BASE STATION CLUSTERING AND MOBILE DATA TRAFFIC PREDICTION IN CLOUD RADIO ACCESS NETWORK FOR MULTIPLEXING GAIN

BY

UMASABOR, Nelson
MEng/SEET/2017/7200

A THESIS SUBMITTED TO THE POSTGRADUATE SCHOOL, FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA, IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF MASTER OF ENGINEERING IN COMMUNICATION ENGINEERING

OCTOBER, 2021

ABSTRACT

Mobile network operators over-dimension network resources due to lack of pre-knowledge of the data traffic volume and this leads to underutilization of the network resources as usage of base station resources vary throughout the day. This over-dimensioning also leads to increased cost. Cloud Radio Access Network (C-RAN) allows for network resources to be shared amongst several base stations thereby reducing cost. Previous works have used clustering to group base stations with similar network load and others have clustered base stations with light and heavy network load together, but this does not capture the true network resource requirement due to the limited size of the network. To prevent the size of the network from affecting multiplexing gains, the base stations need to be able to respond proactively to any change in network load. A Long Short-Term Memory (LSTM) algorithm is going to be used for the prediction of the data traffic volume of a base station thereby providing the pre-knowledge of the network load. This will allow for proper provisioning of network resources, and using different clustering algorithms such as K-means, Hierarchical and Gaussian Mixture Models to cluster these base stations there is a reduction in the needed network resources and this reduces cost. Capacity Utility and cost of deployment are the metrics used in making a comparative analysis of the different clustering algorithms used in this work. From evaluation of the methodology, it showed that the Hierarchical clustering algorithm had a Capacity Utility of 0.0012, Gaussian Mixture Models had 0.0035 and K-means with 0.0044 and when this is evaluated against the Capacity Utility before clustering of 0.63 it can be seen that the Hierarchical clustering algorithm had reduced the needed network resources better than Gaussian Mixture Models and K-means. The 3 clustering algorithms were also able to reduce the number of needed base stations from 182 to 80, thereby reducing cost of deployment.

TABLE OF CONTENTS

Cover Page

Title Page i

Declaration ii

Certification iii

Acknowledgements iv

Abstract v

Table of Contents vi

List of Tables ix

List of Figures x

Abbreviation xii

CHAPTER ONE

1.0 INTRODUCTION 1

1.1 Background of Study 1

1.2 Statement of Research Problem 3

1.3 Aim and Objectives 3

1.4 Justification of Research 4

1.5 Scope of Study 5

1.6 The Structure of the Thesis 5

CHAPTER TWO

2.0 LITERATURE REVIEW 6

2.1 Overview of C-RAN Design and Features 6

2.2 Cloud RAN Architecture 6

2.3 Traditional RAN Architecture 7

2.4 Advantages of Cloud RAN 10

2.5 Challenges of Cloud RAN 10

2.6 Long Short-Term Memory (LSTM) 10

2.7 Concept of Clustering 12

2.8 Concept of Deep Learning 13

2.9 Clustering Algorithms and Mobile Network Performance Metrics 14

2.10 Algorithms for Optimizing Cost for Network Deployment using Clustering Techniques 18

2.11 Clustering Benefit for Network Capacity Utility Improvement 21

2.12 Chapter Summary 24

CHAPTER THREE

3.0 MATERIALS AND METHODS 25

3.1 Software 25

3.2 Source of Dataset 25

3.3 Methodology for Objective 1 27

3.3.1 Clustering Algorithm 27

3.3.2 Evaluation 28

3.4 Methodology for Objective 2 29

3.4.1 Deep Learning 29

CHAPTER FOUR

4.0 RESULTS AND DISCUSSION 34

4.1 Results of Clustering Algorithms 34

4.2 Cluster Calculations for Capacity Utility and Cost of Deployment 39

4.3 Result of Deep Learning 43

CHAPTER FIVE

5.0 CONCLUSION AND RECCOMENDATIONS 45

5.1 Conclusion 45

5.2 Recommendations 46

5.3 Contribution to Knowledge 46

REFERENCES 47

APPENDICES 51

LIST OF TABLES

Table		Page
2.1	Comparison between Traditional RAN and C-RAN	8
3.1	Dataset Description	26
4.1	Results of Cluster Algorithms	38
4.2	Results of Capacity Utility and Cost of Deployment	39
4.3	Comparison of Methodology	44

LIST OF FIGURES

Figure		Page
1.1	A C-RAN with 2 RRHs	2
2.1	Cloud RAN Architecture	7
2.2	Traditional RAN Architecture	8
2.3	3G RAN Architecture	9
2.4	Long Short-Term Memory Block	11
2.5	Deep Learning Framework Overview	16
2.6	C-RAN with Packet Based Architecture	19
2.7	iTREE System Model Structure	20
2.8	Movement of Network load over the Course of a day	22
2.9	C-RAN Network with 2 Fronthaul Segments	23
3.1	Milan Grid	26
3.2	Flowchart of the Clustering Algorithms	27
3.3	Architecture of the LSTM Model	30
3.4	Flowchart of Deep Learning LSTM Prediction	30
3.5	Diagram of Methodology	32
3.6	Flowchart of Cluster Algorithm Comparison	33
4.1	K-Means Clustering Algorithm	34

4.2 Hierarchical Clustering Algorithm 35

4.3 Gaussian Mixture Model Clustering 36

4.4 Comparison of the Cluster Algorithms 37

4.5 Milan Traffic Volume Prediction 44

ABBREVIATIONS

Acronyms	Meaning
BBU	Baseband Unit
BF	Balanced Fair
BnB	Branch and Bound
CPRI	Common Public Radio Interface
C-RAN	Cloud Radio Access Network
DCCA	Distance-Constrained Complementarity-Aware
FFT	Fast Fourier Transform
FH	Fronthaul
GOPS	Giga Operations Per Second
IQ	In-phase/ Quadrature-phase
ITREE	Intelligent Traffic and Resource Elastic Energy
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
NDA	Non-Disclosure Agreements
QoS	Quality of Service
RAN	Radio Access Network
RNN	Recurrent Neural Network

RRH	Remote Radio Head
SA	Simulated Annealing
VBS	Virtual Base Station
WCDMA	Wideband Code Division Multiple Access
WIMAX	Worldwide Interoperability for Microwave Access

CHAPTER ONE

1.0 INTRODUCTION

1.1 Background to the Study

In recent years, due to increase in personal devices and data-hungry mobile applications, the demand for an efficient and always available high data rate radio access network has increased (Visual and Index, 2019). The radio access network part of a base station is one of the most important and about 80% of the capital expenditure is spent on it (Bihnam, 2021). Moreover operating expenditure like power consumption and maintenance is also increasing (Ye Li *et al*., 2011). Cloud Radio Access Network (C-RAN) is an architecture to solve the problem of increase in capital and operating expenditure (Peng *et al*., 2021) . C-RAN was initially proposed by Lin *et al*. (2010) with further details in (Chen, 2011). One major drawback in the achievement of C-RAN is the limited fronthaul capacity. The fronthaul is the connection between the Remote Radio Heads (RRHs) and the Baseband Units (BBUs). The signals sent from the RRHs to the BBUs can be analog or digital (Mcclean and Morrow, 2020). The digitized or In-phase/Quadrature-phase (IQ) transmission is carried out using the Common Public Radio Interface (CPRI) protocol (Common Public Radio Interface, 2015). For C-RAN to be achieved there is need for a fronthaul connection with high bandwidth and low latency but in reality the available fronthaul is both capacity and time-delay constrained (Peng *et al.*, 2015). Different techniques have been used to solve this problem such as lossless compression and the use of power over fibre on the fronthaul (Alimi *et al*., 2018; Lorca *et al*., 2013; Ramalho *et al*., 2017).

In C-RAN there is the centralization of baseband processing of statistically varying traffic loads from different base stations, these loads are multiplexed and the processing resources are also shared. A base station is built to serve the maximum traffic load but due to variations in traffic load throughout the day, there may be times of underutilization. The peak of this multiplexed load across several base stations is seen to be lower than the sum of individual peak loads (Rish *et al*., 2001).

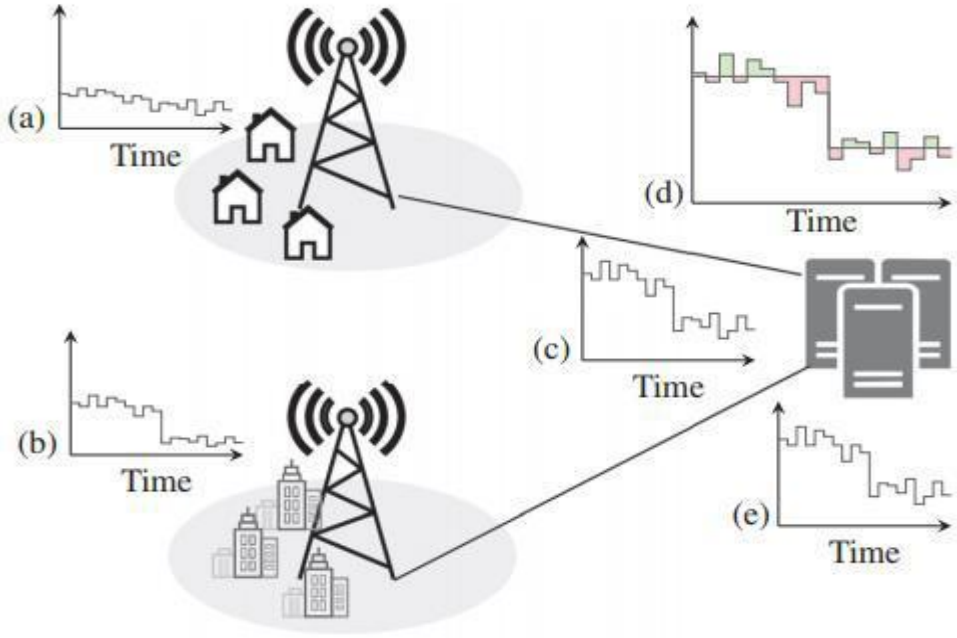

Figure 1.1: A C-RAN with 2 RRHs showing the application of BBU resource multiplexing to short-term (in frames) and long-term (in hours) load fluctuations (Kalor *et al*., 2017)

In Figure 1.1, (a) and (b) load at each RRH. (c) Aggregate load at the BBU. (d) long term multiplexing. (e) short term multiplexing.

This lowered peak load leads to reduction in processing resources and this leads to the statistical multiplexing gain of baseband processing resources as the bandwidth requirement

for each user compensates for each other due to the variability of the data traffic as shown in Figure 1.1. Multiplexing gain has the following benefits: Reduced cost, Reduced energy consumption, Better spectrum utilization, Better resource utilization, Scalability (Liu *et al*., 2014).

Long-term multiplexing is the capacity to adjust to slow variations and this slow variation is the change in traffic load during the course of a day. These changes occur in the minute to hour scale. Short-term multiplexing is the capacity to adjust to fast variations usually in the millisecond to seconds scale (Kalor *et al*., 2017).

1.2 Statement of the Research Problem

Some base station resources may be underutilized as usage of base station resources vary throughout the day and this can be reduced by clustering of base stations effectively (Checko *et al*., 2014) and exploiting their multiplexing gains. However mobile network operators over-dimension network resources due to lack of pre-knowledge of the data traffic volume. This research has considered clustering of base stations and deep learning technique to predict data traffic volume to provide this pre-knowledge.

1.3 Aim and Objectives of the Research

The aim of this research is to develop a cluster technique to cluster base stations effectively and predict its data traffic volume in cloud radio access network for multiplexing gain.

The objectives are:

i. To carry out comparative analysis of clustering algorithms to determine the better C-RAN clustering algorithm based on the following evaluation metrics: capacity utility and cost of deployment.

ii. To build an LSTM prediction model to predict data traffic volume with a low mean absolute error.

1.4 Justification of the Research

Several authors, Rish *et al*. (2001); Pompili *et al*. (2015); Bhamare *et al*. (2018); in their research have proposed several methods for achieving multiplexing gains in cloud radio access network. Other authors like Bhaumik *et al*. (2012); Werthmann *et al*. (2013); Taleb *et al*. (2018a) have proposed using clustering as a way to achieve multiplexing gain, Werthmann *et al*. (2013) proposed clustering several base station sectors into a single cloud base station to achieve operational cost reduction, but this was not tested on a true urban environment. Taleb *et al*. (2018a) proposed clustering algorithm was to achieve reduction in network power consumption and reduce transmission delay, this was complicated since you have to jointly solve for user association and RRH clustering. Bhaumik *et al*. (2012) proposed a Cloud IQ framework to reduce compute resources, its pitfall is that it was for a homogenous system.

Clustering in C-RAN is important because it allows for the increase in capacity utility, and this increase is as a result of the BBUs sharing the data computations of several RRHs in different times of the day (Bhaumik *et al*., 2012). The focus of this work is to improve Capacity Utility and reduction of Cost and this is important because this encompasses every other gain that can be achieved from clustering, it is also the solution to a major problem faced by mobile network operators (Lehrmann *et al*., 2014).

Chen *et al*. (2018) used several deep learning techniques and a Distance Constrained Complementary Aware (DCCA) clustering algorithm and compared the performance of the

various deep learning techniques but to the best of our knowledge none has compared the performance of several clustering algorithms with a single deep learning technique. Python programming language will be used for the simulation, real-world data from the Telecom Italia Big Data Challenge will be used. The evaluation will be comparing the capacity utility and cost reduction achieved by the clustering algorithm.

1.5 Scope of the Research

This research is limited to the use of several clustering algorithms to achieve multiplexing gains such as improved capacity utility and reduced cost, and prediction of mobile traffic data using deep learning technique.

1.6 Structure of the Thesis

This thesis is structured in 5 chapters as follows: Chapter 1 is the Introduction. Chapter 2 introduces various concepts like Multiplexing Gain, C-RAN, LSTM, Clustering, Deep Learning. and the Literature review that presents various approaches in achieving multiplexing gains through Clustering. Chapter 3 is the Research Methodology which presents the use of clustering and prediction algorithms in achieving the objectives of this research. Chapter 4 is the presentation of result and discussion; Chapter 5 is Conclusion and Recommendations.

CHAPTER TWO

2.0 LITERATURE REVIEW

2.1 Overview of C-Ran Design and Features

C-RAN is a network architecture where there is a separation of the Baseband units (BBUs) and Remote Radio Heads (RRHs), the BBUs are responsible for baseband processing such as coding, modulation, Fast Fourier Transform (FFT) and RRHs for radio functionalities such as digital processing frequency filtering and power amplification (Lehrmann *et al*., 2014). Baseband processing are concentrated into the cloud and their functions shared among various base stations in a BBU pool, which allows it to adapt to dynamic traffic patterns and use network resources more efficiently. This allows for the use of fewer BBUs thereby reducing capital and operating expenditure. It also reduces energy consumption; it improves scalability because new BBUs can be added easily. The third part of the C-RAN is the fronthaul transport infrastructure which provides connection between the Remote Radio Heads and Baseband Units. An Optical transport network is used to achieve this, but actualization of this is a big challenge due to the requirements of this interface such as high bandwidth (tens of Gbps), low jitter, low latency and cost.

2.2 Cloud Ran Architecture

The concept of C-RAN was first introduced by IBM (Lin *et al*., 2010). In Cloud RAN as seen in Figure 2.1, the Remote Radio Heads (RRHs) also called Remote Radio Units (RRUs) are separated from the Baseband Units (BBUs), these BBUs are concentrated into pools and these pools are shared among base stations so as to maximize the network resource usage from heavily and lightly loaded base stations. C-RAN is a concept of

combining the applications of cloud computing on mobile network radio access network. C-RAN was proposed to address the deficiencies of the traditional radio access network such as limited capacity, insufficient expendability, low utilization (Chen *et al*., 2015). C-RAN allows for low latency and high-speed communication between base stations. In C-RAN the BBUs are easily expandable since they are in the cloud.

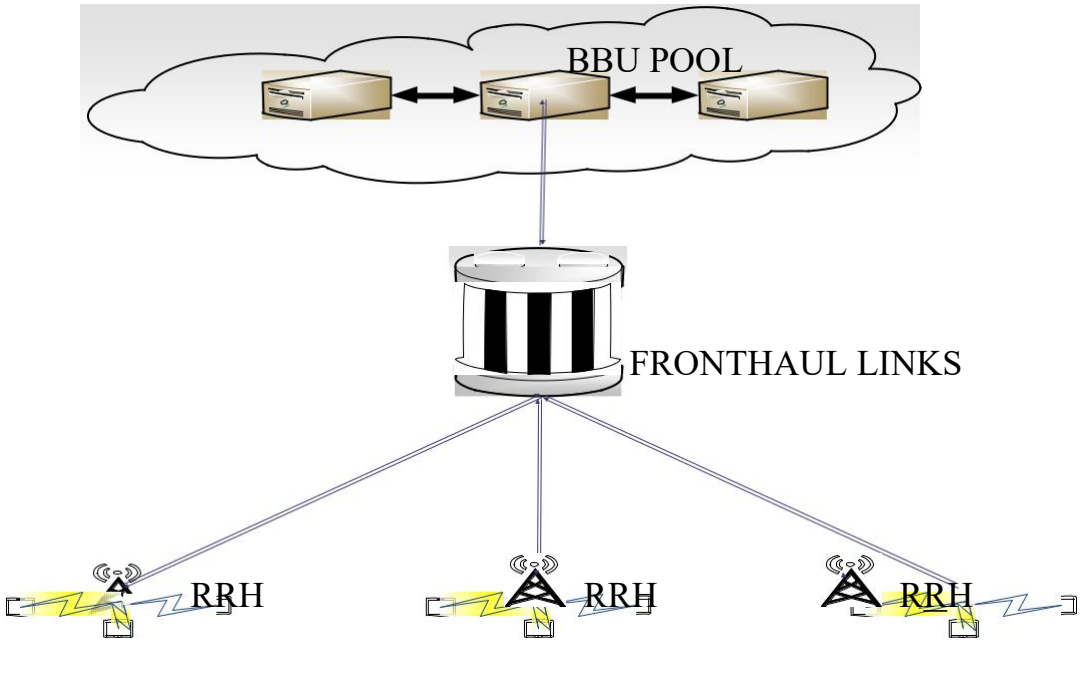

Figure 2.1: Cloud RAN Architecture

2.3 Traditional Ran Architecture

In traditional RAN the radio functionalities and baseband processing are incorporated inside a single base station. The antenna is just a few metres away from the radio module and is connected to each other with a coaxial cable. This was used for 1G and 2G network deployment. The traditional RAN architecture has a few disadvantages that the C-RAN is going to improve such as limited capacity because the traditional RAN imposes a hard capacity on the number of users in the network.

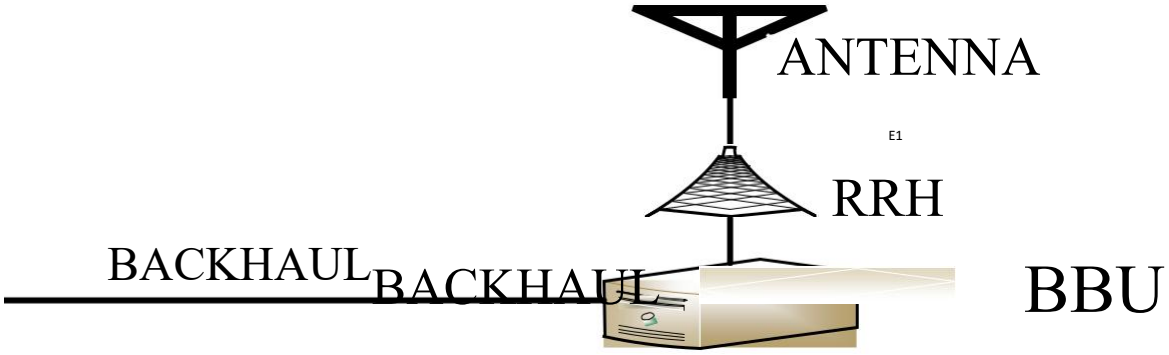

Figure 2.2: Traditional RAN Architecture

Table 2.1: Comparison Between Traditional RAN and C-RAN

RESOURCES	TRADITIONAL RAN	C-RAN
RRH and BBU Location.	The RRH and BBU are located together.	The RRH and BBU are separated by long distances.
Energy Requirement.	There is a higher energy requirement due to the RRH and BBU in the same location.	There is a lower energy requirement due to the separation of the RRH and BBU.
Network Resource Usage.	There is over-dimensioning of network resources which leads to under-utilization.	Network resources are optimally utilized due to proper dimensioning.
Transportation Cost.	Transportation cost are low.	Transportation cost are high due to the separation of RRH and BBU.

There is also the problem of scalability, as the only way to increase the coverage area in a traditional RAN system is to build new base station which imposes huge capital and operating expenditures (Chen, 2011). Because network resources have to be provisioned based on the busy period, the actual average network load is usually lower and this leads to under-utilization of network resources (Chen *et al.*, 2015).

A comparison between traditional RAN and C-RAN is presented in Table 2.1. As seen in Figure 2.2, in traditional RAN the RRHs and BBUs are located together on the base station site with the RRH and BBU just a few metres apart and these base stations are connected to the mobile core network through the backhaul (Lehrmann *et al*., 2014).

Figure 2.3 shows that in 3G networks the fronthaul connection between the RRH and BBU can be up to 40 km, the only limitation to this is processing and propagation delay. Optical and microwave connections can be used here.

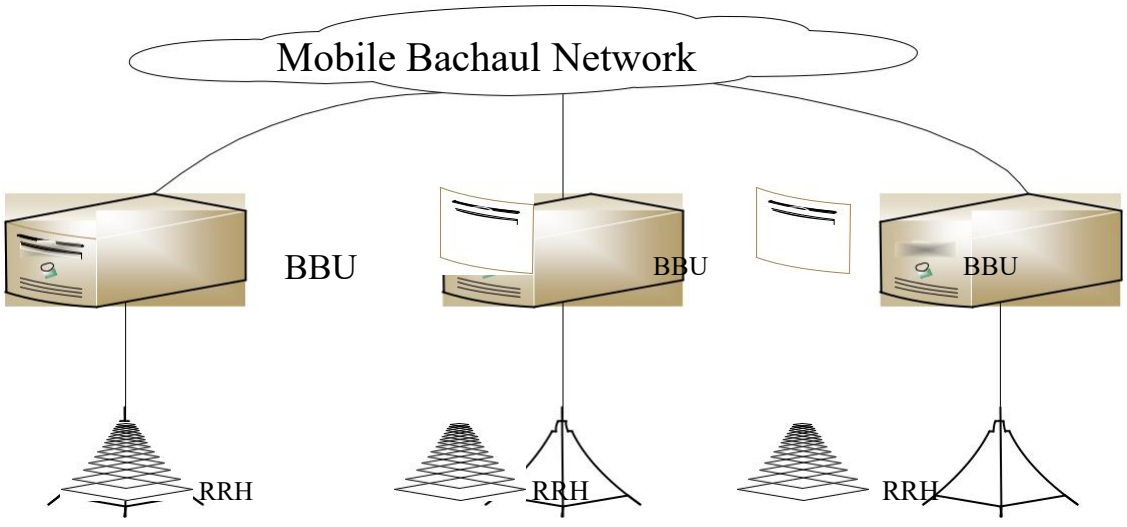

Figure 2.3: 3G Traditional RAN architecture

2.4 Advantages of Cloud Ran

There is reduction of operating cost due to less power consumption as a result of a less complex RRH and reduction in cost of deployment due to reduced price of the RRH. System throughput can be improved by a dense deployment of low power RRHs. It also prevents network resource wastage by aggregating network resources from different base stations.

2.5 Challenges of Cloud Ran

High bandwidth requirement, low latency and jitter and the high cost of the transportation network. Effective clustering to prevent network overloading is also a major challenge (Lehrmann *et al*., 2014). There is also the issue of security of the network, Due to the centralization of BBUs in the cloud there is a higher possibility of a single point of failure (Hossain *et al*., 2019).

2.6 Long Short-Term Memory (LSTM)

Long short term memory was first introduced by German researchers Sepp Hochreiter and Juergen Schmidhuber as a solution to the difficulty in training certain networks (Hochreiter, 1997). This difficulty is referred to as the vanishing gradient problem. This problem is as a result of a large sized input being forced into a small input space, when this occurs, the gradients of the loss function begin to approach zero resulting in training difficulties. The loss function is important because it a way to evaluate how well the model predicts the given data.

Long short-term memory (LSTM) is a type of recurrent neural network (RNN), a RNN is a network that takes input data one at a time and still maintain information from previous

inputs, it takes information from previous input in making new computations. LSTMs are able to pick out similarities in sequences of data like time series data, handwriting, music, stock market price. LSTMs are used for forecasting or prediction (Laptev *et al*., 2017). LSTMs use the state of the last neuron from the last time step as a template to create an output.

An important feature of the LSTM is a memory cell also called a memory block which can preserve its state over a long period of time. An LSTM cell contains gates which controls flow of data into and out of the cell (Greff *et al*., 2017).

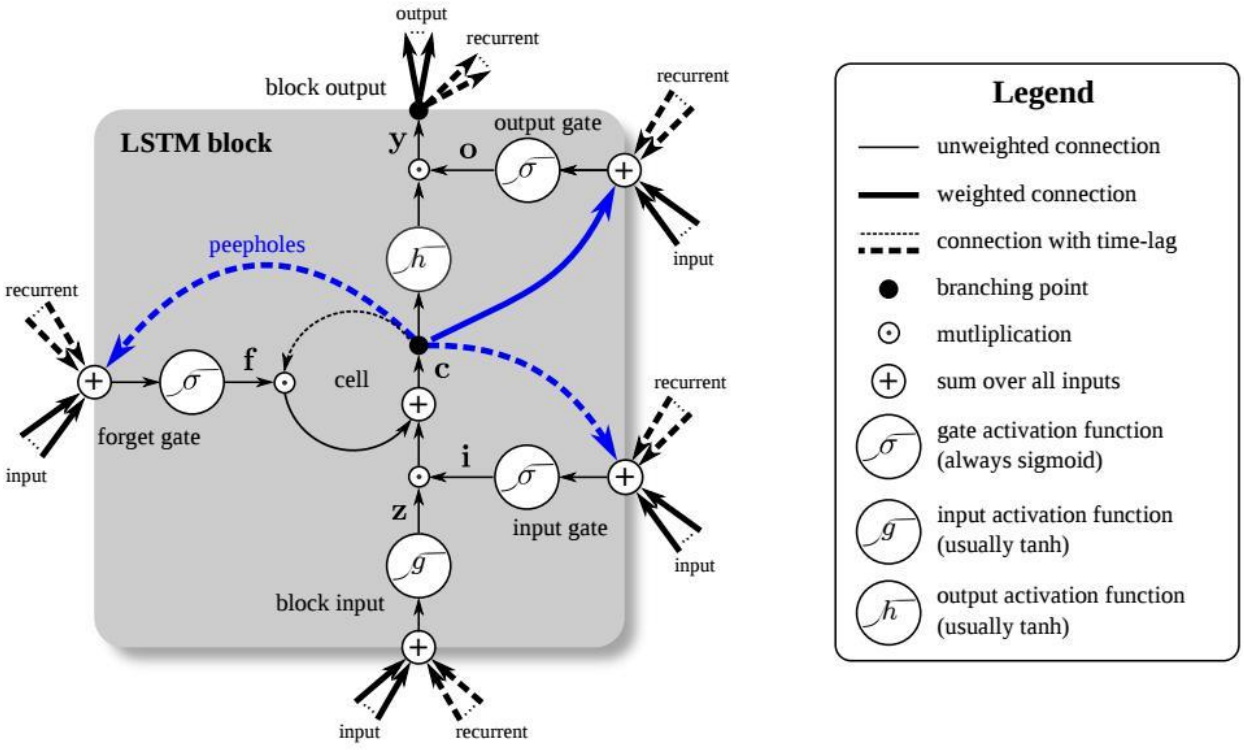

Figure 2.4: Long Short Term Memory Block (Greff *et al*., 2017)

LSTMs have three (3) gates which can be seen in Figure 2.4:

i. Forget gate: This determines what information to remove from the cell.

ii. Input gate: This determines the number of new values that flows into the cell.

iii. Output gate: This determines what information to output due to the input and memory of the cell.

Some applications of LSTMs are time series forecasting, sign language translation, image caption generation, translation of text, handwriting generation.

2.7 Concept of Clustering

Clustering is a method of grouping data. It is used to sort data that have similar attributes into groups (Subramaniyan *et al*., 2020). Clustering is commonly used for statistical data analysis. Some applications of clustering are grouping of authors or music by genre, customers based on purchases. An example of a clustering algorithm is the K-Means clustering algorithm which is a popular algorithm, it is used to segment data into k groups (Xia *et al*., 2020).

Step 1: In K-Means a number of k centroid (average) from the data is picked as the first cluster centres.

Step 2: Assign each data to the closest centroid.

Step 3: The centroids are moved to the Centre of the data and finally step 2 and step 3 are repeated until the clusters do not change.

Another clustering algorithm is the hierarchical clustering algorithm which could either be bottom-up (Agglomerative) or top-down (Divisive). Bottom-up assumes each data point as a cluster in the beginning and then combines pairs of clusters until they become a single cluster which comprises all the data points (Subramaniyan *et al*., 2020). In Gaussian mixture models we identify the clusters by their mean, covariance and size of the cluster.

These parameters are identified for each data point and the probability that it belongs to a cluster is estimated (Sridharan, 2017). These three algorithms were chosen for this work because the K-Means is considered one of the best for clustering due to its efficiency and simplicity (Xia *et al*., 2020). Hierarchical clustering algorithm is able to capture more complex and intricate cluster structures (Subramaniyan *et al*., 2020) and the gaussian mixture model is able to cluster both continuous and categorical data which is an advantage over other algorithms (Rajabi *et al*., 2019).

2.8 Concept of Deep Learning

Deep Learning is a subset of Machine Learning that has been used to extract information from data that are complex, have lots of noise where traditional machine learning techniques have not been as effective (Wang *et al*., 2020). Deep learning can be used to solve either classification, prediction or clustering algorithm problems. A classification algorithm is used to bring out a particular feature from a set of data, for example if a set of data is made up of furniture, a classification algorithm is able to tell which of the data is a chair or a table compared to the total set of data. A prediction algorithm (Regression) takes a time series data and outputs what that data will be in a future time step, and when the predicted data is compared to the initial data, the difference helps in evaluating the performance of the prediction algorithm (Längkvist *et al.*, 2014). A clustering algorithm is performed on data that are not properly labelled and brings out information by segmenting data that are similar. Deep learning is of great importance in mobile networks because due to increase in mobile devices, there is going to be commensurate increase in data produced and deep learning performs better when there is a complex dataset, it is also used on unlabeled data which are prevalent in mobile network data (Zhang *et al*., 2019).

In this section, a presentation is made on how various authors were able to achieve multiplexing gains in C-RAN through clustering. The problems they were trying to solve, methodologies used and results achieved.

2.9 Clustering Algorithms and Mobile Network Performance Metrics

C-RAN provides the means by which BBUs can be aggregated in a central location (pool) and this allows for baseband resources to be shared and this leads to multiplexing gains. To analyze the multiplexing gain which can be gotten from a BBU pool. Werthmann *et al*. (2013) carried out a study using a computation resource model. To do this a model which includes the user and traffic model, compute resource model and the radio network model was used. The simulation showed that due to channel conditions compute resource utilization was about 80% of the theoretical maximum. When more sectors were clustered into a single BBU pool, multiplexing gain increased thereby saving compute resources. This model could also be used for cell deployment. A user's location has also been shown to affect compute resource utilization. This model was tested on a limited size network which does not truly reflect real life scenario. The authors advised that a heterogeneous network to better represent a true urban environment should be used as an improvement on the work.

Liu *et al*. (2016) proposed the use of a multi-dimensional Markov model to investigate the statistical multiplexing gain of a BBU pool. This model captured the changes and restrictions imposed by both the radio and computational resources. This model showed that BBU pools can result in 75% multiplexing gains with 50 BBUs. The multiplexing gain was more obvious with light traffic and a tight QoS requirement. There was no accommodation for different types of data traffic burstiness which was a drawback. This

model could be further refined to accommodate for several resource usage patterns of two data types like real-time and delay-tolerant traffic which are more likely to occur together in real systems.

Chen *et al*. (2018) proposed a deep-learning-based framework shown in Figure 2.5 to achieve statistical multiplexing gain of increased capacity and reduction in deployment cost. Traffic patterns are random in time and locations and due to this they are hard to predict in advance to create adequate clustering schemes, to alleviate this problem a Multivariate LSTM (Long Short-Term Memory) model was used to forecast the traffic patterns of Remote Radio Heads' and these forecasted traffic patterns were used to form clusters of similar Remote Radio Heads' into BBUs. A DCCA (Distance-Constrained Complementarity-Aware) algorithm was also proposed to create adequate clustering schemes. Two months' worth of real-world data from mobile networks in Milan and Trentino, Italy was used to measure the performance of the proposed framework which showed an increase in capacity by 83.4% in Milan and 76.7% in Trentino and a reduction in deployment cost to 48.4% in Milan and 51.7% in Trentino of the traditional radio access network. The datasets and traffic patterns used in this work were limited but further improvements can be made by using different sizes of BBU capacity. More datasets can be used to evaluate the framework and performance of the deep-learning based method can be studied using different traffic patterns.

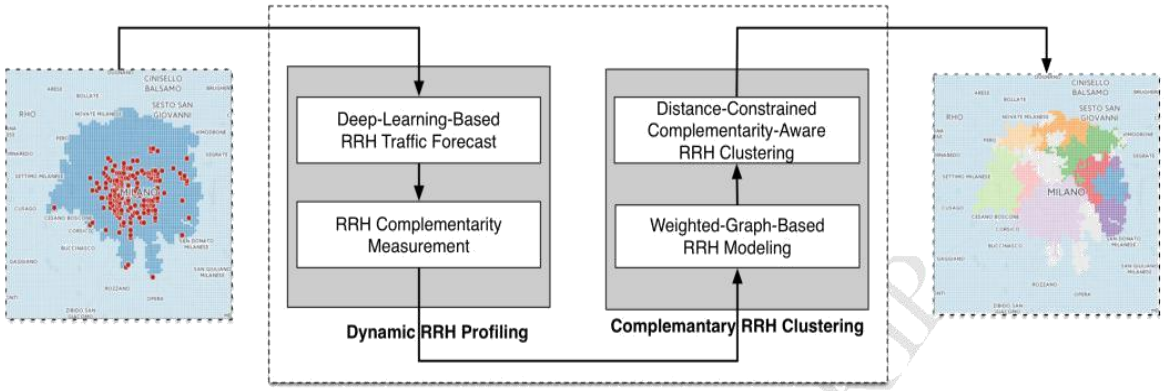

Figure 2.5: Deep Learning Framework Overview (Chen *et al*., 2018)

To address compute resource management, Bhaumik *et al*. (2012) proposed a Cloud IQ framework that had two aims; the first aim was clustering a set of base stations based on similar compute platform for a given statistical guarantee and the second aim was scheduling a cluster of base stations to meet their real-time processing needs. Traffic logs of 21 cell sites of a WCDMA network in a dense urban area were used to calculate the processing load and the result of resource pooling across several base stations. In this paper the authors used the network load between 8:00 am and 10:00 pm to verify if there was a reduction in the network load as a result of the pooling of different traffic across several base stations. It was shown that for a probability of failure of one in a million, this framework saved up to 19% of compute resources. It was also shown that by centralizing the processing of signals from different base stations, there was a reduction in compute resources by at least 22% due to the variations in network load across different base stations. This resource management system was for a homogenous system and should be extended to a heterogeneous system.

For the problem of User Association, which is determining which RRH users will connect to, and the problem of RRH clustering, which is determining which RRH are going to be grouped together to achieve statistical multiplexing gain, Taleb *et al*. (2018a) proposed a solution which was a framework to jointly optimize the User association and RRH clustering to reduce total time-response of the network and power demand of the network. To achieve this framework a network cost function was first derived which is the weighted sum of the total transmission delay and power consumption of the C-RAN network. After the network cost function was found, an optimization problem to minimize the network cost becomes the next step. It should be noted that the joint problem solution was complex. The use of multiple centralized and distributed user association and RRH clustering solutions should be explored for future works. In trying to achieve Delay reduction, Bhamare *et al*. (2018); Wu and Ghosal, 2016) proposed different methods. Wu and Ghosal (2016) analyzed the multiplexing gain of C-RAN compared to the traditional radio access network by looking at how the aggregation of the baseband unit into the cloud would improve the time-response of the system when there was a restrain on the capital expenditure of the network. The main contribution of this paper was the development of an analytic model that allowed for the use of a balanced fair (BF) resource allocation, which permitted adaptive resource sharing. A time-response minimization framework with network cost constraint was presented. When the multiplexing gain of C-RAN and the traditional RAN were compared under this framework. There was significant multiplexing gain and optimum average delay bound was obtained. The analytical and numerical results showed that these gains were dependent on a number of factors like network size, traffic characteristics and resource cost. This model provides some guidelines into the configurations of C-RAN systems.

2.10 Clustering Algorithm for Optimizing Cost for Network Deployment Using Clustering Techniques

Shehata *et al*. (2017) proposed an analytical model that showed how the geographical location type and distribution of users determine the multiplexing gain that could be achieved in a network. This analytical model captured the concentration of the RRHs and concentration of users, users distribution, users to RRH association strategy, scheduling algorithms for the resources of the RRHs and the computational effort model to calculate Giga Operations Per Second (GOPS) for each user. The authors found out that a sub-urban location provided the best multiplexing gain and lowest pooling cost and a higher multiplexing gain could be achieved when the user is normally-distributed as opposed to uniform distribution and when there is a higher cluster of RRH for a given location. Due to large distance needed to be covered by fronthaul fibre, there is an increase in cost which is a limitation to the lowered pooling cost achieved.

In trying to improve on the statistical multiplexing gain in C-RAN, Checko *et al*. (2014) proposed a packet based architecture shown in Figure 2.6 which can adjust to dynamic traffic patterns. Analytical and Modelling methods were used on the traffic patterns to determine the best combination of base stations in a BBU pool. Multiplexing gain was gotten by comparing the resources needed in a traditional RAN to that of C-RAN using equation:

$$\text{MultiplexingGain} = \frac{\sum^{N} \text{PeakThroughput}(n)}{\text{PeakThroughput}_{C-RAN}} \tag{2.1}$$

In their work it was shown that connecting 20-30% of office base stations and 70-80% of residential base stations to the BBU pool increased the statistical multiplexing gain by a

factor of 1.6. The work also showed that by mixing of traffic profiles, the number of BBUs required to provide coverage for an area can be reduced, leading to a reduction in cost. It should also be taken into account that synchronization in packet based fronthaul has strict requirements which is not easy to achieve. For future works the use of integer programming for cost optimization for a scenario with multiple BBUs is advised by the authors.

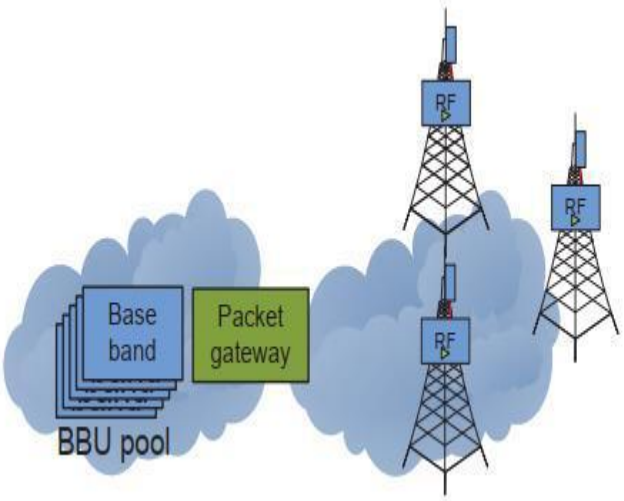

Figure 2.6: C-RAN with Packet Based Architecture (Checko *et al*., 2014)

Sigwele *et al*. (2015) proposed a green Intelligent Traffic and Resource Elastic Energy (iTREE) scheme to achieve multiplexing gain. This scheme in Figure 2.3 was able to reduce the amount of BBU needed by equating the right number of baseband processing with traffic load and also able to switch off BBUs not in use to conserve energy. Results show that there was a 97% reduction of BBUs during off peak period and 66% during peak periods and energy reduction in the radio access network by 27% during off peak period and 18% during peak period. This scheme can be improved by offloading traffic unto a macro cell and switching of the RRH. From Figure 2.7 it can be seen that a number of RRHs were connected to the network and served by a maximum number of BBUs denoted

by “BBU M” in such a way that the number of needed BBUs are reduced. The cloud controller is made up of the BBU-RRH table and the RRH profile, The RRH profile monitors the amount of traffic load in the RRH while the BBU-RRH monitors the BBU usage.

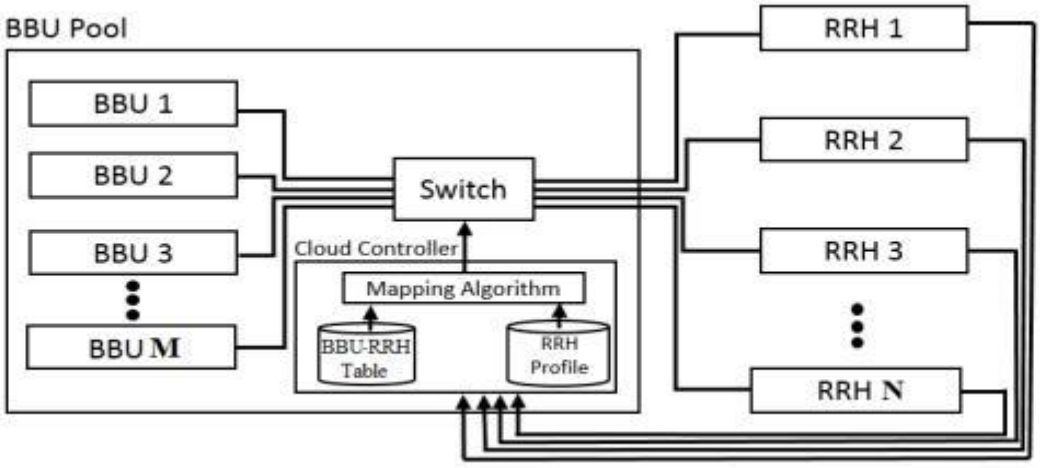

Figure 2.7: iTREE Model Layout (Sigwele *et al*., 2015)

In C-RAN BBU functionalities are carried out on a virtual machine in the cloud, virtual functions are these BBU functionalities built into software. Bhamare *et al*. (2018) proposed a combined optimization model with two methods namely: branch-and-bound (BnB) and simulated annealing (SA) for optimal placement of virtual functions to reduce delay, capital expenditure and operating expenditure of C-RAN. The proposed solution also addresses link delays, service migration delays and cost of cloud deployment by determining the optimum number of clouds to be deployed. One drawback of this work is that the solutions

presented here relied solely on the reactive approach. The solution to this drawback, is the use of machine learning to predict network load so as to respond proactively.

2.11 Clustering Benefit for Network Capacity Utility Improvement

One of the earliest works on multiplexing gain was carried out by Rish *et al*. (2001) they used traffic simulation experiments to evaluate the multiplexing gains in a WiMAX base station. Results of their experiment showed that multiplexing gain increased as the number of base stations increased and when the traffic of these base station are bursty. The authors advised the use of different mixes of traffic profiles for evaluation to see if there will be an improvement on the multiplexing gains.

Pompili *et al*. (2015) proposed a Demand-aware Resource Provisioning to address the problem of tidal effect, which is the change in network load of a base station due to the time of the day and efficient utilization of network resources. A fixed resource allocation by a base station for the worst-case scenario will lead to underutilization for that same base station at some other time. The proposed Demand-aware Resource Provisioning exploits one of the main components of C-RAN which is the centralization of the BBUs into the cloud. These centralized BBUs can also be referred to as virtual base stations (VBS) which can be virtually resized to handle the unstable network load. The Demand-aware Resource Provisioning comprises a proactive and reactive component. In the proactive component the variations in network load is known beforehand and network resources are provided accordingly for a limited period. For the reactive component, it monitors the CPU, Memory and Network utilization of the BBUs to compensate for either an over- or under-provisioning in the network. As shown in Figure 2.8 during the day when people are at work in Downtown, VBS #2 will have more computing resources since it has the largest

network load compared to those in residential area VBS #3 and at the football stadium VBS #1. But at night when there is a game in the stadium more resources will be allocated to VBS #1 to handle the network load and there will also be more resources provided to VBS #3 to handle the load of users in the residential area.

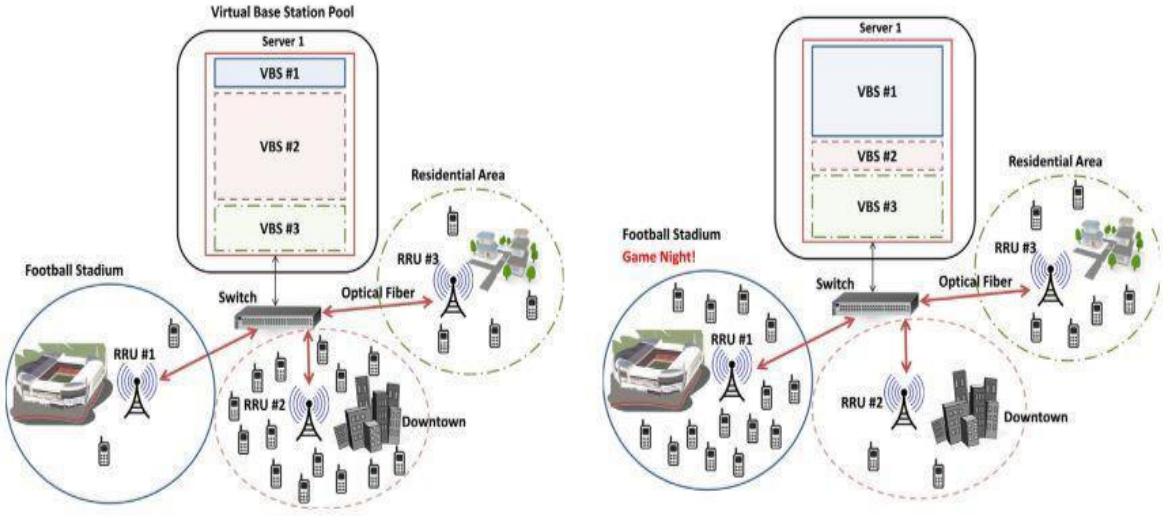

Figure 2.8: Movement of Network Load over the Course of a Day (Pompili *et al*., 2015)

The methodology, Demand-aware Resource Provisioning was able to result in an optimum usage of computational resources but the obtained results were not validated on real-time emulation software. The Use of OpenBTS, OpenLTE which are open-source implementations of software Base-station protocol stack to validate the methodology will be used in future works.

Fronthaul capacity in C-RAN is finite and this becomes a major hurdle due to the large amounts of data as a result of the explosion in mobile devices. Chaudhary *et al*. (2017) has proposed the using of queueing theory and traffic model to achieve multiplexing gain which is the reduction of the required fronthaul capacity. The multiplexing gain will depend

on the difference of the data traffic. Results showed that higher traffic difference led to higher reduction in the required fronthaul capacity. In this paper an architecture where several RRHs are clustered into an aggregation network and the traffic is sent to the BBU through the FH segment II, and the connection between the RRHs and aggregation network is FH segment I is presented in Figure 2.9.

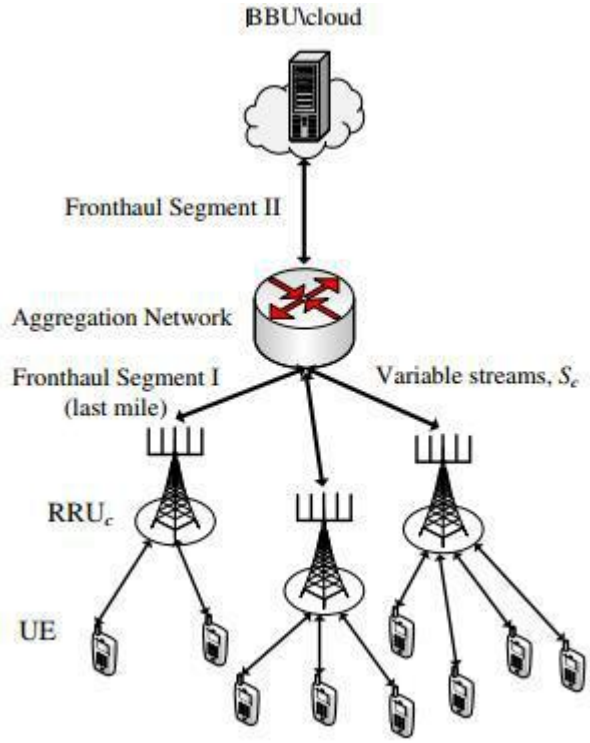

Figure 2.9: C-RAN Network with 2 Fronthaul Segments (Chaudhary *et al*., 2017)

2.12 Chapter Summary

In Wu and Ghosal (2016), the proposed methodology could also provide some guidelines in the configuration of C-RAN systems. The work done by Chen *et al*. (2018) can be improved by using different sizes of BBU capacity. More datasets can be used to evaluate the framework and performance of the deep-learning based method can be studied using different traffic patterns. Pompili *et al*. (2015) in solving the problem of tidal effect their

results were not validated on software that emulated real-time conditions, this can be addressed by the use of different implementations of software Base-station protocol stack to validate the methodology. Bhamare *et al*. (2018) will be improved by the use of deep-learning and machine learning to predict network load so as to respond proactively. The use of multiple centralized and distributed user association and RRH clustering solutions can be used to further the work in Taleb *et al*. (2018b). Checko *et al*. (2014) advised the use of integer programming for cost optimization for a scenario with multiple BBUs in improving their work. The use of heterogeneous network to better represent a true urban environment will improve on the work done by Werthmann *et al*. (2013). The use of different mixes of traffic profiles should be evaluated to find out if there was an improvement on the multiplexing gains in the work done by Rish *et al*. (2001). Liu *et al*. (2016) proposed a multi-dimensional Markov model which can be further refined to accommodate for several resource usage patterns of two data types like real-time and delay-tolerant traffic which are more likely to occur together in real systems. A cloud IQ framework that was proposed by Bhaumik *et al*. (2012) should be extended to heterogeneous systems. The green Intelligent Traffic and Resource Elastic Energy (iTREE) scheme in Sigwele *et al*. (2015) can be improved by offloading traffic unto a macro cell and switching off the RRH.

CHAPTER THREE

3.0 MATERIALS AND METHODS

In this chapter, the methodology used to achieve improved capacity utility and cost reduction by the use of deep learning technique and clustering algorithm is introduced and elaborated.

3.1 Software

The software applied in this work is the python programming language with several modules (libraries) for the deep learning technique and clustering algorithm.

3.2 Source of Dataset

Availability of telecommunication data for research is scarce because companies that generate these data only provide them to researchers who sign non-disclosure agreements (NDAs) and work for them. To aid the ease of research Telecom Italia in conjunction with several companies arranged the “Telecom Italia Big Data Challenge”. The dataset used in this work was extracted from the 2014 edition.

The dataset contains telecommunication activity for the city of Milan and Trentino but focus is on the city of Milan. The description of the dataset used is shown in Table 3.1. The dataset of the city of Milan comprises:

i. A grid dataset which is a geographical representation of the network cells in a grid pattern over the city as shown in Figure 3.1.
ii. A telecommunication dataset which contains a time series data of Incoming and Outgoing Calls, SMS and Internet traffic which is generated every 10 minutes.

Table 3.1: Dataset Description

NUMBER OF CELLID	10,000
NUMBER OF RRHS	182
MEAN OF TRAFFIC VOLUME	361.75

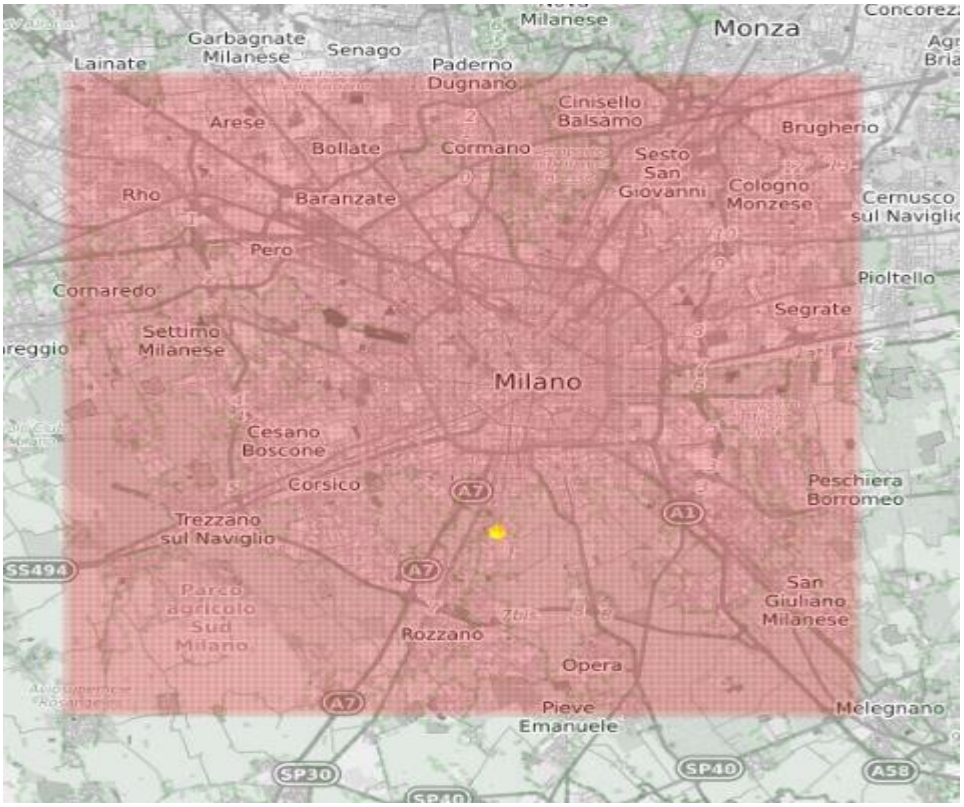

Figure 3.1: Milan Grid

The dataset is available at (https://www.kaggle.com/marcodena/mobile-phone-activity#sms-call-internet-mi-2013-11-01.csv).

3.3 Methodology for Objective 1

In achieving the first objective, clustering was employed. The details of the implementations are spelt out below.

3.3.1 Clustering Algorithm

In the clustering phase, from Figure 3.2 the first step is to import the needed clustering libraries after which the dataset is imported. Missing datapoints were dropped during preprocessing to remove missing information in the dataset so as to avoid errors. During preprocessing the specific features of the dataset to be clustered is chosen. The chosen features are fit to the clustering algorithm to be used which performs the clustering on the dataset. The membership of all clusters is displayed and the cluster graph plotted after which individual cluster membership is displayed.

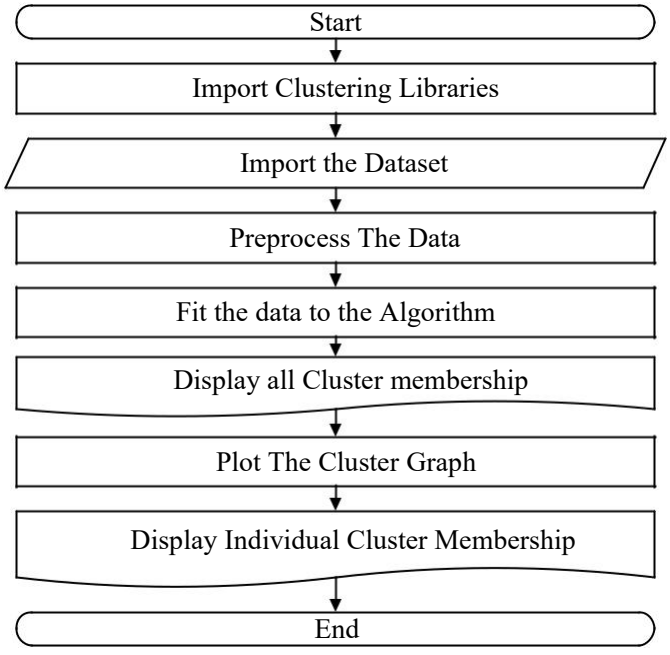

Figure 3.2: Flowchart of the Clustering Algorithms

The python programming language was used with several modules (libraries) to implement the clustering algorithm such as:

i. Sci-kit learn: This is a Python library that contains several unsupervised learning algorithms such as clustering algorithms (Pölsterl, 2020).

ii. NumPy: This library in Python allows for numerical data to be structured into arrays for easy numerical analysis (Harris *et al*., 2020).

3.3.2 Evaluation

The formula to calculate the capacity utility and cost is as shown below:

$\text{Utility(P)} = \text{mean}_{Ck}\, U(C_k)$ (Chen *et al*., 2018) (3.1)

$$Cost(P) = \sum_{k=1}^{K} |C_k| \quad (3.2)$$

Where $P=\{C_1,\ldots C_k\}$ is the clustering scheme. (3.3)

$$U(C) = \left(\frac{mean\, f(C)}{|B|}\right)^{-\frac{mean\, f(C)}{in\,|B|}} \quad (3.4)$$

Equation (3.4) is used to calculate the capacity utility of a BBU B connected to a Cluster C.

$$mean\, f(C) \quad (3.5)$$

Equation (3.5) is the mean aggregated traffic volume of cluster C and $|B|$ is the BBU capacity.

If the traffic volume of a base station is normalized, we can say that the BBU capacity of that base station is 1 so that the number of base stations covering an area is also the BBU capacity for that area.

To calculate the cost of deployment the BBU capacity required in each cluster is added up. The BBU capacity of a cluster used in this work is based on empirical experiments carried out by Chen *et al*. (2018).

3.4 Methodology for Objective 2

In achieving the second objective, deep learning was employed. The details of the implementations is spelt out below.

3.4.1 Deep Learning

From Figure 3.3 the first step is to import the various deep learning libraries to be used after which the training set is also imported. As is usually the case most datasets contain some missing information or the entries are not properly formatted which can lead to errors. This is fixed by preprocessing the data to remove such entries and avoid errors. For ease of computation, feature scaling is done to scale the dataset between 0 and 1. A data structure of 60 time-step and one output is created followed by the reshaping of the data into a 3-D array because the NumPy library only uses a 3-D array. The next step is to build the LSTM prediction model. A 4 stack LSTM model is used after which the test dataset is imported to test the LSTM model that was built. A prediction graph is then plotted to show the comparison between the ground truth and predicted data. To evaluate the performance of the LSTM prediction model, the Mean Absolute Error (MAE) is calculated which is the difference between the historic dataset used and the predicted dataset.

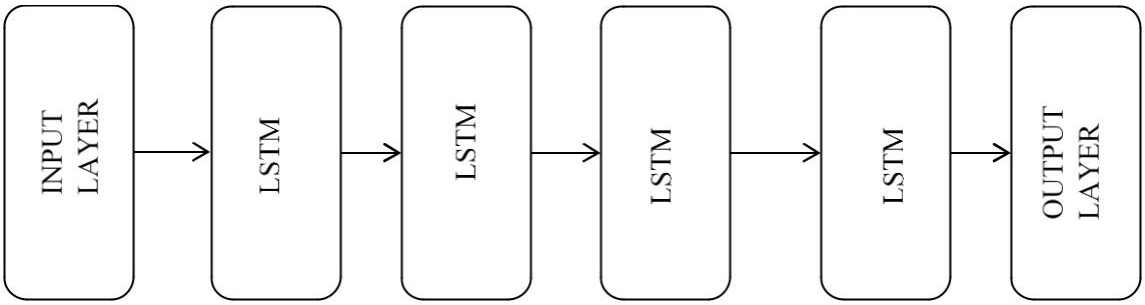

Figure 3.3: Architecture of the LSTM Model

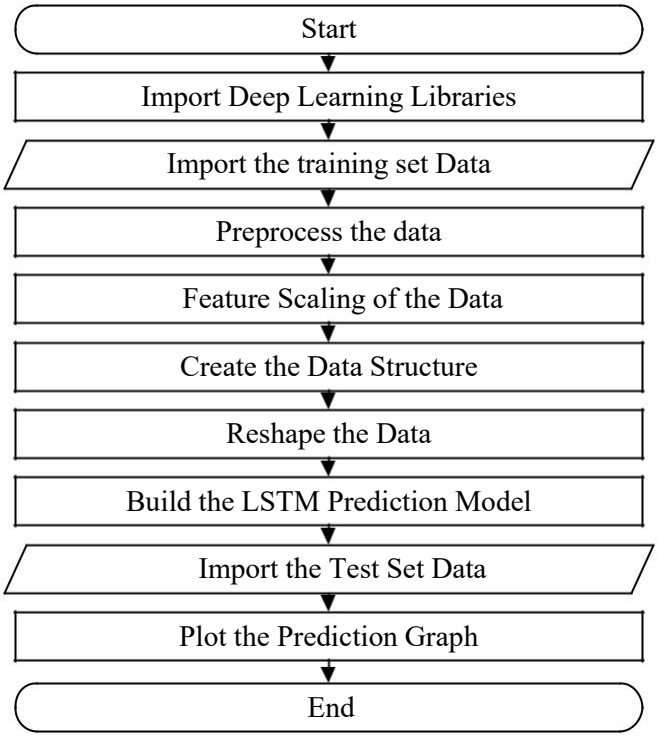

Figure 3.4: Flowchart of the Deep Learning LSTM Prediction

The python programming language was used with several modules (libraries) to implement deep learning such as:

i. Keras: This is a Python library that allows for the creation of powerful networks in deep learning (Ketkar, 2017).

ii. NumPy: This library in Python allows for numerical data to be structured into arrays for easy numerical analysis (Harris *et al*., 2020).

iii. Pandas: This is a Python library for efficient data structuring and extracting features from datasets in different fields like engineering and finance (Sapre and Vartak, 2020).

iv. Matplotlib: This is a Python library used in plotting of various types of graph (Sial *et al*., 2021).

The Mean Absolute Error is Calculated using:

$$\text{MAE} = \frac{1}{n} \left| A_i - P_i \right| \text{ (Wang } et\ al\text{., 2016)} \quad (3.6)$$

where:

i. n is the number of test samples.

ii. A_i is the historic dataset or ground truth.

iii. P_i is the predicted dataset.

In this work, from the dataset the CellID is used to represent an area covered by a RRH and Internet traffic as the mobile traffic volume. The first step is forecasting of mobile traffic data of Remote Radio Heads (RRHs) using the LSTM deep learning technique from previously known mobile traffic data. The purpose of this forecasting is to be able to identify data traffic volume and pattern beforehand or in advance. From Figure 1.1 it is seen that the traffic patterns change under different time (tidal effect), or circumstances, but deep learning forecasting is able to track this change. The data is also clustered to a set of Baseband Unit (BBU) pool using K-Means, Hierarchical and Gaussian Mixture Model

clustering algorithm to show which clustering algorithm achieves higher capacity at low cost. The diagram of methodology is shown in Figure 3.5.

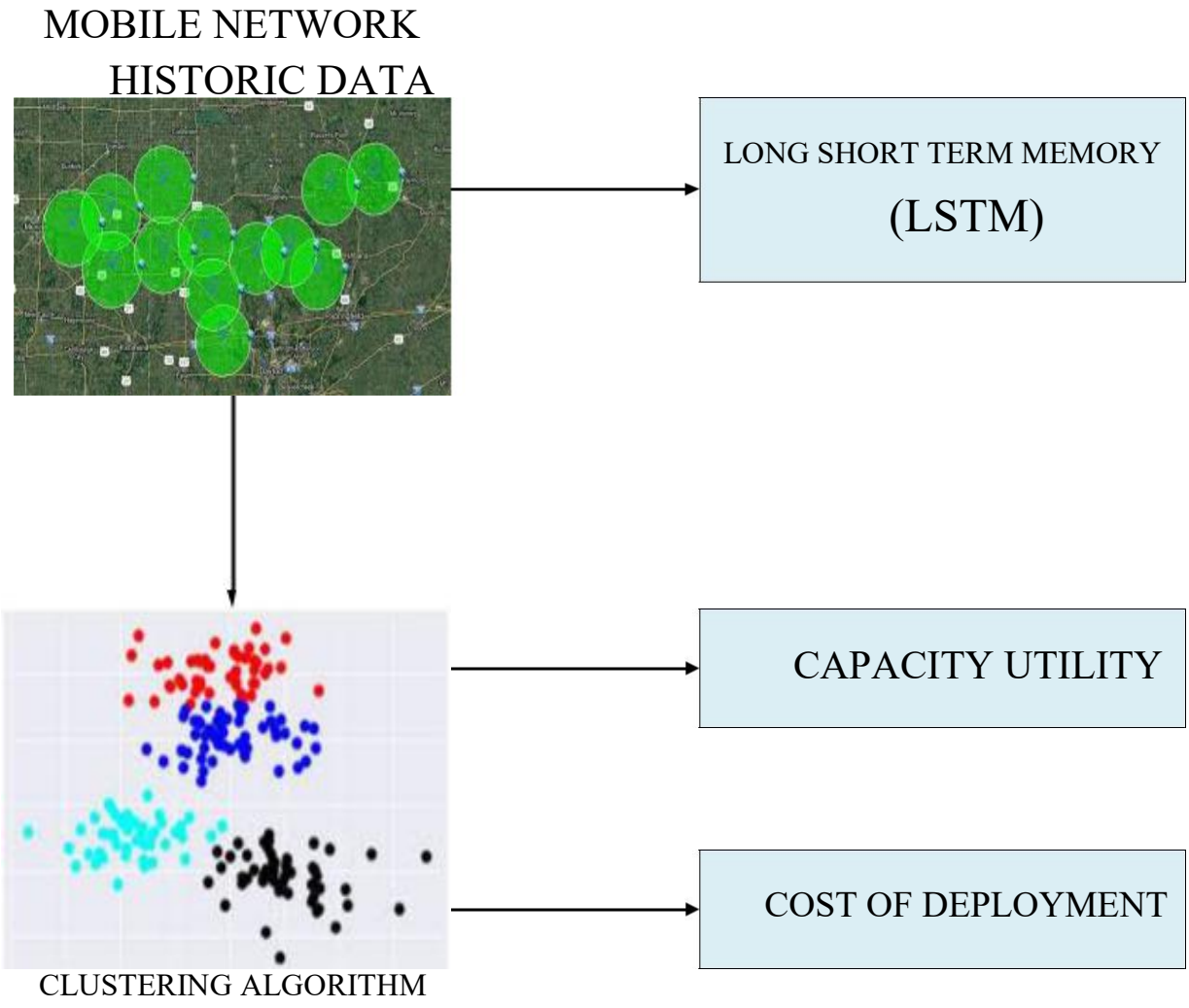

Figure 3.5: Block Diagram of Methodology

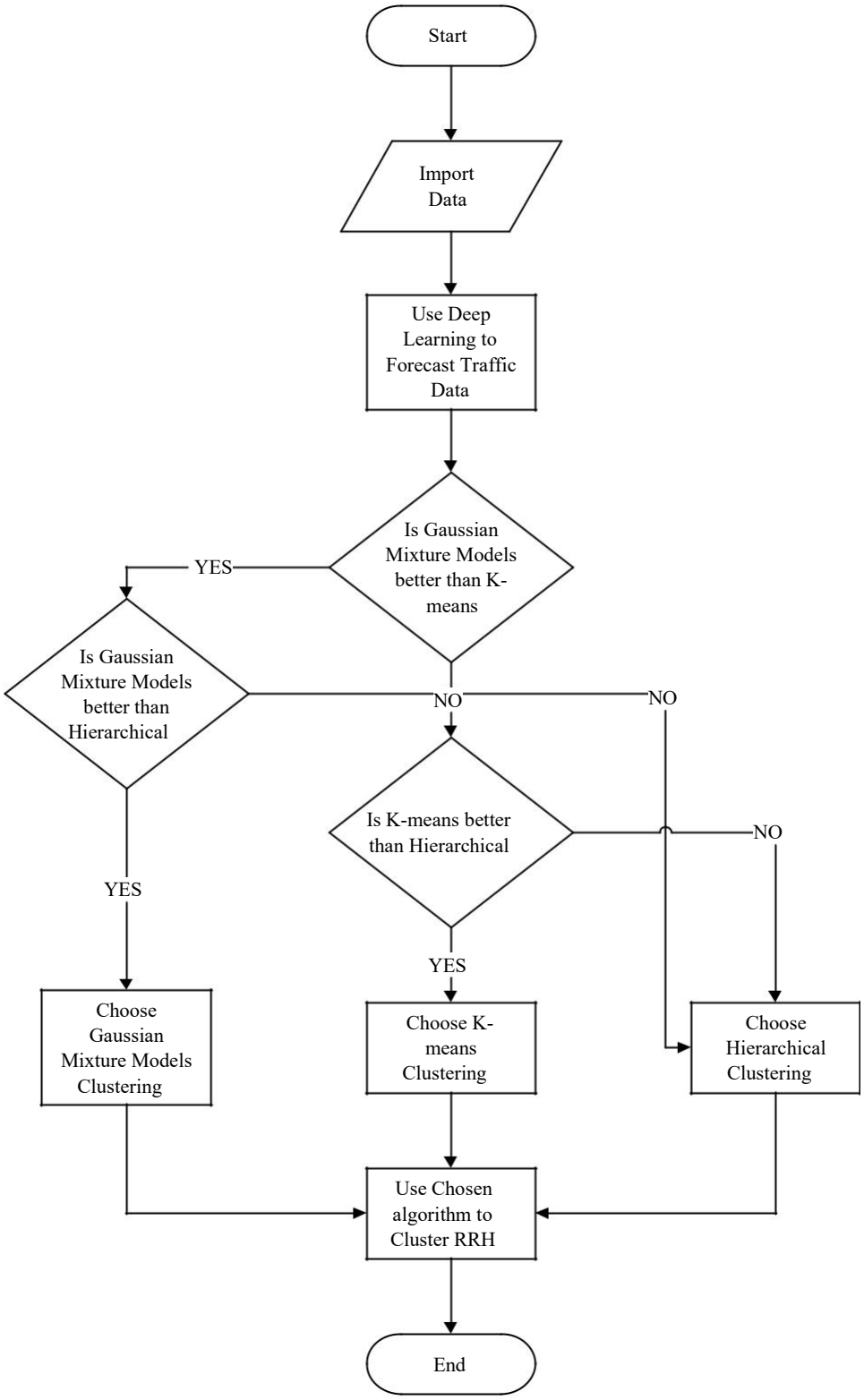

Fig. 3.6: Flowchart of Cluster Algorithm Comparison

CHAPTER FOUR

4.0 RESULTS AND DISCUSSION

4.1 Results of Clustering Algorithms

In this work the K-means cluster algorithm was used to cluster a mobile network of 10,000 cells by their data traffic. 10 clusters were formed and the graph in Figure 4.1 is a representation of each cluster with the mean of its data traffic. It is clear that most of the clusters in this clustering algorithm have similar number of cells in a cluster except the fourth cluster which has a very high number of cells in its cluster. It is also notable that in this cluster algorithm the mean of the data traffic is low. In Table 4.1 the number of cells in a cluster and its mean data traffic is clearly presented.

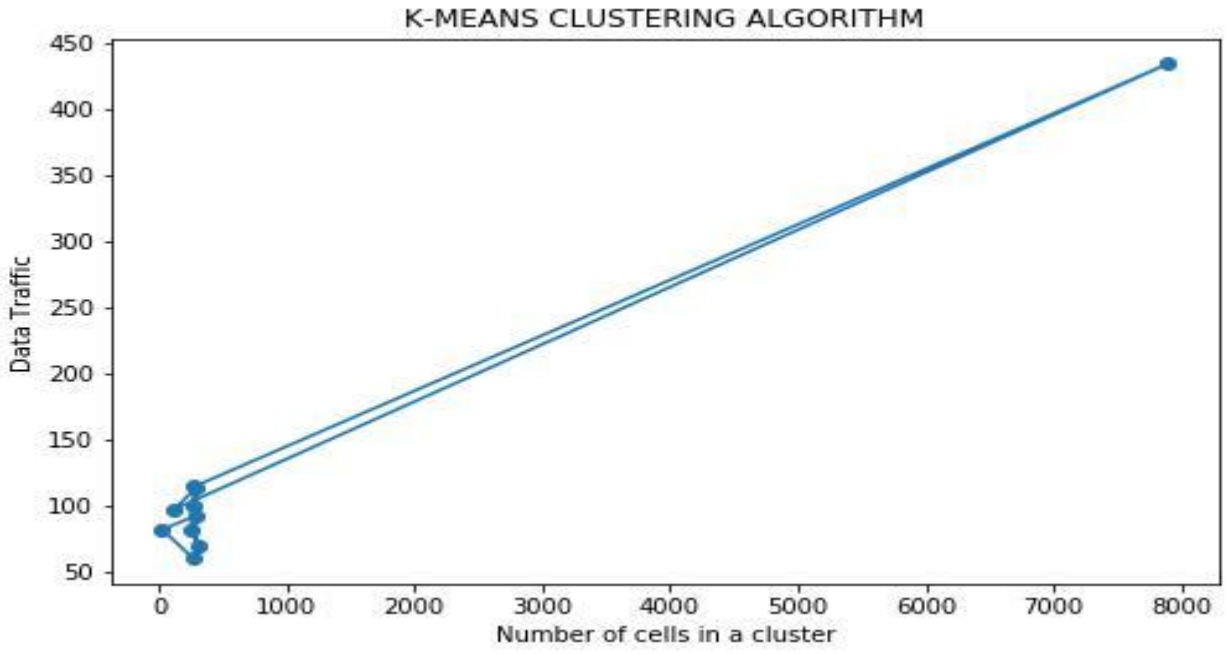

Figure 4.1: K-Means Clustering Algorithm

In this work the Agglomerative Hierarchical cluster algorithm was used to cluster a mobile network of 10,000 cells by their data traffic. 10 clusters were formed and the graph in Figure 4.2 is a representation of each cluster with the mean of its data traffic. In Hierarchical cluster, there is a higher mean of the data traffic and number of cells in each cluster when compared to the K-means.

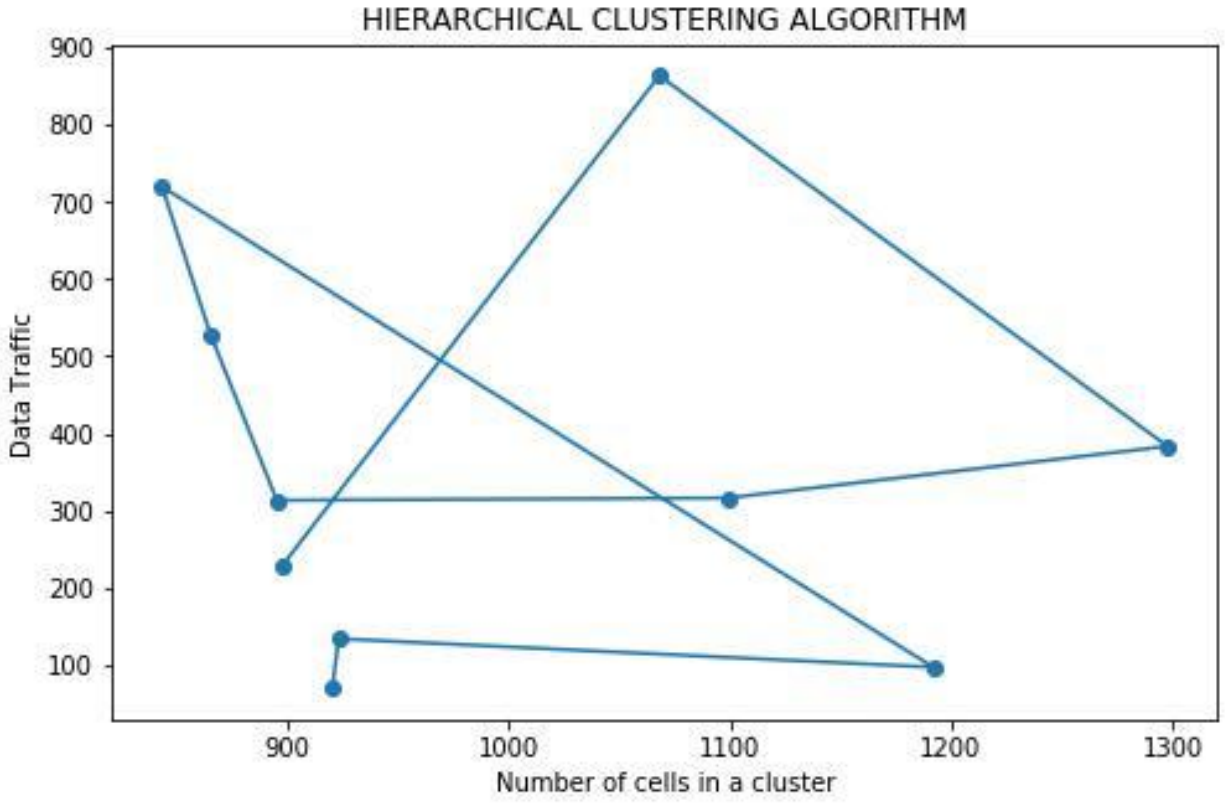

Figure 4.2: Hierarchical Clustering Algorithm

In this work the Gaussian Mixture Models cluster algorithm was used to cluster a mobile network of 10,000 cells by their data traffic. 10 clusters were formed and the graph in Figure 4.3 is a representation of each cluster with the mean of its data traffic. In Gaussian Mixture Models when compared with the Hierarchical clustering algorithm the difference in number of cells and mean of data traffic in each cluster is not so obvious but the higher the number of cells the higher the mean data traffic. The clusters with the highest Data traffic have cells of more than 1,000.

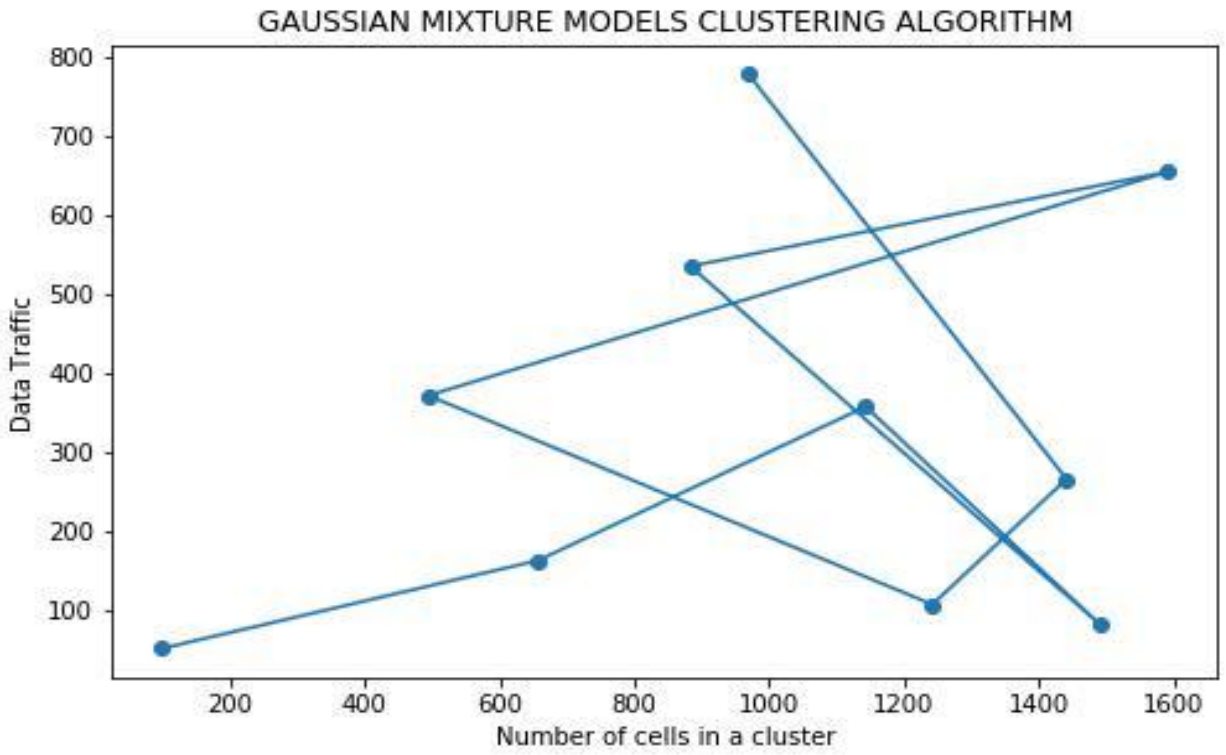

Figure 4.3: Gaussian Mixture Model Clustering Algorithm

Figure 4.4 is a comparison of the 3 clustering algorithms used in this work. From the graph it can be seen that the Gaussian Mixture Models has a cluster with the lowest mean data traffic while the K-means has a cluster with the lowest number of cells. The Hierarchical clustering algorithm has the highest mean data traffic. Across the 3 clustering algorithms, most clusters have cells in the range of 500-1600.

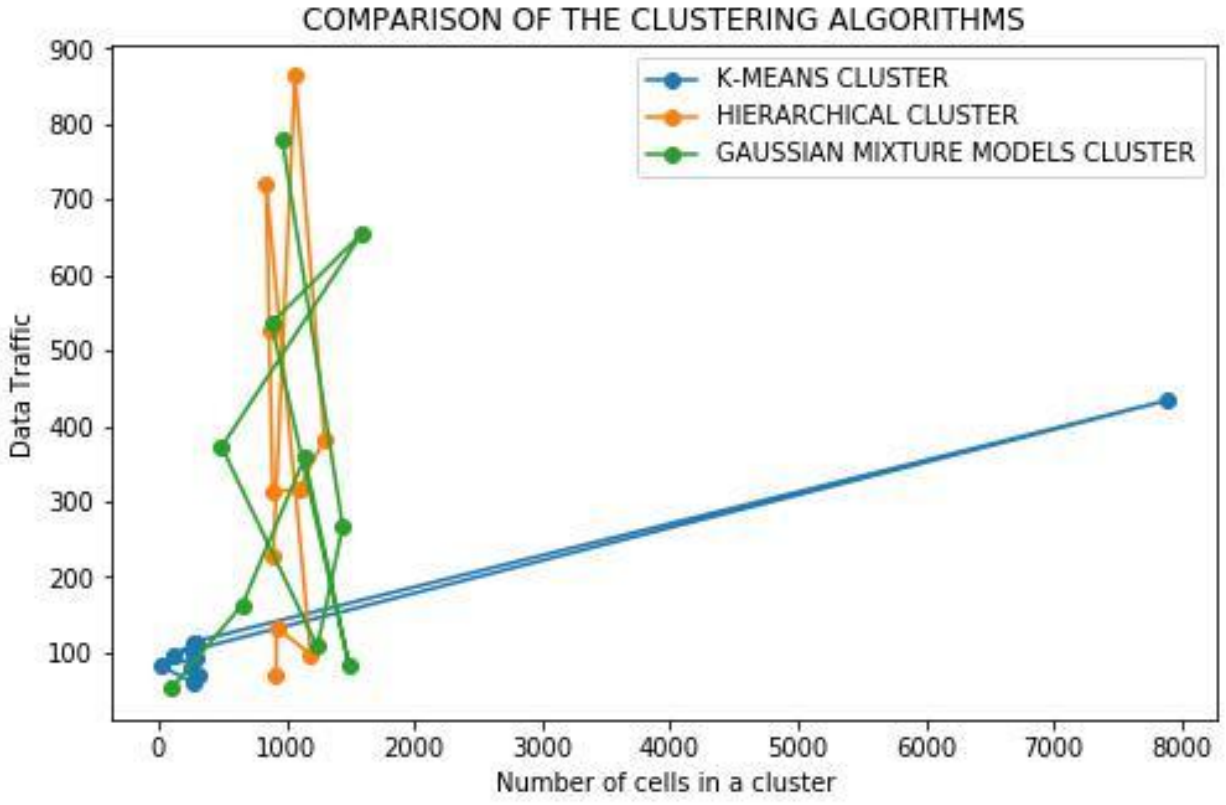

Figure 4.4: A Comparison of the Cluster Algorithms

Figures 4.1, 4.2 and 4.3 show the number of cells in a cluster and its mean data traffic for each cluster algorithm. Table 4.1. shows the results of the 3 clustering algorithms used in this work. From the Table 4.1 it is clear how each clustering algorithm performed across the 10 clusters and easy comparisons can be made.

Table 4.1: Results of Cluster Algorithms

CLUSTER	CLUSTER ALGORITHMS					
	K-MEANS		HIERARCHICAL		GAUSSIAN MIXTURE MODELS	
	NUMBER OF CELLS	MEAN OF INTERNET TRAFFIC	NUMBER OF CELLS	MEAN OF INTERNET TRAFFIC	NUMBER OF CELLS	MEAN OF INTERNET TRAFFIC
0	278	99.82	897	226.82	969	778.88
1	304	69.17	1068	863.92	1441	266.54
2	254	81.40	1298	383.33	1242	107.91
3	276	114.26	1099	316.25	494	372.35
4	7893	434.07	895	313.21	1591	655.26
5	119	96.97	865	528.19	883	535.81
6	293	113.47	843	719.93	1491	82.49
7	294	92.38	1192	96.76	1143	358.17
8	27	81.81	923	133.66	657	163.65
9	272	59.68	920	68.98	99	52.24

Table 4.2 shows the results for the various clustering algorithms and how they performed based on capacity utility and cost of deployment, 10 clusters was formed for each clustering algorithm and the BBU capacity is 8 for each cluster. From the Table 4.2, it is seen that

there was a reduction in capacity utility and cost after clustering was done, it is also noticeable that hierarchical clustering algorithm performed best in comparison to the other used clustering algorithms. The cost of deployment reduced from 182 to 80 which means that the area covered by 182 RRHs can be effectively covered by 80 RRHs.

Table 4.2: Results of Capacity Utility and Cost of Deployment

CLUSTERING ALGORITHM	CAPACITY UTILITY	COST OF DEPLOYMENT
BEFORE CLUSTERING	0.63	182
K-MEANS	0.0044	80
HIERARCHICAL	0.0012	80
GAUSSIAN MIXTURE MODEL	0.0035	80

4.2 Cluster Calculations for Capacity Utility and Cost of Deployment

i. Before Clustering

$$= \frac{\text{mean f(c)} - \ln^{\text{mean f(c)}}}{|R|} = \left(\frac{361}{182}^{.75}\right) - \ln\left(\frac{361}{182}^{.75}\right)$$

$$= 0.63$$

$$\text{Cost}(P) = \sum_{k=1}^{k} |\{C_k\}|$$

$$= 182$$

ii. K-Means Clustering

$$\frac{\text{mean f(c)} - \ln^{\text{mean f(c)}}}{|R|}$$

$$\text{cluster } 0 = \left(\frac{99.82}{8}\right) - \ln\left(\frac{99.82}{8}\right)$$

$$\text{cluster } 1 = \left(\frac{69.17}{8}\right) - \ln\left(69_{8.17}\right) = 0.0017$$

$$\text{cluster } 2 = \left(\frac{81.40}{8}\right) - \ln\left(81_{8.40}\right) = 0.0095$$

$$= \frac{114.26}{} - \ln\left(114.26\right) = 0.0046$$

$$= \frac{434.07}{} - \ln\left(434.07\right) = 0.0008$$

$$\text{cluster } 5 = \left(\frac{96.97}{8}\right) - \ln\left(96_{8.97}\right) = 0.00000012$$

$$= \frac{113.47}{} - \ln\left(113.47\right) = 0.0020$$

$$\text{cluster } 7 = \left(\frac{92.38}{8}\right) - \ln\left(92_{8.38}\right) = 0.00088$$

$$\text{cluster } 8 = \left(\frac{81.81}{8}\right) - \ln\left(81_{8.81}\right) = 0.0025$$

$$\text{cluster } 9 = \left(\frac{69.17}{8}\right) - \ln\left(69_{8.17}\right) = 0.0045$$

$$\text{CAPACITY UTILITY} = \text{mean}_{ck}\, U(C_k) = 0.0176$$

$$= 0.0044$$

$$\text{Cost}(P) = \sum_{k=1}^{K} |\{C_k\}|$$

$$= 8*10$$
$$= 80$$

iii. Hierarchical Clustering

$$\text{cluster } 0 = \left(\frac{\text{mean f(c)}}{|B|} - \ln\frac{\text{mean f(c)}}{|B|}\right) = \left(\frac{226_{8}.82}{0.000014} - \ln\left(\frac{226_{8}.82}{0.000014}\right)\right)$$

$$\left(\frac{863.92}{0.00000000030} - \ln\left(\frac{863.92}{0.00000000030}\right)\right)$$

$$\left(\frac{383.33}{0.00000031} - \ln\left(\frac{383.33}{0.00000031}\right)\right)$$

$$\left(\frac{316.25}{0.0000013} - \ln\left(\frac{316.25}{0.0000013}\right)\right)$$

$$\left(\frac{313.21}{0.0000014} - \ln\left(\frac{313.21}{0.0000014}\right)\right)$$

$$\left(\frac{528.19}{0.0000000024} - \ln\left(\frac{528.19}{0.0000000024}\right)\right)$$

$$\left(\frac{719.93}{0.0000000016} - \ln\left(\frac{719.93}{0.0000000016}\right)\right)$$

$$\text{cluster } 7 = \left(\frac{96_{8}.76}{0.0020} - \ln\left(\frac{96_{8}.76}{0.0020}\right)\right)$$

$$\frac{133.66}{} - \ln(\frac{133.66}{})$$

cluster 9 = $(\frac{68}{8}.\frac{98}{}) - \ln(\frac{68}{8}.98)$

CAPACITY UTILITY = $\text{mean}_{ck}\ U(C_k)$

$$= 0.0012$$

$\text{Cost}(P) = \sum_{k=1}^{K} |\{C_k\}|$

$$= 8*10$$
$$= 80$$

iv. Gaussian Mixture Models Clustering.

$$\frac{\text{mean f(c)} - \ln^{\text{mean f(c)}}}{}$$

$$\frac{778.88}{} - \ln(\frac{778.88}{})$$

$$\frac{266.54}{} - \ln(\frac{266.54}{})$$

$$\frac{107.91}{} - \ln(\frac{107.91}{})$$

$$\frac{372.35}{} - \ln(\frac{372.35}{})$$

cluster 4 = $(\frac{655}{8}.\frac{26}{}) - \ln(\frac{655}{8}.26)$

cluster 5 = $(\frac{535}{8}.\frac{81}{}) - \ln(\frac{535}{8}.81)$

$$= 0.000000021$$

$$\text{cluster } 6 = \left(\frac{82}{8}^{.49}\right) - \ln\left(82_{8}{}^{.49}\right)$$

$$= 0.0042 \quad \frac{358.17}{} - \ln\left(\frac{358.17}{}\right)$$

$$= 0.00000053 \quad \frac{163.65}{} - \ln\left(\frac{163.65}{}\right)$$

$$= 0.00011$$

$$\text{cluster } 9 = \left(\frac{52}{8}^{.24}\right) - \ln\left(52_{8}{}^{.24}\right)$$

$$= 0.0296$$

$$\text{CAPACITY UTILITY} = \text{mean}_{ck}\, U(C_k)$$

$$= 0.0035$$

$$\text{Cost}(P) = \sum_{k=1}^{K} |\{C_k\}|$$

$$= 8*10$$

$$= 80$$

4.3 Result of Deep Learning

Figure 4.5 shows the LSTM prediction of Milan Traffic Volume and the mean absolute error (MAE) which shows that the accuracy of the prediction model is 5.75×10^{-4}. The MAE is an automatically generated output in the python code used for the prediction. Due to its low MAE, it validates its capacity for accurate prediction, and this will help mobile network operators in proper dimensioning of network resources due to a pre-knowledge of the data traffic volume.

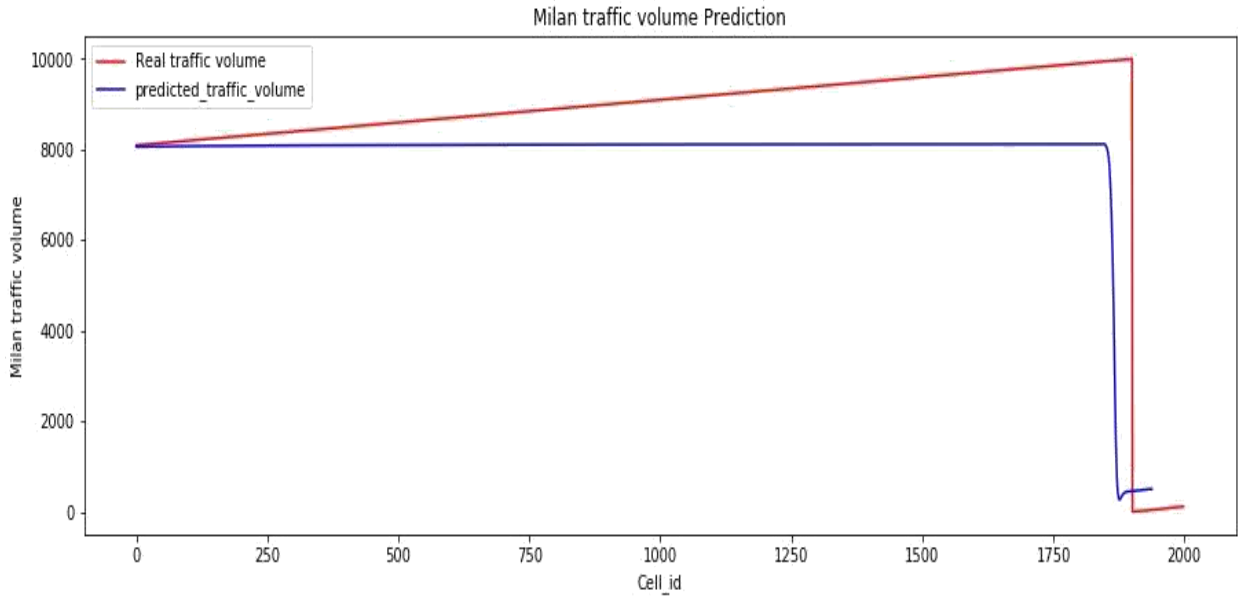

Figure 4.5: Milan Traffic Volume Prediction

The Mean Absolute Error (MAE) = 5.75×10^{-4}

Chen *et al*. (2018) in their work had a Mean Absolute error of 0.074 and were able to reduce the number of base stations from 182 to 88. It is important to note that in their work, the type of feature scaler used was not stated and the number of LSTM stack used are not the same with this work but for comparison's sake, Chen *et al*. (2018) was the closest in terms of objectives. A comparison of this work and Chen *et al*. (2018) is presented in Table 4.3.

Table 4.3: Comparison of Methodology

Authors	Mean Absolute Error (MAE)	Cost of Deployment
Chen *et al*. (2018)	0.074	88
This Work	5.75×10^{-4}	80

CHAPTER FIVE

5.0 CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

As a result of increase in personal devices and data-hungry mobile applications, the demand for an efficient and always available high data rate radio access network has increased. The radio access network part of a base station is one of the most important and about 80% of the capital expenditure is spent on it. Mobile network Operators are looking for ways to cut down this cost.

Cloud Radio Access Network (C-RAN) is an architecture to solve the problem of increase in CAPEX and OPEX. In C-RAN there is the centralization of baseband processing of statistically varying traffic loads from different base stations, these loads are multiplexed and the processing resources are also shared. These multiplexed loads, lead to considerable gains such as reduction in cost and energy consumption.

This work used K-means, Hierarchical and Gaussian Mixture Models clustering algorithms to develop effective clustering techniques for base station clustering to achieve multiplexing gains such as improved capacity utility and reduced cost in C-RAN. The K-Means had a capacity utility of 0.0044, Hierarchical had 0.0012 and the gaussian mixture models had 0.0035. From these results the hierarchical has the best capacity utility out of these 3 algorithms.

Mobile network operators over-dimension network resources due to lack of pre-knowledge of the data traffic volume. Network operators are able to provide network resources proactively if they are able to predict changes in network load before it happens.

An LSTM prediction algorithm was used in this work to provide pre-knowledge to mobile network operators for proper dimensioning of the network. The LSTM used was a 4-stack layer and a Mean Absolute Error of 5.75×10^{-4} was obtained. This low MAE shows that the predicted data traffic is very close to the actual data traffic.

5.2 Recommendations

In the future this work can be enhanced by the use of different prediction algorithms and clustering algorithms. The use of different datasets to verify the performance of the prediction model and clustering algorithms is also advised. The creation of open-source databases containing mobile network datasets for research should be looked into. The multiplexing gains of improved capacity utility and cost of deployment reduction achieved in this work could lead to the exploration of other multiplexing gains such as reduced energy consumption.

5.3 Contribution to Knowledge

This work has contributed to knowledge as follows:

1. Comparative analysis on clustering algorithms to determine the better C-RAN clustering algorithm based on the following evaluation metrics: Capacity Utility and Cost of Deployment. This was done using the K-Means, Hierarchical and Gaussian Mixture Model clustering algorithm, with the Hierarchical clustering algorithm outperforming the others.
2. Prediction of Data Traffic Volume to enable proper dimensioning of network resources. This was done using the deep learning LSTM model.

REFERENCES

Alimi, I. A., Teixeira, A. L., & Monteiro, P. P. (2018). Toward an Efficient C-RAN Optical Fronthaul for the Future Networks: A Tutorial on Technologies, Requirements, Challenges, and Solutions. *IEEE Communications Surveys and Tutorials*, *20*(1), 708–769. https://doi.org/10.1109/COMST.2017.2773462

Bhamare, D., Erbad, A., Jain, R., Zolanvari, M., & Samaka, M. (2018). Efficient virtual network function placement strategies for Cloud Radio Access Networks. *Computer Communications*, *127*, 50–60. https://doi.org/10.1016/j.comcom.2018.05.004

Bhaumik, S., Chandrabose, S. P., Jataprolu, M. K., Kumar, G., Muralidhar, A., Polakos, P., Srinivasan, V., Woo, T., & Berkeley, U. C. (2012). *CloudIQ : A Framework for Processing Base Stations in a Data Center*. 125–136.

Bihnam, A. N. (2021). *Performance evaluation of 5g cloud radio access networks using new fading channel models*. *April*.

Chaudhary, J. K., Bartelt, J., & Fettweis, G. (2017). Statistical multiplexing in fronthaul-constrained massive MIMO. *EuCNC 2017 - European Conference on Networks and Communications*. https://doi.org/10.1109/EuCNC.2017.7980774

Checko, A., Holm, H., & Christiansen, H. (2014, May). Optimizing small cell deployment by the use of C-RANs. In European Wireless 2014; 20th European Wireless Conference (pp. 1-6). VDE.

Chen, C. (2011). C-RAN: The road towards green radio access network. *White Paper*, *0*. http://ss-mcsp.riit.tsinghua.edu.cn/cran/C-RAN ChinaCOM-2012-Aug-v4.pdf

Chen, L., Yang, D., Zhang, D., Wang, C., Li, J., & Nguyen, T. M. T. (2018). Deep mobile traffic forecast and complementary base station clustering for C-RAN optimization. *Journal of Network and Computer Applications*, *121*, 59–69. https://doi.org/10.1016/j.jnca.2018.07.015

Chen, M., Zhang, Y., Hu, L., & Taleb, T. (2015). *Cloud-based Wireless Network : Virtualized , Reconfigurable , Smart Wireless Network to Enable 5G Technologies*. https://doi.org/10.1007/s11036-015-0590-7

Common Public Radio Interface. (2015). CPRI Specification V7.0. *Standard Document Specification*, *0*, 128.

Greff, K., Srivastava, R. K., Koutn, J., & Steunebrink, B. R. (n.d.). *LSTM : A Search Space Odyssey*. 1–12. https://doi.org/10.1109/TNNLS.2016.2582924

Harris, C. R., Millman, K. J., Walt, S. J. Van Der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., & Kerkwijk, M. H. Van. (2020). Array programming with NumPy. *Nature*, *585*(June), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hochreiter, S. (1997). *Long Short-Term Memory*. *1780*, 1735–1780.

Hossain, M. F., Mahin, A. U., Debnath, T., Mosharrof, F. B., & Islam, K. Z. (2019). Recent research in cloud radio access network (C-RAN) for 5G cellular systems - A survey. *Journal of Network and Computer Applications*, *139*(April), 31–48. https://doi.org/10.1016/j.jnca.2019.04.019

Kalor, A. E., Agurto, M. I., Pratas, N. K., Nielsen, J. J., & Popovski, P. (2017). Statistical multiplexing of computations in C-RAN with tradeoffs in latency and energy. *2017 IEEE International Conference on Communications Workshops, ICC Workshops 2017*, *1*, 772–777. https://doi.org/10.1109/ICCW.2017.7962752

Ketkar, N. (2017). Deep Learning with Python. *Deep Learning with Python*, 95–109. https://doi.org/10.1007/978-1-4842-2766-4

Längkvist, M., Karlsson, L., & Loutfi, A. (2014). *A review of unsupervised feature learning and deep learning for time-series modeling q*. *42*, 11–24. https://doi.org/10.1016/j.patrec.2014.01.008

Laptev, N., Yosinski, J., Erran, L., & Slawek, L. (2017). *Time-series Extreme Event Forecasting with Neural Networks at Uber*.

Lehrmann, H., Stübert, M., Checko, A., Christiansen, H. L., & Yan, Y. (2014). *Cloud RAN for Mobile Networks - a Technology Overview*. https://doi.org/10.1109/COMST.2014.2355255

Lin, Y., Shao, L., Zhu, Z., Wang, Q., & Sabhikhi, R. K. (2010). Wireless network cloud: Architecture and system requirements. *IBM Journal of Research and Development*, *54*(1), 4:1-4:12. https://doi.org/10.1147/jrd.2009.2037680

Liu, J., Zhou, S., Gong, J., Niu, Z., & Xu, S. (2016). Statistical multiplexing gain analysis of heterogeneous virtual base station pools in cloud radio access networks. *IEEE Transactions on Wireless Communications*, *15*(8), 5681–5694. https://doi.org/10.1109/TWC.2016.2567383

Lorca, J., & Cucala, L. (2013). *Lossless compression technique for the fronthaul of LTE/LTE-advanced cloud-RAN architectures*. 1–9. https://doi.org/10.1109/wowmom.2013.6583374

Mcclean, S., & Morrow, P. (2020). *Comparison of Analogue and Digital Fronthaul for 5G MIMO Signals*. *June*. https://doi.org/10.1109/ICC40277.2020.9148787

Peng, H., William, T., Fitzgerald, E., & Kihl, M. (2021). *Is Cloud RAN a Feasible Option for Industrial Communication Network ? 17*(2), 97–105.

Peng, M., Wang, C., Lau, V., & Poor, H. V. (2015). Fronthaul-constrained cloud radio access networks: Insights and challenges. *IEEE Wireless Communications*, *22*(2), 152–160. https://doi.org/10.1109/MWC.2015.7096298

Pölsterl, S. (2020). *scikit-survival : A Library for Time-to-Event Analysis Built on Top of scikit-learn*. *21*, 1–6.

Pompili, D., Hajisami, A., & Viswanathan, H. (2015). Dynamic provisioning and allocation in Cloud Radio Access Networks (C-RANs). *Ad Hoc Networks*, *30*, 128–143. https://doi.org/10.1016/j.adhoc.2015.02.006

Rajabi, A., Eskandari, M., Jabbari, M., Li, L., & Zhang, J. (2019). *A comparative study of clustering techniques for electrical load pattern segmentation Symbolic Aggregate approXimation Time of Use*. *August*. https://doi.org/10.1016/j.rser.2019.109628

Ramalho, L., Fonseca, M. N., Klautau, A., Lu, C., Berg, M., Trojer, E., & Höst, S. (2017). An LPC-based fronthaul compression scheme. *IEEE Communications Letters*, *21*(2), 318–321. https://doi.org/10.1109/LCOMM.2016.2624296

Rish, I., Watson, T. J., & Heights, Y. (2001). IBM Research Report. *Science*, *22230*(October), 1–10.

Sapre, A., & Vartak, S. (2020). *Scientific Computing and Data Analysis using NumPy and Pandas*. 1334–1346.

Shehata, M., Elbanna, A., Musumeci, F., & Tornatore, M. (2017). *C-RAN Baseband Pooling : Cost Model and Multiplexing Gain Analysis*. 3–6.

Sial, A. H., Yahya, S., & Rashdi, S. (2021). *Comparative Analysis of Data Visualization Libraries Matplotlib and Seaborn in Python*. 277–281.

Sigwele, T., Pillai, P., & Hu, Y. F. (2015). *iTREE : Intelligent Traffic and Resource Elastic Energy Scheme for Cloud-RAN*. 282–288. https://doi.org/10.1109/FiCloud.2015.104

Sridharan, R. (2017). *Gaussian mixture models and the EM Algorithm (Lecture Notes - MIT CSAIL)*. 11. https://people.csail.mit.edu/rameshvs/content/gmm-em.pdf

Subramaniyan, M., Skoogh, A., Sheikh, A., Bokrantz, J., Johansson, B., & Roser, C. (2020). A generic hierarchical clustering approach for detecting bottlenecks in manufacturing. *Journal of Manufacturing Systems*, *55*(February), 143–158. https://doi.org/10.1016/j.jmsy.2020.02.011

Taleb, H., Helou, M. El, Lahoud, S., Khawam, K., & Martin, S. (2018a). An Efficient

Heuristic for Joint User Association and RRH Clustering in Cloud Radio Access Networks. *2018 25th International Conference on Telecommunications, ICT 2018*, 8–14. https://doi.org/10.1109/ICT.2018.8464852

Taleb, H., Helou, M. El, Lahoud, S., Khawam, K., & Martin, S. (2018b). Multi-Objective Optimization for RRH Clustering in Cloud Radio Access Networks. *2018 International Conference on Computer and Applications, ICCA 2018*, 85–89. https://doi.org/10.1109/COMAPP.2018.8460446

Visual, C., & Index, N. (2017). *Cisco Visual Networking Index: Global Mobile Data About the Cisco Visual Networking Index Q. Why did Cisco develop the Cisco Visual Networking Index ™ (Cisco VNI ™) Forecast?* 2017–2022. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-forecast-qa.pdf

Wang, J., Yu, L., Lai, K. R., & Zhang, X. (2016). *Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model*. 225–230.

Wang, X., Zhao, Y., & Pourpanah, F. (2020). Recent advances in deep learning. *International Journal of Machine Learning and Cybernetics*, *11*(4), 747–750. https://doi.org/10.1007/s13042-020-01096-5

Werthmann, T., Grob-Lipski, H., & Proebster, M. (2013). Multiplexing gains achieved in pools of baseband computation units in 4G cellular networks. *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, 3328–3333. https://doi.org/10.1109/PIMRC.2013.6666722

Wu, J., & Ghosal, D. (2016). Cost-constrained delay minimization of C-RAN and its multiplexing gain. *2016 IEEE International Conference on Communications, ICC 2016*, *1*. https://doi.org/10.1109/ICC.2016.7511576

Xia, S., Peng, D., Meng, D., Zhang, C., Wang, G., Giem, E., & Wei, W. (2020). *Ball k - means : Fast Adaptive Clustering with No Bounds*. *8828*(c), 1–13. https://doi.org/10.1109/TPAMI.2020.3008694

Ye Li, G., Xiong, C., Yang, C., Zhang, S., Chen, Y., & Xu, S. (2011). Energy-Efficient Wireless Communications: Tutorial, Survey, and Open Issues. *IEEE Wireless Communications*, *December*, 28–35.

Zhang, C., Patras, P., & Haddadi, H. (2019). Deep Learning in Mobile and Wireless Networking: A Survey. *IEEE Communications Surveys and Tutorials*, *21*(3), 2224–2287. https://doi.org/10.1109/COMST.2019.2904897

APPPENDIX A

Python Code for Clustering Algorithms

i. K-Means Clustering Python Code

```
# Importing Libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import scipy.cluster.hierarchy as sch



dataset = pd.read_csv('C:\\Users\\Nelson\\Desktop\\milan csv\\milan training\\sms-call-internet-mi-2013-11-01.csv')

# Making another data for clustering

clus_data = non_null_data.iloc[:10000, [1, 7]].copy()

clus_data.head(10)

# Saving the clus_data dataframe into a csv file

clus_data.to_csv('cluster_data.csv')

# Fitting K-Means to the dataset

no_cluster_k_means = 10
```

```
kmeans = KMeans(n_clusters = no_cluster_k_means, init = 'k-means++', random_state =
42)

y_kmeans = kmeans.fit_predict(clus_data)

# This code will show you which column correspond to which cluster

clus_corresonding_data_kmeans =
clus_data.join(pd.DataFrame(y_kmeans)).fillna(method='ffill')

clus_corresonding_data_kmeans.head(100)

# Save this dataframe

clus_corresonding_data_kmeans.to_csv('clus_corresonding_data_kmeans.csv')

#plotting the cluster graph

plt.scatter(clus_data.iloc[:,0],clus_data.iloc[:,1], c=y_kmeans, cmap='rainbow')

plt.title('K-MEANS CLUSTERING ALGORITHM')

plt.xlabel('Number of cell in a cluster')

plt.ylabel('Data Traffic')

print(f'There are {no_cluster_k_means} clusters')

# this code will filter and give you the specific cluster data

from IPython.core.display import HTML

cluster = 0 # You can change this number
```

```
display(HTML(clus_corresonding_data_kmeans.loc[clus_corresonding_data_kmeans[0] ==
cluster].to_html()))
```

ii. Hierarchical Clustering Python Code

```
from scipy.cluster.hierarchy import dendrogram, linkage
from matplotlib import pyplot as plt linked =
linkage(clus_data, 'single')
# Making Dendrogram for clustering algorithm
dendrogram = sch.dendrogram(sch.linkage(clus_data, method = 'ward'))
plt.title('Dendrogram')
plt.show()
# Fitting Hierarchical Clustering to the dataset
no_cluster_hc = 10 # You can change this for experimentation
from sklearn.cluster import AgglomerativeClustering
hc = AgglomerativeClustering(n_clusters = no_cluster_hc, affinity = 'euclidean', linkage =
'ward')
y_hc = hc.fit_predict(clus_data)
# This code will show you which column correspond to which cluster
clus_corresonding_data_hierarchical =
clus_data.join(pd.DataFrame(y_hc)).fillna(method='ffill')
```

```
clus_corresonding_data_hierarchical.head(100)

# Save this datframe

clus_corresonding_data_hierarchical.to_csv('clus_corresonding_data_hierarchical.csv')

# Plot the clustering

plt.scatter(clus_data.iloc[:,1],clus_data.iloc[:,2], c=y_hc, cmap='rainbow')

plt.title('HIERARCHICAL CLUSTERING ALGORITHM')

plt.xlabel('Number Of Cells In A Cluster')

plt.ylabel('Data Traffic')

print(f'There are {no_cluster_hc} clusters')

# this code will filter and give you the specific cluster data

from IPython.core.display import HTML

cluster = 9 # You can change this number

specific_cluster =

display(HTML(clus_corresonding_data_hierarchical.loc[clus_corresonding_data_hierarchi

cal[0] == cluster].to_html()))
```

iii. Gaussian Mixture Models Clustering Python Code

```
%matplotlib inline

import matplotlib.pyplot as plt

import numpy as np
```

```
from sklearn.mixture import GaussianMixture

clus_data = pd.read_csv('C:\\Users\\Nelson\\Desktop\\milan csv\\cluster_data.csv')

gmm = GaussianMixture(n_components=10).fit(clus_data)

labels = gmm.predict(clus_data)

plt.scatter(clus_data.iloc[:, 1], clus_data.iloc[:, 2], c=labels, s=40, cmap='viridis');

plt.title('GAUSSIAN MIXTURE MODELS CLUSTERING ALGORITHM')

plt.xlabel('Number of cells in a cluster')

plt.ylabel('Data Traffic')

# This code will show you which column correspond to which cluster

clus_corresonding_data_GMM =

clus_data.join(pd.DataFrame(labels)).fillna(method='ffill')

clus_corresonding_data_GMM.head(1000)

clus_corresonding_data_GMM.to_csv('clus_corresonding_data_GMM.csv')

# this code will filter and give you the specific cluster data

from IPython.core.display import HTML

cluster = 9 # You can change this number

display(HTML(clus_corresonding_data_GMM.loc[clus_corresonding_data_GMM[0] ==

cluster].to_html()))
```

APPENDIX B

Python Code for LSTM Prediction

```
# Importing the libraries

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

# Importing the training set

dataset_train = pd.read_csv('C:\\Users\\Nelson\\Desktop\\milan csv\\milan cellid and
internet.csv')

training_set = dataset_train.iloc[:8000, 1:2].values



from sklearn.preprocessing import MinMaxScaler

sc = MinMaxScaler(feature_range = (0, 1))

training_set_scaled = sc.fit_transform(training_set)

# Creating a data structure with 60 timesteps and 1 output

X_train = []

y_train = []

for i in range(60, 7999):

    X_train.append(training_set_scaled[i-60:i, 0])
```

```
y_train.append(training_set_scaled[i, 0])

X_train, y_train = np.array(X_train), np.array(y_train)

# Reshaping

X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))

#  Building the RNN

# Importing the Keras libraries and packages

from keras.models import Sequential

from keras.layers import Dense

from keras.layers import LSTM

from keras.layers import Dropout

# Initialising the RNN

regressor = Sequential()

# Adding the first LSTM layer and some Dropout regularisation

regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1],

1)))

regressor.add(Dropout(0.2))

# Adding a second LSTM layer and some Dropout regularisation

regressor.add(LSTM(units = 50, return_sequences = True))

regressor.add(Dropout(0.2))
```

```
# Adding a third LSTM layer and some Dropout regularisation

regressor.add(LSTM(units = 50, return_sequences = True))

regressor.add(Dropout(0.2))

# Adding a fourth LSTM layer and some Dropout regularisation

regressor.add(LSTM(units = 50)) regressor.add(Dropout(0.2))


# Adding the output layer

regressor.add(Dense(units = 1))


regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')

# Fitting the RNN to the Training set

regressor.fit(X_train, y_train, epochs = 100, batch_size = 64)

# Making the predictions and visualising the results

# Getting the real traffic volume of the city of milan

dataset_test = pd.read_csv('C:\\Users\\Nelson\\Desktop\\milan csv\\milan cellid and
internet.csv')

real_traffic_volume = dataset_test.iloc[8001:10000, 1:2].values
```

```
# Getting the predicted traffic volume of the city of milan

inputs = real_traffic_volume

inputs  =  inputs.reshape(-1,1)

inputs  =  sc.transform(inputs)

X_test = []

for i in range(60, 1999):

   X_test.append(inputs[i-60:i, 0])

X_test = np.array(X_test)

X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))

predicted_traffic_volume = regressor.predict(X_test) predicted_traffic_volume

= sc.inverse_transform(predicted_traffic_volume)

# Visualising the results

plt.figure(figsize = (15, 5) )

plt.plot(real_traffic_volume, color = 'red', label = 'Real traffic volume')

plt.plot(predicted_traffic_volume, color = 'blue', label = 'predicted_traffic_volume')

plt.title('Milan traffic volume Prediction')

plt.xlabel('Cell_id')

plt.ylabel('Milan traffic volume')

plt.legend()

plt.show()
```

APPENDIX C

Publication

Umasabor, N., Salihu, B., Salawu, N., & Zubair, S. (2020, March). Multiplexing Gains Through Clustering in Cloud Radio Access Network. In 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS) (pp. 1-6). IEEE.