ENHANCED TRAFFIC FLOW FORECASTING ON A SUBURBAN HIGHWAY USING EMPIRICAL MODE DECOMPOSITION PREPROSESSED DEEP LEARNING MODEL

by

MD. AUSAF ALAM

A thesis submitted to the

Department of Civil Engineering

in partial fulfillment for the degree of

BACHELOR OF SCIENCE IN CIVIL ENGINEERING

Department of Civil Engineering
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh

March 2025

BOARD OF EXAMINERS

The thesis/project titled “Enhanced Traffic Flow Forecasting on a Suburban Highway Using Empirical Mode Decomposition Preprocessed Deep Learning Model” submitted by MD. Ausaf Alam, Student Id.: 1904080, has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Bachelor of Science in Civil Engineering on 17th March 2025.

DECLARATION

I hereby declare that this thesis/project is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or have been accepted for the award of any other degree or diploma at Bangladesh University of Engineering and Technology (BUET) or any other educational institution, except where due acknowledgement is made. I also declare that the intellectual content of this thesis is the product of my own work and any contribution made to the research by others, with whom I have worked at BUET or elsewhere, is explicitly acknowledged.

MD. Ausaf Alam

Dedicated

to

“My Parents”

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor, Dr. Shamsul Hoque, for his invaluable guidance, continuous support, and insightful supervision throughout the course of this research. His expertise and encouragement have been instrumental in shaping this thesis.

I am also sincerely grateful to the rich library of the Civil Engineering Department, which provided access to an extensive collection of reference papers that significantly contributed to the literature review and methodology development of this study.

My heartfelt appreciation goes to the Highway Police Headquarters for providing recorded video footage of the Dhaka-Chittagong National Highway, which served as the primary data source for this research. Their cooperation and assistance were crucial in facilitating the data collection process.

I extend my special thanks to Sigmind, an AI startup, for their support in computer vision technology for automatic vehicle counting. Their expertise in AI-based traffic analysis greatly aided the accuracy and efficiency of data extraction in this study.

Finally, I would like to acknowledge the Civil Engineering Department for their financial support, which enabled the successful completion of this research.

Without the contributions and support of these individuals and institutions, this research would not have been possible. I am truly grateful for their assistance and encouragement.

ABSTRACT

Accurate traffic flow prediction is critical for efficient traffic control, urban planning, and congestion mitigation management. This study focused on using deep learning algorithms to predict short-term traffic flow on suburban roadways. Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Artificial Neural Network (ANN), and 1D Convolutional Neural Networks with LSTM (Conv1D-LSTM) models were used as baseline models The study used Empirical Mode Decomposition (EMD), a non-stationary time series analytic approach, to eliminate outliers and smooth the dataset in order to addressing the stochastic character of traffic flow. The dataset was derived from traffic surveillance video footage collected by the Highway Police Command Center on a two-way, four-lane suburban highway. Computer vision techniques, in collaboration with Sigmind, an AI-driven startup, were used to extract 5-minute interval vehicle counts from 6:00 AM to 12:00 PM. Additionally, manual counting of vehicles from 12:00 PM to 3:00 PM extended the dataset to span 6:00 AM to 3:00 PM over 11 continuous days. Initial data analysis revealed overall trends, hourly fluctuations, and anomalies, which were effectively mitigated using EMD preprocessing.

According to the results, the EMD-preprocessed LSTM model demonstrated better forecasting accuracy, achieving the lowest RMSE (23.18) and MAPE (8.55&) than GRU, ANN, and Conv1D-LSTM in capturing time-dependent interactions and non-linear traffic behaviors. These results highlight how crucial it is to combine deep learning techniques with sophisticated preprocessing approaches like EMD to increase prediction accuracy and resilience. This study highlights the potential of LSTM for precise short-term traffic forecasting on suburban roadways and offers insightful information for creating adaptive traffic management methods.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ANN	– Artificial Neural Network
ARIMA	– Autoregressive Integrated Moving Average
CNN	– Convolutional Neural Network
Conv1D	– One-Dimensional Convolutional Neural Network
Conv1D-LSTM	– One-Dimensional Convolutional Long Short-Term Memory
EMD	– Empirical Mode Decomposition
FCD	– Floating Car Data
GRU	– Gated Recurrent Unit
IMF	– Intrinsic Mode Function
IQR	– Interquartile Range
ITS	– Intelligent Transportation Systems
LSTM	– Long Short-Term Memory
MAE	– Mean Absolute Error
MAPE	– Mean Absolute Percentage Error
MLP	– Multilayer Perceptron
RF	– Random Forest
RMSE	– Root Mean Squared Error
RNN	– Recurrent Neural Network
SAE	– Stacked Auto-Encoders
STARIMA	– Space-Time Autoregressive Integrated Moving Average
SVM	– Support Vector Machine
TPU	– Tensor Processing Unit

CHAPTER 1
INTRODUCTION

Predicting traffic flow is a crucial component of Intelligent Transportation Systems (ITS), which aims to tackle the growing issues of urban development, road congestion, and transportation management. As countries experience growth that necessitates improved connectivity between cities and regions, ensuring efficient traffic movement has become a critical focus for suburban designers and transportation planners. Precise traffic flow forecasts can greatly improve road network performance, alleviate congestion, and promote environmental conservation by cutting down on fuel use and emissions. By implementing proactive traffic control measures, authorities can decrease travel time delays, optimize road infrastructure usage, and enhance public safety through the reduction of potential

The emergence of data-centric technologies has revolutionized Intelligent Transportation Systems (ITS), shifting from conventional traffic management approaches to more anticipatory and adaptive frameworks. By incorporating data driven technology and computational models, ITS now offers a holistic approach to efficiently analyze, forecast, and resolve traffic challenges. This evolution plays a crucial role in meeting the demands of expanding urban communities and facilitating intelligent mobility solutions for contemporary citi es

1.1 Background and Motivation

Traffic flow forecasting plays a critical role in modern intelligent transportation systems (ITS) by enabling effective traffic management, congestion reduction, and improved road safety. Accurate traffic prediction helps optimize signal control, route planning, and transportation infrastructure development, benefiting both commuters and policymakers. However, traffic patterns are inherently complex, nonlinear, and influenced by multiple dynamic factors, including weather conditions, road incidents, peak hours, and urban expansion.

The extremely variable character of real-world traffic flow is frequently difficult to capture for traditional traffic forecasting models, such as statistical techniques (like Autoregressive Integrated Moving Average, or ARIMA) and machine learning models (like Support Vector Machines, or SVM, and Random Forest, or RF). By learning complex behavior from past data, deep learning models like Artificial Neural Networks (ANNs), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRUs) have greatly increased prediction accuracy. Nevertheless, deep learning algorithms still struggle to handle multi-scale variances and high-frequency swings in traffic data.

To address these challenges, Empirical Mode Decomposition (EMD) is introduced as a preprocessing technique to enhance the predictive performance of deep learning models. EMD is a data-driven signal decomposition method that breaks down complex time-series data into a set of simpler oscillatory components known as Intrinsic Mode Functions (IMFs). By decomposing traffic data into different frequency components, EMD helps deep learning models better capture short-term variations and long-term trends, leading to improved forecasting accuracy.

This research explores the integration of EMD-processed deep learning models for enhanced traffic flow forecasting. By applying ANN, LSTM, GRU, and a hybrid Conv1D-LSTM model to EMD-processed data, the study aims to develop a more robust and accurate prediction framework. The proposed approach is expected to improve forecasting precision, particularly in environments with highly volatile and irregular traffic patterns, such as highways and urban road networks.

1.2 Problem Statement

Accurate traffic flow forecasting in suburban areas is essential for effective transportation planning and congestion management. Unlike urban roads with high-density traffic or highways with high-speed, free-flowing movement, suburban roads experience moderate congestion, mixed vehicle types, and irregular peak-hour variations, making prediction more complex. Traditional statistical models, such as

ARIMA, and machine learning methods, like SVM and Random Forest, struggle to handle these nonlinear and fluctuating traffic patterns, leading to inaccurate forecasts.

Deep learning models, including ANN, LSTM, GRU, and Conv1D-LSTM, have shown promising results in capturing complex behavior in traffic data. More importantly, deep learning can leverage its ability to make fairly accurate predictions even with a low dataset.

This study aims to develop an enhanced traffic forecasting framework by integrating Empirical Mode Decomposition (EMD) with deep learning models to handle nonlinearity and the complex dynamics of traffic flow while ensuring accurate predictions even with a limited dataset.

1.3 Objective of the Research

This research aims to develop an enhanced traffic flow forecasting framework by integrating Empirical Mode Decomposition (EMD) with deep learning models to improve predictive accuracy.

The specific objectives are:

1. To capture the random fluctuation of traffic flow and then to smoothen the dataset using EMD.
2. To build and analyze the deep learning models.
3. To evaluate the performance of models by RMSE & MAPE.

1.4 Scope of the Research

This study is applicable to 4-lane suburban highways and roadways with similar geometric configurations. It focuses specifically on traffic flow forecasting, utilizing data collected during the daytime period from 6 AM to 3 PM, which represents typical traffic conditions for peak and off-peak hours in suburban areas. The dataset used for this research consists of total vehicle counts recorded at 5-minute intervals, providing high-resolution temporal data for analysis. Additionally, the study is designed to address traffic flow datasets characterized by random fluctuations and noise, which are common in real-world traffic data. To handle these challenges, the Empirical Mode Decomposition (EMD) technique is employed to preprocess the data. EMD is particularly suited for decomposing complex traffic patterns into intrinsic mode functions, which helps in mitigating the effects of noise and improving the accuracy of the subsequent deep learning models for forecasting.

The scope of this study is limited to a dataset spanning 11 consecutive days, restricting the analysis to a short time window. While the data covers a specific period, the methodologies and findings are applicable to traffic forecasting problems in regions with similar traffic patterns and road structures. Furthermore, due to the limited dataset, the study may not capture long-term traffic variations or account for seasonality, which should be considered when interpreting the results. Despite these limitations, the proposed approach offers valuable insights into enhancing traffic flow prediction under specific conditions.

1.5 Organization of the Research

1. **Chapter 1: Introduction** – gives an introduction of the relevant research background details, presents state of the problem, objectives of the thesis and overview of this thesis.

2. **Chapter 2: Literature Review** – Focus on the review of the previous studies related to thesis papers and other sources related to traffic data, prediction models, and data denoising techniques by comparing different approaches and recommendations

3. **Chapter 3: Methodology** – Presents the basic concepts and structure regarding data collection, extraction, data smoothening, prediction models, and evaluation metrics that have been discussed.

4. **Chapter 4: Data Analysis & Result** – Describes the results and discussion of the proposed methodological framework

5. **Chapter 5: Conclusion & Recommendation –** Highlights the overall study procedure, summarizes the findings, and provide limitations and recommendations

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

Traffic flow forecasting relies on a thorough understanding of the foundational components of traffic data and the predictive models used to analyze them. This chapter provides a comprehensive overview of existing studies and methods that form the basis of this research. The literature review is structured into the following key areas: traffic data, data transcription, traffic counting, and prediction models.

2.2 Literature Review

2.2.1 Traffic Data

Traffic data serves as the backbone of traffic flow forecasting, providing the necessary inputs for analyzing and predicting vehicle movement patterns across road networks. Reliable and comprehensive traffic data is essential for building accurate models, as it captures critical details about the spatial-temporal dynamics of traffic flow.

2.2.1.1 Types of Traffic Data

Traffic flow forecasting relies on the availability of high-quality traffic data, which can be broadly categorized into two types: core traffic data and auxiliary traffic data [1]. Core traffic data forms the quantitative foundation for most forecasting models, comprising fundamental metrics such as traffic volume, speed, density, and flow. These parameters are directly tied to vehicular movement and are typically obtained through technologies like inductive loop detectors, radar sensors, and video analytics. They provide vital insights into the temporal and spatial dynamics of traffic, serving as essential inputs for predictive modeling. On the other hand, auxiliary traffic data accounts for contextual and environmental factors that indirectly influence traffic patterns. This includes information on incidents like accidents, road work, events, holidays, weather conditions, road attributes (e.g., lanes, type, speed limits), and pedestrian and driver behaviors. By integrating these two categories, traffic flow forecasting models can more comprehensively analyze and predict traffic dynamics, addressing both the core metrics of vehicular movement and the broader contextual variables. This dual approach enhances the accuracy and robustness of predictions, supporting more effective strategies for traffic management and planning.

Several studies have focused exclusively on core traffic data, particularly flow characteristics, as the primary input for forecasting. For instance, [2], [3] utilized traffic flow to capture granular temporal dynamics. Similarly, [4] incorporated flow and weekly patterns, emphasizing longer-term trends in traffic forecasting. Other studies, such as [1] used flow per minute, while [5] focused on flow per lane, highlighting the importance of spatial granularity for road-specific traffic analysis. In addition to core traffic data, some researchers have expanded their datasets to include both core and auxiliary data. For example, [6] integrated speed, flow, density, and weather conditions (temperature and precipitation) to capture the impact of environmental factors on traffic behavior. Similarly, [7] utilized traffic volume alongside calendar-based variables, such as year, month, day, hour, and holidays, accounting for predictable variations caused by human activity and scheduling. A few studies have adopted more comprehensive datasets, combining categorized core traffic data with auxiliary variables. For instance, [8] and [9] incorporated categorized flow, speed, days, and lane-specific characteristics, providing a detailed representation of traffic dynamics. These studies underline the importance of integrating core and auxiliary traffic data to better understand and predict complex traffic patterns.

2.2.1.2 Data Collection

Effective traffic flow forecasting relies on robust and comprehensive data collection methods. Traffic data is generally categorized into macroscopic and microscopic levels, each offering unique insights into traffic behavior and dynamics. These data types differ in granularity and application, serving distinct purposes in traffic analysis and forecasting.

1. Macroscopic Data Collection

Macroscopic data focuses on aggregated traffic characteristics and provides a broad overview of traffic flow. It captures system-wide traffic behavior by measuring parameters such as traffic flow, speed, and density at the network or roadway segment level. This type of data is particularly useful for modeling traffic at a higher scale, such as analyzing congestion trends o assessing the overall performance of a transportation system.

Macroscopic data is typically collected using technologies such as:

a) **Inductive Loop Detectors**: One of the most popular instruments for gathering macroscopic traffic data is an inductive loop detector. These gadgets create electromagnetic fields by embedding wire loops into the surface of the road. The system can identify the presence, speed, and number of vehicles by recording the change in the magnetic field that occurs when they travel over or stop on the loop. Inductive loops are perfect for fixed places like roads since they are extremely accurate, run continuously, and perform well in a variety of weather conditions. They are only used on important roads, though, because their installation and upkeep are expensive and interfere with traffic flow.

b) **Radar Sensors**: Radar sensors monitor a vehicle's position, speed, and distance using radio waves. They can monitor several lanes at once and are usually mounted on poles or gantries. Radar sensors are dependable in rain, fog, and darkness because of their exceptional performance in a variety of weather and lighting scenarios. They are frequently used to track traffic and travel times on highways. Radar sensors have many benefits, but they can be costly to install and may be vulnerable to interference from other radio frequencies, which might make them less useful in crowded areas.

c) **Video Surveillance Systems**: Video surveillance systems capture traffic data through cameras deployed in strategic areas, which are frequently paired with image processing software to extract data such as vehicle counts, speeds, and congestion levels. These technologies support real-time visual monitoring and post-event analysis. When combined with artificial intelligence, video systems can detect accidents or stalled vehicles and classify them. However, environmental conditions such as rain, glare, and bad lighting can all have an impact on their performance. These systems also demand a significant amount of data storage and computational resources to function.

2. Microscopic Data Collection

Microscopic data captures individual vehicle behavior and interactions within the traffic flow. This data includes detailed parameters such as vehicle trajectories, lane-changing behavior, acceleration and deceleration patterns. Microscopic data provides a granular view of traffic dynamics, enabling a more precise analysis of vehicular movements and interactions.

Microscopic data is typically collected using the following methods:

a) **GPS Tracking**: GPS tracking uses satellite technology to track the whereabouts, speeds, and movements of vehicles in real time. This approach is widely used in navigation systems, fleet management, and traffic research and is especially helpful for documenting intricate individual trip patterns. Granular data from GPS can be used to examine route preferences and traffic. However, signal obstacles may restrict its usefulness in urban canyons or tunnels, and its dependence on user input may lead to inadequate statistics.

b) **Floating Car Data (FCD)**: Floating car data is derived from GPS-enabled vehicles or mobile devices that transmit location and speed information in real-time. This method offers extensive spatial coverage, especially in regions with limited fixed sensor infrastructure, and is widely used for travel time estimation, congestion monitoring, and route optimization. However, FCD depends on the voluntary participation of users, which can lead to data gaps, and raises concerns regarding privacy due to individual tracking.

c) **Microscopic Simulators**: Microscopic simulators, including SUMO, AIMSUN, and VISSIM, simulate how individual cars travel across a traffic network. These simulators mimic vehicle interactions, lane-changing behavior, and travel times under varied conditions using input from real-world traffic figures. When it comes to testing traffic control tactics and improving network performance without interfering with actual traffic, they are vital. Despite offering in-depth insights, these tools can be computationally demanding and rely on precise calibration and input data to be effective.

Several studies have relied on inductive loop detectors for traffic data collection due to their ability to provide highly accurate measurements of traffic volume, speed, and occupancy. For instance, [6] used inductive loop data to monitor congestion trends on highways, leveraging its reliability in providing continuous, real-time data. Similarly, [10] incorporated loop data for long-term traffic forecasting, focusing on aggregated metrics. The merit of this method lies in its precision and ability to operate under various weather conditions, but its high installation and maintenance costs, along with the need for physical road embedding, limit its application to fixed locations. Radar sensors have been employed in studies like [7], [1] to monitor multi-lane traffic flow on arterial roads. These studies highlight the advantage of radar sensors in collecting data under adverse weather conditions, such as rain or fog, which can compromise the performance of other sensors. However, [11] noted that radar sensors are prone to interference from other radio waves, leading to occasional inaccuracies in dense traffic environments. Despite this limitation, radar sensors remain a preferred choice for freeway monitoring due to their wide coverage and ability to capture real-time speed and density data.

Video surveillance, coupled with advanced analytics, has been widely adopted for urban traffic monitoring. For example [4], [9] utilized video analytics to analyze traffic congestion at intersections, extracting vehicle trajectories and identifying bottlenecks. Similarly, [12] used video systems to classify vehicles and predict traffic density during peak hours. The advantage of video analytics lies in its ability to provide rich contextual data and visual confirmation of traffic patterns. However, both studies noted challenges such as high computational requirements and sensitivity to environmental factors like glare, rain, and poor lighting, which can affect data accuracy. Microscopic studies, such as [13] relied on GPS data from individual vehicles to analyze travel behavior and predict route-specific congestion. The study emphasized the granularity of GPS data, which enables precise trajectory tracking and travel time analysis. Similarly,[14] used GPS data to capture real-time movement patterns in suburban areas. While GPS tracking offers detailed insights and broad spatial coverage, it is susceptible to signal loss in urban canyons or tunnels and depends heavily on user participation, which can result in incomplete datasets. Floating car data has been leveraged in studies like [15], [16] for real-time traffic monitoring and travel time estimation. [15] highlighted FCD's ability to cover large areas at a low cost, particularly in regions lacking traditional sensor infrastructure. However, [16] noted that FCD depends on voluntary data sharing, leading to potential biases and underrepresentation in specific areas. Despite privacy concerns, FCD remains a cost-effective solution for dynamic traffic forecasting and congestion monitoring. Microscopic simulators, such as those used in [17] provide a virtual environment to test traffic management strategies. This study utilized simulators to analyze lane-changing behavior and the impact of traffic control measures under varying scenarios. While these tools allow for scenario testing without disrupting real-world traffic, [17] appointed out that they require extensive calibration and input data to produce realistic results, making them resource-intensive and time-consuming.

2.2.2 Prediction Model

Prediction models used in traffic flow forecasting can be broadly divided into three categories: deep learning, parametric, and genetic programming. Because of their capacity to identify intricate spatial-temporal patterns and nonlinear correlations in traffic data, deep learning models—such as Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), and Graph Convolutional Networks (GCNs)—have become increasingly popular. Despite requiring a significant amount of training data and computational power, these models are excellent at digesting enormous datasets and producing predictions that are extremely accurate. Parametric models, including state-space models and Autoregressive Integrated Moving Average (ARIMA), employ statistical methods predicated on predetermined hypotheses regarding the distribution of the underlying data. Although these models work well for smaller datasets and simpler traffic patterns, they frequently have trouble with the dynamic and nonlinear nature of real-world traffic. Using algorithms influenced by natural selection, genetic programming models optimize traffic forecast functions in an evolutionary manner. Despite the possibility of inconsistent outcomes due to their stochastic character, these models are especially helpful in resolving complicated, multi-objective issues where conventional approaches are inadequate. From real-time operations to long-term planning, traffic forecasting systems can be customized to satisfy a variety of needs by combining these methods or choosing models according to particular use cases.

2.2.2.1 Deep Learning Models

The ability of deep learning models to capture complex patterns and relationships in data has led to their widespread adoption and popularity in traffic flow prediction. Several types of deep learning models have been identified in the literature, namely Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs) and Graph Neural Network (GNN).

1. Recurrent Neural Network (RNN)

The most used methods usually utilize Recurrent Neural Networks (RNN) and their variants, as in Long Short Term Memory (LSTM), to extract spatial relationships from the whole city by modelling citywide traffic. For instance, [18] LSTM to predict traffic flow and speed. The authors used three prediction designs: one-to-one (train the model with the temporal information of one road to predict the traffic flow in the same road), many-to-one (train the model with temporal information of several roads to predict one road), and many-to-many (train the model with temporal information of several roads and predict multiple roads). On the other hand, [19] implemented four different Deep Learning methods (LSTM, Gated Recurrent Unit (GRU), Spatiotemporal Recurrent Convolutional Network (SRCN) and a High Order Graph Convolutional Long Short-Term Memory (HGC-LSTM)) to perform traffic forecasting in urban contexts, using floating car data to predict the average speed of the network road sections. In addition to using statistical models, [20] used LSTM to predict traffic to avoid traffic congestion by adjusting the phase duration of the traffic lights that control each crossroad according to the real traffic conditions. [21] proposed an Outlier Enriched Long Short Term Memory (OE-LSTM) model to predict traffic flow, which employs a multi-step framework to detect outliers in the traffic flow and uses these outliers to learn the spatio-temporal correlations between different locations in the traffic network. Finally, [22] used LSTM, GRU and Stacked Auto-Encoders (SAEs) to predict traffic flow in Finland.

2. Convolutional Neural Networks

Convolutional neural networks (CNN) are typically used to process images and assign importance to different aspects of an image in order to differentiate one from the other. However, one can apply CNNs in different contexts as long as the input is organized in the format of an image. For instance, [23] used the Convo-Recurrent Attentional Neural Network (CRANN) model, which combines neural modules (temporal, spatial, and dense) to exploit the various components identified in a spatio temporal series: seasonality, trend, inertia and spatial relations. Also, [24] proposed an error-recurrent convolutional neural network (eRCNN) to predict in the short term the three fundamental traffic variables (traffic density, traffic flow, and space mean speed) using Floating Car Data (FCD). The FCD was organized to be an image with different channels (like an RGB image). On the other hand, [25] proposed a spatiotemporal traffic prediction model, namely CPM-ConvLSTM, consisting of three steps (congestion propagation pattern graph construction, spatial matrix construction, and congestion level prediction) to make a short-term prediction of the congestion level for each segment of the road.

3. Graph Neural Networks

Regarding non-Euclidean structured data, such as spatial networks, some studies use Graph Convolution Networks (GCNs) to capture spatial patterns. For instance, [26] used a Graph Convolutional Neural Network for traffic flow prediction taking into account daily and weekly patterns of traffic flow distributions. While [27] used a High-Order Graph Convolutional Long Short-Term Memory Neural Network (HGC-LSTM), which applies a CNN to the network graph encoded as a matrix.

4. Other Neural Networks

Besides CNNs, GCNs, and RNNs (and its variants), other neural networks were identified in the literature to tackle the problem of traffic flow prediction, such as General Regression Neural Network (GRNN), or Autoencoders. For instance [28] used a GRNN for short-term traffic prediction, taking advantage of the fast learning ability and the convergence to the optimal surface. To demonstrate the appropriateness of GRNN, the authors compared the result with a Multi-Layer Perceptron (MLP) and a group method for data handling (GMDH) neural network. On the other hand, [29] proposed a spatiotemporal deep learning model for the spatiotemporal potential energy fields (similar to water flow driven by the gravity field), consisting of a temporal component to model the temporal correlation and a spatial component to model spatial dependencies. While [30] used a GRNN, which consists of three independent components with the same structure. Each component considers the recent time series, the daily-periodic time series, and the weekly-periodic time series with different patterns in traffic data, respectively. Also, [4] implemented a Feed-Forward Neural Network (FFNN) to predict traffic for future temporal horizons of 5, 10, 15, 30 and 45 minutes and used four different input settings (only historical values of the target sensor to predict a future value, only historical values from nearest neighbors excluding the target sensor, only historical values from nearest neighbors including the target sensor, and historical values from all sensors).

2.2.2.2 Parametric Model

Parametric models, including both traditional statistical models and ensemble methods, have also been widely used in traffic flow prediction.

1. Auto-regressive models

This kind of model focuses on predicting the future based on data from the past, typically it uses a linear combination of the past values of the variables. Extensive studies exist on traffic flow prediction, and the most used approaches were statistical models, such as Auto-Regressive Integrated Moving Average (ARIMA) [31] . More specifically, [20] applied ARIMA and LSTM to predict real urban traffic from the city of Bucharest and compare and discuss the applicability of both methods to that scenario. While [31] applied ARIMA to predict future traffic density on specific roads of Slovenia. Also, Space-Time ARIMA (STARIMA) model was used by [32] to predict traffic flow by considering multiple traffic-related features from one road that can influence other roads of the network.

2. Regression Model

Regression models are used to model the relationship between the target variable (in this context, traffic flow) and one or more independent variables. There are several types of regression models, namely linear regression, logistic regression, and polynomial regression, among many others. For instance, [33] applied Linear Regression, Sequential Minimal Optimization (SMO) Regression, and M5 Base Regression Tree and Regression Trees to make predictions of traffic flow in the city of Porto, Portugal. [34] applied several different machine learning models, including multiple regression, to predict traffic in the city of Braga, Portugal, using data collected by a fleet of buses from the local public transport company

3. Instance-Based Model

This is a family of algorithms that do not try to model any distribution for the training data but instead compare new instances with the ones seen in the training data. The most used algorithm in this family is the k-nearest neighbor (k-NN). For instance, [34] used k-NN to make predictions on the city of Braga, Portugal, by averaging the predictions to the closest values for the input values.

4. Other Parametric Model

Finally, the search query identified other parametric models such as support vector machines and Kriging-based models. For instance, [35] used Support Vec tor Machine models to predict traffic flow one hour ahead. While [36] used a Linear Predictor (Kriging algorithm) and a Multi Model Bayesian Kriging model to predict urban traffic, which is able to represent congested regions and interactions in upstream and downstream areas.

2.2.2.3 Genetic Programming

Finally, [37] proposed the GENetic Programming with Transfer Learning (GENTLE) algorithm to generate Single Source Trans fer Learning (SSTL) and Multiple Source Transfer Learning (MSTL) models. The resulting algorithm uses knowledge from other road segments to predict vehicle flow through a junction where traffic data are unavailable.

Besides all of these models, many researchers have incorporated data decomposition techniques in order to smoothen the dataset and get more prediction accuracy. Various outliers suppressed techniques like wavelet transform, Fourier transform, Ensemble empirical mode decomposition, principal component analysis has been used by [38], [39], [40] smoothen the dataset and enhance prediction accuracy.

2.2.3 Performance Evaluation

After applying the suitable preprocessing techniques to the data, dividing the dataset into train and test sets (in some cases, train, test, and validation sets), and applying the methods explained in the previous subsection, it is crucial to evaluate the results and the performance of the methods. Hence, the results obtained by the approaches are compared to the real value of the data. Some state-of-the-art metrics are used to do so, but the metrics are different for prediction and classification methods. For prediction models, evaluation metrics measure the error between predicted observations and real values. The most frequently used metrics are Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE)

1. Root Mean Squared Error (RMSE)

RMSE is one of the most widely used metrics for evaluating traffic forecasting models due to its ability to penalize larger errors. Studies such as those by [33], [19] and [24] employed RMSE to measure the overall deviation of predicted traffic values from observed data. The metric's sensitivity to large errors makes it particularly useful for identifying extreme forecasting inaccuracies. However, this sensitivity may overemphasize outliers, which can skew the results in datasets with high variability.

2. Mean Absolute Error (MAE)

MAE is a straightforward metric that calculates the average of absolute errors between predicted and actual traffic values. It is less sensitive to outliers compared to RMSE and provides a clear measure of average prediction accuracy. Researchers like [18], [33], and [22] have employed MAE for evaluating models in scenarios where balanced accuracy across all predictions is prioritized. Its simplicity and interpretability make it a common choice for performance comparisons.

3. Mean Absolute Percentage Error (MAPE)

MAPE expresses errors as a percentage, making it particularly useful for comparing model performance across datasets with varying scales. Studies such as those [35], [41] used MAPE to evaluate forecasting models in diverse traffic scenarios. While MAPE offers interpretability, it can become unreliable when actual traffic values are close to zero, leading to disproportionately high error percentages.

4. Coefficient of Determination (R^2)

The Coefficient of Determination, or R^2, evaluates the proportion of variance in the observed data that is explained by the model. Studies such as those by [18], [29] employed R^2 to assess the goodness of fit of traffic forecasting models. R^2 is particularly useful for understanding model performance in explaining traffic trends, but it does not directly measure error magnitudes.

These measures all offer complementing information about how well traffic flow forecasting algorithms are doing. While MAE and MAPE offer more balanced metrics of average performance, RMSE and MSE are useful for penalizing greater errors. R2 provides an additional level of interpretability by elucidating the percentage of variation that the model captures. When combined, these indicators provide a thorough set of tools for assessing and enhancing forecasting techniques.

2.3 Summary

Through this chapter, the study provided a detailed synthesis of existing knowledge, identifying gaps and opportunities that justify the need for this research. The insights gathered here will serve as a foundation for the methodological framework and experimental design presented in subsequent chapters.

CHAPTER 3
METHODOLOGY

3.1 Introduction

This chapter outlines the methodology employed to conduct the research for this thesis. The primary goal of this study is to enhance traffic flow forecasting using an Empirical Mode Decomposition (EMD)-preprocessed deep learning model. To achieve this, a series of steps are carried out including data collection, extraction, analyzing, smoothening, model development, fine tuning, prediction and performance evaluation.

3.2 Methodology Overview

The methodology begins with data collection and extraction, where traffic flow data, consisting of vehicle counts at 5-minute intervals, were gathered from a 4-lane suburban highway. Following data collection, the data analysis process is performed, which involves identifying patterns and trends within the raw data. The next step is data smoothing, achieved through the application of the EMD technique to decompose the time-series data into intrinsic mode functions (IMFs), thereby reducing noise and random fluctuations.

Once the data is preprocessed, model development takes place. Deep learning models are trained on the preprocessed data, and fine-tuning techniques are applied to optimize the model's performance. Prediction of future traffic flow is then carried out using the trained model, followed by performance evaluation using relevant metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). Figure 3.1 shows the methodological framework of the research.

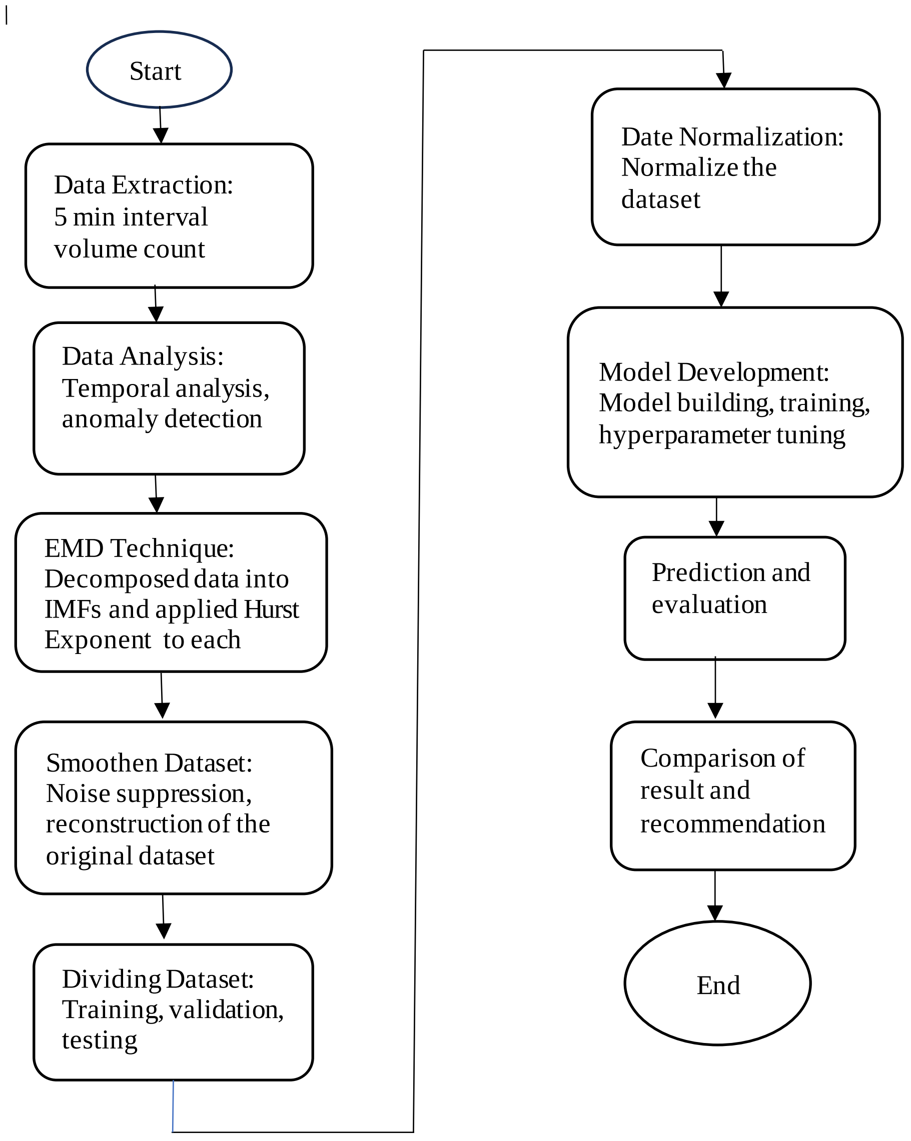

Start
Data Extraction: 5 min interval volume count
Data Analysis: Temporal analysis, anomaly detection
EMD Technique: Decomposed data into IMFs and applied Hurst Exponent to each
Smoothen Dataset: Noise suppression, reconstruction of the original dataset
Dividing Dataset: Training, validation, testing
Date Normalization: Normalize the dataset
Model Development: Model building, training, hyperparameter tuning
Prediction and evaluation
Comparison of result and recommendation
End

Figure 3.1 Flowchart of the Proposed Methodology

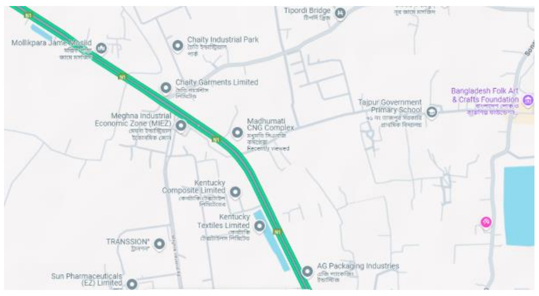
Throughout the chapter, the techniques, procedures, equipment, and specialized software tools used for analysis and modeling are discussed.

3.3 Data Collection

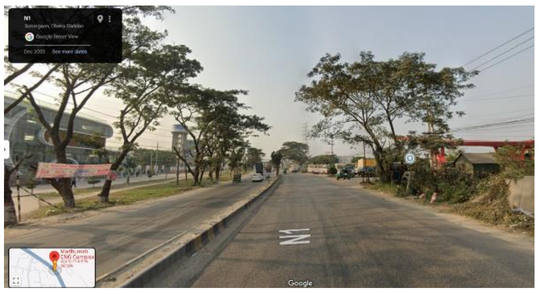
The data was collected from a traffic surveillance video footage of the National Highway. The Highway Police Command and Monitoring Center, located near the Meghna Toll Plaza, provided video footage of traffic flow. A detailed description of the site and data are described below.

3.3.1 Site Description

The traffic flow dataset was collected from the Modhumoti CNG Pump Station, situated around 24 kilometers outside Dhaka city along the Dhaka-Chittagong N-1 National Highway. This is a four-lane, two-way arterial road that serves as a vital transportation corridor, facilitating the movement of goods and passengers between Dhaka, the capital of Bangladesh, and Chittagong, the country's largest port city. Because of its strategic importance in national trade and commerce, the Dhaka-Chittagong highway experiences high traffic volumes; therefore, it a crucial location for studying traffic patterns and flow dynamics. The following two figures 3.1 (a), (b**)** present a Google Maps view highlighting the actual location of the site along the Dhaka-Chittagong N-1 National Highway and a street-level view of showcasing the surrounding road infrastructure and traffic conditions.



Tipordi Bridge
Chaity Industrial Park
Mollikpara Jame Masjid
Chaity Garments Limited
Meghna Industrial Economic Zone (MIEZ)
Madhumati CNG Complex
Tajpur Government Primary School
Bangladesh Folk Art & Crafts Foundation
Kentucky Composite Limited
Kentucky Textiles Limited
TRANSSION
Sun Pharmaceuticals (EZ) Limited
AG Packaging Industries



(a) (b)

Figure 3.2 (a)

3.3.2 Data Description

A single traffic surveillance video camera was used to collect the data, positioned at the selected location to ensure the optimal coverage of the roadway. The data was collected for a span of 11 consecutive days, from September 19, 2024, to September 29, 2024, capturing real-world traffic conditions over an extended period.

The video footage recorded traffic flow in both directions, providing a comprehensive view of vehicle movement patterns. The camera operated at a resolution of 1280×720 pixels, ensuring clear visibility of vehicles and road conditions. Additionally, the footage was recorded at a frame rate of 25 frames per second (fps), allowing for detailed motion analysis and accurate tracking of vehicular movements.

This high-quality video data serves as a valuable resource for traffic flow analysis, vehicle classification, and other transportation-related studies.

3.4 Data Extraction

The total traffic volume was an aggregated count of various vehicle types, including bikes, CNGs, auto-rickshaws, cars, jeeps, pickups, microbuses, buses, trucks, container trailers, and tanker lorries. Traffic volume count data was recorded at 5-minute intervals from 6:00 AM to 3:00 PM over 11 consecutive days.

To ensure accurate and reliable data extraction, computer vision technology was employed to automate the traffic counting process, with technical assistance from Sigmind. Additionally, manual vehicle counting was conducted for a specific period to complement the automated extraction and validate the dataset. This combined approach ensured comprehensive and reliable data collection.

With considering both side traffic directions, the dataset comprised 108 data points per day, resulting in a total of 1,188 data points over the 11-day period.

3.4.1 Automatic Extraction

Traffic counts were primarily extracted using computer vision-based technology of automated detection. The system, conducted by Sigmind, processed the recorded video footage to identify and classify vehicles with high accuracy. This approach paved a way for efficient and consistent data collection throughout the study period. To demonstrate this process, Appendix-A provides a sample of the raw data output from Sigmind of 19th September 2024, showing the detailed vehicle detection and classification information, including timestamps and unique category IDs for each vehicle. Next this raw data was processed using Python code to aggregate the counts into 5-minute intervals, the results of which are presented in Appendix-B. These appendices provide a transparent view of the automated extraction methodology and the subsequent data aggregation process.

3.4.2 Manual Extraction

Manual car counting was done every day from 12:00 PM to 3:00 PM in order to improve the dependability of the dataset. This extra manual check helped resolve any possible inconsistencies in the data and guaranteed the precision of the automatic extraction. When both approaches were combined, a solid and thoroughly tested dataset for traffic analysis was produced.

3.5 Exploratory Data Analysis

The extracted total volume of traffic count in five minutes interval shows hourly pattern. Additionally, it shows random fluctuations. To find the outliers and to find the temporal hourly variation of traffic flow, box plot were generated. Later to smoothen the dataset, EMD technique was applied.

3.5.1 Temporal Pattern and Anomaly Detection

"A box plot (or box-and-whisker plot) shows the distribution of quantitative data in a way that facilitates comparisons between variables or across levels of a categorical variable. The box shows the quartiles of the dataset while the whiskers extend to show the rest of the distribution, except for points that are determined to be "outliers" using a method that is a function of the inter-quartile range" [42].

It represents data distribution through key metrics (minimum, Q1, median, Q3, maximum) and highlights outliers beyond 1.5 times the interquartile range. The figure. 3.3, adapted from [43] illustrates the overall concept of a box plot.

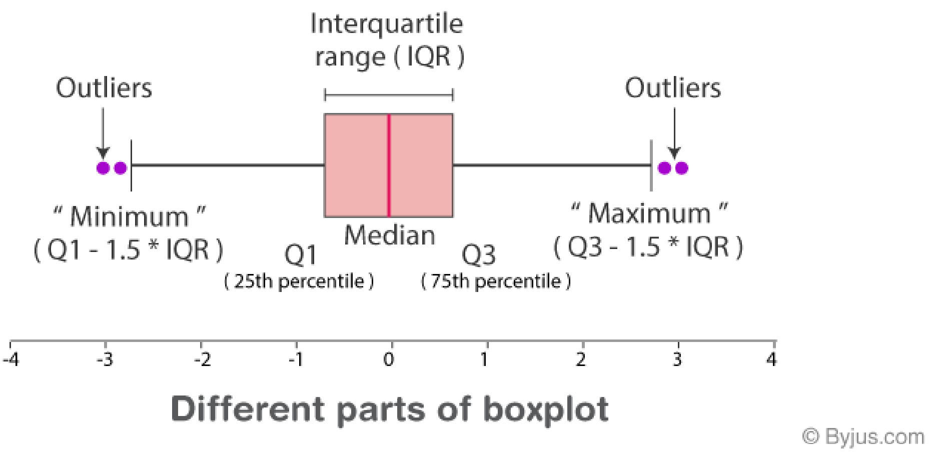


Figure 3.3 Different Parts of Box Plot

Minimum: The lowest value found in the dataset.

First Quartile (Q1): The median of the data set's lower half is the first quartile.

Median: Splits a dataset into two equal portions; is the dataset's middle value. The second quartile is regarded as the median.

Third Quartile (Q3): The median of the data's upper half is the third quartile.

Maximum: The highest value found in the dataset in question.

Interquartile Range (IQR): The interquartile range is the difference between the first and third quartiles. For example, IQR = Q3-Q1.

Outlier: Data is considered outliers if it lies on the extreme left or right side of the sorted data. Outliers typically deviate from the first and third quartiles by more than the designated amount. Outliers are greater than Q3+(1.5 . IQR) or less than Q1-(1.5 . IQR).

The next figure 3.4 taken from [44] shows the general temporal pattern of time series data generated by box plot.

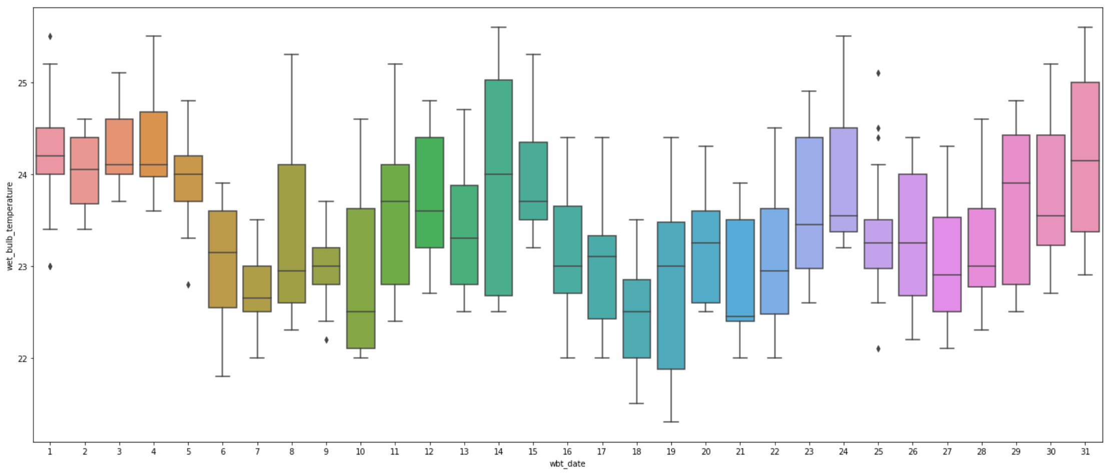


Figure 3.4 Temporal Pattern Generated by Box Plot

3.5.2 Smoothen the Dataset

The EMD has been frequently utilized to decompose a signal into the IMF and apply the Hilbert spectrum analysis to evaluate nonstationary and nonlinear time series data since [45] groundbreaking work on the Hilbert–Huang transform. Nonstationary and nonlinear time series characterize the empirical count of traffic volume over time. Decomposing the time series counts into basic components (IMF) might yield more accurate predictions due to the nonlinear and nonstationary nature of traffic flow dataset.

After applying the empirical mode decomposition algorithm shown in [Fig 3.5], any original traffic flow described by x(t), t = 1, 2,..., T, can be expressed as follows:

$$x(t) = n(t) + \sum d_i(t) \tag{1}$$

Here i = 1, 2.... n is the IMF for various decompositions, and r(t) is the residual following the derivation of n IMFs and $d_i(t)$. An IMF is defined in the EMD as a function that satisfies the subsequent criteria:

1. The number of extrema and zero-crossings in the entire data set must either be equal or deviate by no more than one.
2. The envelope defined by the local minima and the envelope formed by the local maxima have a mean value of zero at any given time.

Along the time axis, an IMF's frequency and amplitude might change. The distinct IMF components are extracted via an iterative method known as the "shifting process." The following is a summary of the specific steps involved in the moving process:

1. Determine each local extrema in the time series x(t), also known as local maxima and minima.
2. To create the upper envelope $e_{max}(t)$, interpolate each local maxima using a cubic spline line.
3. To generate the lower envelope $e_{min}(t)$, repeat the process for the local minima.
4. Using the upper (high) and lower envelopes, calculate the mean envelope m(t) as follows

$$(e_{max(t)} + e_{min(t)})/2 = m(t) \tag{2}$$

5. To create a new mean envelope, extract the original signal's mean envelope as follows:

$$m(t) = x(t) - \mathrm{h}(t) \qquad (3)$$

6. Determine if h(t) is an IMF: Set d(t) = h(t) if h(t) is an IMF. At the same time, replace x(t) with the residual r(t) = x(t) − d(t). If h(t) is not an IMF, replace x(t) with z(t). Repeat steps 2 through 5 until the iterative process reaches the next stopping condition.

3.6 Development of Deep Learning Models

Four deep learning models including ANN, LSTM, GRU, and a hybrid model Conv1d-LSTM were employed for the prediction task. These models were used to take advantage of their capacity to efficiently manage intricate data patterns. Deep learning methods are also suitable for situations with small datasets, which makes them a suitable option for this investigation.

Each model was subjected to kernel regularization (L2) to avoid overfitting. Keras Tuner [46] was used to optimize several best hyperparameters, including the number of layers, neurons, activation functions, L2 regularization value, learning rate, and batch size. To improve model performance and avoid overfitting, an early stopping condition was also applied during training. If the validation loss did not improve for 10 consecutive epochs, it monitored it and stopped training. The proposed deep learning models are described below.

3.6.1 Multilayer Perceptron Model

Dense layers that are fully connected make up MLP, which converts input data between dimensions. Because it has an input layer, one or more hidden layers, and an output layer, it is referred to as "multi-layer." An MLP is a potent tool for a variety of machine learning applications since it is designed to simulate intricate interactions between inputs and outputs.

Input Layer: An input feature is represented by each neuron (or node) in this layer. For example, the input layer has fifteen neurons as it has fifteen input features.

Hidden Layers: Three number of hidden layers, each with an arbitrary number of nodes, are present in an MLP. The data from the input layer is processed by these layers.

Output Layer: The final forecast or outcome is produced by the output layer. The output layer has one neuron since it makes single step forecasting.

The figure 3.5 depicts the structure of MLP with fifteen input time stamps, three hidden layers and one output layer

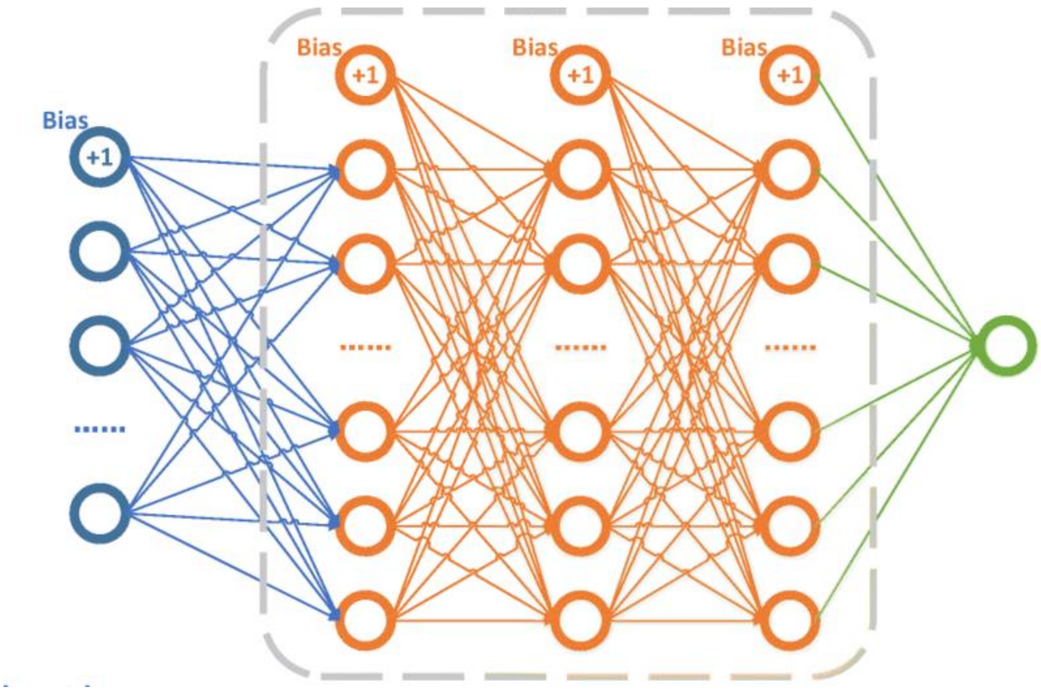


Figure 3.5 Structure of MLP

The working principle of MLP mechanisms such as forward propagation, loss function, backpropagation, and optimization are described below.

Step 1: Forward Propagation: Through any hidden layers, the data moves from the input layer to the output layer during forward propagation. The input is processed as follows by each neuron in the hidden layers:

Weighted Sum: The weighted sum of the inputs is calculated by the neuron.

$$z = \sum \omega_i x_i + b \tag{4}$$

Here,

1. This is the input feature: x_i
2. The matching weight is denoted by w_i
3. The bias phrase is b

Activation Function: The weighted sum z is passed through an activation function to introduce non-linearity. Common activation functions include

Sigmoid : $\sigma_z = \frac{1}{1+e^{-z}}$ (5)

Rectified Linear Unit (ReLU): $f[z] = \dot{m}ax(0, z)$ (6)

Hyperbolic Tangent: $\tan h\ (z) = \frac{2}{1+e^{-2z}} - 1$ (7)

Step 2 Loss Function: Once the network generates an output, the next step is to calculate the loss using a loss function. In supervised learning, this compares the predicted output to the actual label. For this time series dataset, mean squared error (mse) has been used.

$$MSE = \frac{1}{N} \sum_{i=1}^{\mathrm{N}} (y_i - \hat{y})^2 \tag{8}$$

Here:

1. y_i is the actual label.
2. $\tilde{y}$ is the predicted label.
3. N is the number of samples.

Step 3 Backpropagation: The goal of training an MLP is to minimize the loss function by adjusting the network’s weights and biases. This is achieved through backpropagation

1. **Gradient Calculation**: The gradients of the loss function with respect to each weight and bias are calculated using the chain rule of calculus.

2. **Error Propagation**: The error is propagated back through the network, layer by layer.
3. **Gradient Descent:** The network updates the weights and biases by moving in the opposite direction of the gradient to reduce the loss:

$$\omega = w - n\frac{\partial L}{\partial w} \qquad (9)$$

Here,

1. W is the weight
2. n is the learning rate
3. $\frac{\partial L}{\partial w}$ is the gradient of the loss function

3.6.2 Long Short-Term Memory (LSTM)

LSTM is a unique type of recurrent neural network that uses three gates—input, forget, and output gates—and cell or memory states to learn long-term dependencies. Hochreiter & Schmidhuber first presented the LSTM in 1997 [47]. LSTM is currently frequently utilized in time series data processing and performs incredibly well on a wide range of challenging tasks. The long-term dependency and vanishing gradient descent issues that recurrent neural networks cannot handle are specifically avoided by LSTMs. The screen shoot taken from [9] shows the basic internal structure of the LSTM cell in figure 3.6.

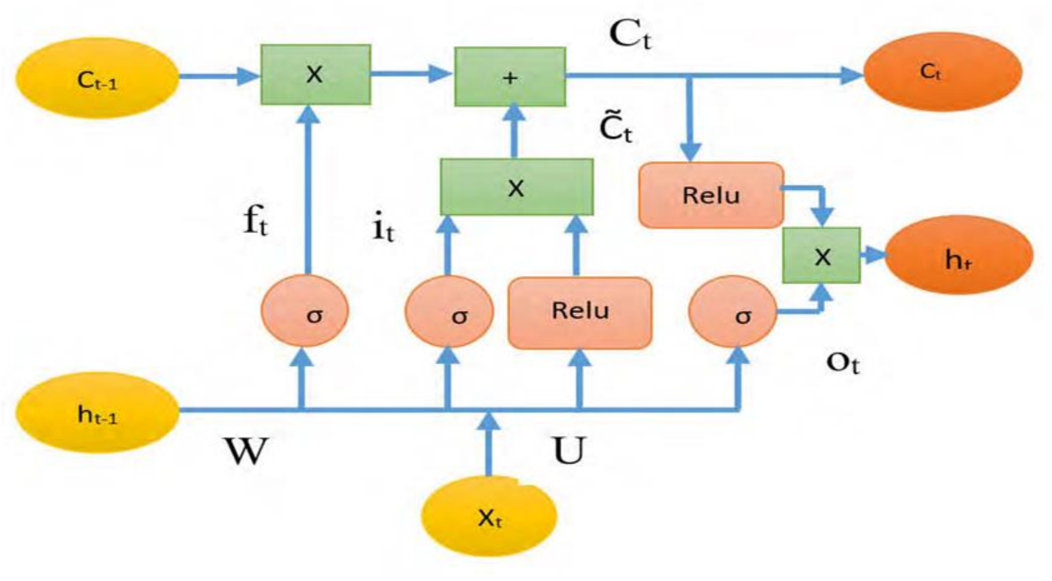


Figure 3.6 LSTM Cell Structure

Cell states are a method through which LSTM information moves. Information can be selectively remembered or forgotten by this cell. Three main processes are used to manipulate the memory blocks or cells that are in charge of storing information: known as gates. Below is a quick description of these three gates' functions.

Forget Gate: A forget gate is responsible for discarding or keeping information from or to the cell state. If the previous cell information is no longer required or has less important for the LSTM cell, then it will be discarded. The forget gate is essential for improving the performance of the LSTM network. The output formula of forget gate is

$f_t = \sigma(U_f . x_t + W_f . h_{t-1} + b_f)$ (10)

Here,

1. W_f, U_f, and b_f are the corresponding forget gate weights and biases
2. f_t is the forget gate output
3. x_t and h_{t-1} are the two inputs of forget gate. Where x_t is the input at that specific time step and h_{t-1} is the hidden state from the previous cell or the output of the previous cell.

4. σ is the sigmoid function. It is either 0 or 1 . The sigmoid function is essentially in charge of selecting which values should be retained and which should be eliminated. The forget gate wishes to completely erase the data if the gate output is 0 for a specific value in the cell state. In a similar vein, the gate wishes to retain or remember all of the data if the gate output is 1.

Input Gate: The input gate is responsible for the addition of input information to the cell state. This addition of information is basically done by the three-step process as shown in below.

a) Managing which values need to be added to the cell state by sigmoid function from ht-1 and xt input sequences.

b) Generating a vector having all potential values that can be added to the cell state by using the rectified linear unit (relu) function f(x) = max (0,x).

c) Multiplying the input gate output value with the output of the ReLU function and then adding this valuable information to the cell state via an addition operation.

The input gate calculation formulas are:

Input gate output is $i_t = \sigma(Ui \,.\, x_t + Wi \,.\, h_{t-1} + bi\ )$ (11)

Cell input state is $C_t = \text{relu}(U_c \,.\, x_t + W_c \,.\, h_{t-1} + b_c)$ (12)

Final information will be added to cell state like $i_t * C_t$ (13)

Here it input gate output, Ct candidate state, (Wi,, Ui , bi), (W_c, U_c , b_c) are the weight and bias of the input gate and candidate state.

Cell or Memory: The cell or memory state is stored important information by using forget gate and input gates outputs. The cell or memory Ct information is the sum of the previous cell state and current cell input information by deciding these gates. It will be forwarded to the next cell as input. The cell state or memory basic calculation function is shown below.

Calculate cell state or memory state is $C_t = f_t * C_{t-1} + i_t * C_t$ (14)

Output Gate: The output gate is responsible for how many portions of the memory state will be represented as prediction output. The output gate activities are done in three steps:

a) Making a vector after applying relu function to the cell state

b) Generating output from h_{t-1} and x_t, input sequences as a result of output gate by using a sigmoid function.

c) Finally multiply the value of step 1 and step 2, and send it as an output of the hidden cell state for the next cell.

The output gate result is $O_t = \sigma(U_o . x_t + W_o . h_{t-1}+ b_o)$ (15)

and final prediction output is $h_t = O_t * \text{relu}(C_t)$ (16)

Here,

1. Ot is the output gate result,
2. h_t is the current cell predicted output,
3. (Wo , Uo , bo) is the output weights and bias,
4. C_t cell or memory state.

3.6.3 Gated Recurrent Unit (GRU)

GRU was introduced by Cho et al. in 2014. The core idea behind GRUs is to use gating mechanisms to selectively update the hidden state at each time step allowing them to remember important information while discarding irrelevant details. GRUs aim to simplify the LSTM architecture by merging some of its components and focusing on just two main gates: the update gate and the reset gate. GRUs Gated Recurrent Units are a streamlined version of LSTMs. They simplify things by combining the input and forget gates into a single update gate, and they also have a reset gate. This makes GRUs less computationally intensive and faster to train than LSTMs, while still being able to handle long-term dependencies effectively. The figure 3.7 taken from [48] represents the cell structure of GRU

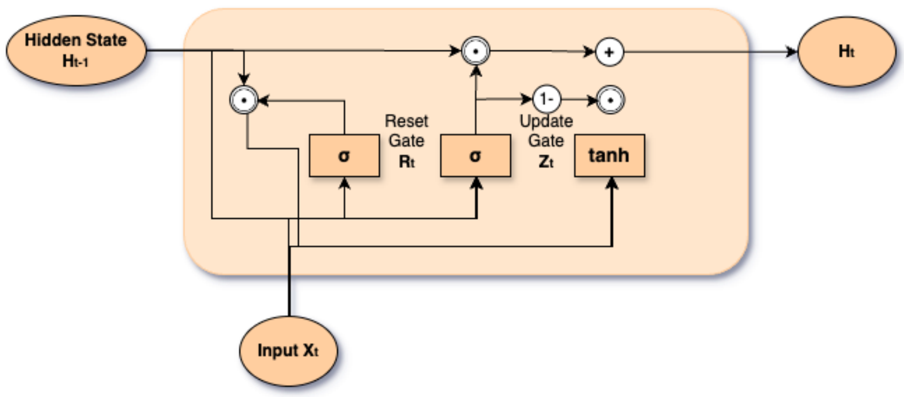


Figure 3.7 GRU Cell Structure

A GRU cell can be thought of as a small control unit in a factory that decides how much of the old information to keep and how much new information to add. It does this through two main components: the update gate and the reset gate.

Update Gate: This gate acts like a smart filter. It determines how much of the past information should be carried forward. If the update gate decides a piece of information is important, it will keep it; otherwise, it will discard it. The update gate _zt is calculated using the current input _xt and the previous hidden state _ht−1. The formula for the update gate is:

$$z_{(t)} = \sigma\{w_z[h_{t-1}, x_t]\} \tag{17}$$

Here:

1. z_t is the update gate at time step t.
2. σ is the sigmoid activation function, which squashes the output to be between 0 and 1.
3. W_z is the weight matrix for the update gate.

4. $h_{t\text{-}1}$ is the hidden state from the previous time step.

5. x_t is the current input.

6. [h_t, x_t] represents the concatenation of the previous hidden state and the current input.

Reset Gate: This gate decides how much of the past information to forget. When the reset gate is activated, it allows the cell to ignore parts of the past data, which helps in focusing on new, relevant information. The reset gate _rt is calculated using the current input _xt and the previous hidden state _ht−1. The formula for the reset gate is:

$$r_t = \sigma\{w_r[h_{t-1}, x_t]\} \qquad (18)$$

Here:

1. r_t is the reset gate at time step t.

2. σ is the sigmoid activation function, which squashes the output between 0 and 1.

3. W_r is the weight matrix for the reset gate.

4. h_t is the hidden state from the previous time step.

5. x_t is the current input.

6. [h_t , x_t] is the concatenation of the previous hidden state and the current input, as described in the previous section.

Candidate State: The candidate activation in a GRU represents the new information that could potentially update the hidden state. This step integrates the reset gate’s decision on how much past information to consider with the current input.

$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t \qquad (19)$$

Here:

1. h_t is the candidate activation at time step t.

2. tanh is the hyperbolic tangent activation function, which outputs values between -1 and 1.

3. *W* is the weight matrix for the candidate activation.

4. r_t is the reset gate value at time step *t* (*refer to section 3.2)*.

5. h_{t-1} is the hidden state from the previous time step.

6. x_t is the current input.

7. $[r_t * h_t , x_t]$ represents the concatenation of the reset-modified previous hidden state and the current input.

3.6.4 Conv1D LSTM

The Conv1D-LSTM model is a hybrid deep learning architecture that combines one-dimensional convolutional neural networks (Conv1D) with Long Short-Term Memory (LSTM) networks to effectively capture both spatial and temporal dependencies in sequential data. This architecture is especially helpful for time-series prediction problems since the LSTM layer models long-term dependencies, while the Conv1D layer is in charge of extracting local data and patterns.

The Conv1D operation applies a convolutional filter over the input sequence, extracting spatial features. Mathematically, the output of a Conv1D layer can be expressed as:

$$y_i = f \sum_{k=0}^{k-1} \omega_k x_{i+k} + b \tag{20}$$

Here:

1. y_i is the output of the convolution at position iii,

2. W_k represents the filter weights,

3. x_{i+k} is the input sequence,

4. b is the bias term,

5. f is the activation function.

The extracted features from Conv1D are then passed to an LSTM layer, which processes the sequential data using the equations described in [section 3.6.2].

3.7 Evaluation Metrics

Two popular assessment metrics, Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE), were used to evaluate the deep learning models' performance. By calculating the gap between the actual and anticipated values, these metrics offer information about the precision and dependability of the forecasts.

3.7.1 Root Mean Squared Error (RMSE)

RMSE is a standard metric used to measure the average magnitude of prediction errors. It penalizes larger errors more than smaller ones due to squaring the differences, making it particularly useful when large deviations need to be minimized. RMSE is mathematically defined as:

$$RMSE = \sqrt{\frac{1}{\eta}\sum_{i=1}^{n}\{y_i - \tilde{y}_i\}^2} \tag{21}$$

Here:

1. n is the total number of observations,
2. y_i represents the actual value,
3. $\tilde{y}_i$ represents the predicted value.

A lower RMSE value indicates a better model performance, as it signifies a smaller average error.

3.7.2 Mean Absolute Percentage Error (MAPE)

MAPE measures the percentage error between actual and predicted values, providing an intuitive way to interpret model accuracy. It is particularly useful for understanding the relative size of errors with respect to the actual values. MAPE is defined as:

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (22)$$

Here:

1. n is the total number of observations,
2. y_i is the actual value,
3. $\tilde{y}_i$ is the predicted value.

A lower MAPE value indicates a more accurate model. However, one limitation of MAPE is that it can be unreliable when actual values are close to zero, as the percentage error becomes disproportionately large.

3.8 Implementation Environment

The entire model implementation and trial process was carried out on Kaggle, a cloud-based online platform that facilitates collaboration for challenges related to data science and machine learning. By providing access to strong GPU and TPU resources, Kaggle makes it possible to train and assess deep learning models quickly.

The Python TensorFlow Keras library was used to create the model. The open-source deep learning framework TensorFlow offers a scalable and adaptable environment for neural network construction and training. TensorFlow's Keras high-level API makes model building easier by providing a user-friendly interface for creating, assembling, and refining deep learning architectures.

Rapid model prototyping, smooth deep learning workflow execution, and effective handling of big datasets were all made possible by the implementation process, which made use of Kaggle's computational resources and the TensorFlow-Keras ecosystem. The entire coding is available in Appendix C

3.9 Summary

This chapter outlined the methodology used to enhance traffic flow forecasting with an Empirical Mode Decomposition (EMD)-preprocessed deep learning model. The process began with data collection from a 4-lane suburban highway, followed by data analysis and smoothing using EMD to reduce noise. Model development involved training deep learning models on the preprocessed data, with fine-tuning to optimize performance. Predictions were evaluated using metrics like MAE and RMSE.

In the next chapter, the results of the model's performance are presented and analyzed, comparing predicted traffic flow with actual data and discussing the implications of the findings.

CHAPTER 4
RESULTS AND DISCUSSIONS

4.1 Introduction

This chapter presents the findings of the research framework and discusses the possible reasons behind the results obtained. The purpose of this chapter is to analyze the performance of the proposed methodology and provide insights into how the application of Empirical Mode Decomposition (EMD) and deep learning models impacts traffic flow forecasting.

The chapter is organized into several key sections: Analysis of Data Pattern & Noise, where the characteristics of the raw data are examined to understand inherent patterns and noise; Data Denoising/Smoothing, which discusses the effectiveness of the EMD technique in reducing noise; and Deep Learning Model Development, which details the process of developing and fine-tuning the deep learning models used for forecasting. Finally, a Comparison and Evaluation section is included, where the performance of the models is evaluated and compared using relevant metrics, providing a clear assessment of the model's accuracy and overall effectiveness.

By thoroughly discussing the results in these areas, this chapter aims to highlight the strengths and limitations of the proposed methodology and provide a deeper understanding of how the models performed in predicting traffic flow.

4.2 Data Pattern and Noise Analysis

The hourly distribution of traffic counts was examined using a box plot analysis in order to comprehend the temporal patterns and variability in the traffic flow data. In Dhaka, Bangladesh, the hourly variation of traffic counts (5-minute intervals) from 6 AM to 3 PM is shown in figure 4.1.

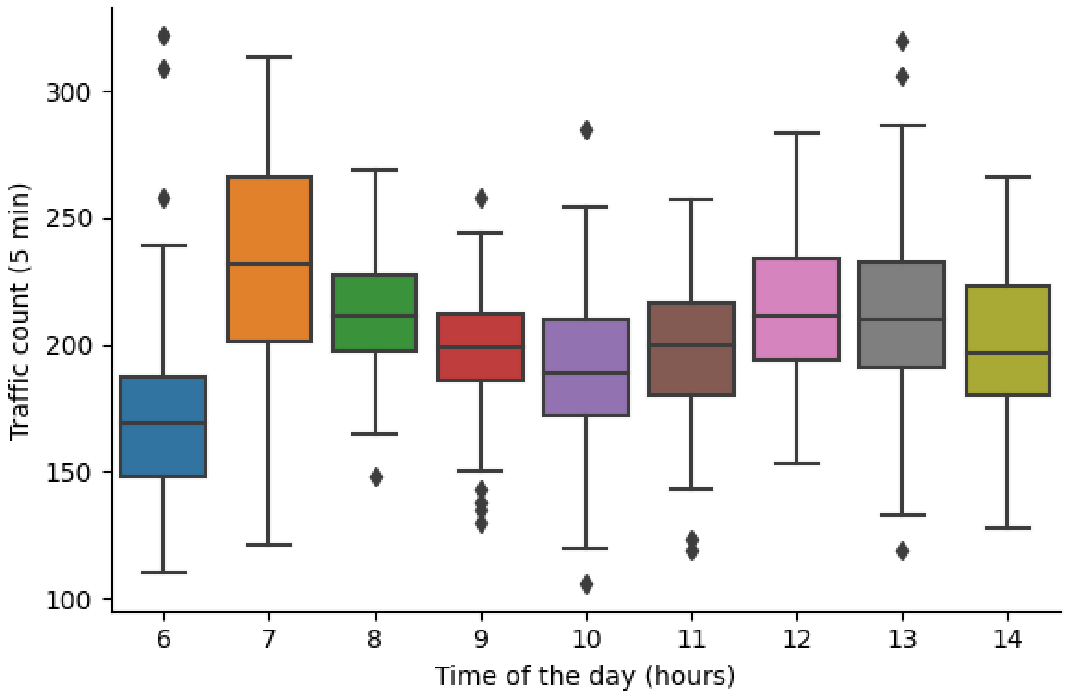


Figure 4.1 Hourly Box Plot of Traffic Count

The box plots show clear hourly changes in traffic volume, as seen in [Fig 4.1] As the morning rush hour begins, the median traffic count rises sharply between 6 and 7 AM. After then, with only minor variations, the median stays comparatively high and steady until 1 PM. Interestingly, there is more variability in traffic flow during peak hours (7 AM to 1 PM), as evidenced by the wider interquartile range (IQR). The box plots show exhibit the occurrence of outliers, particularly around 6 AM, 7 AM, 10 AM, and 1 PM. These outliers point to possible irregularities or variations in traffic flow that go beyond the usual range, which could be brought on by accidents, abrupt shifts in traffic patterns, road closures and emergencies.

The observed hourly pattern, which shows distinct peaks and troughs reflecting daily activity patterns, emphasizes the non-stationary character of traffic data in Dhaka. Accurate forecasting is made more difficult by the greater IQR during peak hours, which indicates increased congestion and fluctuation. Outliers are a sign of noise or anomalous occurrences that may compromise the accuracy of forecasts. The necessity for reliable forecasting models that can account for the intrinsic unpredictability and noise in the traffic flow data as well as the underlying temporal trends is highlighted by this investigation.

4.3 EMD-based Denoising and Smoothing the Dataset

Empirical Mode Decomposition (EMD) was used to examine and reduce the random fluctuations found in the traffic flow data. The original time series is broken down by EMD into a residual component and a collection of intrinsic mode functions (IMFs). Every IMF captures multiple frequency bands, from low-frequency trends to high-frequency noise, and indicates a particular oscillatory mode buried in the data.

Figure 4.2 illustrates the IMFs and residual obtained from decomposing the traffic flow data using EMD. As observed, the decomposition yielded seven IMFs and a residual. IMF 1 and IMF 2 exhibit high-frequency fluctuations, indicating the presence of noise or short-term variations in the traffic flow. IMF 3 through IMF 6 show progressively lower frequencies, suggesting the capture of more structured oscillatory patterns. IMF 7 represents a low-frequency trend, and the residual captures the overall long-term trend of the data, showing a gradual decline over the sampling period.

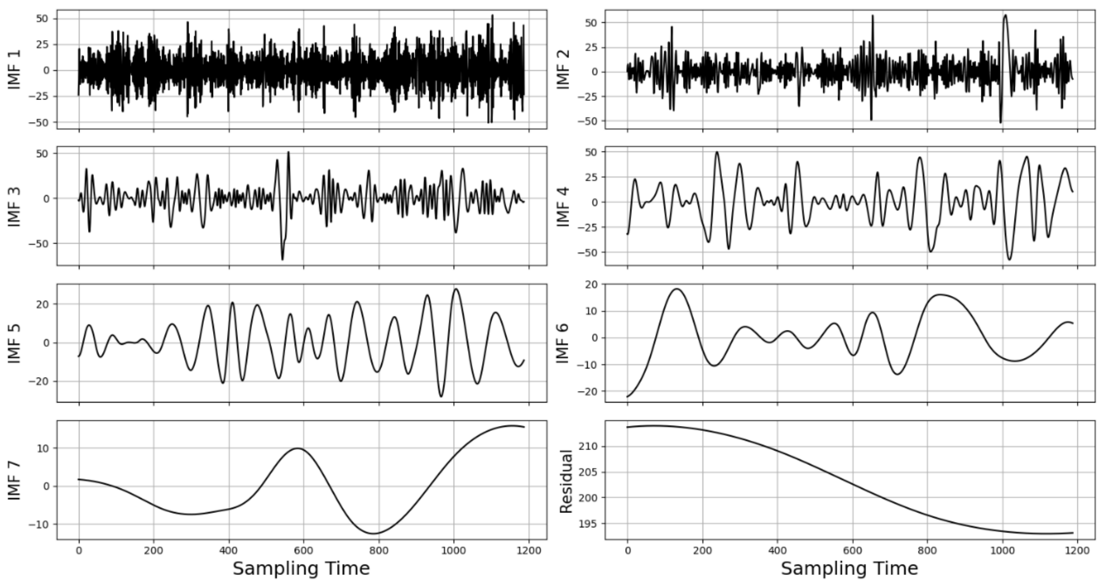


Figure 4.2 IMF Components and Residual

The Hurst exponent was computed to evaluate each IMF's long-term memory and self-similarity properties. A statistical metric called the Hurst exponent (H) measures how persistent a time series is and how previous patterns affect subsequent behavior.

Table 4.1 presents the Hurst exponent values for each IMF. As shown, IMF 1 exhibits a Hurst exponent of 0.584, indicating a slightly persistent behavior. IMF 2 shows a stronger persistence with a Hurst exponent of 0.815. Notably, IMFs 3 through 7, as well as the residual, have Hurst exponents greater than 1.0, suggesting a strong trend-reinforcing behavior and long-term dependencies.

Table 4-1. The Hurst Exponent of each IMFs

IMF	Hurst Exponent
1	0.584
2	0.815
3	1.045
4	1.085
5	1.083
6	1.085
7	1.088
Residual	1.089

The original traffic data was not reconstructed using IMF 1, which had a Hurst exponent near 0.5 and looked like high-frequency noise based on the Hurst exponent study. The residual and all of the remaining IMFs (2–7) were added together to recreate the original dataset. In order to increase the signal-to-noise ratio of the data and improve forecasting accuracy, this selective reconstruction sought to reduce noise and keep only the IMFs with significant long-term patterns.

The impact of this reconstruction procedure on the traffic flow data over nine days in a row (September 19–27, 2024) is shown in figure 4.3. As can be seen, the profile of the reconstructed data (orange) is smoother than that of the original data (blue). Effective noise suppression is demonstrated by the reconstructed data's notable reduction in the original data's high-frequency fluctuations and sharp peaks. After removing IMF 1, the information was smoothed to better depict the underlying traffic flow patterns, which makes it more appropriate for precise forecasting. Particularly in a dynamic sub urban setting where traffic patterns are naturally erratic and influenced by a number of outside influences, this noise reduction is essential.

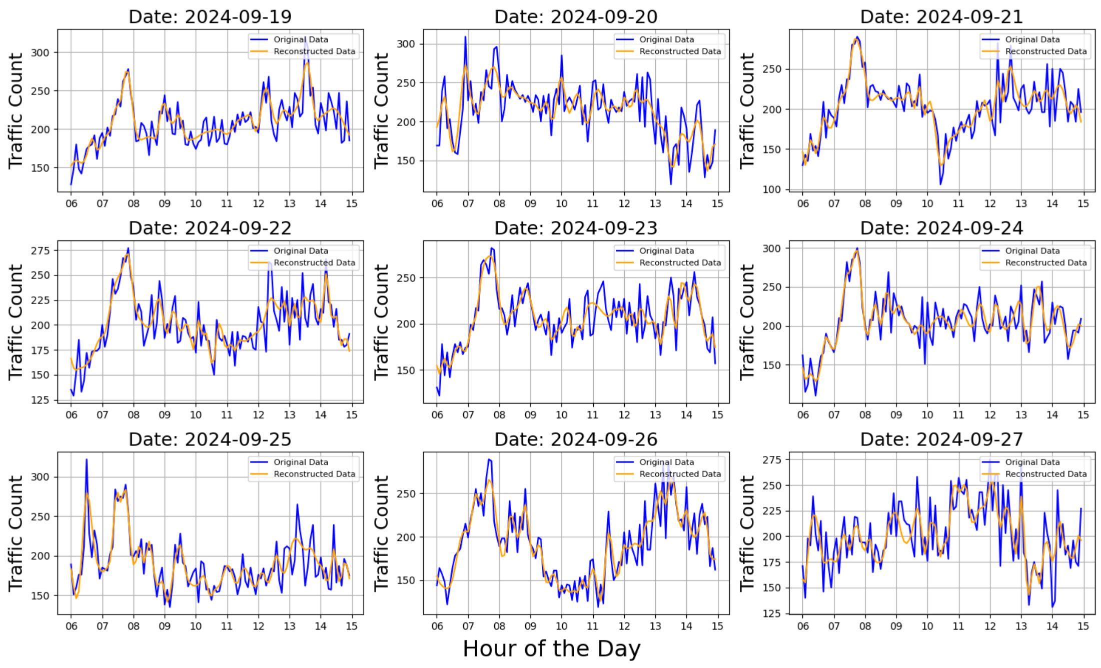


Figure 4.3 Original and reconstructed dataset for individual day

4.4 Model Development

Developing effective deep learning models involves careful selection of hyperparameters, which significantly influence the model's performance and generalization ability. To determine the optimal hyperparameters for each deep learning model, the Keras Tuner library was employed. Keras Tuner automates the hyperparameter tuning process, efficiently searching through a defined hyperparameter space to identify the combination that yields the best performance on a given dataset.

Table 4-2. The Final Hyperparameters of the Corresponding Models

Hyperparameters/ Models	Optimizer	Learning Rate	No of layers	Neurons	Activation Function	L2 Regularization	Batch Size	Sequence Length	Epoch
ANN	adam	.0001	3	92, 144, 16	tanh, relu, relu	002, .001, .008	16	15	50
LSTM	adam	.0010	2	64, 32	tanh	.001, .001	16	15	50
GRU	adam	.0008	1	176	tanh	.00006	16	15	40
Conv1d-LSTM	adam	.0010	2	64, 32	tanh	.001, .001	16	15	100
EMD-LSTM	adam	.0010	2	64, 32	tanh	.001, .001	16	15	50

The Keras Tuner-identified optimal hyperparameters for each deep learning model are shown in Table 4.2. The number of layers, units per layer, activation functions, dropout rates, and learning rates are some examples of these hyperparameters. The best architecture for each model is indicated by the values in Table 4.2, which show the configuration that produced the highest forecasting accuracy on the validation set. The final assessment and comparison are then conducted using these improved models, guaranteeing that each model is performing at its best for precise traffic flow forecasts.

4.5 Comparison and Evaluation

The effectiveness of the ANN, LSTM, GRU, Conv1D-LSTM, and EMD-LSTM models in predicting traffic flow was thoroughly assessed following their development and optimization. Two important measures, MAPE and RMSE, were used to evaluate the models' accuracy in predicting traffic counts for the target period.

The RMSE and MAPE values for every deep learning model are shown in Table 4.3. The EMD-LSTM model obtained the lowest MAPE of 8.55% and the lowest RMSE of 23.18, as indicated in the table. On the other hand, the GRU model had the greatest MAPE of 9.51% and the highest RMSE of 29.42. With RMSE values ranging from 28.32 to 28.70 and MAPE values ranging from 9.31% to 9.80%, the ANN, LSTM, and Conv1D-LSTM models demonstrated intermediate performance.

Figure 4.4 presents a visual comparison between the models' predictions and the actual traffic flow in order to further examine and contrast the models' performance. Over the course of a day (7 AM to 3 PM), this graphic displays the actual traffic counts against the figures predicted by each model.

Table 4-3. Performance Evaluation

Model	**RMSE**	**MAPE (%)**
ANN	28.42	9.80
LSTM	28.32	9.31
GRU	29.42	9.51
Conv1D LSTM	28.70	9.62
EMD LSTM	**23.18**	**8.55**

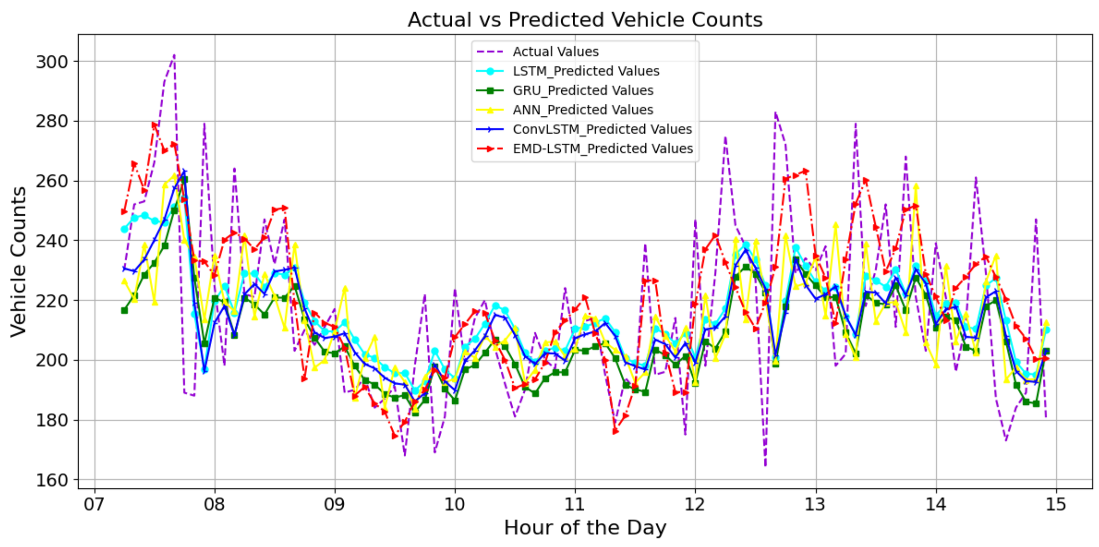


Figure 4.4 Traffic flow forecasting of various models

Figure 4.4 illustrates how well the EMD-LSTM model (red line with circle markers) captures the temporal dynamics of the traffic data by closely tracking the real traffic flow (purple dashed line) throughout the day. Notably, the EMD-LSTM model well handles the non-stationary nature of traffic data by correctly predicting the peaks and troughs in traffic flow. Although it performs well as well, the LSTM model (blue line with square markers) deviates significantly more from the true val ues than the EMD-LSTM model. The RMSE and MAPE figures demonstrate the reduced accuracy of the GRU model, which displays the most significant deviations (green line with square markers). With forecasts that are more accurate than the EMD-LSTM but closer to the actual values than the GRU, the ANN (yellow line with diamond marks) and Conv1D-LSTM (blue line without markers) models perform in the middle.

The merits and shortcomings of each deep learning model in traffic flow predictions are well understood thanks to this complete examination, which combines quantitative measures (RMSE and MAPE) with visual analysis (Figure 4.5). In terms of both RMSE and MAPE, the results unequivocally show that the EMD-LSTM model, which used Empirical Mode Decomposition for noise reduction, performed noticeably better than the other models. This implies that in the intricate traffic environment of Dhaka, Bangladesh, the preprocessing step of EMD successfully improved the model's capacity to identify the underlying patterns in the traffic flow data, resulting in more precise forecasts.

4.6 Summary

This chapter presented the findings of the research framework and discussed the possible reasons behind the results. The analysis of data patterns and noise highlighted the presence of fluctuations and irregularities in the raw traffic flow data, emphasizing the need for a robust preprocessing technique. The data denoising and smoothing process using Empirical Mode Decomposition (EMD) effectively reduced noise, extracting meaningful traffic trends for improved forecasting accuracy.

The deep learning model development section detailed the training and fine-tuning process of the models, showcasing their ability to learn from preprocessed data. In the comparison and evaluation section, the performance of the models was assessed using evaluation metrics, demonstrating the effectiveness of the proposed approach in enhancing traffic flow forecasting. The findings indicate that EMD preprocessing improves prediction accuracy by addressing random fluctuations and noise in traffic data.

The next chapter will summarize the key contributions of this research, discuss its limitations, and suggest potential directions for future studies.

CHAPTER 5
CONCLUSION AND RECOMMENDATION

5.1 Introduction

This chapter presents the key findings, limitations, recommendations, and potential future research directions based on the results of this study. The primary goal of this research was to enhance short-term traffic flow forecasting using Empirical Mode Decomposition (EMD)-preprocessed deep learning models. The study demonstrated that EMD-based LSTM significantly improves forecasting accuracy by addressing random fluctuations and noise in traffic data. The findings provide valuable insights for transportation planning, congestion management, and intelligent traffic systems.

5.2 Key Findings of the Study

The study compared multiple deep learning models, including ANN, LSTM, GRU, and Conv1D-LSTM, with and without EMD preprocessing. The following key findings were observed:

1. EMD-based preprocessing significantly improved model performance by reducing the impact of noise and fluctuations in traffic data.
2. The EMD-LSTM model achieved the best prediction accuracy, with the lowest Root Mean Square Error (RMSE) of 23.18 and Mean Absolute Percentage Error (MAPE) of 8.55%, outperforming other baseline models.
3. GRU and Conv1D-LSTM exhibited slightly higher error rates, indicating their relative inefficiency in handling the non-stationary nature of traffic flow data compared to EMD-LSTM.

4. Despite the Limited dataset, this proposed method is suitable for traffic flow forecasting.

5.3 Limitations of the Study

The study could have been further strengthened with a broader dataset and additional modeling techniques. However, due to limited time, academic constraints, and financial limitations, certain aspects could not be explored in depth. The key limitations include:

1. The dataset used was limited to 11 consecutive days, covering only the period between 6 AM and 3 PM each day. As a result, the findings may not fully capture long-term traffic trends or night-time traffic variations.
2. The study was conducted on a single 4-lane suburban highway, limiting the generalizability of the results to other road types, such as urban roads or highways with different geometric configurations.
3. The deep learning models require significant computational resources, including auxiliary traffic data like velocity, density, headways, and other variables, including roadway capacity, occupancy rate, and weather conditions

5.4 Recommendation

Based on the key findings and identified limitations, the following recommendations are proposed to enhance the accuracy, applicability, and real-world implementation of traffic flow forecasting models:

1. Expand data collection to cover a wider period (24-hour data) and multiple roadway types to improve model generalizability.

2. Enhance preprocessing techniques by comparing EMD with other denoising methods like Wavelet Transform or Variational Mode Decomposition (VMD) for further performance improvements.
3. Optimize model efficiency by exploring lightweight deep learning architectures for real-time traffic forecasting applications.
4. Integrate real-time data sources, such as IoT-based traffic sensors and GPS data, to enhance real-world applicability.

5.5 Directions for Future Work

To further advance traffic flow forecasting research, the following areas are suggested for future exploration:

1. Hybrid forecasting models: Combining LSTM with attention mechanisms, transformer models, or ensemble learning could further improve accuracy.
2. Multi-source data integration: Incorporating weather conditions, road incidents, and real-time GPS data could enhance prediction reliability.
3. Deployment in Intelligent Transportation Systems (ITS): Implementing the proposed model in real-time adaptive traffic signal control and dynamic route planning systems.
4. Exploration of alternative deep learning techniques: Investigating the potential of Graph Neural Networks (GNNs) and Reinforcement Learning for traffic forecasting.

5.6 Conclusion

This study demonstrated the effectiveness of using Empirical Mode Decomposition (EMD) with deep learning models for short-term traffic flow forecasting. The EMD-LSTM model outperformed other baseline models, proving its ability to capture the stochastic nature of traffic flow data. The findings highlight the importance of data preprocessing in improving prediction accuracy and resilience.

While the study had limitations in dataset size and scope, the results indicate strong potential for scaling the approach to real-world intelligent transportation systems. Future work should focus on expanding datasets, integrating real-time traffic data, and optimizing computational efficiency for real-time deployment. The insights gained from this research contribute to the development of more reliable, adaptive, and intelligent traffic management systems, ultimately aiding in congestion reduction and urban mobility planning.

REFERENCES

[1] Z. Abbas, A. Al-Shishtawy, S. Girdzijauskas, and V. Vlassov, “Short-Term Traffic Prediction Using Long Short-Term Memory Neural Networks,” in *Proceedings - 2018 IEEE International Congress on Big Data, BigData Congress 2018 - Part of the 2018 IEEE World Congress on Services*, Institute of Electrical and Electronics Engineers Inc., Sep. 2018, pp. 57–65. doi: 10.1109/BigDataCongress.2018.00015.

[2] C. Ma, G. Dai, and J. Zhou, “Short-Term Traffic Flow Prediction for Urban Road Sections Based on Time Series Analysis and LSTM_BILSTM Method,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5615–5624, Jun. 2022, doi: 10.1109/TITS.2021.3055258.

[3] C. Li and P. Xu, “Application on traffic flow prediction of machine learning in intelligent transportation,” *Neural Comput Appl*, vol. 33, no. 2, pp. 613–624, Jan. 2021, doi: 10.1007/s00521-020-05002-6.

[4] W. Li, X. Wang, Y. Zhang, and Q. Wu, “Traffic flow prediction over muti-sensor data correlation with graph convolution network,” *Neurocomputing*, vol. 427, pp. 50–63, Feb. 2021, doi: 10.1016/j.neucom.2020.11.032.

[5] X. Chen *et al.*, “Traffic flow prediction by an ensemble framework with data denoising and deep learning model,” *Physica A: Statistical Mechanics and its Applications*, vol. 565, Mar. 2021, doi: 10.1016/j.physa.2020.125574.

[6] A. Essien, I. Petrounias, P. Sampaio, and S. Sampaio, “A deep-learning model for urban traffic flow prediction with traffic events mined from twitter,” *World Wide Web*, vol. 24, no. 4, pp. 1345–1368, Jul. 2021, doi: 10.1007/s11280-020-00800-3.

[7] Y. Li, S. Chai, Z. Ma, and G. Wang, “A Hybrid Deep Learning Framework for Long-Term Traffic Flow Prediction,” *IEEE Access*, vol. 9, pp. 11264–11271, 2021, doi: 10.1109/ACCESS.2021.3050836.

[8] K. Kumar, M. Parida, and V. K. Katiyar, “Short Term Traffic Flow Prediction for a Non-Urban Highway Using Artificial Neural Network,” *Procedia Soc Behav Sci*, vol. 104, pp. 755–764, Dec. 2013, doi: 10.1016/j.sbspro.2013.11.170.

[9] M. Abul and K. Azad, “A SMART TRAVEL TIME PREDICTION MODEL FOR URBAN TRAFFIC USING LONG SHORT-TERM MEMORY NETWORK,” 2021.

[10] B. Huang, H. Dou, Y. Luo, J. Li, J. Wang, and T. Zhou, “Adaptive Spatiotemporal Transformer Graph Network for Traffic Flow Forecasting by IoT Loop Detectors,” *IEEE Internet Things J*, vol. 10, no. 2, pp. 1642–1653, Jan. 2023, doi: 10.1109/JIOT.2022.3209523.

[11] S. Zhang, Y. Guo, P. Zhao, C. Zheng, and X. Chen, “A Graph-Based Temporal Attention Framework for Multi-Sensor Traffic Flow Forecasting,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7743–7758, Jul. 2022, doi: 10.1109/TITS.2021.3072118.

[12] A. Fedorov, K. Nikolskaia, S. Ivanov, V. Shepelev, and A. Minbaleev, “Traffic flow estimation with data from a video surveillance camera,” *J Big Data*, vol. 6, no. 1, pp. 1–15, Dec. 2019, doi: 10.1186/S40537-019-0234-Z/TABLES/3.

[13] E. Necula, “Dynamic Traffic Flow Prediction Based on GPS Data,” *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, vol. 2014-December, pp. 922–929, Dec. 2014, doi: 10.1109/ICTAI.2014.140.

[14] S. Sun, J. Chen, and J. Sun, “Traffic congestion prediction based on GPS trajectory data,” *Int J Distrib Sens Netw*, vol. 15, no. 5, May 2019, doi: 10.1177/1550147719847440/ASSET/IMAGES/LARGE/10.1177_1550147719847440-FIG17.JPEG.

[15] G. Valenti, “Traffic Estimation And Prediction Based On Real Time Floating Car Data CIPRNet View project Traffic forecasting and behavioural model View

project,” pp. 197–203, 2008, Accessed: Dec. 25, 2024. [Online]. Available: https://www.researchgate.net/publication/224364822

[16] C. De Fabritiis, R. Ragona, and G. Valenti, “Traffic estimation and prediction based on real time floating car data,” *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 197–203, 2008, doi: 10.1109/ITSC.2008.4732534.

[17] D. Kim and O. Jeong, “Cooperative Traffic Signal Control with Traffic Flow Prediction in Multi-Intersection,” *Sensors 2020, Vol. 20, Page 137*, vol. 20, no. 1, p. 137, Dec. 2019, doi: 10.3390/S20010137.

[18] Z. Wang, P. Thulasiraman, and R. Thulasiram, “A Dynamic Traffic Awareness System for Urban Driving,” in *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, IEEE, Jul. 2019, pp. 945–952. doi: 10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00167.

[19] J. J. Vázquez, J. Arjona, M. Linares, and J. Casanovas-Garcia, “A Comparison of Deep Learning Methods for Urban Traffic Forecasting using Floating Car Data,” *Transportation Research Procedia*, vol. 47, pp. 195–202, Jan. 2020, doi: 10.1016/J.TRPRO.2020.03.079.

[20] J. Culita, S. I. Caramihai, I. Dumitrache, M. A. Moisescu, and I. S. Sacala, “An Hybrid Approach for Urban Traffic Prediction and Control in Smart Cities,” *Sensors 2020, Vol. 20, Page 7209*, vol. 20, no. 24, p. 7209, Dec. 2020, doi: 10.3390/S20247209.

[21] W. Fitters, A. Cuzzocrea, and M. Hassani, “Enhancing LSTM prediction of vehicle traffic flow data via outlier correlations,” *Proceedings - 2021 IEEE 45th Annual Computers, Software, and Applications Conference, COMPSAC 2021*, pp. 210–217, Jul. 2021, doi: 10.1109/COMPSAC51774.2021.00039/VIDEO.

[22] Q. Chu, G. Li, R. Zhou, and Z. Ping, “Traffic Flow Prediction Model Based on LSTM with Finnish Dataset,” *2021 IEEE 6th International Conference on*

Intelligent Computing and Signal Processing, ICSP 2021, pp. 389–392, Apr. 2021, doi: 10.1109/ICSP51882.2021.9408888.

[23] R. de Medrano and J. L. Aznarte, “A spatio-temporal attention-based spot-forecasting framework for urban traffic prediction,” *Appl Soft Comput*, vol. 96, p. 106615, Nov. 2020, doi: 10.1016/J.ASOC.2020.106615.

[24] J. Mena-Oreja and J. Gozalvez, “On the impact of floating car data and data fusion on the prediction of the traffic density, flow and speed using an error recurrent convolutional neural network,” *IEEE Access*, vol. 9, pp. 133710–133724, 2021, doi: 10.1109/ACCESS.2021.3115709.

[25] X. Di, Y. Xiao, C. Zhu, Y. Deng, Q. Zhao, and W. Rao, “Traffic congestion prediction by spatiotemporal propagation patterns,” *Proceedings - IEEE International Conference on Mobile Data Management*, vol. 2019-June, pp. 298–303, Jun. 2019, doi: 10.1109/MDM.2019.00-45.

[26] A. Agafonov, “Traffic Flow Prediction Using Graph Convolution Neural Networks,” *10th International Conference on Information Science and Technology, ICIST 2020*, pp. 91–95, Sep. 2020, doi: 10.1109/ICIST49303.2020.9201971.

[27] J. J. Vázquez, J. Arjona, M. Linares, and J. Casanovas-Garcia, “A Comparison of Deep Learning Methods for Urban Traffic Forecasting using Floating Car Data,” *Transportation Research Procedia*, vol. 47, pp. 195–202, Jan. 2020, doi: 10.1016/J.TRPRO.2020.03.079.

[28] I. Loumiotis, K. Demestichas, E. Adamopoulou, P. Kosmides, V. Asthenopoulos, and E. Sykas, “Road Traffic Prediction Using Artificial Neural Networks,” *South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference, SEEDA_CECNSM 2018*, Nov. 2018, doi: 10.23919/SEEDA-CECNSM.2018.8544943.

[29] J. Ji, J. Wang, Z. Jiang, J. Ma, and H. Zhang, “Interpretable spatiotemporal deep learning model for traffic flow prediction based on potential energy fields,” *Proceedings - IEEE International Conference on Data Mining, ICDM*, vol.

2020-November, pp. 1076–1081, Nov. 2020, doi: 10.1109/ICDM50108.2020.00128.

[30] A. A. Mungen and I. Cetintas, "An Investigation about Traffic Prediction by Using ANN and SVM Algorithms," *3rd International Conference on Electrical, Communication and Computer Engineering, ICECCE 2021*, Jun. 2021, doi: 10.1109/ICECCE52056.2021.9514250.

[31] A. Sinha, R. Puri, U. Balyan, R. Gupta, and A. Verma, "Sustainable Time Series Model for Vehicular Traffic Trends Prediction in Metropolitan Network," *2020 6th International Conference on Signal Processing and Communication, ICSC 2020*, pp. 74–79, Mar. 2020, doi: 10.1109/ICSC48311.2020.9182755.

[32] I. Kalamaras, A. Drosou, K. Votis, D. Kehagias, and D. Tzovaras, "A multi-objective data mining approach for road traffic prediction," *IFIP Adv Inf Commun Technol*, vol. 519, pp. 425–436, 2018, doi: 10.1007/978-3-319-92007-8_36/FIGURES/3.

[33] I. Alam *et al.*, "Pattern mining from historical traffic big data," *TENSYMP 2017 - IEEE International Symposium on Technologies for Smart Cities*, Oct. 2017, doi: 10.1109/TENCONSPRING.2017.8070031.

[34] C. Silva and F. Martins, "Traffic Flow Prediction Using Public Transport and Weather Data: A Medium Sized City Case Study," *Advances in Intelligent Systems and Computing*, vol. 1160 AISC, pp. 381–390, 2020, doi: 10.1007/978-3-030-45691-7_35.

[35] A. A. Mungen and I. Cetintas, "An Investigation about Traffic Prediction by Using ANN and SVM Algorithms," *3rd International Conference on Electrical, Communication and Computer Engineering, ICECCE 2021*, Jun. 2021, doi: 10.1109/ICECCE52056.2021.9514250.

[36] K. J. Offor, P. Wang, and L. Mihaylova, "Multi-Model Bayesian Kriging for Urban Traffic State Prediction," *2019 Symposium on Sensor Data Fusion: Trends, Solutions, Applications, SDF 2019*, Oct. 2019, doi: 10.1109/SDF.2019.8916655.

[37] A. Ekart, A. Patelli, V. Lush, and E. Ilie-Zudor, “Genetic Programming with Transfer Learning for Urban Traffic Modelling and Prediction,” *2020 IEEE Congress on Evolutionary Computation, CEC 2020 - Conference Proceedings*, Jul. 2020, doi: 10.1109/CEC48606.2020.9185880.

[38] W. Lu, Y. Rui, Z. Yi, B. Ran, and Y. Gu, “A Hybrid Model for Lane-Level Traffic Flow Forecasting Based on Complete Ensemble Empirical Mode Decomposition and Extreme Gradient Boosting,” *IEEE Access*, vol. 8, pp. 42042–42054, 2020, doi: 10.1109/ACCESS.2020.2977219.

[39] P. Sun, N. Aljeri, and A. Boukerche, “A Fast Vehicular Traffic Flow Prediction Scheme Based on Fourier and Wavelet Analysis,” *Proceedings - IEEE Global Communications Conference, GLOBECOM*, 2018, doi: 10.1109/GLOCOM.2018.8647731.

[40] R. Wang, W. Shi, X. Liu, and Z. Li, “An Adaptive Cutoff Frequency Selection Approach for Fast Fourier Transform Method and Its Application into Short-Term Traffic Flow Forecasting,” *ISPRS International Journal of Geo-Information 2020, Vol. 9, Page 731*, vol. 9, no. 12, p. 731, Dec. 2020, doi: 10.3390/IJGI9120731.

[41] I. Loumiotis, K. Demestichas, E. Adamopoulou, P. Kosmides, V. Asthenopoulos, and E. Sykas, “Road Traffic Prediction Using Artificial Neural Networks,” *South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference, SEEDA_CECNSM 2018*, Nov. 2018, doi: 10.23919/SEEDA-CECNSM.2018.8544943.

[42] “seaborn.boxplot — seaborn 0.13.2 documentation.” Accessed: Mar. 07, 2025. [Online]. Available: https://seaborn.pydata.org/generated/seaborn.boxplot.html

[43] “Box Plot (Definition, Parts, Distribution, Applications & Examples).” Accessed: Mar. 07, 2025. [Online]. Available: https://byjus.com/maths/box-plot/

[44] “Plotting Time Series Boxplots | Towards Data Science.” Accessed: Mar. 07, 2025. [Online]. Available: https://towardsdatascience.com/plotting-time-series-boxplots-5a21f2b76cfe/

[45] N. E. Huang *et al.*, “The empirical mode decomposition and the Hubert spectrum for nonlinear and non-stationary time series analysis,” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1971, pp. 903–995, 1998, doi: 10.1098/rspa.1998.0193.

[46] “KerasTuner.” Accessed: Mar. 08, 2025. [Online]. Available: https://keras.io/keras_tuner/

[47] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/NECO.1997.9.8.1735.

[48] “The Math Behind Gated Recurrent Units | Towards Data Science.” Accessed: Mar. 09, 2025. [Online]. Available: https://towardsdatascience.com/the-math-behind-gated-recurrent-units-854d88aded65/

APPENDIX

Appendix A

Table A-1. Original Traffic Data Log Provided by Sigmind, AI Startup (September 19, 2024) [first few row is shown here]

date	time	camera_no	track_id	direction	hour	category_id
9/19/2024	07:00:05:040	Cam1	1	DOWN	71	6
9/19/2024	07:00:05:600	Cam1	15	DOWN	71	10
9/19/2024	07:00:06:760	Cam1	6	DOWN	71	7
9/19/2024	07:00:07:360	Cam1	12	DOWN	71	11
9/19/2024	07:00:11:040	Cam1	8	DOWN	71	9
9/19/2024	07:00:12:960	Cam1	53	DOWN	71	8
9/19/2024	07:00:14:280	Cam1	47	DOWN	71	6
9/19/2024	07:00:16:000	Cam1	5	UP	71	11
9/19/2024	07:00:17:600	Cam1	62	UP	71	11
9/19/2024	07:00:18:920	Cam1	73	UP	71	10
9/19/2024	07:00:19:400	Cam1	46	DOWN	71	11
9/19/2024	07:00:24:320	Cam1	90	UP	71	10
9/19/2024	07:00:26:760	Cam1	85	DOWN	71	9
9/19/2024	07:00:27:360	Cam1	64	DOWN	71	9
9/19/2024	07:00:30:120	Cam1	97	DOWN	71	6
9/19/2024	07:00:30:840	Cam1	98	DOWN	71	6
9/19/2024	07:00:32:680	Cam1	114	DOWN	71	3
9/19/2024	07:00:35:240	Cam1	118	DOWN	71	8

9/19/2024	07:00:35:480	Cam1	139	UP	71	5
9/19/2024	07:00:35:720	Cam1	115	DOWN	71	9
9/19/2024	07:00:38:880	Cam1	154	DOWN	71	5
9/19/2024	07:00:48:960	Cam1	181	DOWN	71	3
9/19/2024	07:00:49:840	Cam1	165	DOWN	71	11
9/19/2024	07:00:51:040	Cam1	179	DOWN	71	9
9/19/2024	07:00:51:240	Cam1	192	DOWN	71	10
9/19/2024	07:00:54:400	Cam1	209	DOWN	71	11
9/19/2024	07:00:54:720	Cam1	194	DOWN	71	10
9/19/2024	07:00:54:720	Cam1	208	UP	71	9
9/19/2024	07:00:57:120	Cam1	232	UP	71	9
9/19/2024	07:00:57:920	Cam1	238	UP	71	9
9/19/2024	07:00:59:680	Cam1	212	DOWN	71	4
9/19/2024	07:01:05:520	Cam1	254	UP	71	11
9/19/2024	07:01:06:080	Cam1	257	UP	71	10
9/19/2024	07:01:08:640	Cam1	263	UP	71	10
9/19/2024	07:01:11:520	Cam1	274	UP	71	7
9/19/2024	07:01:24:520	Cam1	309	UP	71	11
9/19/2024	07:01:24:760	Cam1	290	DOWN	71	
9/19/2024	07:01:25:920	Cam1	331	UP	71	4
9/19/2024	07:01:26:560	Cam1	301	DOWN	71	11
9/19/2024	07:01:27:680	Cam1	326	UP	71	9
9/19/2024	07:01:27:680	Cam1	340	UP	71	9
9/19/2024	07:01:27:720	Cam1	312	DOWN	71	4
9/19/2024	07:01:38:560	Cam1	366	DOWN	71	5
9/19/2024	07:01:42:160	Cam1	373	DOWN	71	5

9/19/2024	07:01:44:800	Cam1	365	DOWN	71	9
9/19/2024	07:01:45:760	Cam1	402	DOWN	71	5
9/19/2024	07:01:45:920	Cam1	404	UP	71	13
9/19/2024	07:01:46:320	Cam1	407	UP	71	2
9/19/2024	07:01:46:880	Cam1	415	DOWN	71	11
9/19/2024	07:01:48:760	Cam1	418	DOWN	71	11
9/19/2024	07:01:49:280	Cam1	426	UP	71	6
9/19/2024	07:01:49:600	Cam1	434	UP	71	11
9/19/2024	07:01:50:240	Cam1	433	DOWN	71	9
9/19/2024	07:01:51:520	Cam1	442	DOWN	71	6
9/19/2024	07:01:52:600	Cam1	428	DOWN	71	9
9/19/2024	07:01:54:000	Cam1	423	DOWN	71	4
9/19/2024	07:01:54:400	Cam1	477	UP	71	12
9/19/2024	07:01:54:680	Cam1	436	DOWN	71	6
9/19/2024	07:01:55:680	Cam1	450	DOWN	71	7
9/19/2024	07:01:55:840	Cam1	473	UP	71	4
9/19/2024	07:01:56:320	Cam1	464	DOWN	71	11
9/19/2024	07:01:57:560	Cam1	462	DOWN	71	9
9/19/2024	07:01:57:760	Cam1	481	DOWN	71	3
9/19/2024	07:01:58:800	Cam1	523	DOWN	71	3
9/19/2024	07:01:59:360	Cam1	492	DOWN	71	4
9/19/2024	07:01:59:360	Cam1	522	DOWN	71	9
9/19/2024	07:01:59:520	Cam1	479	UP	71	8
9/19/2024	07:02:00:800	Cam1	521	DOWN	71	11
9/19/2024	07:02:01:440	Cam1	524	UP	71	10
9/19/2024	07:02:02:240	Cam1	490	DOWN	71	9

9/19/2024	07:02:02:440	Cam1	534	UP	71	4
9/19/2024	07:02:03:680	Cam1	536	DOWN	71	3
9/19/2024	07:02:04:360	Cam1	544	DOWN	71	3
9/19/2024	07:02:04:560	Cam1	557	DOWN	71	11
9/19/2024	07:02:05:600	Cam1	547	UP	71	5
9/19/2024	07:02:08:240	Cam1	569	UP	71	11
9/19/2024	07:02:08:640	Cam1	584	UP	71	6

Appendix B

Table B-1 Five minute interval vehicle count (September 19, 2024)

date	time	bike	3 wheller	car/jeep	bus	open truck	truck	Total
19-09-2024	06:30 - 06:35	2	15	43	25	39	50	174
19-09-2024	06:35 - 06:40	4	22	43	19	35	55	178
19-09-2024	06:40 - 06:45	6	22	45	18	24	65	180
19-09-2024	06:45 - 06:50	8	27	46	19	30	62	192
19-09-2024	06:50 - 06:55	9	24	51	22	14	41	161
19-09-2024	06:55 - 07:00	8	26	50	21	27	56	188
19-09-2024	07:00 - 07:05	18	26	43	28	26	51	195
19-09-2024	07:05 - 07:10	11	23	47	21	24	52	178
19-09-2024	07:10 - 07:15	25	30	51	22	22	49	202
19-09-2024	07:15 - 07:20	5	23	45	39	18	58	191
19-09-2024	07:20 - 07:25	9	55	49	35	12	53	217
19-09-2024	07:25 - 07:30	13	65	40	39	22	36	219
19-09-2024	07:30 - 07:35	15	64	56	35	18	47	239
19-09-2024	07:35 - 07:40	8	81	52	29	15	38	229
19-09-2024	07:40 - 07:45	6	89	51	44	23	44	262
19-09-2024	07:45 - 07:50	21	90	51	37	23	41	269
19-09-2024	07:50 - 07:55	12	105	54	29	25	47	278
19-09-2024	07:55 - 08:00	12	86	46	33	18	45	245
19-09-2024	08:00 - 08:05	15	58	59	39	16	36	226
19-09-2024	08:05 - 08:10	11	39	48	31	14	38	184
19-09-2024	08:10 - 08:15	9	40	53	24	17	40	185

19-09-2024	08:15 - 08:20	13	41	51	42	15	44	208
19-09-2024	08:20 - 08:25	14	37	56	45	27	23	204
19-09-2024	08:25 - 08:30	14	50	51	24	19	36	194
19-09-2024	08:30 - 08:35	9	33	40	35	16	31	166
19-09-2024	08:35 - 08:40	12	38	61	39	25	35	210
19-09-2024	08:40 - 08:45	20	32	56	29	24	29	191
19-09-2024	08:45 - 08:50	15	44	46	26	19	26	179
19-09-2024	08:50 - 08:55	24	38	58	34	30	36	223
19-09-2024	08:55 - 09:00	27	37	57	30	20	44	220
19-09-2024	09:00 - 09:05	14	50	60	41	30	46	244
19-09-2024	09:05 - 09:10	24	38	59	33	20	35	210
19-09-2024	09:10 - 09:15	23	39	61	39	25	36	227
19-09-2024	09:15 - 09:20	19	38	52	32	28	21	194
19-09-2024	09:20 - 09:25	23	32	45	22	33	36	193
19-09-2024	09:25 - 09:30	18	53	54	31	25	54	235
19-09-2024	09:30 - 09:35	22	45	50	26	24	31	201
19-09-2024	09:35 - 09:40	29	39	56	29	20	36	211
19-09-2024	09:40 - 09:45	23	36	28	28	25	40	180
19-09-2024	09:45 - 09:50	25	39	30	24	19	38	179
19-09-2024	09:50 - 09:55	21	42	48	32	16	35	197
19-09-2024	09:55 - 10:00	15	43	41	30	23	27	181
19-09-2024	10:00 - 10:05	20	22	46	29	24	29	174
19-09-2024	10:05 - 10:10	23	41	34	30	17	38	183
19-09-2024	10:10 - 10:15	21	42	39	26	27	29	186
19-09-2024	10:15 - 10:20	23	35	35	38	25	50	210

19-09-2024	10:20 - 10:25	14	54	54	24	22	46	215
19-09-2024	10:25 - 10:30	11	46	34	32	17	36	178
19-09-2024	10:30 - 10:35	19	39	39	20	24	47	191
19-09-2024	10:35 - 10:40	19	44	43	36	20	52	216
19-09-2024	10:40 - 10:45	15	30	46	29	26	35	183
19-09-2024	10:45 - 10:50	17	59	45	22	15	26	188
19-09-2024	10:50 - 10:55	21	44	50	33	15	48	212
19-09-2024	10:55 - 11:00	10	43	39	28	16	43	181
19-09-2024	11:00 - 11:05	17	42	40	24	15	42	180
19-09-2024	11:05 - 11:10	8	44	46	42	13	37	191
19-09-2024	11:10 - 11:15	16	39	49	28	25	52	211
19-09-2024	11:15 - 11:20	12	35	43	44	16	44	196
19-09-2024	11:20 - 11:25	11	53	56	28	12	55	219
19-09-2024	11:25 - 11:30	15	37	56	34	19	42	205
19-09-2024	11:30 - 11:35	21	50	44	28	21	57	222
19-09-2024	11:35 - 11:40	16	44	49	37	24	38	209
19-09-2024	11:40 - 11:45	18	41	47	32	21	50	212
19-09-2024	11:45 - 11:50	17	44	58	32	20	49	222
19-09-2024	11:50 - 11:55	17	41	46	39	15	43	201
19-09-2024	11:55 - 12:00	11	53	42	26	20	49	203

Appendix C

The entire coding of the work is available at

https://github.com/Ausaf99/tree/main/UndergradThesis