Nevin Çini

A Ph.D. Thesis

AGU 2024

DEEP LEARNING MODELS FOR TRAFFIC VOLUME PREDICTION

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Nevin Çini

January 2024

DEEP LEARNING MODELS FOR TRAFFIC VOLUME PREDICTION

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Nevin Çini
January 2024

SCIENTIFIC ETHICS COMPLIANCE

I hereby declare that all information in this document has been obtained in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Name-Surname: Nevin Çini
Signature :

REGULATORY COMPLIANCE

Ph.D. thesis titled “Deep learning models for traffic volume prediction” has been prepared in accordance with the Thesis Writing Guidelines of the Abdullah Gül University, Graduate School of Engineering & Science.

Prepared By	Advisor
Nevin Çini	Assoc. Prof. Zafer Aydın
Signature	Signature

Head of the Electrical and Computer Engineering Program

Assist. Prof. Samet Güler

Signature

ACCEPTANCE AND APPROVAL

Ph.D. thesis titled “Deep learning models for traffic volume prediction” and prepared by Nevin Çini has been accepted by the jury in the Electrical and Computer Engineering Graduate Program at Abdullah Gül University, Graduate School of Engineering & Science.

12/01/2024

(Thesis Defense Exam Date)

JURY:

Advisor : Assoc. Prof. Zafer Aydın………………………………...............................…

Member: Assoc. Prof. Mete Çelik……………………………........................……....

Member: Assist. Prof. Bekir Hakan Aksebzeci…………………………………………

Member: Assist. Prof. Rıfat Kurban…………………………………………………….

Member: Assist. Prof. Yasin Görmez…………………………………………………...

APPROVAL:

The acceptance of this Ph.D.thesis has been approved by the decision of the Abdullah Gül University, Graduate School of Engineering & Science, Executive Board dated ….. /….. / 2024 and numbered .…………..……. .

……….. /……….. / ………..

Graduate School Dean
Prof. Dr. İrfan ALAN

ABSTRACT

DEEP LEARNING MODELS FOR TRAFFIC VOLUME PREDICTION

Nevin Çini
Ph.D. in Electrical and Computer Engineering Department
Advisor: Assoc. Prof. Zafer AYDIN
January 2024

In the last 50 years, with the growth of cities and increase in the number of vehicles and mobility, traffic has become troublesome. As a result, traffic flow prediction started to attract attention as an important research area. However, despite the extensive literature, traffic flow prediction still remains as an open research problem, specifically for long-term traffic flow prediction. Compared to the models developed for short-term traffic flow prediction, the number of models developed for long-term traffic flow prediction is very few. Based on this shortcoming, in this study, we focus on long-term traffic flow prediction and propose a novel deep ensemble model (DEM). In order to build this ensemble model, first, we developed a convolutional neural network (CNN), a long short term memory (LSTM) network, and a gated recurrent unit (GRU) network as deep learning models, which formed the base learners. In the next step, we combine the output of these models according to their individual forecasting success. We use another deep learning model to determine the success of the individual models. Our proposed model is a flexible ensemble prediction model that can be updated based on traffic data. To evaluate the performance of the proposed model, we use a publicly available dataset. Numerical results show that our proposed model performs better than individual deep learning models (i.e., LSTM, CNN, GRU), selected traditional machine learning models (i.e., linear regression (LR), decision tree regression (DTR), k-nearest-neighbors regression (KNNR) and other ensemble models such as random-forest-regression(RFR).

Keywords: Traffic Flow Prediction, Deep Learning, Time Series Prediction, Ensemble Learning, Convolutional neural network.

ÖZET

TRAFİK YOĞUNLUĞU TAHMİNİ İÇİN DERİN ÖĞRENME MODELLERİ

Nevin Çini
Elektrik ve Bilgisayar Mühendisliği Anabilim Dalı Doktora
Tez Yöneticisi: Doç. Dr. Zafer AYDIN
Ocak 2024

Son 50 yılda şehirlerin büyümesi, araç sayısının ve hareketliliğin artmasıyla birlikte trafik sıkıntılı hale geldi. Bunun sonucunda trafik akış tahmini önemli bir araştırma alanı olarak dikkat çekmeye başladı. Bununla birlikte, kapsamlı literatüre rağmen trafik akışı tahmini, özellikle uzun vadeli trafik akışı tahmini için hala açık bir araştırma problemi olarak kalmaktadır. Kısa vadeli trafik akışı tahmini için geliştirilen modellerle karşılaştırıldığında uzun vadeli trafik akışı tahmini için geliştirilen modellerin sayısı oldukça azdır. Bu eksiklikten yola çıkarak, biz bu çalışmada uzun dönem trafik akış tahmini problemine odaklanıyoruz ve yeni bir derin topluluk öğrenme modeli öneriyoruz. Bu topluluk öğrenme modelini oluşturabilmek için öncelikle, temel öğreniciler olarak kullandığımız 3 farklı derin öğrenme mimarisini (yani LSTM, CNN ve GRU) kullanarak 3 farklı derin öğrenme modeli geliştirdik. Daha sonra, bu modellerin bireysel tahmin başarılarına göre tahmin sonuçlarını birleştirdik. Bunun için ayrı bir derin öğrenme modeli kullandık. Önerdiğimiz model esnek ve dinamik bir yapıya sahiptir, model güncel trafik durumuna göre kendini yenileyebilir. Önerilen modelin performansını değerlendirmek için halka açık bir veri seti kullanıyoruz. Sayısal sonuçlar, önerilen modelimizin bireysel derin öğrenme modellerinden (örn. LSTM, CNN, GRU), seçilmiş geleneksel makine öğrenme modellerinden (örn. doğrusal regresyon (LR), karar ağacı regresyonu (DTR), k-en yakın komşular) ve rastgele orman regresyonu (RFR) gibi diğer topluluk öğrenme modellerinden daha iyi performans sergilediğini gösteriyor.

Anahtar kelimeler: Trafik Akış Tahmini, Derin Öğrenme, Zaman Serileri Tahmini, Topluluk Öğrenme, Yapay Zeka

Acknowledgements

I would like to thank to my advisor Assoc. Prof. Zafer AYDIN for his contribution in this thesis. And I would like to thank to everyone who contributed.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

EL	Ensemble Learning
DEM	Deep Ensemble Learning
DL	Deep Learning
ML	Machine Learning
LSTM	Long Short-Term Memory
GRU	Gated Recurrent Unit
CNN	Convolutional Neural Network
LR	Linear Regression
DTR	Decision Tree Regression
KNNR	K-Nearest Neighbors Regression
RFR	Random Forest Regression
TFP	Traffic Flow Prediction
FFNN	Feed Forward Neural Network
ARIMA	Autoregressive Integrated Moving Average
SARIMA	Seasonal Autoregressive Integrated Moving Average
RNN	Recurrent Neural Networks
SVM	Support Vector Machine
FC	Fully Connected
FPM	Final Prediction Model
MLP	Multi-Layer-Perceptron
NN	Neural Networks
K-NN	K-nearest-neighbor
DBN	Deep belief networks
ITS	Intelligent Transportation Systems
APE	Absolute Percentage Error
MRE	Mean Relative Error

To My Father

Chapter 1

Introduction

Traffic flow refers to the number of vehicles passing a certain road section per unit time. This data is collected automatically, usually with the help of sensors. Since vehicles can only move on the roads prepared for them, accurate estimation of the traffic flow in a certain area prevents possible congestion and ensures more efficient use of the roads [1,2,3].

Traffic flow prediction plays an important role in the construction of Intelligent Transportation Systems (ITS). Transportation planners try to predict the location and time of potential traffic congestion with the traffic flow forecasting applications. Thus, by controlling the traffic, they can increase the safety and comfort of the drivers and passengers.

Traffic congestion causes many problems that we can examine under different headings such as economic, environmental, social. Among them, the most emphasized is the increase in cost with the lengthening of the travel time. These two key issues lead to the emergence of other problems. For example, prolonged travel time causes social and psychological problems, environmental problems such as noise pollution and even accidents from time to time. Although increasing the cost of travel is an economic problem, increasing fuel consumption also leads to environmental problems such as air pollution.

Decision makers who reach information about when and where traffic congestion may occur with traffic flow forecasting can direct drivers to safer roads so that resources can be used more efficiently. With a more effective planning, it is possible to use public transportation more efficiently as well. In this way, the environmental impact caused by traffic can be reduced. For this reason, traffic flow forecasting is of key importance in controlling traffic congestion and solving many problems that may occur, and is an indispensable component for intelligent transportation systems.

Since the early 1980s, transportation engineers have conducted a lot of research to minimize the negative effects of traffic on life quality. However, the inadequacy of the number and quality of data and the inadequacies in data processing technologies limited the success of many studies. Thanks to the technological developments in hardware and software in recent years, it is possible to process large amounts of data in a short time. In addition, the available traffic data has increased. Today, researchers are investigating how big data can be processed most efficiently with smart algorithms and how more successful smart transportation applications can be developed with new technologies.

That's why, in recent years, many studies in this area have focused on developing reliable and realistic traffic flow prediction models using the latest technologies [2,4,5]. However, most of these studies have presented short-term traffic flow prediction models [6]. Few of the proposed forecasting models are capable of long-term forecasting. However, long-term forecasting is as important and useful as short-term forecasting [1,7,8]. Furthermore, long-term traffic flow forecasting is of practical importance for decision makers. An accurate forecast model will facilitate traffic management even during the rush hours, and will enable effective measures to be taken by informing in advance of possible negative events.

However, long-term forecasting is a challenging task. This is due to the stochastic nature of the dynamics that make up the traffic flow data, which is nonlinear and contains complex dependencies [9]. It is also not identical in both temporal and spatial dimensions. Modeling dynamic temporal and spatial dependencies for traffic flow prediction is very burdensome and arduous. These complex dependencies increase in number and become more and more complex in long-term predictions. As the forecast horizon increases, even in the best models, the prediction quality decreases and the average error increases [10].

As a result, reliable long-term prediction becomes a difficult task, and it is almost impossible to model long-term dependencies of traffic flow with simple and traditional prediction models [8,11,12].

In this study, we propose a deep learning-based ensemble framework for long-term traffic flow prediction. While deep learning (DL) models can learn dynamic and complex dependencies of traffic data better than traditional learning algorithms, ensemble learning (EL) provides flexibility by increasing the generalization ability of the final model. Because many different predictive models collaborate to solve the given problem in ensemble learning, it is often expected that the ensemble model will exceed the predictive success of a single model.

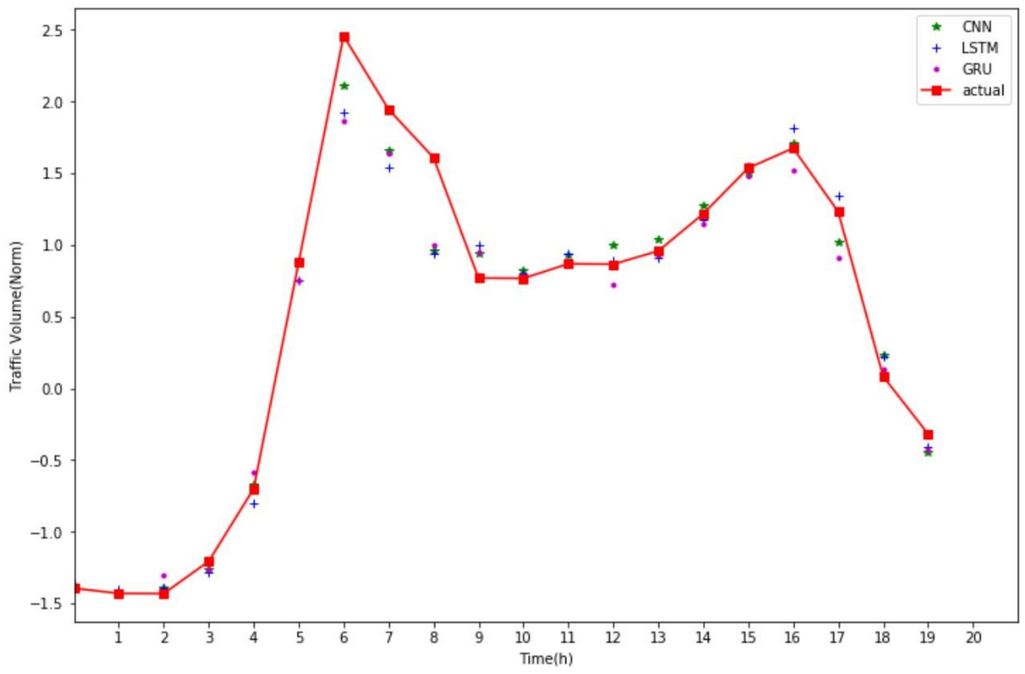


Figure 1.1 Different performances of base learners

The most important feature of the proposed EL model is that we employ three different DL models (i.e CNN, LSTM and GRU) as base learners. This increases model diversity so that a failure of one model can be compensated by another model. As shown in Figure 1.1, the performances of all three models change as traffic conditions change. From this figure (Figure 1.1), it is clear that we cannot achieve the best prediction performance with a single model. Because each model has strengths and weaknesses, the contribution of the base models to the final prediction result cannot be equal. In a successful ensemble model, a base model with high predictive performance is expected to contribute more to the final result than less successful models. In our ensemble model, we have developed a meta-learner to provide this. Owing to this meta-learner, we have dynamically weighted the base-models, that is, we have ensured that each model contribute in the final prediction result according to its current prediction performance. We leverage this capability of ensemble learning to improve long-term prediction accuracy. In order to assess the accuracy, we conducted several experiments, in which we compared the proposed model with widely used prediction models.

There are three main contributions of this study:

1. In this study, we proposed a fully DL-based ensemble learning framework for long-term traffic flow prediction. To the best of our knowledge, this is the first time a fully DL-based ensemble model is proposed for long-term traffic flow prediction.

2. We used three different DL models as base-learners. In the model we developed, we use LSTM and GRU together. We have not come across a model in literature that uses these two techniques together. Since these two techniques are versions of recurrent neural networks, it is not preferred to use them together in a prediction model. However, although these two techniques are similar to each other, their performances are quite different as seen in Figure 1.1. So where one fails, the other can be quite successful. For this reason, we preferred to use these two techniques together.

3. We use deep learning architectures in our model, both as base learners and the meta-learner. Thanks to a feed forward neural network (FFNN), which is the most basic deep learning technique, we decide the weights of the base learners. We train a feed forward neural network as a meta-learner in order to obtain the final prediction result. In this way, we ensured that the base-learners dynamically contribute in the final prediction result according to their prediction success (more successful ones contribute more, less successful ones contribute less).

The study is organized as follows: A brief overview of current literature on traffic flow prediction is provided in chapter 2. In chapter 3 we provide a background section and introduce the details of our deep learning-based ensemble framework. Then, we present dataset, preprocessing steps in chapter 4 and experimental results and discussion in chapter 5. Chapter 6 includes conclusion and future work.

Chapter 2

Related Work

The importance of traffic flow prediction in transportation engineering is increasing, and accordingly, we can say that there is a very large literature in this field. Most studies propose a model to predict traffic flow. We will examine these proposed models under two topics by following the tradition in the literature: Parametric models and non-parametric models [4, 12, 15–19].

We summarize the related literature in Table 2.1.

2.1 Parametric Models

Models in this class can be explained by traffic flow theories of transportation engineering, statistics and probability. In a parametric model, traffic flow is represented as a function of random variables (e.g., accident, instantaneous decisions of drivers), time-dependent variables (e.g., time of day, day of the week, or season), and auxiliary variables (e.g., weather, public holidays, sports or concert events). That is, traffic flow is defined as the total number of vehicles passing through a certain road segment at a certain time period under the influence of many dependent or independent variables, each of which is dynamic in itself. Modeling with parametric approaches is relatively easy, but these models are suitable for uncomplicated small-sized data sets [19].

The most widely used parametric approaches in literature are ARIMA, kalman filtering and linear regression.

ARIMA is a time series modeling approach that explores the temporal relationship between data points of a time series. There are many traffic flow forecasting models developed using ARIMA and its advanced versions i.e., ARIMAX, SARIMA, SARIMAX in literature [20, 21].

Kalman Filtering is a widely used traffic flow prediction method. Its main idea is to predict future traffic flow using historical traffic flow data with a recursive or iterative process [22].

Linear regression is a pretty simple parametric approach. This method describes the traffic flow as a linear combination of the independent variables [23].

2.2 Non-parametric Models

Models in this category are more advanced than parametric models, and their performance varies according to the quality and size of the dataset. These models can achieve satisfactory prediction success with big data, but this requires quite a lot of computational capacity. K-nearest-neighbor (K-NN), support vector machine (SVM) and neural networks (NN) are the approaches we can count in this category.

K-NN can be used for classification or regression. In this model, common patterns are tried to be extracted from historical traffic flow data. By using the best match with these defined patterns, future traffic patterns are tried to be predicted [24].

Another parametric model used in traffic flow prediction is SVM [15]. Although the estimation accuracy can be increased by using different ”kernels”, the computational load of model training is quite high, especially compared to K-NN and NN. Therefore, it is not practical for large datasets.

Indeed, K-NN and SVM are not popular models developed for traffic flow forecast. The most popular models in this category are the NN-based models. And the reasons why NN-based models are so popular can be listed as follows: (1) They are suitable for big data, (2) They have fast convergence, (3) They can achieve high prediction accuracy. A wide variety of NN models have been proposed for traffic flow prediction [1, 7, 12, 18].

2.3 Deep Learning-Based Models

Although deep learning models are also non-parametric models, we wanted to examine these studies separately since they have been very popular in this field in recent years and have a fairly wide literature. The simplest DL models that can be found in literature in this field are multi-layer-perceptron (MLP)-based models developed using multiple hidden layers [25].

However, the most widely used DL technique in solving the traffic flow prediction problem is recurrent neural networks (RNN). Especially, GRU and LSTM techniques, which are variants of RNN, are the most common methods since they are successful in capturing dependencies at different times. For example [26], developed a two-layer LSTM-based model. It used a fully connected layer as the extraction layer in the first layer, and the LSTM layer as the prediction layer in the second layer. The proposed other LSTM-based models are in [6].

A GRU-based model is proposed in [27]. In this study, weather data was used in addition to traffic data. Apart from RNN, CNN-based models also have been proposed for short-term traffic flow forecasting problems [28–30]. CNN-based models are especially preferred because they can produce results faster than other neural networks [31].

2.4 Hybrid and Ensemble Models

Understanding that it is not possible to model the complexity of traffic data with simple and traditional methods, many researchers have turned to hybrid models, especially in recent years. While in early studies we can see the combination of several parametric models, in recent studies, many of the hybrid models were build by combining two or more non-parametric methods [7, 8, 11, 32, 33].

Especially LSTM and CNN are used together in recently developed hybrid models [29, 34, 35]. There are also hybrid models developed by using parametric and non-parametric methods together [11, 36].

On the other hand, EL-based models emerge as a new trend [37–42] . There are only a limited number of EL-based prediction models in the literature [16, 17, 21, 43–45]. However, none of these studies focus on long-term forecasting. And this is a research gap that we want to fill in this study.

Table 2.1 Summary of the related literature

Ref.	**Horizon**	**Input data**	**Data size**	**Method/ Technologies used**	**Evaluation metrics**
[9]	24 hour	Highways Agency Network Traffic Flow Data	15 minutes resolution from 1 Jul 2018 to 28 Jan 2020	Wavelet decomposition, CNN, LSTM	RMSE, MAE, R square
[7]	Up to 24h	Caltrans Performance Measurement System (PeMS) dataset	5 minutes interval. data size not mentioned.	LSTM encoder-decoder	RMSE, Symmetric MAPE
[1]	24 hour	Dataset obtained from the DRIVENET	From February 1, 2015 to March 31, 2016	Deep neural network (DNN)	APE, MAPE
[8]	Up to 4h	GPS-data taken from the GAIA	from October to November 2016	Graph CNN-LSTM	RMSE, MSE, MAE and MAPE
[12]	Up to 1h	Victoria Street (Melbourne)	One-year data (2016)	Convolutional GRU with attention network	Weighted MAPE, RMSE, MAE
[16]	single step (unspecified)	PeMS	District 5 named Central Coast in 2013	Ensemble learning , CNN	MAE, RMSE, MRE, and the standard deviations MAE, MRE and RMSE

Table 2.2 Summary of the related literature (Continues)

Ref.	Horizon	Input data	Data size	Method/ Technologies used	Evaluation metrics
[21]	1-hour	Princes Highway, Victoria Road, Canterbury Road, and M1 in Sydney	Hourly traffic count from November 2017 to November 2018	Ensemble learning, ARIMA	RMSE, MAPE
[29]	5 hours	Data from Hangzhou Integrated Transportation Research Center and PeMSD10	from 16th October, 2013 to 3rd October, 2014 and from 1st January, 2018 to 31st March, 2018 (15 minutes resolution)	Graph convolutional network, Recurrent neural network	RMSE and MAPE
[32]	1 hour (12 steps)	PeMSD4 and PeMSD8	from January to February in 2018 and from July to August in 2016 (5-minutes interval)	Encoder-decoder, attention network, LSTM	RMSE, MAE, MAPE, median absolute error (MdAE), mean absolute scaled error (MASE)

Table 2.3 Summary of the related literature (Continues)

Ref.	Horizon	Input data	Data size	Method/ Technologies used	Evaluation metrics
[17]	Up to three-steps	Portland-Vancouver	from March 4 to June 28, 2019	Ensemble learning, Ensemble empirical mode decomposition, DBN (Deep belief networks)	RMSE, MAPE
[37]	1 hour (4 steps)	arterial sensors in Arcadia, CA in 2015	15-minutes interval data	Ensemble learning,ARMAX, Partial Least Squares, Support Vector Regression, Kernel Ridge Regression, Gaussian Process Regression	MAE and StdAE(standard deviation)
[38]	Up to 30 min	PeMS(``freeway segment located in San Diego")	from September 1, to September 30, 2019, interval is 5 min	Ensemble Empirical Mode Decomposition, Wavelet, LSTM	RMSE, MAE, MAPE
[45]	Up to 30 min	Yuanda Road, Furong District, Changsha City	from September to October in 2013, excluding weekend time interval is 5 min'	optimized variational mode decomposition (OVMD) and (LSTM)	RMSE, MAE

Chapter 3

Methodology

3.1 Problem Formulation

Traffic flow forecasting models are often based on a simple assumption: the future depends on the past. In other words, data that generated traffic conditions in the past will affect current and future traffic situation. Therefore, continuity of data is important. Traffic flow prediction is a time-series problem, and as with all time-series problems, past values are used as target function parameters in the traffic flow estimation problem. In other words, the target/prediction value at time T_n becomes one of the target function parameters at time $T_{(n+1)}$. This is for single-step prediction. In multi-step prediction, more than one value at consecutive time steps, participates in the process at the same time. To formulate this problem mathematically, we use the notation to f_t^i define traffic flow from station i at time t. In order to extract spatial and temporal features of traffic flow here, we construct spatial-temporal feature matrix as follows:

$$f_t^s = \begin{bmatrix} f_1^1 & f_2^1 & f_3^1 & \cdots\cdots & f_t^1 \\ f_1^2 & f_2^2 & f_3^2 & \cdots\cdots & f_t^2 \\ . & . & . & \cdots\cdots & . \\ . & . & . & \cdots\cdots & . \\ . & . & . & \cdots\cdots & . \\ f_1^s & f_2^s & f_3^s & \cdots\cdots & f_t^s \end{bmatrix} \tag{3.1}$$

Here, s denotes the number of stations. We construct this flow matrix with temporal information horizontally and spatial information vertically. In the next step, we can formulate the traffic flow prediction problem as follows:

$$\begin{bmatrix} f^{d}_{t-\beta} \\ f^{d}_{t-(\beta-1)} \\ f^{d}_{t-(\beta-2)} \\ . \\ . \\ . \\ f^{d}_{t-1} \end{bmatrix}^{T} \xrightarrow{\theta} \begin{bmatrix} f^{d}_{t} \\ f^{d}_{t+1} \\ f^{d}_{t+2} \\ . \\ . \\ . \\ f^{d}_{t+(h-1)} \end{bmatrix}^{T} \tag{3.2}$$

Here, since we use historical flow data to predict future flows, the matrix on the left hand side represents historical flow data and the matrix on the right hand side represents prediction values. The traffic flow prediction model is represented by a prediction function, which is represented by θ. f^{d} denotes traffic flow from station d. β is the looked-back steps, and h is the prediction horizon.

3.2 The Differences Between Short and Long-term Prediction

In fact, the difference between the short and long-term forecast goes far beyond the period we determine with only the prediction horizon. In literature, long-term forecasting is categorized as predicting an hour later or a few steps later (usually 5 steps or more), while short-term forecasting is defined as predicting one step or a few minutes later. Here, we can say that a categorization based on this definition is not reliable due to the lack of a consensus in terms of the prediction horizon. However, according to the assessments made taking into account the time interval of the data, it is reasonable in our opinion to consider 5 steps and beyond as a long-term forecast.

3.3 Related Technologies

In this section, the related technologies used in the proposed model and methods used to compare are described.

3.3.1 Linear Regression (LR)

The linear regression model, which is considered the simplest method to model the correlation between the dependent variable and independent variables, makes inferences assuming that the data are randomly distributed. Linear regression is a parametric method that tries to describe the relationship between at least two variables with a linear function.

This method can be widely used to predict future data from existing data or to analyze and understand existing data. Linear regression is a frequently used technique, and with this method it is possible to create good predictive models for many problems because there are usually linear as well as non-linear relationships between variables. And linear regression can easily describe these simple relationships.

However, linear regression makes modeling with many assumptions that are not possible in practice. For example, it assumes that the variance of the predicted value is the same for all values of the independent variable. This is not accurate for many problems and causes incorrect prediction results. Linear regression can be simply defined as $z=kx+c$. Here, z is the dependent variable, x is the independent variable, k is the weight value that needs to be optimized, and c is the point where the equation intersects the constant or axis. In this way, linear regression allows us to understand how and by how much the dependent variable will change as the independent variables change.

3.3.2 K-Nearest Neighbors (KNN)

In fact, k-nearest neighbors (k-NN) is a classification algorithm, a non-parametric method that is widely used in different fields, which we can count among the supervised learning methods of machine learning.

Although it is recommended for classification problems, there is also a version used in regression problems. In the simplest sense, KNN creates a prediction by looking at the closeness of the data to be predicted to the existing data, without actually creating a mathematical prediction model. Euclidean, Manhattan and Minkowski distance measurement formulas are generally used to measure the distance between two data. The K value indicates the number of nearest neighbors and is a value that greatly affects the prediction performance. This value can be optimized by various methods or the best K value can be found by trial and error.

This method is quite simple and effective. The main disadvantage is that it is necessary to scan the entire data set each time to obtain each predictive value. This is very time consuming, especially for large data sets. However, it is quite faster than many machine learning methods (such as SVM) because it does not waste time finding a prediction function at the beginning.

The classification version uses the class value with the densest K nearest neighbors in the current data set to decide the class value of the data to be classified. The regression version produces prediction values by simply averaging the y values of the K nearest neighbors.

3.3.3 Decision Trees (DT)

Decision trees are a well-known non-parametric method in machine learning and were first proposed as a classification algorithm. Especially its simplicity, understandability and ease of implementation has made it a preferred method in solving many problems.

The advantages of this method based on generating rules are that the data set-specific rules can be easily visualized and the prediction process is understandable. Decision trees are a supervised learning method created by continuously dividing the data set sequentially and gradually in order to maximize the difference variable between data points included in defined classes. So this method does not actually use a mathematical model to produce predictive values (unlike SVM or Linear regression). It simply divides the data set according to a certain criterion, and after each division, two groups within and outside of this certain criterion are formed. The method considers these two groups as two different classes.

The most important difficulty for decision trees is where to split the data set and when to stop splitting the data set. To overcome these difficulties, entropy information is often used. Entropy is a versatile concept, but as used in decision trees, Entropy measures the irregularity of the set of data that forms the classes that will emerge as a result of each split. In other words, the less data there is from other classes among the data in a class, the lower the Entropy value.

Although decision trees were developed as a classification algorithm, there is also a version proposed for regression problems. In this version, decision trees divide the data set and create many different groups from the data set, as in the classification version. However, instead of class names, leaf nodes contain the average of the target values of the data that make up that class. Max depth (maximum depth of the tree) or minimum information gain can be used as stopping criterion.

3.3.4 Random Forest (RF)

The random forest model is an ensemble learning model in which decision trees are used as single learners. It is a prediction model developed for discrete problems, but it also has a version developed for continuous problems.

The random forest model can consist of many decision trees depending on the difficulty level of the problem. In its simplest form, the model works as follows: First of all, the model creates different sub-data sets from the raw data set for each decision tree. Using the data sets it creates, it trains an independent and different decision tree base learner model for each data set. Trained models produce their individual predictions. For classification problems, the most preferred class among the predictions becomes the final prediction class. For regression problems, the final prediction result is found by calculating the average of the predictions of the decision trees.

The most important feature of the random forest model is that it reduces the risk of overfitting because it consists of many decision trees with different architectures. In addition, since the random forest model is an ensemble learning model, the developed prediction model is expected to have high generalizability.

3.3.5 Recurrent Neural Network (RNN)

This model is one of the most important DL techniques particularly developed for time series problems. RNN has a simple feedback loop in order to learn dependencies among the different time intervals. However, the basic RNN architecture is insufficient to capture complex relationships in long time intervals, so two different versions have been proposed.

The first of these versions is long-short-term memory (LSTM) and the other is gated-recurrent network (GRU). LSTM has three different gates (input, output, forget) while GRU has two different gates (update and reset) and by the agency of these gates, they remove unnecessary information from the model that comes from the past states, and allowing the model to focus on only useful information. In this way, the model can learn long-term dependencies with ease. Figure 3.1 presents the general structure of the RNN, LSTM and GRU.

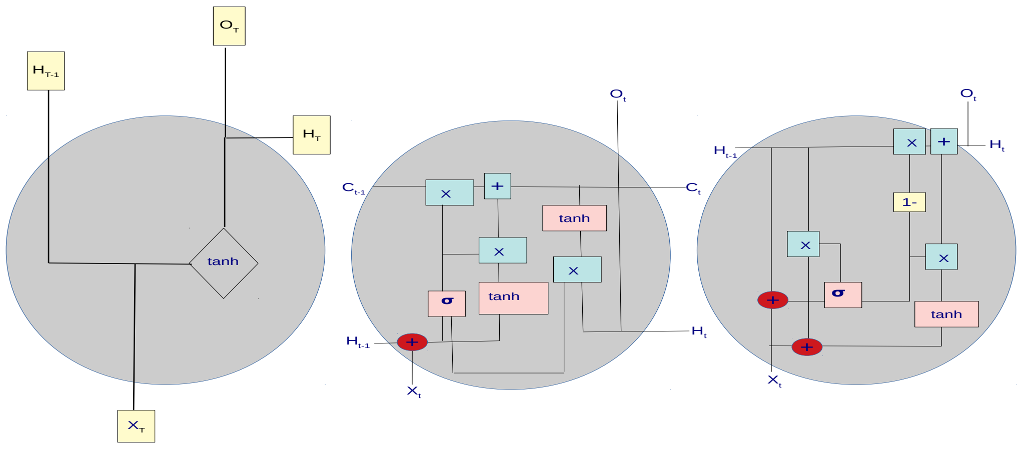


Figure 3.1 Structure of a RNN (on the left), LSTM (in the middle) and GRU (on the right).

3.3.6 Convolutional Neural Network (CNN)

Since CNN was not developed for time series problems, it was not used for time series prediction for a long time. However, with the increase in the amount of data, the increase in computational load and the inability to parallelize the RNN algorithm efficiently led to new searches. CNN is promising for time series problems as it can be parallelized and produces faster results.

In recent years, successful CNN-based time series forecasting models have been developed. Due to its architecture, CNN is used to reveal the relationships of different time series, especially in problems that need learning temporal dependencies and spatial dependencies together. A simple CNN model includes convolution layers, pooling layers, fully connected layers (FC), and an output layer as can be seen in Figure 3.2. Filters are used in the convolution and pooling layers and the results are combined in the FC layer. In this way, learning is provided at each convolution layer.

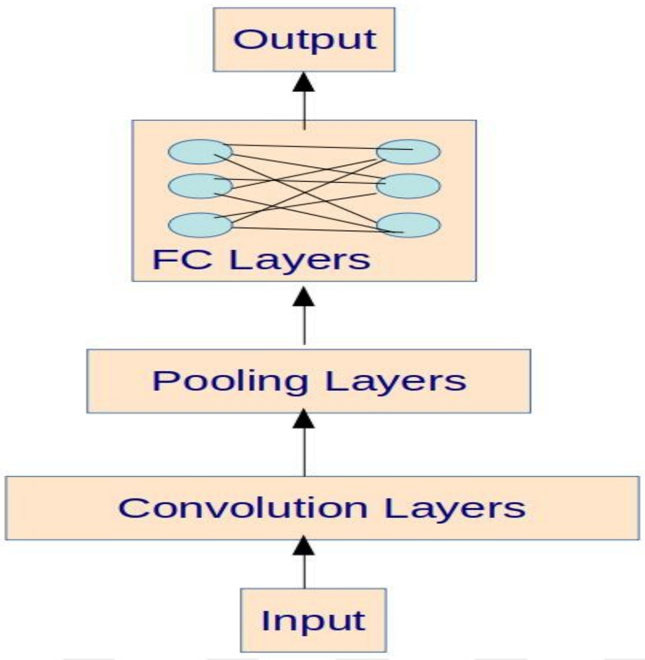


Figure 3.2 A simple CNN model

3.3.7 Deep Ensemble Learning

Ensemble learning models combine several base or individual models with different strategies in order to provide better generalization and improve final prediction performance [13, 14]. Moreover, today, deep learning models with complex and layered architecture outperform traditional prediction models. Deep ensemble learning models, on the other hand, aim to build a more successful prediction model by combining the peculiar advantages of these two models.

There are many models developed for traffic flow prediction in the literature, but few of them are ensemble learning-based. Whereas, ensemble learning-based models provide higher accuracy and generalizability because they are constructed by combining either individual models developed with different combinations of the same method or individual models developed using different methods. Combining multiple models in this way for traffic flow forecasting can increase the final forecasting accuracy while preventing overfitting. Because each individual model deals with one aspect of the final model, as a result, the final model provides a more general representation and achieves a higher predictive accuracy compared to individual models. To this end, we focus on ensemble learning approaches in this study and propose a novel deep ensemble model for traffic flow prediction. The formula for an ensemble model is as follows:

$$FPM(t) = \sum_{k=1}^{K} W_k \alpha_k(t). \tag{3.3}$$

Where FPM is final prediction model, α_k is the k th individual model, W_k is the weight of the k th individual model and K is the number of individual models. This form of ensemble learning is called “Stacking Ensemble” in literature.

According to this formula, the ensemble learning model gives weight to each individual model. The most common approach in the literature, for this purpose, is to give equal weight to each model. One issue of this approach is that each model conduces equally to the final prediction, without considering the prediction performance of single models. When we give a fixed weight to each single model, we limit the performance of the ensemble model due to a reduction in its generalization ability. Therefore to improve the prediction accuracy, we propose a flexible and robust deep ensemble model in this study. The proposed model assigns the weights based on the individual model performance and traffic situation change. Equation 3.3 is the general formula of an ensemble model and is its simplest form. We propose a complex FFNN-based meta-learner to optimize the weights in our proposed ensemble model.

3.4 The Proposed Model

The proposed model is a deep ensemble model which is capable of properly fusing the prediction results of multiple deep learning models. Our model learns the strengths and weaknesses of individual models and weights the predictions of single models according to their prediction performance. In addition, our model is flexible and performs well under different traffic conditions since our model receives actual data as well as prediction results from each model to obtain the final result.

Figure 3.3 demonstrates the details of our deep ensemble traffic flow prediction framework. Our proposed model consists of three stages. The first stage is the preprocessing and dataset preparation. We will explain this stage in detail in the next section.

The second stage is base model selection. At this stage, we adjust the configurations of the three base models namely LSTM, GRU and CNN. For this, we run models LSTM, GRU and CNN multiple times with different time-lags, numbers of hidden layers and neurons. We optimize the internal parameters of each base model and select the best models with the highest accuracy. After selecting the base models, in the third stage, we decide how much each base model will contribute to the final model according to their performance. That is we develop a meta-learner to dynamically weight each base learner. For this, we first form new training, validation and test sets using each base model, then by using these new datasets we build a feed forward neural network-based (FFNN) model with deep architecture and the outputs of this final model (i.e., FFNN) or meta-learner are the predicted traffic flow values. With this meta-learner, we can learn the weights of each base model. Thus, the weight of each base model is determined automatically. Figure 3.3 shows the general structure of a feed forward neural network.

Meanwhile, in order to capture the traffic condition changes, we use raw input data as well during the construction of the final model. Consequently, we separate the base models weighting step from the base models selection and tuning step so that the ensemble model can be dynamic and can change with the traffic conditions.

As we mentioned in section 3.3.7, an ensemble model can be built in two different approaches: It can built by combining either individual models developed with different combinations of the same method or individual models developed using different methods. The novelty of our model is that it combines these two approaches. Moreover, the base-learners and meta-learner we use in our model all have deep architecture, and we don’t use a fixed weight for each base-learner, we introduce a meta-learner with the ability of dynamically weighting the base-learners according to their predictive success.

3.5 The Alternative Model

We suggest a different ensemble model that can be an alternative to the model We suggested in the previous section. as seen in Figure 3.4, the alternative model is quite similar to the proposed model that we introduced in Section 3.3. In this model, We use three different deep learning models as base learners: These are LSTM, CNN and GRU and we use FFNN as a meta-learner, as well. However, in the first model, we provide 4 inputs to meta-learner. 3 of these are outputs from base-learners. The 4th input is the raw data input. In other words, in order to train the meta-learner, we reuse the data we prepared as the training set to train the base learners. In the alternative model, we provide only the outputs from base-learners as input to meta-learner, that is, we provide 3 inputs in total to meta-learner. we proposed this model as an alternative ensemble model and repeated all experiments for this model as well. Our goal is to see how the difference between two ensemble models will affect the result. In other words, we wanted to measure how and to what extent the result is affected by the absence of raw data input. In the results section, the experimental results of the two models for 8 time horizons can be found comparatively.

For the experiments, we used the same dataset in both models. The base learners we use in both models are the same, but since the meta learners are different, we only optimized the meta learners separately for each time period. In this way, we tried to get the best performance from both models for each time horizon. We call this alternative ensemble model "Ens1" in the results tables.

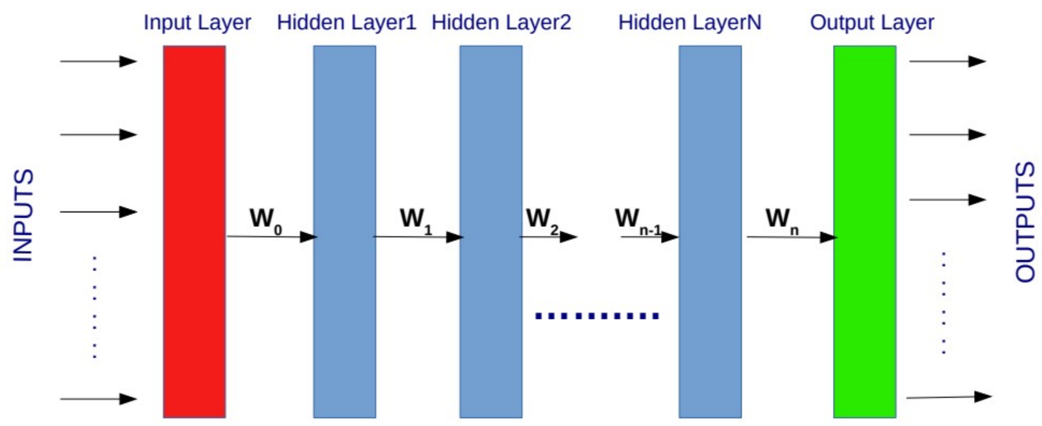


Figure 3.3 General Structure of Feed Forward Neural Network

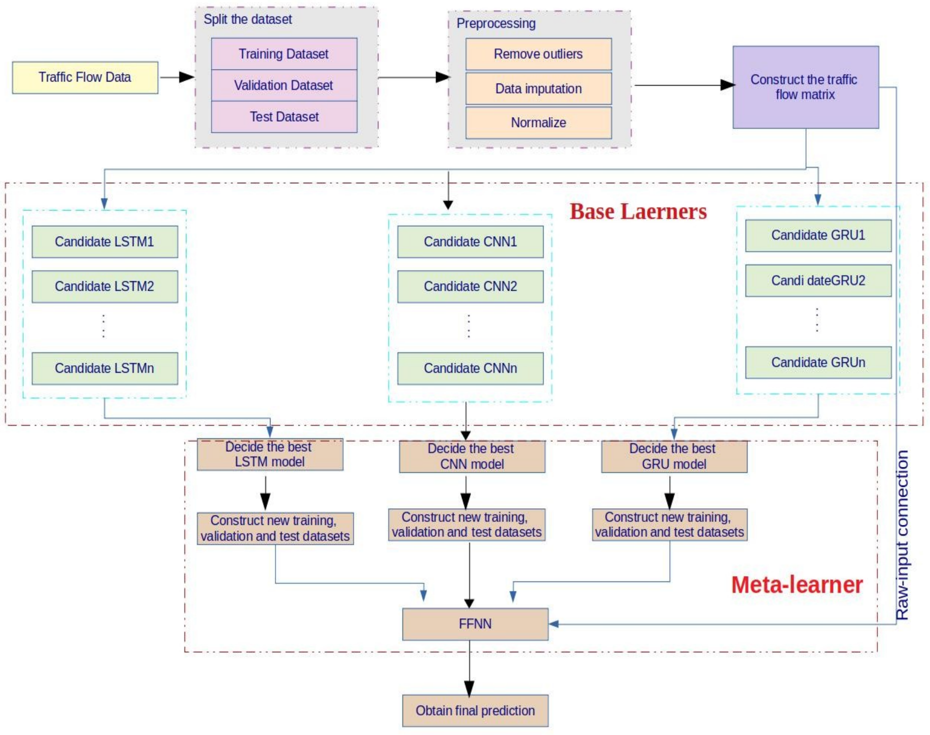


Figure 3.4 General structure of the proposed model.

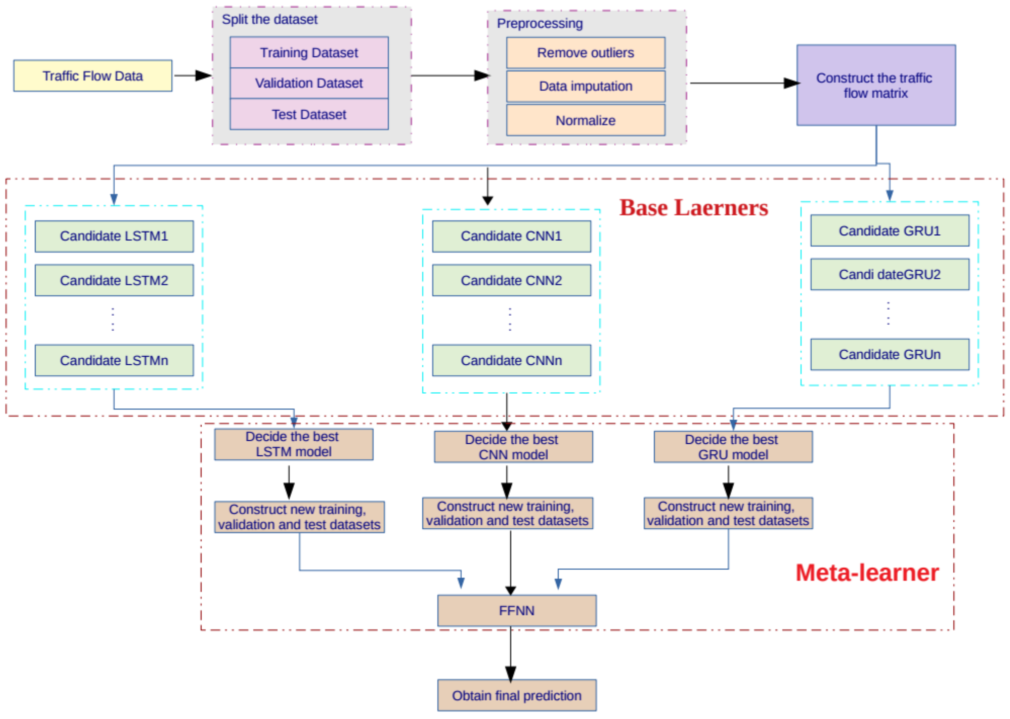


Figure 3.5 General structure of the alternative model.

Chapter 4

Experiments

4.1 Dataset and Preprocessing

We conducted this study with a publicly available and a real-world dataset[1]. The dataset contains a total of 274 stations. The data was collected from January 1st, 2015 to December 31st, 2015, which contains both weekends and weekdays and aggregated one hour intervals.

In addition to the traffic volumes for each hour as a feature, the dataset also includes supplementary information which can be used to build the prediction model and is shown in Table 4.1.

Table 4.1 Attributes and descriptions.

Attributes	**Descriptions**
Date	Date the data was collected
Day of data	What day of the month
Day of week	What day of the week
Direction of travel	Direction of the road
Lane of travel	Which lane of the road
Month of data	Which month of the year
Station id	Unique id assigned to each road/road segment
Year of data	Year
Functional classification name	Type of the road
Traffic Flow	Traffic flow information aggregated one hour intervals

Although the dataset contains 274 stations, some stations only have data for 3-4 months, for instance, station 116820 has data only for the 2nd, 9th, 11th and 12th months. That is, for some stations there are too many missing values, and this disrupts the continuity of the dataset.

[1]Source: www.transportation.gov/data, and it is available at: https://cloud.google.com/bigquery/public-data

However, this is not desirable for time series and can significantly reduce the forecasting quality. Therefore, both because our computational resources are limited and because we want our model to produce more reliable predictive results, we have selected 100 stations with as few missing values as possible and we tested all prediction models by using these 100 stations. we can list the criteria we consider when choosing these 100 stations as follows: Stations should

- have the same state code
- include data for all months of a year
- include at least fifteen day data for a month

Figure 4.1 shows the 10 stations we have selected and Figure 4.2 shows the locations of the selected stations.

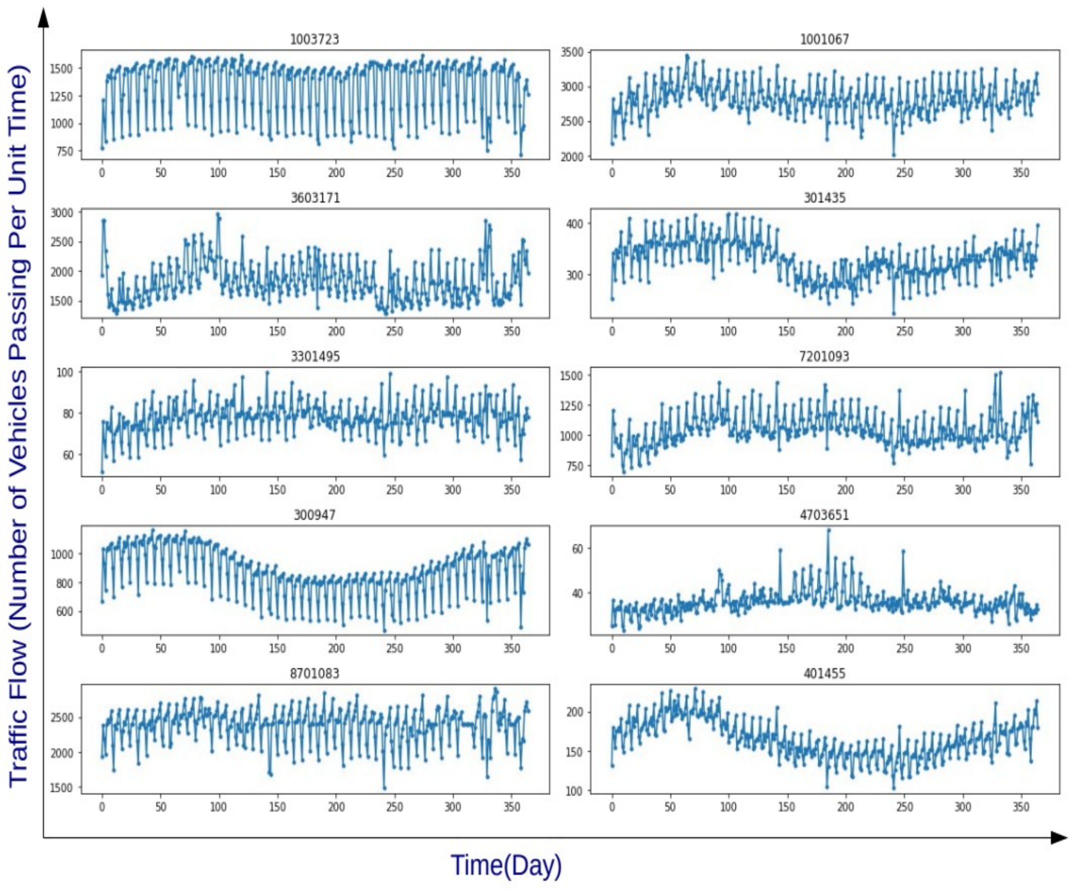


Figure 4.1 Dataset (10 stations).

We filled the missing values of the stations used in the experiments by averaging the data of the previous and the next hour. Thus the total number of data samples is 100∗365. We chose this method to fill the missing data because data that are closer together, whether spatially or temporally, are more related to each other than data that are far apart. This idea is based on Tobler’s first law, which says that things that are close together are more related to each other [21]. Inspired by this, we used this method to fill the missing data.

While choosing the stations we will use in our experiments, we also took into account the road type to which the station belongs, in addition to the amount of data because we wanted to show how robust and generalizable our model is for different road types.Table 4.2 shows the road types we used and their percentages in the data set.

We separated the dataset into three: We organized 65% of the dataset (about the first eight months) as the training set, the last two months as the validation set, and the remaining part as the test set. And we performed Z-Score normalization.

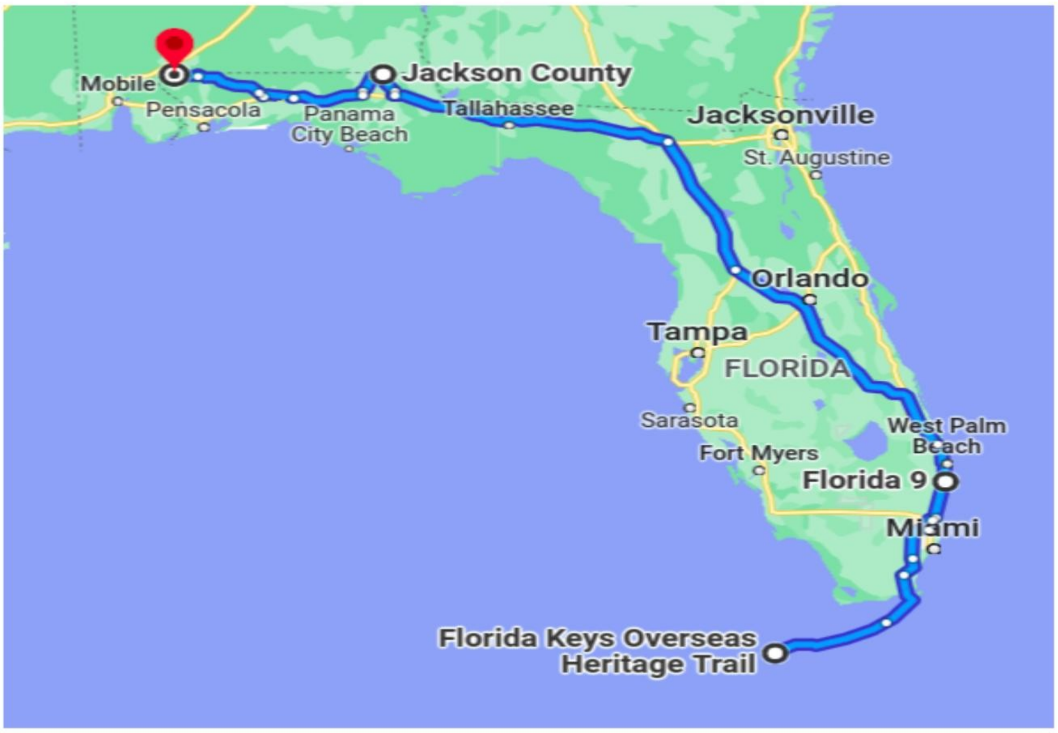


Figure 4.2 Road network used for experiments.

Table 4.2 Road Types

Urban: Principal Arterial - Other	30%
Rural: Principal Arterial - Other	20%
Urban: Principal Arterial - Interstate	14%
Urban: Minor Arterial	10%
Rural: Minor Arterial	10%
Urban: Principal Arterial - Other Freeways or Expressways	8%
Rural: Principal Arterial - Interstate	6%
Rural: Major Collector	2%

4.2 Constructing Traffic Flow Matrix

We tried to find the optimum time lag by running each deep learning model (base learner) many times with different time lags, i.e., the current traffic flow depends on how many steps in the past traffic flow. Thus, we have obtained an optimum time lag for each base learner.

If we show the time-lag value with W; we set W hours as the time lag and added W new features, each of which indicates hourly traffic volumes in a W-hour period. In this way, prediction models try to predict the traffic volume in the (W+1)th hour by using previous W hours of data.

We tested the proposed model for multiple horizon values: The prediction horizon h is specified as 1 for single step prediction, and 2, 3, 4, 5, 9, 12, 24 for multi-step prediction (i.e., long-term prediction). That is, we used W hours historical data to predict the following h hour(s) traffic flow value. Accordingly, we constructed the traffic flow matrix as input(X) and output(Y) matrix as follows:

$$X_h^{s1} = \begin{bmatrix} f_t^{s1} & f_{(t+1)}^{s1} & f_{(t+2)}^{s1} & \cdot & \cdot & \cdot & f_{t+(W-1)}^{s1} \\ f_{(t+d)}^{s1} & f_{(t+d+1)}^{s1} & f_{(t+d+2)}^{s1} & \cdot & \cdot & \cdot & f_{t+(W-1)+d}^{s1} \\ \vdots & \vdots & \vdots & \cdot & \cdot & \cdot & \vdots \\ \vdots & \vdots & \vdots & \cdot & \cdot & \cdot & \vdots \end{bmatrix} \tag{4.1}$$

$$Y_h^{s1} = \begin{bmatrix} f_{t+(W-1)+1}^{s1} & f_{t+(W-1)+2}^{s1} & f_{t+(W-1)+3}^{s1} & \cdot\ \cdot\ \cdot & f_{t+(W-1)+h}^{s1} \\ f_{t+(W-1)+2}^{s1} & f_{t+(W-1)+3}^{s1} & f_{t+(W-1)+4}^{s1} & \cdot\ \cdot\ \cdot & f_{t+(W-1)+h+1}^{s1} \\ \vdots & \vdots & \vdots & . \ . \ . & \vdots \\ \vdots & \vdots & \vdots & . \ . \ . & \vdots \end{bmatrix} \tag{4.2}$$

In equation 4.1 and 4.2, f_t^{s1} indicates the traffic flow of station 1 at time t. h represents prediction horizon, W denotes time-lag(or time-window-size) and d is the stride value which is a parameter that determines how much of the time window we will shift. The term h in matrix X is added just to match the matrices X and Y

4.3 Experiments Settings

TensorFlow[2] and Keras[3] , which are open source libraries of Python, were used to build the proposed deep ensemble model and other deep learning models. We used the scikit-learn[4] as the machine learning library to implement the LR, KNN, DT, RF models.

We made a lot of trials to determine the best time-lag. As a result of these trials, we found that the best time-lag is 24h for all models.

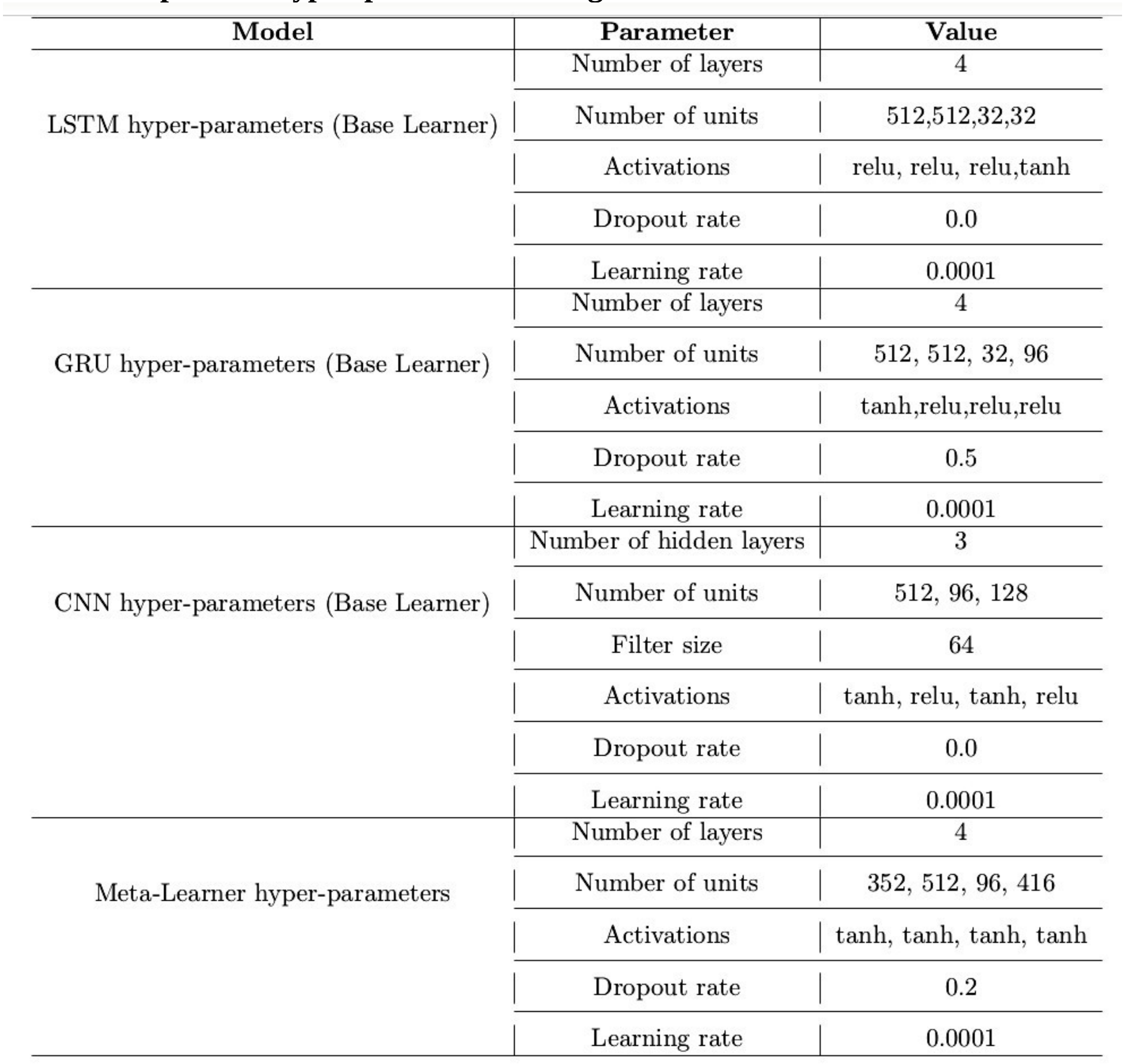
We optimized the hyper-parameters of each model separately. For deep learning models the number of hidden neurons, activation function, dropout rate and learning rate were optimized by using ‘Bayesian Search’ algorithm. Table 4.3 shows the hyper-parameter values that we obtained as a result of optimization for each deep learning model. We used ‘Random Search’ algorithm for optimizing hyper-parameters of LR, KNN, DT, RF models. The Adam algorithm is used to optimize the loss function of all deep learning models and the ensemble model. The maximum number of epochs is set to 100 however due to early stopping, there was no model that reached 100 epochs.

[2]www.tensorflow.org

[3]www.keras.io

[4]www.scikit-learn.org

Table 4.3 Optimum hyper-parameter settings.



Model	Parameter	Value
LSTM hyper-parameters (Base Learner)	Number of layers	4
	Number of units	512,512,32,32
	Activations	relu, relu, relu,tanh
	Dropout rate	0.0
	Learning rate	0.0001
GRU hyper-parameters (Base Learner)	Number of layers	4
	Number of units	512, 512, 32, 96
	Activations	tanh,relu,relu,relu
	Dropout rate	0.5
	Learning rate	0.0001
CNN hyper-parameters (Base Learner)	Number of hidden layers	3
	Number of units	512, 96, 128
	Filter size	64
	Activations	tanh, relu, tanh, relu
	Dropout rate	0.0
	Learning rate	0.0001
Meta-Learner hyper-parameters	Number of layers	4
	Number of units	352, 512, 96, 416
	Activations	tanh, tanh, tanh, tanh
	Dropout rate	0.2
	Learning rate	0.0001

4.4 Comparison Metrics

We use four metrics to measure the performance of the developed models, mean absolute error (MAE), mean squared error (MSE), Mean Squared Logarithmic Error (MSLE) and R-squared score which are the most frequently used metrics for traffic forecasting.

MAE, MSE, MSLE, $R^2$are defined as:

$$MSE = \frac{1}{T}\sum_{n=1}^{T}(t_n - p_n)^2.$$

$$MAE = \frac{1}{T}\sum_{n=1}^{T}|t_n - p_n|. \tag{4.3}$$

$$R^2 = 1 - \frac{\sum_{n=1}^{T}(t_n - p_n)^2}{\sum_{n=1}^{T}(t_n - v)^2}.$$

$$MSLE = \frac{1}{T}\sum_{n=1}^{T}(log(1 + t_n) - log(1 + p_n))^2.$$

where t, p and T indicate the actual value, prediction value and the total number of samples, respectively. And v indicates that mean value of the actual traffic flow data.

Chapter 5

Results and Discussions

In order to assess the performance of our ensemble model (i.e., Ens2 in Table 5.1 and Table 5.2), we compare the proposed model with base models, i.e., LSTM, GRU and CNN. Besides these models, we also compare our model with some selected traditional machine learning models including LR, KNN, DT, RF. We use raw input connection in meta-learner so that our ensemble model can be dynamic and capture the traffic conditions well. We construct an alternative model, which has the same architecture as the proposed ensemble model except the raw input connection (i.e., Ens1 in Table 5.1 and Table 5.2). We also compared the prediction performance of this model with the model we propose.

5.1 Comparison of the Proposed Model with Traditional Machine Learning Models

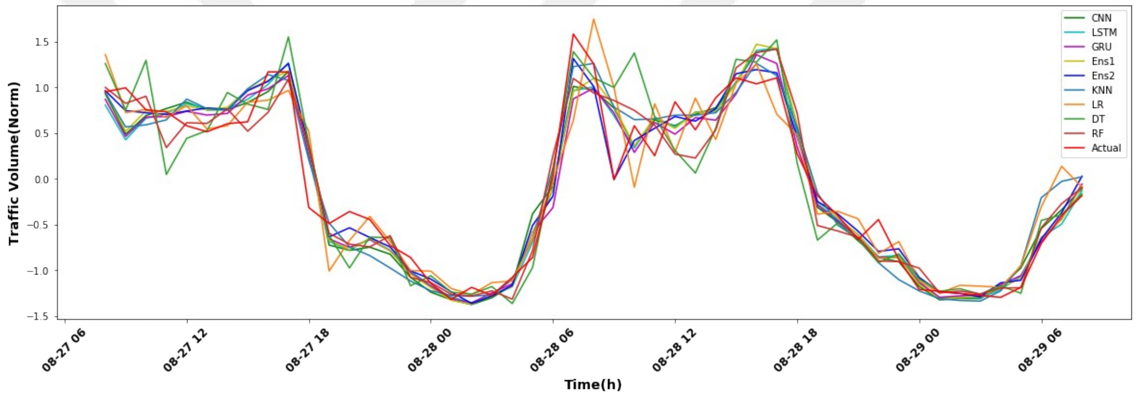
We compared our proposed model with four traditional machine learning methods. These methods are linear regression, K nearest neighbors, decision trees, and random forest. For comparison, we trained models for 8 different horizons using these ML methods. We calculated MSE and MAE values for each model.

Table 5.1 shows all the results we obtained for this study. As seen in this table, the model we propose is quite successful compared to traditional ML models. There is a significant difference between the proposed model and traditional ML models for both MSE and MAE values for all time horizons. For example, when the time horizon is 1 hour, LR has an MSE value of 0.1007, while the MSE value of the proposed model is only 0.0613. When the time horizon was 4 hours, the MAE value of the DT model was 0.3188, and the MAE value of the proposed model decreased to 0.2302. KNN, another traditional ML model, reached 0.3019 MSE value when the forecast horizon was 24 hours, but the proposed model decreased to 0.2539.

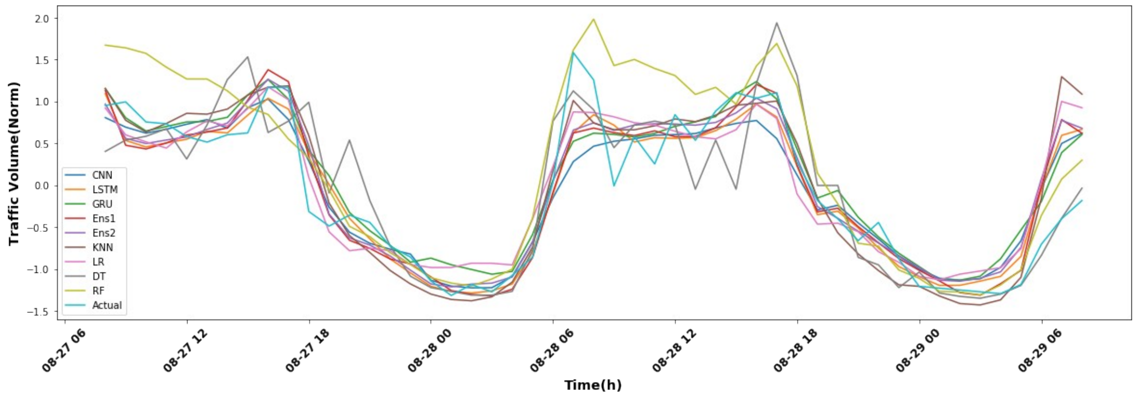
Figure 5.1 shows the 3-day forecast performance for all models. While Figure 5.1a shows single-step prediction performances, Figure 5.1b shows multi-step prediction performances. When we examine these two figures only by considering traditional ML models, we can reach the following conclusions:

Traditional machine learning models are not sufficient for especially long-term prediction. Among the traditional machine learning-based (ML-based) models we have compared, the best performance belongs to KNN. However, the performance of tree-based models is quite low. The prediction performance of Random Forest(RF), which is a tree-based ensemble model, is quite far behind KNN.

When Figure 5.1 (a) and Figure 5.1 (b) are compared, it is seen that the proposed model is relatively more successful in sharp ups and downs.



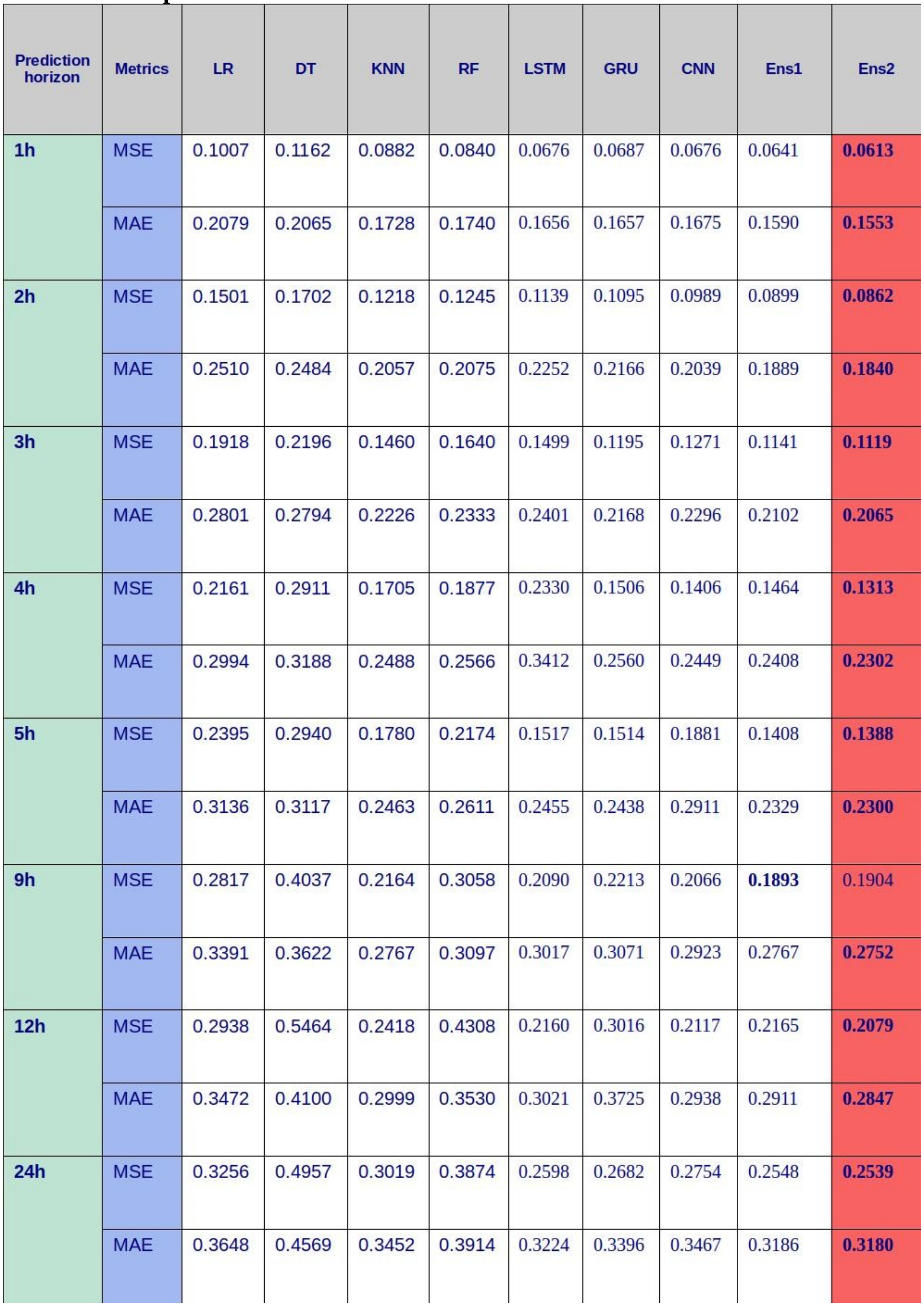
(a)



(b)

Figure 5.1 Comparison of the prediction results: (a) TimeHorizon=1(Single-step prediction). (b) TimeHorizon=24 (Multi-stepprediction).

Table 5.1 Comparison of prediction performances of the proposed ensemble model and other competitive models for 8 time horizons.



Prediction horizon	Metrics	LR	DT	KNN	RF	LSTM	GRU	CNN	Ens1	Ens2
1h	MSE	0.1007	0.1162	0.0882	0.0840	0.0676	0.0687	0.0676	0.0641	**0.0613**
	MAE	0.2079	0.2065	0.1728	0.1740	0.1656	0.1657	0.1675	0.1590	**0.1553**
2h	MSE	0.1501	0.1702	0.1218	0.1245	0.1139	0.1095	0.0989	0.0899	**0.0862**
	MAE	0.2510	0.2484	0.2057	0.2075	0.2252	0.2166	0.2039	0.1889	**0.1840**
3h	MSE	0.1918	0.2196	0.1460	0.1640	0.1499	0.1195	0.1271	0.1141	**0.1119**
	MAE	0.2801	0.2794	0.2226	0.2333	0.2401	0.2168	0.2296	0.2102	**0.2065**
4h	MSE	0.2161	0.2911	0.1705	0.1877	0.2330	0.1506	0.1406	0.1464	**0.1313**
	MAE	0.2994	0.3188	0.2488	0.2566	0.3412	0.2560	0.2449	0.2408	**0.2302**
5h	MSE	0.2395	0.2940	0.1780	0.2174	0.1517	0.1514	0.1881	0.1408	**0.1388**
	MAE	0.3136	0.3117	0.2463	0.2611	0.2455	0.2438	0.2911	0.2329	**0.2300**
9h	MSE	0.2817	0.4037	0.2164	0.3058	0.2090	0.2213	0.2066	**0.1893**	0.1904
	MAE	0.3391	0.3622	0.2767	0.3097	0.3017	0.3071	0.2923	0.2767	**0.2752**
12h	MSE	0.2938	0.5464	0.2418	0.4308	0.2160	0.3016	0.2117	0.2165	**0.2079**
	MAE	0.3472	0.4100	0.2999	0.3530	0.3021	0.3725	0.2938	0.2911	**0.2847**
24h	MSE	0.3256	0.4957	0.3019	0.3874	0.2598	0.2682	0.2754	0.2548	**0.2539**
	MAE	0.3648	0.4569	0.3452	0.3914	0.3224	0.3396	0.3467	0.3186	**0.3180**

5.2 Comparison of the Proposed Model with Base-Learner Models

Table 5.1 shows MSE and MAE results we have obtained as a result of our experiments for eight time horizons. In addition to the results of the ensemble model we proposed (which appears as Ens2 in the table), we also added the results of the base learners that make up our ensemble model. In the table, it can be seen that the ensemble model we proposed performs better than base learners. This result is valid for all time horizons in the table. For example, while the MSE value of LSTM for time horizon 3 hours is 0.1499, this value is 0.1119 for Ens2. When the time horizon is 5 hours, the MAE value of CNN is 0.2911 and the Ens2 value is 0.2300. When the time horizon is 12 hours, the MAE value of GRU is 0.3725, while this value is 0.2847 for Ens2. As can be seen from these results, the proposed model is quite successful compared to base learners.

When we evaluate the results in the table only from the perspective of base-learners, we can reach the following conclusions: According to this, among the base-learner models, CNN has shown the best performance in many horizons. This result is interesting because CNN was not originally developed for time-series problems. But before we can generalize this result, we need to do more experiments. For the dataset we used in this study, CNN performs quite successfully. In other words, we can say that we have developed a prediction model compatible with the dataset. Nevertheless, we cannot make a general conclusion that CNN is the most successful DL technique for time series problems. However, we can say that CNN is promising for such problems.

According to the results of our experiments, although CNN is more successful DL model than others, the performance of the other DL models is roughly competitive with CNN. These results indicate that DL-based models offer the opportunity to develop more successful prediction models because they can better capture long-term dependencies.

The 3-day forecasting performance of the base models and the ensemble models are shown in Figure 5.2 (a) and (b). Figure 5.2 (a) shows single-step prediction (i.e., prediction-horizon=1) and Figure 5.2 (b) shows multi-step predictions (i.e., prediction-horizon=24). The forecasting graphs belonging to the same period (month/day/hour) were selected in order to compare the performance of the top and bottom forecasting horizons. Based on these figures, the predictive success of the proposed deep ensemble model has increased considerably for both time horizons compared to single models.

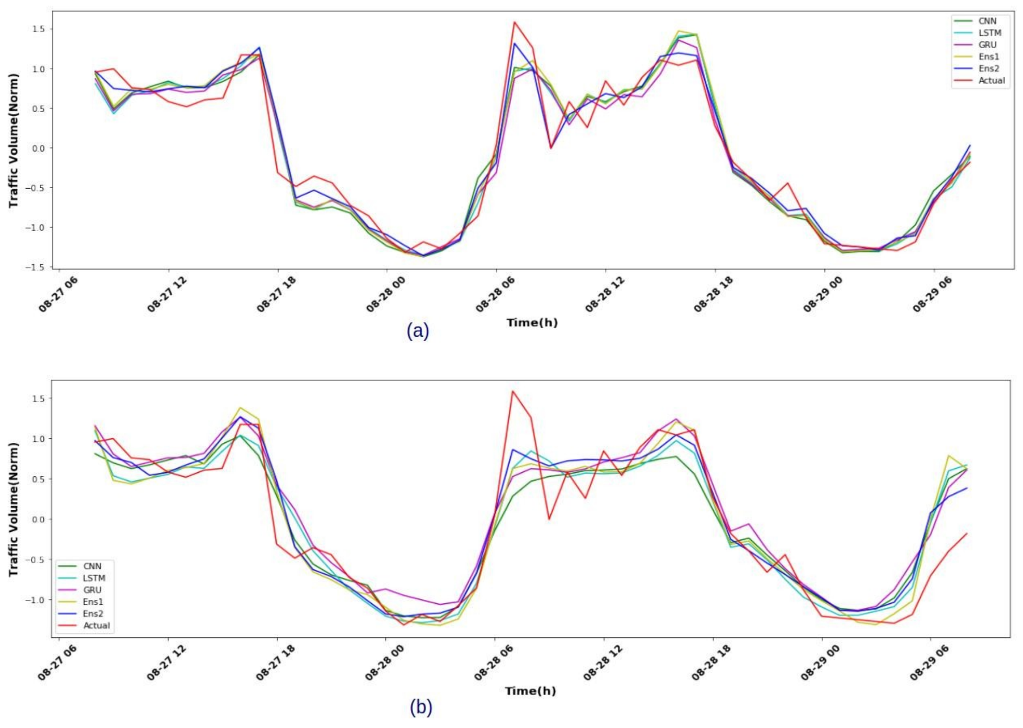


Figure 5.2 Comparison of the prediction results of ensemble models and single models: (a) TimeHorizon=1(Single-step prediction). (b) TimeHorizon=24 (Multi-stepprediction).

The histograms in Figure 5.3 (a)-(e) show forecasting performances for time horizon=1 for different days of the week and different times of the day. To obtain these histograms, for each model, first, the difference between the each prediction value and its ground truth was taken separately. Then, for each prediction value, the model with the smallest of this difference was awarded a score, and in the end, the models' scores were added up. These histograms show the total score of each model. The highest score in all five histograms belongs to the proposed ensemble model. In fact, these histograms show that the proposed model is decisively ahead of the other models.

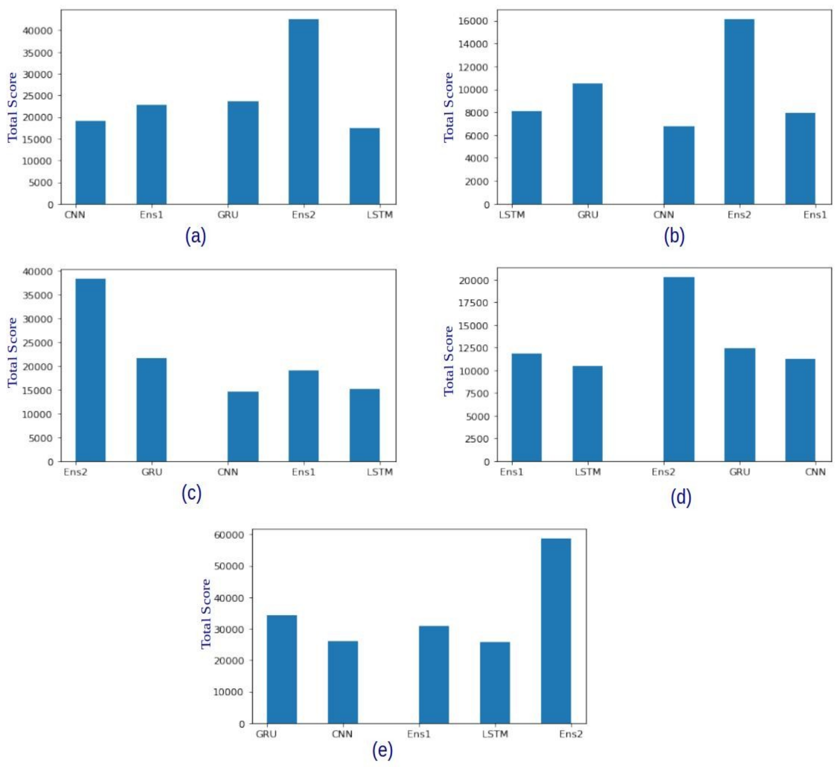


Figure 5.3 Comparison of the prediction results: (a) Prediction results on weekdays. (b) Prediction results on weekends. (c)Prediction results on rush hours. (d) Prediction results on off-peak hours. (e) Overall performance (TimeHorizon=1).

When the histograms in Figure 5.3 are examined, it is seen that the prediction performance of the proposed model is quite good both on weekdays and during peak hours. However, the performance of CNN among the single models is the lowest for these two categories.

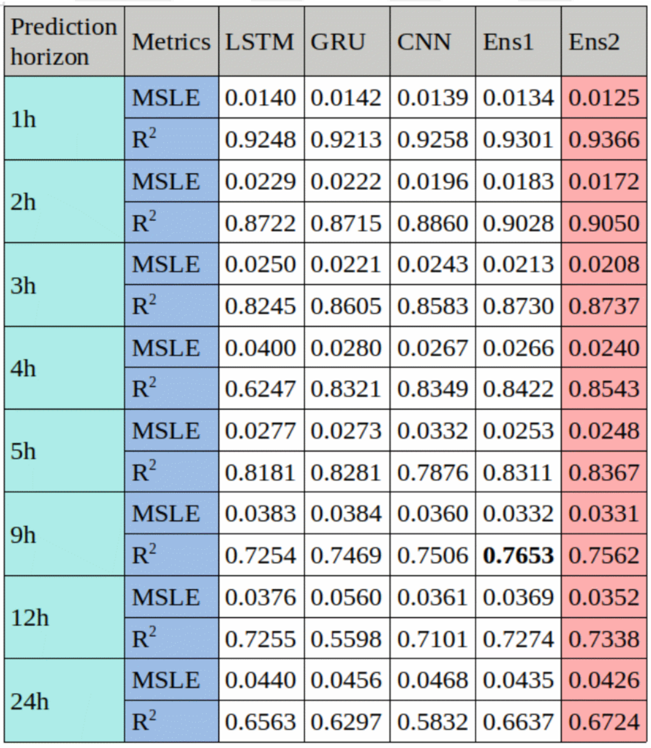
When we examine the histograms in Figure 5.3, we see that the most successful single model is GRU. GRU outperforms even our alternative ensemble model (Ens1) for all categories.

In Table 5.2, we compare ensemble models with base models using MSLE and R^2 metrics. We chose these two metrics because the first metric measures relative error rather than actual error. That is, it gives approximately equal weight to small and large differences between actual and predicted values. The second metric is used to compare the quality of models with each other, rather than to decide the overall quality of a model. This metric takes a value between 0-1. and the model is considered good as the value get closer to 1. However, it is very difficult to obtain values close to 1 for difficult problems such as traffic flow prediction. Therefore, each prediction problem should be evaluated on its own.

In general, as prediction horizon increases, the prediction performance of all the models we compare, including the model we propose, decreases, which proves that the long-term prediction is more difficult.

Although the prediction performance of our proposed model decreases as the prediction horizon increases, this decrease is small compared to the other models we compared. For example, when the forecast horizon is 4 hours, the MSE of our model is 0.1313, and this value increases to 0.1388 when the forecast horizon is 5 hours. In contrast, when the forecast horizon is 4 hours, the MSE of CNN is 0.1406. However, when the forecast horizon increases to 5 hours, this value increases to 0.1881

Table 5.2 Comparison of prediction performances of the proposed ensemble models and base models for 8 time horizons.



Prediction horizon	Metrics	LSTM	GRU	CNN	Ens1	Ens2
1h	MSLE	0.0140	0.0142	0.0139	0.0134	0.0125
	R^2	0.9248	0.9213	0.9258	0.9301	0.9366
2h	MSLE	0.0229	0.0222	0.0196	0.0183	0.0172
	R^2	0.8722	0.8715	0.8860	0.9028	0.9050
3h	MSLE	0.0250	0.0221	0.0243	0.0213	0.0208
	R^2	0.8245	0.8605	0.8583	0.8730	0.8737
4h	MSLE	0.0400	0.0280	0.0267	0.0266	0.0240
	R^2	0.6247	0.8321	0.8349	0.8422	0.8543
5h	MSLE	0.0277	0.0273	0.0332	0.0253	0.0248
	R^2	0.8181	0.8281	0.7876	0.8311	0.8367
9h	MSLE	0.0383	0.0384	0.0360	0.0332	0.0331
	R^2	0.7254	0.7469	0.7506	**0.7653**	0.7562
12h	MSLE	0.0376	0.0560	0.0361	0.0369	0.0352
	R^2	0.7255	0.5598	0.7101	0.7274	0.7338
24h	MSLE	0.0440	0.0456	0.0468	0.0435	0.0426
	R^2	0.6563	0.6297	0.5832	0.6637	0.6724

Based on our observations during our experiments, we can also make a comparison between DL-based models in terms of computation times. CNN also performs best in terms of computation time among DL-based models. This is probably due to the fact that CNN can be parallelized more efficiently than GRU and LSTM. However, LSTM was the model with the worst performance in terms of computation time.

5.3 Comparison of the Proposed Model with and without Raw Input Connection (Ens1 vs Ens2)

As we explained in Section 3, we propose another ensemble model that can be an alternative to the model we proposed. To compare these two models, we repeated all experiments for the alternative model. With these experiments, we tried to understand how much the raw input connection improves our model. This alternative model is called as Ens1 and the proposed model is called as Ens2.

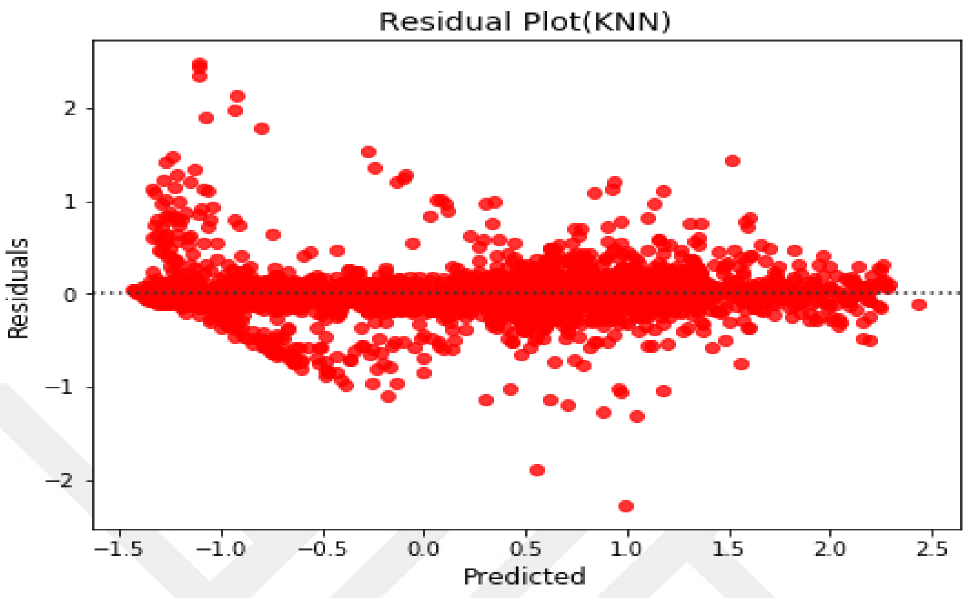
Table 5.1 shows MSE and MAE results we have obtained as a result of our experiments for eight time horizons. As can be seen in the table, our model(i.e., Ens2) is the most successful model in all time horizons except time horizon=9, in which Ens1 model was the most successful. This result shows how much the raw input connection improves our model. Morover these results prove that the ensemble models perform better, especially in long-term traffic flow prediction.

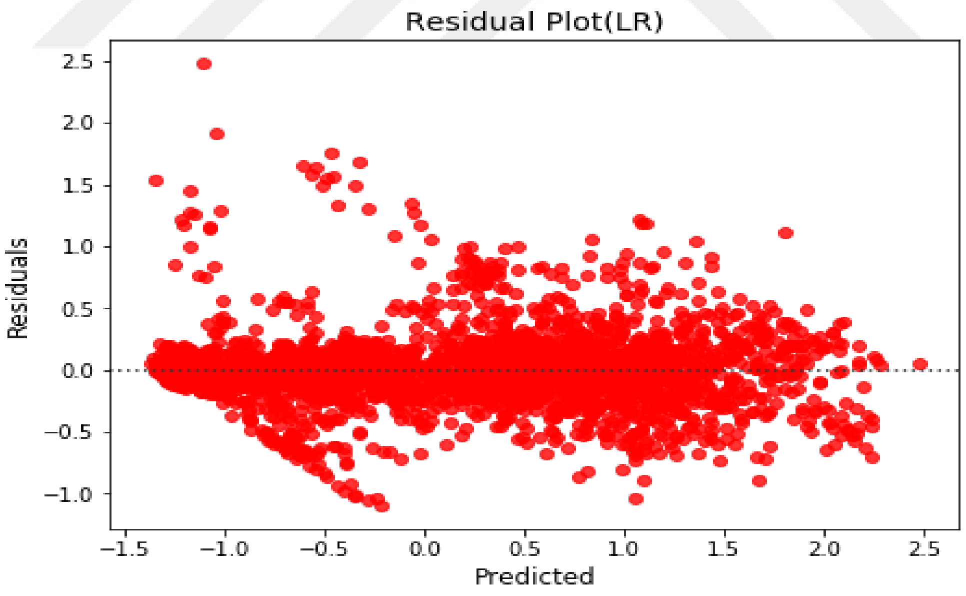
In addition to these main results, when we examine both metrics values in Table 5.2, we see that the model we recommend is the best model for all prediction horizons except 9h. This shows us the results in Table 5.2 are consistent with the results in Table 5.1. The results show that we can achieve a significant performance improvement when we combine ensemble learning architecture and deep learning techniques with raw input connection.

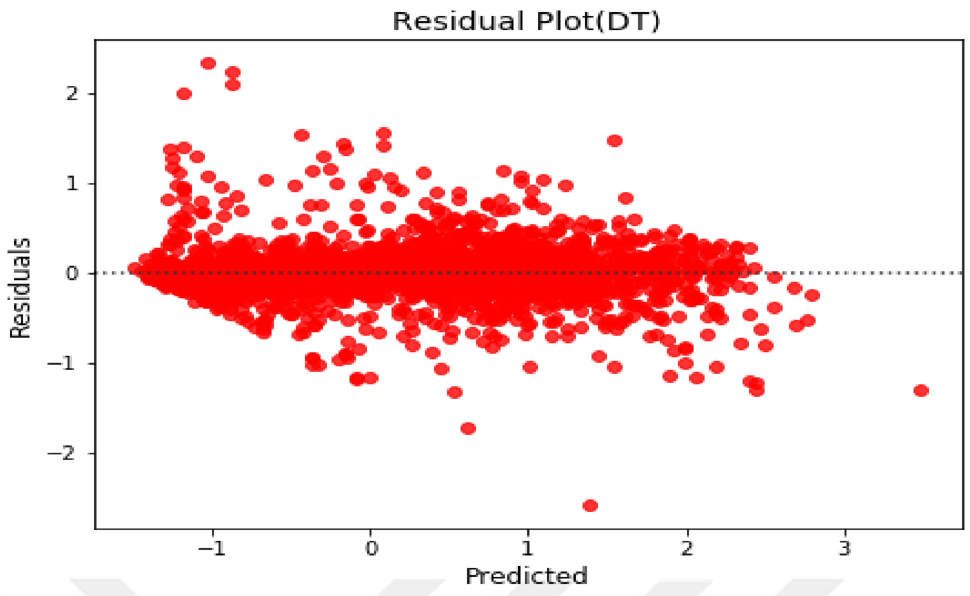
When we examine the results in Table 5.1 in detail, we see that the error differences between the two models are not large. However, we can say that there is a slight improvement. To give a few examples of this, for example, for time horizon 2 hours, the Ens1 model has 0.0899 MSE and 0.1889 MAE values, while for Ens2 these values are 0.0862 and 0.1840, respectively. While the Ens1 model has 0.2165 MSE and 0.2911 MAE values for the time horizon 12 hours, these values for Ens2 are 0.2079 and 0.2847, respectively.

5.4 Comparison of all Models via Residual Plots

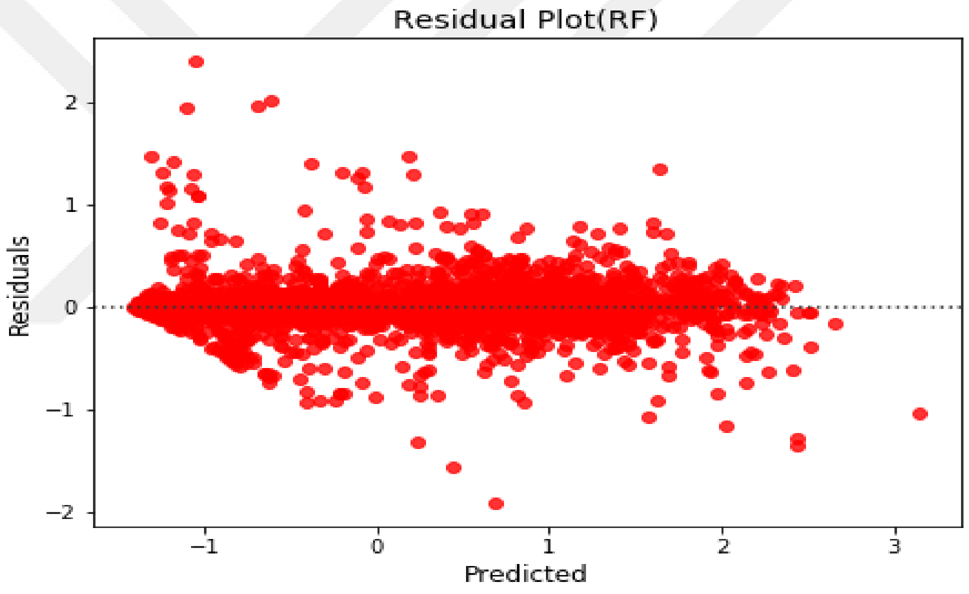
Figure 5.4 shows the residual plots of our proposed model and the models we compared The residual plot shows the difference between actual values and predicted values. The closer the points forming the graph are to the starting point, the better the developed model. When these graphs are examined in detail, the superiority of our proposed model over other models is clearly seen.



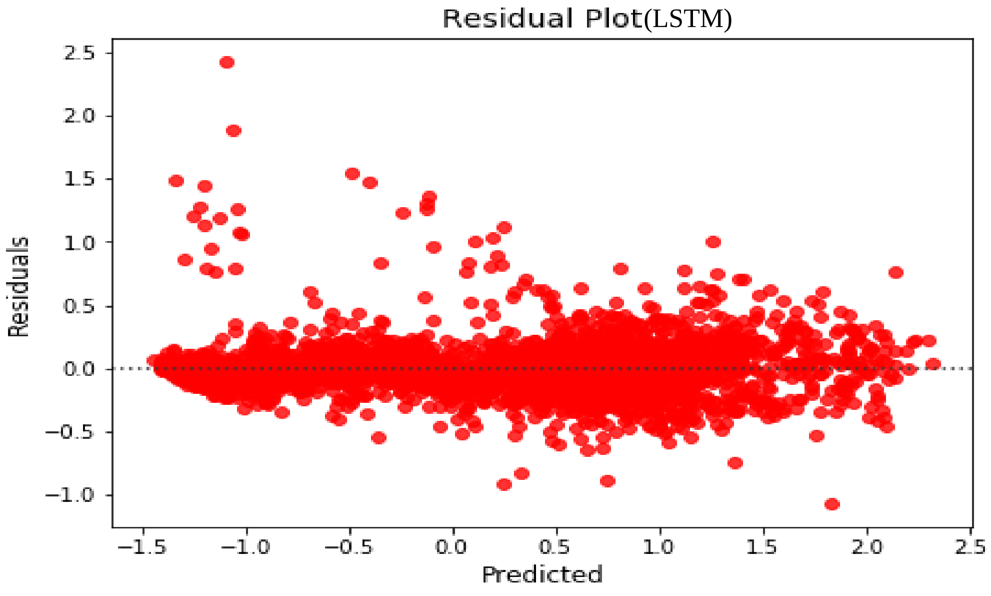




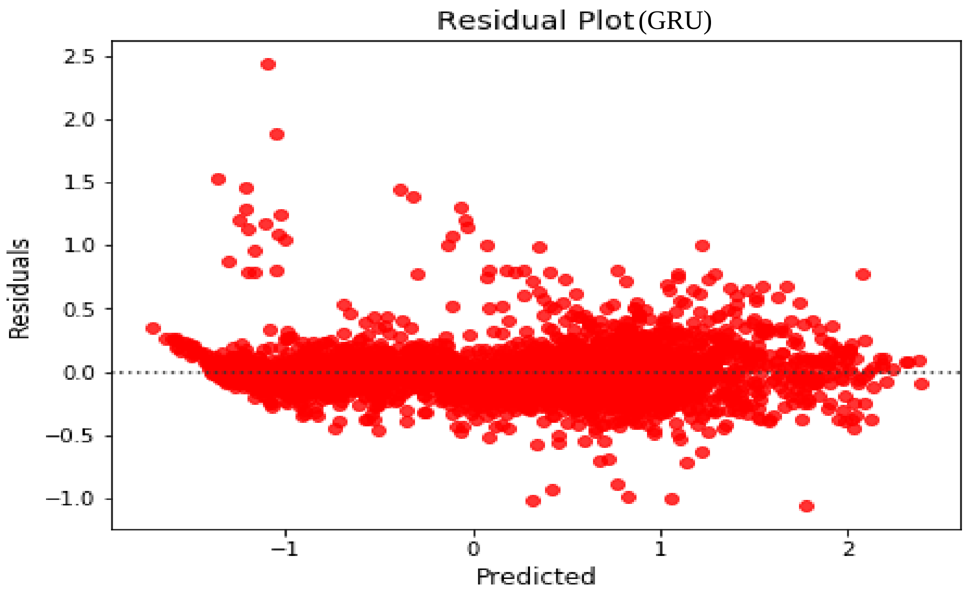
Residual Plot(DT)
Residuals
Predicted
2
1
0
−1
−2
−1
0
1
2
3



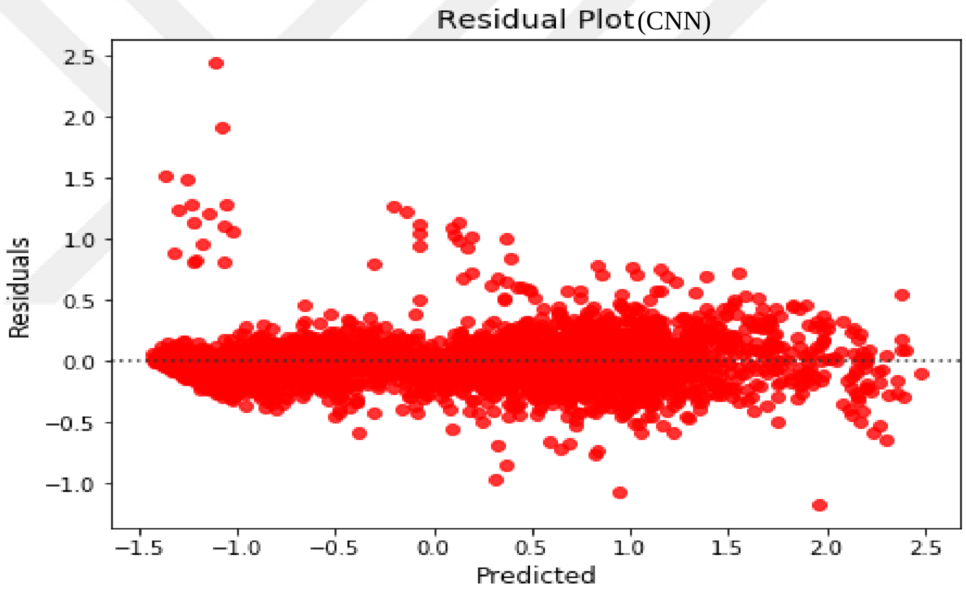
Residual Plot(RF)
Residuals
Predicted
2
1
0
−1
−2
−1
0
1
2
3



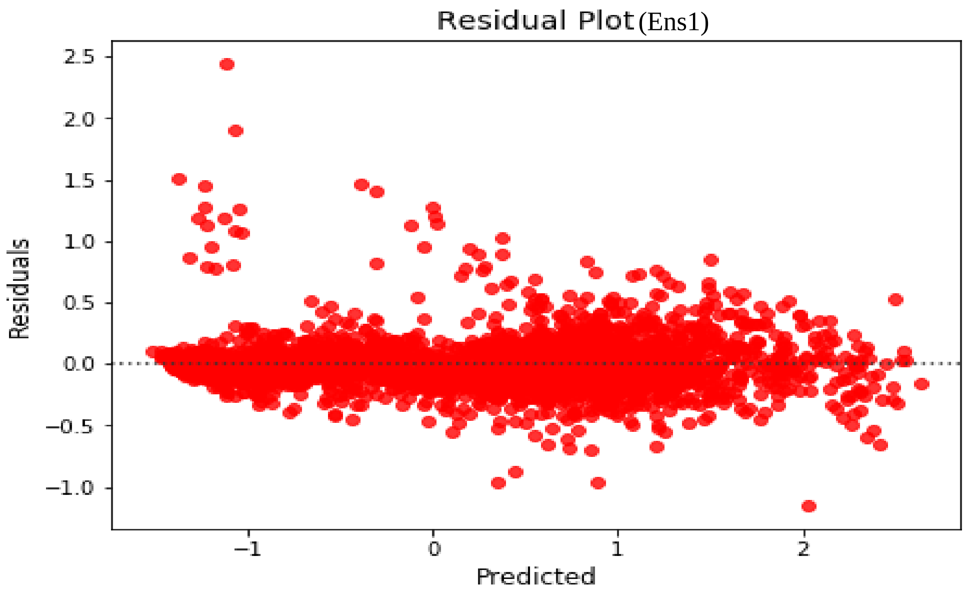
Residual Plot(LSTM)
Residuals
Predicted
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5



Residual Plot(GRU)
Residuals
Predicted
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1
0
1
2



Residual Plot(CNN)
Residuals
Predicted
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5



Residual Plot(Ens1)
Residuals
Predicted
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1
0
1
2

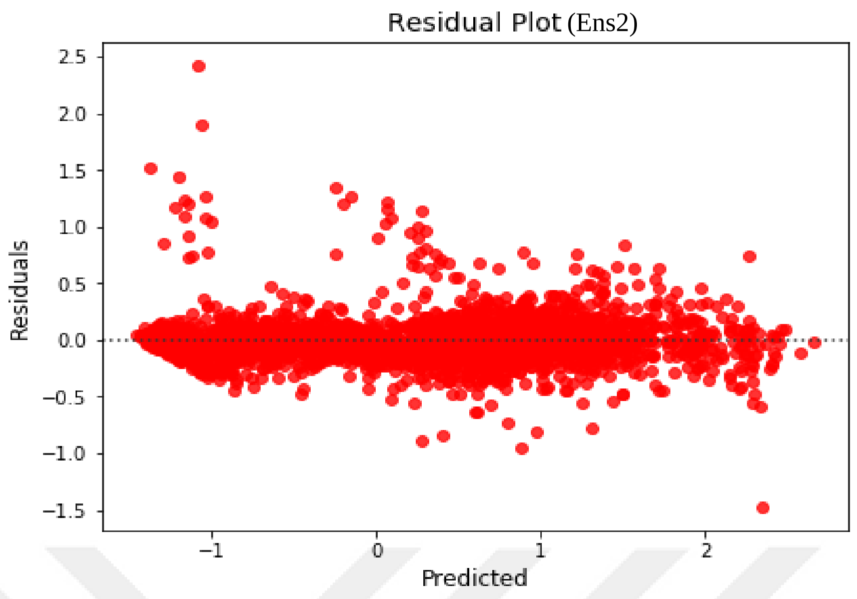


Figure 5.4 Residual Plots

5.5 Two-tailed Z-test

We want to see whether the improvements obtained by the proposed ensemble model over the base-learners are statistically significant or not. To this end, we perform two-tailed Z-test between the proposed model and the best base-learner model for each time horizon. Table 5.3 shows the p-values that are computed during the Z-test experiment that compares our deep ensemble model and the best base-learner model. According to two-tailed Z-test results for all time horizon, the performance improvements of our deep ensemble model are statistically significant because all p-values less than 0.05.

Table 5.3 Two-tailed Z-test results for the best base-learner and the proposed model for each time horizon

Time Horizon	Base Model	p-Value
1	CNN	7.483790763285578e-21
2	CNN	3.0300077486715994e-115
3	GRU	0.0

4	CNN	3.4729959685477615e-159
5	GRU	4.0339859311062113e-134
9	CNN	8.987073830610557e-08
12	LSTM	0.00021070979763453582
24	LSTM	7.930674522337066e-217

5.6 Experiments to Find Optimum Time-Lag

In many studies, it has been stated that the time-lag value is one of the important parameters affecting the performance of the forecast model in time series problems and it is emphasized that it should be optimized. For this reason, we optimized the time window parameters of the LSTM, GRU, CNN models, which we used as the basic learner in the ensemble model we proposed in this study, separately for each time horizon. For this, we set 5 time window values: 5, 10, 15, 20, 24. Using these time windows, we conducted 5 experiments for each base learner and each time horizon. That is, we reconstructed the traffic flow matrix for training, validation and test sets according to the time window value each time. Using these data sets, we trained 5 separate LSTM, 5 separate GRU and 5 separate CNN models for each time horizon. So, since we built 5 separate prediction models for each time horizon, we trained 8 * 5 = 40 different prediction models for a base learner. Since we trained these 40 prediction models separately for 3 base learners, we developed a total of 120 different prediction models. we compared the performance of the forecast models separately for each time horizon.

We used MSE and MAE metrics to compare. We determined the time window value of the forecast model that give the lowest MSE and MAE values as the best time window value.

We present the results of all the experiments we conducted in Table 5.4. According to these experiment results, the best time window value for all time horizon values is 24, as seen in Table 5.4. And this value is the same for 3 base learners. In other words, as a result of our experiments, we found the best time window value to be 24 for the LSTM, CNN and GRU deep learning models that we use as the base learner in our proposed model and for all time horizon values

Table 5.4 Optimum Time-Lag for each time horizon.

Horizon	Time-Lag	Metrics	CNN	LSTM	GRU
1	5	MSE	0.0983	0.0964	0.0993
		MAE	0.2050	0.2059	0.2100
	10	MSE	0.1098	0.0932	0.0913
		MAE	0.2266	0.2011	0.1993
	15	MSE	0.0849	0.0858	0.0940
		MAE	0.1905	0.1928	0.2024
	20	MSE	0.1008	0.0885	0.0897
		MAE	0.2175	0.1972	0.1972
	24	MSE	0.0676	0.0676	0.0687
		MAE	0.1675	0.1656	0.1657
2	5	MSE	0.1852	0.1681	0.1756
		MAE	0.2967	0.2805	0.2836
	10	MSE	0.1792	0.1426	0.1448
		MAE	0.2933	0.2560	0.2628
	15	MSE	0.1551	0.1379	0.1328
		MAE	0.2761	0.2471	0.2364
	20	MSE	0.1210	0.1351	0.1358
		MAE	0.2252	0.2426	0.2403

	24	MSE	0.0989	0.1139	0.1095
		MAE	0.2039	0.2252	0.2166
3	5	MSE	0.2160	0.2230	1.0323
		MAE	0.3153	0.3176	0.8726
	10	MSE	0.1588	0.1606	0.1559
		MAE	0.2628	0.2675	0.2607
	15	MSE	0.2018	0.1600	0.1570
		MAE	0.3272	0.2660	0.2620
	20	MSE	0.2113	0.2097	0.1614
		MAE	0.3435	0.3214	0.2677
	24	MSE	0.1271	0.1499	0.1195
		MAE	0.2296	0.2401	0.2168
4	5	MSE	0.2660	0.2966	0.3001
		MAE	0.3514	0.3879	0.3828
	10	MSE	0.2985	0.2636	0.2725
		MAE	0.3952	0.3571	0.3713
	15	MSE	0.2233	0.2402	0.1950
		MAE	0.3259	0.3537	0.2981
	20	MSE	0.2411	0.2466	0.1741
		MAE	0.3441	0.3492	0.2866
	24	MSE	0.1406	0.2330	0.1506
		MAE	0.2449	0.3412	0.2560

5	5	MSE	0.3153	0.2875	0.2793
		MAE	0.3940	0.3691	0.3616
	10	MSE	0.2404	0.2052	0.2125
		MAE	0.3537	0.3097	0.3127
	15	MSE	0.2315	1.0000	0.2072
		MAE	0.3190	0.8685	0.3068
	20	MSE	0.1888	0.1968	0.1912
		MAE	0.2991	0.2910	0.2944
	24	MSE	0.1881	0.1517	0.1515
		MAE	0.2911	0.2455	0.2438
9	5	MSE	0.3412	0.3865	0.3464
		MAE	0.4097	0.4399	0.4196
	10	MSE	0.2863	0.2562	0.2524
		MAE	0.3647	0.3413	0.3479
	15	MSE	0.2510	0.2399	0.2362
		MAE	0.3443	0.3272	0.3265
	20	MSE	0.2196	0.2148	0.2582
		MAE	0.3033	0.3046	0.3441
	24	MSE	0.2066	0.2090	0.2214
		MAE	0.2923	0.3018	0.3071
	5	MSE	0.3583	0.3407	0.3465
		MAE	0.4149	0.4112	0.4146

12	10	MSE	0.2866	0.2760	0.2865
		MAE	0.3626	0.3542	0.3690
	15	MSE	0.2860	0.2900	0.3249
		MAE	0.3649	0.3728	0.3822
	20	MSE	0.2781	0.2712	0.3492
		MAE	0.3546	0.3466	0.4
	24	MSE	0.2117	0.2160	0.3016
		MAE	0.2938	0.3021	0.3725
24	5	MSE	0.4001	0.3940	0.4411
		MAE	0.4615	0.4302	0.4678
	10	MSE	0.3191	0.3131	0.3392
		MAE	0.3900	0.3842	0.3971
	15	MSE	0.2955	0.2921	0.2860
		MAE	0.3715	0.3593	0.3561
	20	MSE	0.2840	0.2898	0.3298
		MAE	0.3566	0.3659	0.3948
	24	MSE	0.2754	0.2598	0.2683
		MAE	0.3467	0.3224	0.3396

5.7 Comparison of Two Dataset Splitting Approaches

As we emphasized before, the continuity of the data set is very important for time series problems. However, in order to create a prediction model, dividing the existing data into train, validation and test sets is a commonly used strategy. However, unlike data sets of other problems, time series data sets have features (such as seasonality) that can significantly affect forecasting performance. For this reason, we can say that forecasting models developed using classical data set division approaches for time series problems have a high risk of being underfit or overfit.

For this reason, we think that new data set partitioning approaches should be developed for time series, which can be an alternative to classical methods. For this purpose, in this study, we carried out a series of experiments using two different data set partitioning methods for time series.

The first method is a very simple method. We call this method as Split1. In this approach we divide the entire data set as 60% train, 20% validation, 20% test set

Since continuity is important in time series datasets and in order not to disrupt this continuity when dividing the dataset, the second splitting method follows this approach: Instead of dividing the entire data set as 60% train, 20% validation, 20% test set, we divide each month in the dataset as 60% train, 20% validation and 20% test set (We named this alternative method Split2). Thus, it contains data from 12 months in certain proportions in 3 sub-datasets (i.e. train, validation, test).

We conducted 5 separate experiments to measure the performance of this method. For each experiment, we selected one station with different road types from the total data set. First of all, we divided each station separately into train, validation and test sets using the Split2. And we trained 3 base deep learning models (i.e. LSTM, CNN, GRU) using this dataset. We did this process separately for each station. We then conducted a second experiment using data sets from the same 5 stations. Using the entire dataset of each station, we divided it into 60% train, 20% validation, and 20% test set, and using these datasets, we trained 3 base deep learning models (i.e. LSTM, CNN, GRU) separately for each station. As a result, we compared the results of these two experiments with each other. We used MSE and MAE metrics to compare. We present the comparison results in Table 5.5.

As seen in Table 5.5, we used datasets from 5 different road types for the experiments. Three of these belong to the “Urban” road category and two belong to the “Rural” road category. The MSE and MAE values we obtained as a result of the experiments are presented separately in the table. When we examine these values, we can see that the Split1 method produces better results. However the alternative method, that is Split2, produces more erroneous predictions. Indeed, when we look at the error values, we can see that the difference between the two methods is quite large. The results are the same for all 5 stations. For example, while the CNN model of the Split2 method for station “110177” has an MSE value of 0.2364, for Split1 this value is only 0.0950. For another station (50272), the CNN model of the Split2 method has an MAE value of 0.4964, while for Split1 this value is only 0.2926.

The reasons why the performance of the Split2 method is so bad may be as follows: Since we divide the months in the dataset into 3 as train, test and validation, we actually use the future data to predict the past data when training the model. This may cause erroneous results. Because in time series problems, normally future data can be predicted using past data.

We think these results are interesting and do not mean that the alternative method is useless, but we can say that more experiments are needed to find the underlying reason for these results.

Table 5.5 Comparison of two dataset splitting approaches

Station_id	Road_Type	Split_Type	Metrics	CNN	LSTM	GRU
110177	Urban: Principal Arterial - Other	Split2	MSE	0.2364	0.2430	0.2167
			MAE	0.3063	0.3576	0.3216
		Split1	MSE	0.0950	0.1353	0.1224
			MAE	0.2271	0.2788	0.2527
970407	Urban: Principal Arterial - Other Freeways or Expressways	Split2	MSE	0.3114	0.3341	0.2902
			MAE	0.3242	0.3832	0.3316
		Split1	MSE	0.1020	0.1870	0.2085
			MAE	0.1973	0.3001	0.2825
30191	Urban: Principal Arterial - Interstate	Split2	MSE	0.3196	0.3198	0.3236
			MAE	0.3693	0.3848	0.3704

		Split1	MSE	0.1191	0.2320	0.2516
			MAE	0.2320	0.3669	0.3676
550349	Rural: Principal Arterial - Other	Split2	MSE	0.2132	0.2616	0.2035
			MAE	0.3016	0.3758	0.3082
		Split1	MSE	0.0632	0.1393	0.0991
			MAE	0.1825	0.2823	0.2386
50272	Rural: Minor Arterial	Split2	MSE	0.4120	0.3646	0.3300
			MAE	0.4964	0.4543	0.4171
		Split1	MSE	0.1638	0.2101	0.2281
			MAE	0.2926	0.3609	0.3807

5.8 The Results of Model Averaging Approach

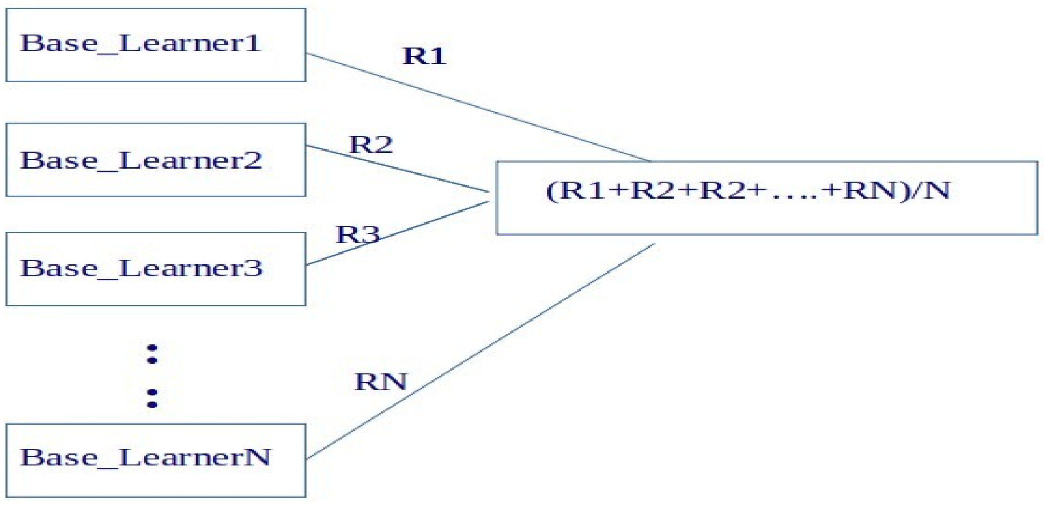


Figure 5.5 Model Averaging Approach

As we emphasized before, when developing an ensemble model, several strategies have been proposed to determine how much each base learner contributes to the final result. The simplest of these is the model averaging approach.

As can be seen in Figure 5.5, in this approach, the final prediction result is obtained by taking the arithmetic average of all base learners' results. In other words, each base learner contributes equally to the result. This approach ignores the individual performances of base learners. That is, a very successful base learner and a highly unsuccessful base learner contribute to the final prediction result at the same rate.

Our purpose in conducting this experiment is to understand the contribution of the meta-learner we developed to the performance of our model. For this, we repeated all prediction experiments with the model averaging approach. We present our experimental results in Table 5.6. As seen in this table, we calculated MSE and MAE values for all time horizons. When we compare these results with the results of our proposed model (Ens2) and our alternative model (Ens1), we can say that the model we proposed is more successful for all time horizons.

Table 5.6 Results of Model Averaging

Time Horizon	Metrics	Model Averaging	Ens1	Ens2
1 Hour	MSE	0.0662	0.0641	**0.0613**
	MAE	0.1632	0.1590	**0.1553**
2 Hours	MSE	0.0956	0.0899	**0.0862**
	MAE	0.2031	0.1889	**0.1840**
3 Hours	MSE	0.1307	0.1141	**0.1119**
	MAE	0.2274	0.2102	**0.2065**
4 Hours	MSE	0.1595	0.1464	**0.1313**

	MAE	0.2600	0.2408	**0.2302**
5 Hours	MSE	0.1643	0.1408	**0.1388**
	MAE	0.2537	0.2329	**0.2300**
9 Hours	MSE	0.1996	**0.1893**	0.1904
	MAE	0.2793	0.2767	**0.2752**
12 Hours	MSE	0.2264	0.2165	**0.2079**
	MAE	0.3018	0.2911	**0.2847**
24 Hours	MSE	0.2633	0.2548	**0.2539**
	MAE	0.3247	0.3186	**0.3180**

Chapter 6

Conclusions and Future Prospects

6.1 Conclusions

Long-term traffic flow forecasting is essential for traffic management issues such as congestion control and better route selection. This importance will become more evident in the future with the development of related technologies.

However, compared to studies on short-term traffic flow forecasting, there are few studies in the literature on long-term traffic flow forecasting. In addition, the prediction performances of existing studies are not sufficient.

Therefore, it is critical to try to improve long-term traffic flow forecasting performance. That's why, this study proposed a novel ensemble model for long-term traffic flow prediction. The proposed model is a deep ensemble model built by properly combining 3 different deep learning techniques as base models. We designed our model that can dynamically produce the weights of the base models based on both each base model's performance and traffic condition. Experimental results show that the proposed approach outperforms all the models compared.

The main purpose of this thesis is to develop an effective deep learning-based prediction model for the traffic flow prediction problem, and this study has achieved its purpose. In addition, since our model uses basic deep learning methods as the base learner, we compared these basic deep learning methods among themselves as a result of our experiments. As a result of these comparisons, we reached potentially interesting results.

In our opinion, the most important of these is that CNN is more successful than other basic deep learning models in terms of both calculation speed and prediction accuracy. This result is interesting because it is the opposite of what was expected. Because CNN is not a method developed for time series forecasting.

On the other hand, LSTM and GRU are methods developed for time series. However, we think that this result largely depends on the dataset we used. Because there are other studies in the literature that compare these methods using different datasets, but none of them reached a conclusion that supports our results. We think this is due to the different data set we use.

Another reason for the lower performance of LSTM and GRU may be overfitting or underfitting. The number of data may have been relatively small for these models and may have caused the models not to reach the optimum. However, if this is true, we can say that CNN can produce more successful predictions even with less data than GRU and LSTM.

6.2 Societal Impact and Contribution to Global Sustainability

It is of vital importance for all stakeholders in different areas to be able to know traffic flow information in advance. Especially the rapid development in smart city and smart traffic applications has enabled the traffic flow prediction problem to become a critical element in research. If we consider the social, economic and environmental effects of traffic congestion, smart systems in which traffic flow prediction applications will be integrated will minimize traffic congestion and make transportation planning and management more efficient.

The main goal of traffic forecasting applications is to reduce the time spent in traffic due to traffic congestion. Because as the time spent in traffic increases, all problems increase exponentially. We can collect these problems under the following 3 main headings:

Environmental Problems: As time spent in traffic increases, the amount of fuel consumed will increase. This can cause serious problems, especially air pollution, with increased carbon emissions in the long run. These problems, which we know as climate change and whose future effects we cannot even predict, arise as a result of the increase in human carbon footprint in nature, and as a result, it is predicted that the world may experience major disasters in the near future. This shows us how important it is for new generation electric vehicles to become widespread.

Economic Problems: We stated that fuel consumption will increase as time spent in traffic increases. We would like to emphasize that this has not only environmental but also economic consequences. This means that natural resources cannot be used efficiently as fuel costs increase. We can understand how serious this problem is, especially when we consider that the energy used by transportation vehicles today is not renewable. Studies carried out to use renewable energy resources in this field are very valuable in this respect. The natural resources in the world are not inexhaustible. Therefore, we would like to emphasize that in terms of global sustainability, even a traffic application that recommends shorter and more efficient routes to drivers is more than a simple navigation tool.

Social Problems: Especially in big cities, the traffic problem has become an inextricable situation due to the rapidly increasing number of vehicles, roads whose capacity cannot be increased, and unpredictable traffic jams. Many medium-sized cities experience traffic problems at least as much as large cities due to traffic infrastructure that is not properly planned and managed.

Many social problems arise due to increased traffic, especially on certain days and hours. Both people using public transportation and people using their own vehicles complain about not being able to reach the places they want to go on time. We know that this causes many different individual problems. It is a fact that as the time spent in traffic increases, people become more angry and experience mental burnout, and therefore their productivity decreases both at work and at school. However, it is also known that with the change in the mood of people in traffic, the probability of drivers having an accident increases and more accidents occur.

6.3 Future Prospects

In future research, we plan to investigate the effectiveness of our model with using different base models and data-sets. We will also implement a 1D-CNN followed by a recurrent neural network (such as LSTM or GRU) as base learner and investigate the effect of including this network into our ensemble model. In addition, the fact that the CNN-based prediction model we developed was quite successful compared to other DL models motivated us to conduct more research in this area.

As a future work, we plan to make more experiments to compare the forecasting performance of CNN using different time series datasets. More than that, we will try to understand why CNN is performing better. We also plan to address the issue of interpretability of DL-based models. Although deep learning algorithms provide high prediction performance, the interpretability of DL-based models is very low. This is also true for our model. Therefore, as a future study, we plan to analyze the outputs of the base learners of our model separately. Thus, we will try to discover the critical hours that affect the outcome for each model.

It would also be beneficial to try to understand the temporal and spatial components to which our ensemble model gives more weight. We used only temporal features in this study, but we know that spatial information also affects prediction performance. For this reason, we plan to conduct new studies to see its contribution to the deep ensemble learning model we developed by providing the most appropriate representation of spatial information.

In addition, it has been stated in many studies that the variables we call auxiliary variables (e.g., weather, public holidays, traffic accidents, sports or concert events) increase the prediction performance. However, the data set we used in this study does not contain this information. If we can access a dataset containing this information, testing the model we developed with this dataset and measuring how much its performance has changed may be guiding for future studies.

In addition to all these, we can list the experiments and analyzes that can be done in the future to expand this study as follows:

- Model performance can be tested by using the some attributes in Table 4.1 appropriately.
- Model performance can be increased by optimizing the batch size.
- Ensemble models that use only two base learners, such as CNN+LSTM, CNN+GRU or GRU+LSTM, can be tested and performance comparisons can be made..
- During the preprocessing step, different methods can be tried and how these methods affect the model performance can be investigated.
- Different methods such as reinforcement learning can be used
- The prediction performance of the model for different road types can be analyzed.

BIBLIOGRAPHY

[1] L. Qu, W. Li, W. Li, D. Ma, and Y. Wang, “Daily long-term traffic flow forecasting based on a deep neural network,” Expert Systems with applications, vol. 121, pp. 304–312, 2019.

[2] L. N. Do, N. Taherifar, and H. L. Vu, “Survey of neural network-based models for short-term traffic state prediction,” Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 9, no. 1, p. e1285, 2019.

[3] K. Sattar, F. Chikh Oughali, K. Assi, N. Ratrout, A. Jamal, and S. Masiur Rahman, “Transparent deep machine learning framework for predicting traffic crash severity,” Neural Computing and Applications, vol. 35, no. 2, pp. 1535–1547, 2023.

[4] I. Lana, J. Del Ser, M. Velez, and E. I. Vlahogianni, “Road traffic forecasting: Recent advances and new challenges,” IEEE Intelligent Transportation Systems Magazine, vol. 10, no. 2, pp. 93–109, 2018.

[5] S. Abbar, R. Stanojevic, S. Mustafa, and M. Mokbel, “Traffic routing in the ever-changing city of doha,” Communications of the ACM, vol. 64, no. 4, pp. 67–68, 2021.

[6] E. Doğan, “Lstm training set analysis and clustering model development for short-term traffic flow prediction,” Neural Computing and Applications, vol. 33, no. 17, pp. 11 175–11 188, 2021.

[7] Z. Wang, X. Su, and Z. Ding, “Long-term traffic prediction based on lstm encoder-decoder architecture,” IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 10, pp. 6561–6571, 2020.

[8] T. Bogaerts, A. D. Masegosa, J. S. Angarita-Zapata, E. Onieva, and P. Hellinckx, “A graph cnn-lstm neural network for short and long-term traffic forecasting based on trajectory data,” Transportation Research Part C: Emerging Technologies, vol. 112, pp. 62–77, 2020.

[9] 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.

[10] A. Belhadi, Y. Djenouri, D. Djenouri, and J. C.-W. Lin, “A recurrent neural network for urban long-term traffic flow forecasting,” Applied Intelligence, vol. 50, pp. 3252–3265, 2020.

[11] H. Chen, S. Grant-Muller, L. Mussone, and F. Montgomery, “A study of hybrid neural network approaches and the effects of missing data on traffic forecasting,” Neural Computing & Applications, vol. 10, pp. 277–286, 2001.

[12] L. N. Do, H. L. Vu, B. Q. Vo, Z. Liu, and D. Phung, “An effective spatial-temporal attention based neural network for traffic flow prediction,” Transportation research part C: Emerging Technologies, vol. 108, pp. 12–28, 2019.

[13] O. Sagi and L. Rokach, “Ensemble learning: A survey,” Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 8, no. 4, p. e1249, 2018.

[14] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, “A survey on ensemble learning,” Frontiers of Computer Science, vol. 14, pp. 241–258, 2020.

[15] W.-C. Hong, Y. Dong, F. Zheng, and C.-Y. Lai, “Forecasting urban traffic flow by svr with continuous aco,” Applied Mathematical Modelling, vol. 35, no. 3, pp. 1282–1291, 2011.

[16] Y. Chen, H. Chen, P. Ye, Y. Lv, and F.-Y. Wang, “Acting as a decision maker: Traffic-condition-aware ensemble learning for traffic flow prediction,” IEEE

Transactions on Intelligent Transportation Systems, vol. 23, no. 4, pp. 3190–3200, 2020.

[17] J. Liu, N. Wu, Y. Qiao, and Z. Li, “Short-term traffic flow forecasting using e nsemble approach based on deep belief networks,” IEEE Transactions on I ntelligent Transportation Systems, vol. 23, no. 1, pp. 404–417, 2020.

[18] Y. Gu, W. Lu, X. Xu, L. Qin, Z. Shao, and H. Zhang, “An improved bayesian combination model for short-term traffic prediction with deep learning,” IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 3, pp. 1332–1342, 2019.

[19] F. Aljuaydi, B. Wiwatanapataphee, and Y. H. Wu, “Multivariate machine learning-based prediction models of freeway traffic flow under non-recurrent events,” Alexandria engineering journal, vol. 65, pp. 151–162, 2023.

[20] B. M. Williams, “Multivariate vehicular traffic flow prediction: evaluation of arimax modeling,” Transportation Research Record, vol. 1776, no. 1, pp. 194–200, 2001.

[21] S. Shahriari, M. Ghasri, S. Sisson, and T. Rashidi, “Ensemble of arima: combining parametric and bootstrapping technique for traffic flow prediction,” Transportmetrica A: Transport Science, vol. 16, no. 3, pp. 1552– 1573, 2020.

[22] I. Okutani and Y. J. Stephanedes, “Dynamic prediction of traffic volume through kalman filtering theory,” Transportation Research Part B: Methodological, vol. 18, no. 1, pp. 1–11, 1984.

[23] H. Sun, H. X. Liu, H. Xiao, R. R. He, and B. Ran, “Use of local linear regression model for short-term traffic forecasting,” Transportation Research Record, vol. 1836, no. 1, pp. 143–150, 2003.

[24] P. Cai, Y. Wang, G. Lu, P. Chen, C. Ding, and J. Sun, “A spatiotemporal correlative k-nearest neighbor model for short-term traffic multistep forecasting,” Transportation Research Part C: Emerging Technologies, vol. 62, pp. 21–34, 2016.

[25] N. G. Polson and V. O. Sokolov, “Deep learning for short-term traffic flow prediction,” Transportation Research Part C: Emerging Technologies, vol. 79, pp. 1–17, 2017.

[26] X. Dai, R. Fu, Y. Lin, L. Li, and F.-Y. Wang, “Deeptrend: A deep hierarchical neural network for traffic flow prediction,” arXiv preprint arXiv:1707.03213, 2017.

[27] D. Zhang and M. R. Kabuka, “Combining weather condition data to predict traffic flow: a gru-based deep learning approach,” IET Intelligent Transport Systems, vol. 12, no. 7, pp. 578–585, 2018.

[28] W. Zhang, Y. Yu, Y. Qi, F. Shu, and Y. Wang, “Short-term traffic flow prediction based on spatio-temporal analysis and cnn deep learning,” Transportmetrica A: Transport Science, vol. 15, no. 2, pp. 1688–1711, 2019.

[29] M. Lv, Z. Hong, L. Chen, T. Chen, T. Zhu, and S. Ji, “Temporal multi-graph convolutional network for traffic flow prediction,” IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 6, pp.3337–3348, 2020.

[30] C. Chen, K. Li, S. G. Teo, X. Zou, K. Li, and Z. Zeng, “Citywide traffic flow prediction based on multiple gated spatio-temporal convolutional neural networks,” ACM Transactions on Knowledge Discovery from Data (TKDD), vol. 14, no. 4, pp. 1–23, 2020.

[31] M. Méndez, M. G. Merayo, and M. Núñez, “Long-term traffic flow forecasting using a hybrid cnn-bilstm model,” Engineering Applications of Artificial Intelligence, vol. 121, p. 106041, 2023.

[32] X. Shi, H. Qi, Y. Shen, G. Wu, and B. Yin, "A spatial–temporal attention approach for traffic prediction," IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 8, pp. 4909–4918, 2020.

[33] Z. Chen, Z. Lu, Q. Chen, H. Zhong, Y. Zhang, J. Xue, and C. Wu, "Spatial–temporal short-term traffic flow prediction model based on dynamical-learning graph convolution mechanism," Information Sciences, vol. 611, pp. 522–539, 2022.

[34] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," Transportation Research Part C: Emerging Technologies, vol. 90, pp. 166–180, 2018.

[35] H. Zheng, F. Lin, X. Feng, and Y. Chen, "A hybrid deep learning model with attention-based conv-lstm networks for short-term traffic flow prediction," IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 11, pp. 6910–6920, 2020.

[36] S. Lu, Q. Zhang, G. Chen, and D. Seng, "A combined method for short-term traffic flow prediction based on recurrent neural network," Alexandria Engineering Journal, vol. 60, no. 1, pp. 87–94, 2021.

[37] H. Zhan, G. Gomes, X. S. Li, K. Madduri, A. Sim, and K. Wu, "Consensus ensemble system for traffic flow prediction," IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 12, pp. 3903–3914, 2018.

[38] X. Chen, H. Chen, Y. Yang, H. Wu, W. Zhang, J. Zhao, and Y. Xiong, "Traffic flow prediction by an ensemble framework with data denoising and deep learning model," Physica A: Statistical Mechanics and Its Applications, vol. 565, p. 125574, 2021.

[39] G. Zheng, W. K. Chai, V. Katos, and M. Walton, "A joint temporal-spatial ensemble model for short-term traffic prediction," Neurocomputing, vol. 457, pp. 26–39, 2021.

[40] Y. Liu, Z. Liu, H. L. Vu, and C. Lyu, "A spatio-temporal ensemble method for large-scale traffic state prediction," Computer-Aided Civil and Infrastructure Engineering, vol. 35, no. 1, pp. 26–44, 2020.

[41] L. Chen and C. P. Chen, "Ensemble learning approach for freeway short-term traffic flow prediction," in 2007 IEEE International Conference on System of Systems Engineering. IEEE, 2007, pp. 1–6.

[42] H. Yan, L. Fu, Y. Qi, D.-J. Yu, and Q. Ye, "Robust ensemble method for short-term traffic flow prediction," Future Generation Computer Systems, vol. 133, pp. 395–410, 2022.

[43] Y. Zhang and D. Xin, "A diverse ensemble deep learning method for short-term traffic flow prediction based on spatiotemporal correlations," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 9, pp. 16 715–16 727, 2021.

[44] J. Xiao, Z. Xiao, D. Wang, J. Bai, V. Havyarimana, and F. Zeng, "Short-term traffic volume prediction by ensemble learning in concept drifting environments," Knowledge-Based Systems, vol. 164, pp. 213–225, 2019.

[45] G. Dai, J. Tang, and W. Luo, "Short-term traffic flow prediction: An ensemble machine learning approach," Alexandria Engineering Journal, vol. 74, pp. 467–480, 2023.

CURRICULUM VITAE

2010 – 2012	B.Sc., Ege University, İzmir, TURKIYE
2013 – 2016	M.Sc., Meliksah University, Kayseri, TURKIYE
2017 – 2024	Ph.D., Abdullah Gul University, Kayseri, TURKIYE

SELECTED PUBLICATIONS AND PRESENTATIONS

J1) N. Cini, G. Yalcin, A Methodology for Comparing the Realiablty of CPU and GPU-based HPCs published in ACM Computing Surveys (February 2020).

J2) N. Cini, Z. Aydin, A Deep Ensemble Approach for Long-Term Traffic Flow Prediction, published in Arabian Journal for Science and Engineering (January 2024).