Real-time Congestion Detection using Public Transport Data

Student Name: Yogesh Pandey

IIIT-D-MTech-CS-GEN-MT18140

Dec, 2019

Indraprastha Institute of Information Technology
New Delhi

Thesis Committee
Dr. Vikram Goyal (Advisor)
Dr. Debajyoti Bera
Prof. S.K. Gupta

Submitted in partial fulfillment of the requirements
for the Degree of M.Tech. in Computer Science,
in General Category

Keywords: Congestion Index(CI), bus transit system, Detection, GPS, Congestion, Spatio-temporal.

Certificate

This is to certify that the thesis titled **"Real-time Congestion Detection using Public Transport Data"** submitted by **Yogesh Pandey** for the partial fulfillment of the requirements for the degree of *Master of Technology* in *Computer Science & Engineering* is a record of the bonafide work carried out by him under my guidance and supervision at Indraprastha Institute of Information Technology, Delhi. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree. This work has not been submitted anywhere else for the reward of any other degree.

Dr. Vikram Goyal

Department of Computer Science

Indraprastha Institute of Information Technology, New Delhi (IIIT Delhi)

Abstract

Congestion is one of the biggest problems which affects life quality and has an impact on social and economic conditions. Tackling traffic congestion has always been a challenge. The situation is more aggravated for developing countries, like India, due to their huge population and over-utilization of basic resources. Existing solutions in terms of policies like the construction of new flyovers, underpasses, widening of roads have failed miserably. Prevalent technical solutions for congestion detection like deployment of the camera, RFID and other sensors are quite popular but they are more popular in developed countries. Developing countries are not much readily involved in such strategies as they require economic contributions and maintenance. Also web mapping service providers like Google, Bing, HERE uses crowdsourced information of mobile devices through their applications. Such kind of private data is not available to the government agencies which they can utilize to improve their transport system and solve congestion problems. So we study the effectiveness of congestion detection if only public transport data is available. We investigate the utility of the real-time bus Spatio-temporal data, which is sparse and has missing values for the task of congestion detection. Such a system if works with good accuracy, it would make government authorities not to depend on private players. We provide a real-time congestion detection mechanism that exploits GPS sensors installed on Delhi's DIMTS cluster buses to provide fast & reliable congestion status. We compare multiple strategies and observe 70% f1 score and 80% recall at best by a simple statistical-based method. We also analyze this data for the application of hotspot detection and identifying popular bus stops.

Acknowledgments

Foremost I would like to give sincere thanks to my advisor Dr. Vikram Goyal at IIIT Delhi for providing me the opportunity to work on this thesis which involves real-world application. I would also like to provide him my sincere gratitude for his valuable guidance, suggestions, continuous support and encouragement. The door to Dr. Goyal's office was always open whenever I faced problems and he guided me in the right direction. I would also like to thank esteemed thesis committee members Dr. Debajyoti Bera and Prof. S.K. Gupta for evaluating my work. I would like to thank my parents for giving me birth in the first place and giving their blessings & constant support throughout my life. I would also like to thank Nikhil Gola for his valuable comments and discussions. Additional thanks to Ashish Jain & Sarosh Hasan for providing me access to their dashboard whose images are used in Chapter 5. At last, I would like to thank my friends and college mates for their immense support.

Author

Yogesh Pandey

Contents

List of Figures

Chapter 1

Introduction and Motivation

Congestion is one of the biggest problems which affects life quality and has an impact on social and economic conditions. Tackling traffic congestion has always been a challenge. The situation is more aggravated for developing countries, like India, due to their huge population and over-utilization of basic resources. Existing solutions in terms of policies like the construction of new flyovers, underpasses, widening of roads have failed miserably [15]. Such kind of solutions require time in terms of planning & implementation and their implementation itself results in congestion as it leads to road and traffic blockage.

Prevalent technical solutions for congestion detection like deployment of the camera, RFID and other sensors are quite popular but they are more popular in developed countries. Developing countries are not much readily involved in such strategies as they require economic contributions and maintenance. Also web mapping service providers like Google, Bing, HERE uses crowdsourced information of mobile devices through their applications. They take people's private data like location & deduce the traffic status. Such kind of private data is not available to the government agencies hence it makes them dependent on private players. So we need some better & accurate alternate solutions, which they can utilize to improve their transport system and solve congestion problems.

Currently, Delhi public transportation agency DIMTS has a web app that tracks the location of buses in the desired route. Delhi government has also come with an app "Poochh-O" which helps in selecting buses for your desired destination. It can tell which buses are approaching the nearest bus stop, get bus schedules, search bus stops. The bus tracking via the DIMTS app and poochh-0 app can be seen in Figure 1.1 and Figure 1.2 respectively.

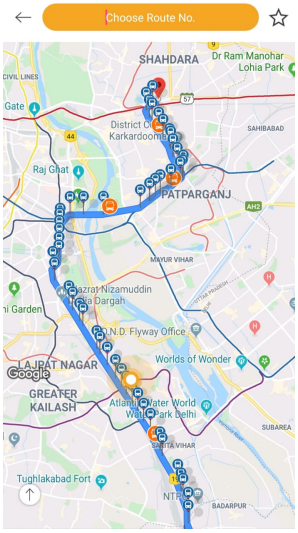

Figure 1.1: poochh-O app

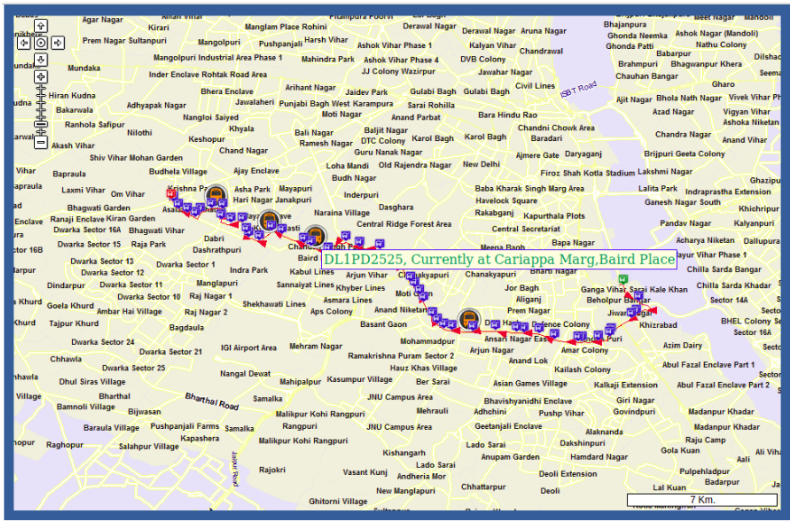

Figure 1.2: DIMTS webapp

Both these app provides live location but do not consider the prevalent congestion scenario. They are bounded by the time table data which is predefined for each bus. None of the apps shows whether the bus is in congestion or not. They just provide the live location which is less useful for the end-user and even for the authorities. So this is high time that we need some mechanism which can tell us whether the bus is in congestion or not and whether the bus would be able to reach the desired location at a particular time or not. Congestion detection can help in better functioning of these prevalent applications.

In this regard, we present a methodology for congestion detection where we take help from GPS sensors deployed on public bus transport. The problem we address in this thesis is dealing with the realtime Spatio-temporal data of public transport through which we can detect congestion. The main problem with this data is that we do not have consistency due to some external and internal factors which result in missing data values. The data is also sparse as we have a limited number of buses for this task as compared to the number of vehicles that actually run on the road. Government agencies mainly rely on this data for analysis purposes, otherwise, they have to pay a healthy cost to private firms for the same task. The proposed framework will also help them to deduce other factors like congestion prone areas, region-wise bus frequency, etc. Experimental results on our methodology give 70% f1-score and 80% recall at best when compared with the ground truth from HERE maps jam factor.

Chapter 2

Research Aim & Thesis outline

2.1 Research Aim

In the thesis, the aim is to address the following issues:

- Get road wise congestion information in real-time so that we can tell whether the bus is in congestion or not.
- Get congestion status with the least inputs features, the only thing we get as input is the live GPS location of DIMTS buses. We do not get any other information like crowdsourced data from mobile or information like traffic images from cameras etc as used by existing techniques.
- We then compare results got from our methodology with the HERE maps which is one of the popular maps services used by AUDI, BMW, Amazon, etc. A comparison with such service would describe the efficiency of our algorithm.
- We also list popular bus stops, hotspots, and congested regions.

2.2 Thesis Outline

The thesis is organized as follows, In the next chapter, we first discuss the current situation of the Delhi transport system, the subsequent chapter discusses previous work in the area of congestion detection. In chapter 6, We discuss the proposed methodology, its architecture and related components. In chapter 7, Experimental results obtained from the real-time detection of congestion are discussed. In the same chapter, we list out the hotspots, popular bus stops and jam regions of Delhi. Finally, in the last chapter, conclusions made from the study along with the future work is presented.

Chapter 3

Current Delhi transport system

Delhi aka. National Capital Territory of Delhi (NCT) is a union territory with its own legislature and government. It is the point of convergence of social, monetary and political exercises of the country. It also acts as a focal point for trade and commerce, thus being the largest commercial center in northern India. It has an area of 1484 sq. km. being the second most populated & second wealthiest city of the country. Coming to Delhi's transport system, It has the largest road density of 2103km/100 sq. km in India divided into 11 districts [9]. The total road length of Delhi is 28,508 km including 388 km of National Highways(NH 1, NH 2, NH 8, NH 10 and NH 24.) [10] There are 4 categories of road network Arterial, sub arterial, minor arterial and collector. About 1000 Km. Road length carries most traffic in Delhi. Road network accounts for about 21% of the total area, which is way above the average mark of 12 to 15% for urban areas [11].
Buses are popular transport means within Delhi catering 60% of the total demand [9]. It is one of India's largest bus system. The major service provider is Delhi Transport Corporation(DTC) owned by the state government which is one of the largest CNG-powered bus service operators in the world [13]. There are 3900 [13]DTC buses moving around the city traveling about 210 km per day and carrying 3.5 million commuters every day [12]. Cluster buses also operate under public-private partnership under Delhi Integrated Multi-Modal Transit System (DIMTS). It divides Delhi's 657 bus routes into 17 clusters and selected private operators through bidding for each cluster. It is currently operating 1634 buses [13] in the city. For consistency, both DTC and DIMTS follow a joint timetable. DIMTS buses being GPS equipped helps in their realtime tracking.
Since Delhi is such a popular city with a high population, high trade, and commerce, it is obvious to have a high density of people moving across the city. The advent of the metro has not reduced the popularity of buses in the city. RITES [7] concludes that the Delhi metro meets only 20% whereas the public bus system meets more than 70% of the city travel demand. [7] concludes that the transport system has not much helped in evasion of congestion, thus making the entire transport system a real mess.

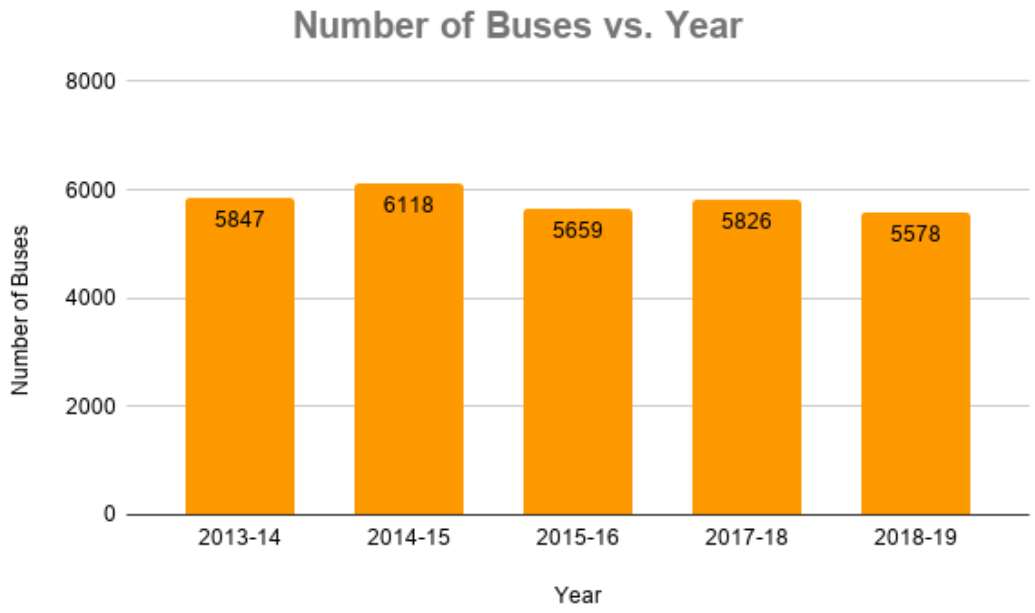

Figure 3.1: Buses operational in last 5 years

The number of buses in the city has been around 6000 marks (Figure 3.1) for the last five years which is very less as compared to the city population. Comparing with London which has 9000 buses catering 8.6 million population, Delhi has just 5578 buses for 16.7 million people. Since buses are the cheap mode of transportation in Delhi as prices are between Rs 5-15 for non AC buses and Rs 10-25 for AC buses, it caters to the need of even the poorest of the population. Congestion had made DTC lose around 5000 trips monthly summing about Rs 9 crore. So, much investment in terms of time needs to be made in their better functioning as they are the lifeline of the city.

Chapter 4

Related Work

The existing literature in this domain can be looked upon with two different perspectives:

4.1 Real-Time congestion detection

Real-time congestion detection is one of the hot favorite research domains on Spatio-temporal data. Real-time congestion detection itself is a wide domain but we can classify it into three different approaches like congestion detection using image data or video data [23] [18], Probe-based method in which the crowdsourcing mechanism is used and congestion detection with other different techniques.

4.1.1 Techniques Involving Image Data

In the Domain of congestion detection, there has been a lot of research that uses image data to predict congestion in real-time. Image analysis is done on the live images and data captured by a camera and different sensors installed on the traffic signal and it then predicts the traffic density as well as congestion status [16]. These methods use different classes of deep learning for congestion detection. In [17] it uses real-time images captured by surveillance cameras on highways which use the backlight as a vehicle feature to estimate vehicle density on roads and compare this density with a threshold to detect congestion. [18] uses an image correlation coefficient on the images provided by the local government as a metric to detect congestion. They detect vehicles using Haar-like features and then apply the threshold to vehicle count and a coefficient for consecutive images. [19] uses vehicle density estimation based on texture feature extraction & texture analysis. They do feature differences between normal road image and congested road image to deduce vehicle density. All the authors have claimed their method to be 90-100% accurate but haven't put their method on live data for a few days to get the feel of the accuracy and just tested it on demo data. In our case where we need to get congestion information on the roads where bus travel, Installing cameras especially for this purpose is not possible. Also, such hardware requires maintenance. We do not have a cost-friendly mechanism for such purpose. Also, the camera has

a limited range, we would need lots of cameras for our purpose. So the above-discussed methods are not feasible.

4.1.2 Probe based congestion detection

[20] makes use of smartphones in vehicles as a probing device& make use of client-server architecture. The location of a vehicle is fetched after a certain period, then they store it as a node to find the distance between them. They take source & destination and find a path between them using google API. They then analyze each path by analyzing stored from probe vehicles. They then return the best & least congested path. [21] uses a hidden Markov model to best predict the traffic congestion between user desired source and destination. They also use GPS data from smartphone equipped devices. Both the papers involve similar principles as they collect data from smartphone equipped vehicles. The first paper hasn't produced the technique used by it to detect congestion & haven't presented any accuracy measures. The method used in second involves congestion prediction on a particular path which is different from our aim as we want to detect the congestion status of all the buses roaming around Delhi which involves around 1000-2000 buses. The similar method used by the author cannot be used due to time constraints to predict congestion.

4.1.3 Congestion Detection using other Technologies

[22] uses Active RFID and GSM technology to predict congestion in real-time. They predict congestion which leads to traffic light junctions. The technique has a limited scope as it can only detect congestion near traffic lights. Also, routers and other devices used in this method need continuous power to operate. These devices require maintenance and security. Such a technique cannot be used in our case as traffic light is not present on every road where the bus goes & also due to the high cost required in the maintenance and installation of devices used.

4.2 Static Congestion Detection

This section involves all those techniques which are used to predict congestion on previously collected data which are obtained from various sources. The section is named static because congestion is not predicted at run time, these techniques are only used to study and analyze traffic patterns throughout the city of interest.

4.2.1 Analysing Traffic flow from multi-sourced data

[1] is the study of detecting various parameters based on multi-sourced data which act as a driving force for traffic congestion. Their theoretical framework combines remote sensing to classify the spatiotemporal pattern of real-life traffic information of Beijing, China & using Geo

detector for determining their potential. They get congestion information from Autonavi which is a map service provider in China. Users feed in live location information which they fetch using an HTTP request. Based on this information they applied clustering to classify the spatiotemporal patterns over the day and a week. They then compared them & identified relationships between patterns other factors using Geo-detector. The paper wants to study the reasons for congestion taking into consideration building height, regional area, etc. The technique can't be mapped to Delhi as there is no fixed segregation of regions in the city also they are treating congestion detection by Autonavi as a BlackBox building things on top of it. There is no such service for Delhi, also they utilize data from various sources but in our case, we have GPS data only.

4.2.2 Congestion detection for Urban planning

[2] presents LoTAD use crowdsourced GPS data from bus spreading all over the city to explore regions which show long term poor traffic situation. They wanted to find those regions which are of interest to the city planners or regions whose situations could not be improved unless some urban planning is implemented. They extract average velocity for traffic conditions and average stop time for travel demand on the Temporal-Spatial segments extracted from bus trajectory. They then calculate the anomaly index (AI) of such segments as they are a bottleneck for traveling on a particular line. All such segments are identified & studied to find similar patterns if any. This tells us about the traffic condition around the city. Since they are studying the entire city they can conclude which regions have more demand based on traffic & urban condition this can be used for city planning. But we are interested in finding all anomalies in real-time so that it could be immediately acted upon for better efficiency of the road transport system. [3] uses the data set of 700 cars with GPS embedding to predict certain traffic-related information for a given route like percentage of a traffic jam for the whole route, average time & speed required to cover the whole route & specific route segments in which the route was divided. [4] Also make use of GPS enabled bus transit systems to record data set and extract valuable information for traffic characteristics and get a detailed view of traffic road congestion which can benefit policymakers.

4.2.3 Other Congestion Detection strategies

[5] took a different area in the domain of congestion detection. They developed a methodology to predict nonrecurrent congestion (NRC) which is caused due to accidents or unexpected incidents. They determined a threshold of 40% which means for routes that take a large duration of time takes 40% more time when there is NRC. Such a technique can be useful but we are interested to find both recurrent and non-recurrent congestion. [6] uses GPS embedded bus as a probe to predict congestion. They use dwell time of buses at stops to predict congestion. Later they compare congestion shown by personal vehicles to the congestion shown by buses but these techniques can't be applied to detect congestion in real-time.

Chapter 5

Problem Statement

Unlike conventional methods which require input like an image from surveillance cameras, crowdsourced locations from mobile phones or information from different sensors like RFID to analyze congestion at real-time, we just have a real-time GPS location from buses moving throughout Delhi.

The problem at hand for real-time congestion detection involves dealing with a raw Spatio-temporal dataset which is uneven and has missing values. Figures 5.1 and 5.2 depicts the missing data value.

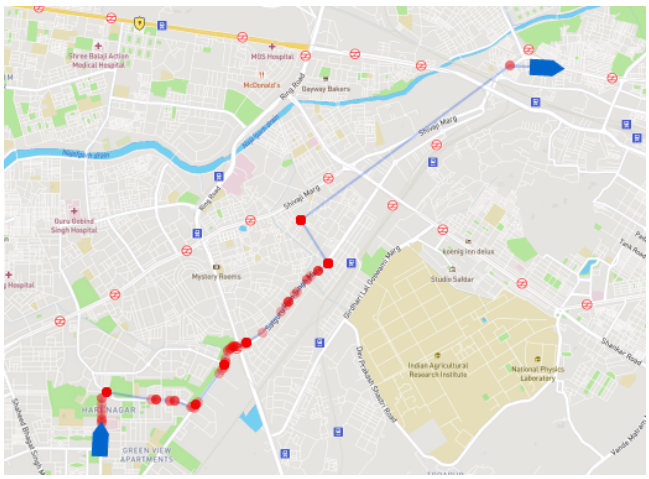

Figure 5.1: Inconsistent Data on 13th Dec 2019 (1)

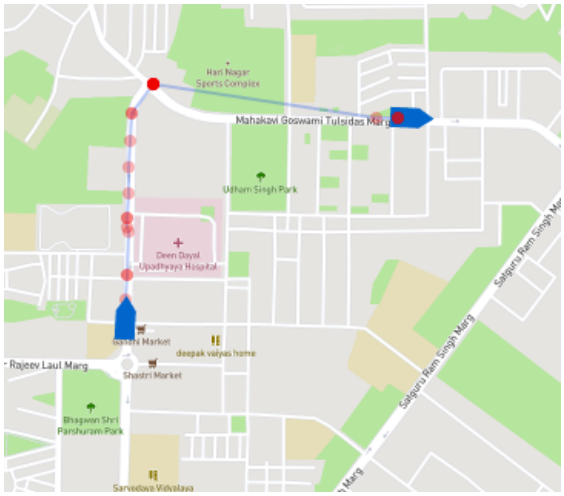

Figure 5.2: Inconsistent Data on 13th Dec 2019 (2)

These images are obtained from the dashboard created for the analysis of our data. Both the images depict the data transmitted by the bus DL1PC4706 which is on route 108Down and moving from Hari Nagar clock tower to Nehru vihar on 13th December 2019. The blue marker is the bus itself, the red dot shows the position where the bus transmitted the data & the blue line connects all such transmitted points. We can easily see how unevenly the bus sends the sensor data.

The data we get for congestion detection is quite sparse as we just have DIMTS cluster buses which are very less in number compared to the actual number of vehicles that run on the road. The problem also requires managing large amounts of data, as we get around 1000-1500 bus geodata every 10 seconds which results in around 1GB of data each day. The management of such a dataset is important so that we can utilize it for our purpose and keep it in a form that can be useful for analysis. The analysis is itself a challenging task as we need to balance out the prediction based on previous data as well as need to incorporate real-time entries fetched from the buses. The time taken to predict such congestion should be less than 10 seconds so that we can update the congestion status at every consecutive data fetch operation and at the same time label the data entries according to congestion status.

So our major task is to predict accurate congestion status in real-time based on sparse and uneven Spatio-temporal input data so that the whole task becomes cost-effective and can serve the purpose for the government authorities.

Chapter 6

Methodology

6.1 Proposed Architecture

The entire thesis revolve around the proposed architecture as shown in Figure 6.1. The architecture depicts the flow to detect real-time congestion without latency. The architecture modules are explained in subsequent subsections.

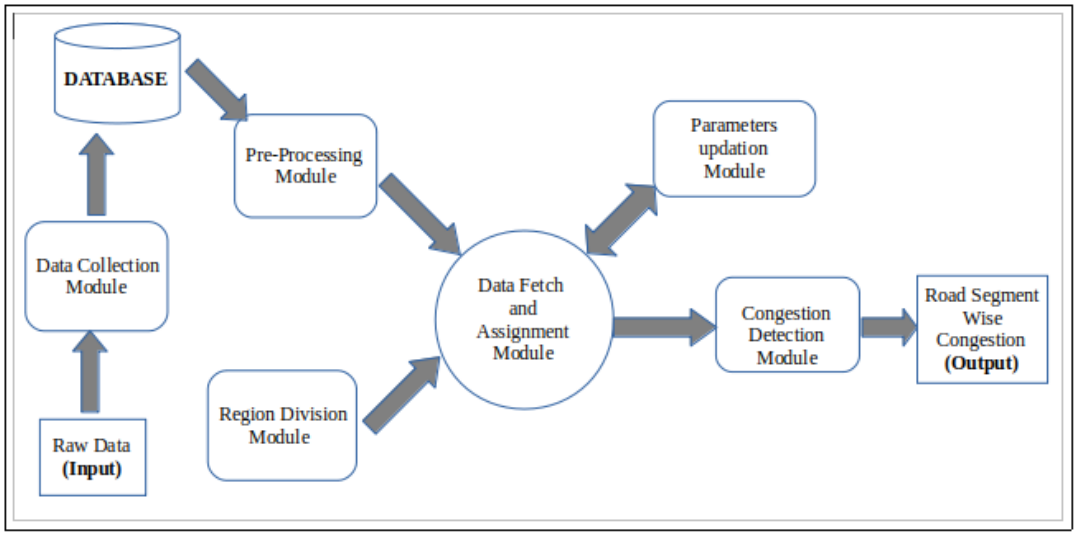

Figure 6.1: Proposed Congestion Detection Architecture

6.2 Data Collection Module

We get Live bus data of 1500-1700 DIMTS cluster buses which are operational in Delhi. We use Open transit Data, Delhi (OTD) API to get the live location of buses. It gives XML response of the live location of all the buses moving around Delhi, the information gets updated every 10 seconds. So we make API calls once every 10 seconds. We then take the XML response and parse it to extract the required information. We collect data regularly and store it inside the POSTGRESQL database. The information retrieved from API can be seen in Table 6.1:

Table 6.1: Information Retrieved from API

Field	**Significance**
Trip Id	Signifies To/From journey & journey number
Route Id	Route no. on which the bus in functional
Latitude	Latitude of the bus
Longitude	Longitude of the bus
Vehicle Id	Bus No. of the bus
TimeStamp	Time of GPS data feed onto the server
Speed	Instantaneous speed of the bus (m/s)

6.3 Region Division Module

To Analyze the congestion of Delhi, We first need to divide it spatially & for this we take help from shapefile. A shapefile consists of geometric information (lines, points, polygons) along with attribute features for a geographic location. We got shapefile from OpenStreetMaps(OSM). The Delhi map obtained from OSM(Open Street Maps) is divided into Grids, where each cell of the grid is of similar area.

For division, we took bounding box coordinates of Delhi from OSM. We divide each cell with the boundary latitude and longitude, and we also have the vertical and horizontal distance of the next neighbor cell which is the same in our case for every cell. For division into grids, we pass in a number 'n' which creates $(n)^2$ number of cells. Coordinates (latitude, longitude) of each cell could be calculated with the formula in 6.1.

$$\begin{aligned} \text{latitude} &= \text{minimum\_Latitude} + \text{row} * \text{latitude\_step} \\ \text{longitude} &= \text{minimum\_Longitude} + \text{row} * \text{longitude\_step} \end{aligned} \tag{6.1}$$

where (minimum latitude, minimum longitude) & (maximum latitude, maximum longitude) are diagonal coordinates of Delhi's bounding box. latitude step and longitude step is the difference between the minimum & maximum latitude and longitude respectively divided by 'n'.

Figure 6.2: Connaught place, after dividing Delhi into grids

Connaught place after the spatial division of Delhi into the grid can be seen in Figure 6.2. We also extracted latitudes & longitudes of all the roads in the particular grid cell. We store latitude longitude of roads of a grid cell as a posting list. For mapping a particular pair of latitude and longitude to a grid cell we use a hashing function mentioned below (Source Code 6.1). Here grid size refers to 'n' as discussed before. Hashing helps us to map any point to the road in O(1) time and saves lots of computations, thus making analyzing faster in real-time.

Source Code 6.1: Hashing Function

```
import math
def hash(latitude,longitude,gridSize):
    latStep = (maximum_latitude - minimum_latitude)/gridSize
    longStep= (maximum_longitude - minimum_longitude)/gridSize
    latDiff = latitude - minimum_latitude
    longDiff = longitude -minimum_longitude
    row = math.floor(latDiff/latStep)
    col = math.floor(longDiff/longStep)
    cell_id = row*gridSize + col
    return cell_id
```

6.4 Data Preprocessing Module

The collected data stored in the Database is preprocessed before doing any kind of analysis. We apply the following pre-processing rules:

- We remove the redundant entries which are sometimes fetched in consecutive data retrieval operation.
- Instantaneous speed fetched from the API is discarded as its full of error. We calculate speed ourselves.

- GPS signals are believed to have an error rate of about 3-5m. So to avoid any false point, We map the fetched coordinate to the nearest road segment by first mapping the point to a particular grid using the hashing function (Source Code 6.1). We then use a posting list of roads lying in that particular grid cell and calculate haversine distance between the fetched geopoint & the point of roads in the cell. The road point whose distance is minimum to a fetched point is marked like a road for the fetched point.

- We discard all the points which are found near the bus stops as the speed at bus stops would be low so that passengers can board & de-board the bus, such speed entries cannot be used for detection mechanism.

6.5 Data Fetch and Assignment Module

As we are continuously adding data to the database. For Analysis we need to collect data for testing & assign it to roads of the grid cells. We now fetch data from our database. The fetched data is mapped to roads of previously divided grid cells obtained from section 6.2. by setting the appropriate value to grid size 'n'. Each entry of the data-set corresponds to latitude & longitude of the vehicle id at the timestamp when the GPS entry was fed. We hash each entry using the hash function given in source code 6.1. Now to analyze temporally we divide the data-set into a certain time interval, which gives us various slots for the entire day. For example, if we define time interval to be 15 min we get 48 slots for the entire day. So we can map the bus entries to these slots and can train the model separately for each slot.
This results in a structure that stores data entries corresponding to each time slot for each cell of the grid. For each bus that is present in a particular grid at a particular time slot, we calculate speed between two entries of a particular bus id. Speed is calculated as in equation 6.2.

$$Speed = \text{Haversine Distance between points/difference between timestamps} \tag{6.2}$$

This results in a series of the speed of a particular bus. We also store distance traveled and time is taken by the bus along with the speed as it would help us to calculate the average speed for the bus. We also keep track of several different buses within a time slot. Since we have got speed entries we calculate the average speed of the bus. The collected and analyzed data is passed onto the next module for congestion detection.

6.6 Congestion Detection Module

The data collected is analyzed in this module. The main aim of this module is to detect immediate congestion reflected by the buses. We have tried various techniques for providing congestion status from bus speeds, the techniques are listed below:

- **Kernel Density Estimation(KDE)**
  To analyze congestion, we use the KDE algorithm with the gaussian kernel. To train the

model we use our collected dataset & map it spatially to divided grid cells, inside the grid cells we map the data into various temporal slots of the particular time interval. Each slot stores the speed of a vehicle belonging to a particular road inside a particular grid cell. We use these speeds to train the kernel density estimator for a timeslot belonging to a road of a grid. To get the bandwidth we use the grid search algorithm where we take a range of bandwidth from 1 to 5 and use 5 fold cross-validation to predict the bandwidth. Using the bandwidth we train a slot speed and get two congestion intervals. We then map the immediate speed reflected by the bus to get the congestion. The congestion status shown by all the buses at a particular road is averaged to get congestion of a road.

- **Jenks Natural Break Optimisation Algorithm:**
  To analyze congestion, we use the Jenks algorithm. To train the model we use our collected dataset & map it spatially to divided grid cells, inside the grid cells we map the data into various temporal slots of the particular time interval. Each slot stores the speed of a vehicle belonging to a particular road inside a particular grid cell. We now run a Jenks algorithm to get a congestion interval. We then map the immediate speed to get the congestion reflected by the bus. The congestion status shown by all the buses at a particular road is averaged to get congestion of a road.

- **Comparing speed with Standard Deviation:**
  We Identified that the speed of the buses in a particular cell of the grid corresponds to Normal distribution. We learned all normal distribution parameters for 40 days and have stored learned to Mean & standard deviation corresponding to a particular cell at a particular time slot. The slots speed trend for all regions in 40 days can be seen in Figure 6.3. We test the speed of the bus with the stored normal distribution parameters. We then decide the threshold for binary congestion classification, If speed is less than the threshold it is termed as congestion & free flow if above it. We have taken various variations of speeds for its testing. We take a mean average of all the congestion labels for a particular road and return as output mean congestion belonging to a particular road.

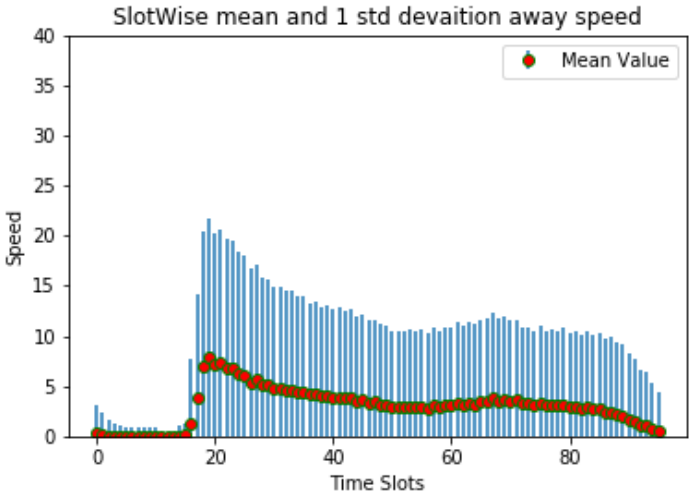

Figure 6.3: Speed Trend for 96 slots of 15 min each.

- **Probability using Normal Distribution (Normal CDF) :**
  We Identified that the speed of the buses in a particular cell of the grid corresponds to Normal distribution. We have stored learned Mean & standard deviation corresponding to a particular cell at a particular time slot. We then calculate the probability of the speed to be less than 3km/hr. We use the probability result as congestion percent.

- **K-means algorithm :** We train the K-means model using the speed belonging to a road in a grid cell for a particular time slot. We set k as 2 since we have decided the congestion intervals to be 2 (Free-flowing, Congested). We use the trained model to get the label for the new speed entries which we get at run time. We take a mean average of all the congestion labels for a particular road and return as output mean congestion belonging to a particular road.

- **Count Congestion :** We calculate the speed between successive entries of the bus in a particular grid cell. We define a metric called the congestion index as shown in equation 6.3. This metric is an improved version of the metric mentioned in [3].

$$CongestionIndex(C.I.) = BusEntries < 3km/hr/\text{Total Bus entries} \tag{6.3}$$

  This index is calculated for all roads which belong to a particular grid cell. This index depicts the congestion of a particular road.

At last, we combine congestion observed across for all road segments of a particular road throughout all grid cells and return road wise congestion of all Delhi roads. We have used all the techniques to analyze the congestion. All the algorithms show a different accuracy. The best among all the algorithms are used for detection purpose.

6.7 Parameters updation module

In this phase, the entire data collected in the previous day are fetched. We then use successive bus entries to calculate the speed of the bus in a particular time slot in a particular cell of the grid. These entries are combined with the previous learned entries of mean and standard deviation. This new data is used to calculate congestion using standard deviation and using a normal probability distribution. We also add the speed entries to the roads dataset saved before. This updated dataset is run again to train the k-means model, Jenks normal algorithm and kernel density estimation algorithm. The newly updated data is used in the analysis for the current day congestion analysis. This module is executed only once throughout the day during the night time from 12 am to 6 am because during this time very few buses are observed, So this time is perfect for training our model which can be used for the entire day analysis.

Chapter 7

Experimental Results

7.1 5 Day Analysis using Count Congestion

To map the spatial data-set collected throughout the entire city. We use our method in Chapter 6 & set the grid size to 27 which results in 729 grid cells. Each cell is of size 2 km square. We saved this grid divided object & started to map data-set corresponding to each day. The speed and bus distributions of cell 234 can be seen in Figure 7.1.

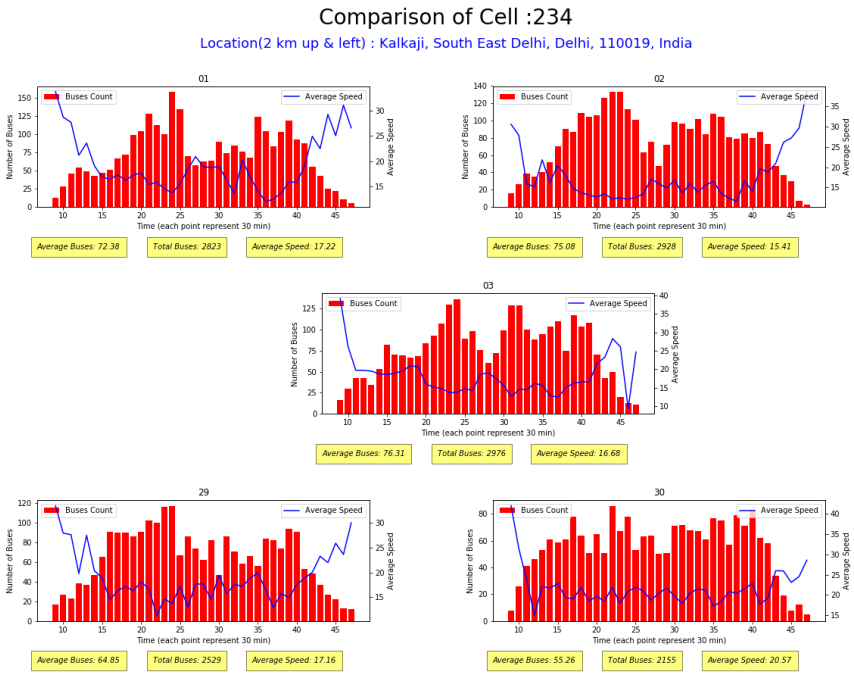

Figure 7.1: Comparison of cell 234 among 5 days

The figure shows cell which depicts area near Kalkaji, Southeast Delhi, location information is obtained by reverse Geo-encoding using OSM nominatim API. The cell is quite a heavy cell as

around 2500 buses float through this cell throughout the day. On average there are 68 buses in a time slot. There are no buses from 1 am to 4 am. Throughout the day the average speed is around 17km/hr. The maximum permissible speed for a DTC bus is 40km/hr. According to [8] RITES says average speed on roads of Delhi is around 28km/hr in peak hours and 31km/hr in non-peak hour & Centre for Science and Environment(CSE) says there is not much difference in speed in peak and non-peak hours as former comes out to be 27km/hr and 28km/hr respectively. So taking the average speed of 28km/hr, We can conclude that this cell is a moderately congested cell based on its average speed as it is less than the studied average speed.

Since we aim to get congestion information, we cannot just conclude only based on the average speed of the cell. We need some better metrics. We now use the count congestion method mentioned in chapter 6. Congestion index on our previously mentioned cell can be seen in Figure 7.2

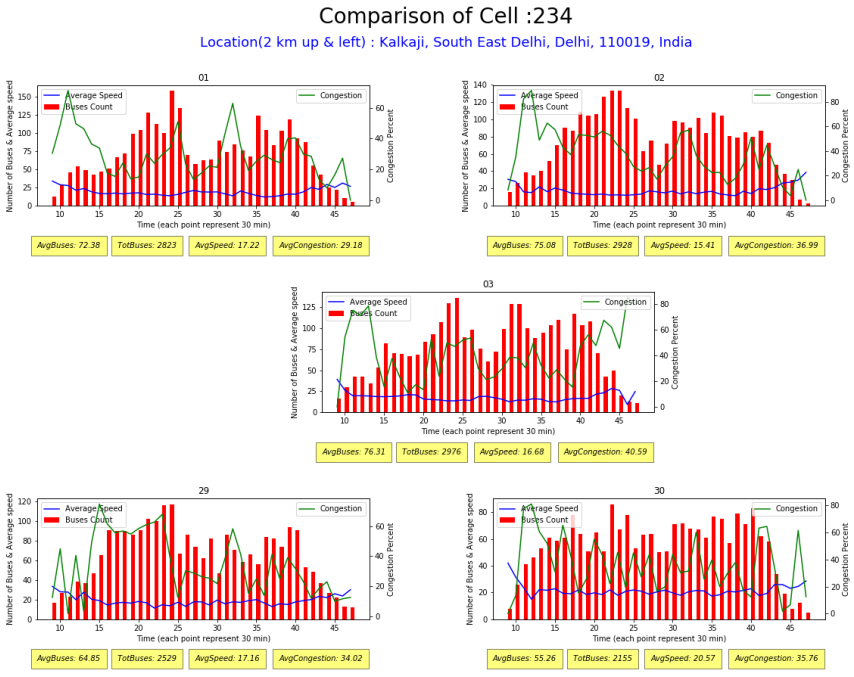

Figure 7.2: Comparison of cell 234 with Congestion Index(C.I.) among 5 days

From this figure, it is evident that whenever there is a dip in average speed there is a spike in the congestion index. Also, C.I. shows bit different results as in figure 7.1 as average congestion comes out to be around 35% which means there is not much congestion in this cell and traffic is free-flowing.

Now just concluding congestion for a cell doesn't give much insight as a cell can have many roads. Each road, in turn, can show different behavior. Thus we must study the congestion pattern for each road of the cell. Also, we need to know how many buses show a similar pattern for the road. This count shows confidence as higher the count of buses for the road more precisely we can

conclude our observation. The analysis of grid cells according to its road can be seen in Figure 7.4 to Figure 7.8. The actual area depicting cell 234 can be seen in Figure 7.3.

Figure 7.4 through Figure 7.8 gives road wise analysis of Cell 234 at time 8 pm to 8:30 pm from 29th July to 3rd August except for 31st July. In figure 7.7 for Road id 'A_8633' there are five buses & all of them show null congestion. All the 5 buses give 20 entries in total. Hence with such a high number of buses, it is evident that this road has null congestion. Similarly for road id 'A_116431' there are 10 entries from 2 buses both of them show 100% congestion.

Figure 7.3: Actual Area of cell 234 on OSM maps

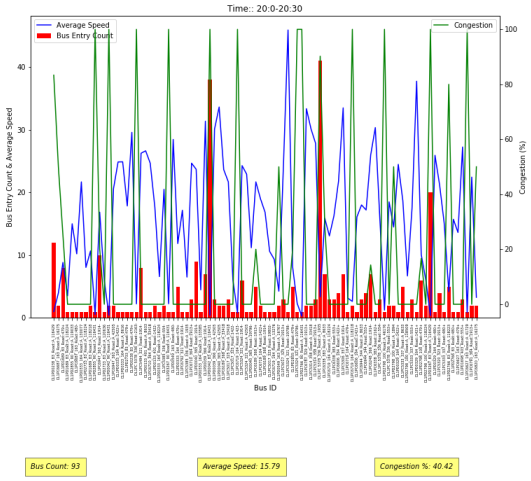

Figure 7.4: 20 -20:30 on 29 july

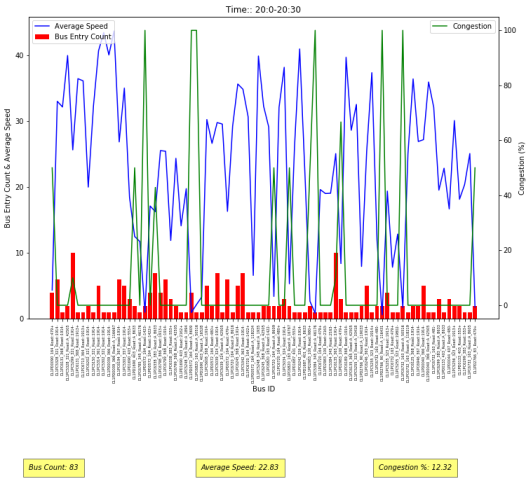

Figure 7.5: 20 -20:30 on 30 july

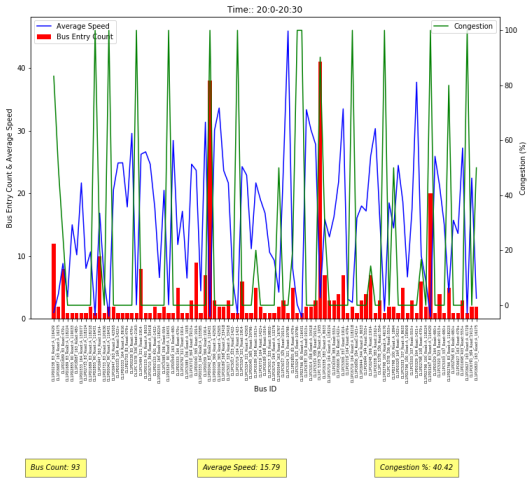

Figure 7.6: 20-20:30 on 1 Aug

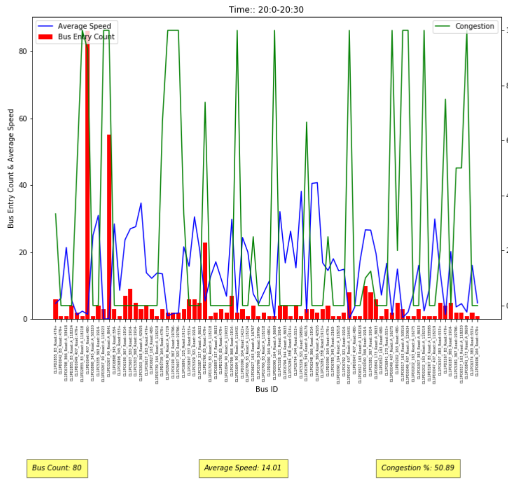

Figure 7.7: 20-20:30 on 2 Aug

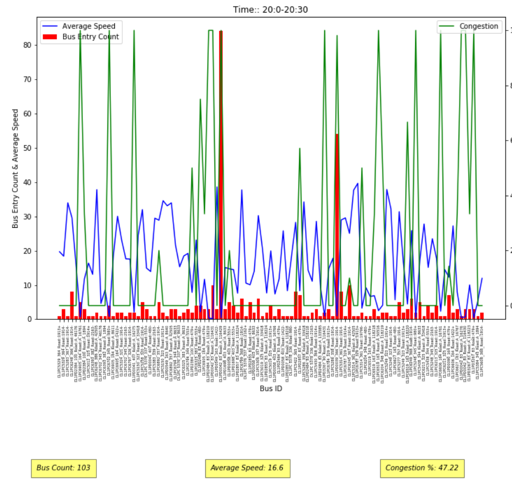

Figure 7.8: 20-20:30 on 3 Aug

To deeply Analyze this cell, the comparison table can be seen in Figure 7.9 and Figure 7.10. The column 'id' in the table stands for road id, all other columns give congestion info for a particular day. The first entry in the list gives a count of entries having speed less than 3km/hr, those entries which are near to bus stop within a range of 10m are ignored from the consideration of congestion entry because it means the bus has stopped near the bus stop for carrying passengers. the second entry is total entries for the bus & all other entries tell C.I. for a bus on that road. Leaving the first 2 entries, More the entries in the list more the number of buses on that road. The cell has a total of 60 roads & 29 stops. The road id '1914-' which corresponds to Outer Ring Road/M. P. Road/Bhakti Vedant Swami Marg according to HERE Maps is a bus heavy road as there are a lot of bus entries in it. Also, this road is not much congested in this cell as only 1 or 2 buses contribute to congestion count rest all are zero. Similarly, road id '980+' which corresponds to Okhla Road/Shri Maa Anandmayee Marg is a very congested road as whenever there is a bus on this road, it is always in heavy congestion. On similar grounds road id '479+' which corresponds to Lala Lajpat Rai Marg/Hans Raj Sethi Marg has very mild congestion as only a few buses are given congestion entries which signify it has running traffic.

	id	29	30	01	02	03
0	1422+	[0, 1, 0.0]	[0, 5, 0.0, 0.0]	[0, 8, 0.0, 0.0, 0.0, 0.0]	[0, 3, 0.0]	[0, 4, 0.0, 0.0]
1	1422-	[0, 3, 0.0]	[0, 2, 0.0]	[0, 3, 0.0, 0.0]	0	[0, 4, 0.0, 0.0]
2	1913+	[0, 9, 0.0, 0.0, 0.0, 0.0]	[0, 12, 0.0, 0.0, 0.0, 0.0, 0.0]	[1, 22, 0.0, 20.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[1, 6, 33.33, 0.0]	[0, 14, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
3	1914+	[0, 2, 0.0]	[0, 1, 0.0]	0	[0, 1, 0.0]	[0, 7, 0.0, 0.0, 0.0, 0.0]
4	1914-	[3, 87, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 16.67, 10.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[2, 97, 0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[3, 56, 20.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 14.29]	[2, 70, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 10.0, 12.5, 0.0]	[55, 137, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 20.0, 0.0, 0.0, 0.0, 0.0, 98.15, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0]
5	19796-	[1, 2, 100.0, 0.0]	0	[1, 2, 0.0, 100.0]	[5, 7, 100.0, 100.0, 0.0, 50.0]	[1, 2, 50.0]
6	19800-	[0, 2, 0.0, 0.0]	0	0	[0, 1, 0.0]	0
7	2162+	[0, 1, 0.0]	[0, 1, 0.0]	[0, 1, 0.0]	0	[0, 1, 0.0]
8	2163-	[0, 5, 0.0, 0.0, 0.0]	[0, 2, 0.0, 0.0]	[1, 1, 100.0]	[0, 1, 0.0]	[0, 7, 0.0, 0.0, 0.0, 0.0, 0.0]
9	479+	[13, 45, 50.0, 27.27, 0.0, 20.0, 25.0, 0.0, 66.67, 25.0, 0.0, 30.0]	[5, 17, 50.0, 0.0, 0.0, 66.67, 0.0, 50.0]	[4, 28, 25.0, 0.0, 0.0, 0.0, 20.0, 0.0, 16.67, 0.0]	[26, 51, 33.33, 0.0, 50.0, 0.0, 66.67, 73.91, 0.0, 25.0, 75.0, 0.0]	[10, 31, 50.0, 33.33, 0.0, 0.0, 57.14, 0.0, 14.29, 100.0]
10	480-	[51, 64, 91.84, 0.0, 33.33, 28.57, 75.0]	[0, 5, 0.0, 0.0]	[0, 2, 0.0, 0.0]	[81, 96, 93.02, 0.0, 0.0, 0.0, 20.0]	[3, 7, 0.0, 75.0, 0.0, 0.0]
11	553+	[0, 12, 0.0, 0.0, 0.0]	[0, 10, 0.0, 0.0, 0.0, 0.0, 0.0]	[0, 15, 0.0, 0.0, 0.0, 0.0, 0.0]	[1, 21, 0.0, 0.0, 0.0, 0.0, 20.0, 0.0]	[1, 23, 0.0, 0.0, 0.0, 20.0, 0.0, 0.0]
12	554-	[0, 1, 0.0]	0	[0, 5, 0.0, 0.0]	[0, 3, 0.0]	0
13	980+	[7, 8, 100.0, 100.0, 100.0, 50.0, 100.0]	[1, 2, 50.0]	0	0	[6, 6, 100.0, 100.0]
14	A_116431	[1, 1, 100.0]	[2, 2, 100.0]	[43, 43, 100.0, 100.0, 100.0, 100.0, 100.0]	[10, 10, 100.0, 100.0]	[12, 12, 100.0, 100.0]
15	A_118218	[0, 6, 0.0, 0.0, 0.0, 0.0, 0.0]	0	[2, 4, 0.0, 100.0]	[1, 7, 100.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[1, 5, 0.0, 0.0, 33.33]
16	A_118219	[0, 1, 0.0]	[0, 1, 0.0]	0	[1, 2, 100.0, 0.0]	[4, 5, 100.0, 50.0]
17	A_118224	[0, 11, 0.0, 0.0, 0.0, 0.0]	[0, 1, 0.0]	[0, 8, 0.0, 0.0, 0.0, 0.0, 0.0]	[0, 5, 0.0, 0.0]	[0, 4, 0.0, 0.0]
18	A_119033	[1, 1, 100.0]	[0, 2, 0.0]	[0, 1, 0.0]	[0, 2, 0.0]	0
19	A_121504	[2, 3, 66.67]	0	0	0	0
20	A_130055	[0, 1, 0.0]	0	0	0	0
21	A_133585	[15, 15, 100.0]	0	[0, 1, 0.0]	[0, 1, 0.0]	[0, 1, 0.0]
22	A_16275	[0, 1, 0.0]	0	[2, 4, 50.0, 50.0]	[2, 3, 100.0, 0.0]	[2, 2, 100.0]
23	A_16528	[1, 2, 100.0, 0.0]	[0, 1, 0.0]	0	0	0
24	A_16767	[1, 1, 100.0]	[1, 2, 50.0]	[1, 2, 50.0]	[1, 4, 25.0]	[1, 3, 33.33]
25	A_1894	[0, 2, 0.0, 0.0]	[0, 1, 0.0]	[1, 2, 50.0]	0	[1, 5, 33.33, 0.0, 0.0]
26	A_1935	[0, 2, 0.0]	[0, 1, 0.0]	[37, 42, 0.0, 90.24]	0	0
27	A_33418	[0, 1, 0.0]	[0, 1, 0.0]	[0, 8, 0.0, 0.0, 0.0, 0.0]	[0, 1, 0.0]	[0, 4, 0.0, 0.0, 0.0, 0.0]
28	A_42505	[0, 9, 0.0, 0.0, 0.0, 0.0]	[0, 9, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0, 8, 0.0, 0.0, 0.0, 0.0]	[0, 5, 0.0, 0.0]	[0, 6, 0.0, 0.0, 0.0]
29	A_45003	[1, 3, 33.33]	0	0	0	0

Figure 7.9: Five day comparison table(1)

30	A_46176	[1, 2, 50.0]	[1, 2, 0.0, 100.0]	[1, 1, 100.0]	[2, 3, 66.67]	[0, 2, 0.0, 0.0]
31	A_50516	[1, 1, 100.0]	[3, 3, 100.0, 100.0]	0	[3, 3, 100.0]	[1, 1, 100.0]
32	A_8633	[0, 8, 0.0, 0.0, 0.0]	[4, 14, 50.0, 42.86, 0.0, 0.0, 0.0]	[2, 16, 0.0, 28.57, 0.0, 0.0]	[0, 20, 0.0, 0.0, 0.0, 0.0, 0.0]	[0, 16, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
33	1423-	0	[2, 2, 100.0]	[1, 1, 100.0]	0	0
34	19802-	0	[0, 1, 0.0]	[0, 1, 0.0]	0	0
35	480+	0	[0, 4, 0.0, 0.0]	[0, 3, 0.0, 0.0, 0.0]	[0, 1, 0.0]	0
36	481-	0	[0, 1, 0.0]	0	0	0
37	A_116667	0	[0, 1, 0.0]	0	0	0
38	A_8605	0	[0, 1, 0.0]	0	0	0
39	A_8609	0	[5, 6, 100.0, 50.0]	0	[2, 2, 100.0, 100.0]	0
40	A_8616	0	[0, 1, 0.0]	[0, 1, 0.0]	0	0
41	1421+	0	0	[0, 2, 0.0, 0.0]	0	0
42	19798-	0	0	[0, 1, 0.0]	0	0
43	A_116429	0	0	[30, 32, 83.33, 100.0]	0	[84, 84, 100.0]
44	A_118223	0	0	[4, 5, 80.0]	[3, 3, 100.0]	[1, 3, 33.33]
45	A_118509	0	0	[0, 1, 0.0]	0	0
46	A_120636	0	0	[0, 1, 0.0]	0	0
47	A_130077	0	0	[0, 1, 0.0]	0	0
48	A_13276	0	0	[0, 1, 0.0]	0	0
49	A_41729	0	0	[1, 1, 100.0]	[2, 2, 100.0]	0
50	A_118220	0	0	0	[1, 2, 50.0]	[2, 3, 66.67]
51	A_120634	0	0	0	[1, 1, 100.0]	0
52	A_50320	0	0	0	[0, 1, 0.0]	0
53	A_8641	0	0	0	[55, 55, 100.0]	0
54	980-	0	0	0	0	[0, 1, 0.0]
55	A_116430	0	0	0	0	[3, 3, 100.0]
56	A_16276	0	0	0	0	[1, 1, 100.0]
57	A_16761	0	0	0	0	[6, 6, 100.0, 100.0]
58	A_16766	0	0	0	0	[0, 1, 0.0]
59	A_8611	0	0	0	0	[0, 1, 0.0]
60	A_8640	0	0	0	0	[0, 1, 0.0]

Figure 7.10: Five day comparison table(2)

7.2 Comparison of Algorithms for Congestion Detection

For this experiment, we stabilized grid size to be 250 sq. meter (230 x 230) and varied test time duration from 1 to 10 minutes for determining binary congestion status. Training time duration is kept as 15 minutes which means we would have 96 training timeslot in one day. In this regard we took 8 different Algorithms:

- **Mean + 2*Std**:: Here we took mean+2*std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if greater than it.
- **Mean + 1*Std**:: Here we took mean+std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **Kmeans Algorithm**:: Here we used K-means algorithm as described in section 6.6
- **KDE Algorithm**:: Here we used Kernel density estimation algorithm described in section 6.6
- **Jenks**:: Here we used Jenks natural break distribution algorithm as described in section 6.6
- **Count Confidence Congestion**:: Here we reported percentage of speed entries that are less than 3 km/hr for all buses & number of buses as confidence score to conclude congestion
- **Count General Congestion**:: Here we reported a percentage of speed entries that are less than 3 km/hr for all buses and reported the average as congestion percentage.
- **Normal CDF**:: Here we used probability distribution method as described in section 6.6

The graphs can be seen in Figure 7.11 and 7.12.

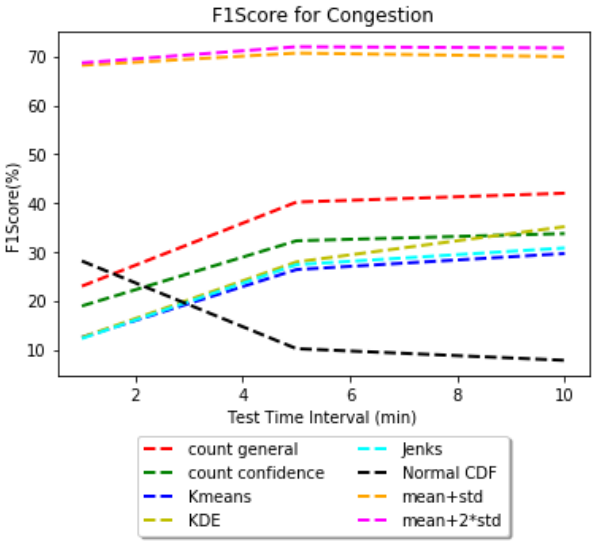

Figure 7.11: F1-Score

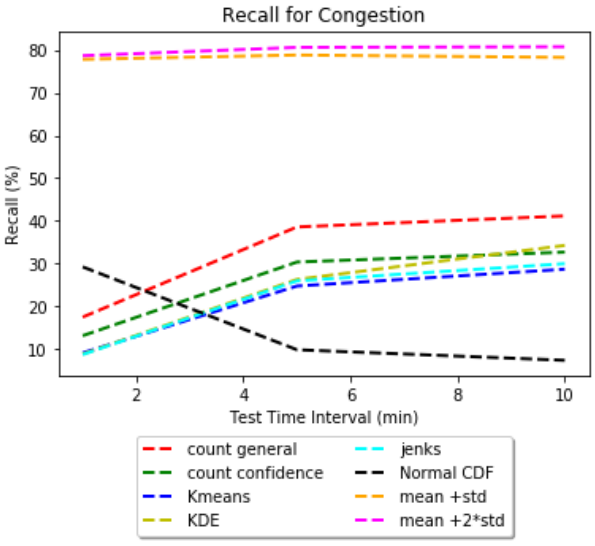

Figure 7.12: Recall

From the above experiment, it is quite evident that unsupervised learning algorithms like Kmeans,

Jenks and Density estimation algorithms like Kernel Density estimation (KDE) perform poorly for the problem of congestion detection. This is because we have very sparse data as very few buses are operational at any particular location. Also due to non-continuous data, we get fewer sample points which are not sufficient to train such models. Also training these models to take a lot of time as they are quite compute-intensive and it won't be possible retrain model every night as we mentioned in section 6.7. The simple statistical method which compares speed with mean and standard deviation surpasses all other methods in performance.

7.3 Effect of Grid Size in Congestion Detection

For this experiment, we have a varied grid cell size from 100 to 250 sq. meters for determining binary congestion status. Training time i.e. time to learn the mean and standard deviation for each road of the grid cell, has been kept as 15 minutes, which means we would have 96 training timeslots in a day. We have tried the above experiment for a test time period of 1 & 5 minutes. In this Regard we took 6 different Algorithms:

- **Mean + 2*Std**:: Here we took mean+2*std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **Mean + 1*Std**:: Here we took mean+std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **Mean**:: Here we took mean as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **Count Confidence Congestion**:: Here we reported percentage of speed entries that are less than 3 km/hr for all buses & number of buses as confidence score to conclude congestion
- **Count General Congestion**:: Here we reported a percentage of speed entries that are less than 3 km/hr for all buses and reported the average as congestion percentage.
- **Normal CDF**:: Here we used probability distribution method as described in section 6.6

The Graph can be seen in Figure 7.13 to 7.16.

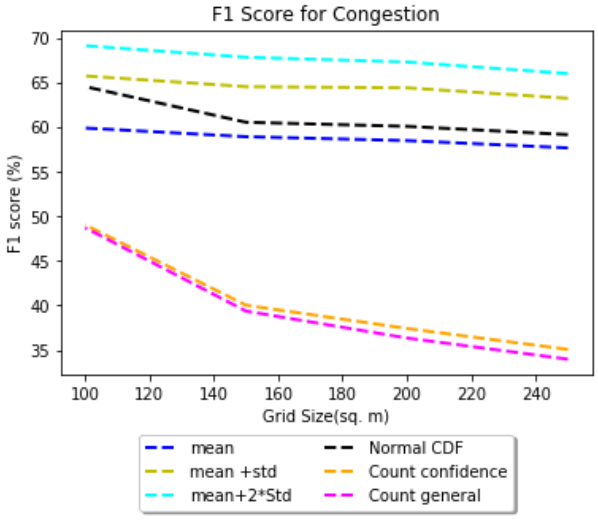

Figure 7.13: F1-Score (Test time : 1 min)

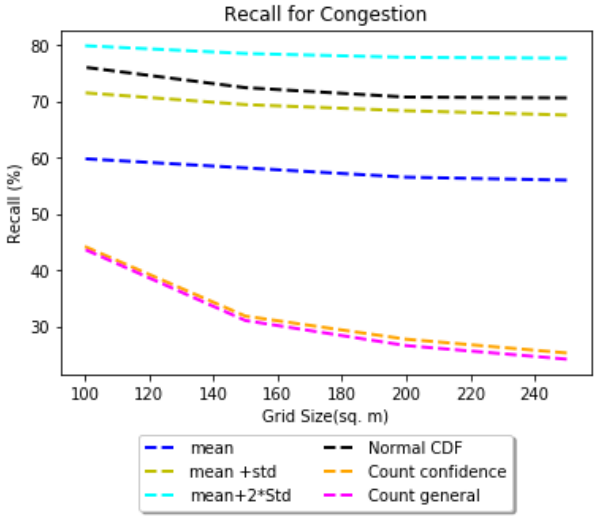

Figure 7.14: Recall (Test time : 1 min)

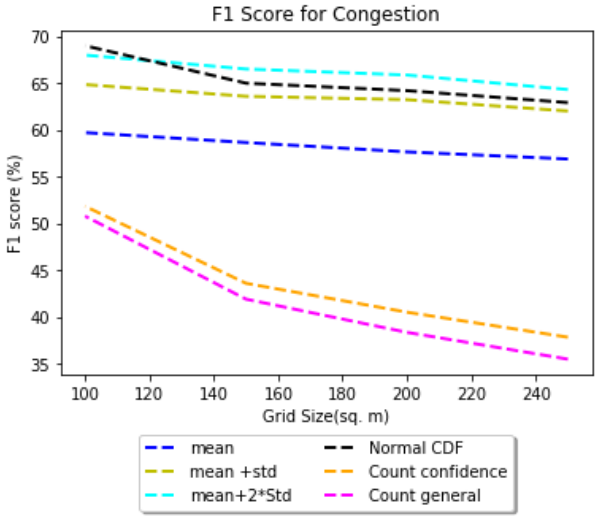

Figure 7.15: F1-Score (Test time : 5 min)

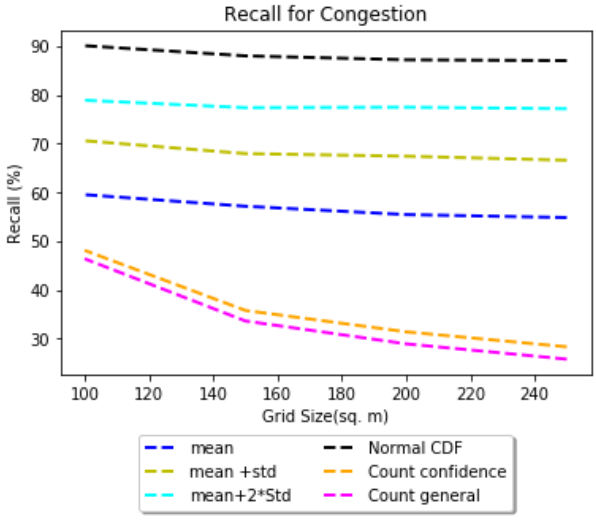

Figure 7.16: Recall (Test time : 5 min)

From the above experiment, we can conclude that as the grid size increases there is a dip in accuracy & recall. This is because when the size is small we get a lot of samples for smaller areas resulting in better-trained parameters. Increasing the time duration leads to better recall but the F1-Score remain nearly the same with little fluctuations. The more time we spend on analyzing congestion, the more the chance we have to get all the congestion entries. The algorithm shows the same results for all time intervals showing an F1-score of about 70%.

7.4 Effect of Test Time period in Congestion Detection

For this experiment, we stabilized grid size to be 250 sq. meter (230 x 230) and varied test time duration from 1 to 10 minutes for determining binary congestion status. Training time duration is kept as 15 minutes which means we would have 96 training timeslot in one day. In this regard we took 8 different Algorithms:

- **Mean + 2*Std**:: Here we took mean+2*std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if greater than it.
- **Mean + 1*Std**:: Here we took mean+std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **MeanSpeedA**:: Here we took mean+2*std as a breakpoint for a road segment inside the grid cell at a training timeslot& concluded congestion if the mean of all speed entries on a particular road is less than breakpoint and free flow if above it.
- **MeanSpeedB**:: Here we took mean+std fas a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if the mean of all speed entries on a particular road is less than breakpoint and free flow if above it.
- **MedianSpeedA**:: Here we took mean+2*std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if the median of all speed entries on a particular road is less than breakpoint and free flow if above it.
- **MedianSpeedB**:: Here we took mean+std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if the median of all speed entries on a particular road is less than breakpoint and free flow if above it.
- **ConfidenceSpeedA**:: Here we took mean+ 2*std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if most of the buses mean speed is less than breakpoint and free flow if above it.
- **ConfidenceSpeedB**:: Here we took mean+ std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if most of the buses mean speed is less than breakpoint and free flow if above it.
- **Normal CDF**:: Here we used probability distribution method as described in section 6.6
- **Count Congestion**:: Here we counted speed entries that are less than 3 km/hr for all buses & number of buses as confidence score to conclude congestion

The graphs can be seen in Figure 7.17 and 7.18.

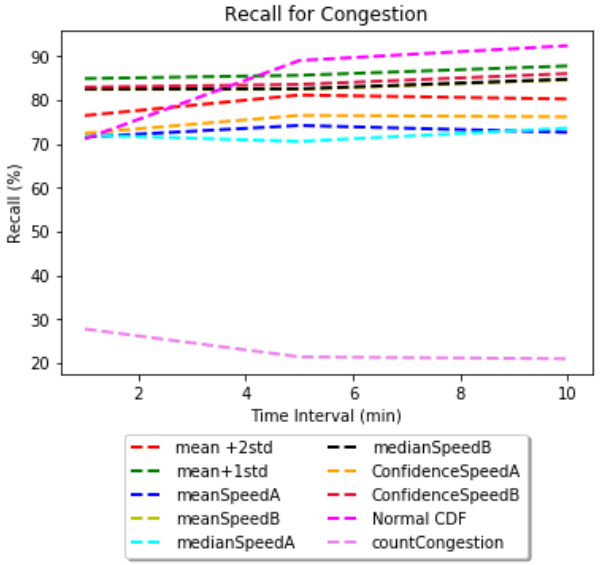

Figure 7.17: Recall

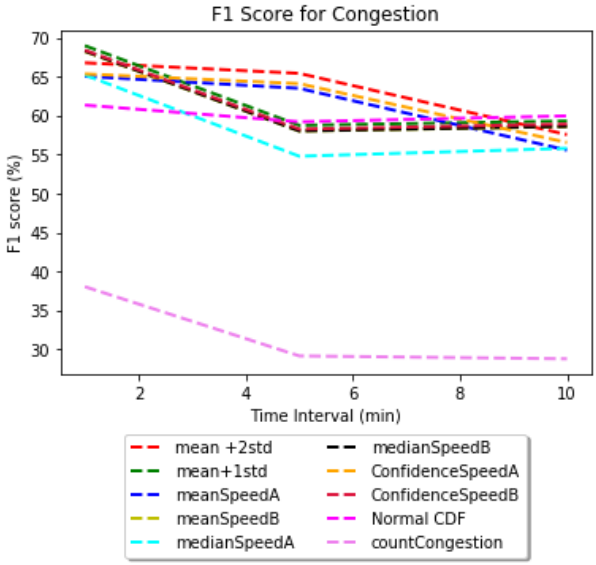

Figure 7.18: F1-Score

Based on graphs we can observe that time intervals of 1 and 5 are best for congestion identification and increasing further decreases our F1-score. This can be because if analyzing for longer duration it gives more missing values and due to those missing values we are not able to conclude congestion more precisely. We get the best F1-score at a time interval of 1 minute because during this time we have a high probability to get all the buses points and if we do not get values we won't be evaluating congestion for that area.

7.5 Effect of Training time duration in Congestion Detection

For this experiment, we have varied training time duration from 15 to 60 minutes for determining binary congestion status. Training means we will learn the mean and standard deviation of each of the above duration for each road of the grid cell. We have tried the above experiment for Grid Sizes 100 sq. mtr, 150 sq. mtr, 200 sq. mtr & 250 sq. mtr. In this Regard we took 7 different Algorithms:

- **Mean - 2*Std**:: Here we took mean-2*std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **Mean - 1*Std**:: Here we took mean-std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **Mean**:: Here we took to mean as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **Mean + 1*Std**:: Here we took mean+std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **Mean + 2*Std**:: Here we took mean+2*std as a breakpoint for a road segment inside the grid cell at a training timeslot & concluded congestion if individual speed entry is less than breakpoint and free flow if above it.
- **Normal CDF**:: Here we used probability distribution method as described in section 6.6
- **Count Congestion**:: Here we counted speed entries that are less than 3 km/hr for all buses & number of buses as confidence score to conclude congestion

Graphs can be seen in Figure 7.19 to 7.26.

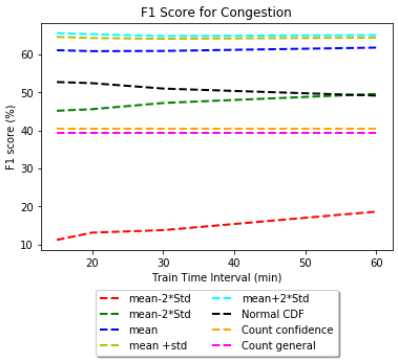

Figure 7.19: F1-Score(Size:250m^2)

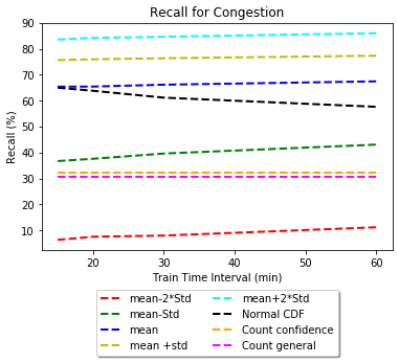

Figure 7.20: Recall(Size:250m^2)

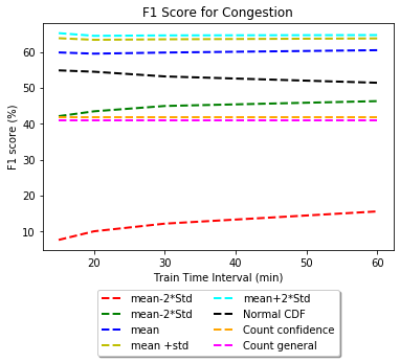

Figure 7.21: F1-Score(Size:200m^2)

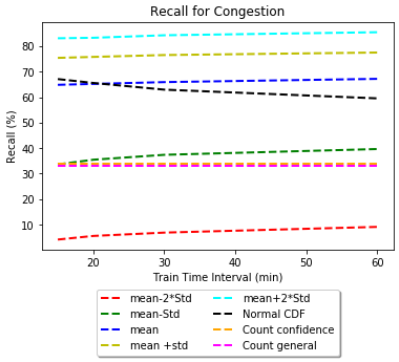

Figure 7.22: Recall(Size:200m^2)

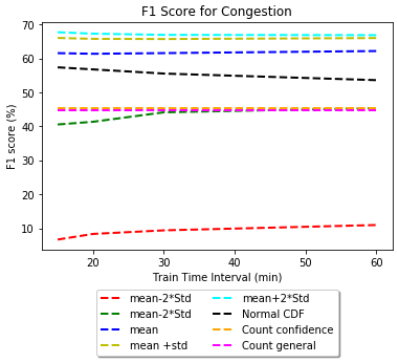

Figure 7.23: F1-Score(Size:150m^2)

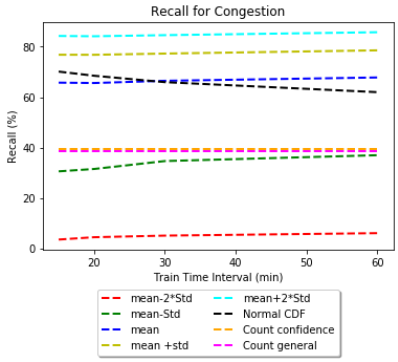

Figure 7.24: Recall(Size:150m^2)

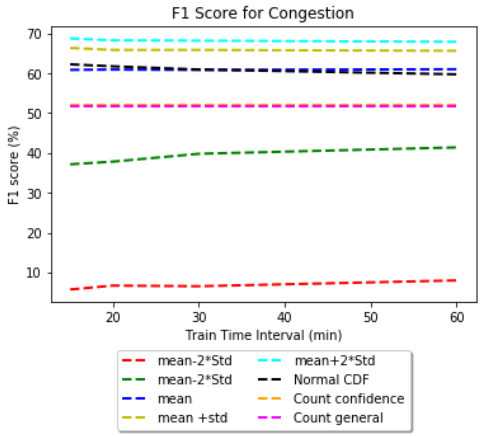

Figure 7.25: F1-Score(Size:100m^2)

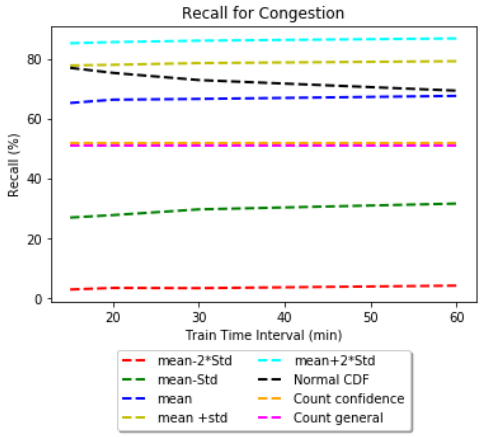

Figure 7.26: Recall(Size:100m^2)

From the above experiment, we identified that difference in training time does not impact much in the detection of Congestion. Both f1-Score and recall remain nearly constant for all the training time intervals. If at all there is some difference, it is close to minimal. This can be an indication that similar congestion trends exist for a longer duration and it can persist for an hour. Another observation that we can make is, as the size of the individual grid cell decreases, F1-score increases whereas recall remains nearly constant. This is because as size decreases the training gets restricted to a certain area, Hence we get much accurate trained parameters & in-turn better results.

7.6 Hotspots, Popular Stops & Jam Regions

Since we get geodata every 10 seconds, we monitored all such entries for 40 days. We identified regions from where we get maximum GPS signals and termed it as Hotspots. From our observation, we found that most of the entries are from Bus Depots that means buses send a signal by standing still at the bus Depots and waste fuels. Also, regions like Lodhi road and Barapullah road also gives maximum GPS locations, We have also identified popular bus stops which means the bus stops from where we get maximum GPS locations.

Table 7.1: Hotspots of Delhi(40 Days)

Cell ID	**Region**	**Entries**
24176	Kair Bus Depot	1211384 (1.46%)
21075	Intersection of Lodhi Road and Dyal singh college road	999908 (1.2%)
27317	Anand Vihar bus Terminal	991810 (1.19%)
10293	Badarpur metro station	927522 (1.11%)
31464	Seemapuri Bus Depot	832735 (1.00%)
21017	Manglapuri terminal	740332 (0.89%)
24920	Uttam Nagar Terminal	731707 (0.881%)
25352	Dichaun Kalan DTC Depot	730218 (0.880%)
19922	Barapullah Road	605600 (0.72%)

Table 7.2: Popular BusStops (40 Days)

Bus Stops	**Entries**
Kair Depot	169954 (3.4%)
Rajghat Cluster Depot	165590 (3.34%)
Badarpur Border	102148 (2.06%)
Dilshad Garden Depot Cluster	95063 (1.91%)
Uttam Nagar Terminal	84178 (1.69%)
Rithala Village	76539 (1.54%)
Madhu Vihar	72091 (1.45%)
Shastri Park	72041 (1.45%)
Mangla Puri Terminal	56612 (1.14%)
Uttam Nagar / A1 Janak Puri	53234 (1.07%)
Mori Gate Terminal	50591(1.02%)
Minto Road Terminal	46739 (0.94%)
ISBT Kashmere Gate Terminal	37256 (0.75%)
Nehru Place Terminal	35298 (0.71%)
Shivaji Stadium Terminal	32814 (0.66%)
Anand Vihar ISBT Main Road	32265 (0.65%)
Anand Vihar ISBT Terminal	30359 (0.61%)

Table 7.3: Jam Regions (Speed <0.5km/hr)(40 Days)

Cell ID	**Region**	**Entries**
24176	Kair Bus Depot	1014132 (1.95%)
21075	Intersection of Lodhi Road and Dyal singh college road	869661 (1.67%)
31464	Seemapuri Bus Depot	780322 (1.50%)
10293	Badarpur metro station	774988 (1.49%)
27317	Anand Vihar bus Terminal	762400 (1.46%)
25352	Dichaun Kalan DTC Depot	648750 (1.24%)
24920	Uttam Nagar Terminal	615789 (1.18%)
19922	Barapullah Road	596023 (1.14%)
21017	Manglapuri terminal	574386 (1.10%)
33717	BBM DTC Deport	499999 (0.96%)
13045	Tehkhand Bus Depot	444557 (0.85%)
24690	Bhagwan Mahavir Marg	418835 (0.80%)
29583	DTC Kashmere gate Bus Station	373905 (0.71%)
31693	Aradhak Marg (Outside Seemapuri bus Depot)	343765 (0.66%)
26833	Bus Depot Near Rajghat	330877 (0.63%)
28664	HC Sen Road & shyama Prasad Mukherjee Marg(Near Chandni Chowk)	260580 (0.50%)
29582	Lala hardev sahai marg	213350 (0.41%)

The tables 7.1, 7.2, 7.3 lists down Hotspots, popular bus stops & Jam Regions (Regions with speed less than <0.5km/hr) respectively in descending order of bus entries. Most of the entries we get in table 7.1 and 7.3 are from bus depots but few areas like Lodhi road, Barapullah road, Shyama Prasad Mukherjee Marg & Lala Hardev Sahai marg show maximum entries which are less than 0.5 km/ hr thus terming them as the severely congested regions of Delhi. The regions like Lodhi road [25] and Barapullah road [24] have already been termed as congested regions by authorities.

Chapter 8

Conclusion, Limitations, Future Work

8.1 Conclusion

Real-time congestion detection is a challenging problem. It is quite a difficult task to get accurate congestion results with limited input data which itself is sparse and has precision errors. Such kind of study is quite important for government authorities who have limited information access but wants a mechanism which gives near to accurate results.
In this thesis, we contributed towards detecting Congestion with limited spatio-temporal input data. In chapter 3, we looked at the current bus system prevalent in Delhi. In Chapter 6 we discussed the methodology used to detect congestion in real-time without latency. In Chapter 7 we discussed various experiments conducted in the Thesis study. We first discussed the effect of various algorithms. Subsequently, we looked at the effect of Grid Size, training time and test time for congestion evaluation. In our study, we reached 70% F1-score and 80% Recall at best with a simple statistical technique when compared to HERE maps jam factor.
We also saw how unsupervised learning algorithms failed in congestion detection due to sparse data. We also looked at Jam regions, Popular bus stops and Hotspots region of Delhi.

8.2 Limitations

- The Accuracy parameters which we measure in our study are based on the authenticity of the HERE maps as we did not find any other free map service provider which can solve our purpose. Our results might be much better at ground level because we take into account the speed behavior of buses which has direct relevance with the congestion.
- We can predict the congestion only at those areas where we find DIMTS cluster buses, for all other regions we cannot conclude congestion.
- We only tell about congestion in real-time, We do not deal with congestion that might arise in the near future i.e. we do not deal with the problem of congestion prediction.

We limited our study to the problem of congestion detection only and did not explore the time prediction of buses at bus stops keeping in mind the congestion pattern prevalent in real-time on

the roads.

8.3 Future Work

Our study opens opportunities for congestion prediction which can tell whether the bus which the user needs to pick can reach in time to the required bus stop keeping in mind the congestion pattern that exists at real-time or that might arise in near future based on previous data.
Our study mainly focuses on Congestion detection in real-time with minimal input data which can benefit the government and local authorities for analysis & development purposes. The field is now open for more contributions which can improve the accuracy of results using sparse & uneven data with low latency.

Bibliography

[1] Jinchao Song, Chunli Zhao, Shaopeng Zhong, Thomas Alexander Sick Nielsen, and Alexander V. Prishchepov. Mapping Spatio-temporal patterns and detecting the factors of traffic congestion with multisource data fusion and mining techniques. Computers, Environment and Urban Systems journal (ELSEVIER), Volume 77, September 2019, 101364. https://doi.org/10.1016/j.compenvurbsys.2019.101364

[2] Xiangjie Kong, Ximeng Song, Feng Xia, Haochen Guo, Jinzhong Wang, and Amr Tolba LoTAD: long-term traffic anomaly detection based on crowdsourced bus trajectory data. World Wide Web Journal, May 2018, Volume 21, Issue 3, pp 825–847, SpringerLink. https://doi.org/10.1007/s11280-017-0487-4

[3] Koushik Roy, Abdullah Al-Imam, Nur Islam and Adnan Firoze.A Traffic Jam Prediction Model for dynamic routes using Global Positioning Systems Data from Vehicles at Different Times of the Day. 2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS), 5-8 Dec. 2018, Toyama, Japan, 18674277, https://doi.org/10.1109/SCIS-ISIS.2018.00061

[4] Raffaele Carli, Mariagrazia Dotoli, Senior Member, IEEE, Nicola Epicoco, Biagio Angelico and Antonio Vinciullo. Automated Evaluation of Urban Traffic Congestion Using Bus as a Probe. 2015 IEEE International Conference on Automation Science and Engineering (CASE), Aug 24-28, 2015. Gothenburg, Sweden, 15508949, https://doi.org/10.1109/CoASE.2015.7294224

[5] Berk Anbaroglu, Benjamin Heydecker, and Tao Cheng. Spatio-temporal clustering for non-recurrent traffic congestion detection on urban road networks. Transportation Research Part C: Emerging Technologies (ELSEVIER), Volume 48, November 2014, Pages 47-65. https://doi.org/10.1016/j.trc.2014.08.002

[6] Selvaraj Vasantha Kumar and Ramaswamy Sivanandan. Traffic Congestion Quantification for Urban Heterogeneous Traffic Using Public Transit Buses as Probes. Periodica Polytechnica Transportation Engineering. 47(4), pp. 257-267. https://doi.org/10.3311/PPtr.9218.

[7] Rinchen Norbu Wangchuk, As Delhi Chokes on Its Traffic, the Solutions May Come From London, January 13, 2018, `https://www.thebetterindia.com/127696/delhi-chokes-traffic-solutions-London/`

[8] Congestion on Delhi roads has worsened–says a new analysis by CSE of latest Google map data, New Delhi, July 10, 2017, https://www.cseindia.org/congestion-on-Delhi-roads-has-worsened--6994

[9] Delhi Wikipedia, https://en.wikipedia.org/wiki/Delhi

[10] Transport in Dehi , Wikipedia , https://en.wikipedia.org/wiki/Transport_in_Delhi

[11] ILFS ECOSMART Chapter – 11: Review of Road Network and Transport System https://ccs.in/sites/default/files/files/Ch11_Review%20of%20Road%20Network%20and%20Transport%20System.pdf

[12] Story by Anirudh Raheja, DTC: Modern and dependable, October 27, 2016, http://commercialvehicle.in/dtc-modern-and-dependable/

[13] Wikipedia,Delhi Transport corporation , https://en.wikipedia.org/wiki/Delhi_Transport_Corporation

[14] PTI, 'DTC incurs Rs 9 cr monthly loss due to congestion on roads' ,December 17, 2017. https://timesofindia.indiatimes.com/city/delhi/dtc-incurs-rs-9-cr-monthly-loss-due-to-congestion-on-roads/articleshow/62105931.cms

[15] Adarsh Kapoor, A congestion plan in the name of decongesting Delhi, 7th June 2015, https://www.downtoearth.org.in/blog/a-congestion-plan-in-the-name-of-decongesting-Delhi-47931

[16] Revanth Ayala Somayajula, Real-time traffic congestion detection using images, Spring 2018, Iowa State University, Ames, Iowa https://lib.dr.iastate.edu/creativecomponents/2

[17] Ahmed Nidhal, Dr. Umi Kalthum Ngah, Dr. Widad Ismail, REAL-TIME TRAFFIC CONGESTION DETECTION SYSTEM, IEEE, 2014, https://doi.org/10.1109/ICIAS.2014.6869538

[18] Chan-Tong Lam, Hanyang Gao, Benjamin Ng, A Real-Time Traffic Congestion Detection System Using On-Line Images, 2017 17th IEEE International Conference on Communication Technology, https://doi.org/10.1109/ICCT.2017.8359891

[19] Li Wei a, Dai Hong-Ying, Real-time Road Congestion Detection Based on Image Texture Analysis, 137(2016) 196 – 201, Procedia Engineering. https://doi.org/10.1016/j.proeng.2016.01.250

[20] Joydip Dhar G. Garg, Real-time traffic congestion detection and optimal path selection using a smartphone, IEEE, 2014, https://doi.org/10.1109/POWERI.2014.7117645.

[21] Hnin Thant Lwin, Thinn Thu Naing, Estimation of Road Traffic Congestion using GPS Data, IJARCCE, Vol. 4, Issue 12, December 2015, https://doi.org/10.17148/IJARCCE.2015.41201

[22] Siuli Roy, Somprakash Bandyopadhyay, Munmun Das, Suvadip Batabyal, Sankhadeep Pal, Real-time traffic congestion detection and management using Active RFID and GSM technology, IEEE, 5-7 Oct. 2011, https://doi.org/10.1109/ITSC.2011.6082954

[23] Hu, Shan, and Wu, Jiansheng and Xu, Ling, Real-time traffic congestion detection based on video analysis,2012, JOURNAL OF INFORMATION &COMPUTATIONAL SCIENCE

[24] Ashish Mishra, Stretch near DND to be widened to ease congestion on Delhi's Ring Road, 19 Nov 2018. https://www.hindustantimes.com/delhi-news/stretch-near-dnd-to-be-widened-to-ease-congestion-on-delhi-s-ring-road/story-Kcj6k0OHMzYhIBIsBU8kkN.html

[25] Youth ki Awaaz, You Wouldn't Like To Attend This Social Gathering- Traffic Jams In Delhi https://www.youthkiawaaz.com/2011/10/you-wouldnt-like-to-attend-this-social-gathering-traffic-jams-in-delhi/