the computation from the cloud to the fog as much as possible.

The goal is to process a service S effectively. We begin with decomposing services into a set of linked-microservices MS to distribute them among nodes in our architecture. A good illustration of this is shown in Figure 4.2, there are three services namely activity s_1, image s_2 and text s_3 recognition. These services will be decomposed into linked-microservices (MS) before the distribution process.

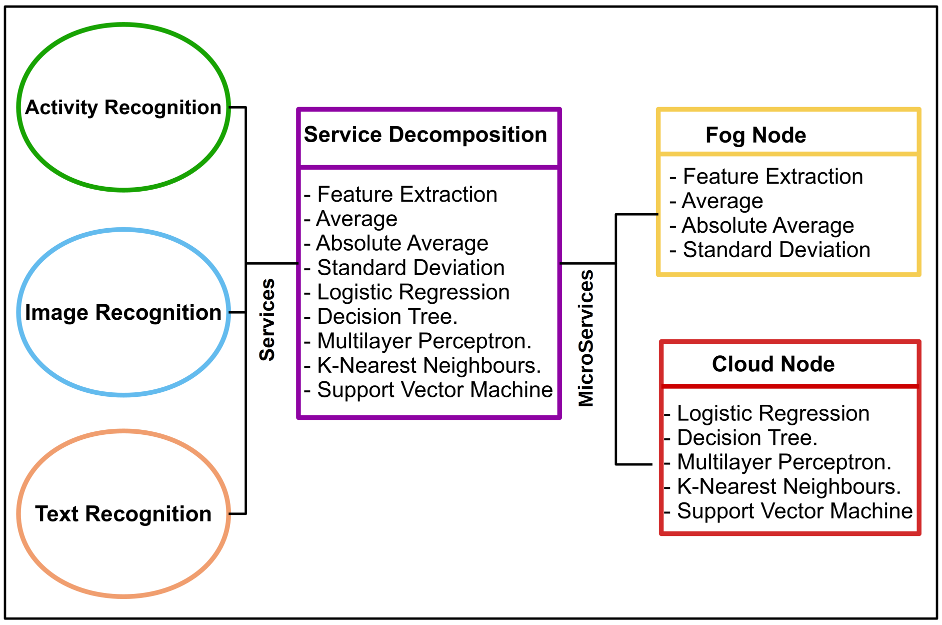


Figure 4.2: Service decomposition

Then, the distribution process will distribute the micro services depending on the constraints on services and nodes of the nodes. Then we apply data processing techniques to sensors' data to extract features and reduce the number of data points in the fog node (FN). This phase is significant because it is not possible to analyse raw time-series data by algorithms of classification effectively. Then, the cloud will receive the extracted features for creating inferences and training purposes. Therefore, to get the results we compose both the fog node and the cloud node to process the service.

We conduct experiments for each of the above architectures in order to explore how the hybrid data analytics architecture would be beneficial in a variety of ways.

We use three types of datasets namely numerical, text and image. The details of each dataset including dataset description, algorithms and process will be discussed below. Each dataset has its description and algorithms. However, they have a common process in terms of decomposing the services and distributing the computation over the nodes. It is worth discussing the similarities before discussing the details of each experiment.

Objective Function

$minimize/maximize\ f_m(x)$

$m = \{$ min−data, min−exec, max−privacy $\}$

f is the objective function, m is the set of objectives and x is a vector of variables.

4.6 Process of three types of experiments

We conducted 4 sets of experiments under each of the three types of experiments which are Numerical, Text and Image data. The process of the experiments is shown in Figure 4.3. The explanation of the four experiments related to all types of datasets are below:

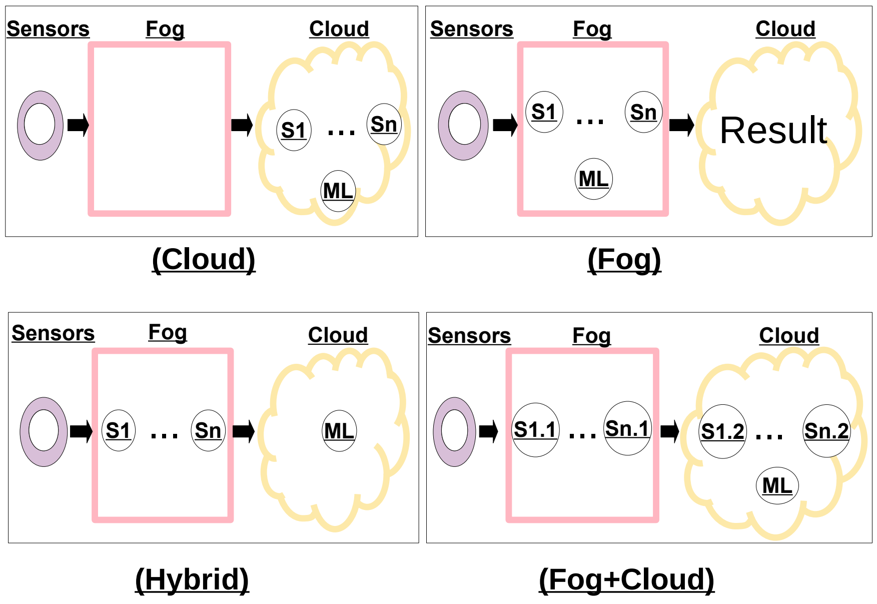
First experiment. We sent all raw data to the cloud and data transformation methods are applied to the raw data to extract features. Then, we applied Machine learning to the altered data based on the extracted features in the cloud as shown in Figure 4.3 (Cloud). We calculate the accuracy of each algorithm, the amount of data that is sent to the cloud and the execution time.

Second experiment. We applied data transformation methods to the raw data to create features. Then, we applied analytical algorithms to the modified data based on the extracted features in the fog as shown in Figure 4.3 (Fog). We check how feasible is the resource constraint device when processing the data and we measure the time of the execution.

Third experiment. We applied the feature extraction methods to the raw data to extract features in the fog. By applying this the data is minimised as much as possible in the fog, then the transformed data will be sent to the cloud for further analysis as shown in Figure 4.3 (Hybrid). We calculate the accuracy of each algorithm, the amount of data that is sent to the cloud and the time of execution.

Fourth experiment. This is similar to the third experiment in terms of applying data aggregation algorithms to the raw data in order to extract features in the fog. However, in this approach, we applied the feature extraction on part of the raw data in fog and the remaining in the cloud. We divided the dataset into two parts randomly, where 70% of the data will be used in fog and the remaining 30% will be sent to the cloud. In this experiment, the statistical measurements in the Fog are presented as S1.1, S2.1, S3.1, S4.1, S5.1 and S6.1, which are applied on 70% of the raw data.

However, in the Cloud, the statistical measurements are presented as S1.2, S2.2, S3.2, S4.2, S5.2 and S6.2, which are applied on 30% of the raw data. In numerical data, in the first part, we applied fusion methods on the 70% of the raw data which is 768746 rows that is equal to 35 MB (70% of the file size) in the fog. Then, we sent the transformed data which is equal to 0.84 MB and remaining raw data (30% of the data) which is equal to 15 MB to the cloud for data transformation and then analysis as shown in Figure 4.3 (Fog+Cloud). In text data, the first part has 14 newsgroups



S = Statistical Measurements as Services, ML = Machine Learning Algorithms as Services
(Cloud) = all the services are processed in the cloud, (Fog) = all the services are processed in the fog, (Hybrid) = the statistical measurement and machine learning services are distributed into fog and cloud respectively, (Fog+Cloud) = 70% of statistical measurements services processed in the fog and the rest with machine learning services are processed in the cloud.

Figure 4.3: Process of three experiments.

which are 70% of the data and it will be processed in the fog. The second part has 6 newsgroups which are 30% of the data and it will be transferred to the cloud. Then, we sent the transformed data and remaining raw data to the cloud for further data transformation and analysis as shown in Figure 4.3 (Fog+Cloud).

In image data, the first part is training data which has nearly 70% of all images and equals to 17185 images (392 MB file size) and it will be processed in the fog. The second part is testing data which have nearly 30% of all images and equals to 7815 images (178 MB file size) and it will be sent to the cloud. Then, we sent the transformed data and remaining raw data to the cloud for further analysis as shown in Figure 4.3 (Fog+Cloud). In this experiment, we calculate the accuracy of each algorithm, the amount of data that is sent to the cloud and the time of execution.

4.7 Experiment 1: Numerical data with 6 measurements

4.7.1 Dataset description

The dataset that is used is called WISDM [55] that has been discussed earlier in Chapter 2.

4.7.2 Algorithms

We have used six statistical measurements that are used in [55]. There are 43 features that are created including the mean (S1), standard deviation (S2), the average absolute difference (S3), the time between peaks (S4) of every axis, average resultant acceleration of all axis (S5) and binned distribution for each axis (10 equal sized bins and total 30 bins) (S6). After preprocessing the data, five methods of classification (ML) are applied including Naive Bayesian (NB), Logistic Regression (LR), K-Nearest Neighbours (KNN), Decision Tree (DT) and Multilayer Perceptron (MP).

4.8 Experiment 2: Text data

4.8.1 Dataset description

The dataset that is used is called Twenty Newsgroups [56] which has been discussed earlier in Chapter 2.

4.8.2 Algorithms

To apply analytical algorithms to text data, it is important to convert the text into a numerical feature vector. To extract features, first, we will use Tokenizing text with scikit-learn (S1) which has text pre-processing and filtering. Second, from occurrences to frequencies (S2) which counts the occurrences and divides the available occurrences of every word by the whole words of the file. These features have a specific name which is term frequencies (tf). Also, downscaling (tf-idf: Term Frequency time inverse document frequency) the weights for words that appear in most of the files have less knowledgeable information than the words that appear in very small parts of the file. After extracting features from data, it is possible to apply the classifier (ML) to give a prediction of the category in the post. The first classifier is naive Bayes classifier in scikit-learn which has a variety of the classifiers and the most suitable is a multinomial variant for this dataset. The second classifier is a support vector machine (SVM) which can be considered one of the most used algorithms in text classification.

4.9 Experiment 3: Image data

4.9.1 Dataset description

The dataset that is used is called Dogs vs Ċats dataset[1] which has been discussed earlier in Chapter 2.

4.9.2 Algorithms

To apply machine learning algorithms, we need to convert the images into a feature vector. We are going to use two methods that take input and produce feature vectors as output. First, the $image\_to\_feature\_vector$ (S1) function that takes the image as input and changes the size of the image to stable height and width and the intensity level of RGB is converted into a single set of numerical data. Second, $extract\_color\_histogram$ (S2) function gets an image as input and produces the histogram of colour to describe the image colour classification. Then, we use the k-nearest neighbours algorithm (k-NN) (ML) classifier to give a prediction of the category as either dog or cat.

While doing all the three experiments, we expected that we can achieve a similar accuracy with the hybrid approach while maintaining the processing time and with considerably minimising data communication over the network.

[1]https://www.kaggle.com/c/dogs-vs-cats

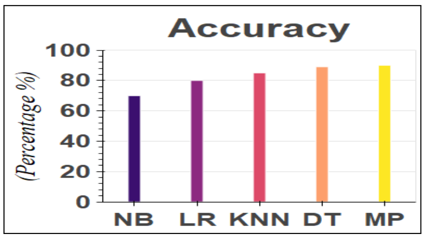
4.10 Experimental set up and results

As discussed earlier, creating features from raw data such as numerical data, text data and image data, helps in the process of applying analytical algorithms on them. In our experiments, we have used a Raspberry Pi 3 model as the resource-constrained device. The device specification is 1GB RAM, with Raspbian Jessie as the operating system. These experiments can be performed on smartphones as well, but in our experiments we preferred Raspberry pi as it has similar specifications to smartphones, and is much cheaper than smartphones, with the cost also being a factor in IoT environments. Some papers have obtained and processed data on smartphones, as in [29, 86, 102]. Furthermore, a Linux based System which has 16GB RAM is used to mimic the device of the cloud. For data aggregation, segmentation and feature extraction, we have used Java and python libraries.

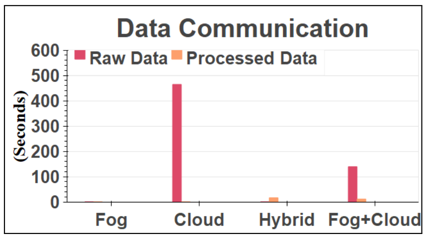
We used python 2.7.12 and 3.5.2 and the Weka 3.8 tool for machine learning (classification methods). For the numerical experiments, the Weka tool is used and the heap size is adjusted in both cloud and fog. In the cloud environment, the size of the heap was set to 8GB, whereas in the fog, the size was 650MB RAM for the numerical data experiment. However, the remaining experiments were done in python. While conducting these experiments, the internet upload speed was 1 Mbps. We used several packages and libraries in python for image analysis including NumPy, Argparse, OpenCV packages, Scikit-learn library and imutils library. In addition, Scikit-learn library and NumPy packages are used for text analysis.

4.10.1 Experiment 1: Numerical - 6 measurements

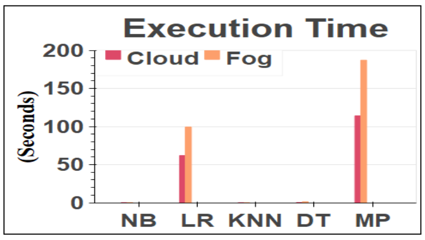
It is clear from results that the multilayer perception has the highest accuracy percentage. The results show us that two algorithms (logistic regression and multilayer perceptron) have significant differences between the two sides.



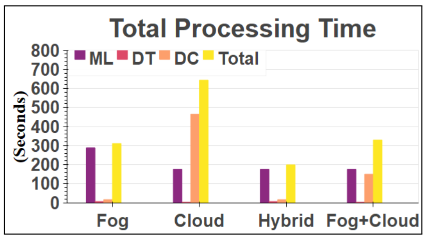
(a)



(b)



(c)



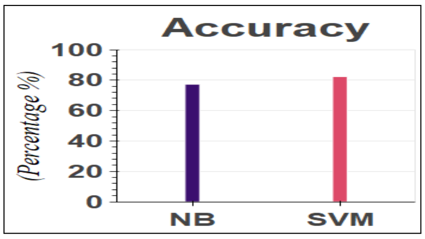
(d)

NB = Naive Bayes, LR = Logistic Regression, KNN = K Nearest Neighbours, DT = Decision Tree J48, MP = Multilayer Perceptron

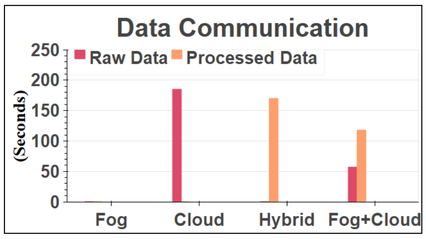
ML = Machine Learning, DT = Data Transformation, DC = Data Communication

Summary: (*a*) Shows the accuracy results of the analysis algorithms that we applied to the transformed data. (*b*) Shows the data communication time over the network from fog to the cloud. (*c*) Illustrates the execution time of the analytics algorithms in both the cloud and fog device. (*d*) Demonstrates the total processing time for the four architectures.

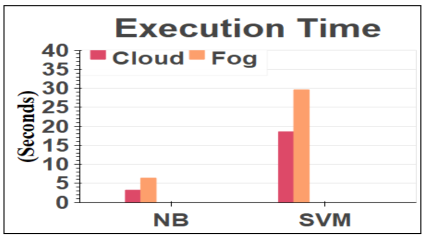
Figure 4.4: Experiment 1: Numerical - 6 measurements



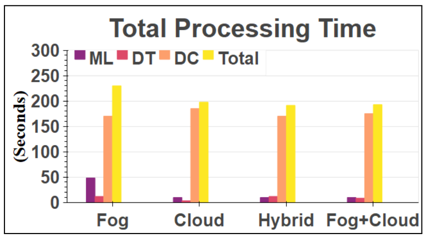
(a)



(b)



(c)



(d)

SVM = Support Vector Machine, NB = Naive Bayes

ML = Machine Learning, DT = Data Transformation, DC = Data Communication

Figure 4.5: Experiment 2: Text data

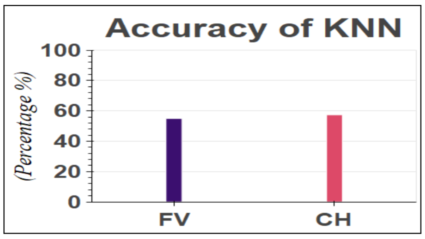
4.10.2 Experiment 2: Text data

It is clear from the results that the support vector machine has a higher accuracy percentage than Naive Bayes. There are two bars visible in Figure 4.5. *Experiment* 2: Text Data (*b*): one for the raw data and the other for the transformed data. The results show us that the two algorithms (Support Vector Machine and Naive Bayes) have significant differences between the two sides.

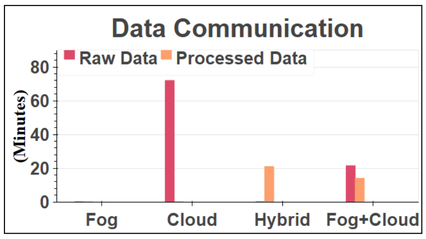
4.10.3 Experiment 3: Image data

It is clear from the results that applying the K-NN classifier on extract_-color_histogram has a higher accuracy percentage than Image_to_feature_vector. There are two bars visible in Figure 4.6. *Experiment* 3:

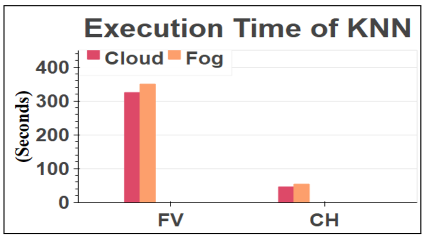
Image Data. (b): One for raw data and the other for transformed data. It is clear from the results that applying K-NN classifier Image_to_feature_vector significantly takes more time to execute than extract_color_histogram. However, comparing between cloud and fog there is no big difference in execution as in data communication.



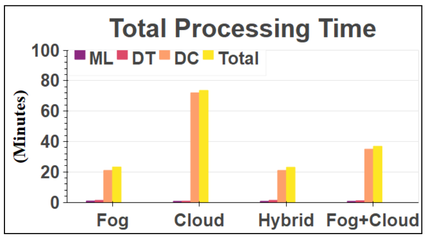
(a)



(b)



(c)



(d)

KNN = K Nearest Neighbours, FV = Image to Feature Vector, CH = Extract Color Vector

ML = Machine Learning, DT = Data Transformation, DC = Data Communication

Figure 4.6: Experiment 3: Image data

4.10.4 The results of the three experiments

It is clear that the fog only device has no data communication cost because the processing happened in the device and no communication to the cloud took place. However, in the cloud approach, the raw data communication over the network from fog to the cloud is higher than other approaches. On the other hand, in the hybrid approach, the transformed data communication over the network from fog to the cloud is lower than both cloud and fog+cloud approaches. This is not a surprising result confirms that we can save significantly on data communications by pre-processing data early in the chain. Obviously, the fog takes more time than the cloud to execute classification algorithms because of its resource constraints. In total processing time graph, there are three measurements for each architecture including the execution time of analytics (ML) algorithms, the execution time of the data transformation process and the data communication time between the local device and the cloud as shown in Figure 4.4, 4.5 and 4.6. The total processing time graph in Figure 4.4, 4.5 and 4.6 needs a bit more explanation as the results are more interesting, so the details are as follows:

Fog. The data transformation process is conducted locally and it is clear that the processing time is higher than in the cloud. The analytics algorithms have been processed locally and they took much more time than cloud because of the processing power. However, data communication (the time to send data to the cloud) is very low as only aggregated data is being sent for storage. So, the overall processing time of Fog approach in Experiments 4.10.1 and 4.10.3 is in the middle of the measured approaches. However, the overall processing time in Experiment 4.10.2 shows that fog is the slowest approach in the given setting.

Cloud. Data Communication is the time that the raw data takes from the IoT device to the cloud, which is higher as all raw data is being transmitted. The data transformation process was done in the cloud and due to the available resource runs quickly. Also, the analytics algorithms have been processed in the cloud and they took much less time than the fog because of the processing power. However, in Experiments 4.10.1 and 4.10.3 the overall due to the significant amount of transmission time the cloud is the slowest approach in the given setting. However, in Experiment 4.10.2 the overall cloud is in the middle of the measured approaches.

Hybrid. Here the data transformation process is done locally (in the fog) on the Raspberry Pi, with the usual observation. Data Communication is the time the transformed data takes from the IoT device to the cloud which as before is low. Finally, the analytics algorithms have been processed in the cloud on the transformed data; they took much less time than locally because of the processing power. Overall by combining the various strengths, this leads to a good execution time.

Fog + Cloud. This is similar to the hybrid approach in terms of the data transformation process is done locally (in the fog). However, as mentioned earlier, the data are divided into two parts: 70% of the data which is transformed locally and the other 30% in the cloud. Data Communication is the time the transformed data and remaining raw data take from the IoT device to cloud which is lower than cloud and higher than both fog and hybrid approaches. Finally, the analytics algorithms have been processed in the cloud after data transformation is applied to the remaining data to have all data transformed. The machine learning algorithms took much less time than locally because of the processing power. So, overall processing time is in the middle of the measured approaches.

4.10.5 Lessons learnt and take away messages

The accuracy and execution time is the same as in Figure 3.2, but data communication and total processing time is differed from Figure 3.2 because we have added a new approach called fog+cloud. So we have used the same dataset and the same approaches in 3.2 and we added the fog+cloud approach where we split the dataset to see how effective will be the data decomposition in terms of total processing time when compared with other approaches.

Based on the result, in numerical and image data there was a notable difference in total processing time among the four architectures, however, in text data there was no significant difference. One of the reasons can be that in the tokenization process the text will be replaced with something else that will not affect the size of the data significantly. Another reason for this might be the data transformation algorithms that are used. However, the idea is not just to reduce the size of data, but also to create meaningful data to get more insights. We have used three types of datasets including numerical, text and image. These datasets are publicly available, but they are not taken from an industrial application due to unavailability of public IoT industrial application datasets to conduct our experiments. However, the types of data in the used datasets are quite similar to industrial data in terms of numerical, text and image data. The survey in [71] has reviewed over 100 Internet of Things solutions and divided them in the industry marketplace into five classes including smart home, smart wearable, smart environment, smart city and smart enterprise. The datasets that we have used in our experiments fall into some of these classes in terms of data types (numeric, text and image).

For completeness, while working with the image dataset, one of the difficulties faced was installing imutils[2] library and OpenCV[3] library for python. Also, we used Linux based system for our experiments because we faced issues with installing libraries/packages and adapting the environments for the experiments while using other operating systems. Additionally, the accuracy of the image dataset might be a bit low, but it is possible to increase the accuracy by using different methods than KNN such as Convolutional Neural Networks (CNN). Similarly, in both datasets including numerical and text, higher accuracy can be obtained either by tuning the machine learning algorithms or by using different algorithms such as CNN. In addition, the data transformation methods can be utilised or different methods can be used to create features with more insights. These issues can be important aspects of data analytics by finding answers regarding how to increase the accuracy, but it is important to use lightweight solutions to avoid having high execution time. However, in this work, we focused on exploring the most effective fog computing architectures for the IoT.

4.11 Summary

This chapter presents an efficient approach where the raw data is preprocessed in the fog node before being transmitted to the cloud to minimise the data communication over the network. Three types of datasets are used for the experiments including numerical data, text data and image data. Furthermore, we conducted 4 experiments, including 4 architectures namely cloud, fog, hybrid and fog + cloud for each dataset to explore which one

[2]https://pypi.python.org/pypi/imutils
[3]https://opencv.org/

is the most effective for the Internet of Things. The results show that the hybrid approach is efficient in terms of minimising the cost of data communication over the network while maintaining accuracy. However, fog + cloud approach could be useful to employ in situations where fog device has limited processing capability and cannot perform all the required processing. In such situations, fog + cloud approach would perform better than the cloud only approach. In addition, we used the WISDM dataset [55], 20 Newsgroups dataset [56] and Kaggle dogs vs cats dataset [49] to validate our architecture. The feature extraction service demonstrates the value of partitioning process because the feature extraction obviously looks at the whole dataset that's coming in from a user, and then identify some sort of key elements that you do want to pass to your data analytics. So in that step, we can actually reduce the data volume significantly. So that is something we do want to distribute, we want to move it as close as possible to the user. Whereas maybe some elements of the actual data analysis we do not want to move as close as to the user because we need the much wider insight of data from all your users. Also, anything that sort of very time critical, we might want to do quite close to the user as well where feasible because then we are cutting out on that chain of going to the cloud and coming back. Based on the objective function, we wanted to minimise data communication (min-data), minimise execution time (min-exec), maximise privacy (max-privacy). It is clear from the results that data communication is minimised, execution time is minimised, also the privacy is maximised by doing the processing as much as possible locally.

Chapter 5

Service distribution strategy for the IoT

5.1 Introduction

The IoT and cloud computing are the most effective technologies that impact the lives of most people in diverse manners. It is not possible to ignore their good impact on our lives, but the shortcomings of both technologies exist. They are inexpensive and convenient, but the services of the cloud make extremely high data transfer rates in the network which makes the process of data transfer much expensive because the network requires to upgrade the bandwidth. Besides, the physical distance between the devices and the servers make a frequent delay issue in traditional cloud architecture. Fog computing can be proposed to tackle these challenges as a model that distribute the data processing in the network. Also, fog computing can be seen as the middle layer between the IoT devices and the cloud. The fog computing is located at the edge of the network close to the IoT devices. This makes the management of devices easy with a small number of devices, but the number of connected devices has been increased which will increase the difficulty of managing the devices.

Moreover, the IoT devices have limited capabilities and constraints that also a big challenge when distributing the services among the devices as the services have constraints. The problems that can arise are an inefficient use of resources, slow data processing, and data communication issues. Therefore, it is important to optimise the usage of resources and the usage of the network to avoid the shortcomings. In this chapter, we propose a distribution strategy called "MEIN" Most Efficient IoT Node, it uses a list of all the IoT devices present in the network. This strategy allows defining the best order to use devices to execute the different services. We have used the best-fit algorithm in the bin packing problem. Bin Packing is "*NP-hard [40] and heuristics have been developed to approximate the minimum number of bins*" [51]. In addition, the author stated that the following algorithms best-fit, next-fit and first-fit are the most classic ones. Furthermore, the author has described the best-fit algorithm as "*best-fit maintains a list of current bins ordered by sizes and upon arrival of item x, puts it in the current fullest bin in which it fits, opening a new bin for x if this fails*".

The criterion to sort the list of devices is physical capabilities of devices like battery, memory and storage. Thus, all devices capabilities are stored in a list for each physical aspect.

The main contributions of this chapter are as follows:

- In IoT environments, the nodes have different processing power capabilities as IoT nodes are constrained, but cloud nodes are powerful. Moreover, the services require different processing power to run on a node. Therefore, we need to identify the capabilities of the IoT nodes and cloud nodes, also, identifying the technical requirements of the services is important to distribute the services to the nodes efficiently.

- Service distribution strategy is important when distributing services to the nodes in the IoT is a challenging task as there are different types of devices and different capabilities of nodes. Therefore, we propose a distribution strategy called Most Efficient IoT Node (MEIN), it uses a list of all the IoT devices present in the network. This strategy allows defining the best order to use devices to execute the different services. We use the best-fit algorithm in the bin packing problem as a baseline. The criterion to sort the list of devices is physical capabilities of devices like CPU, memory and storage. Thus, all devices capabilities are stored in a list for each physical aspect. We conduct a trade-off analysis to demonstrate the efficiency of the service distribution strategy.

- In IoT environments, the number of services and devices will be large, so there is a need to distribute the services and optimise resource usage efficiently. As a result, we explore the importance of distributing the services into the nodes and optimising the resource usage in a fog computing architecture for quality of service improvements, meeting the optimised resources usage requirements of IoT-fog applications.

- We evaluate the MEIN strategy by using the randomly generated dataset, also, we create 15 combinations of capabilities for fog nodes to explore how efficient is the proposed distribution strategy and explore the most effective combination of capabilities for fog nodes to handle most of the services.

In this chapter, our aim is to demonstrate that distributing the IoT services to the right nodes should help in distributing and executing the IoT services efficiently and in an optimised way.

The remainder of this chapter is organised as follows: First, presents a problem analysis and motivating scenario that is representative of the problem domain, second, presents our methodology through the proposed distribution strategy. In addition, experimental setup and results are presented, finally, the summary of the chapter is drawn.

5.2 Problem analysis

As a result of the increased number of connected IoT devices to the internet, the number of services has increased and companies started deploying more services for various purposes. These IoT devices have constraints on resources like RAM, CPU and storage and shortage in battery power. Moreover, every service that is deployed has a similar restriction expressed in terms of the same resources. Additionally, in the execution and distribution of service, it is important to take into account the data processing capabilities of the IoT devices. Hence, we need to model the data of services to get the information and the knowledge regarding their limitations before distributing them to the IoT nodes.

For example, the image and video analysis service requires more computational power than text and number analysis and some of the video analysis services need more computational power because of their size and the used algorithms which mean that the capability of the IoT devices should be powerful enough to execute these services. Thus, the IoT devices have limitations in computational power which are not possible to execute these services on them. In this regard, optimisation in service distribution is a significant aspect of the IoT architecture because of resource-constrained devices in terms of the hardware that lead them to handle tasks that are not intensive. In the IoT environment, many services are distributed on IoT nodes in the network for different purposes. Also, some of the nodes are not used because of their low power capabilities to process a service which means some nodes are there for data processing, but they are ignored because of their power limitation. This can be worse when we consider billions of services that are going to be distributed to billions of nodes and processed by them.

This means there are wasting of resource usage in the network which might delay the processing time of the services. Resource wastage means that the resources are either not used or partially used due to their constraints, and they remain in the network without usage that means wasting the available resources. The main challenge is to determine which services should be distributed on which node in a given IoT architecture, by considering both overall efficiency and feasibility. This is similar to the bin packing problem which is one of the most known problems in bin distribution. Basically, the idea of bin packing problem is Minimising the number of Bins. The problem includes a number of items with various weights between 0 and 1 and bins with a fixed capacity. The idea is to allocate items to a bin but the number of used bins should be minimised and it can be that items have fewer weights than the capacity of the bin. Then, the problem asks for an algorithm which produces an optimal allocation of items to bins in the optimised way and minimising the number of used bins.

We suggest that in the longer-term there is a need to optimise the distribution of the services to maintain the use of services and the traffic on the network, also to process the services in an acceptable time.

5.3 Motivating scenario

Badr is a tourist in the centre of London with his wife and two children. He would like to visit most of the famous places in London including London Eye, Big Ben and River Thames. However, before starting the tour he is willing to go to a restaurant to buy food, and then continue the tour. Therefore, he needs a City Map, weather forecast as the weather in London is usually rainy and Audio Translation as he is not a native English speaker, also some useful information regarding London City like the best restaurant, cafe and attractions. London city offers an official website and mobile application for information searching and useful services for tourists. In Figure 5.1, there is an IoT environment in the city including various IoT devices and embedded devices.

Additionally, there are five IoT Nodes (N_i) including smart shop (N_1), smart mail post box (N_2), smart light (N_3), smart telephone cabinet (N_4) and drone (N_5). Every Node has a different configuration (RAM, Storage) and the nodes can compute a number of services (S_i). Similarly, the services also have a different configuration of requirements which are similar to the nodes' configuration. There are six types of configuration for nodes as explained in Table 5.1. Similarly, there are six types of configuration for services as explained in Table 5.2. Every configuration type has a different combination of RAM and storage. For example, configuration Type C (1) (1024, 8) in nodes means it has 1024 MB RAM and 8GB Storage. There are two types of devices in terms of services' requirements including pre-packaged devices and configurable devices. Pre-packaged devices (Fog computing) have packages of services such as City Map, Navigation, weather forecast, audio translation and city Information.

Table 5.1: The capabilities of nodes

Name	Values	Units
RAM	1024, 2048, 4096, 8192, 16, 32	MB
Storage	8, 16, 32, 64, 128, 256	GB
Configuration Type of Nodes Capabilities (C)	A combination of: (RAM, Storage) Respectively.	C (1) (1024, 8)
		C (2) (2048, 16)
		C (3) (40.96, 32)
		C (4) (8192, 64)
		C (5) (16, 128)
		C (6) (32, 256)

Table 5.2: The technical requirements of services

Name	Values	Units
RAM	256, 512, 1024, 2048, 4096, 8192	MB
Storage	2, 4, 8, 16, 32, 64	GB
Configuration Type of Service Requirements (R)	A combination of: (RAM, Storage) Respectively.	R (1) (256, 2)
		R (2) (512, 4)
		R (3) (1024, 8)
		R (4) (2048, 16)
		R (5) (4096, 32)
		R (6) (8192, 64)

Whereas, configurable devices (smart light, smart shop, smart mail post box and smart telephone cabinet) means whenever the user needs a service, the user asks the system to install it. Fog computing devices get instruction and download the services to distribute them to available nodes, then the devices process the services to send the result to the user. The system selects the type of device based on device capabilities (RAM and storage). Badr's mobile in the morning was a full battery, empty storage and internet connection was good. Badr wants to go to the London Eye from Oxford street. He needs a map, audio translation and weather status.

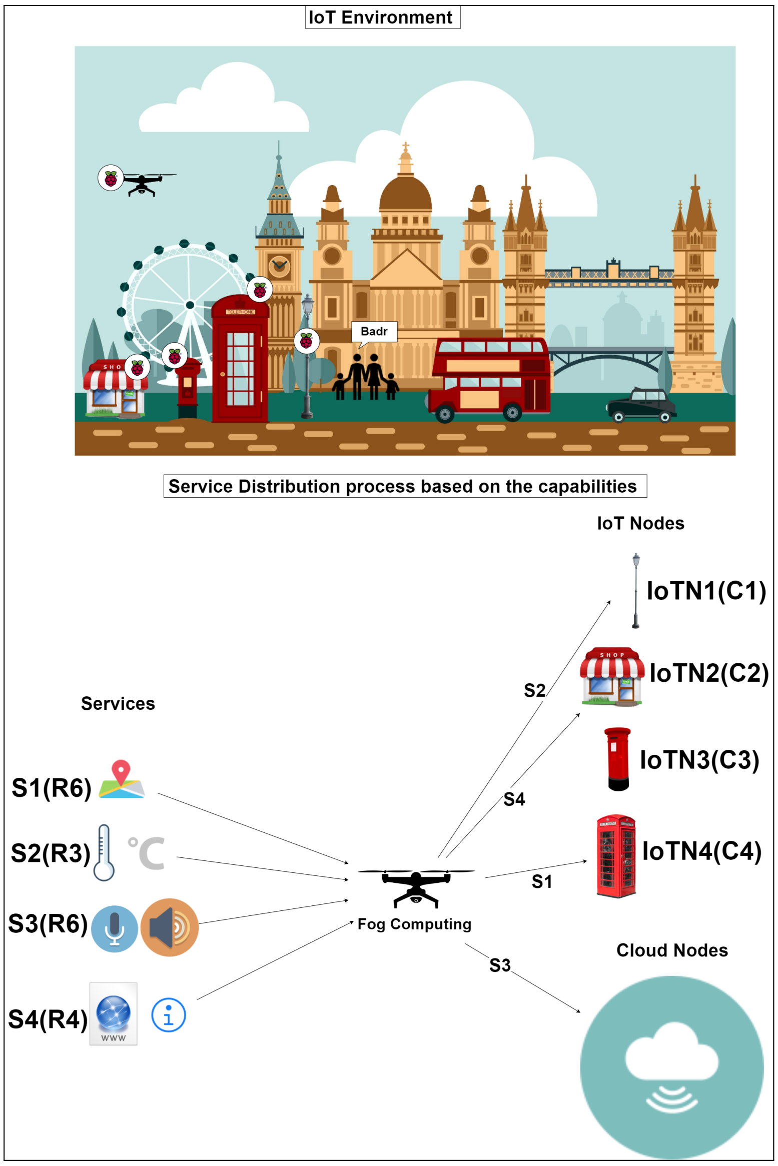


Figure 5.1: Scenario: A tourist in London city

Then, the devices that are distributed will give the required service to Badr when he sends a request. However, whenever the IoT nodes cannot process the services, then the services will be distributed to the cloud for processing. In the evening, Badr's mobile has less battery, less storage and less internet connection, and he needs to go back to the hotel. The system will get the instructions and distribute the required services to the available nodes for processing, then the results will be sent to the user. The overall architecture should consume less communication bandwidth or energy while distributing services othen nodes. The easiest case of service distribution is distributing all the services to the cloud nodes from individual user devices even though it consumes a lot of network bandwidth. However, if we can distribute services to IoT nodes, then it consumes less bandwidth as there will be no communication over the network with the cloud. The users' devices are connected with the drone via Bluetooth which is cheap in terms of communication.

This scenario introduces a number of challenges since it involves distributing services on nodes in a busy large city. London City is a very crowded city full of tourists and people are likely to move around while they are on the tour which introduces unpredictability in their location. The service distributor needs to be aware of the available resources and their capabilities when distributing service to them. It is important to consider which services should be deployed to the drone. This scenario illustrates the problems involved in service distribution on nodes (taking into account varying hardware and data processing capabilities) and taking into account data communication costs and resource usage.

5.3.1 Assumptions

- In our experiments, we assumed that fog computing nodes have low capability nodes on memory, and storage when comparing with cloud nodes. We have generated random data for the experiments. The generated data has 120 services with different technical requirements ranging between R1 and R6. Similarly, we have generated data for the fog nodes with various capabilities ranging between C1 and C4 cloud nodes capabilities ranging between C1 and C6.

- Also, for our experiments we have used the services that are used in Chapters 3 and 4 and the process of using them is as follows: We have used services (Si) namely data transformation (S1), decision tree (S2), Naïve Bayes (S3) and logistic regression (S4) as services that need to be distributed to the nodes either fog or cloud nodes based on the services demand and the capabilities of the nodes. To obtain the technical requirements of the services, we have run the services and observe the technical requirements of each service and take note of the required memory to be run on nodes. Then, we will apply mapping process based on the service requirements that are obtained from observations.

- The technical requirements of services and the capabilities of the nodes are known.

- The speed of processing a data by a fog or cloud nodes relies on the data size and the types of nodes' configuration. This means that powerful nodes in terms of configuration will be able to process the low size data quicker.

- The connectivity between nodes are not considered in our experiments as it is out of the scope of these experiments.

Limitations

- Capabilities of nodes like memory and storage were designed to mimic the real existing nodes, but in a simplified way. In addition, the cloud nodes are considered as powerful nodes. The reason is that real-world configurations are complicated when considering various factors, which goes out of the scope of this project and to simplify the algorithms.
- In some of the experiments, we have used a generated data while considering the constraints of real-world like IoT nodes hardware power because of the absence of the data that are taken from real world.

process

- The process starts with generating the dataset for both the service requirements and nodes capabilities to make them ready for the mapping process, which is saved in CSV documents.
- The capabilities of fog and cloud nodes are preconfigured, and the capabilities of fog nodes are less powerful when compare it with the cloud.
- We use the service distribution strategy to map between service demands and the capabilities of the nodes.
- The services will be distributed either to fog or cloud nodes based on the requirements of the services and the capabilities of the nodes.

5.4 Methodology

5.4.1 Most Efficient IoT Node (MEIN)

We propose a distribution strategy called "MEIN" Most Efficient IoT Node; it uses a list of all the IoT devices present in the network. This strategy allows defining the order to use devices to execute the different services. We have used the bin packing problem as a baseline as it is one of the most common optimisation problems. The criterion to sort the list of devices is physical capabilities of devices like CPU, RAM and storage. Thus, all devices capabilities are stored in a list for each physical aspect.

The main goal is to distribute the services in an efficient and optimised way by expressing the goal in the following ways:

The services are stored in the cloud because of the unlimited power and storage of Cloud Node (CN). We begin with deploying services from the Cloud Node to Fog Node (FN) to prepack the services for users' request. Then, the Fog Node will distribute the services either to the IoT Nodes (IoTN) or to cloud node based on the capabilities of nodes and the computational requirements of the services. Therefore, to get the results we need to use the three nodes including Cloud Node, Fog Node and IoT Node to process the service in an efficient way.

The IoT nodes have limited computational capabilities as they are constrained devices which makes the process of distributing services to these devices much more difficult than distributing the services to cloud nodes because of the computational power differences between the IoT nodes and the cloud nodes. Also, the services need computational requirements to run on IoT nodes such as RAM and storage. We will take the configuration of

0

nodes C(1) (1 GB RAM, 8 GB storage) as a baseline that is given earlier in the motivating scenarios as parameters for nodes' configurations. For example, if a service requires 256 MB RAM and 2 GB storage, then it means that it requires the quarter of the baseline configuration.

Objective Function

minimize/maximize $f_m(x)$

$m = \{$ min−data, max−node, max−number $\}$

f is the objective function, m is the set of objectives and x is a vector of variables.

Example 1. *Services:* $\{S1_{R1}, S2_{R1}, S3_{R1}, S4_{R2}, S5_{R2}, S6_{R3}, S7_{R3}, S8_{R4}, S9_{R5}, S10_{R5}, S11_{R6}\}$;

Number of fog nodes = 4;

Fog nodes' capabilities = $\{FN_{C1}, FN_{C2}, FN_{C3}, FN_{C4}\}$;

Number of Cloud nodes = 6;

Cloud nodes' capabilities = $\{CN_{C1}, CN_{C2}, CN_{C3}, CN_{C4}, CN_{C5}, CN_{C6}\}$;

Fog Node 1: Capability: C1

Service 1 - Requirement: R1

Service 2 - Requirement: R1

Service 3 - Requirement: R1

75% of the capability of node 1 is used

Fog Node 2: Capability: C2

Service 4 - Requirement: R2

Service 5 - Requirement: R2

Service 6 - Requirement: R3

100% of the capability of node 2 is used

Fog Node 3: Capability: C3

Service 7 - Requirement: R3

Service 8 - Requirement: R4

75% of the capability of node 3 is used

Fog Node 4: Capability: C4

Service 9 - Requirement: R5

Service 10 - Requirement: R5

100% of the capability of node 4 is used

Transferred services to the cloud: {Service 11};

Cloud nodes are required to run the following services

[Service 11 - Requirement: R6]

Cloud Node 4: Capability: C4

Service 11 - Requirement: R6

100% of the capability of node 4 is used

MEIN distribution strategy is used to distribute a part of or all the services to a constant number of IoT nodes based on their capabilities or to an unlimited number of cloud nodes, with different capabilities with the distribution strategy. Also, the distribution strategy is for maximising the usage of node.

Maximising the usage of node. This part of the strategy distributes the service to the node that is the first lowest capability node among the free nodes. The strategy searches the complete list of nodes' capabilities to find the smallest node whose capability is greater than or equal to the requirement of service.

Algorithm 1 maximising the usage of node

```
    Input:  NodeCap, ServiceReq, DistID
    Output:  Distributed services to fog nodes and cloud nodes
 1: for $i = 0$ to $DistID$ length do
 2:     $DistID[i] = -1$
 3: end for
 4: for $i = 0$ to $ServiceReq$ length do
 5:     $bestID = -1$
 6:     for $j = 0$ to $NodeCap$ length do
 7:         if $NodeCap[j] >= ServiceReq[i]$ then
 8:             if $bestID = -1$ then
 9:                 $bestID = j$
10:             else if $NodeCap[bestID] > bestID[j]$ then
11:                 $bestID = j$
12:             end if
13:         end if
14:         if $bestID != -1$ then if (bestIdx != -1)
15:             $DistID[i] = bestID$
16:             $NodeCap[bestID]$-= $ServiceReq[i]$
17:         end if
18:     end for
19: end for
```

In Algorithm 1, we give the code of maximising the usage of node while distributing the services to the nodes based on their capabilities. Firstly, we need to input services requirements and nodes capabilities. Then, we initialise all the nodes to become free. After that, we pick each service and find the minimum node capability that could have the current service. In other words, find Min($NodeCap_{[1]}$, $NodeCap_{[2]}$,... $NodeCap_{[n]}$) $\rightarrow Service_{[current]}$, if a node is found then allocate the node to the current service. However, If a node could not be found then leave that service and continue checking the more services.

Distributing services with maximum technical requirements This approach is based on Multiple knapsack problem which will be used to compare with the previous approach to check which one meets the criteria of maximising the usage of fog nodes while maintaining the unused capabilities to minimum. Given a set of services S containing the computational requirements of 'R' distinct services, and a set of IoT Nodes (IoTN) that can withstand $IoTN_1$, $IoTN_2$, ..., $IoTN_n$ capabilities, the solution is to find the sum of the largest subset of the services 'S', that can be distributed to the IoT nodes. Algorithm 2 illustrates a solution that recursively tries all the ways of distributing services to fill the nodes and select the one that can handle the maximum service requirement. Then, determining the states of meinDP, that we built our algorithm upon. This can be be represented in states like (NoServices, $remainNode_1$, ...,$remainNode_n$), 'NoServices' is the index, $remainNode_1$ is the remaining capability of first node, and $remainNode_n$ is the remaining capability of n node. In this solution, we tried to solve the problem by using multidimensional dynamic programming.

Example 2. *Services: {R1, R1, R1, R2, R2, R3, R3, R4, R5, R5, R6};*

Number of IoT nodes = 4;

IoT nodes' capabilities = {C1, C2, C3, C4};

Number of Cloud nodes = 6;

Cloud nodes' capabilities = {C1, C2, C3, C4, C5, C6};

Fog Node 1: Capability: C1

Algorithm 2 Distributing a subset of the services that have maximum technical requirements

```
**Input:** NodeCap, ServiceReq, NoServices, remainNode, meinDP
**Output:** Distributed services on fog nodes and cloud nodes
$maxService(ServiceReq, NoServices, remainNode_1, remainNode_n, i)$
if $i == NoServices$ then
    $return\ 0;$
end if
if $meinDP[i][remainNode_1][remainNode_n]! = -1$ then
    $return\ meinDP[i][remainNode_1][remainNode_n]$
end if
$DistS_1\ DistS_2\ DistS_n$
$DistNone$
if $remainNode_1 >= ServiceReq[i]$ then
    $DistS_1 = ServiceReq[i] + maxService(ServiceReq, NoServices,$
    $remainNode_1 - ServiceReq[i], remainNode_n, i + 1);$
end if
if $remainNode_n >= ServiceReq[i]$ then
    $DistS_n = ServiceReq[i] + maxService(ServiceReq, NoServices,$
    $remainNode_1, remainNode_n - ServiceReq[i], i + 1);$
end if
$DistNone = maxService(ServiceReq, NoServices,$
$remainNode_1, remainNode_n, i + 1);$
$meinDP[i][remainNode_1][remainNode_n] = Math.max(DistNone,$
$Math.max(remainNode_1, remainNode_n));$
```

Service 4 - Requirement: R2

Service 5 - Requirement: R2

100% of the capability of node 1 is used

Fog Node 2: Capability: C2

Service 8 - Requirement: R4

100% of the capability of node 2 is used

Fog Node 3: Capability: C3

Service 10 - Requirement: R5

100% of the capability of node 3 is used

Fog Node 4: Capability: C4

Service 11 - Requirement: R6

100% of the capability of node 4 is used

Transferred services to the cloud: {Service 1, Service 2, Service 3, Service 6, Service 7, Service 9};

Cloud nodes are required to run the following services

[Service 1: Requirement R1, Service 2: Requirement R1 , Service 3: Requirement R1 , Service 6: Requirement R3, Service 7: Requirement R3, Service 9: Requirement R5]

Cloud Node 6: Capability: C6

Service 1 - Requirement R1

Service 2- Requirement R1

Service 3 - Requirement R1

Service 6 - Requirement R3

Service 7 - Requirement R3

Service 9 - Requirement R5

21.1% of the capability of node 6 is used

In the above solutions, we are not decomposing the services into microservices instead we distribute service to one of the nodes either IoT node or Cloud node as a whole based on their capabilities.

5.4.2 Architecture

The architecture of our proposal has been divided into two main parts including Cloud computing and IoT Environments. First, the cloud computing side has the ability to handle all the data and services and deploy the required service to the IoT environment. Second, the IoT environment part includes two sub parts, Fog Computing and the IoT Nodes. Fog computing is responsible to distribute the services to the nodes in an efficient and optimised way by maintaining the resources usage in a way that all available resources should be used. The IoT nodes are IoT smart devices that will get the services from fog computing and serve the users by providing the available services. The aim is to optimise the use of available resources in an efficient way, save energy, reduce the data transmission traffic over the network by distributing the data locally as much as possible. The communications among the cloud computing, Fog computing and IoT nodes can be undertaken in different ways as is typical (such as WIFI, 3G and Bluetooth). Figure 5.2 shows the architecture of the system. The architecture includes three types of the node including IoT devices, fog, and cloud computing as follows:

- **The IoT node** is at the edge of the network of the system and it is typically embedded in physical everyday objects. IoT nodes are small-sized and cheap in terms of price to make the process of deploying IoT devices to objects easy and inexpensive. It delivers services to users when they are required. These nodes are connected to the Fog nodes via Bluetooth which is considered an inexpensive network connection as mentioned earlier. The IoT environment should be helpful in many ways including energy saving, less cost, optimised use of resources and reducing the data transmission cost over the network.

- **The Fog node** resides next to the IoT nodes in the IoT Environment or along the communication path to the cloud and collects services (or data received from an upstream Cloud node) and applies data distribution strategy to distribute the services to the nodes in the IoT environment in an optimised way. For our proposal the fog computing form as the main node. Obviously, a fog node has less power and a less global data view than a cloud node and hence it can store fewer data and distribute limited services. In an investigation into energy limitation, the authors of [25] found that these devices have restricted energy for particular tasks.

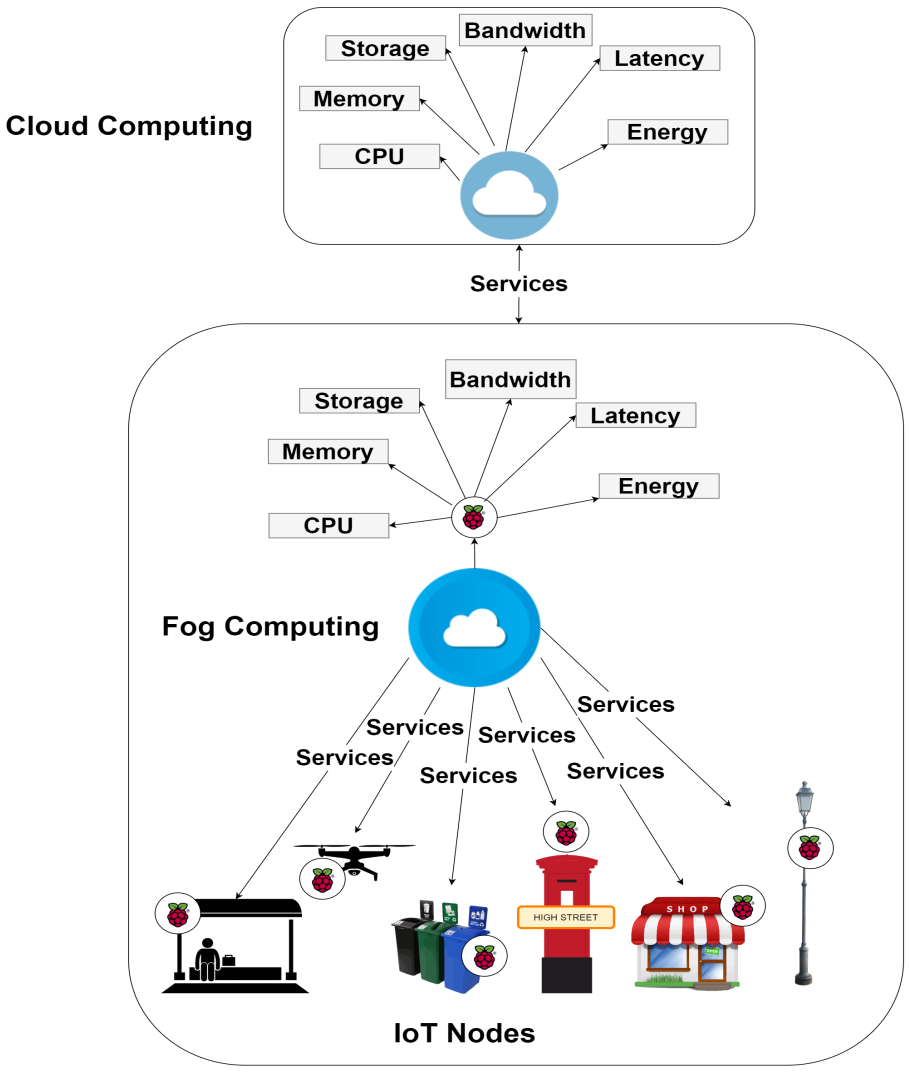


Figure 5.2: The distribution process

- **The Cloud node** resides in the cloud and process services that come from fog nodes. It is clear that the processing power and storage capability of the cloud is higher than fog node and IoT node. This power can be used even more effectively by using the presented approach. According to [25], there is no energy limitation in the devices which are in the cloud.

5.5 Experimental setup and results

We have conducted 15 experiments to find the most effective configuration setup of fog nodes to distribute the services as much as possible in fog nodes. Every experiment has a different configuration of capabilities of fog nodes as shown in Table 5.3. This table shows the each experiment name in the first column and the capabilities of fog nodes in the second column. In experiments 1 to 15, we have used different capabilities (C1, C2, C3 and C4) for fog nodes (40 fog nodes) to distribute 120 services to these fog nodes as much as possible. The distribution of capabilities in different ways and different amounts, with either more specialised nodes for a very generic ones (running all capabilities). In addition to the 15 experiments, we have conducted two experiments with the second approach in 5.4.1 and the capabilities of the nodes are the same as Experiment 12 (meinC1C2C3C4).

Table 5.3: Capabilities setup of fog nodes

Experiment no.	Fog Nodes with Capabilities			
	C1	C2	C3	C4
Experiment 1 (meinAllC1)	40 Fog nodes			
Experiment 2 (meinC1C2)	20 Fog nodes	20 Fog nodes		
Experiment 3 (meinC1C2C3)	13 Fog nodes	13 Fog nodes	14 Fog nodes	
Experiment 4 (meinC1C3)	20 Fog nodes		20 Fog nodes	
Experiment 5 (meinC1C4)	20 Fog nodes			20 Fog nodes
Experiment 6 (meinC2C3)		20 Fog nodes	20 Fog nodes	
Experiment 7 (meinC2C4)		20 Fog nodes		20 Fog nodes
Experiment 8 (meinC3C4)			20 Fog nodes	20 Fog nodes
Experiment 9 (meinC1C2C4)	13 Fog nodes	13 Fog nodes		14 Fog nodes
Experiment 10 (meinC1C3C4)	13 Fog nodes		13 Fog nodes	14 Fog nodes
Experiment 11 (meinC2C3C4)		13 fog nodes	13 fog nodes	14 fog nodes
Experiment 12 (meinC1C2C3C4)	10 Fog nodes	10 fog nodes	10 fog nodes	10 fog nodes
Experiment 13 (meinAllC2)		40 fog nodes		
Experiment 14 (meinAllC3)			40 fog nodes	
Experiment 15 (meinAllC4)				40 fog nodes

5.5.1 Experiments

The aim of the experiments was primarily to evaluate the feasibility of MEIN distribution strategy in the IoT environment. We will have 2 groups of devices, the first group of devices with limited power belong to the fog. The second group of devices with unlimited computational power belong to the cloud. Firstly, the strategy will check which group of devices can handle the services by checking the services computational requirements and the capabilities of the devices in both groups (Fog and Cloud). Then the strategy will select the best group that can efficiently process the services. Generally, the cloud has unlimited power and the best choice based on the capabilities but as our intention is to distribute the services as much as possible near to the data source which is the reason to do a checking point before applying the strategy. After that, the strategy will distribute the services to the devices in the selected groups based on their capabilities and the services computational requirements efficiently.

In the cloud, the devices (Cloud Nodes) can be extendable as whenever it is required we can add a node to handle the services possibly by adding a big set of devices for the cloud. We use unlimited power devices which do not necessarily have to be unlimited but we can have an unlimited number of devices that can handle all the services which mean we can add a device with the best configuration or multiple devices with the different configurations in the cloud which reflects the reality of cloud computing.

However, in the fog, the devices with limited capabilities, so when the services cannot be handled by the fog we transfer them to the cloud. The transferring step should be done earlier in the process before selecting the fog or cloud, so we have a device like a switch in the fog that allows us to switch to cloud or fog nodes based on the capabilities of each group. In the switch device, we check whether the services can be handled in fog or cloud.

5.5.2 Data description

We have generated random data for the experiments with the distribution strategy. The generated data has 120 services with different technical requirements ranging between R1 and R6. Similarly, we have generated data for the fog nodes with various capabilities ranging between C1 and C4 cloud nodes capabilities ranging between C1 and C6.

5.5.3 Results

In this section, we will discuss the results of the distribution strategy. As we mentioned earlier, the strategy will firstly distribute the services to the fog nodes based on their capabilities, then, distribute the undistributed services in the fog to the cloud nodes. In both levels, we have applied the same distribution strategy. Every bar chart has one bar called the number of processed services in a node. In addition, there is a table in the graph that shows the capabilities of the nodes (C1 to C4), the identity of each fog nodes (FN1 to FN40) and the technical requirements of each service (R1 to R6). In fog nodes distribution strategy, there are 40 fog nodes that are ranging from capability C1 to C4 and they are sorted in the graphs from C1 to C4. However, there are 20 cloud nodes that are ranging from C3 to C6, but the cloud nodes can be extended when it is required. In the experiments, we tried to distribute the minimal service requirements to the nodes in a way to use the capabilities of the nodes with the services that need minimum technical requirements while maintaining the usage of nodes to maximising the usage of node. There are two bar charts namely fog nodes and cloud nodes for each experiment. Both charts belong to the same strategy, but the aim is to show the distribution strategy process in both levels, the fog and cloud.

Experiment 1 (MEINAllC1): Figure 5.3 shows that in this experiment the fog nodes FN 1 to F35 were given the capability of C1 = (1024 MB RAM and 8 GB storage) which means that every fog node can process 4 services with technical requirement R1 = (256 MB RAM and 2 GB storage), 2 services with the technical requirement, R2 = (512 MB RAM and 4 GB storage) or 1 service with technical requirement R3 = (1024 MB RAM and 8 GB storage). In other words, every fog node can handle four services because every service requires a quarter of the technical requirement of the fog node, so all the capabilities of the nodes can be used and there are no wastage. There are 60 services that have been distributed to 35 fog nodes and the remaining 60 services that could not be distributed to fog nodes and they are transferred to the cloud nodes.

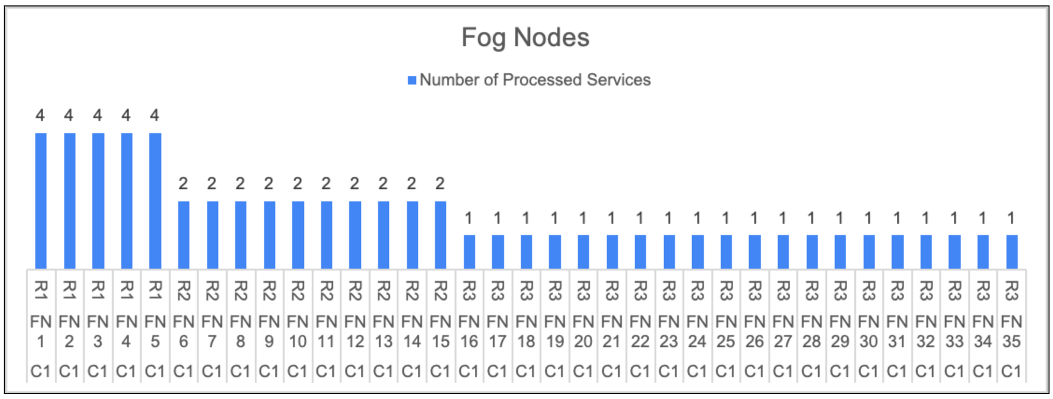


Figure 5.3: Experiment 1. Fog nodes

Similarly, Figure 5.4 shows the 60 distributed services to the cloud nodes using the same strategy. In the cloud, 20 cloud nodes are used out of 20 nodes because the services require the full power of the cloud as the fog nodes could not handle most of the services because of the limited power of fog nodes. Moreover, the cloud node CN 8 handled two different service requirements to maximise the usage of node. For example, CN 8 with the capability of C4 that handled three services with service requirements of R4 and R5, in other words, S1 (2048 MB, 16 GB), S2 (2048 MB, 16 GB), S3 (4096 MB, 32 GB) -> CN 8 (8192 MB, 64 GB).

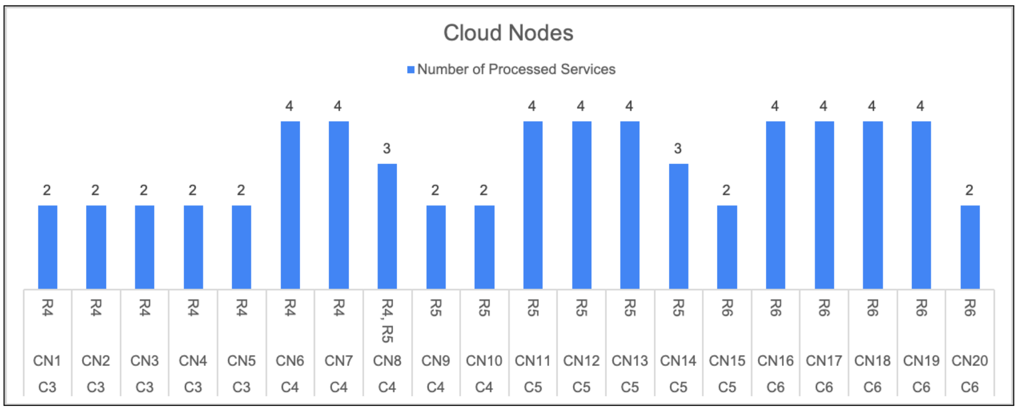


Figure 5.4: Experiment 1. Cloud nodes

Experiment 2 (MEINAllC2): Figure 5.5 shows that in this experiment the fog nodes FN 1 to F38 were given the capability of C2 = (2048 MB RAM and 16 GB storage). As shown in Figure 5.5 that the fog nodes FN 3 and FN 8 handled two different service requirements to maximise the usage of nodes. For example, FN 3 with the capability of C2 that handled 6 services with service requirements of R1 and R2, in other words, S1 (256 MB, 2 GB), S2 (256 MB, 2 GB), S3 (256 MB, 2 GB), S4 (256 MB, 2 GB), S5 (512 MB, 4 GB), S5 (512 MB, 4 GB) -> FN 3 (2048 MB, 16 GB). There are 80 services that have been distributed to 38 fog nodes and the remaining 40 services that could not be distributed to fog nodes and they are transferred to the cloud nodes.

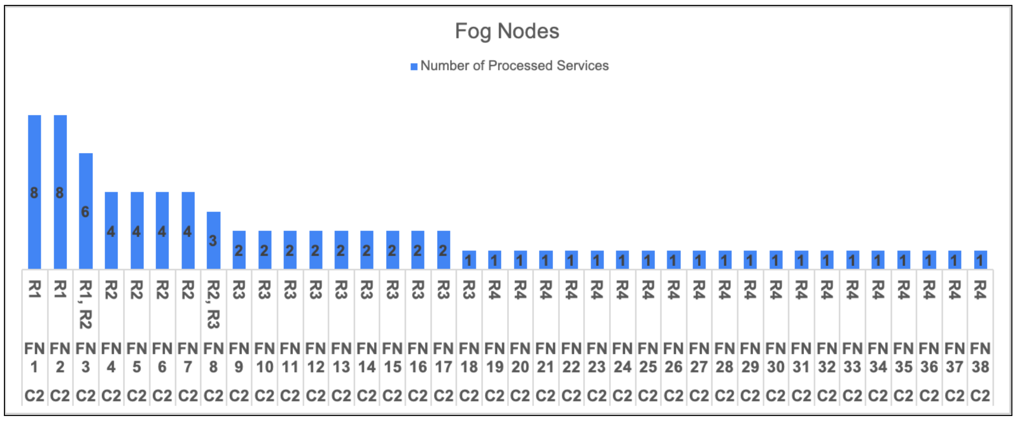


Figure 5.5: Experiment 2. Fog nodes

Similarly, Figure 5.6 shows the 40 distributed services to the cloud nodes using the same strategy. In the cloud, 19 cloud nodes are used out of 20 nodes because the services require the full power of the cloud as the fog nodes could not handle all of the services because of the limited power of fog nodes. Moreover, the cloud node CN 12 handled two different service requirements to maximise the usage of nodes. For example, CN 12 with the capability of C5 that handled two services with service requirements of R5 and R6, in other words, S1 (4096 MB, 32 GB), S2 (8192 MB, 64 GB), -> CN 12 (16384 MB, 128 GB).

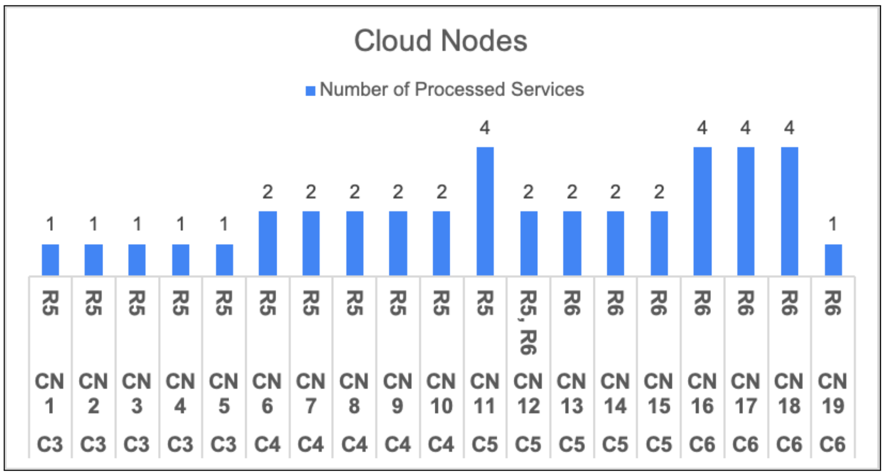


Figure 5.6: Experiment 2. Cloud nodes

Experiment 3 (MEINAllC3): Figure 5.7 shows that in this experiment the fog nodes FN 1 to F39 were given the capability of C3 = (4096 MB RAM and 32 GB storage). As shown in Figure 5.7 that the fog node FN 2 handled two different service requirements to maximise the usage of nodes. For example, FN 2 with the capability of C3 that handled 10 services with service requirements of R1 and R2, in other words, S1 (256 MB, 2 GB), S2 (256 MB, 2 GB), S3 (256 MB, 2 GB), S4 (256 MB, 2 GB), S5 (256 MB, 2 GB), S6 (512 MB, 4 GB), S7 (512 MB, 4 GB), S8 (512 MB, 4 GB), S9 (512 MB, 4 GB)), S10 (512 MB, 4 GB) -> FN 2 (4096 MB, 32 GB). There are 100 services that have been distributed to 39 fog nodes and the remaining 20 services that could not be distributed to fog nodes are transferred to the cloud nodes.

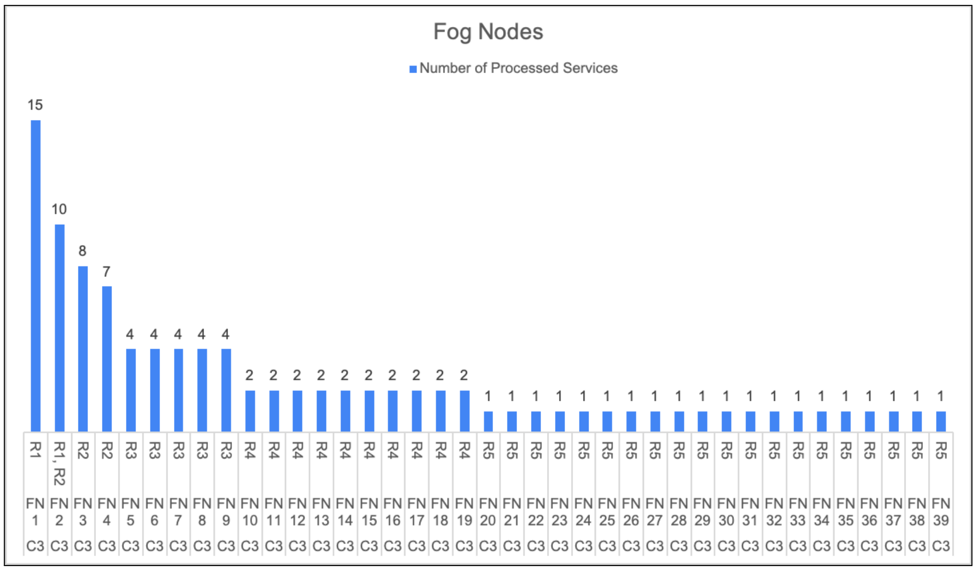


Figure 5.7: Experiment 3. Fog nodes

Similarly, Figure 5.8 shows the 20 distributed services to the cloud nodes using the same strategy. In the cloud, 12 cloud nodes are used out of 20 nodes because the services did not require the remaining nodes as they are powerful in terms of processing power to handle the available services.

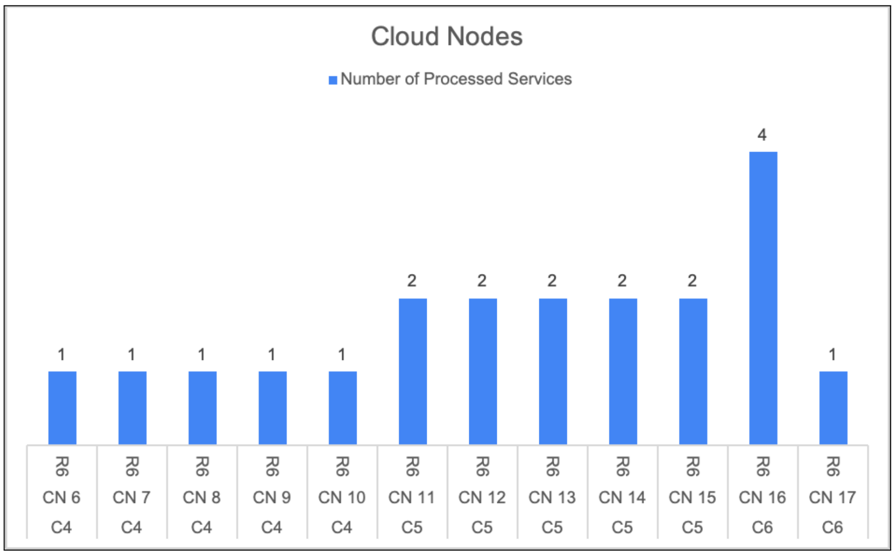


Figure 5.8: Experiment 3. Cloud nodes

Experiment 4 (MEINAllC4): Figure 5.9 shows that in this experiment the fog nodes FN 1 to F40 were given the capability of C4 = (8192 MB RAM and 64 GB storage). As shown in Figure 5.9 that the fog node FN 1, FN 5 and FN 10 handled two different service requirements to maximise the usage of nodes. There are 120 services that have been distributed to 40 fog nodes. This means that all the services have been distributed to the fog nodes because the capabilities of the fog nodes were powerful enough to handle the available services and no services have been transferred to the cloud.

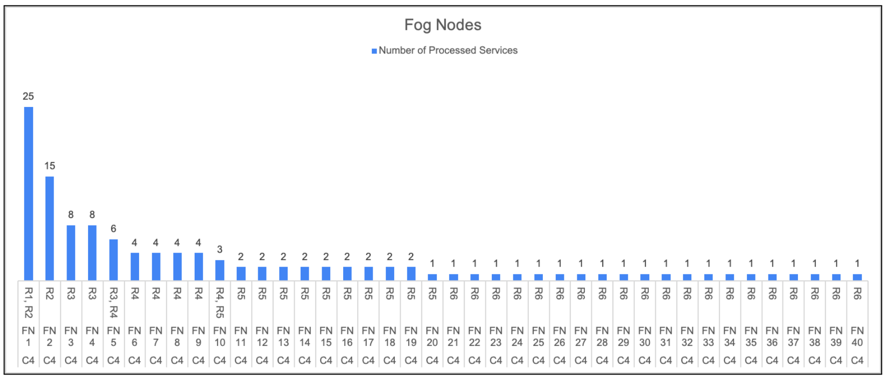


Figure 5.9: Experiment 4. Fog nodes

Experiment 5 (MEINC1C2): Figure 5.10 shows that in this experiment the fog nodes FN 1 to 20 were given the capability of C1 = (1024 GB RAM and 8 GB storage) and the fog nodes FN 21 to 40 have the capability of C1 = (2048 GB RAM and 16 GB storage). There are 72 services out of 120 services have been distributed to 40 fog nodes and the remaining 48 services that are not distributed to the fog nodes are transferred to cloud nodes.

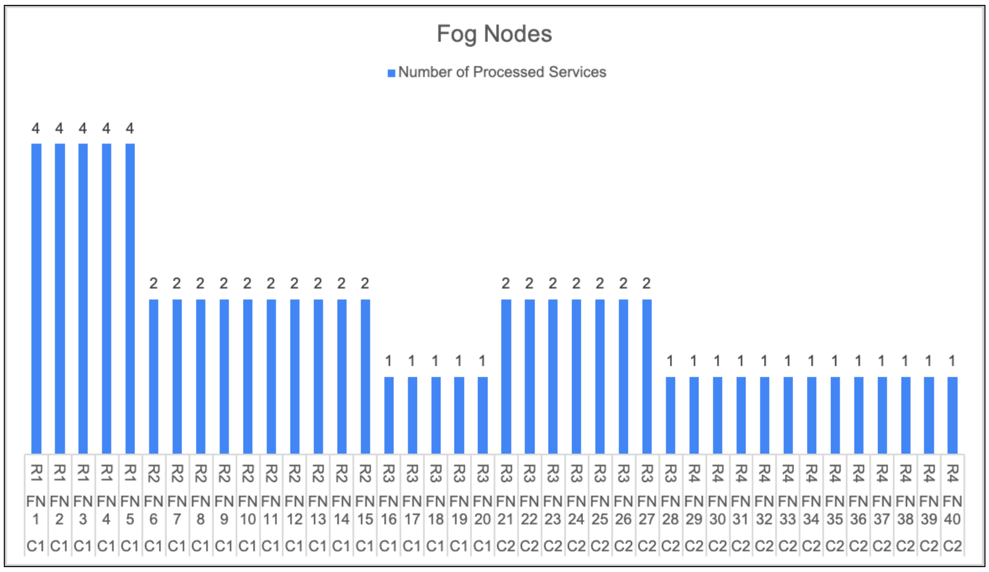


Figure 5.10: Experiment 5. Fog nodes

Similarly, Figure 5.11 shows the 48 distributed services to the cloud nodes using the same strategy. In the cloud, 19 cloud nodes are used out of 20 nodes because the services require the full power of the cloud as the fog nodes could not handle all of the services because of the limited power of fog nodes. Moreover, the cloud node CN 13 handled two different service requirements to maximise the usage of nodes. For example, CN 13 with the capability of C5 that handled two services with service requirements of R5 and R6, in other words, S1 (4096 MB, 32 GB), S2 (8192 MB, 64 GB) -> CN 13 (16384 MB, 128 GB).

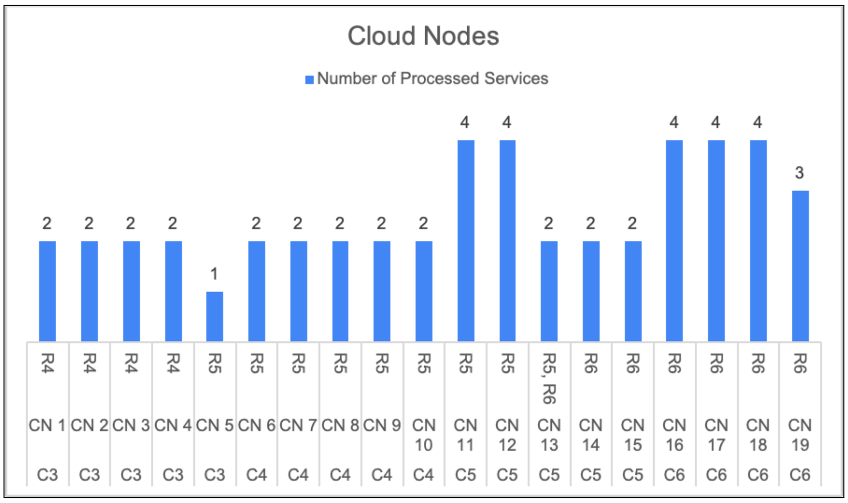


Figure 5.11: Experiment 5. Cloud nodes

Experiment 6 (MEINC1C2C3): Figure 5.12 shows that in this experiment the fog nodes FN 1 to 13 were given the capability of C1 = (1024 GB RAM and 8 GB storage), the fog nodes FN 14 to 26 have the capability of C1 = (2048 GB RAM and 16 GB storage) and the fog nodes FN 27 to 40 have the capability of C3 = (4096 GB RAM and 32 GB storage). There are 85 services out of 120 services have been distributed to 40 fog nodes and the remaining 35 services that are not distributed to the fog nodes are transferred to cloud nodes.

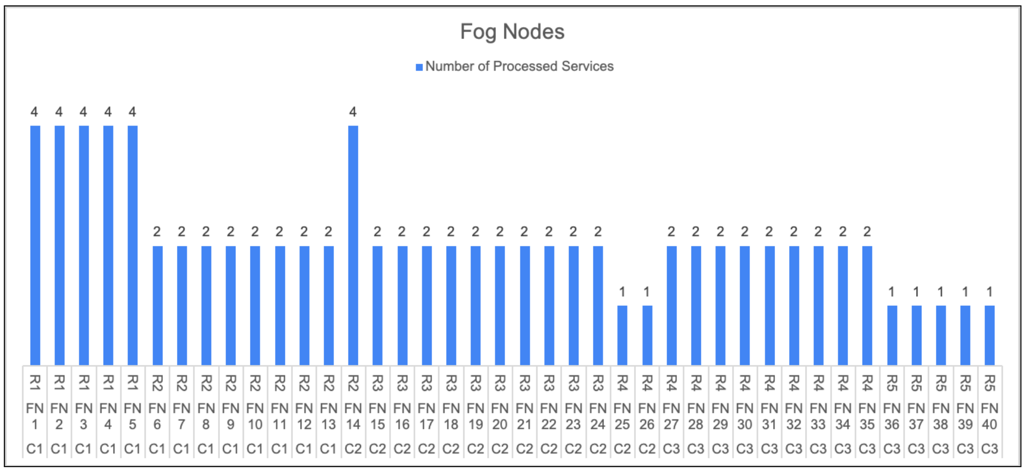


Figure 5.12: Experiment 6. Fog nodes

Similarly, Figure 5.13 shows the 35 distributed services to the cloud nodes using the same strategy. In the cloud, 18 cloud nodes are used out of 20 nodes because the services require the full power of the cloud as the fog nodes could not handle all of the services because of the limited power of fog nodes.

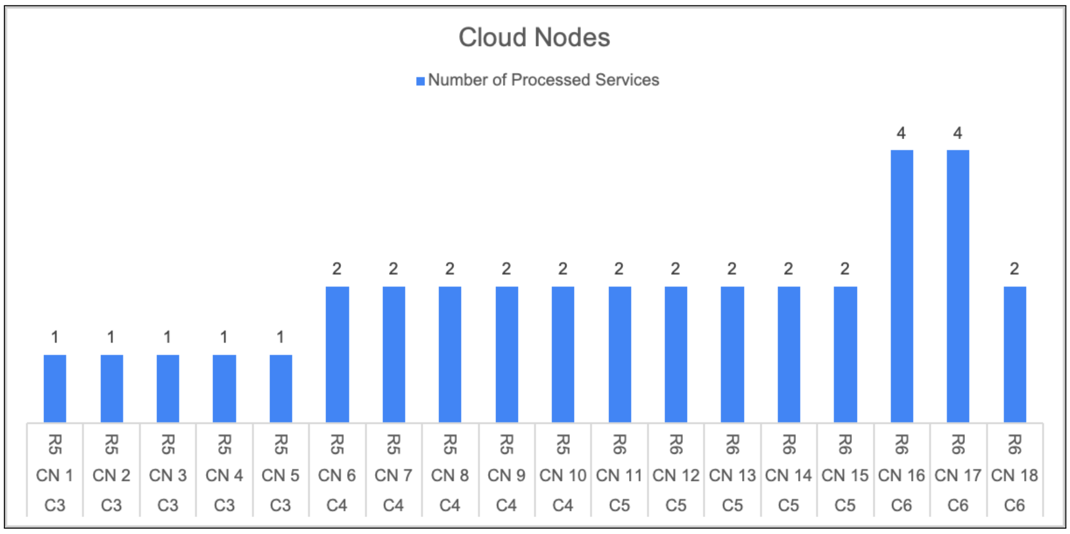


Figure 5.13: Experiment 6. Cloud nodes

Experiment 7 (MEINC1C2C4): Figure 5.14 shows that in this experiment the fog nodes FN 1 to 13 were given the capability of C1 = (1024 GB RAM and 8 GB storage), the fog nodes FN 14 to 26 have the capability of C2 = (2048 GB RAM and 16 GB storage) and the fog nodes FN 27 to 40 have the capability of C4 = (8192 GB RAM and 64 GB storage). There are 99 services out of 120 services have been distributed to 40 fog nodes and the remaining 21 services that are not distributed to the fog nodes are transferred to cloud nodes.

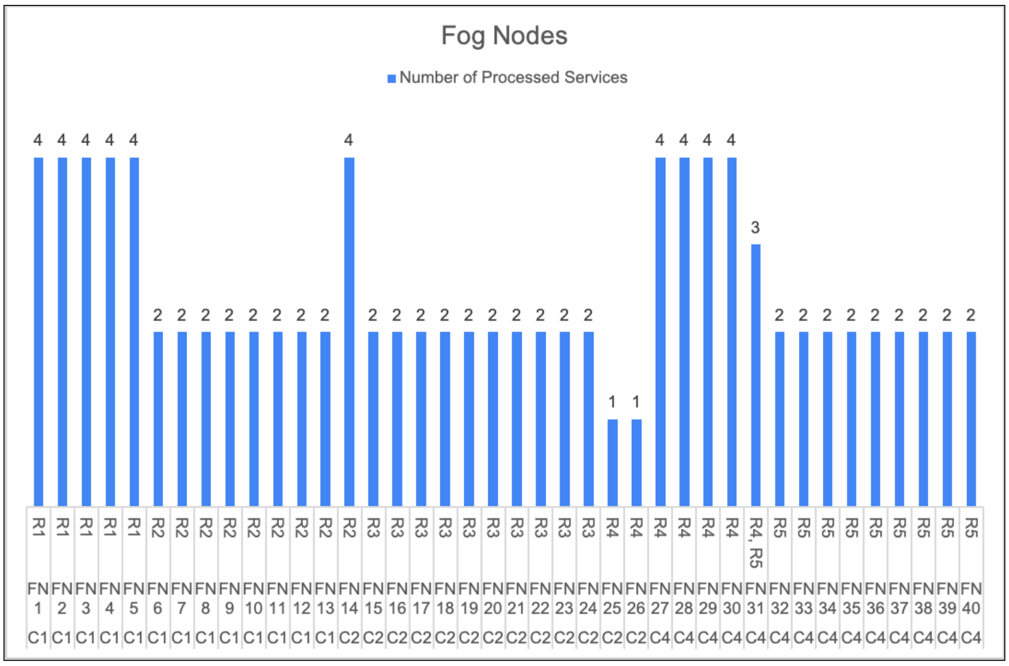


Figure 5.14: Experiment 7. Fog nodes

Similarly, Figure 5.15 shows the 21 distributed services to the cloud nodes using the same strategy. In the cloud, 13 cloud nodes are used out of 20 nodes because the services did not require the remaining nodes as they are powerful in terms of processing power to handle the available services.

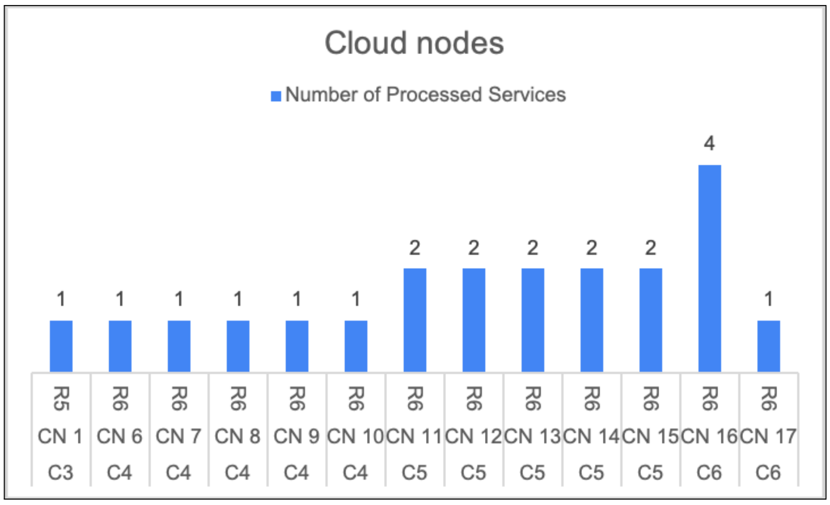


Figure 5.15: Experiment 7. Cloud nodes

Experiment 8 (MEINC1C3): Figure 5.16 shows that in this experiment the fog nodes FN 1 to 20 were given the capability of C1 = (1024 GB RAM and 8 GB storage) and the fog nodes FN 21 to 40 have the capability of C3 = (4096 GB RAM and 32 GB storage). There are 86 services out of 120 services have been distributed to 40 fog nodes and the remaining 34 services that are not distributed to the fog nodes are transferred to cloud nodes.

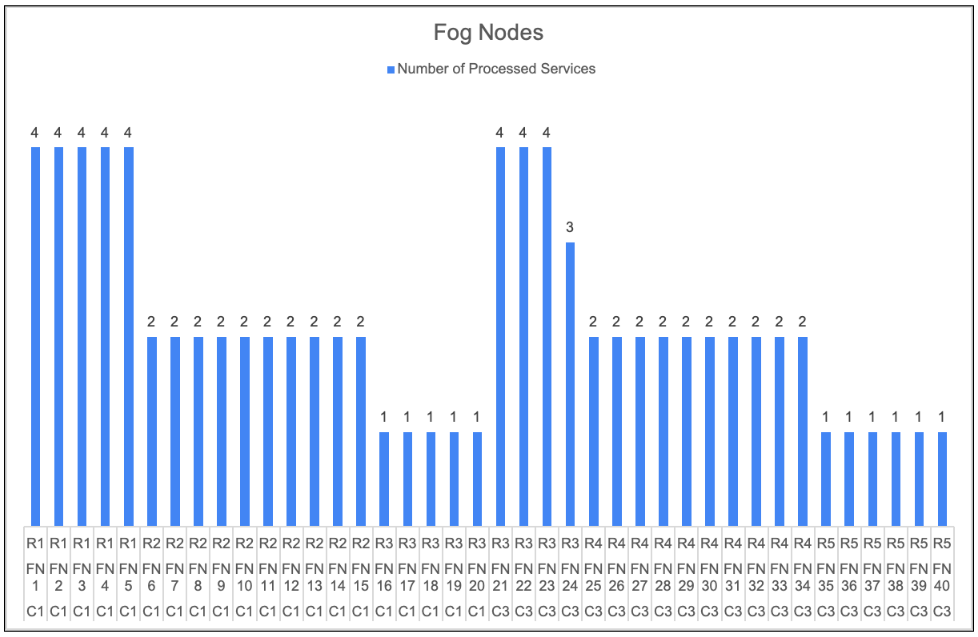


Figure 5.16: Experiment 8. Fog nodes

Similarly, Figure 5.17 shows the 34 distributed services to the cloud nodes using the same strategy. In the cloud, 18 cloud nodes are used out of 20 nodes because the services require the full power of the cloud as the fog nodes could not handle all of the services because of the limited power of fog nodes.

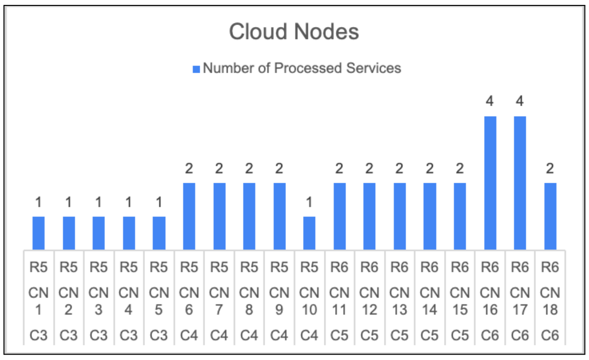


Figure 5.17: Experiment 8. Cloud nodes

Experiment 9 (MEINC1C3C4): Figure 5.18 shows that in this experiment the fog nodes FN 1 to 13 were given the capability of C1 = (1024 GB RAM and 8 GB storage), the fog nodes FN 14 to 26 have the capability of C3 = (4096 GB RAM and 32 GB storage) and the fog nodes FN 27 to 40 have the capability of C4 = (8192 GB RAM and 64 GB storage). There are 102 services out of 120 services have been distributed to 40 fog nodes and the remaining 18 services that are not distributed to the fog nodes are transferred to cloud nodes.

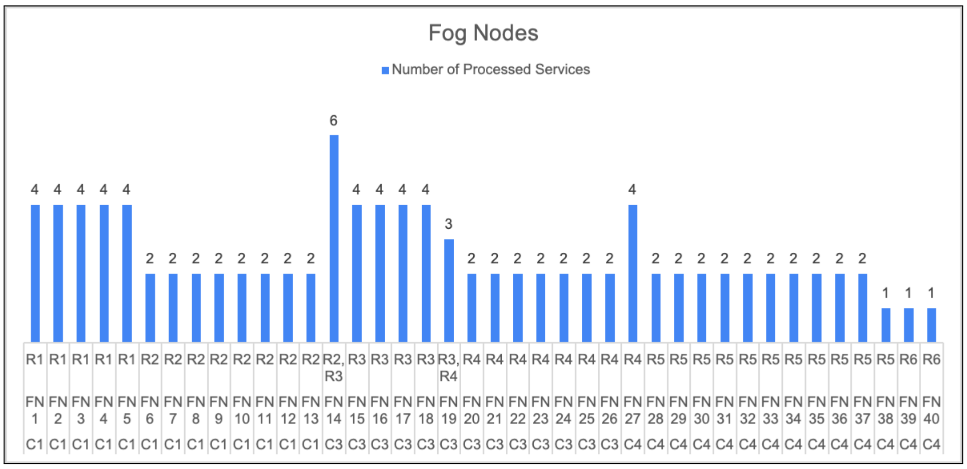


Figure 5.18: Experiment 9. Fog nodes

Similarly, Figure 5.19 shows the 18 distributed services to the cloud nodes using the same strategy. In the cloud, 11 cloud nodes are used out of

20 nodes because the services did not require the remaining nodes as they are powerful in terms of processing power to handle the available services.

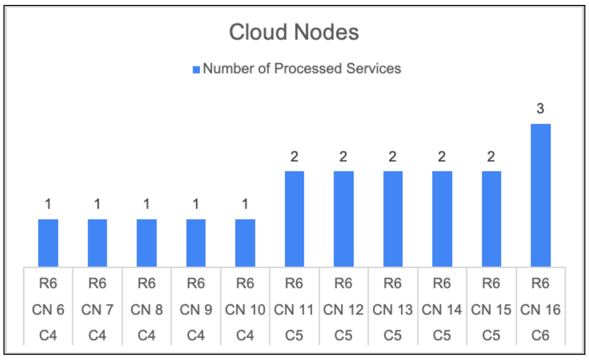


Figure 5.19: Experiment 9. Cloud nodes

Experiment 10 (MEINC1C4): Figure 5.20 shows that in this experiment the fog nodes FN 1 to 20 were given the capability of C1 = (1024 GB RAM and 8 GB storage) and the fog nodes FN 21 to 40 have the capability of C4 = (8192 GB RAM and 64 GB storage). There are 103 services out of 120 services have been distributed to 40 fog nodes and the remaining 17 services that are not distributed to the fog nodes are transferred to cloud nodes.

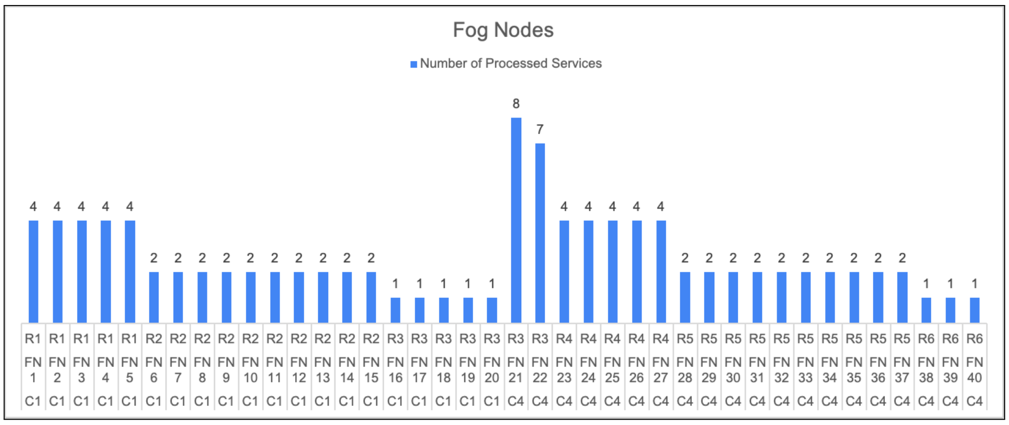


Figure 5.20: Experiment 10. Fog nodes

Similarly, Figure 5.21 shows the 17 distributed services to the cloud nodes using the same strategy. In the cloud, 11 cloud nodes are used out of 20 nodes because the services require the full power of the cloud as the fog nodes could not handle all of the services because of the limited power of fog nodes.

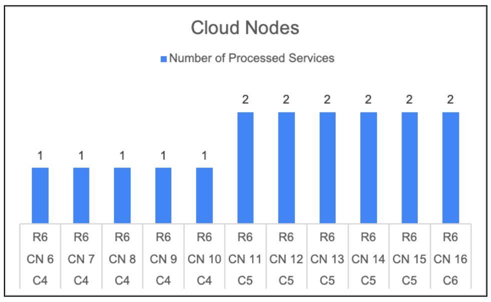


Figure 5.21: Experiment 10. Cloud nodes

Experiment 11 (MEINC2C3): Figure 5.22 shows that in this experiment the fog nodes FN 1 to 20 were given the capability of C2 = (2048 GB RAM and 16 GB storage) and the fog nodes FN 21 to 40 have the capability of C3 = (4096 GB RAM and 32 GB storage). There are 91 services out of 120 services have been distributed to 40 fog nodes and the remaining 29 services that are not distributed to the fog nodes are transferred to cloud nodes.

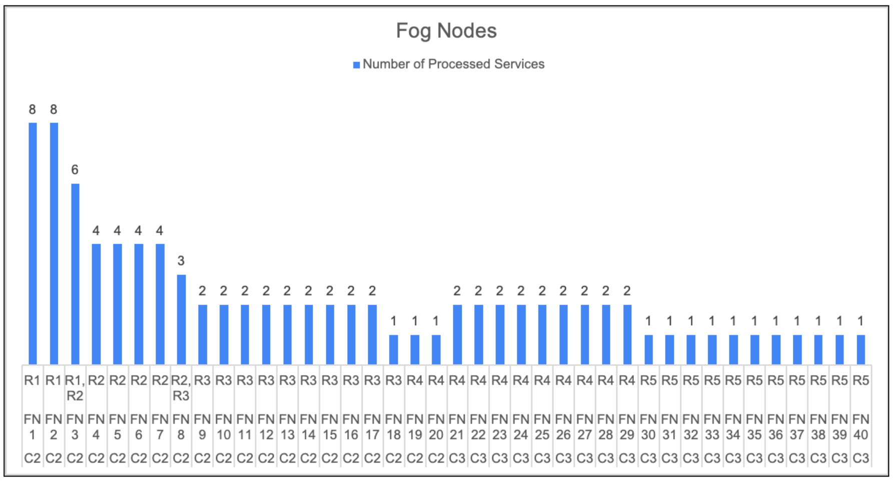


Figure 5.22: Experiment 11. Fog nodes

Similarly, Figure 5.23 shows the 29 distributed services to the cloud nodes using the same strategy. In the cloud, 17 cloud nodes are used out of 20 nodes because the services require the full power of the cloud as the fog nodes could not handle all of the services because of the limited power of fog nodes.

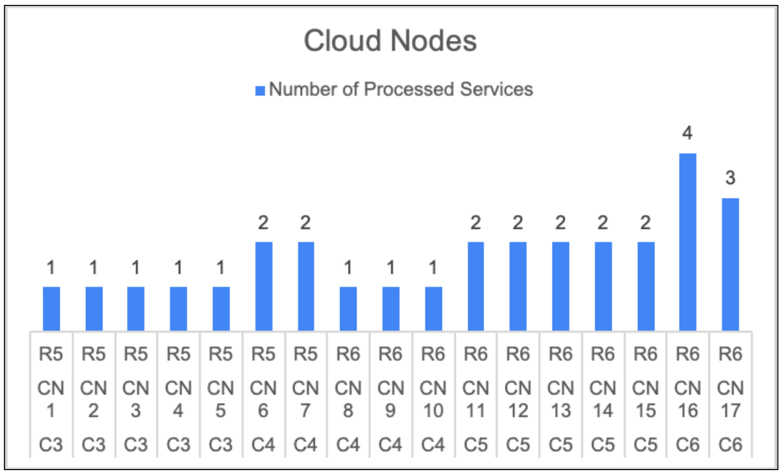


Figure 5.23: Experiment 11. Cloud nodes

Experiment 12 (MEINC2C3C4): Figure 5.24 shows that in this experiment the fog nodes FN 1 to 13 were given the capability of C2 = (2048 GB RAM and 16 GB storage), the fog nodes FN 14 to 26 have the capability of C3 = (4096 GB RAM and 32 GB storage) and the fog nodes FN 27 to 40 have the capability of C4 = (8192 GB RAM and 64 GB storage). There are 104 services out of 120 services have been distributed to 40 fog nodes and the remaining 16 services that are not distributed to the fog nodes are transferred to cloud nodes.

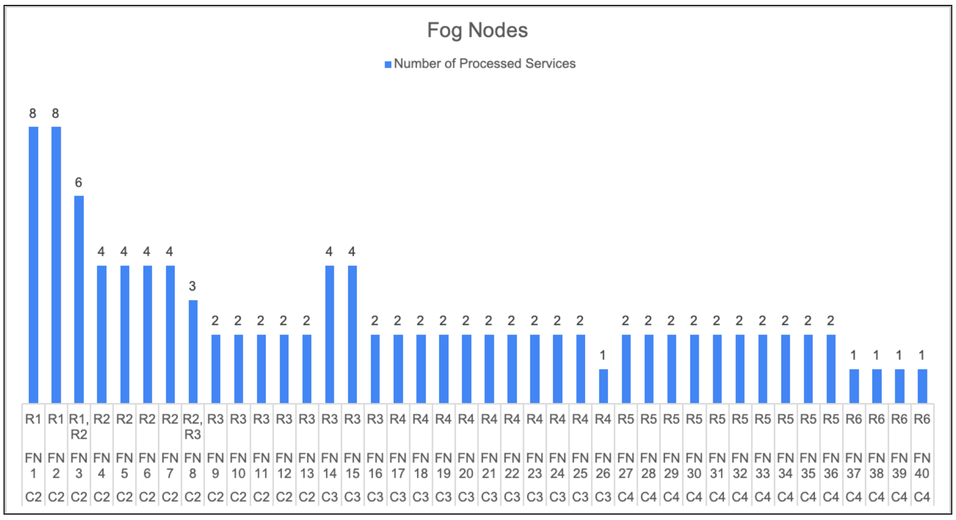


Figure 5.24: Experiment 12. Fog nodes

Similarly, Figure 5.25 shows the 16 distributed services to the cloud nodes using the same strategy. In the cloud, 11 cloud nodes are used out of 20 nodes because the services did not require the remaining nodes as they are powerful in terms of processing power to handle the available services.

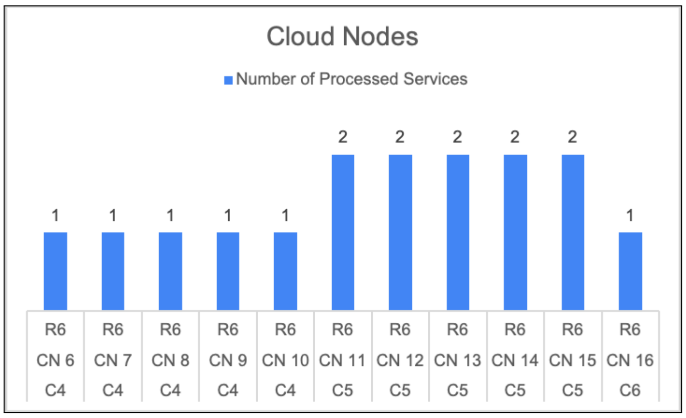


Figure 5.25: Experiment 12. Cloud nodes

Experiment 13 (MEINC2C4): Figure 5.26 shows that in this experiment the fog nodes FN 1 to 20 were given the capability of C2 = (2048 GB RAM and 16 GB storage) and the fog nodes FN 21 to 40 have the capability of C4 = (8192 GB RAM and 64 GB storage). There are 105 services out of 120 services have been distributed to 40 fog nodes and the remaining 15 services that are not distributed to the fog nodes are transferred to cloud nodes.

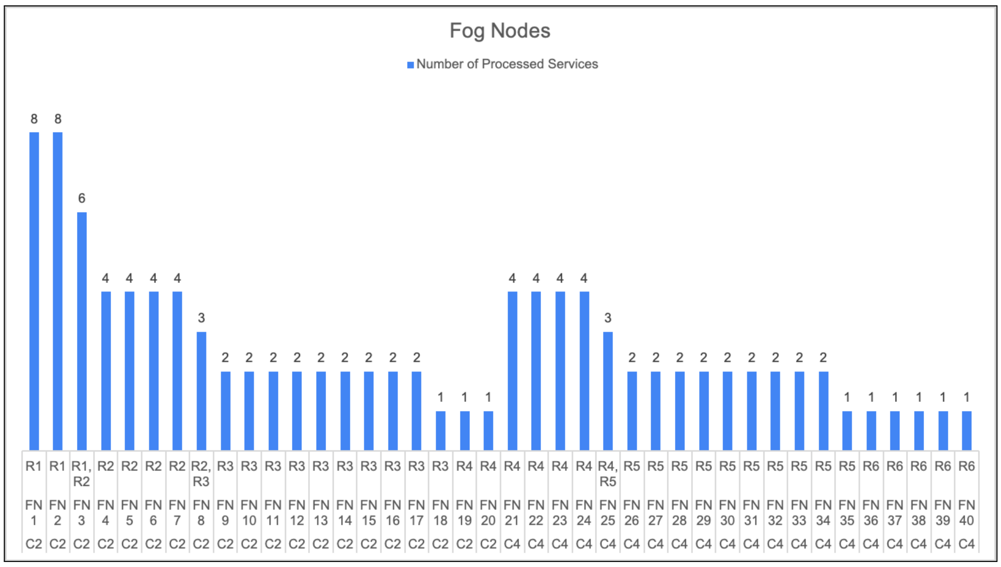


Figure 5.26: Experiment 13. Fog nodes

Similarly, Figure 5.27 shows the 15 distributed services to the cloud nodes using the same strategy. In the cloud, 10 cloud nodes are used out of 20 nodes because the services did not require the remaining nodes as they are powerful in terms of processing power to handle the available services.

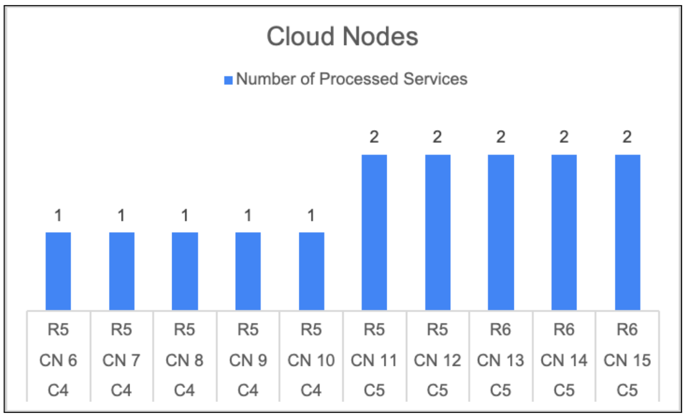


Figure 5.27: Experiment 13. Cloud nodes

Experiment 14 (MEINC3C4): Figure 5.28 shows that in this experiment the fog nodes FN 1 to 20 were given the capability of C3 = (4096 GB RAM and 32 GB storage) and the fog nodes FN 21 to 40 have the capability of C4 = (8192 GB RAM and 64 GB storage). There are 110 services out of 120 services have been distributed to 40 fog nodes and the remaining 10 services that are not distributed to the fog nodes are transferred to cloud nodes.

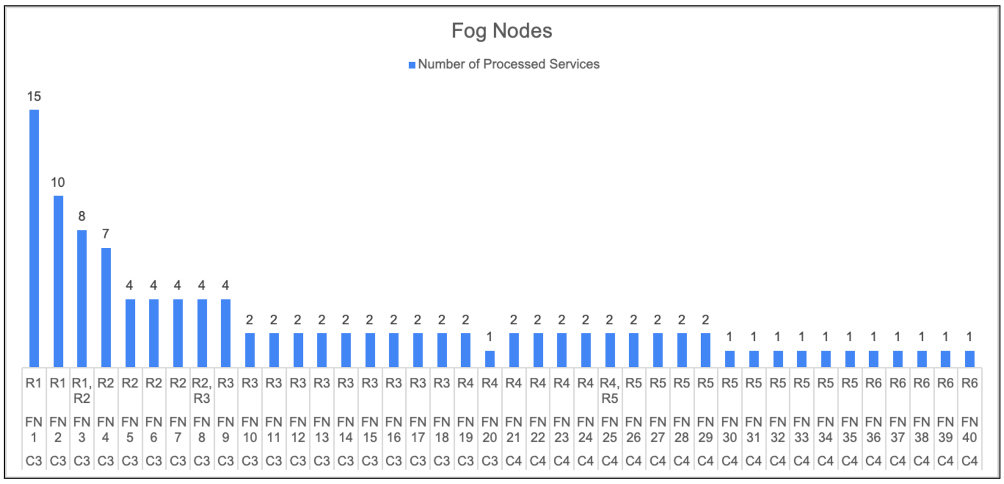


Figure 5.28: Experiment 14. Fog nodes

Similarly, Figure 5.29 shows the 10 distributed services to the cloud nodes using the same strategy. In the cloud, 8 cloud nodes are used out of 20 nodes because the services did not require the remaining nodes as they are powerful in terms of processing power to handle the available services.

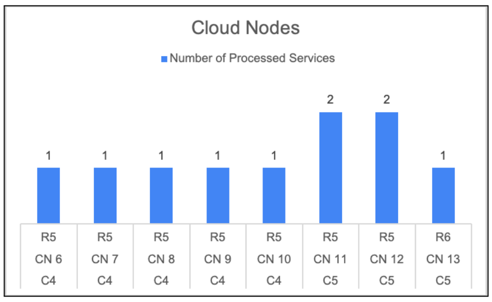


Figure 5.29: Experiment 14. Cloud nodes

Experiment 15 (MEINC1C2C3C4) Figure 5.30 shows that in this experiment the fog nodes FN 1 to 10 were given the capability of C1 = (1024 GB RAM and 8 GB storage) that process the 20 services with technical requirement R1 = (256 MB RAM and 2 GB storage). In other words, every fog node can handle 4 services because every service requires a quarter of the technical requirement of the fog node, so all the capabilities of the nodes are used and there are no wastages. Additionally, fog nodes FN 11 to 20 have the capability of C2 = (2048 GB RAM and 16 GB storage), fog nodes FN 21 to 30 have the capability of C3 = (4096 GB RAM and 32 GB storage) and fog nodes FN 31 to 40 have the capability of C4 = (8192 GB RAM and 64 GB storage).

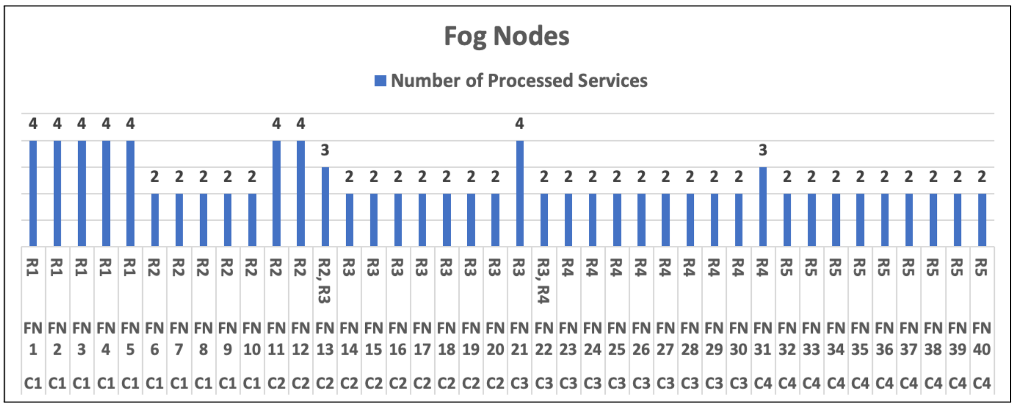


Figure 5.30: Experiment 15. Fog nodes

However, the fog nodes FN 13 and FN 22 handled two different service requirements to maximise the usage of nodes. For example, FN 13 with the capability of C2 that handled three services with service requirements of R2 and R3, in other words, S_1 (512 MB, 4 GB), S_2 (512 MB, 4 GB), S_3 (1024 MB, 8 GB) -> FN_{13} (2048 MB, 16 GB). There are 98 services have been distributed to forty fog nodes and the 22 services that are not processed distributed to the cloud nodes for processing.

Similarly, Figure 5.31 shows the 22 distributed services to the cloud nodes using the same strategy. In the cloud, 14 cloud nodes are used out of 20 nodes because the services did not require the remaining nodes as they are powerful in terms of processing power.

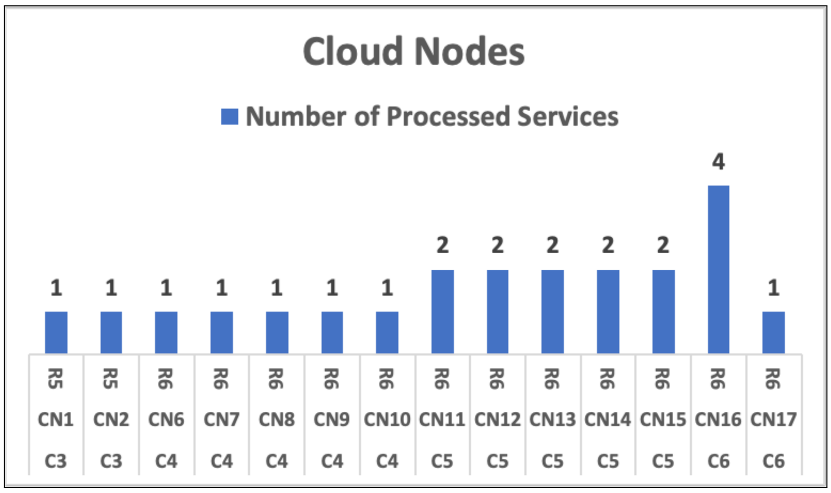


Figure 5.31: Experiment 15. Cloud nodes

Experiment 16: distributing the services with maximum technical requirements There are two bar charts namely MEIN Maximising – fog node and MEIN Maximising – cloud node. Both charts in Figure 5.32 and Figure 5.33 belong to the same strategy, but to show the distribution strategy process in both levels, the fog and cloud.

In this experiment, we tried to distribute the services with maximum service requirements to the nodes as much as possible in a way to use the capabilities of the nodes with the services that need maximum technical requirements while maintaining the usage of nodes. It is clear from the Figure 5.32 that fog nodes FN 1 to 10 have the capability of C1 = (1024 GB RAM and 8 GB storage) that processed the 10 services with technical requirement of R1 = (1024 GB RAM and 8 GB storage). In other words, every fog node can handle one service because every service requires technical requirement that matches the fog nodes' capabilities.

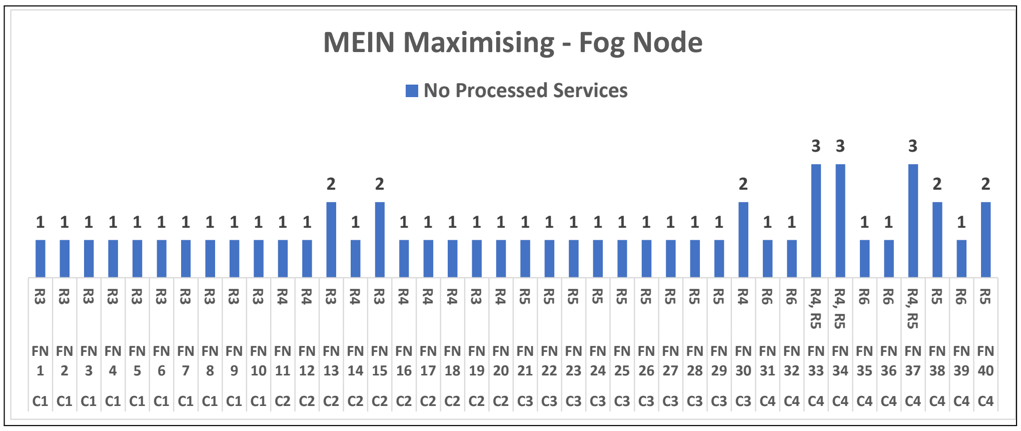


Figure 5.32: Experiment 16. Fog nodes

However, the fog nodes FN 33, FN 34 and FN 37 handled two different service requirements to fulfil the capabilities of nodes. For example, FN 33 with the capability of C4 that handled three services with service requirements of R4 and R5, in other words, S1 (2048 MB, 16 GB), S2 (2048 MB, 16 GB), S3 (4096 MB, 32 GB) $->$ FN 33 (8192 MB, 64 GB). There are 51 services have been distributed to forty fog nodes and the 69 services that are not process distributed to the cloud nodes for processing. Similarly, the Figure 5.33 shows the 69 distributed services to the cloud nodes using the same strategy. In the cloud, 20 cloud nodes are used out of 20 nodes even though the services did not require the remaining nodes as they are powerful in terms of processing power.

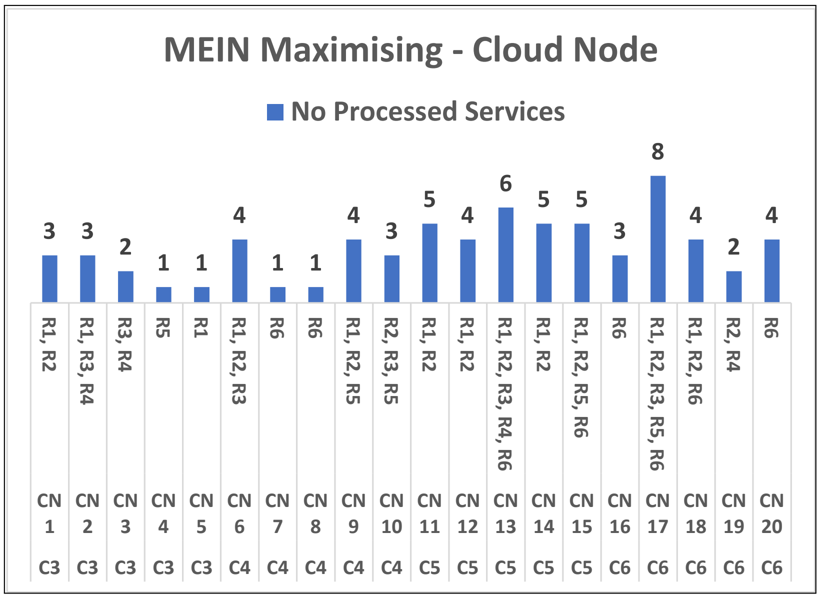


Figure 5.33: Experiment 16. Cloud nodes

It is clear from the results that the first strategy 5.30 and 5.31 could consume slightly more data communication over the network than the second strategy 5.32 and 5.33, as in the first, we minimised the wastage of the nodes usage by distributing the small service requirements to the IoT nodes. However, in the second, we distributed the services with the maximum service requirements that can fit into the IoT nodes, as a result the remaining services that are not distributed to the IoT nodes are distributed to the cloud with mostly low service requirements service as much as possible. In the first strategy: Minimising the wastage of node usage, as the goal of the strategy was to minimise the wastage of nodes usage, most of the services (98 services out of 120 services) with minimum service requirements are distributed to the IoT nodes. However, in the second strategy: distributing the services with maximum technical requirements, the aim was to distribute the services with highest service requirements which result that nearly half of the services (51 services out of 120 services) are distributed to the IoT nodes. It is clear that the IoT devices are limited with the power of processing and the number of devices when compared with the cloud computing, therefore there is need to optimise the usage of fog and IoT nodes to minimise the waste of resources and efficiently use of devices in the IoT environments. The results of the two experiments show that the resources are used effectively in the fog nodes. In the first strategy, most of the services are distributed to the IoT nodes while minimising the wastage of the usage of the node. Similarly, in the second strategy, nearly half of the services with maximum service requirements distributed to the IoT nodes while using most of the capabilities of the resources. On the other hand, the first experiment was good in terms of optimising the usage of the devices in the cloud where 14 out of 20 devices are used which means that only required number of devices are used, so they are minimised in terms of resources usage and number of devices. However, the second experiment was not good in terms of optimising the usage of nodes in the cloud as all the cloud nodes are used, but there was wastage of nodes capabilities. In addition, this experiment used all the nodes power which can effectively fast.

It is clear from Figure 5.34 that Experiment 4 (MEINAllC4) handled all services due to having powerful fog nodes with the capability of C4. However, Experiment 1 (MEINAllC1) handled half of the services which is considered as the worst number of services that are distributed to fog nodes due to having the lowest capabilities of fog nodes, but this combination of capabilities can be considered as the cheapest one among other combinations. Surprisingly, Experiments 3, 7, 9, 10, 15 have a small difference in the number of distributed services to fog nodes which might need extra attention to the cost of the nodes when selecting the most effective among them. Based on the results that the average number of distributed services to the fog nodes are between 90 and 100 services with the given combinations of fog nodes capabilities in the experiments.

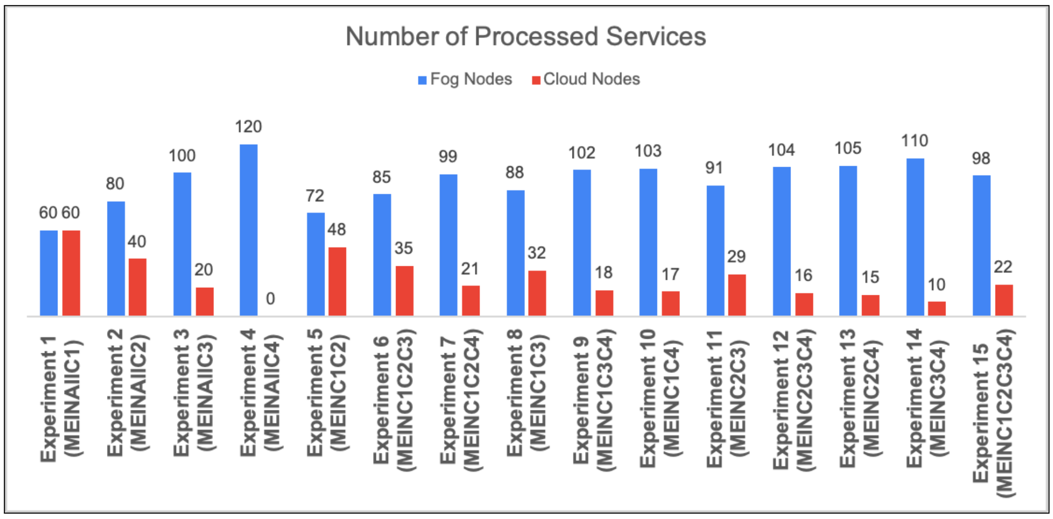
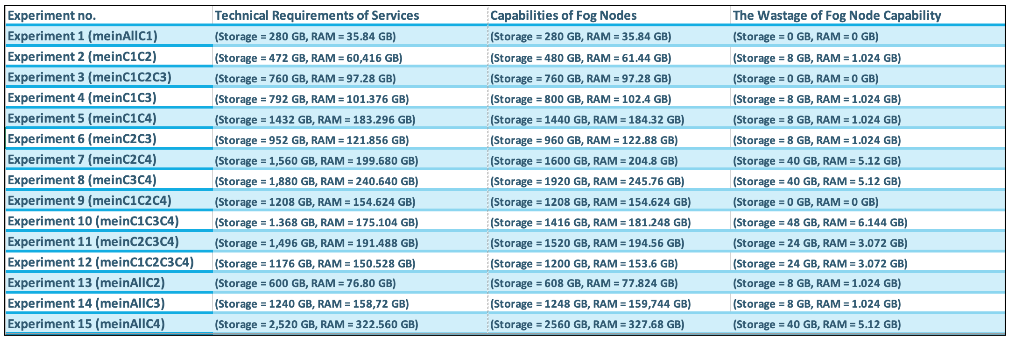


Figure 5.34: Number of processed services

Table 5.4 shows the usage of nodes after distributing the services to the fog nodes based on their capabilities. It can be seen that the usage of nodes capabilities is sensibly maximised. In particular the experiments 1, 3 and 9 do not show the usage of nodes capability. The capabilities of fog nodes have been calculated according of Table 5.1, 5.2 and 5.3.

Table 5.4: The usage of fog node capability



Experiment no.	Technical Requirements of Services	Capabilities of Fog Nodes	The Wastage of Fog Node Capability
Experiment 1 (meinAllC1)	(Storage = 280 GB, RAM = 35.84 GB)	(Storage = 280 GB, RAM = 35.84 GB)	(Storage = 0 GB, RAM = 0 GB)
Experiment 2 (meinC1C2)	(Storage = 472 GB, RAM = 60,416 GB)	(Storage = 480 GB, RAM = 61.44 GB)	(Storage = 8 GB, RAM = 1.024 GB)
Experiment 3 (meinC1C2C3)	(Storage = 760 GB, RAM = 97.28 GB)	(Storage = 760 GB, RAM = 97.28 GB)	(Storage = 0 GB, RAM = 0 GB)
Experiment 4 (meinC1C3)	(Storage = 792 GB, RAM = 101.376 GB)	(Storage = 800 GB, RAM = 102.4 GB)	(Storage = 8 GB, RAM = 1.024 GB)
Experiment 5 (meinC1C4)	(Storage = 1432 GB, RAM = 183.296 GB)	(Storage = 1440 GB, RAM = 184.32 GB)	(Storage = 8 GB, RAM = 1.024 GB)
Experiment 6 (meinC2C3)	(Storage = 952 GB, RAM = 121.856 GB)	(Storage = 960 GB, RAM = 122.88 GB)	(Storage = 8 GB, RAM = 1.024 GB)
Experiment 7 (meinC2C4)	(Storage = 1,560 GB, RAM = 199.680 GB)	(Storage = 1600 GB, RAM = 204.8 GB)	(Storage = 40 GB, RAM = 5.12 GB)
Experiment 8 (meinC3C4)	(Storage = 1,880 GB, RAM = 240.640 GB)	(Storage = 1920 GB, RAM = 245.76 GB)	(Storage = 40 GB, RAM = 5.12 GB)
Experiment 9 (meinC1C2C4)	(Storage = 1208 GB, RAM = 154.624 GB)	(Storage = 1208 GB, RAM = 154.624 GB)	(Storage = 0 GB, RAM = 0 GB)
Experiment 10 (meinC1C3C4)	(Storage = 1.368 GB, RAM = 175.104 GB)	(Storage = 1416 GB, RAM = 181.248 GB)	(Storage = 48 GB, RAM = 6.144 GB)
Experiment 11 (meinC2C3C4)	(Storage = 1,496 GB, RAM = 191.488 GB)	(Storage = 1520 GB, RAM = 194.56 GB)	(Storage = 24 GB, RAM = 3.072 GB)
Experiment 12 (meinC1C2C3C4)	(Storage = 1176 GB, RAM = 150.528 GB)	(Storage = 1200 GB, RAM = 153.6 GB)	(Storage = 24 GB, RAM = 3.072 GB)
Experiment 13 (meinAllC2)	(Storage = 600 GB, RAM = 76.80 GB)	(Storage = 608 GB, RAM = 77.824 GB)	(Storage = 8 GB, RAM = 1.024 GB)
Experiment 14 (meinAllC3)	(Storage = 1240 GB, RAM = 158,72 GB)	(Storage = 1248 GB, RAM = 159,744 GB)	(Storage = 8 GB, RAM = 1.024 GB)
Experiment 15 (meinAllC4)	(Storage = 2,520 GB, RAM = 322.560 GB)	(Storage = 2560 GB, RAM = 327.68 GB)	(Storage = 40 GB, RAM = 5.12 GB)

5.6 Summary

We presented two service distribution strategies in which services are distributed to both IoT nodes and cloud nodes and the resources are used in an optimised way. The first distribution strategy is based on the best-fit algorithm from the bin packing problem in order to maximise the usage of node and the second one is based on multiple knapsack problem to distribute a subset of the services that have maximum technical requirements. The results show that our distribution strategies are successful in terms of distributing the services to the nodes in the IoT environments in an optimised way and the data communication over the network is reduced by distributing the services as much as possible to the IoT nodes. Bin packing is one solution among solutions, with given context the results show that it is good optimization, but there are other optimization solutions. To solve this type of optimization problem, bin packing is the most obvious one to use as we are trying to, not necessarily to build a new optimization algorithm here, but actually, we wanted to solve a different problem as bin packing is to reduce the number of bins, but in best fits you reduce the number of bins, but the idea is to try to keep the smallest left over. So we tried to take it from that concept. So we want to use all the capabilities and to use the available resources or the available devices to the maximum. We used Java simulation approach to focus on the key things and on the criteria, we were interested in that we want to measure without being distracted by various other aspects because a lot of other aspects would be part of the simulation.. It possible to do the simulation with other tools like ifogsim [42] by just integrating our algorithms there and also adding some of the networking aspect which can be found in ifogsim in which we are not doing it as we are not modelling a wider network. Based on the objective function, we wanted to minimise data communication (min-data), maximise usage of nodes (max-node) and maximise number of distributed services to fog nodes (max-number). It is clear from the results that data communication is minimised, usage of node is maximised, also the number of distributed services to fog nodes is maximised.

Chapter 6

Evaluation and discussion

In this chapter, we will evaluate and discuss the results and experiments of Chapter 3, 4 and 5 as follows. The evaluation of the three chapters will discuss and evaluate the results by giving the most important observations. In addition, it will outline some of the limitations that will be presented in this chapter. In the following subsections, we will discuss and evaluate each chapter's (Chapter 3, 4 and 5) experiments' results.

6.1 The proposed hybrid architecture

The initial results of Chapter 3 show us that the proposed approach is good enough for the chosen dataset and analytical methods. It is clear from the results that data communication is efficient and provides significant gains.

Observation 1: Data communication over the network. It is accepted that when the size of data is large the data communication will be more expensive. The raw data was around 1 million rows which are equal to approximately 50 MB. However, after aggregating data into features by using data aggregation algorithms, the number of rows became 5418 rows and the size became 1.2 MB. This means that very significant savings to data transmission and storage can be made by early aggregation. This observation will gradually gain in importance as the number and quality of sensors increases rapidly and thus the rate and resolution at which data will be delivered grow quickly. By fusing the data locally before sending it to the cloud, we are not only reducing the data we are also determining which data is meaningful and only send that. This will reduce the energy consumption of fog and sensor devices which typically gain internet connectivity through 3, 4 or 5G, thus batteries in the devices will last longer.

Observation 2: Accuracy. Aggregated data leads to less accuracy in the results compared to working with raw data. All presented approaches are effected in the same way. Overall, the loss of accuracy is not drastic: the lowest bar is 75% with the highest being around 93% as shown in Figure 3.2. Clearly the used analysis method has an impact with trade-offs such as the used local processing power as well as methods optimised for this localised setting being factors that can influence the accuracy. The right balance will be in terms of privacy, accuracy, resource cost, energy consumption and data transmission will need to be identified and our future work will further this area.

Out of these experiments and observations we can conclude that one important aspect for future work is a development that combines the ideas of distributed data aggregation with analysis methods that can also be distributed effectively and be run in low power environments. The fact that they can operate on smaller data sets will help, but somehow the methods need to contain a core part based on global understanding.

Aside, for completeness we like to note that we initially attempted to conduct the experiments with a Raspberry Pi model b with 512MB RAM, however this was not capable of applying some of the Weka toolbox analysis algorithms because of the RAM constraint. Hence, we used a slightly more powerful version as reported above.

6.2 Selecting the most effective architecture

The three approaches Fog, Hybrid and Fog+Cloud approaches have similarities in most cases. Whereas, the main difference between the approaches lies in the location where the ML/data transformation takes place. In the hybrid approach, the data transformation processing is done in the fog node, and then the transformed data are transmitted from the fog to the cloud which then applies classification algorithms to the preprocessed data. However, in the fog approach, the data transformation and machine learning process have been done at the fog node itself. The hybrid approach has an advantage over the fog approach in that it is utilising the cloud's processing power for applying more sophisticated algorithms that require further processing power such as machine learning algorithms. Thus, this phase is important for minimising the processing time which has an effect on the total processing time. The results in Chapter 4 demonstrate that the proposed approach is perfect for the given datasets and algorithms. As it can be seen that the results show data communication over the network is effective and gives considerable gains.

Summary of results

Experiment 1: Numerical - 6 measurements

The results show that the highest percentage of accuracy achieved is in multilayer perceptron. In addition, both algorithms logistic regression and multilayer perceptron have important diversity between the two sides. Clearly, in the execution time, the cloud takes less time than the fog to perform ML algorithms because of its unlimited power. The total processing time graph has three calculations for every architecture including the ML algorithms' execution time, the data transformation's execution time and the time of data communication between the cloud and fog as shown in Figure 4.4. *Experiment* 1: Numerical - 6 Measurements. (*d*). This graph's results have been summarised in Table 6.1.

Experiment 2: Text data

The results show that the highest accuracy is obtained by the support vector machine. There are two bars visible in Figure 4.3 *Experiment* 2: Text data (*b*): one for the transformed data and the other for the raw data. It is obvious that in the fog only device, the processing happens locally, therefore there is no data communication cost over the network. However, in terms of data communication from fog to cloud, the cloud approach was the highest as all raw data are sent. On the other hand, in the hybrid approach, the data communication from fog to the cloud is lower than both cloud and fog+cloud approaches as only the transformed data are transmitted. This result emphasises that it is possible to reduce data communications by preprocessing the data early in the network. The results show us that the two algorithms (Support Vector Machine and Naive Bayes) have considerable diversity between the two sides. It is clear that the cloud takes less time than the fog to execute ML algorithms because of its unlimited power. The Total Processing Time graph has three calculations for every architecture including the ML algorithms' execution time, the data transformation's execution time and the time of data communication between the cloud and fog as shown in Figure 4.5 *Experiment* 2: Text Data (*d*). This graph's results have been summarised in Table 6.1.

Experiment 3: Image data

The results show that applying K-NN classifier on *extract_ color_ histogram* has a higher accuracy percentage than *Image_ to_ feature_ vector*. There are two bars visible in Figure 4.6. *Experiment* 3: Image data. (*b*): one for the transformed data and the other for the raw data. It is obvious that in the fog only device, the processing happens locally, therefore there is no data communication cost over the network. However, the cloud approach was the highest in terms of data communication from fog to cloud as all raw data are transmitted. On the other hand, the hybrid approach was lower than both cloud and fog+cloud approaches in terms of data communication from fog to the cloud as only transformed data are transmitted. This result emphasises that it is possible to reduce data communications by pre-processing the data earlier in the network. It is clear from the results that applying K-NN classifier *Image_ to_ feature_ vector* significantly takes more time to execute than *extract_ color_ histogram*. However, comparing between cloud and fog there is no big difference in execution as in data communication. Obviously, the cloud takes less time than the fog to execute classification algorithms due to its unlimited processing power as shown in Figure 4.6. *Experiment* 3: Image Data. (*c*). The Total Processing Time graph has three calculations for every architecture including the ML algorithms' execution time, the data transformation's execution time and the time of data communication between the cloud and fog as shown in Figure 4.6. *Experiment* 3: Image Data. (*d*). This graph's results have been summarised in Table 6.1.

The first observation: Data communication over the network. It is widely accepted that when we increase the data size, the data communication over the network will be higher and costly.

Experiment 1: Numerical data In this experiment, there were approximately 1 million rows of raw data that are collected from mobile phones which nearly equal to 50MB in terms of data size. On the contrary, after we applied data fusion algorithms on the raw data we extracted features, and the rows of data are reduced to 5418 rows which equal 1.2 MB data size.

Experiment 2: Text data In this experiment the raw data were around 20000 newsgroups file which equals to 22.4 MB. After extracting features, the size is decreased but not significantly when we used the method to convert text into a numerical feature vector. Maybe we do not have large savings, but we created features which will allow us to do more analysis.

Table 6.1: Total processing time of the three experiments

	ML	DT	DC	Total
Fog	In all experiments, ML is Processed in the Fog and processing time is much more than the cloud due to limited processing power.	In all experiments, DT is Conducted in the Fog and processing time is slightly higher than cloud.	In both numerical and image data, the time that transformed data takes from Fog to cloud is very low. However, in text data, the time is in the middle of the measured approaches.	In both numerical and image data, total processing time is in the middle of the measured approaches. However, in text data, the total processing time is the slowest.
Cloud	In all experiments, ML is Processed in the Cloud and processing time is much less than the Fog due to unlimited processing power.	In all experiments, DT is Conducted in the Cloud and processed quickly due to the unlimited power of the Cloud.	In all experiments, the time that all raw data takes from Fog to cloud is the highest as all raw data is being sent.	In both numerical and image data, the total processing time is the slowest of the measured approaches. However, in text data, the total processing time is in the middle.
Hybrid	The same as Cloud architecture.	The same as Fog architecture	The same as Fog architecture.	In all experiments, the total processing time is the fastest of the measured approaches.
Fog+Cloud	The same as Cloud architecture	In all experiments, 70% of the data is transformed in the Fog and the other 30% in the Cloud.	In all experiments, the time that transformed data (70%) and remaining raw data (30%) take from Fog to Cloud is lower than Cloud and higher than both fog and hybrid architectures.	In all experiments, the total processing time is in the middle of the measured approaches.

ML = Machine Learning, DT = Data Transformation, DC = Data Communication

Experiment 3: Image data In this experiment, the raw data were around 570 MB. However, after extracting features from images, the size became 170 MB. This shows that extracting the features from images in network level can help with significant savings.

According to the experiments, significant savings can be achieved in data communication over the network by applying data transformation earlier in the network. This observation will be more important when the quality and number of sensors rise significantly and therefore the resolution and rate of data will be growing quickly. By applying data fusion to the data in the fog/edge node near to the data source before they are sent to the cloud, we will reduce the data and send only meaningful data. By doing this, the energy consumption in fog devices will be reduced, these devices obtain their internet connection through networks like 3G, 4G or even 5G, therefore the devices' batteries will be lasting longer too.

The second observation: Accuracy of each algorithm. In the results, transformed data can be seen as less accurate than working with the raw data.

Experiment 1: Numerical data Overall, the accuracy loss is between 7 - 25% which is not excessive: the lowest level of accuracy is 75%, however, the highest level accuracy is approximately 93% as shown in Figure 4.4.

Experiment 2: Text data Overall, the accuracy loss is between 17 - 25% which is not extreme: the lowest level of accuracy is 77.3%, however, the highest level accuracy is approximately 82.3% as shown in Figure 4.5.

Experiment 3: Image Data Overall, the accuracy loss is around 40% which is greater than both previous experiments, but not significant: the lowest level of accuracy is 54.9%, however, the highest level accuracy is approximately 57.34% as shown in Figure 4.6.

Clearly, the analytical algorithm that is used as an influence with trade-offs. For example, the algorithms are optimised for this localised setting as well as the used local processing power that can impact on the accuracy. The correct balance in terms of data transmission, privacy, energy consumption, accuracy and resource cost will need to be identified and our future work will further this area. The results show that the traditional architecture which is based on sending all the data from data source to a single server point has limitations. This is particularly evident when using large data volume with restrictions on time. The experiments show how data communication over the network can be very expensive while sending large data volume without applying any transformation in advance. Additionally, it shows how effective our hybrid approach is in this regard.

6.3 The service distribution strategy

The initial results of Chapter 5 indicate that the proposed distribution strategy is good enough for the given setup and configuration. It is clear from the results that services are distributed in an efficient way and the usage of nodes is maximised.

Observation 1: Number of distributed services to the fog

As we mentioned earlier that our primary intention is that distributing the services as much as possible nearer to the data while maintaining the optimisation of resource usage, data communication of the network and balancing the service distribution in the IoT system. The number of distributed services depend on several aspects including the strategy, capabilities and service requirements.

We have conducted 15 experiments to explore the most effective combination of fog nodes capabilities that can handle as much as possible services. As expected having more powerful fog nodes can help us to distribute more services to fog nodes as shown in Figure 5.34. However, if we have powerful fog nodes, then it will be costly for designing IoT systems which are against the characteristics of IoT. In our experiments, the goal of the strategy was to maximise the usage of nodes, most of the services with minimum service requirements are distributed to the IoT nodes.

Observation 2: resources usage

The IoT devices are limited with the power of processing and the number of devices when compared with the cloud computing, therefore there is a need to optimise the usage of fog and IoT nodes to maximise the usage of resources and efficient use of devices in the IoT environments. The results of the two experiments show that the resources are used effectively in the fog nodes. In our experiments, most of the services are distributed to the IoT nodes while maximising the usage of nodes.

In addition, the experiments were good in terms of optimising the usage of the devices in the cloud which means that only a required number of devices are used, so they are minimised in terms of resource usage and a number of devices.

Observation 3: data communication over the network

It is clear from the results that some of the experiments could consume more data communication over the network, as in our experiments, we maximised the usage of nodes by distributing the small service requirements to the fog nodes and remaining services that are not distributed to the fog nodes are distributed to the cloud with mostly high service requirements which means high data communication over the network. However, we can address the issue of high communication over the network by selecting the powerful combination of fog nodes to distribute most or all the services to the fog nodes which can eliminate the communication over the network and reduce the cost of data communication.

The combinations of fog nodes capabilities that are used in our experiments can help us when selecting a combination which depends on several factors such as technical specification of fog nodes including the RAM and storage and the technical requirements of services. We can use the distribution strategy based on the services requirements and the capabilities of the nodes and select which one fits on service distribution either to fog nodes or cloud nodes.

6.4 Threats to validity

In this subsection, we identify the main threats to the validity of our research project including the scenarios, experimental setup and experiments.

Scenario. We have introduced a motivating scenario that is simpler than a whole IoT system which is not exactly the same as an IoT scenario, but this small scenario is showing the problem and it is not very complicated because it should be easy to read and understandable, but when applying to an IoT system it will be more sophisticated.

Experimental setup of the experiments. The sample size that we used in our experiments might not be big enough as the real IoT system as the amount of internet-connected devices has been increasing. It is not possible to mimic the devices in the IoT system due to the lack of resources. We addressed this threat by conducting a number of experiments using simulation.

Experiment. We did some real experiments, but we could not make all experiments in real setup due to the lack of resources and the cost of having all elements of the IoT system. We addressed this threat by using simulated experiments as it was not easy to apply the real experiments as mentioned earlier.

6.5 Experimental approach and generalisability

The experimental approach allows us to use tools and to show things. But, it is difficult to generalise. However, in our case there is enough generality, we have use datasets, butt here isn't anything in our approach that is very particularly geared to these datasets. So if we had access to a different datasets, then our approach would probably still be fine. So in our approach, we do not have a specific shape or structure that the data should be in to work with our approach. We have used our experiments on some randomly chosen datasets, and if we had different data we probably would have come to exactly the same insights.

6.6 Summary

This chapter has presented the evaluation of Chapter 3 (The proposed architecture), Chapter 4 (Selecting the most effective architecture) and Chapter 5 (The service distribution strategy). The most important observations from the results of experiments have been discussed. In the observations, we focused on several metrics namely data communication over the network, resource usage, number of distributed services, the accuracy of machine learning algorithms. In addition, threats to the validity of our research are identified.

Chapter 7

Conclusion and future work

In conclusion, this thesis presented an efficient data analytics architecture for the Internet of Things based on fog computing. Overall research challenges were as follows; (1) Processing the data in the IoT efficiently while maintaining the load over the network, (2) selecting the most effective data analytics architecture in the context of IoT and finding efficient strategies while processing the data over the network, (3) exploring the best place in the network for services and then distributing them to the right nodes based on their capabilities while optimising the usage of resources.

In this chapter, we will present the contributions of our research by addressing the stated research challenges. Then, we will discuss several possibilities of future directions.

7.1 Research contributions

7.1.1 An efficient architecture for the IoT

We presented an efficient approach for data analytics architecture where data fusion and data pre-processing have been done near to the source of data particularly in the fog and machine learning algorithms are done in the cloud to reduce data communication over the network. Therefore, the raw data will not be transmitted instead more meaningful data will be transmitted over the network to the cloud for further analysis while maintaining the latency. The results show that our architecture was good with the given setup and datasets in terms of reducing the data traffic over the network while maintaining the accuracy level of later decision making. We presented the proposed approach and its relevant methods. In addition, we used the WISDM dataset [55] to validate our architecture.

7.1.2 Exploring the most efficient architecture for the IoT

We explored four different architectures namely cloud, fog, hybrid and fog + cloud to find the most effective architecture and we presented an efficient strategy where the data processing happened near to the source of data especially in the fog before being transmitted to the cloud to minimise the communication of data over the network. In our experiments, we have used three types of datasets namely numerical data, text data and image data. Besides, we conducted 4 experiments for each dataset to explore which architecture is the most efficient for the IoT. The results indicate that our proposed architecture which is called the hybrid architecture is successful in terms of reducing data traffic over the network without significantly reducing the accuracy of later decision making. The datasets that are used in the experiments are WISDM dataset [55], 20 Newsgroups dataset [56] and Kaggle dogs vs cats dataset [49] to validate our architecture.

7.1.3 Distributing the services in the IoT

We presented a service distribution strategy where the IoT services are distributed to fog nodes and cloud nodes based on their capabilities. In the strategy, we have used the bin packing algorithm particularly best-fit algorithm as a baseline. We have used a distribution strategy to maximise the usage of nodes. We have created 15 combinations of fog nodes capabilities and conducted an experiment for each of the combinations. The results of 15 experiments showed that the distribution strategy is good in terms of distributing the services to the right nodes in an optimised way. We presented the proposed distribution strategy and its relevant methods and algorithms. In addition, we have used java programming to simulate the distribution strategy to evaluate the proposed strategy.

7.2 Future work

We present below several research challenges that we believe to overcome for practical realisations of fog computing in the IoT domain. Future work will concentrate on these challenges

7.2.1 Provisioning

The first challenge is the provisioning of edge nodes for executing workloads that are offloaded by other fog nodes [115] or from cloud servers [102]. The reduced hardware and processing configurations, heterogeneity of available resources across the range of possible edge nodes and the "lack of standard protocols for initialising services on a potential edge node" [102] add to the complexity. However, harnessing the capabilities of the diverse resources can contribute to extending the boundaries of a cloud system and provide additional revenue models for network providers, by offering incremental data processing as it moves from the source to its destination [112].

Moreover, matching service execution requirements to the fog nodes' available configurations is also key, necessitating composition or decomposition. For example, consider a service which needs a device configuration of a quad-core processor, 2 GB RAM and 4 GB Storage. Executing this service on most fog nodes is not possible due to their limited processing power. Therefore, there is a need to decompose this service into smaller services in a systematic way, which comes with its own challenges. Similarly, for service composition, it is not possible to provide a composition of two services with high resource demands on fog nodes.

7.2.2 Resource management

The second challenge is resource management in the fog nodes. Due to their limited computation power, distributing the workloads to edge nodes is challenging because this needs to be done dynamically with the given limited configuration of resources. Therefore, managing resources like battery consumption, CPU usage, RAM usage, storage usage and bandwidth are difficult in an environment that changes dynamically and unexpectedly which makes the process of resource allocation difficult as well.

7.2.3 Describing nodes

The third challenge is discovering and describing node capabilities, made especially difficult due to the heterogeneous and volatile nature of the IoT, making it difficult to capture and model data about offered services which have to be done dynamically. While there exist some efforts for a standardised terminology for constrained devices, such as the RFC 7228 [24], there is a need to describe their capabilities dynamically which is challenging and impossible to do manually. Analogously, there is a need to capture and model the data about the services (e.g. RAM usage, CPU usage and others) to orchestrate and allocate them to the right nodes.

7.2.4 Decomposition and composition

The fourth challenge is decomposition; as we have shown in this thesis, decomposition plays an important role in increasing the quality of service, but decomposing the IoT services dynamically and in an automated way is challenging in fog nodes due to the resource constraints and the dynamic of the IoT. In addition, decomposing the services into linked-microservices is challenging because it is difficult to create services that are distributed to different nodes and linked to their linked-partners. Similarly, in the composition process, this will be challenging because identifying linked-microservices which are distributed across the network and then composing them is a sophisticated process in an IoT environment. Additionally, this has to be achieved without affecting the quality of service and data, while using constrained devices.

7.2.5 Fog node storage

The fog nodes have limited storage capability than the cloud nodes, there is a need to give attention to the duration of the data that will be stored in the edge of the network in the fog node before deleting or sending it to the cloud. There is a need to find a strategy to tackle storage limitations in an efficient and automated way as there is unlimited data stream flow.

7.2.6 Energy consumption

Both fog devices and IoT devices are limited with computational power and storage and they are mostly battery powered which means the energy usage should be used efficiently. In our work, we focused on reducing the communication over the network and using the processing power to process the data, but we did not focus on the battery usage aspect. Therefore, there is a need to have trade-off analysis between the usage of network and usage of battery when processing the services to maintain the usage of both the network and battery.

7.2.7 Hierarchical fog nodes

In hierarchical fog node overlays, data processing will be done in several stages. This will be helpful when moving the computation away from the centralised cloud. If we hypothesise that there are two levels of fog in the network, then we can process some parts of the data in the first level and the other parts can be processed in the second level which will eliminate the necessity using the cloud for further processing and giving fast repose to time-related applications.

Bibliography

[1] ipokemon. https://github.com/Kjuly/iPokeMon.

[2] Wikipedia dataset. https://dumps.wikimedia.org/enwiki/latest/.

[3] Ericsson mobility report. https://www.ericsson.com/assets/local/mobility-report/documents/2015/ericsson-mobility-report-nov-2015.pdf, 2015.

[4] Fog computing and the internet of things: Extend the cloud to where the things are. *CISCO White Paper*, 2015.

[5] What is fog computing? https://www.ibm.com/blogs/cloud-computing/2014/08/25/fog-computing/, Sep 2016.

[6] Nips: bag of words data set. https://archive.ics.uci.edu/ml/datasets/bag+of+words, 2018.

[7] Cisco annual internet report - cisco annual internet report (2018–2023) white paper. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html, Mar 2020.

[8] M. Aazam and E.-N. Huh. Fog computing and smart gateway based

communication for cloud of things. In *2014 International Conference on Future Internet of Things and Cloud*, pages 464–470. IEEE, 2014.

[9] M. Aazam, S. Zeadally, and K. A. Harras. Fog computing architecture, evaluation, and future research directions. *IEEE Communications Magazine*, 56(5):46–52, May 2018.

[10] A. Abdelgawad and M. Bayoumi. Data fusion in wsn. In *Resource-aware data fusion algorithms for wireless sensor networks*, pages 17–35. Springer, 2012.

[11] A. Abdelgawad and M. Bayoumi. *Resource-Aware data fusion algorithms for wireless sensor networks*, volume 118. Springer Science & Business Media, 2012.

[12] M. Abu-Elkheir, M. Hayajneh, and N. A. Ali. Data management for the internet of things: Design primitives and solution. *Sensors*, 13(11):15582–15612, 2013.

[13] M. Ahmad, M. B. Amin, S. Hussain, B. H. Kang, T. Cheong, and S. Lee. Health fog: A novel framework for health and wellness applications. *The Journal of Supercomputing*, 72(10):3677–3695, 2016.

[14] Y. Ai, M. Peng, and K. Zhang. Edge computing technologies for internet of things: a primer. *Digital Communications and Networks*, 4(2):77–86, 2018.

[15] C. Akasiadis, G. Tzortzis, E. Spyrou, and C. Spyropoulos. Developing complex services in an iot ecosystem. In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, pages 52–56, Dec 2015.

[16] B. Alturki, S. Reiff-Marganiec, and C. Perera. A hybrid approach for data analytics for the internet of things. In *Proceedings of the Seventh International Conference on the Internet of Things*, page 7. ACM, 2017.

[17] B. Alturki, S. Reiff-Marganiec, C. Perera, and S. De. Exploring the effectiveness of service decomposition in fog computing architecture for the internet of things. *IEEE Transactions on Sustainable Computing*, pages 1–1, 2019.

[18] H. C. M. Andrade, B. Gedik, and D. S. Turaga. *Fundamentals of Stream Processing: Application Design, Systems, and Analytics*. Cambridge University Press, 2014.

[19] M. Antonini, M. Vecchio, F. Antonelli, P. Ducange, and C. Perera. Smart audio sensors in the internet of things edge for anomaly detection. *IEEE Access*, 6:67594–67610, 2018.

[20] H. R. Arkian, R. E. Atani, A. Pourkhalili, and S. Kamali. Cluster-based traffic information generalization in vehicular ad-hoc networks. *Vehicular communications*, 1(4):197–207, 2014.

[21] D. Athanasopoulos, A. V. Zarras, G. Miskos, V. Issarny, and P. Vassiliadis. Cohesion-driven decomposition of service interfaces without access to source code. *IEEE Transactions on Services Computing*, 8(4):550–562, 2015.

[22] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu. Fog computing: A platform for internet of things and analytics. In *Big Data and*

Internet of Things: A Roadmap for Smart Environments, pages 169–186. Springer, 2014.

[23] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16, 2012.

[24] C. Bormann, M. Ersue, and A. Keranen. Terminology for constrained-node networks. RFC 7228, RFC Editor, May 2014. http://www.rfc-editor.org/rfc/rfc7228.txt.

[25] C. Bormann, M. Ersue, and A. Keranen. Terminology for constrained-node networks. Technical report, 2014.

[26] A. Canedo. Industrial iot lifecycle via digital twins. In *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, page 29. ACM, 2016.

[27] F. Castanedo. A review of data fusion techniques. *The Scientific World Journal*, 2013, 2013.

[28] F. Chen, C. Ren, J. Dong, Q. Wang, J. Li, and B. Shao. A comprehensive device collaboration model for integrating devices with web services under internet of things. In *2011 IEEE International Conference on Web Services*, pages 742–743, July 2011.

[29] N. Chen, N. Cardozo, and S. Clarke. Goal-driven service composition in mobile and pervasive computing. *IEEE Transactions on Services Computing*, 11(1):49–62, Jan 2018.

[30] S. Chhabra and D. Singh. Data fusion and data aggregation/summarization techniques in wsns: A review. *International Journal of Computer Applications*, 121(19), 2015.

[31] M. Chiang and T. Zhang. Fog and iot: An overview of research opportunities. *IEEE Internet of Things Journal*, 3(6):854–864, 2016.

[32] O. Consortium et al. Openfog reference architecture for fog computing. *Architecture Working Group*, pages 1–162, 2017.

[33] L. Coyne, J. Dain, E. Forestier, P. Guaitani, R. Haas, C. D. Maestas, A. Maille, T. Pearson, B. Sherman, C. Vollmar, et al. *IBM private, public, and hybrid cloud storage solutions.* IBM Redbooks, 2018.

[34] B. V. Dasarathy. Sensor fusion potential exploitation-innovative architectures and illustrative applications. *Proceedings of the IEEE*, 85(1):24–38, 1997.

[35] F. C. Delicato, P. F. Pires, and T. V. Batista. *Middleware Solutions for the Internet of Things.* Springer Briefs in Computer Science. Springer, 2013.

[36] H. R. Dhasian and P. Balasubramanian. Survey of data aggregation techniques using soft computing in wireless sensor networks. *IET Information Security*, 7(4):336–342, 2013.

[37] H. F. Durrant-Whyte. Sensor models and multisensor integration. *The international journal of robotics research*, 7(6):97–113, 1988.

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

[39] J. García and C. Maureira. A knn quantum cuckoo search algorithm applied to the multidimensional knapsack problem. *Applied Soft Computing*, 102:107077, 2021.

[40] M. R. Garey and D. S. Johnson. *Computers and intractability*, volume 174. freeman San Francisco, 1979.

[41] R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal*, 45(9):1563–1581, 1966.

[42] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya. ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

[43] K. Habak, C. Shi, E. W. Zegura, K. A. Harras, and M. Ammar. *Elastic Mobile Device Clouds*, chapter 7, pages 159–188. John Wiley Sons, Ltd, 2017.

[44] D. L. Hall and J. Llinas. An introduction to multisensor data fusion. *Proceedings of the IEEE*, 85(1):6–23, 1997.

[45] P. Harrington. *Machine learning in action*, volume 5. Manning Greenwich, CT, 2012.

[46] M. E. P. Hernández and S. Reiff-Marganiec. Towards a software framework for the autonomous internet of things. In *2016 IEEE 4th international conference on future internet of things and cloud (FiCloud)*, pages 220–227. IEEE, 2016.

[47] P. Hu, H. Ning, T. Qiu, Y. Zhang, and X. Luo. Fog computing-based

face identification and resolution scheme in internet of things. *IEEE Transactions on Industrial Informatics*, 2017.

[48] M. Hung. Leading the iot, gartner insights on how to lead in a connected world. *Gartner Research*, pages 1–29, 2017.

[49] Kaggle. Dogs vs. cats dataset. https://www.kaggle.com/c/dogs-vs-cats.

[50] D. Kaur, G. S. Aujla, N. Kumar, A. Y. Zomaya, C. Perera, and R. Ranjan. Tensor-based big data management scheme for dimensionality reduction problem in smart grid systems: Sdn perspective. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1985–1998, Oct 2018.

[51] C. Kenyon et al. Best-fit bin-packing with random order. In *SODA*, volume 96, pages 359–364, 1996.

[52] M. E. Khanouche, Y. Amirat, A. Chibani, M. Kerkar, and A. Yachir. Energy-centered and qos-aware services selection for internet of things. *IEEE Transactions on Automation Science and Engineering*, 13(3):1256–1269, 2016.

[53] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24, 2007.

[54] N. Kumar, J. J. P. C. Rodrigues, M. Guizani, K. R. Choo, R. Lu, C. Verikoukis, and Z. Zhong. Achieving energy efficiency and sustainability in edge/fog deployment. *IEEE Communications Magazine*, 56(5):20–21, May 2018.

[55] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

[56] K. Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339, 1995.

[57] R. Mahmud, K. Ramamohanarao, and R. Buyya. Latency-aware application module management for fog computing environments. 2017.

[58] J. Manyika. *The Internet of Things: Mapping the value beyond the hype.* McKinsey Global Institute, 2015.

[59] Y. Min, Y. Y. Htay, and K. K. Oo. Comparing the performance of machine learning algorithms for human activities recognition using wisdm dataset. *International Journal of Computer (IJC)*, 38(1):61–72, 2020.

[60] H. B. Mitchell. *Multi-sensor data fusion: an introduction.* Springer Science & Business Media, 2007.

[61] N. H. Motlagh, M. Bagaa, and T. Taleb. Uav-based iot platform: A crowd surveillance use case. *IEEE Communications Magazine*, 55(2):128–134, February 2017.

[62] N. H. Motlagh, T. Taleb, and O. Arouk. Low-altitude unmanned aerial vehicles-based internet of things services: Comprehensive survey and future perspectives. *IEEE Internet of Things Journal*, 3(6):899–922, Dec 2016.

[63] M. Mukherjee, L. Shu, and D. Wang. Survey of fog computing: Fundamental, network applications, and research challenges. *IEEE Communications Surveys & Tutorials*, 20(3):1826–1857, 2018.

[64] A. Munir, P. Kansakar, and S. U. Khan. Ifciot: Integrated fog cloud iot: A novel architectural paradigm for the future internet of things. *IEEE Consumer Electronics Magazine*, 6(3):74–82, 2017.

[65] M. Muntjir, M. Rahul, and H. A. Alhumyani. An analysis of internet of things (iot): novel architectures, modern applications, security aspects and future scope with latest case studies. *Int. J. Eng. Res. Technol*, 6(6):422–447, 2017.

[66] S. Newman. *Building microservices: designing fine-grained systems.* " O'Reilly Media, Inc.", 2015.

[67] I. Ng. Engineering a market for personal data: The hub-of-all-things (hat), a briefing paper. *WMG Service Systems Research Group Working Paper Series*, 2014.

[68] L. Ni, J. Zhang, C. Jiang, C. Yan, and K. Yu. Resource allocation strategy in fog computing based on priced timed petri nets. *IEEE Internet of Things Journal*, 2017.

[69] K. K. Oo. Daily human activity recognition using adaboost classifiers on wisdm dataset. 2019.

[70] C. Perera, C. H. Liu, and S. Jayawardena. The emerging internet of things marketplace from an industrial perspective: A survey. *IEEE Transactions on Emerging Topics in Computing*, 3(4):585–598, Dec 2015.

[71] C. Perera, C. H. Liu, S. Jayawardena, and M. Chen. A survey on internet of things from industrial market perspective. *IEEE Access*, 2:1660–1679, 2014.

[72] C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos. Fog computing for sustainable smart cities: A survey. *ACM Computing Surveys (CSUR)*, 50(3):32, 2017.

[73] C. Perera, S. Y. L. Wakenshaw, T. Baarslag, H. Haddadi, A. K. Bandara, R. Mortier, A. Crabtree, I. C. L. Ng, D. McAuley, and J. Crowcroft. Valorising the iot databox: creating value for everyone. *Transactions on Emerging Telecommunications Technologies*, 28(1):e3125, 2017. e3125 ett.3125.

[74] C. Pfister. *Getting started with the Internet of Things.* O'Reilly, 1 edition, 2011.

[75] J. Preden, J. Kaugerand, E. Suurjaak, S. Astapov, L. Motus, and R. Pahtma. Data to decision: pushing situational information needs to the edge of the network. In *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2015 IEEE International Inter-Disciplinary Conference on*, pages 158–164. IEEE, 2015.

[76] S. Pyne, B. P. Rao, and S. Rao. *Big Data Analytics.* Springer India, 1 edition, 2016.

[77] M. Pérez-Hernández, B. Alturki, and S. Reiff-Marganiec. Fabiot: A flexible agent-based simulation model for iot environments. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE*

Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pages 66–73, 2018.

[78] C. M. Rahman and T. A. Rashid. A new evolutionary algorithm: Learner performance based behavior algorithm. *Egyptian Informatics Journal*, 22(2):213–223, 2021.

[79] M. A. Rahman, M. S. Hossain, E. Hassanain, and G. Muhammad. Semantic multimedia fog computing and iot environment: Sustainability perspective. *IEEE Communications Magazine*, 56(5):80–87, May 2018.

[80] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke. Middleware for Internet of Things: a Survey. *Internet of Things Journal, IEEE*, PP(99):1, 2015.

[81] S. Rodríguez-Valenzuela, J. Holgado-Terriza, J. L. Muros-Cobos, and J. M. Gutiérrez-Guerrero. Data fusion mechanism based on a service composition model for the internet of things. *Actas de las III Jornadas de Computación Empotrada (JCE), Septiembre*, pages 19–21, 2012.

[82] D. G. Roy, D. De, A. Mukherjee, and R. Buyya. Application-aware cloudlet selection for computation offloading in multi-cloudlet environment. *The Journal of Supercomputing*, 73(4):1672–1690, 2017.

[83] M. Satyanarayanan. The emergence of edge computing. *Computer*, 50(1):30–39, 2017.

[84] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos. Edge analytics in the internet of things. *IEEE Pervasive Computing*, 14(2):24–31, 2015.

[85] S. SECTOR and O. ITU. Series y: Global information infrastructure, internet protocol aspects and next-generation networks.

[86] S. Servia-Rodriguez, L. Wang, J. R. Zhao, R. Mortier, and H. Haddadi. Personal model training under privacy constraints. *arXiv preprint arXiv:1703.00380*, 2017.

[87] D. Shadija, M. Rezai, and R. Hill. Microservices: granularity vs. performance. In *Companion Proceedings of the10th International Conference on Utility and Cloud Computing-UCC'17 Companion*, pages 215–220. ACM Press, 2017.

[88] Y. Shi, G. Ding, H. Wang, H. E. Roman, and S. Lu. The fog computing service for healthcare. In *2015 2nd International Symposium on Future Information and Communication Technologies for Ubiquitous HealthCare (Ubi-HealthTech)*, pages 1–5. IEEE, 2015.

[89] S. M. Soma Bandyopadhyay, Munmun Sengupta and S. Dutta. ROLE OF MIDDLEWARE FOR INTERNET OF THINGS: A STUDY. In *Proceedings - IEEE International Conference on Distributed Computing in Sensor Systems, DCOSS 2015*, pages 230–235, 2015.

[90] P. Stelmach. Service composition scenarios in the internet of things paradigm. In *Doctoral Conference on Computing, Electrical and Industrial Systems*, pages 53–60. Springer, 2013.

[91] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelfflé. Vision and challenges for realising the internet of things. *Cluster of European Research Projects on the Internet of Things, European Commision*, 2010.

[92] C. Systems. Fog computing and the internet of things: Extend the cloud to where the things are. 2015.

[93] B. Tang, Z. Chen, G. Hefferman, S. Pei, W. Tao, H. He, and Q. Yang. Incorporating intelligence in fog computing for big data analysis in smart cities. *IEEE Transactions on Industrial Informatics*, 2017.

[94] H. Tschofenig, J. Arkko, D. Thaler, and D. McPherson. Architectural considerations in smart object networking. *RFC 7452*, 2015.

[95] L. M. Vaquero and L. Rodero-Merino. Finding your way in the fog: Towards a comprehensive definition of fog computing. *ACM SIGCOMM Computer Communication Review*, 44(5):27–32, 2014.

[96] P. Vashisht and V. Gupta. Big data analytics techniques: A survey. In *Green Computing and Internet of Things (ICGCIoT), 2015 International Conference on*, pages 264–269. IEEE, 2015.

[97] M. Verma, N. Bhardwaj, and A. K. Yadav. Real time efficient scheduling algorithm for load balancing in fog computing environment. *Int. J. Inf. Technol. Comput. Sci*, 8(4):1–10, 2016.

[98] O. Vermesan, P. Friess, P. Guillemin, S. Gusmeroli, H. Sundmaeker, A. Bassi, I. S. Jubert, M. Mazura, M. Harrison, M. Eisenhauer, P. Doody, F. Peter, G. Patrick, G. Sergio, B. Harald, Sundmaeker Alessandro, J. Ignacio Soler, M. Margaretha, H. Mark, E. Markus, and D. Pat. Internet of Things Strategic Research Roadmap. *Internet of Things Strategic Research Roadmap*, pages 9–52, 2009.

[99] H. Wang, S. Zhou, and Q. Yu. Discovering web services to improve

requirements decomposition. In *2015 IEEE International Conference on Web Services*, pages 743–746, June 2015.

[100] M. Wang, C. Perera, P. P. Jayaraman, M. Zhang, P. Strazdins, and R. Ranjan. City data fusion: Sensor data fusion in the internet of things. *arXiv preprint arXiv:1506.09118*, 2015.

[101] M. Wang, C. Perera, P. P. Jayaraman, M. Zhang, P. Strazdins, R. Shyamsundar, and R. Ranjan. City data fusion: Sensor data fusion in the internet of things. *Int. J. Distrib. Syst. Technol.*, 7(1):15–36, Jan. 2016.

[102] N. Wang, B. Varghese, M. Matthaiou, and D. S. Nikolopoulos. Enorm: A framework for edge node resource management. *IEEE Transactions on Services Computing*, 2017.

[103] P. Wang, Z. Ding, C. Jiang, and M. Zhou. Constraint-aware approach to web service composition. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 44(6):770–784, 2014.

[104] G. M. Weiss. Wisdm smartphone and smartwatch activity and biometrics dataset. *UCI Machine Learning Repository: WISDM Smartphone and Smartwatch Activity and Biometrics Dataset Data Set*, 2019.

[105] F. E. White. Data fusion lexicon. Technical report, DTIC Document, 1991.

[106] J. Woolsey. Powering the next generation cloud with azure stack. *Nano Server & Windows Server*, 2016.

[107] F. Xhafa and L. Barolli. Semantics, intelligent processing and services for big data, 2014.

[108] J. Xu, K. Ota, and M. Dong. Saving energy on the edge: In-memory caching for multi-tier heterogeneous networks. *IEEE Communications Magazine*, 56(5):102–107, May 2018.

[109] Z. Yan, J. Liu, A. V. Vasilakos, and L. T. Yang. Trustworthy data fusion and mining in internet of things. *Future Generation Computer Systems*, 49(C):45–46, 2015.

[110] S. Yi, Z. Hao, Z. Qin, and Q. Li. Fog computing: Platform and applications. In *Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on*, pages 73–78. IEEE, 2015.

[111] S. Yi, C. Li, and Q. Li. A survey of fog computing: concepts, applications and issues. In *Proceedings of the 2015 workshop on mobile big data*, pages 37–42. ACM, 2015.

[112] A. R. Zamani, M. Zou, J. Diaz-Montes, I. Petri, O. Rana, A. Anjum, and M. Parashar. Deadline constrained video analysis via in-transit computational environments. *IEEE Transactions on Services Computing*, 2017.

[113] A. Zaslavsky, C. Perera, and D. Georgakopoulos. Sensing as a service and big data. *arXiv preprint arXiv:1301.0159*, 2013.

[114] A. B. Zaslavsky, C. Perera, and D. Georgakopoulos. Sensing as a service and big data. *CoRR*, abs/1301.0159, 2013.

[115] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, S. N. Srirama, and R. Buyya. mcloud: A context-aware offloading framework for heterogeneous mobile cloud. *IEEE Transactions on Services Computing*, 10(5):797–810, Sept 2017.

[116] Y. Zhou, S. De, W. Wang, K. Moessner, and M. S. Palaniswami. Spatial indexing for data searching in mobile sensing environments. *Sensors*, 17(6), 2017.