Short-Term Traffic Flow Prediction Using Google Traffic Data

By

Y. Ranawaka

2015/CS/111

This dissertation is submitted to the University of Colombo School of Computing

In partial fulfillment of the requirements for the

Degree of Bachelor of Science Honours in Computer Science

University of Colombo School of Computing

35, Reid Avenue, Colombo 07,

Sri Lanka

July 2020

Declaration

I, Y. Ranawaka (2015/CS/111), hereby certify that this dissertation entitled “Short-Term Traffic Flow Prediction Using Google Traffic Data” is entirely my own work and it has never been submitted nor is currently been submitted for any other degree.

…… ……

Date Signature of the Student

I, Prof. K.P. Hewagamage, certify that I supervised this dissertation entitled “Short-Term Traffic Flow Prediction Using Google Traffic Data” conducted by Y. Ranawaka in partial fulfillment of the requirements for the degree of Bachelor of Science Honours in Computer Science.

…… ……

Date Signature of the Supervisor

Abstract

The latest statistics show that by 2050, 60% of the world's population will live in cities, which is expected to double the number of cars around 2.5 billion by 2050. Therefore, today and in the near future, traffic congestion has become an inevitable situation in the growing metropolitan areas of the world. Therefore there is a need for the development of a methodology of obtaining reliable and consistent traffic jams data to manage and come up with alternative routes to avoid traffic congestions which allow authorities to provide efficient service to society.

This project carried out with the aim, to develop a methodology to integrate public data useful for short-term traffic flow prediction. The traffic data were extracted from capturing images of the Google map traffic layer in a particular area using Google Maps JavaScript API. All the predictions were achieved on experimenting only on the Sri Lankan contest where images are captured on a particular area of Colombo. These sequential images formed both the input and the prediction target are spatiotemporal sequences. Therefore a Convolutional LSTM (ConvLSTM) model approach was used for the traffic flow prediction, considering three main procedures which are based on predicting time 20 minutes, 30 minutes, and 1 hour in the future.

The experiment that was carried 20 minutes into the future, has gained results in the precision of 0.551 and F1-measure of 0.647. This study yielded a profitable and affordable methodology for short-term traffic flow prediction appropriately for Sri Lankan Context.

Preface

The prediction model that was provided by the Keras documentation [25] has been used in this study. A perceptual distance function that measures the similarity of two images, has been applied to the model which is provided by L. Biewald on Github [26]. Apart from the specific model and the code segment mentioned under Chapter 3 and Chapter 4, the methodology designing, experiments and analysis mentioned in this study is the work of the author of this document.

Acknowledgement

I would like to express my sincere appreciation towards my research supervisor Prof. K. P. Hewagamage and Co-supervisor Mr. K.V.D.J.P. Kumarasinghe for the continuous support on this project, for their motivation and guidance throughout this project.

I appreciate the feedback and motivation provided by my friends to achieve my research goals. This thesis is also dedicated to my loving family who has been an immense support to me throughout this journey of life

Table of Contents

List of Figures

List of Tables

List of Acronyms

3D	3-Dimension
ANN	Artificial Neural Network
ARIMA	Autoregressive Integrated Moving Average
ARIMA-GARCH	General Autoregressive Conditional Heteroscedasticity
CNN	Convolutional Neural Network
GRU	Gated Recurrent Units
KNN	K-Nearest Neighbor
LSTM	Long Short Term Memory
NN	Neural Network
RNN	Recurrent Neural Network
SARIMA	Seasonal Autoregressive Integrated Moving Average
SVM	Support Vector Machine

Chapter 1
Introduction

1.1 Background to the research

In recent years under busy schedules, traffic congestion is a growing problem faced by travelers all over the world. It causes many problems such as time and energy-wasting, air and sound pollution, human stress, weariness, and accidents. The ability to predict traffic flow is very useful to reduce occurring those problems and prevent unfortunate incidents, such as traffic jams or other abnormal conditions on the road.

The main reason for the increase in traffic congestion is the rapid pace of economic growth and the accompanying increase in the population of vehicles. There is a limit to how many additional roads can be added to meet the growing number of vehicles. There are broadly two factors, which affect the traffic congestion which are micro-level factors and macro-level factors [5]. Examples for the micro-level factors are many people and freight want to move at the same time, too many vehicles for limited road space. Examples of macro factors include regional economic dynamics, income levels, employment patterns, infrastructure investment, land-use patterns, trends in car ownership, etc. [5].

For planning and route decisions before travel, it is controversial that predictive information is more useful than real-time or historical information [1]. It is of great importance to understand traffic congestion in a specific area rather than for a single location for make better route choices, to support road network management, and systematically allocate resources [8].

There is no standard method of measuring traffic congestion on roads. Many definitions have been proposed to measure traffic congestion on road networks [5]. Because it can indicate traffic congestion by unifying various factors, such as travel time, speed, density, volume, etc. [3].

Various data collected through sensing devices, GPS tracking units, and mobile phones can be used to analyze traffic congestion. However, there are limiting factors in

monitoring the traffic congestion in the above methods such as high cost and limited deployment. [4]. Also in the context of transport management, it is usually focused more on the overall traffic situation of the road network than just on individual road links.

In most of the developed countries, traffic sensors, surveillance systems, and other detecting devices are being installed, and data gathering platforms are developed. But the installation of traffic detector systems is not applicable for developing countries because of the high cost of the installation and developing economy is more important [12].

Therefore there is a need for the development of a methodology of obtaining reliable and consistent traffic jams data. Hence in this study, public traffic congestion data extracted from Google maps will experiment.

Traffic congestion data will be extracted from capturing images of the Google map traffic layer in a particular area. In the Google map traffic layer, real-time traffic conditions are displayed by support of a set of color lines which indicate the speed limits based on the current traffic conditions [15].

Traffic flow prediction is mainly categorized into two directions which are long-term and short-term traffic prediction. The long-term traffic prediction targets, to predict the traffic flow for one or more days in the future and can be used for plan and design urban spaces, road networks and so on. The short-term traffic forecasting aims to predict recent traffic from a few minutes to possibly a few hours. Adaptive traffic signal control and route guidance systems are few of the many applications it can be used [20 and 18].

This study is to generate a traffic prediction model to predict the short-term future traffic situation of a given area. The main purpose of this study is to reduce traffic congestion by predicting the traffic flow within a specific period of time at a given road network. Hence both travelers and traffic management authorities have better benefits from having the predicted traffic flow information of road networks rather than having real-time traffic flow information. Thus from having predicted information on traffic flow in a particular area, travelers can avoid traffic congestion by using alternative road segments and traffic management authorities can have better decisions to control traffic congestion efficiently. Also, energy-saving and time-saving can be mentioned as the indirect benefits of having the predicted traffic flow information.

1.2. Research Problem and Research Questions

What is the possibility of devising a methodology which integrated public data for short-term traffic prediction?

In order to develop a methodology to integrate public data useful for short-term traffic prediction, there are three research questions.

Research questions,

- What is the most suitable way to collect traffic flow information?
- How to collect traffic flow information?
- What is the most suitable model building approach for short-term traffic prediction?

Since traffic congestions are being increased in urban cities, there is a need for a solution to avoid such traffic congestions. The project aims are to analyze traffic patterns and develop a methodology for short-term traffic flow prediction.

As mentioned earlier even though there is no standard method of measuring traffic congestion on roads many definitions have been come forward to measure traffic congestion.

Traffic flow information of roads network in a particular area is most important rather than the traffic state of an individual position or individual road links.

Therefore there is a need for the development of a methodology of obtaining reliable and consistent traffic jams data.

As traffic congestion data should be collected over successive periods of time, the data acquisition interval should be defined. Hence the data acquisition interval depends on how data are going to gather and possibilities. Although traffic congestion data is a time series, the model building approach should be selected also considering the dimension of the collected data.

1.3. Justification for the Research

The purpose of this study is to reduce traffic congestion by predicting the traffic flow within a specific period of time at a given road network. The latest statistics show that by 2050, 60% of the world's population will live in cities, which is expected to double the number of cars around 2.5 billion by 2050 [16]. Hence in recent years and also in the future, traffic congestion is an increasingly serious problem.

Both travelers and traffic management authorities have better benefits from having the predicted traffic flow information of road networks. Hence from having predicted information on traffic flow in a particular area, travelers can avoid traffic congestion by using alternative road segments and also traffic management authorities can have better decisions to control future traffic congestion efficiently.

Under hectic schedules, time-saving is a major concern for everyone. Managing and coming up with alternative routes to avoid traffic congestion would save a lot of time which is wasted on roads due to heavy traffic congestion. As well it is beneficial to know about future traffic conditions for traffic management authorities. Then those authorities can give efficient service to society and can save their own time by taking better decisions early.

The traffic congestion causes many problems such as air and sound pollution, human stress, weariness, and accidents. Hence the ability to predicting traffic conditions will be useful to those problems as well.

According to D. Schrank and his research team, due to traffic congestion in 2014, energy costs in 471 urban areas across the United States have increased. The wasted price tag was $160 billion and also 3 billion gallons of fuel were wasted [6]. Hence energy-saving is also a great concern for people throughout the world.
Therefore reducing traffic congestion by predicting the traffic flow is influential for the world.

1.4. Methodology

In this study traffic congestion data will be extracted from capturing images of the Google map traffic layer in a particular area using Google Maps JavaScript API with five minutes of data acquisition interval. Missing and unclear images are deleted from the data set. As the dataset included with sequential images, both the input and the prediction target are spatiotemporal sequences. Convolutional LSTM (ConvLSTM) is the selected model approach for the traffic flow prediction. In this study, three main methodologies are considered which are based on predicting time 20 minutes, 30 minutes and 1 hour in the future. In the first method, four sequential images (20 min) are inputted to the model and the next four sequential images (20 min) will be the output. In the second method, six sequential images (30 minutes) are inputted to the model and

the next six images (30 minutes) will be the output. In the third method, twelve sequential images (1 hour) are inputted to the model and the next twelve images (1 hour) will be the output. The overviews of the methodology and the model are shown in figure 1.1 and figure 1.2.

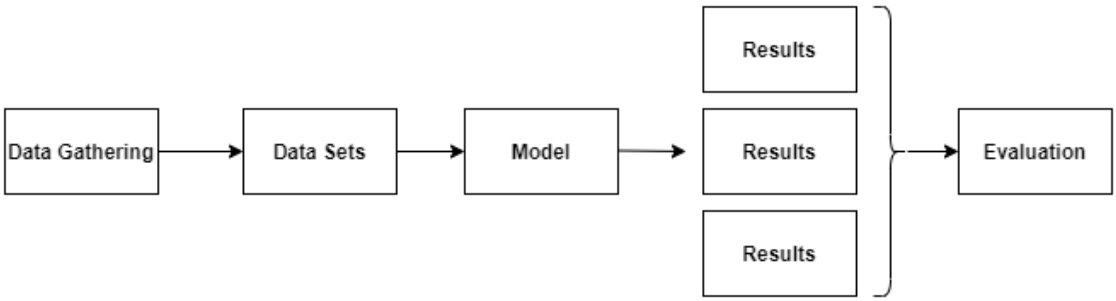


Figure 1-1-1: Overview of the Methodology

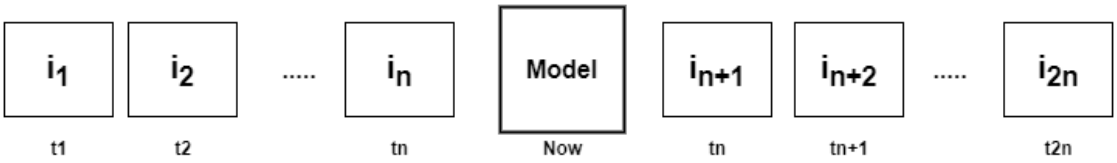


Figure 1-1-2: Overview of the Model

As travelers and traffic controllers focused more on the overall traffic situation of the road network than just on individual road links [2], various problems can be occurred such as high costs and distribution in large regions. Hence there is a need for the development of a methodology of obtaining reliable and consistent traffic jams data. As a solution, in this study traffic congestion data will be extracted from capturing images of the Google map traffic layer in a particular area. When a user enables Google Maps with location information, the user's mobile phone will send anonymous data back to Google to describe the speed of the user's movement. When there are thousands of Google users who are moving around an urban city, Google can display real-time traffic conditions in real time. Google will continue to merge this data and send it back to you for free in the Google Maps traffic layer [14]. Hence by combining these data shared by many thousands of people moving in a city, Google could give better estimates and information regarding real-time traffic conditions of roads. As well as the use of Google traffic data is an economical solution as it does not need any investment in equipment and distribution. Real-time traffic conditions are now available in Google map traffic layer which indicated by color lines based on the speed limits of a road considering the traffic conditions [15].

In recent years, most researchers have been trying to use deep learning based Neural Network methods to solve this traffic prediction problem and better performance has been reported than other methods [8, 10, and 11]. However, which kind of deep neural network is the most appropriate model for traffic flow prediction remains unsolved.

In this study, as images are captured from Google maps traffic layer for every five minutes, the dataset consists of an image sequence taken at a successive equally spaced time interval. Therefore, the collected data can be regarded as a time series. Thus the prediction model approach should have the ability to work with sequential data as well as spatiotemporal data.

In order to learn time series, long-term short-term memory (LSTM) neural network (NN) has been used in short-term traffic flow prediction and better performance has been reported [10, 11, 18, and 21]. As LSTM is not good for spatial vector as input, there is a need to extract the features from the images. To extract the spatial traffic data from images, the convolutional neural network (CNN) is best as its special trait of working with images [8, 21, and 22]. Since the disadvantage of CNN is that it is difficult to extract temporal features from the image, as a solution a combination of LSTM and CNN are used to model spatio-temporal data [22]. A combination of LSTM and CNN have used for short-term traffic prediction and better performance has been reported in [21], [23] and [24]. Although in recent years LSTM, CNN, and CNN + LSTM have been used for short-term traffic flow prediction, convolutional LSTM (ConvLSTM) which is also a different type of combination of LSTM and CNN, is not used yet in this field. Hence another target of this study is to see how the ConvLSTM can be applied to predict the short-term traffic flow.

1.5. Outline of the Dissertation

This dissertation is organized under six main chapters and references. The current chapter (chapter 1) consists of an introduction and motivation to the research along with the description of methodology and justification. Chapter 2 represents a literature review which explains the past studies related to the short-term traffic flow prediction. Chapter 3 describes the research design in detail. Chapter 4 explain the implementation details of this study. Chapter 5 includes the results and evaluation and provides a detailed analysis. Chapter 6 concludes the research study and describes future work which can be improved related to this research area.

1.6. Delimitation of Scope

In this study, the traffic flow information that was evaluated and presented by the Google is used. Hence, the perfection of their real-time data is beyond this study's control. It means that this study fully depends on Google's real-time traffic data.

When the images are being captured on Google maps traffic layer using the Google Maps JavaScript API, there is a limited number of API calls that can be made without being paid. Therefore, maintaining a low data acquisition interval and good map zoom level, all cities in Sri Lanka can't be covered without being paid by capturing images. Hence, this study experiments only on the Sri Lankan context and the images are captured on a particular Colombo area. This study does not experiment with different cities in Sri Lanka.

Even though the ConvLSTM has the ability to deal with sequential and spatiotemporal data, there have been some limitations to the structure of the ConvLSTM layer. In here, the combination of spatiotemporal features and other features is limited by the convolutional structure of a ConvLSTM unit. Thus, the study of the traffic flow with other socio-conceptual data such as weather information, holidays, accidents, and special events is limited in this study. Hence, the weather information and other factors would not be considered in this project.

1.7. Summary

This chapter represents the overall picture and importance of this study which includes the background to the research, research problems, justification, and methodology, the outline of the dissertation, definitions, and scope of the study. Firstly, the background of the research and research problems are clearly described. Then the research was justified and the methodology was briefly described. The dissertation was outlined. Then definitions and delimitation of the scope were presented.

Chapter 2

Literature Review

2.1. Introduction

This chapter details a review of short-term traffic prediction and builds a theoretical foundation based on past studies. This will also show the link between the research problems and the knowledge of past studies.

2.2. Traffic congestion

According to [13], there is no standard definition of traffic congestion and many definitions have been proposed to describe traffic congestion on roadways. These definitions can be divided into three categories which are cost-related, delay-travel time related and demand capacity related. In the definition related to delayed travel time, Anthony Downs' definition is that congestion can be defined as a situation where traffic is driving at a speed lower than the design capacity of the road.

2.3. Importance of predicted traffic information

According to [16], the latest statistics show that by 2050, 60% of the world's population will live in cities, which is expected to double the number of cars around 2.5 billion by 2050. Hence in recent years and also in the future, traffic congestion is a growing problem facing transportation authorities and travelers [16].

According to D. Schrank and his research team, due to traffic congestion in 2014, energy costs in 471 urban areas across the United States have increased. The wasted price tag was $160 billion and also 3 billion gallons of fuel were wasted [6].

The traffic flow information are categorized into three groups which are predictive, real-time and historical. Predictive is the use of historical or real-time information to predict future information. Real-time is based on the current information. Historical information is based on archived data. The predicted traffic congestion data has more advantages over real-time or historical traffic congestion data for pre-journeys and route planning [1].

Since most of the factors such as limited road-space, growing vehicle population and bad weather which affect traffic congestion cannot be controlled, one of the best ways to mitigate traffic congestion is to use predicted traffic congestion information from having better management for the transportation systems and informing travelers to avoid predictive traffic congestion [3 and 8].

2.4. Factors

The physical capacity constraint is one of the main reasons for traffic congestion on the roads. It means that the existing infrastructure of roads is insufficient to meet the traffic requirements. Also, there are other main aspects that leverage traffic congestion which are vehicle crashes, special events, constructions, rough climates, and special time periods [2, 3].

According to [5], there are generally two factors, which affect the traffic congestion which are micro-level factors and macro-level factors. Examples for the micro-level factors are many people and freight want to move at the same time, too many vehicles for limited road space. Examples of macro factors include regional economic dynamics, income levels, employment patterns, infrastructure investment, land-use patterns, trends in car ownership, etc. [5].

2.5. Measuring

According to [13], traffic congestion is measured in different methods and those measures are further categorized as level of service, ratio measures, indices, and basic measures.

In the study [5], different mechanisms are mentioned to measure traffic congestion. Those are travel delay, traffic density, average speed, travel time index (TTI), travel rate index, level of service, etc. Congestion is a speed reduction operation, which leads to higher vehicle operating costs, emissions of air pollutants, fuel consumption and time loss. Threshold setting directly related to driving speed is the most appropriate. According to the congestion indices evaluation matrix in [5], speed-related measures fulfill most of the criteria that are ease of data collection, continuous value, stability, repeatability, simplicity and the magnitude of congestion. The most valuable advantage of using speed related measures is that the data can be collected using the latest technologies such as GPA which is an effective and economical technology [5].

2.6. Measuring techniques

Various data collected through sensing devices, GPS tracking units, and mobile phones can be used to analyze traffic congestion [2]. However, there are limiting factors in monitoring the traffic congestion in the above methods such as high cost and limited deployment. [4]. Also in the context of transport management, it is usually focused more on the overall traffic situation of the road network than just on individual road links.

In most of the developed countries, traffic sensors, surveillance systems, and other detecting devices are being installed, and data gathering platforms are developed. Since in the developing countries, developing economy is more important as there are many other important requirements to be done, the installation of traffic detector sensors and cameras is not suitable due to the high cost of installation [12].

2.7. Google data

In the study [12], Kumarage and sakitha mentioned that using Google traffic data is an economical solution as there is no need to spend any cost for equipment installation or calibration. Also, the study [12] stated that Google could give better coverage in Sri Lanka out of all available traffic information providers.

When considering the Google maps, according to [15], in the Google map's traffic layer, traffic conditions are displayed by support of a set of color lines which are green, orange, red and dark red colors. The color lines indicate the speed limits based on traffic congestion. Green color indicates there is no traffic delay, orange color indicates there are medium amounts of traffic, red color indicates there are traffic delays, dark red color indicates that there are stop-and-go traffic and gray color indicates that there is no data currently available [15].

Although the use of Google real-time traffic data is an economical solution, there is another important problem with how Google map's traffic layer is guaranteed that they provide us accurate information on real-time traffic conditions.

According to [14], when a user enables Google Maps with location information, the user's mobile phone will send anonymous data back to Google to describe the speed of the user's movement. When there are thousands of Google users who are moving around

an urban city, Google can display a good picture of real-time traffic conditions in real time. Google will continue to merge this data and send it back to you for free in the Google Maps traffic layer. Hence by combining these data shared by many thousands of people moving in a city, Google could have better real-time traffic information on roads [14].

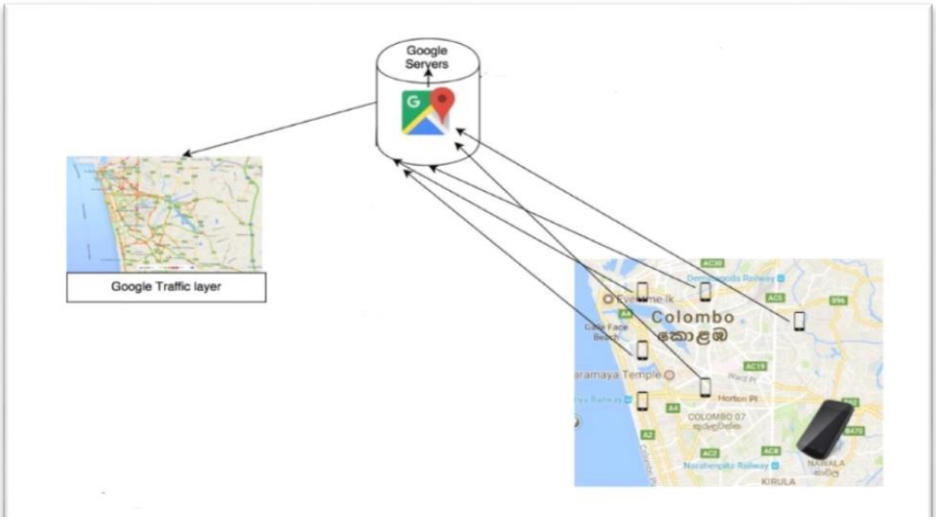


Figure 2-1: According to [12], How Google provides traffic information on the map [12]

In [17], Google privacy policy says that a user's location can be found with different accuracy through IP address, sensor data, GPS, and information that can be collected in the vicinity of the user's device (cell towers, Wi-Fi access points, and Bluetooth-enabled devices).

According to [12] and [17], when a user agrees to the privacy policy of Google, it allows the company to use any information of users such as GPS location Wi-Fi or cell tower MAC addresses, the information provided by sensors such as accelerometer and gyroscope. This clause allows Google to use the information sent by sensors embedded in smartphones and similar devices to determine the speed of users and the direction of travel. Hence this clause allows Google to identify travel patterns and travel modes of users [12, and 17].

According to [12], the algorithm or methodology of estimating traffic information is not revealed by Google and kept black-boxed to the public. But, the traffic information representation is given to the public via Google Maps and Google Maps APIs [12].

2.8. Models

According to [10], [11], [21], and [22] traffic flow prediction model approaches are categorized under parametric models and nonparametric models.

2.8.1. Parametric Models

Parametric model approaches are designed for given models with a fixed structure restricted by some assumptions [10]. According to [10], [11], and [21], the ARIMA model, the Kalman filter (KF) and exponential smoothing (ES) can be mentioned as examples for the parametric model approaches. Among them, the ARIMA model is mostly recognized and commonly used as a framework for building traffic prediction models. After that, time series prediction models were proposed based on ARIMA such as Kohenen-ARIMA, subset ARIMA, seasonal ARIMA, fractional ARIMA, vector autoregressive ARIMA, and those models also have used for traffic forecast [10, 11, and 21].

Vinay B and et al studied the ARIMA model along with the variations defined as SARIMA and ARIMA-GARCH to discover the its suitability and potential applicability for traffic prediction. The study [9] concluded that ARIMA-GARCH is better than ARIMA and SARIMA models for traffic flow prediction.

Table 2-1: Summary of ARIMA variations [2]

SARIMA	Seasonal ARIMA	Data suitable for short-distance repetitive patterns
FARIMA	Fractional ARIMA	Consider long-distance repetitive patterns
MARIMA/ARIMAX	Multivariate ARIMA	Include other time series as dependent variables
k-factor GARIMA	Gegenbauer Polynomials ARIMA	Consider the short-distance and long-distance dependence of different K data frequencies
Switching ARIMA	Different ARIMA models are fitted	Different ARIMA for different characteristic

According to study [10] and [21], there are few traits in parametric model approaches. Firstly, those models are usually simple and easy to understand. Secondly, the solutions of parametric models are easier than nonparametric models and take less time to execute. The parametric model approaches with traffic flow data which follows regular variations have higher prediction accuracies, but the non-linear and stochastic nature of traffic flow, may drop the accuracy of prediction [10 and 21].

Due to those problems of parametric model approaches, in recent years most researchers are willing to use non-parametric model approaches for short-term traffic prediction.

2.8.2. Non-parametric Models

The parameters and structures of non-parametric models are not fixed. Hence, these models are more flexible and complex compared to the parametric model approaches. Non-parametric regression, Kaman filter, KNN, SVM and ANN can be stated as examples for the non-parametric model approaches [10, 11, 21 and 22].

In most of the researches, ANN is one of the most trending methodology in traffic forecasting. In the study [1], a comparative analysis between ANN and SVM has conducted to predict the traffic speed. For the ANN model, the performance quality of the network depends according to the amount of available data in training phase. Hence, when there is less available data and the training data does not represent the entire data well, an alternative method is needed for prediction. According to the results of [1], when the quality and quantity of training data are small, the SVM performs better than ANN. But as the time interval of training data gradually increased by days, artificial neural network also started to improve and performed well than the support vector machine in a gradual manner. [1].

A. Padiath and his team compare the performance of historical methods, artificial neural networks and model-based methods using Kalman filter technology for short-term traffic prediction [3]. Data were collected from a video camera and the traffic density has been used to represent the traffic congestion which indicates the quantity of vehicles in a given distance of road segments. The study [3] has concluded that the model-based approach outperforms the historical method and ANN.

The study conducted in [7] also used the Neural Network approach for vehicle speed prediction. The data were collected from 52 sensors installed on a particular road for 31 days at every 5-min intervals. The conclusion of the study [3] shows that the proposed

neural network model can detect fluctuations in traffic dynamic and predict speed of vehicles correctly for up to 30 minutes.

In the study [19] Kranti Kumar and et al have conducted short-term traffic flow prediction using ANN for a non-urban highway and have gained good performance.

In recent years, most researchers have been trying to use deep learning-based Neural Network methods to solve this traffic prediction problem and better performance has been reported [8, 10, 11, 21, 23 and 24]. However, which kind of deep neural network is the most suitable model for traffic flow prediction remains indecipherable.

In the study [10], GRU and LSTM Neural Networks have been used to predict traffic flow. The study [10] reviews that among most deep learning-based methods, LSTM NN has been reported better performance than some common prediction models. As a result of the complexity in the structure of LSTM NN, it takes a long time to train. To speed up training, in 2014, GRU Neural Networks was proposed as a modification of LSTM. According to the study [10], it was the first attempt that GRU NN is applied to predict traffic flow, which GRU NNs showed slightly better performance than LSTM NNs.

Zheng Zhao and et al have introduced a novel short-term traffic flow prediction model based on the LSTM network in the study [11] year 2017. From using video cameras, traffic data is collected from more than 500 observatories with a rate of occurrence of 5 minutes. Data on traffic factors such as the number of vehicles, average vehicle speed, and lane occupancy are collected within six months. The LSTM network performance has been compared with other models such as Sacked Auto-Encoder (SAE), SVM, RNN, RBF Neural Network and ARIMA model. This comparison proved that LSTM NN performed better than most of the other models [11].

In the study [8], a deep CNN based method has been proposed for traffic flow prediction which learns traffic flow as images. The research [8] was conducted under to two main strategies, which in the first strategy network traffic information were converted into images considering space and time of the transportation network as the dimensions. The second strategy is to evaluate the CNN using created images for traffic flow prediction. Also, a comparison has been evaluated with other prediction model approaches such as KNN, random forest, ANN and deep learning architectures that are stacked auto-encoder (SAE), RNN, and LSTM. This study [8] shows that the introduced CNN based method outperforms mentioned other model approaches. As claimed by researches, the proposed

method can have more interesting extensions such as a combination of LSTM NN and CNN.

In the study [21], Haiyang Yu and et al have conducted a research study for traffic forecasting from inheriting the benefits of both deep CNN and LSTM NN. The data are gathered from GPS devices in moving cars on a road network and then average traffic speeds are mapped into a sequence of images. In this study [21], the traffic information is gathered from the floating cars with GPS devices and then average traffic speeds are converted into a series of images. With the intention of capturing the spatial characteristics of network-wide traffic images, deep CNN has been exploited and to learn the temporal dynamics characteristics of traffic flow, LSTM neural network has been used. Also, the comparison between this methodology and other methodologies such as SVM, LSTMs, DCNN, and SAEs was conducted. Researches have concluded that the methodology of this study outperforms the other mentioned methodologies.

Another approach of using a combination of LSTM NN and CNN to achieve short-term traffic flow prediction has been proposed by Wu Yuankai and et al in the study [23]. In this case, one dimensional CNN is adopted to capture the spatial characteristics of traffic flow and two LSTM NN layers are applied to gain the temporal dynamics characteristics of traffic flow. For the datasets of this study, traffic flows of 33 locations have been utilized. As a comparison, performances of other algorithms like LSTM, SAE, and NN have been analyzed. Also, the results of this study have indicated that a combination of LSTM NN and CNN neural networks have better performances for traffic flow prediction.

Although in recent years, most researches used the combination of LSTM NN and CNN for traffic flow prediction, in 2018 Shengdong Du and et al in the study [24] have come up with a method called hybrid multimodal deep learning framework (HMDLF) that is included a combination of GRU NN and CNN with the attention mechanism. In this study also, one dimensional CNN is adopted to capture the spatial features of traffic flow. But due to the simplicity and having fewer parameters of the GRU neural network, instead of using LSTM NN layers, GRU NN layers are utilized to gain the temporal dynamics features of traffic flow. According to the results of this study, the proposed HMDLF model is satisfied with better accuracy and effectiveness than other commonly used models.

In the study [22], Usman Ali and Tariq Mahmood have conducted a systematic literature review about the deep learning-based methods for short-term traffic flow prediction. This review has exposed that the LSTM NNs are the most commonly used deep learning model to predict the short-term traffic flow due to its special capability of managing the sequential traffic flow data. Furthermore, as CNN has the strength to manage the spatial traffic flow data, this review reveals that a hybrid of deep CNN and LSTM NN can be adopted to achieve better results for short-term traffic flow prediction.

2.9. Summary

This chapter mainly considered the past research works that related to this research area. The details of this chapter helped in identifying various problems and solutions appropriate to this study.

Chapter 3

Design

This research is designed to deal with the following objectives.

- Obtain the public traffic flow information
- Integrate such data into a model suitable for the prediction.
- Evaluate the model for traffic flow data sets.
- Analyze the obtained results of traffic flow.

3.1. Overview of the project

An overview of the research design is given in figure 3.1 which elaborates the above objectives.

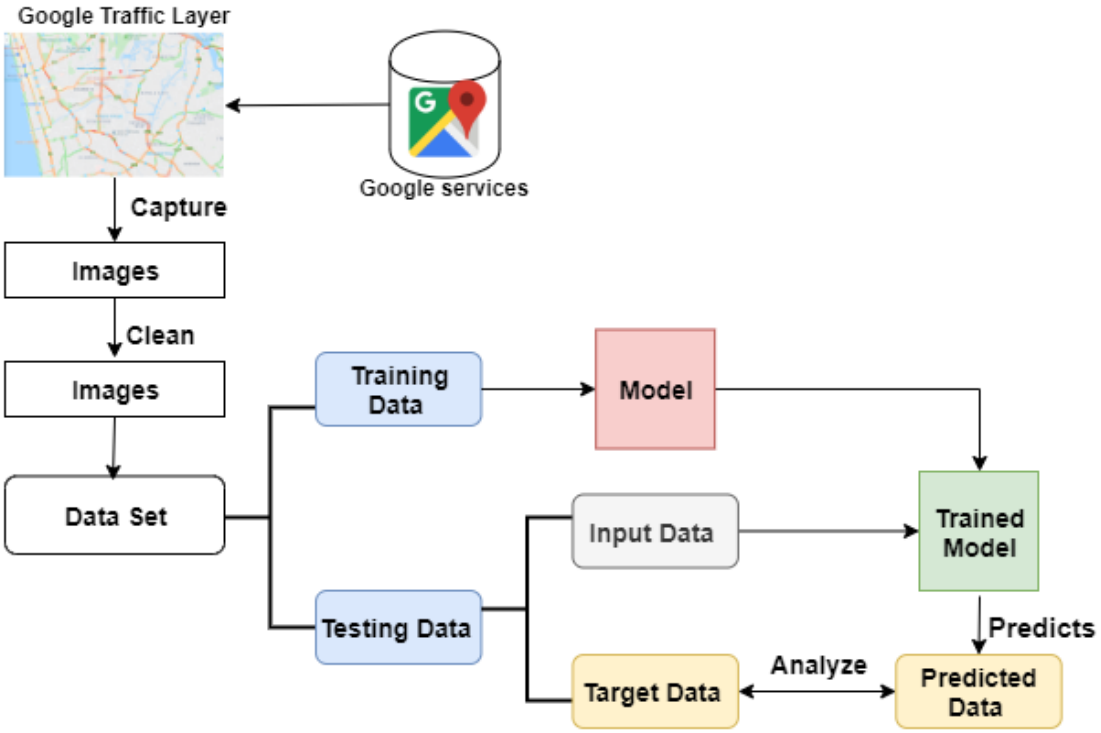


Figure 3-1: High level overview of the research

As seen in the figure 3.1, this research is started with obtaining the traffic flow information as sequences of images. After dropping missing and unclear images, the

data set is created appropriate to the prediction model. Then the model is trained according to different experiments with training data sets and the analysis part is conducted using testing data sets.

3.2 Gathering Traffic flow information

According to the literature review in chapter 2, as an economical solution and as having better information about real-time traffic flow information, Google Maps is used to obtain the traffic flow information in this research. But those real-time traffic data are not directly available. However there is a better geographical representation of real-time traffic congestion in Google Maps Traffic Layer which is publicly available. Hence as seen in figure 3.1, public real-time traffic flow data are obtained from the Google Maps Traffic Layer as sequences of images. In order to obtain the traffic layer map, Google Maps JavaScript API is used.

Using Google Maps JavaScript API, images on Google maps traffic layer, are captured considering in a particular area of Colombo, with 5 minutes of traffic acquisition interval. According to [15], in the Google map traffic layer, real-time traffic conditions are displayed by support of a set of color lines which indicate the speed limits based on the current traffic conditions. Green color indicates there is no traffic delay, orange color indicates there are medium amounts of traffic, red color indicates there are traffic delays, dark red color indicates that there are huge traffic congestion and gray color indicates that there is no data currently available [15].

Table 3-1: Obtaining traffic flow information

Centered Location	Latitude = 6.8889298 and Longitude = 79.8710876
Zoom level of Google maps	15.5
Data acquisition interval	5 Minutes

3.3 Suitable Model Approach

Since the data set contains sequences of images that represent the traffic flow information geographically, there should be a capability in the model approach to deal with time-series data and as well as spatial features.

According to the literature review in chapter 2, as a special neural network of RNN, LSTM NNs have been utilized in several previous studies for sequence modeling due to its nature of remembering information. But the problem is that the LSTM NN is not good for spatial vector as input. Hence there is a need to extract the features from the images. To extract the spatial features from images, the CNN is most suitable due to its nature of working with images. Hence there is a need to deal with both sequential and spatiotemporal data. In this research, ConvLSTM NN is used as the prediction model approach.

3.3.1 Long Short-Term Memory

Since the traffic state of several steps earlier can be affected by the current traffic state, the LSTM NNs are the most powerful neural network to predict traffic flow due to the capability of learning earlier dependencies.

Major problems of traditional RNNs are vanishing and exploding gradients which happen when the number of time steps is increased. The LSTM was introduced with three gates which are output gate, input gate (update gate) and forget gate. The forget gates decide the amount of information from the past should be remembered. The update gate takes new inputs and decides how much of the unit is added to the current state. The output gate takes all calculated results and decides which part of the current cell makes it to the output. The structure of a LSTM Unit is described in figure 3.2.

The key equations of the LSTM cell are shown in (1) below, where *W* is denoted as the weight matrices, '*b*' are denoted as the bias terms, σ stands for the sigmoid function, *tanh* is the hyperbolic tangent function and '∘' denotes the Hadamard product (element-wise product). In the equation (1), h_t is the hidden state, ĉ is the cell gate, c_{t-1} is the previous cell state, h_{t-1} is the previous hidden state, i_t is the input gate, f_t is the forget gate, o_t is the output gate, and c_t is the memory cell.

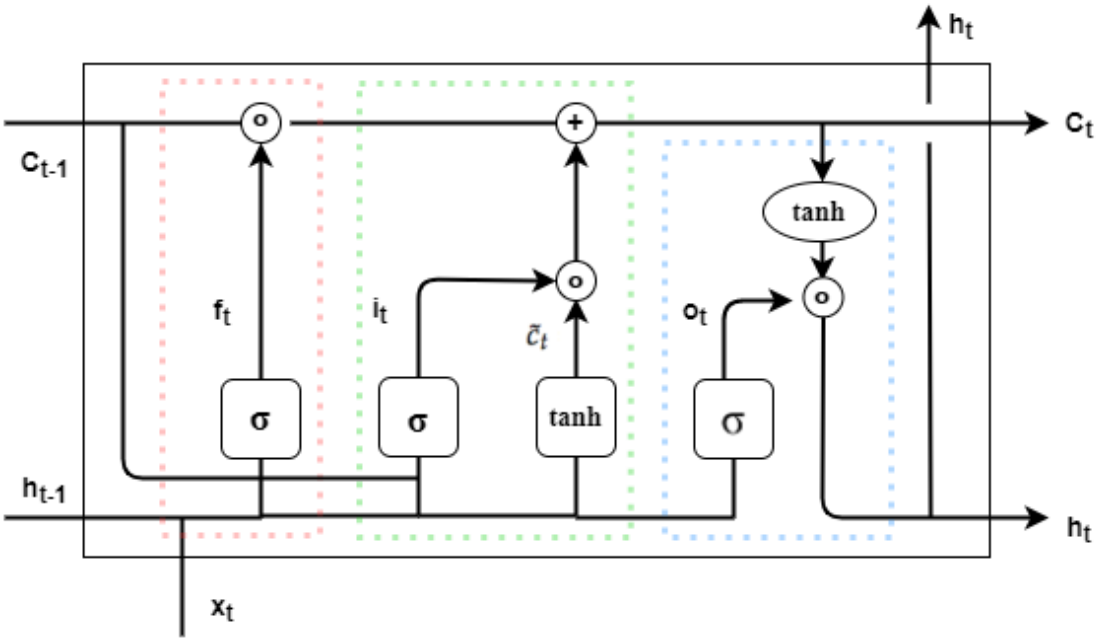


Figure 3-2: Structure of a LSTM Unit

$$
\begin{aligned}
i_t &= \sigma\left(W_{ix}x + W_{ih}h_{t-1} + W_{ic}\ c_{t-1} + b_i\right)\\
f_t &= \sigma\left(W_{fx}x + W_{fh}h_{t-1} + W_{fc}\ c_{t-1} + b_f\right)\\
\hat{c}_t &= tanh\left(W_{cx}x + W_{ch}h_{t-1} + b_c\right)\\
c_t &= f_t \circ c_{t-1} + i_t \circ \hat{c}_t\\
o_t &= \sigma\left(W_{ox}x + W_{oh}h_{t-1} + W_{oc}\ c_{t-1} + b_o\right)\\
h_t &= o_t \circ tanh\left(c_t\right)
\end{aligned}
\qquad (1)
$$

3.3.2 ConvLSTM Network

Since LSTM NN should have one-dimensional input data, it cannot be used to handle spatial sequence data such as videos and images. Convolutional LSTM (ConvLSTM) network is an extension of LSTM networks which also have the capability of learning long-term dependencies. Therefore, the gates that are in the LSTM cell are also in the ConvLSTM cell. But internal 1D matrix multiplications in the LSTM cell are replaced with convolution operations in the ConvLSTM cell. Hence ConvLSTM networks allow multidimensional data with convolutional operations in each gate. The structure of a LSTM Unit is described in figure 3.3.

The key equations of the ConvLSTM cell are shown in equation (2) below, where '*' denotes the convolution operation, '∘' denotes the Hadamard product (element-wise product). In the equation (2), H_t is the hidden state, C_{t-1} is the previous cell state, H_{t-1} is the previous hidden state, C_t is the memory cell and rest of the notations are same as Equation (1).

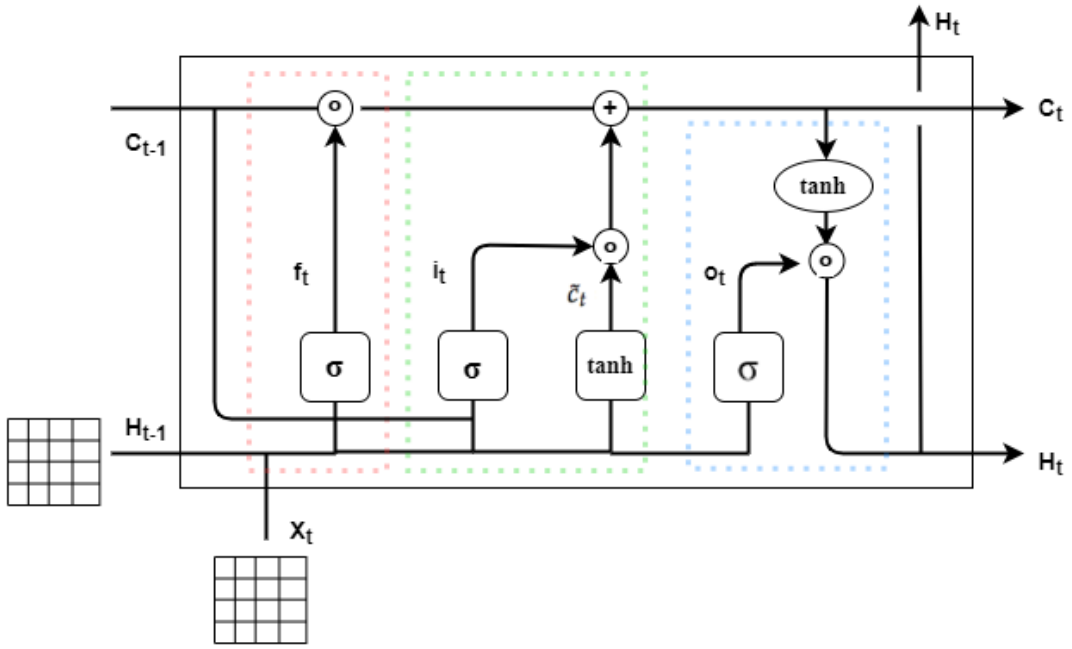


Figure 3-3: Structure of a ConvLSTM Unit

$$
\begin{aligned}
i_t &= \sigma\left(W_{ix} * X + W_{ih} * H_{t-1} + W_{ic} \circ C_{t-1} + b_i\right) \\
f_t &= \sigma\left(W_{fx} * X + W_{fh} * H_{t-1} + W_{fc} \circ C_{t-1} + b_f\right) \\
\hat{c}_t &= tanh\left(W_{cx} * X + W_{ch} * H_{t-1} + b_c\right) \\
C_t &= f_t \circ C_{t-1} + i_t \circ \hat{c}_t \\
o_t &= \sigma\left(W_{ox} * X + W_{oh} * H_{t-1} + W_{oc} \circ C_{t-1} + b_o\right) \\
H_t &= o_t \circ tanh\left(C_t\right)
\end{aligned}
\qquad (2)
$$

3.4 Model Evaluation for the Traffic Flow Data

The ConvLSTM network model which is in the Keras documentation [25] is used by adapting to this scenario.

This model is included with four ConvLSTM layers. Further, there is a BatchNormalization layer each after every ConvLSTM layers. Finally, a Conv3D layer has been added to the model as shown in figure 3.4.

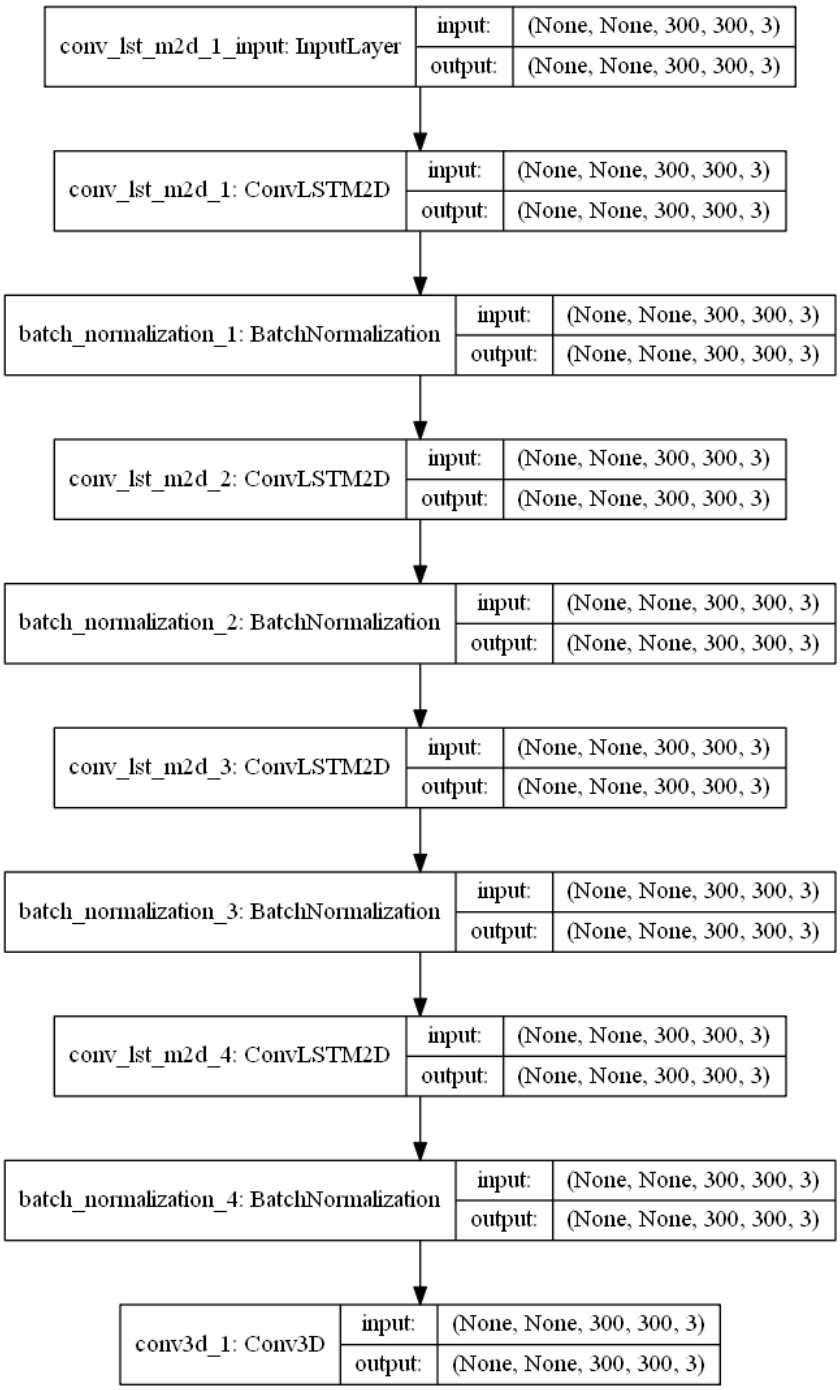


Figure 3-4 : Model Summary

A sequence of images is inputted into the model and the output is also a sequence of images that have the same length of the inputted sequence. The model architecture that was experimented with is shown in figure 3.5.

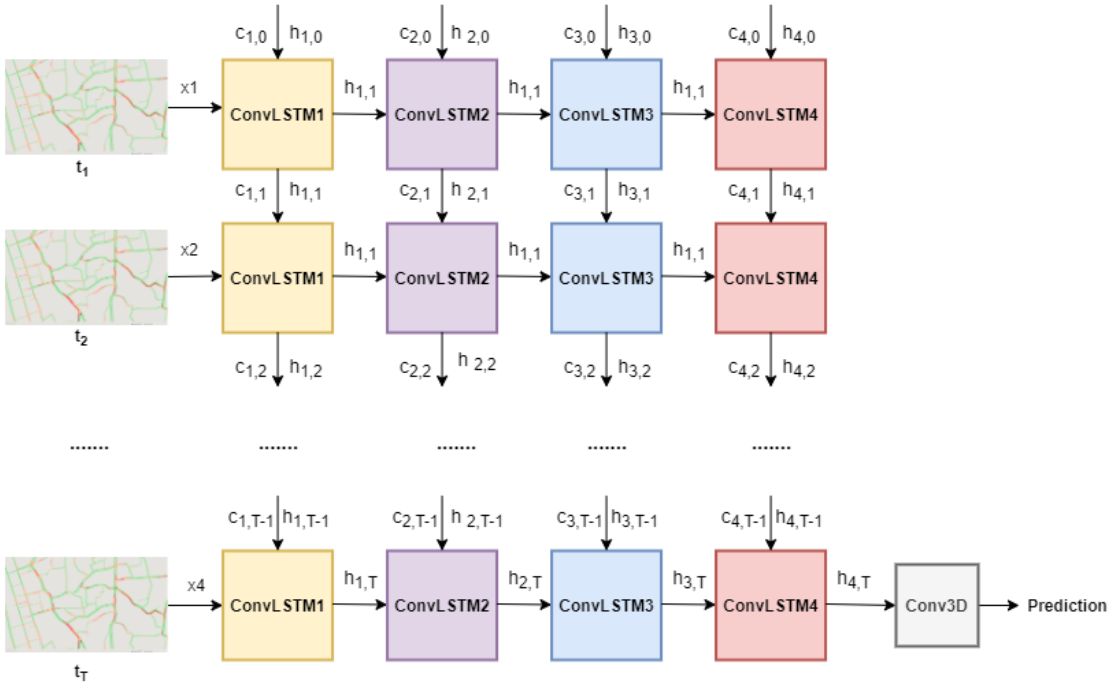


Figure 3-5 : Architecture of the ConvLSTM network

3.4.1 Twenty Minutes into the Future

In here sequences are considered from 15 minutes to 15 minutes which means that one sequence contains 4 images. As there are 4 input images in a sequence, there are also 4 target images in a sequence. When those 8 images are considered all together, it is called a batch.

As shown in figure 3.6, this experiment is designed to predict the next image sequence 20 minutes into the future.

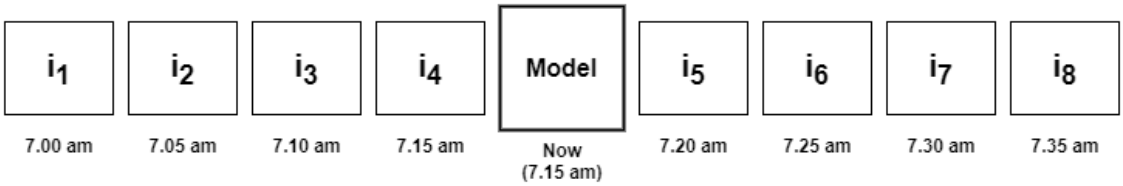


Figure 3-6: Twenty minutes into the future

- As the first experiment, the model is trained by using images that were captured within a specific time on a specific day. Then the plan is to conduct the testing process using images that were captured within the same specific time the next day.

- As the second experiment, the model is trained by using images that were captured within a specific time on a specific 5 days a week. Then the plan is to conduct the testing process using images that were captured within the same specific time on 2 days next week.

3.4.2 Thirty Minutes into the Future

In here sequences are considered from 25 minutes to 25 minutes which means that one sequence contains 6 images. Hence one sequence contains 6 images within 25 minutes. As described in previous design 3.4.1, there are 6 input images and 6 target images in a batch. This experiment is designed to predict the next image sequence 30 minutes into the future as shown in figure 3.7.

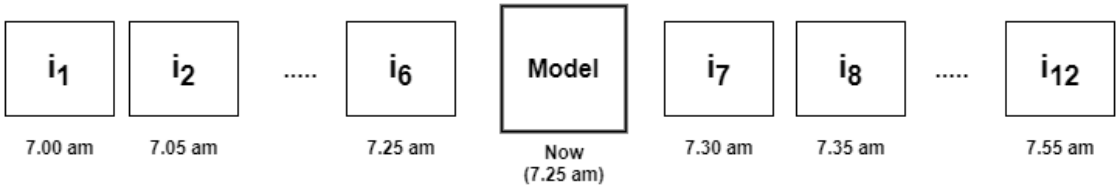


Figure 3-7: 30 minutes into the future

3.4.3 One Hour into the Future

In here sequences are considered from 55 minutes to 55 minutes which means that one sequence contains 4 images. Hence one sequence contains 12 images within 55 minutes. As described in previous design 3.4.1, there are 12 input images and 12 target images in a batch. This experiment is designed to predict the next image sequence one hour into the future as shown in figure 3.7.

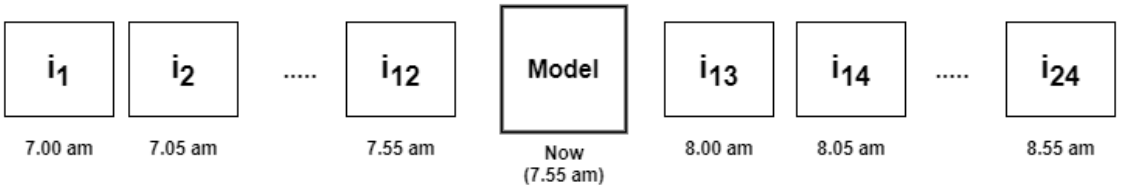


Figure 3-7: One hour into the future

In this scenario, the model is also `trained by using images that were captured within a specific time on a specific 5 days a week. Then the plan is to conduct the testing process using images that were captured within the same specific time on 2 days next week.

3.5 Results Evaluation

When there is a predicted image on a time step, there should be a methodology to compare the predicted image with the actual image. Hence predicted image should be evaluated to see how much the traffic flow is predicted correctly. In that case, the confusion matrix is used as shown in table 3.2. Considering traffic color codes as shown in table 3.3, true positive, false positive, true negative and false negative values are calculated. Using these values, the measurements that are sensitivity, specificity, precision, F measure, and accuracy are calculated and analysis is conducted using these measurements.

Table 3-2: Confusion Matrix

		Actual	
		Positive Traffic Congestion (Red / Orange)	**Negative** No Traffic Congestion (Green)
Predicted	**Positive** Traffic Congestion (Red / Orange)	True Positive TP	False Positive FP
	Negative No Traffic Congestion (Green)	False Negative FN	True Negative TN

Table 3-3: Considered Traffic color codes

On a particular Position in the Image	
Green color	There is no traffic congestion
Orange / Red colors	There are traffic congestion

- **Sensitivity** is a measure of actual positives that are correctly identified which is also known as recall or true positive rate. Calculated by equation (3).

$$\text{Sensitivity} = \frac{TP}{TP + FN} \qquad (3)$$

- **Specificity** is a measure of actual negatives that are accurately recognized which is also known as the true negative rate. Calculated by equation (4).

$$\text{Specificity} = \frac{TN}{TN + FP} \qquad (4)$$

- **Precision** is the positive predictive value which is the probability that a positive test happen. Calculated by equation (5).

$$\text{Precision} = \frac{TP}{TP + FP} \qquad (5)$$

- **F-Measure** provides a measurement that combines both precision and sensitivity into a single measure and captures both properties. Calculated by equation (6).

$$\text{F1 Measure} = 2 . \frac{\text{Sensitivity x Precision}}{\text{Sensitivity + Precision}} \qquad (6)$$

- **Accuracy** is the total number of correct predictions from total number of predictions made. Calculated by equation (7).

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \qquad (7)$$

Initially, the actual images and the predicted images are in RGB format. For easy of the images evaluation process, those images are converted to the HSV format. RGB stands for red, green and blue colors and these colors are made through the light which can have any intensity values. The HSV color model is closer to how humans see the colors. HSV stands for hue, saturation, and value. It describes the color (H) in terms of their amount of gray (S) and their brightness value (V). Therefore, here, it is easy to use the HSV color model for the image evaluation process rather than using the RGB color model.

Even though the images are evaluated using measurements that were described above, it is also better if it can be represented on images as to how correct predictions and errors were made. In such a case, the correct predictions and the errors of predicted images can be represented using a black mask and logic operations as elaborated in figure 3.8 and figure 3.9.

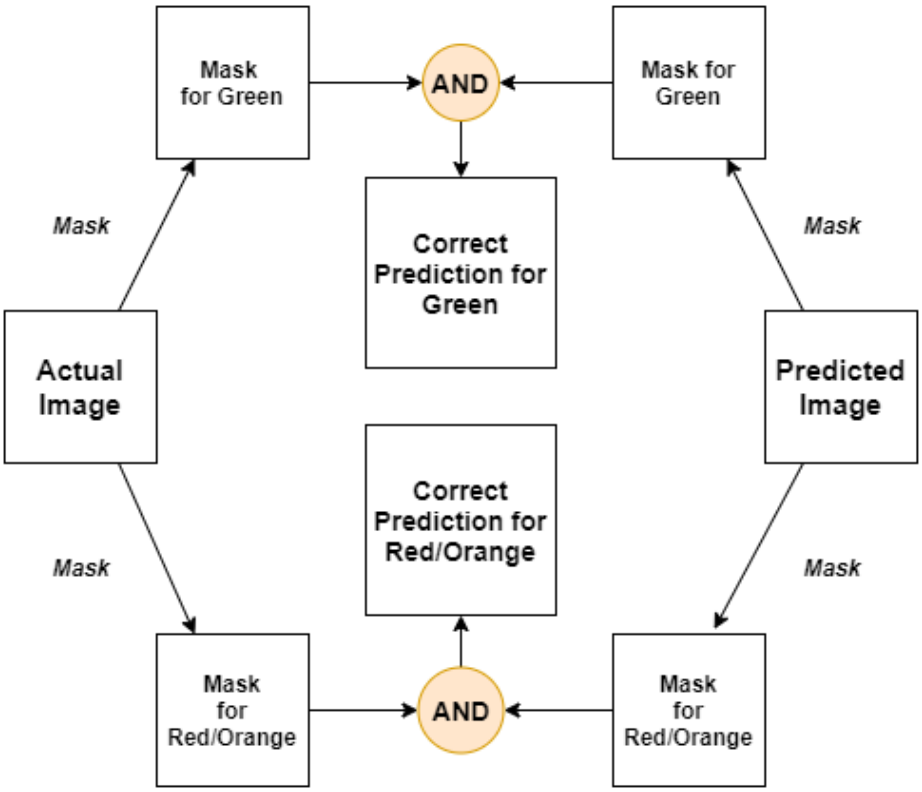


Figure 3-8: Process of representation the correct prediction with black Mask

3.6 Summary

This chapter mainly considered the description on the research design. First, this chapter described the overview design of the research. After that, the design of data gathering, data sets creation, model evaluation, and analysis were described in detail.

Chapter 4

Implementation

This chapter explicates the implementation details of the proposed methodology. The implementation of this study was designed to deal with the following objectives.

- Gathering the public traffic flow data.
- Creating the data sets
- Model
- Training
- Testing
- Analyzing

This chapter is organized to elaborate on the above-mentioned objectives.

For the data gathering implementation, PHP and JavaScript were used. Python programming language was used for the implementation of data set creating, model training, testing and results analyzing. Keras and OpenCV were the main two libraries used in this study and the training procedures were done in the Google Colaboratory.

4.1. Data Gathering

Using Google Maps JavaScript API, images on Google maps traffic layer, are captured considering in a particular area of Colombo, with 5 minutes of traffic acquisition interval. Latitude and longitude of the considered region are 6.8889298 and 79.8710876 respectively. The zoom level of Google Maps is 15.5. PHP scripts were coded for the image capturing process. A personal computer was adopted to execute the PHP and JavaScript codes.

By modifying the parameters of the Google Maps JavaScript API, the Google map traffic layer was adapted to have a better representation of traffic information and roads network. The implementation of the initializing of the Google Maps traffic layer is shown in figure 4.1.

```
function initMap() {
    var map = new google.maps.Map(document.getElementById('map'), {
        zoom: zm,
        center: {lat: latitude, lng: longitude},
        styles: [
            {
                "featureType": "all",
                "elementType": "all",
                "stylers": [
                    { "visibility": "off" }
                ]
            },
        ],
        disableDefaultUI: false,
        mapTypeControl: false,
        scaleControl: false,
        zoomControl: false,
        streetViewControl: false,
        fullscreenControl: false,
    });
    var trafficLayer = new google.maps.TrafficLayer();
    trafficLayer.setMap(map);
}
```

Figure 4-1: Google Map Traffic Layer Initialization

Google map traffic layer was displayed on the web browser and then screenshots on the Google map traffic layer are captured as the map layer was refreshed every 5 minutes. To capture the screenshot on the traffic map layer html2canvas JavaScript library was used and PHP source code was programmed to save the captured screenshot images. The captured images are saved with the name of capturing date-time.

a) Capturing a screenshot

```
html2canvas(document.getElementById("map"), {
    useCORS: true,
    dpi: 192,
    onrendered: function(canvas) {
        var imgData = canvas.toDataURL("image/png");
        $.ajax({
            url:'save.php',
            type:'post',
            dataType:'text',
            data:{base64data:imgData,pic_name:caption},
            success: function(data){ console.log(data); }
        });
    }
});
```

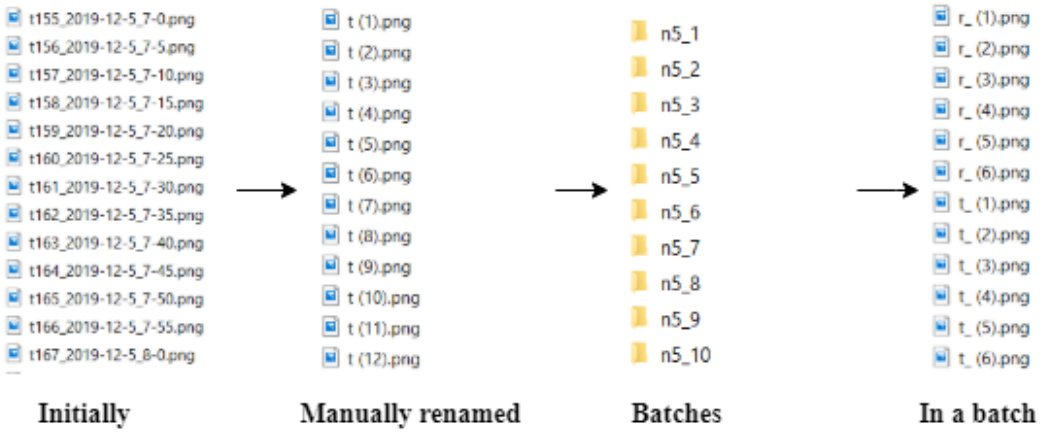
b) Saving a captured screenshot

```
$pic_name = filter_input(INPUT_POST, "pic_name");
$data =filter_input(INPUT_POST, "base64data");

$image = explode('base64',$data);

file_put_contents('./images/'.$pic_name.'.png',base64_decode($image[1]));

echo 'Saved';
```

Figure 4-2: (a) Capturing the screenshot of Google Map Traffic Layer and (b) Saving the captured screenshot.

4.2. Data Sets

There is a sequence of images with 5 minutes of interval and initially, those images were saved with names of date and time. Hence, the data sets should be created separately for every experiment. As an example, for the experiment that predicts traffic flow images into the next 20 minutes, there should be 4 input images and 4 output images in a batch. When those data sets are created, the images should be renamed for ease of further implementation. To create data sets considering the above details, the implementation that is shown in figure 4.3, was applied.




```
14  c = 1
15
16  for d in range(1,15):
17
18      os.makedirs('C:/Users/Dell/Desktop/new1/n5_'+str(d)+'/')
19
20      for i in range(1,7):
21
22          img = Image.open('C:/Users/Dell/Desktop/nov 5/t ('+str(c)+').png')
23          img = img.convert(mode='RGB')
24
25          img.save('C:/Users/Dell/Desktop/new1/n5_'+str(d)+'/t_ ('+str(i)+').png')
26
27          c = c + 1
28
29      for j in range(1,7):
30
31          img = Image.open('C:/Users/Dell/Desktop/nov 5/t ('+str(c)+').png')
32          img = img.convert(mode='RGB')
33
34          img.save('C:/Users/Dell/Desktop/new1/n5_'+str(d)+'/r_ ('+str(j)+').png')
35
36          c = c + 1
37
38      c = c - 6
```

Figure 4-3: Implementation of creating a data set

4.3. Model

The Convolutional LSTM which is also known as ConvLSTM is used to predict the next images which contain the traffic colors and road networks. The ConvLSTM network model which is in the Keras documentation [25] is used by adapting into this scenario which is shown in figure 4.4. The input shape of ConvLSTM is (batches, time_steps, rows, columns, channels) and output shape is 5D tensor with shape (batches, time_steps, rows, columns, filters). As described in chapter 3, there are four ConvLSTM layers and the architecture has a BatchNormalization layer each after every ConvLSTM layers. Here, the BatchNormalization layer is a normalization layer which normalizes the activation of the previous layer at each batch to speed up training. In the end, there is a Conv3D layer. The Conv3D layer is a 3D convolution layer which creates a convolution kernel to convolve with the input.

```
seq = Sequential()

seq.add(ConvLSTM2D(filters=3, kernel_size=(3, 3),
                   input_shape=(None, 300, 300, 3),
                   padding='same', return_sequences=True))
seq.add(BatchNormalization())

seq.add(ConvLSTM2D(filters=3, kernel_size=(3, 3),
                   padding='same', return_sequences=True))
seq.add(BatchNormalization())

seq.add(ConvLSTM2D(filters=3, kernel_size=(3, 3),
                   padding='same', return_sequences=True))
seq.add(BatchNormalization())

seq.add(ConvLSTM2D(filters=3, kernel_size=(3, 3),
                   padding='same', return_sequences=True))
seq.add(BatchNormalization())

seq.add(Conv3D(filters=3, kernel_size=(3,3,3) ,activation='sigmoid',
               padding='same' ))

seq.compile(loss='mean_squared_error', optimizer='adam', metrics=[perceptual_distance])
```

Figure 4-4: The used model (adapting from [25])

The mean squared error (MSE) was applied as the loss function in this model. It measures the average of the squares of the errors as shown in equation (1) where n is the number of predictions, y is the vector of actual values and $\tilde{y}$ is the vector of predicted values.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \qquad (1)$$

In here, the function that is called perceptual_distance was applied [26] to see how similar the actual image and the predicted image are (shown in figure 4.5).

```
def perceptual_distance(y_true, y_pred):
    y_true *= 255.
    y_pred *= 255.
    rmean = (y_true[:, :, :, 0] + y_pred[:, :, :, 0]) / 2
    r = y_true[:, :, :, 0] - y_pred[:, :, :, 0]
    g = y_true[:, :, :, 1] - y_pred[:, :, :, 1]
    b = y_true[:, :, :, 2] - y_pred[:, :, :, 2]

    return K.mean(K.sqrt((((512+rmean)*r*r)/256) + 4*g*g + (((767-rmean)*b*b)/256)))
```

Figure 4-5: Perceptual_distance function ([26])

4.4. Training

As the data sets are included with a huge number of image sequences, the model can't be fed all at once. Such a memory problem happens when working with images and videos. It means that Keras fit() function that is usually used to train a model can't be used here, because the entire training data set is fitted into RAM when Keras fit() function is used. In that case, a solution is to use Keras fit_generator() function instead of feeding all data set at once with Keras fit() function. To use Keras fit_generator() function, there is a need to have a generator function that generates batches of samples indefinitely to apply data augmentation, random shuffling, resizing, etc. the images on the fly.

The generator function that was implemented for this study is shown in figure 4.6. In this function, the images are opened as RGB images and then those images were resized into the images that have 300 rows and 300 columns. The generator function returns python NumPy arrays that have the shape of (batch_size, n_images, rows, columns, channels) for input and target arrays. In this function, since the 'yield' statement has been used, it generates the value of (input_images, output_images) arrays on the fly as shown in figure 4.3 because it doesn't store all those values in memory.

Before starting the training process, values of essential parameters such as batch size, number of input images in a batch, number of epochs and number of steps per epoch should be defined. The batch size is defined according to the maximum size of the input data array that can be handled by the memory at once in the training process. The number of input images is defined according to the predicting time step. The number of steps per

epoch is determined as the number of batches in the training data set divided by the batch size.

```
def my_generator(batch_size,n_images, img_dir):

    dirs = glob(img_dir + "/*")
    counter = 0

    while True:
        input_images = np.zeros((batch_size,n_images,300,300, 3), dtype=np.float)
        output_images = np.zeros((batch_size,n_images,300,300, 3), dtype=np.float)

        random.shuffle(dirs)

        if (counter+batch_size >= len(dirs)):
            counter = 0
        for i in range(batch_size):

            for j in range(n_images):

                input_img_path = dirs[counter + i] + "/t_ ("+str(j+1)+").png"

                in_img = Image.open(input_img_path)
                in_img_cnvt = in_img.convert(mode='RGB')
                in_img_rsz = in_img_cnvt.resize((300,300))
                input_images[i,j] = img_to_array(in_img_rsz)/255

                output_img_path = dirs[counter + i] + "/r_ ("+str(j+1)+").png"

                out_img = Image.open(output_img_path)
                out_img_cnvt = out_img.convert(mode='RGB')
                out_img_rsz = out_img_cnvt.resize((300,300))
                output_images[i,j] = img_to_array(out_img_rsz)/255

        yield (input_images,output_images)

        counter += batch_size
```

Figure 4-6: Generator function

The number of epochs in the training process is considered as 1000 and the trained models are saved in every after 100 epochs and take the converged trained model considering mean-square-error value and perceptual distance values.

4.4.1. Twenty Minutes into the Future

The first process is to train the model to predict the traffic flow images in the next 20 minutes. As the predicting time is 20 minutes, the number of input images in a batch is 4. As described in chapter 3, this process is conducted with two experiments.

- In the first experiment, there are 12 batches in the training data set. Because in here the images that are applied to the training were captured on 5th November

from 7 am to 11 am (one day). Hence, the batch size can be defined as 12 and then the number of steps per epoch is 1.

- In the second experiment, there are 210 batches in the training data set. Because in here the images that are applied to the training were captured from 5th November to 11th November (5 weekdays) and from 7 am to 9 pm. Hence, the batch size can be defined as 20 and then the number of steps per epoch is 10 (the number of batches in the training data set divided by the batch size.

4.4.2. Thirty Minutes into the Future

The second process is to train the model to predict the traffic flow images in the next 30 minutes. As the predicting time is 30 minutes, the number of input images in a batch is 6. There are 130 batches in the training data set. Because the images that are applied to the training were captured from 5th November to 11th November (5 weekdays) and from 7 am to 8 pm (13 hours per day). Hence batch size can be defined as 15 and then the number of steps per epoch is 8.

4.4.3. One Hour into the Future

The second process is to train the model to predict the traffic flow images in the next 60 minutes (1 hour). As the predicting time is 60 minutes, the number of input images in a batch is 12. There are 60 batches in the training data set. Because the images that are applied to the training were captured from 5th November to 11th November (5 weekdays) and from 7 am to 9 pm. Hence batch size can be defined as 6 and then the number of steps per epoch is 10.

4.5. Testing

Testing data sets of each experiment process are contained with subdirectories that are included with a small number of testing batches. Then the testing processes are done by using these subdirectories separately. In that case, the prediction was done by using the trained model and the Keras predict() function. Thus, predicted images are saved according to those subdirectories for the easy of the evaluation process.

4.6. Results Evaluation

Before analyzing, the results images are enhanced using python PIL library as shown in figure 4.7. Then the analyzing process was conducted using these predicted enhanced images.

```
enhancer = ImageEnhance.Color(predict_img)
enhanced = enhancer.enhance(1.2)

enhancer = ImageEnhance.Contrast(enhanced)
enhanced = enhancer.enhance(1.2)

enhancer = ImageEnhance.Sharpness(enhanced)
enhanced = enhancer.enhance(1.2)
```

Figure 4-7: Enhancing the predicted images

As described in chapter 3, the confusion matrix is used for performance evaluation. In the confusion matrix true positive, true negative, false positive and false negative values are counted according to the algorithm shown in figure 4.8. In this algorithm, if there are green colors (no traffic congestion) on a position of the actual image, then the same position is checked on the predicted image whether there are green colors or not. After calculating these values of the confusion matrix, the measurements which are sensitivity, specificity, precision, F1-measure, and accuracy are evaluated according to the equations described in chapter 3.

```
Actual_Image <- convert RGB to HSV

Predicted_Image <- convert RGB to HSV

for x in raw_size:

    for y in column_size:

        reject background

        if Green color in Actual_Image:            ## There is no traffic congestion in Actual_image

            if Green color in Predicted_Image:

                True_Nagative++                    ## Also there is no traffic congestion in Predicted_Image

            else:

                False_Positive++                   ## But there are traffic congestion in Predicted_Image

        elif Red|Orange color in Actual_Image:     ## There are traffic congestion in Actual_image

            if Red|Orange color in Predicted_ImageImage:

                True_Positive++                    ## Also there are traffic congestion in Predicted_Image

            else:

                False_Negative++                   ## But there is no traffic congestion in Actual_image
```

Figure 4-8: Algorithm for TP, TN, FP and, FN values calculation

To represent how the correct predictions and errors were made on images, the masks were obtained for green color portions and red/orange color portions of images. As shown in figure 4.9, OpenCV inRange() function was applied to obtain the masks considering lower bounds and upper bounds for green and red/orange colors. Then using OpenCV bitwise logic operations (AND and XOR), the desired output can be obtained.

```
5   #--------- actual image and predicted image
6   true_image = cv2.imread(true_path)
7   predicted_image = cv2.imread(predict_path)
8
9   #------------- Orange/Red range
10  orange_red_lower = np.array([0, 100, 100],np.uint8)
11  orange_red_upper = np.array([30, 255, 255],np.uint8)
12
13  orange_red_true = cv2.inRange(true_hsv, orange_red_lower, orange_red_upper)
14  orange_red_predicted = cv2.inRange(predicted_hsv, orange_red_lower, orange_red_upper)
15
16  #----------- Kernal
17  kernal = np.ones((5 ,5), "uint8")
18
19  orange_red_true = cv2.dilate(orange_true, kernal)
20  orange_red_predicted = cv2.dilate(orange_predicted, kernal)
21
22  #------------orange correct
23  correct_traffic = cv2.bitwise_and(orange_red_true,orange_red_predicted)
24
25  #------------orange faults/ error
26  error_traffic = cv2.bitwise_xor(orange_red_true,orange_red_predicted)
```

Figure 4-9: Implementation of representing Red/Orange colors with the mask

4.7. Summary

In this chapter, the implementation of the study has been described. The implementations of data gathering, data set creating, model utilizing, experimenting and analysis process were elaborated further in detail.

Chapter 5

Results and Evaluation

5.1 Data Gathering

Using Google Maps JavaScript API, images on Google maps traffic layer, are captured considering in a particular area of Colombo, with 5 minutes of traffic acquisition interval. These images are captured from November 5th, 2019 to December 30th, 2019 (8 weeks) successfully.

Table 5-1: Gathered Information

Centered Location	Latitude = 6.8889298 and Longitude = 79.8710876
Zoom level of Google maps	15.5
Data acquisition interval	5 minutes
Duration	November 5th, 2019 - December 30th, 2019 (8 weeks)

Google map traffic layer was adapted to have a better representation of traffic information and roads network by changing parameters of Google Maps JavaScript API as presented in figure 5.1.

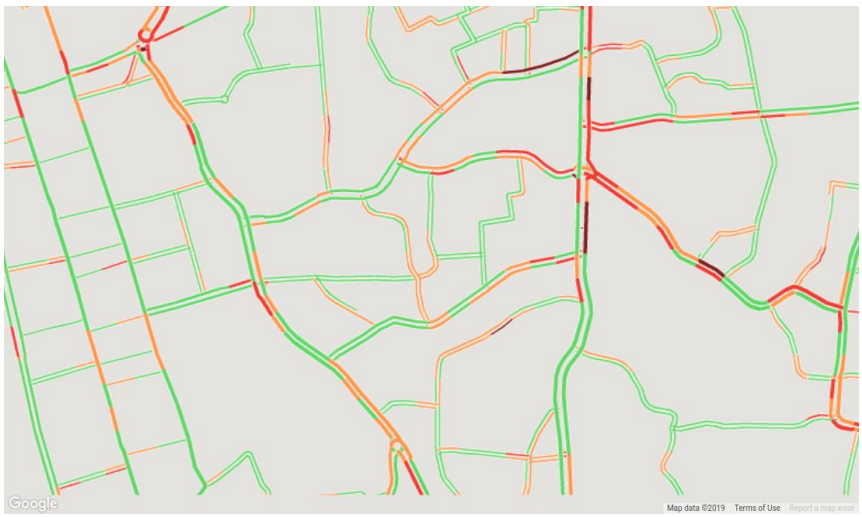


Figure 5-1: Captured image which represents only traffic congestion colors and roads network

5.2 Experiments and Results

As this study is based on short-term traffic prediction, experiments were conducted with two main approaches which are based on predicting time 20 minutes and 1 hour in the future. For the evaluation of the results, the confusion matrix is adopted as described in chapter 3.

5.2.1. Twenty Minutes into the Future

One batch is included with four input images and four target images for a specific period as shown in the table 5.2 and figure 5.2.

Table 5-2: Example for a batch

Input Images	Target Images
7.00 am	7.20 am
7.05 am	7.25 am
7.10 am	7.30 am
7.15 am	7.30 am

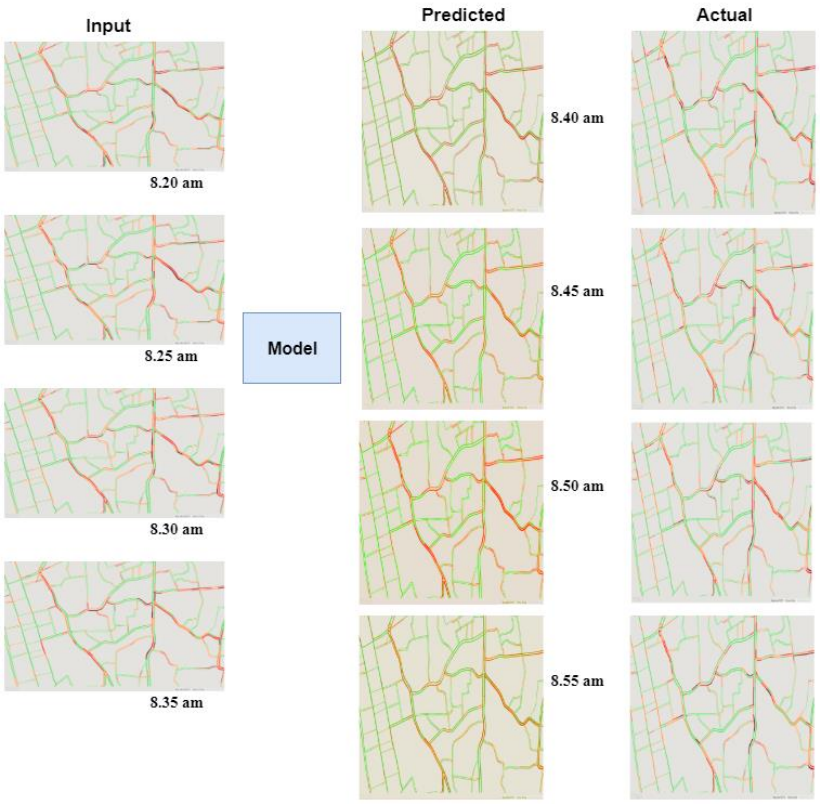


Figure 5-2: One batch

5.2.1.1. First Experiment

In this experiment, the training was conducted using 12 batches which contain images that are captured on 5th November from 7 am to 11 am (one day). The testing was conducted using 10 batches which contain images that are captured on 6th November from 7 am to 11 am (next day). Hence the main purpose of this experiment is to see how traffic congestion can be predicted by using only previous day data during the same time period. Table 5.3 shows the performance values which were gained from this experiment.

Table 5-3: Results values for all batches

Batch	Input Time	Predicted Time	Minutes	Sensitivity	Specificity	Accuracy
01	8.20 am	8.40 am	05	0.7064	0.7363	0.7236
	8.25 am	8.45 am	10	0.6735	0.7851	0.7370
	8.30 am	8.50 am	15	0.7461	0.6977	0.7195
	8.35 am	8.55 am	20	0.7212	0.6755	0.6941
Average				0.7118	0.7237	0.7185
02	7.40 am	8.00 am	05	0.7129	0.7322	0.7231
	7.45 am	8.05 am	10	0.7413	0.7691	0.7567
	7.50 am	8.10 am	15	0.7799	0.6768	0.7267
	7.55 am	8.15 am	20	0.7402	0.6774	0.7085
Average				0.7436	0.7139	0.7287
03	10.40 am	11.00 am	05	0.6344	0.7327	0.7020
	10.45 am	11.05 am	10	0.6161	0.8137	0.7481
	10.50 am	11.10 am	15	0.6691	0.7675	0.7339
	10.55 am	11.15 am	20	0.5870	0.7586	0.7038
Average				0.6267	0.7681	0.7220
04	9.00 am	9.20 am	05	0.6972	0.6745	0.6822
	9.05 am	9.25 am	10	0.6824	0.7438	0.7239
	9.10 am	9.30 am	15	0.71781	0.6168	0.6460
	9.15 am	9.35 am	20	0.6663	0.6870	0.6807
Average				0.6909	0.6805	0.6832
	8.40 am	9.00 am	05	0.7349	0.7095	0.7200

05	8.45 am	9.05 am	10	0.6840	0.7514	0.7246
	8.50 am	9.10 am	15	0.7686	0.6302	0.6837
	8.55 am	9.15 am	20	0.7210	0.6134	0.6528
Average				0.7271	0.6761	0.6953
06	9.20 am	9.40 am	05	0.6931	0.7241	0.7147
	9.25 am	9.45 am	10	0.5803	0.7679	0.7101
	9.30 am	9.50 am	15	0.6921	0.7024	0.6995
	9.35 am	9.55 am	20	0.61132	0.7283	0.6947
Average				0.6442	0.7307	0.7048
07	7.20 am	7.40 am	05	0.7068	0.6039	0.6519
	7.25 am	7.45 am	10	0.7089	0.6906	0.6995
	7.30 am	7.50 am	15	0.7701	0.6249	0.6967
	7.35 am	7.55 am	20	0.6864	0.6261	0.6556
Average				0.7181	0.6364	0.6759
08	7.00 am	7.20 am	05	0.5985	0.7549	0.6775
	7.05 am	7.25 am	10	0.5947	0.7900	0.6979
	7.10 am	7.30 am	15	0.6968	0.7460	0.7214
	7.15 am	7.35 am	20	0.6619	0.6960	0.6794
Average				0.6380	0.7467	0.6940
09	10.00 am	10.20 am	05	0.6261	0.7632	0.7271
	10.05 am	10.25 am	10	0.5255	0.7956	0.7248
	10.10 am	10.30 am	15	0.5824	0.7284	0.6925
	10.15 am	10.35 am	20	0.5720	0.7701	0.7152
Average				0.5765	0.7643	0.7149
10	8.00 am	8.20 am	05	0.7708	0.6607	0.7067
	8.05 am	8.25 am	10	0.7436	0.7222	0.7318
	8.10 am	8.30 am	15	0.8144	0.6742	0.7394
	8.15 am	8.35 am	20	0.7507	0.6571	0.6997
Average				0.7699	0.6786	0.7194

Table 5-4: Average Sensitivity, Specificity and Accuracy for all 40 images

All 40 testing Images	
Sensitivity	0.68472
Specificity	0.71195
Precision	0.60081
F1 Measure	0.63698
Accuracy	0.70572

Table 5-5: Average Sensitivity, Specificity and Accuracy for Predicted Time Periods

	05 minutes	10 minutes	15 minutes	20 minutes
Sensitivity	0.68815	0.65510	0.72378	0.67184
Specificity	0.70923	0.76300	0.68655	0.68900
Precision	0.59945	0.63502	0.59343	0.57533
F1 Measure	0.63792	0.64342	0.64850	0.61807
Accuracy	0.70292	0.72549	0.70598	0.68850

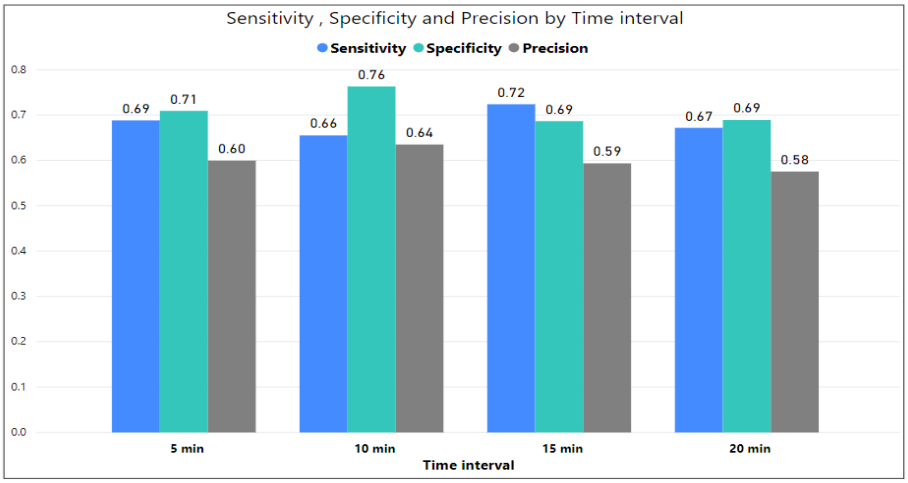


Figure 5-3: Sensitivity, Specificity & Precision by time interval -20 min

When the prediction time step is in 20 minutes, precision value and F1- measure are lower compared to the other time steps values.

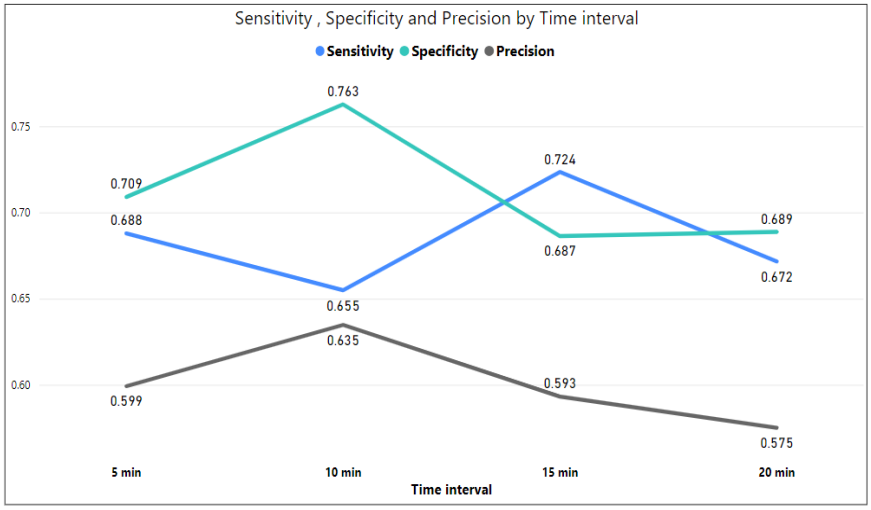


Figure 5-4: Line chart for sensitivity, specificity & precision by time interval -20 min

As this line chart shows in figure 5.4, when the prediction time step is in 10 minutes, the gap between sensitivity and specificity is bigger than compared to other prediction time steps.

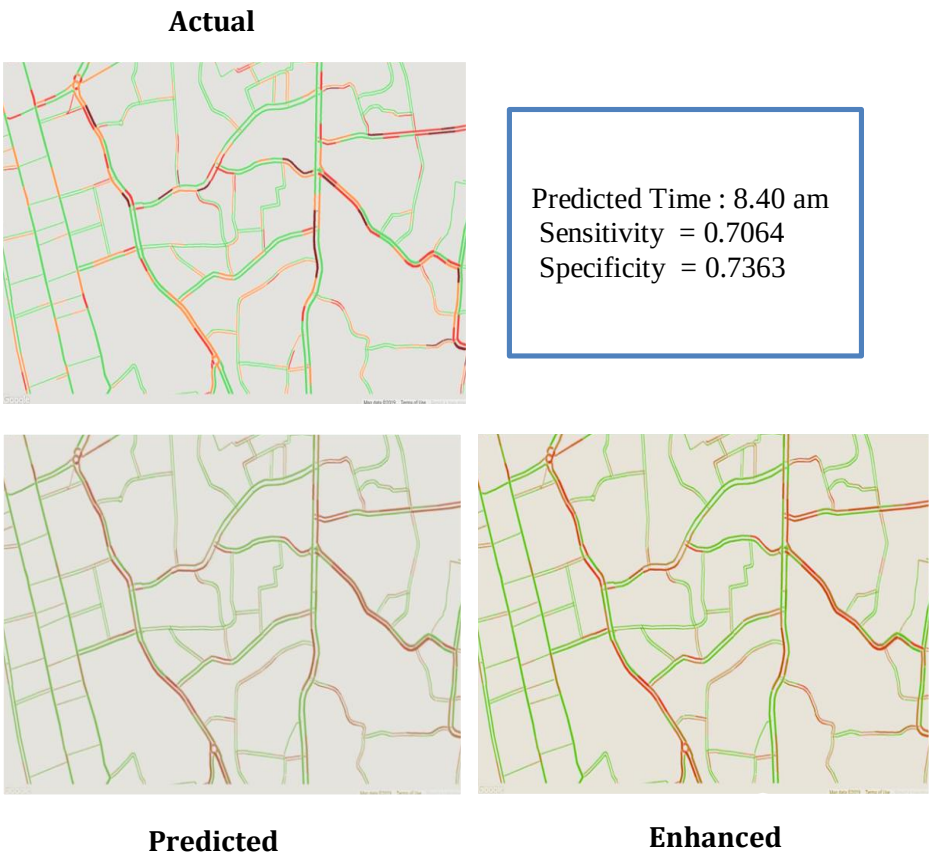


Figure 5-5: Actual and Predicted image comparison of an image

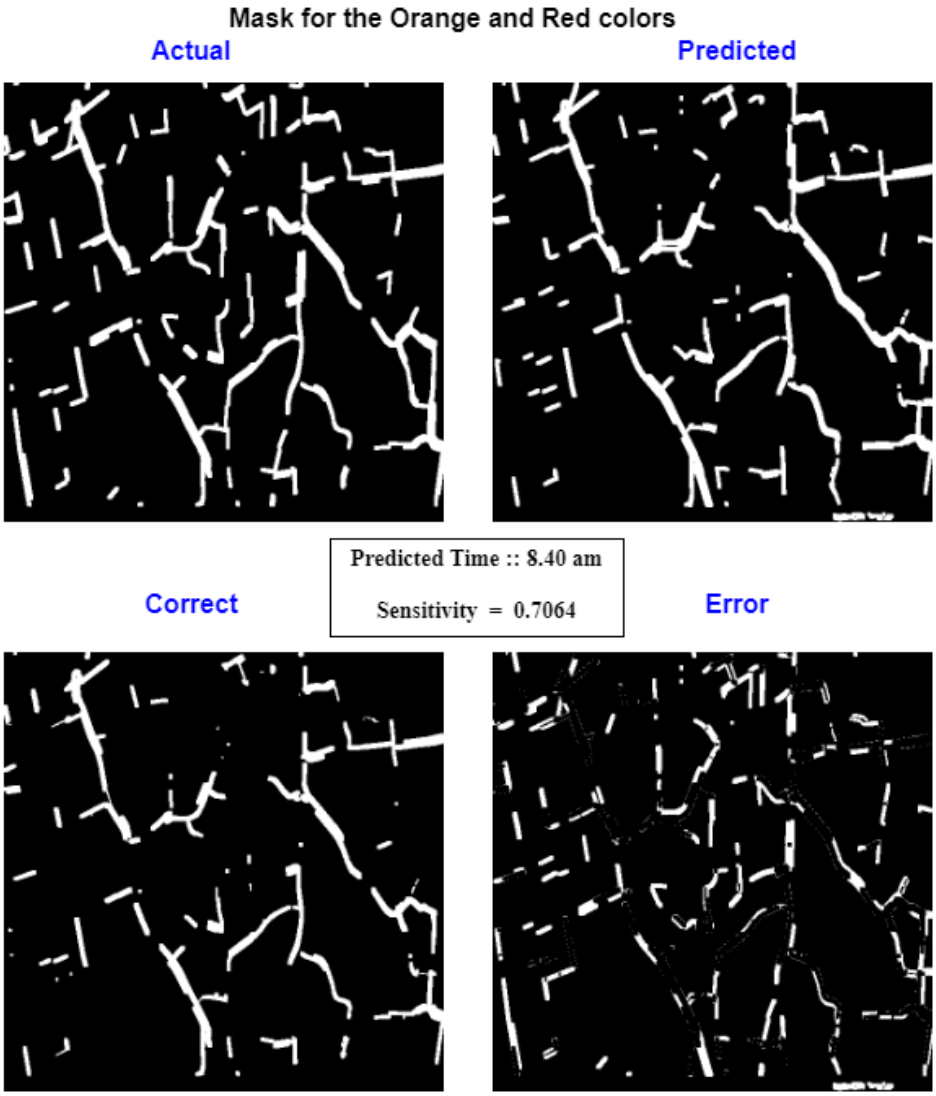


Figure 5-6: Representation of Orange & Red column with the mask

Figure 5.6 shows the correctness and the error of predicting red/orange colors by representing clearly. And also figure 5.7 shows the correctness and the error of predicting green color.

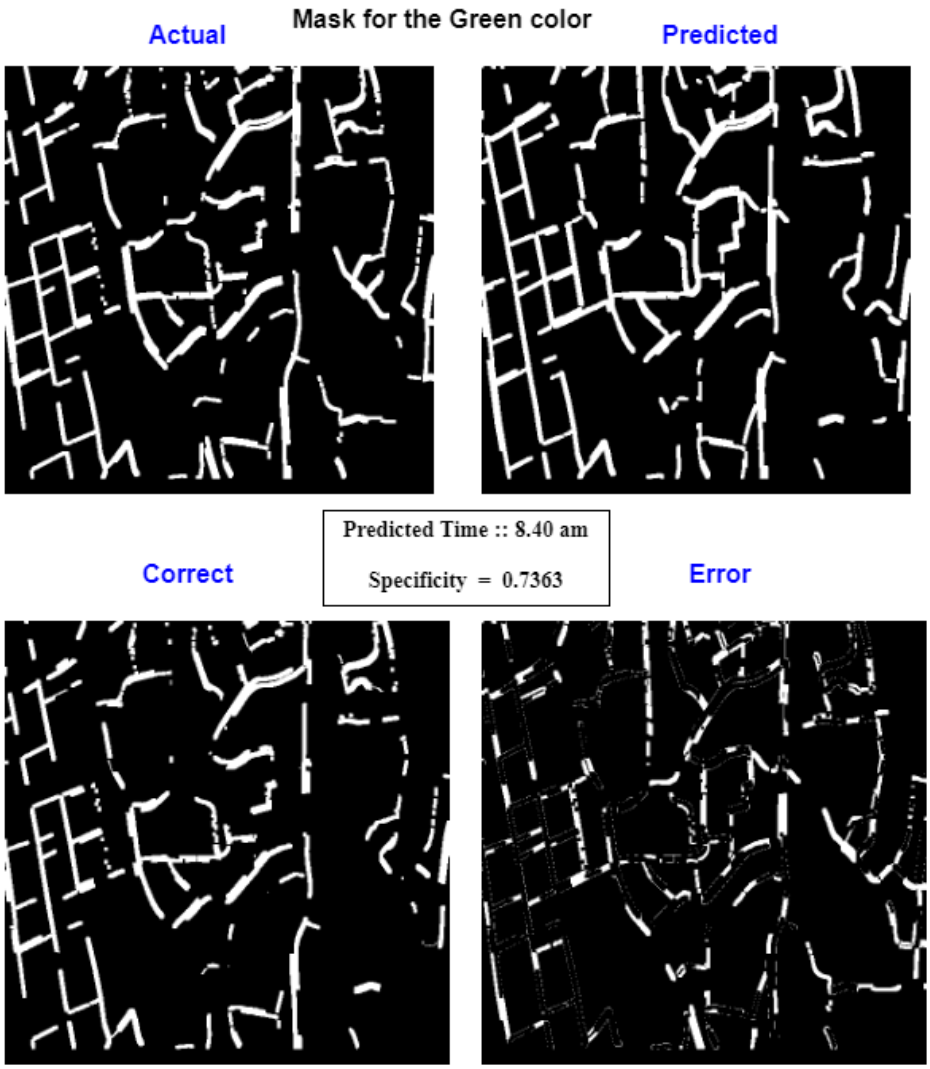


Figure 5-7: Representation of Green colors with the mask

5.2.1.2. Second Experiment

In this experiment, the training was conducted using 210 batches which contain images that are captured from 5th November to 11th November (5 weekdays). When considered all these 210 batches, there are 840 input images and 840 target images for the training procedure. The testing was conducted using 84 batches which contain images that are captured on 13th November and 14th November (next week 2 days data). When considered all these 84 batches, there are 336 input images and 336 target images for the testing procedure. The images that are captured from 7 am to 9 pm on weekdays, was exploited for this observation. Hence the main purpose of this experiment is to see how traffic congestion can be predicted by using only previous week data during the same period.

Table 5-6: Average Sensitivity, Specificity and Accuracy for all 336 testing images

All 336 testing Images	
Sensitivity	0.671561
Specificity	0.752338
Precision	0.551288
F1 Measure	0.647784
Accuracy	0.733022

Table 5-7: Average Sensitivity, Specificity and Accuracy for Predicted Time Periods

	05 minutes	10 minutes	15 minutes	20 minutes
Sensitivity	0.668656	0.686630	0.691103	0.639856
Specificity	0.723930	0.763929	0.748441	0.773053
Precision	0.527896	0.567366	0.551159	0.558733
F1 Measure	0.586104	0.618363	0.609783	0.591805
Accuracy	0.712751	0.745788	0.736974	0.736575

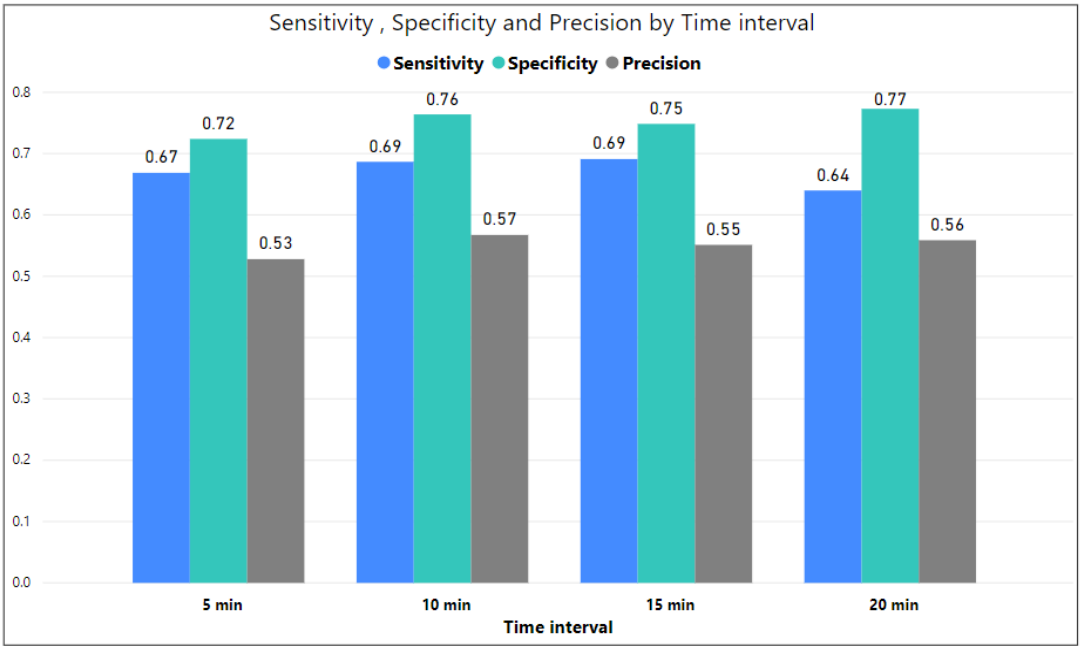


Figure 5-8: Sensitivity, Specificity & Precision by time interval experiment two

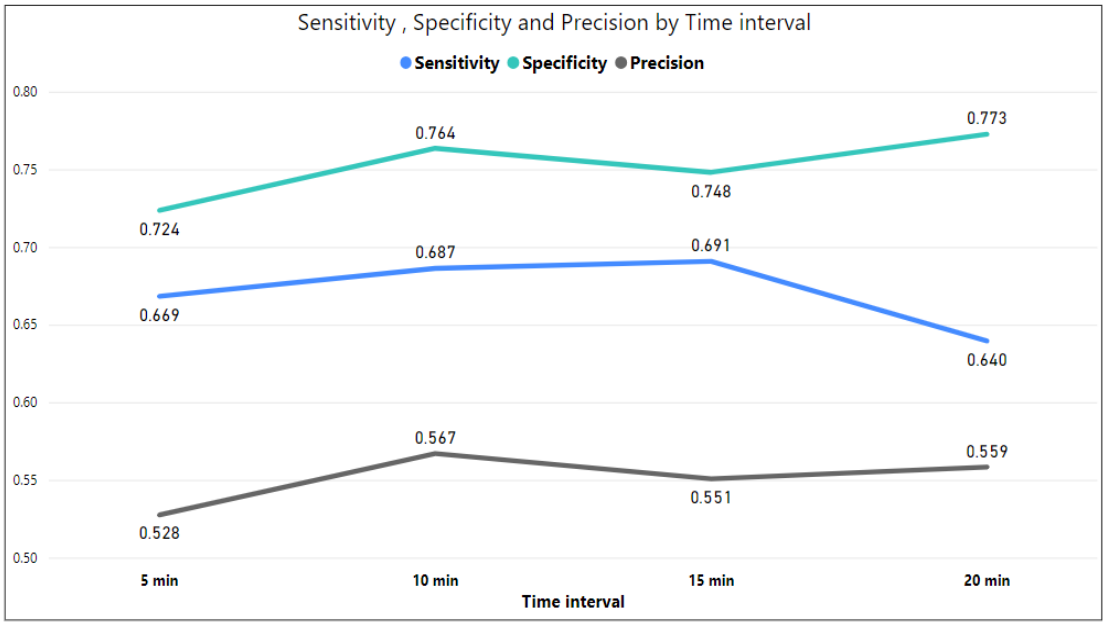


Figure 5-9: Line chart - Accuracy comparison by time interval for experiment two

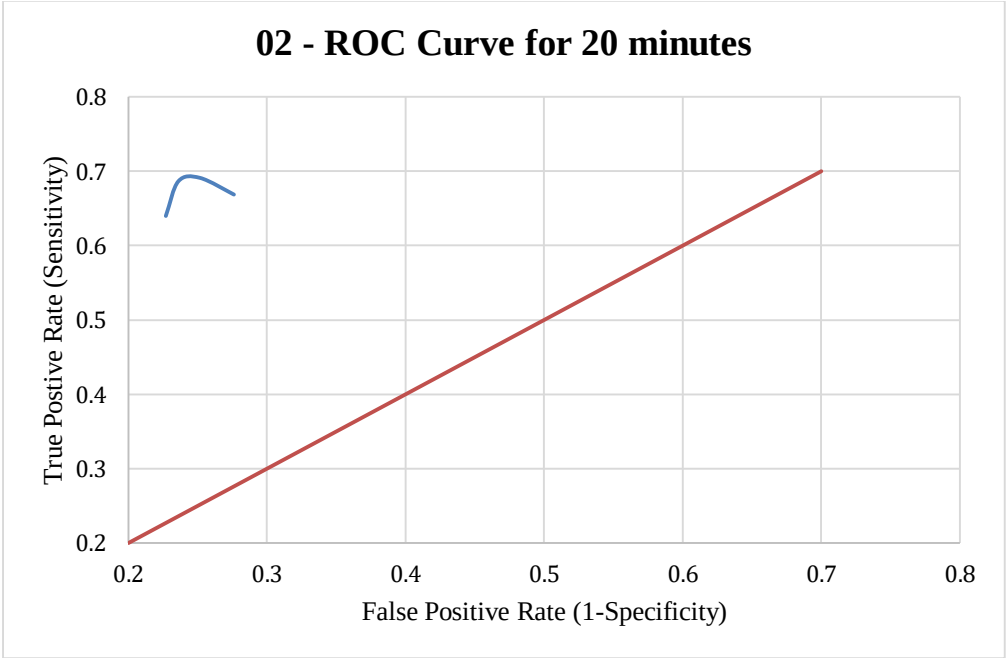


Figure 5-10: ROC Curve for 20 minutes experment two

In this experiment, the precision values of all prediction time steps are approximately close. However, as the line chart is shown in figure 5.9 when the prediction time step is in 20 minutes, the gap between sensitivity and specificity is bigger than compared to other prediction time steps.

5.2.2. Thirty Minutes into the Future

In this one batch is included with 6 input images and 6 target images for a specific period as shown in table 5.2.

Table 5-8: Example for a batch for

Input Images	Target Images
7.00 am	7.30 am
7.05 am	7.35 am
7.10 am	7.40 am
7.15 am	7.45 am
7.20 am	7.50 am
7.25 am	7.55 am

In this experiment, the training was conducted using 130 batches which contain images that are captured from 5th November to 11th November. When considered all these 130 batches, there are 780 input images and 780 target images for the training procedure. The testing was conducted using 52 batches which contain images that are captured on 13th November and 14th November (next week data). When considered all these 52 batches, there are 312 input images and 312 target images for the testing procedure. The images that are captured from 7 am to 8 pm (13 hours) on weekdays, was exploited for this observation. Hence the main purpose of this experiment is to see how traffic congestion can be predicted thirty minutes into the future by using only previous week data for training during the same period.

Table 5-9: Average Sensitivity, Specificity and Accuracy for all 312 testing images – 30 minutes

All 312 testing Images	
Sensitivity	0.626034
Specificity	0.742498
Precision	0.540546
F1 Measure	0.573869
Accuracy	0.708158

Table 5-10: Average Sensitivity, Specificity, and Accuracy for Time interval

Predicted Time (minutes)	Sensitivity	Specificity	Precision	F1 Measure	Accuracy
05	0.673192	0.674092	0.498872	0.568499	0.677909
10	0.643027	0.737542	0.544226	0.584276	0.709600
15	0.625704	0.738025	0.537596	0.573304	0.704725
20	0.609835	0.768893	0.557793	0.577613	0.721142
25	0.612487	0.775447	0.567949	0.583941	0.725953
30	0.591958	0.760989	0.536843	0.555580	0.709620

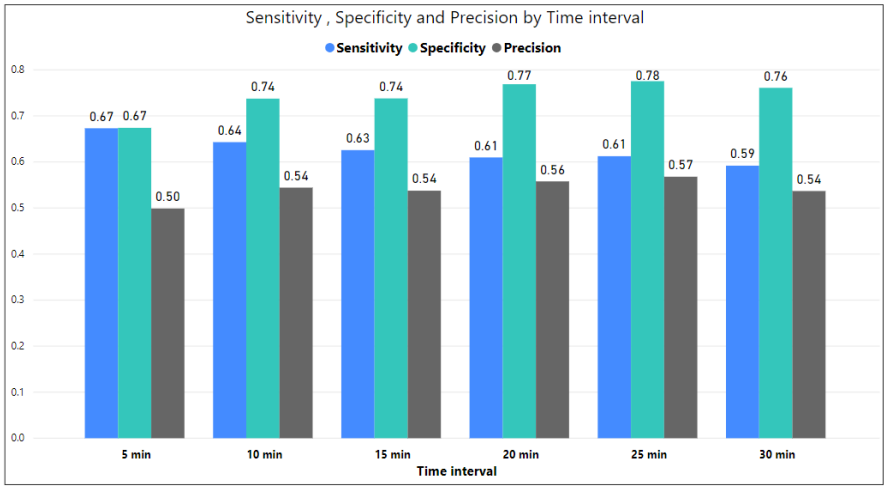


Figure 5-11: Sensitivity, Specificity & Precision by time interval-30 min

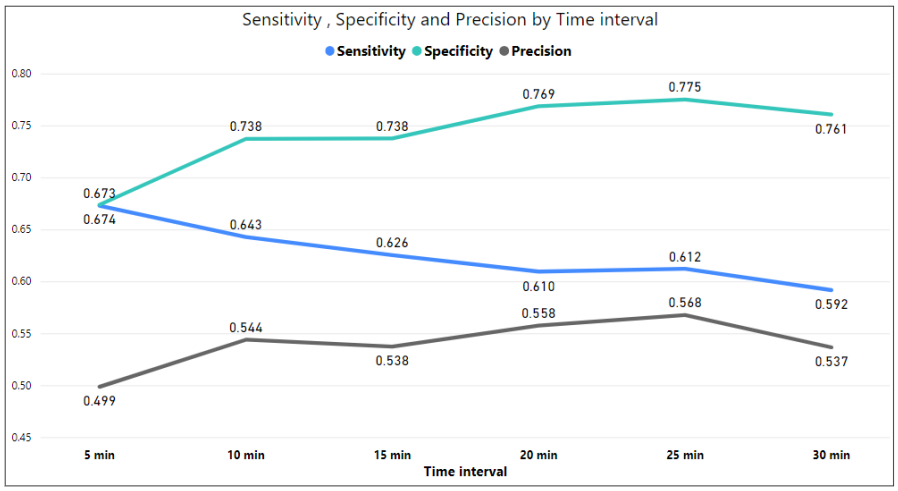


Figure 5-12: Line chart for sensitivity, specificity & precision -30 min

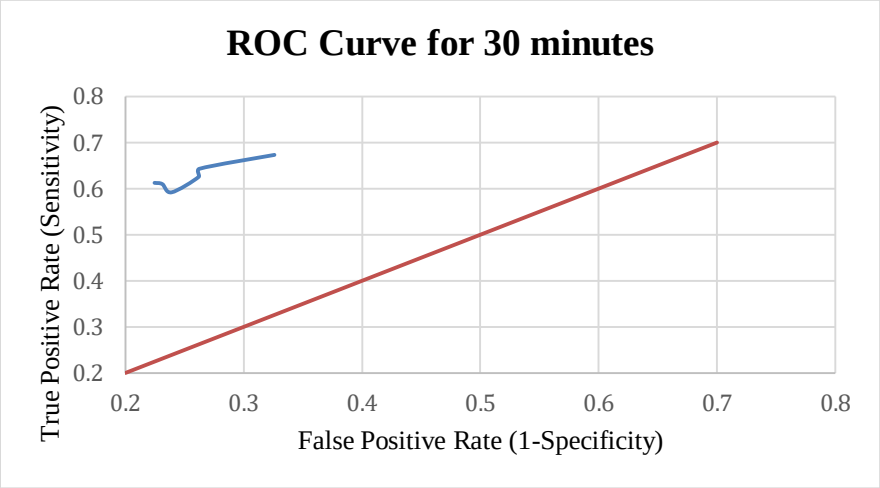


Figure 5-13: ROC Curve for 30 min experiment

In this experiment, these results show that, as displayed in line-chart figure 5.12, the gap between sensitivity and specificity was being increased while the prediction time step was moving. And also the precision value and F1-measure value are a little bit lower compared to the values of the previous experiment.

5.2.3. One Hour into the Future

One batch is included with 12 input images and 12 target images for a specific period as shown in table 5.2.

Table 5-11: Example for a batch

Input Images	Target Images
7.00 am	8.00 am
7.05 am	8.05 am
7.10 am	8.10 am
7.20 am	8.20 am
7.25 am	8.25 am
7.30 am	8.30 am
7.30 am	8.30 am
7.35 am	8.35 am
7.40 am	8.40 am
7.45 am	8.45 am
7.50 am	8.50 am
7.55 am	8.55 am

In this experiment, the training was conducted using 65 batches which contain images that are captured from 5th November to 11th November. When considered all these 65 batches, there are 780 input images and 780 target images for the training procedure. The testing was conducted using 26 batches which contain images that are captured on 13th November and 14th November (next week data). When considered all these 26 batches, there are 312 input images and 312 target images for the testing procedure. The images that are captured from 7 am to 9 pm (14 hours) on weekdays, was exploited for this observation. Hence the main purpose of this experiment is to see how traffic congestion can be predicted one hour into the future by using only previous week data during the same period.

Table 5-12: Average Sensitivity, Specificity and Accuracy for all 336 testing images

All 312 testing Images	
Sensitivity	0.570436
Specificity	0.717574
Precision	0.484351
F1 Measure	0.510373
Accuracy	0.668668

Table 5-13: Average Sensitivity, Specificity and Accuracy for Predicted Time

Predicted Time (minutes)	Sensitivity	Specificity	Precision	F1 Measure	Accuracy
05	0.567337	0.691292	0.466889	0.502134	0.649948
10	0.607718	0.691167	0.478940	0.525114	0.662657
15	0.611606	0.682614	0.477803	0.525452	0.658146
20	0.608319	0.688877	0.476543	0.523192	0.661650
25	0.595656	0.714419	0.500429	0.532115	0.674045
30	0.571511	0.719437	0.483690	0.511184	0.670619
35	0.562260	0.717783	0.477069	0.503642	0.667237
40	0.564741	0.728344	0.495574	0.515692	0.673745
45	0.560840	0.731811	0.490391	0.508312	0.674814

50	0.554459	0.745871	0.501194	0.511791	0.682453
55	0.572506	0.739130	0.496468	0.516003	0.683462
60	0.4682801	0.760144	0.467222	0.449849	0.665235

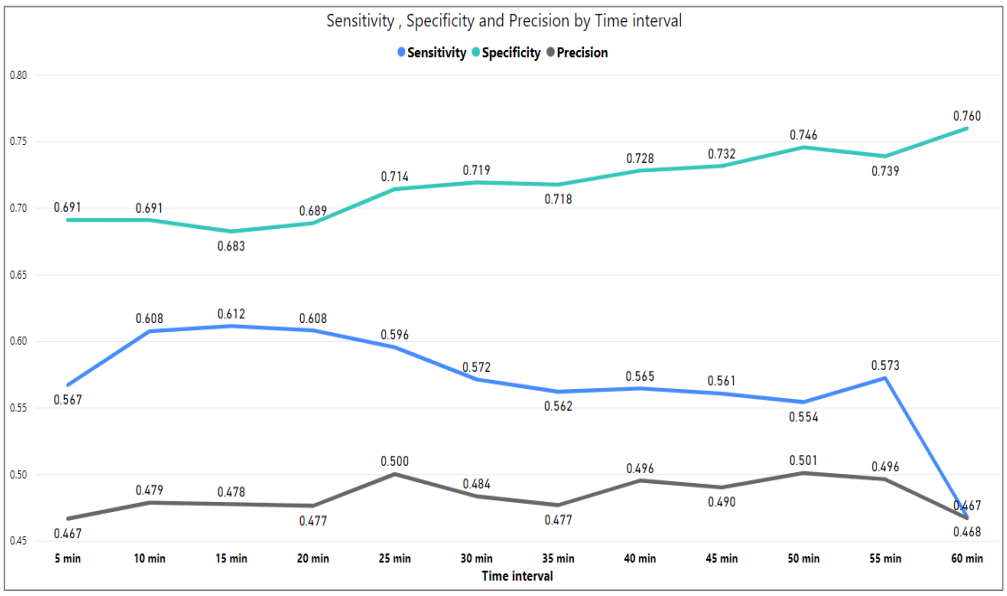


Figure 5-14: Line chart for sensitivity, specificity & precision by time interval-60 min

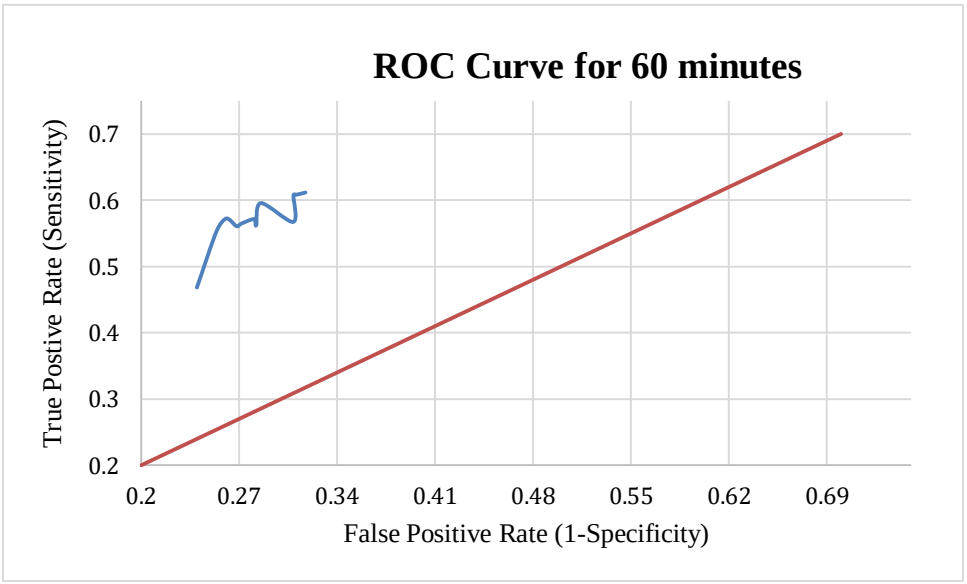


Figure 5-15: ROC Curve for the 60 min experiment

In this experiments, the results of the precision measurement is much lower compared to the previous experiments values. As shown in figure 5.14, the gap between sensitivity and specificity was being increased and when the prediction time step is 60 minutes there are much bigger compared to the other prediction time steps and other experiments.

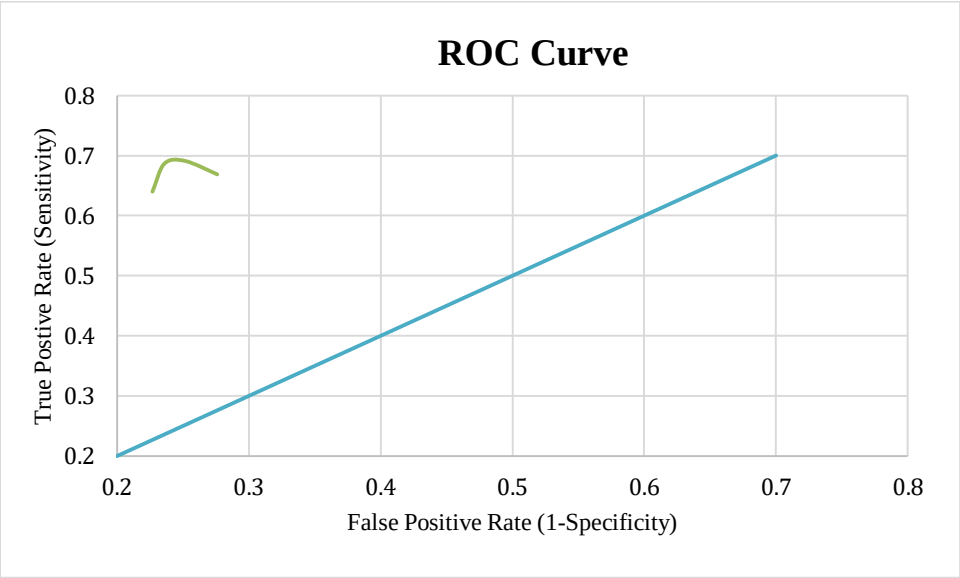


Figure 5-16: ROC Curve for all the three experiments

Table 5-14: Comparison of F1-measure and precision of three experiments

Experiment	Sensitivity	Specificity	Precision	F1-Measure
20 minutes	0.671561	0.752338	0.551288	0.647784
30 minutes	0.626034	0.742498	0.540546	0.573869
60 minutes	0.570436	0.717574	0.484351	0.484351

As shown in figure 5.16 and table 5.14, when the experimenting time step is increased, values of F1-measure and precision measure had been decreased. Also figure 5.16 has shown that the experiment that predicts traffic flow into 20 minutes' future, is more acceptable compared to the other two experiments.

5.3 Summary

This chapter elaborated the results and evaluation of the project methodology. It was described with three designed experimentations. And also a comparison of those three experiments was discussed.

Chapter 6
Conclusions

6.1. Introduction

This chapter shows conclusions about the research problem and the research questions. Also the limitations of the research and the further implications are described in this chapter.

6.2. Conclusion about Research Questions

AS depicted in chapter 1, there have been three research questions and the first question impacts the second question and the second question impacts the third question.

6.2.1. What is the most suitable way to collect traffic flow information?

The most suitable way to collect traffic flow information is Google.

In modern-day, it is not useful having traffic flow data only in a few positions. It is because traffic congestion is an increasingly serious problem that varies from time to time and region to region. Hence having traffic flow information on a road network is more profitable rather than having only data in a few positions. In such a situation, the problem is what is the traffic data collection way. As indicated in this study, the solution is to use Google instead of using high-cost devices such as sensors and cameras.

6.2.2. How to collect traffic flow information?

The traffic flow information can be gathered by capturing the images on the Google map traffic layer which is publicly available and it can be achieved by using Google Map JavaScript API.

Even though the answer is to use the Google for above first question, their traffic flow data is not directly available. However, those real-time traffic flow data have been elaborated geographically on the Google map traffic layer that is publicly available. Thus, the real-time traffic flow information can be gathered in the way this study did.

6.2.3. What is the most suitable model building approach for short-term traffic prediction?

The ConvLSTM model approach is accommodated for short-term traffic prediction.

After having answered the above two questions, data sets are included with sequences of images on the Google map traffic layer. Hence the model should have the capability of dealing with the sequential data and as well as with spatiotemporal (images) data. Thus one solution is to use a model based on ConvLSTM neural network as exploited in this study.

6.3. Conclusion about Research Problem

What is the possibility of devising a methodology which integrated public data for short-term traffic prediction?

This study has yielded a profitable and affordable methodology for short-term traffic prediction appropriately for Sri Lankan Context.

As a conclusion, this study has introduced a possible methodology that integrated public traffic flow data for short-term traffic prediction. From being answered the above research questions, conducting various experiments and presenting performance values for those experiments it was established further.

6.4. Limitations

The traffic flow information that was evaluated and presented by the Google is used in this study. Thus the perfection of their real-time data is beyond this study's control. It implies that this study fully depends on Google's real-time traffic data.

When the images are being captured on Google maps traffic layer using the Google Maps JavaScript API, there is a limited number of API calls that can be made without being paid. Therefore, maintaining a low data acquisition interval and good map zoom level, all cities in Sri Lanka can't be covered without being paid by capturing images.

Even though the ConvLSTM has the ability to deal with sequential and spatiotemporal data, there have been some limitations to the structure of the ConvLSTM layer. In here, the combination of spatiotemporal features and other features is limited by the convolutional structure of a ConvLSTM unit. Thus, the study of the traffic flow with other socio-conceptual data such as holidays, special events and weather data is limited in this study.

6.5. Implication for Further Research

The proposed methodology studied the traffic prediction for short-term predictions. Also, experiments can be expanded further for long-term traffic prediction as well. And also, this study can be experimented by creating different patterns of sequences according to predicting time steps.

From creating a new model based on ConvLSTM which was addressed for the above mentioned structural limitation, weather information and other features can be applied to study the traffic flow prediction and it would be an interesting attempt.

Chapter 7 References

[1] L. Vanajakshi and L. R. Rilett, "A comparison of the performance of artificial neural networks and support vector machines for the prediction of traffic speed," in *IEEE Intelligent Vehicles Symposium*, Parma, Italy, 2004.

[2] N. Petrovska and A. Stevanovic, "Traffic Congestion Analysis Visualisation Tool," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Las Palmas, 2015.

[3] A. Padiath, L. Vanajakshi, S. C. Subramanian and H. Manda, "Prediction of traffic density for congestion analysis under Indian traffic conditions," in *2009 12th International IEEE Conference on Intelligent Transportation Systems*, St. Louis, MO, 2009.

[4] Yoon, Jungkeun & Noble, Brian & Liu, Mingyan, " Surface Street Traffic Estimation," in *MobiSys'07: Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, 207.

[5] Rao, Mohan & Rao, K Ramachandra, "Measuring Urban Traffic Congestion – A Review," *International Journal for Traffic and Transport Engineering*, vol. 2, pp. 286-305, 2012.

[6] D. Schrank, B. Eisele, T. Lomax, and J. Bak, "2015 urban mobility scorecard," The Texas A&M Transportation Institute and INRIX, 2015. [Online]. Available: https://trid.trb.org/view/1367337.

[7] J. Park et al., "Real time vehicle speed prediction using a Neural Network Traffic Model," in *The 2011 International Joint Conference on Neural Networks*, San Jose, CA, 2011.

[8] Ma, X.; Dai, Z.; He, Z.; Ma, J.; Wang, Y.; Wang, Y., "Learning Traffic as Images: A Deep Convolutional Neural Network for Large-Scale Transportation Network Speed Prediction," *Sensors*, vol. 17, p. 818, 2017.

[9] Gavirangaswamy, Vinay & Gupta, Gagan & Gupta, Ajay & Agrawal, Rajeev, "Assessment of ARIMA-based prediction techniques for road-traffic volume," in

Proceedings of the 5th International Conference on Management of Emergent Digital EcoSystems, MEDES 2013. .

[10] R. Fu, Z. Zhang and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, Wuhan, 2016.

[11] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen and J. Liu, "STM network: a deep learning approach for short-term traffic forecast," *in IET Intelligent Transport Systems*, vol. 11, pp. 68-75, 2017.

[12] Kumarage, Sakitha., "Use of Crowdsourced Travel Time Data in Traffic Engineering Applications," 2018.

[13] Aftabuzzaman, M., "Measuring traffic congestion- A critical review," in *30th Australasian Transport Research Forum*, 2007.

[14] "The Bright Side of Sitting in Traffic: Crowdsourcing Road Congestion Data," Googleblog, 2009. [Online]. Available: https://googleblog.blogspot.com/2009/08/bright-side-of-sitting-in-traffic.html.

[15] Google Maps Help, "View places, traffic, terrain, biking, and transit with Google Maps – What the colors and symbols mean on the legend," [Online]. Available: https://support.google.com/maps/answer/3092439?co=GENIE.Platform%3DDesktop&hl=en.

[16] A. Koesdwiady, R. Soua and F. Karray, "Improving Traffic Flow Prediction With Weather Information in Connected Cars: A Deep Learning Approach," *in IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, pp. 9508-9517, Dec 2016.

[17] Google Inc, "Privacy Policy," 2018. [Online]. Available: https://policies.google.com/privacy.

[18] Qu, Licheng & Li, Wei & Li, Wenjing & Wang, Y., "Daily long-term traffic flow forecasting based on a Deep Neural Network," *Expert Systems with Applications*, vol. 121, Dec 2018.

[19] Kumar, Kranti & Parida, Manoranjan & Katiyar, V., "Short Term Traffic Flow Prediction for a Non Urban Highway Using Artificial Neural Network," *Procedia - Social and Behavioral Sciences*, vol. 104, 2013.

[20] Zheng, Z., Wang, D., Pei, J., Yuan, Y., Fan, C., & Xiao, F, "Urban Traffic Prediction through the Second Use of Inexpensive Big Data from Buildings," *In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management - CIKM '16,* p. 1363–1372, 2016.

[21] Yu, Haiyang & Wu, Zhihai & Wang, Shuqin & Wang, Yunpeng & Ma, Xiaolei, "Spatiotemporal Recurrent Convolutional Networks for Traffic Prediction in Transportation Networks," *Sensors,* vol. 27, p. 1501, 2017.

[22] Ali U., Mahmood T., Using Deep Learning to Predict Short Term Traffic Flow: A Systematic Literature Review, 2018, pp. 90-101.

[23] Yuankai, Wu & Tan, Huachun, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," 2016.

[24] Du, Shengdong & Li, Tianrui & Gong, Xun & Yu, Zeng & Horng, Shi-Jinn, " A Hybrid Method for Traffic Flow Forecasting Using Multimodal Deep Learning," *International Journal of Computational Intelligence Systems,* vol. 13, 2018.

[25] Chollet, "Keras Documentation," Github, 2015. [Online]. Available: https://keras.io/examples/conv_lstm/.

[26] L. Biewald, Github, January 2019. [Online]. Available: https://github.com/lukas/ml-class/blob/master/videos/video-predict/video.ipynb .