TRAFFIC VOLUME PREDICTION USING LSTM IMPROVED WITH RULES OBTAINED FROM ANOMALIES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

YASIN BERK GÜLTEKIN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

SEPTEMBER 2021

Approval of the thesis:

TRAFFIC VOLUME PREDICTION USING LSTM IMPROVED WITH RULES OBTAINED FROM ANOMALIES

submitted by **YASIN BERK GÜLTEKIN** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Halit Oğuztüzün
Head of Department, **Computer Engineering** _____

Prof. Dr. İsmail Hakkı Toroslu
Supervisor, **Computer Engineering, METU** _____

Dr. Cevat Şener
Co-supervisor, **Computer Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Pınar Karagöz
Computer Engineering, METU _____

Prof. Dr. İsmail Hakkı Toroslu
Computer Engineering, METU _____

Assoc. Prof. Dr. Tansel Dökeroğlu
Computer Engineering, Ankara Science University _____

Date: 09.09.2021

I hereby declare that all information in this document has been obtained and presented 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 material and results that are not original to this work.

Name, Surname: Yasin Berk Gültekin

Signature :

ABSTRACT

TRAFFIC VOLUME PREDICTION USING LSTM IMPROVED WITH RULES OBTAINED FROM ANOMALIES

Gültekin, Yasin Berk
M.S., Department of Computer Engineering
Supervisor: Prof. Dr. İsmail Hakkı Toroslu
Co-Supervisor: Dr. Cevat Şener

September 2021, 68 pages

Traffic volume forecasting is crucial in order to create a successful smart transportation system. The precision and timeliness of the traffic flow data are critical to the forecast's effectiveness. The lack of data has led to the usage of shallow architectures in traffic forecasting models or the creation of models based on fabricated measurement data. These models did not have a high level of success in predicting outcomes. In the world of big data, the variety and scale of the acquired traffic data has increased in lockstep with the increase in traffic density. It has a great importance for any field to be able to determine the situations that may occur in the future from the events of the past. History repeats itself constantly and in many time series problems it might be possible to make predictions for the future from what happened in the past. The LSTM structure in the Machine Learning has been implemented in order to be able to do these predictions. LSTM is a special type of recurrent neural network and its main difference from standard recurrent neural network is its success in modeling long-term information. For LSTM usage, there must be ongoing data for a certain period of time, and from this historical data the future state is predicted. In this research,

we use LSTM to estimate the number of vehicles in traffic and by estimating it, we aim to see traffic congestions before they occur. In order to overcome the difficulty of LSTM when predicting vehicle count at an anomaly state, we added a new layer that puts these anomalies in a certain rule set and we used this rule set to improve accuracy of the predictions. In this study, an improved hybrid model was created by adding anomaly-based rules on top of the LSTM model, thus a model that achieves successful results even in anomaly situations.

Keywords: Machine Learning, Deep Learning, Long-Short Term Memory, Traffic Volume Prediction, Anomaly Detection, Hybrid LSTM

ÖZ

ANOMALİLERDEN ELDE EDİLEN KURALLARLA GELİŞTİRİLMİŞ UZUN-KISA SÜRELİ BELLEKLER İLE TRAFİK HACİM TAHMİNİ

Gültekin, Yasin Berk
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. İsmail Hakkı Toroslu
Ortak Tez Yöneticisi: Dr. Cevat Şener

Eylül 2021 , 68 sayfa

Başarılı bir akıllı ulaşım sistemi oluşturmak için trafik hacim tahmini çok önemlidir. Akış verilerinin kesinliği ve zamanlılığı, tahminin etkinliği için kritik öneme sahiptir. Veri eksikliği, trafik tahmin modellerinde sığ mimarilerin kullanılmasına veya fabrikasyon ölçüm verilerine dayalı modellerin oluşturulmasına yol açmıştır. Bu modeller, sonuçları tahmin etmede yüksek düzeyde bir başarıya sahip değildi. Günümüzün büyük veri çağında, trafik yoğunluğunun artmasıyla birlikte elde edilen trafik verilerinin çeşitliliği ve ölçeği hızla artmıştır. Geçmişte yaşanabilecek olaylardan gelecekte oluşabilecek durumları belirleyebilmek her alan için büyük önem taşımaktadır. Tarih kendini sürekli olarak tekrar eder ve birçok zaman serisi probleminde geçmişte olanlardan geleceğe yönelik tahminlerde bulunmak mümkün olabilir. Bu çalışmada trafik sıkışıklığının tahmini için Machine Learning'deki Uzun Kısa Süreli Bellek yapısı hayata geçirilmiştir. UKSB, özel bir tekrarlayan sinir ağı türüdür ve standart tekrarlayan sinir ağlarından temel farkı, uzun vadeli bilgileri modellemedeki başarısıdır. UKSB kullanımı için belirli bir süre için devam eden bir veri olmalıdır ve bu geçmiş verile-

riyle gelecekteki durum tahmin edilir. Bu araştırmada trafikteki araç sayısını tahmin etmek için UKSB kullanıyoruz ve bunu tahmin ederek trafik sıkışıklıklarını oluşmadan önce görmeyi hedefliyoruz. UKSB'nin kullanılan verinin anomali içeren kısımlarında araç sayısında yaşadığı zorluğu yenebilmek için bu anormallikleri belirli bir kural setine oturtup tahminlerimizde bu kurallara göre düzeltmeler yapan yeni bir katman ekledik. Bu çalışmada, UKSB modelinin üzerine anomali tabanlı kurallar eklenerek geliştirilmiş bir hibrit model oluşturulmuştur, bu sayede anomali durumlarında dahi başarılı sonuçlar elde eden bir model elde edilmiştir.

Anahtar Kelimeler: Makine Öğrenmesi, Derin Öğrenme, Uzun-Kısa Süreli Bellek, Trafik Hacim Tahmini, Anomali Tespiti, Hibrit UKSB

To my lovely family

ACKNOWLEDGMENTS

First and foremost, I would like to thank my supervisors, Prof. Dr. İsmail Hakkı Toroslu and Dr. Cevat Şener, without their great support this thesis would not have been possible. I was very lucky to have their excellent guidance and assistance throughout the process.

I'd also like to thank my family for their unwavering support, not just during the writing of this thesis, but throughout my whole life. They've always encouraged me to pursue my goals. I am grateful for my family's unconditional, unequivocal, and loving support.

I would like to open a separate paragraph for my father Ahmet Cevdet Gültekin. The vision, self-confidence and courage he gives to his children should be the greatest legacy a father leaves to his children. I know you were always proud of me and I will always be proud of you.

My dear friend Hasan Ali Duran, with whom I had always endured the most difficult periods during the educational process and triumphed together, was also one of my most greatest supporters in this though period. I will be eternally thankful for your assistance in completing my thesis and for being there for me throughout the process.

I'd also like to express my gratitude to my coworkers for sharing their thoughts and experiences with me throughout this process. Their experiences have proven to be extremely beneficial in assisting me at the challenging situations.

Finally, I'd want to express my gratitude to all of my friends for their support during this process although not in the thesis part but in the entertainment part I was very lucky that they were always with me.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

LSTM	Long Short Term Memory
NN	Neural Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
MLP	Multilayer Perceptron
AI	Artificial Intellegence
KF	Kalman Filter
kNN	K-Nearest Neighbour
SVM	Support Vector Machine
BPNN	Back-Propagation Neural Networks
CTM	Cell Transmission Model
ARIMA	Autoregressive Integrated Moving Average
VAR	Vector Autoregressive

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

Congestion is described as an increase in the usage of the road network, progress at slower speeds, travel for longer periods of time, and anticipation of long queues. If the tools are slow, traffic density hinders the movement of the vehicles, and traffic jams are regarded to have occurred. If traffic density reaches or exceeds the road's capacity, an emergency would be declared.

When the field demand on the road exceeds the road's present capacity, a traffic jam occurs. There are a variety of factors that might result in traffic congestion. Traffic density can be caused by a road constriction at a certain location or distance, an increase in the number of cars using the route, weather conditions, or traffic accidents. The capacity of a path is determined by the speed of the vehicles.

With the advance of the technology, car usage time of the people is increasing day by day. Although recently the number of vehicles increased rapidly, the amount of road count and width did not meet that increase. As a result of these, the traffic jam is becoming one of the greatest problems of every city growing. One of the most effective ways of prevention of this traffic is that this is the best determination of when and where this traffic jam occurs.

Some of the main reasons for the formation of traffic jams are rush hours and accidents. Efforts to encourage public transportation and reduce the number of vehicles can be shown as a solution to this problem. All these solutions offered are both long-term investments and expensive. Trying to reduce traffic jam by using big data

obtained with smart traffic systems can be a more cost-effective and quick solution. In the digitalized world, constant green light periods as in the past are not acceptable. Although the traffic jams that will occur can be predicted, no solution can be found for them. However, thanks to the smart traffic system to be installed, where, when, and how much traffic jam will occur, it can be predicted in advance and since this traffic jam is determined in advance, it can be taken very easily.

1.2 Proposed Method & Contributions

This study starts with base LSTM model in order to predict future car counts in the junctions. Firstly we have tried to improve LSTM accuracy with improving the data quality that we have given to the model. We have expanded the data by adding new features to it. Main difference that we have done is clustering the junctions and using these clustered junctions data to train the model. In this way, thanks to the training of the model with the data of similar junctions, we learned many scenarios that could occur at the junction and ensured that it was trained more comprehensively.

We have seen that LSTM produces inaccurate results when estimating the number of vehicles in anomaly situations. As a solution to this, we have worked to detect anomalies at these intersections in advance and to make the LSTM's prediction more accurate when this situation occurs. While detecting these anomaly situations, we found the base value of these intersections according to how many vehicles were at the intersections in the previous weeks of the same day. We found the situations that deviated from this base value and marked them as anomaly+ and anomaly-. We made this marking according to whether the number of vehicles is more than 30% below or above the base value.

After these markings, we found the anomaly at which intersection caused the anomaly at which intersection and created rules between the intersections. In this way, we have determined how it will affect which intersection after an anomaly situation at an intersection.

After the rule definition, it was left to integrate this rule definition into our LSTM model. We did not affect the tool predictions made by LSTM in any way and allowed

it to continue generating predictions based on the trained model. However, according to the rules we created, if we expect an increase in the number of vehicles at an intersection due to another intersection and this was not estimated by LSTM, then we intervened and decided to reduce or increase this estimate. In this way, we have produced a hybrid model that contributes to the predictions that LSTM has difficulty in anomaly situations and enables it to produce better results.

1.3 The Outline of the Thesis

The organization of the thesis can be expressed as, Chapter 1 consists of motivation and problem definition, proposed methods and models and contributions. Chapter 2 explains about the background information that is needed in order the understand the study subject of this thesis. It begins with Deep Neural Network which is so generic about the study and ends with Long-Short Term Memory which is more context specific. Chapter 3, on the other hand, is the chapter where the previous studies on the traffic prediction and these methods are explained in five main topics. In Chapter 4, firstly we explain data that we used and express the details in order to show how we managed to make it usable in our model. After the definement of the data we outline the methodology we use and how our proposed solution functions. Chapter 5 is the section where we present our experiments on the data and show how we improve our solution part by part. We explain the parameters that we used and show the experiment results that we achieved, also show the comparison betwwen our study and the previous studies that are held before. In our last chapter, Chapter 6, we make a summary conclusion of our study and express our solution steps by presenting the structure of the work that we bring to completion.

CHAPTER 2

BACKGROUND

2.1 Machine Learning and Deep Learning

Machine learning is a type of artificial intelligence that allows systems to learn from their mistakes and improve without having to be explicitly programmed. Machine learning is concerned with the creation of computer programs that can access data and learn on their own.

The learning process starts with observations or data, such as examples, firsthand experiences, or instructions, with the goal of finding patterns in the data and applying the lessons learned to make better decisions in the future. The main objective is for computers to learn on their own and adapt their behavior accordingly, without the need for human interaction or input. [12]

Artificial neural networks, which are algorithms based on the structure and function of the human brain, are the subject of deep learning and a branch of machine learning. Deep learning is a fast expanding discipline with a wide range of applications.

Neural networks are a class of algorithms that detect underlying relationships in a data stack. The way the brain functions has a big effect on these algorithms. These networks can adapt to a variety of inputs and produce the best results without any need to design the output criteria again and again. They are very identical with the biological neuron systems. Deep learning techniques, which are based on neural networks, are a key component of machine learning. There are a variety of neural network topologies and each one has its own unique features. While choosing between them it's important to choose the best suited one for the task you want to achieve.

Deep learning can be described as a set of algorithms that allows computers to learn from example in the same way that humans do. There are various subjects that deep learning found better solutions and these areas are widening day by day. Voice control is just an small example that deep learning brought into our lives, as a result it eases so much things just by doing this. Deep learning has gotten a lot of media lately, and with good cause. It's accomplishing accomplishments that were previously unattainable.[98], [99]

A trained deep learning model is capable of categorization tasks directly from the inputs that are given which can be images, texts, or sounds. The precision on these categorization tasks is astronomical, furthermore the precision is better than what a human can do most of the times. While understanding the deep learning seeing the examples of it is a key process. There are lots of examples but i will talk about just some of them to in order to show the big picture.

- Automated Driving

  To operate an autonomous vehicle, one must have expert human driving skills. A considerable amount of real data is required in order to analyze roads, stop signs and traffic lights, pedestrians, speed limits, and many more situations like these. The performance of the algorithms will be enhanced of the vast data, which will increase the decision-making flow.

- Facial Recognition

  Facial recognition has a variety of applications, ranging from security to social media platform tagging mechanisms. Along with its significance, it also has its share of problems. To recognize the same person with weight growth/loss, beard/no beard, new hairstyles, and so on.

- Predictive Analysis

  Predictive analytics is one of the most promising examples of machine learning. Everything from predicting amount of power production of solar systems to predicting trafic congestion times of the roads are applications of it.

2.2 Difference Between Machine Learning and Deep Learning

Deep learning is subsection of the machine learning area. The first step in a machine learning workflow begins with the process of extracting beneficial characteristics and useful features of the data manually. The features are then utilized to build a model for the informations in the data. Relevant features are automatically retrieved from data using a deep learning approach. Furthermore, deep learning does "end-to-end learning" which means it learns how to complete all model processing on its own, which is referred to as automated.

Another significant distinction is that deep learning algorithms scale as data grows, whereas shallow learning methods converge. Machine learning algorithms that settle at a specific level of performance as more instances and training data are added to the network are referred to as shallow learning.

Deep learning networks have the advantage of continuously improving as the quantity of data grows.

Machine learning provides a number of approaches and models from which to pick depending on application, the amount of data to be processed, and the problem trying to be solved. A successful deep learning application needs a huge quantity of data (thousands of pictures) for training the model, as well as GPUs (graphics processing units) to handle the data quickly.

When selecting between machine learning and deep learning, consider there are high-performance GPU and a large amount of labeled data. Machine learning may be a better choice than deep learning any one of those traits doesn't exist. Because deep learning is more difficult than standard machine learning, there has to be at least a few thousand data to get reliable results.

Deep learning models divide into two main subsections: Supervised Models and Unsupervised Models. Multilayer Perceptrons, CNNs, RNNs, LSTMs are algorithms that are under supervised models while Boltzmann Machines and AutoEncoders are listed under the unsupervised models.

There are several characteristics that divide the two, but the most important distinc-

tion is in how the models are trained. In unsupervised models the input data is given merely and there is no output data that the model can learn from, whereas the input that is given to supervised models includes labeled set of data. In an unsupervised model, data is not labeled we are trying to find the this result column like what happens in the most clustering algorithms.

2.3 Deep Learning Models

2.3.1 Classic Neural Networks (Multilayer Perceptrons)

Multilayer perceptron (MLP) is the most basic neural network which improves itself with the feed forwarding method. An MLP is made up of three layers of nodes: input, hidden, and output. Except for input layer all the other layers have a nonlinear activation function. Hidden layer can be more than one but the input and output layers are always unique.

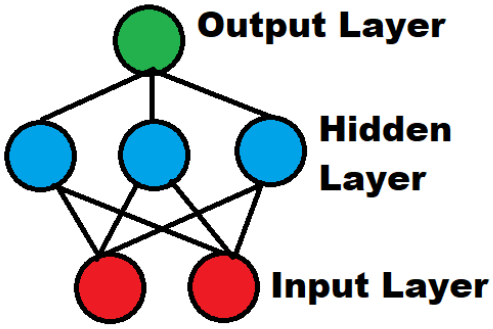


Figure 2.1: "Neural Network"

Backpropagation is a method that enables MLPs to upgrade themselves at the every epoch. At every iteration the weights between the nodes are updated in order to create an output that is closer to the true output. MLPs are particularly well suited to projects requiring tabular datasets, classification prediction challenges, and regression prediction tasks.

2.3.2 Convolutional Neural Networks

Convolutional Neural Network is a more sophisticated and advanced form of basic artificial neural networks. CNNs are created in order to deal with increasing complexity in computation and also speed up the data preprocessing. CNNs can be esteemed as the most efficient method while classifying image data since they were built for image data. While CNNs were not meant to operate with non-imagery data, they can nevertheless deliver amazing results when used with other types of data as well.

The goal of a CNN is to use convolutions to learn more specific attributes of the data. Image categorization and object recognition can be given in the first place as examples of the areas where this algorithm works best [2]. CNNs are based on multilayer perceptrons in terms of architecture. Local spatial correlation is exploited by CNN by imposing local connection limitations between neurons in nearby network layers [9]. CNN's primary goal is to process data using the convolutional algorithm. When any signal is converted to another signal, a third signal is created that provides more information about the signal than the original signal.

2.3.3 Recurrent Neural Networks

A recurrent neural network (RNN) is a special form of artificial neural network that works mostly with time series or sequential data. The main difference of RNNs from basic neural networks is its capability to remember previous data. Giving the output obtained from the previous state as an input to the next state enables it to perform this remembering action. This feedback method is applied at the RNN's hidden layer furthermore this layer can hold much more earlier states' information and use them while predicting the next output.

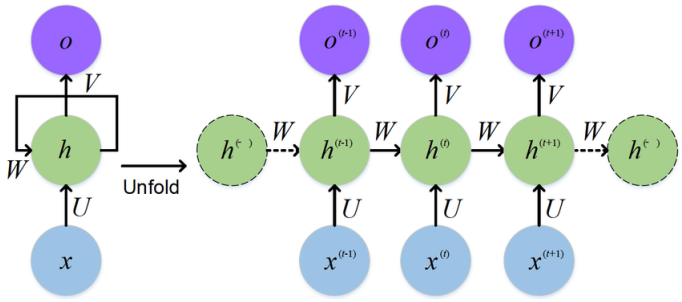


Figure 2.2: "Node Presentation of RNN"

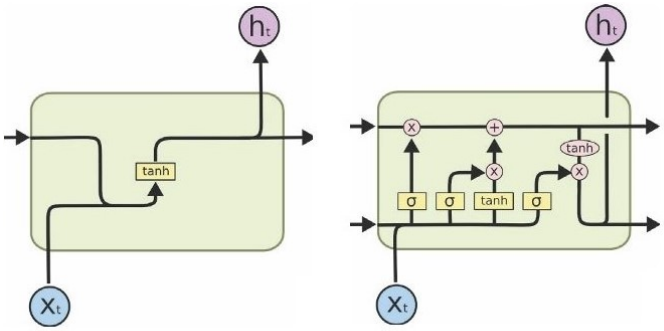
The memory of the RNNs remembers all the previous data and use them while generating new output but it doesn't use all the data with same weight. It remembers and uses last state outputs with higher weights on the other hand outputs long past are used with smaller weights and their effect on the output is smaller as a result.

Due to its better learning capacity and ability to perform challenging tasks like learning handwriting or speech recognition, RNN is one of the most popular and mostly used NN model. Prediction issues, machine translation, video tagging, text summarization, and even music creation are some of the other domains where RNN is used.

2.3.4 Long-Short Term Memory Neural Networks

Sepp Hochreiter and Jürgen Schmidhuber introduced long-short-run network architecture as a solution to the vanishing gradients problem [8]. The basic principle behind this network architecture is that the network reliably transmits important information to the future in multiple iterations [7].

LSTMs are a special kind of RNN that can learn order dependency in sequence prediction tasks. This is a behavior that is required in a variety of difficult problem areas, including voice recognition, sequence prediction, and others. Short-term memory is a problem for recurrent neural networks. They'll have a hard difficulty transferring information from earlier time steps to later ones if the sequence is lengthy enough. If you're attempting to predict anything from a paragraph of text, RNNs may leave out essential information at the start.



(a) RNN (b) LSTM

Figure 2.3: "RNN and LSTM"

During backward propagation, the vanishing gradient problem affects recurrent neural networks. Gradients are the values that are utilized to update a neural network's weights in order to predict better results. The vanishing gradient problem occurs when a gradient declines as it extends back in time. When a gradient value goes below a specific threshold it has no benefit for the model anymore. It ceases to be useful in learning and it loses its meaning.

Layers that receive a minimal gradient update in recurrent neural networks hence stop learning. Those are the layers that are frequently the oldest. RNNs can forget what they've seen in longer sequences since these layers don't learn, giving them a short-term memory.

Hochreiter&Schmidhuber (1997) introduced the term of LSTM and numerous other searchers developed the LSTM concept in their own works. They are currently frequently utilized and function exceptionally effectively on a wide range of situations. LSTMs are designed to solve a specific problem which is long-term dependency. Since their main purpose of creation is solving this problem they don't have hard time when remembering and applying long past values. All recurrent neural networks are made up of a series of repeated neural network modules. This repeating module in ordinary RNNs will have a relatively simple structure, such as a single tanh layer.

Before LSTM structure recurrent neural networks weren't able to remember the past in terms of long term. It removed the vanishing gradient problem by putting a series of gates which are stored in memory blocks of the LSTM structure. An LSTM and

a RNN has a similar structures, they all processes data passing on information as it propagates forward. The difference that seperates them the operations that are happening inside of the LSTM cells.

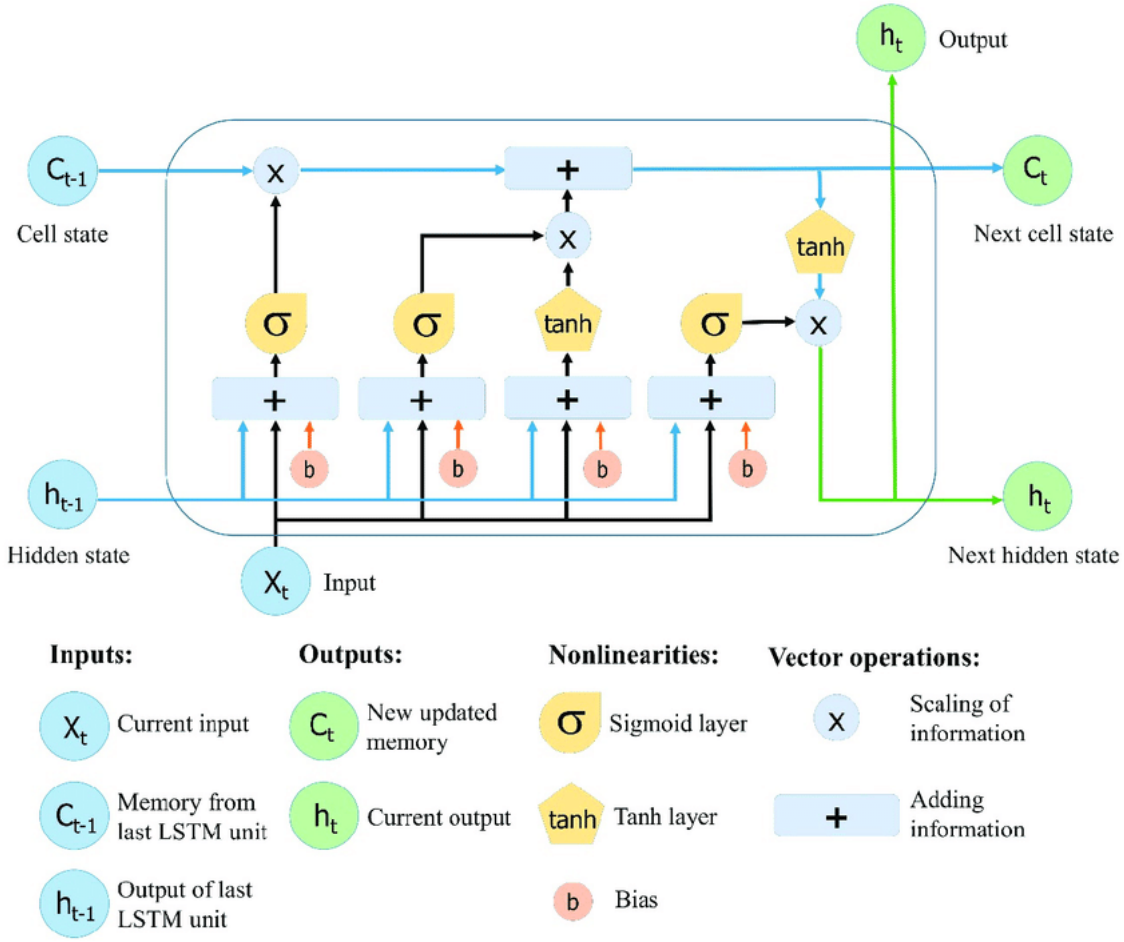


Figure 2.4: "LSTM Structure"

The LSTM uses these procedures to remember or forget information. While looking at these procedures can be intimidating, we'll do it step by step.

LSTMs are built around the cell state and its several gates. As relative information flows down the sequence chain, the cell state acts as a transportation highway. It's the network's "memory". The cell state should, in theory, be able to keep relevant details throughout the sequence processing."As a result, information from earlier time steps can make its way to later time steps, reducing the impact of short-term memory." As the cell state moves, information is added to or removed from it via gates. The gates

are a set of neural networks that determine which information about the state of the cell is allowed. During training, the gates might learn what information is important to keep or forget.

The cell state and its numerous gates form the foundation of LSTMs. The cell state decides how to use the information while it moves down the chain of sequences and by doing this constructs the network's memory. This permits knowledge from earlier time steps to flow into later time steps, limiting the impact of data that is closer to the present time. It decides whether the information that comes from gates will be remembered or deleted according to the cell state. The gates are a group of neural networks that determine what information about the cell's state is permitted. The gates may learn what information is necessary to preserve during the training.

The forgot gate is responsible of deciding whether data should be preserved or destroyed. The sigmoid function passes information from the previous hidden state as well as current input information. The outcomes range from 0 to 1. The closer you go to 0, the less you remember, and the closer you get to 1, the more you remember.

The input gate is used to update the cell state. First, we use a sigmoid function to combine the previous hidden state and the current input. By changing the values to be between 0 and 1, this determines which values will be updated. A value of 0 indicates that it is not important, whereas a value of 1 indicates that it is important. To further manage the n, you also send the hidden state and current input into the tanh function to squish values between -1 and 1. The sigmoid result is then multiplied by the tanh output. The sigmoid output will determine which information from the tanh output should be kept.

Firstly the cell state and the forget vector is multiplied but this multiplication operation occurs in a pointwise manner. If the result of this multiplication is closer to the 0 then this value is closer to forget but if it is closer to 1 then this value is worth to remember. While the process moves on a pointwise addition occurs and new cell state is found.

At the end of the structure there's the output gate. The next state of the hidden state is decided by the output gate. Since we are trying to carry the previous data information

all along the hidden state contains this earlier input information and and it puts the neccessary information while predicting result. Previous hidden state and current input combined with the help of sigmoid function, after this process tanh fucntion steps in and it processes the newly adjusted cell state. In order to make structure understand what information will be remembered and which will be forget tanh operation and sigmoid operation results are multiplied. The result of this multiplitication operations gives us the the hidden state. The new cell state and hidden state are used in the next steps at the very beginning of this procedure.

2.4 Clustering Algorithms

Clustering is one of the most used unsupervised machine learning algorithms. Clustering is an algorithm when there is a data that is required to be divide into groups according to some of its features. While the process of grouping 2 or 3 dimensional data on a plane is quite simple with the human eye, while the size of the data increases, this grouping process becomes more and more difficult, and various algorithms have been developed to solve this problem.

Clustering techniques are employed when there is no class to forecast and the instances need to be sorted into natural groups. [12]

In feature space, a cluster is a dense area in which domain examples (observations or rows of data) are closer together than other clusters. The cluster's center of gravity, as well as an edge or extension, can be represented by a pattern or point feature space.

These clusters are meant to represent a mechanism at work in the domain from which the examples are being drawn, a mechanism that causes certain instances to be more similar than others. [12]

Although there are numerous clustering techniques in the literature, it is impossible to categorize them precisely. [13] Clustering algorithms can be classified depending on the type of data inputs, the discovery of similarities between data items, or the theory and concepts on which clustering analysis methods are built to make the application methods more understandable. [14] From this perspective, the categorizations based

on the fundamental theories and concepts on which clustering algorithms are built are shown below.

2.4.1 Hirearchial Clustering

The technique of changing the proximity matrix into a series of nested divisions is known as hierarchical clustering. [15] When constructing clusters, a hierarchical structure is followed. The resulting hierarchy is commonly represented by a dendogram, which is a tree structure [20]. The complete data set investigated is contained in the root node of the tree structure known as a dendogram, and the leaf nodes are the partitions obtained as a result of binary fragmentation. [16] To further understand the hierarchical clustering approach, a mathematical representation of the concept of nested two sets and a general description of hierarchical clustering will be beneficial. The definitions [16], [17] are as follows:

- (Nested Two Sets) With a $A = A_1, A_2, ..., A_r$ aggregation another cluster $B = B_1, B_2, ..., B_s$ is nested if and only if r>sand there is a set $B_j \epsilon B$ for every $A_i \epsilon A$.
- (Clustering by Hierarchy) The clustering of an X data set with $H_1 = x_1, x_2, ..., x_n$ is known as hierarchical clustering. Clustering ile $H_1 = x_1, x_2, ..., x_n$ is a set of $H = H_1, H_2, ..., H_q(qn)$ nested partitions with $C_i \epsilon H_r, C_j H_s$, and r > s. This suggests that $\forall i, j \neq i, r, s = 1, ..., q$ for $C_i \epsilon C_j$ or $C_i \bigcap C_j = \emptyset$.

Clusters that exist at every level of the hierarchical structure obtained in the hierarchical clustering process can be divided into new clusters or combined as a cluster under certain conditions. Based on these two different approaches, hierarchical clustering algorithms are divided into two categories. These categories are called divisive and agglomerative methods, respectively.

- Agglomerative Methods: As the name implies, merging and other combining methods are based on the idea of combining existing sets. To begin, each data object is handled as an independent set. It continues until a single cluster is obtained at each level by joining the two nearest clusters or clusters [18] or until a defined termination threshold is reached. [19]

- Divisional Methods: Starts by treating data objects as if they were all part of the same set. Each cluster at each level of the hierarchy is separated into its own subsets under certain criteria. This procedure is repeated until the desired number of data clusters has been generated or a specified termination criterion has been satisfied.

Bottom-up methods go from the leaves to the root, while associative methods go from the root to the leaves; partitioning methods are also called top-down hierarchical methods since they go from the root to the leaves. [18]

The conditions for merging or splitting sets in associative and dividing hierarchical methods have a significant impact on the outcome. This is because when a cluster pair is merged or a cluster is created, these operations cannot be undone. [19]

The clusters are merged or separated using a variety of similarity methods. Hierarchical clustering techniques change depending on how these similarities are defined. As a result, associative and partitioning hierarchical clustering algorithms can be divided into two categories based on how they find similarities among themselves. For associative hierarchical clustering approaches, the single linkage, complete linkage, and average linkage clustering methods are the most commonly used. [21]

- Single Linkage: The similarities (or differences) of the clusters are found using this method, which is also known as the closest neighbor method, based on the cluster pair with the shortest distance between them, with one element from each cluster. This method frequently results in slender and elongated shaped clusters. Objects with different structures can be located in the same cluster as a result of this circumstance. [22]

- Complete Linkage: The cluster pair with the highest distance between 22 clusters, with one element from each cluster, is used to determine the similarity (or differences) of the clusters. Methods that use a completely linked approach are more likely to produce compact, globular clusters. [22] This method is also known as the farthest neighborhood approach in the literature. [22]

- Average Linkage: In this method, the clusters' similarity (or differences) is

found by averaging the distances between pairs of objects, with one element from each cluster.

Generally, the pros and cons of hierarchical methods can be as shown below [23]:

- Pros of Hierarchical Methods:
  - It is intuitive and easily understandable. It can be informative in data representation, as an ordering of objects is obtained.
  - Can obtain small clusters that can be useful in the knowledge discovery process.
- Cons of Hierarchical Methods:
  - It is ineffective against noise and contradictory data.
  - Separation or merge operations performed at any level cannot be undone.
  - They may have difficulty dealing with clusters of different sizes.
  - They may be inconvenient for large-sized data.

CHAPTER 3

LITERATURE REVIEW

This chapter presents a comprehensive review of the literature, with a focus on traffic speed prediction and transportation deep learning. Currently, there are two types of traffic speed forecast methods: statistical modeling and artificial intelligence (AI). This section will go over two types of past prediction studies based on the categories, as well as the methodology used in this study.

For forecasting traffic speeds, statistical analysis has long been the most commonly utilized method. Traffic flow theory, time series models, and the Kalman Filter are among the approaches that are used for traffic forecasting.

With the help of machine learning modellings, many more techniques and technologies will be revealed to traffic researchers and engineers. It was also able to predict traffic flow in the short and long term using machine learning and data mining techniques with enough data. This section covers a variety of shallow machine learning algorithms, including k-nearest neighbor, support vector machine, and artificial neural network, as well as their applications in traffic flow prediction (ANN). Deep learning, which has increased in popularity recently, can now be used to predict traffic. As a result, the focus of this section will be on the applications of deep learning algorithms in transportation and how they influenced this study.

3.1 Nearest Neighbor Based Models

The kNN algorithm, which is a nonparametric pattern matching tool, is widely utilized in the data mining industry, also have been applied in the transportation studies.

Nearest neighbor algorithm have been used by Smith and Demetsky (1996) while forecasting traffic volume on motorway stretches at several periods in advance.[30] Scalability and the capacity to estimate traffic volume in a long term were two advantages. Handley et al. (1998) presented the kNN model to forecast trip time at many sites on a motorway, in addition to volume. They analyzed three nearest neighbors, many predictive variables, and normalization procedures, and found that speed prediction is less reliable than volume prediction especially at the peak hours of the traffic. In addition, to improve journey time prediction, a hybrid model combining a geographic information system (GIS) and nonparametric regression has been constructed.[33] Clark (2003) used the kNN model to forecast the traffic state for multivariate traffic characteristics.[34] The approach was tested on a single highway location with one month of data and shown to be a good predictor of traffic flow and occupancy, but not of speed.

In prior research, the pattern of traffic volume-such as the peak hour pattern and seasonal pattern—seemed to be easier to capture than traffic speed. As a result, this strategy is more frequently utilized in traffic volume forecasting. Many studies on traffic flow prediction have been undertaken by researchers.[32], [35], [37] Efforts to construct a real-time system have also been made. [31], [38] Researchers tried out different input variable configurations to see if they might enhance the prediction. Kindzerske and Ni (2007) developed a composite strategy in closest neighbor search to anticipate traffic conditions. Instead of using the complete network as input, they chose two upstream and two downstream sensors to create a tiny network for making predictions to reduce error. [39]

There hasn't been a lot of research into using kNN to estimate traffic speeds. To estimate traffic speed in 5-minute intervals, Yildirim and Cataltepe (2008) used a combination of kNN and SVM.[40] The results revealed that SVM outperforms kNN. In their other speed prediction study, Chen et al. (2014) integrated Gaussian process regression with kNN.[44] They used kNN to extract data features and fed them into Gaussian process regression to generate a reliable prediction. Overall, this method is simple to implement and transfer since it uses simple features extracted from historical data; yet, in terms of prediction accuracy, it does not beat more advanced machine learning approaches.

3.2 Artificial Neural Network Based Models

By stacking a huge number of factors and extracting significant properties automatically during the training phase, the neural network (NN) has the benefit of being able to imitate any function. By the effect of improvement of big data this approach has been widely employed in the area of traffic forecast.

A frequent example of a typical NN is the back propagation NN (BPNN). Back propagation is based on the idea that inputs are sent forward to the network, and then errors between the output and the 15 target are propagated backwards through the network, with the weights modified in the meantime using some optimization technique. As early as the late 1990s, the BPNN was used to anticipate traffic volume.[46], [29], [47] Huang and Ran (2003) evaluated traffic speed during poor weather using BPNN for a single connection in terms of traffic speed prediction. [48] They created a fully linked NN using weather and traffic data as inputs. Lee et al. (2007) took into account the time of day and weekday.[49] Other researches tested the ANN to classical ARIMA or a pattern matching model to predict traffic speed.[51], [53], [65], [55], [56]

Much progress has also been made as a result of the NN idea. Focusing on network structure optimization is one improvement. To determine layer connections or nodes, researchers employed a genetic algorithm. Several researchers have also focused on preparing traffic data in order to improve forecast accuracy Park as well as Rilett (1998) investigated 2 clustering techniques: Kohonen self-organizing feature maps and Rilett self-organizing feature maps. To extract features from data before feeding it into the NN, a fuzzy c-means model was used. [62], [57], [64], [63], [65]

Jiang and Adeli (2005), Xie and Zhang (2006), and Boto-Giralda et al. (2010) used wavelet transformations to denoise data for network training and produce traffic volume projections. Within the Neural Network, some of the researchers attempted to substitute the sigmoid function with the radial basis function in hidden layers. They compared their model to different methods for forecasting traffic volume . This sort of NN is simple to understand and has a simple structure. The RBF's performance, on the other hand, is contingent on the accuracy with which the center and width

parameters are estimated. [65], [67], [69], [70], [65], [35]

On different traffic conditions, different NNs could be used. By clustering the data first and then developing a modular Neural Network for each categorization, Park and Rilett (1998) suggested a modular Neural Network. Unlike other function approximation approaches, this method may estimate the full function without the need for prior knowledge. Chen et al. (2001) clustered traffic data using a self-organizing map (SOM) and then utilized BPNN for each cluster. Before utilizing ANN to forecast journey time, Lee (2009) employed the k-means clustering approach. Other researchers have also adopted this concept.[73], [74], [75] To deal with the nature of temporal dependency in traffic data, many research have modified the NN structure with a recurrent characteristic. Yasdi (1999) used recurrent Jordan networks to estimate traffic volume, Lint et al. (2002) used a recurrent state-space NN to predict journey time, and a time-lagged recurrent network to predict short-term traffic speed.[76], [93], [77] Ishak et al.(2003) explored Jordan-Elman networks, partially recurrent networks, and time-lagged feedforward networks for speed prediction.— and experimented with various input parameters in order to improve network speed.[77]

Similar studies evaluating different RNN topologies on traffic speed and trip time prediction were undertaken by Zeng and Zhang (2013) and Jiang et al. (2016).[36]

Some hybrid models have been studied by embedding statistical modeling in ANNs.[36], [78] Zheng et al. (2006) sought to integrate two types of NN using the Bayesian approach (BPNN and RBFNN).[72] It outperformed the single predictor, according to the data. Alecsandru and Ishak (2004) also looked at several topologies of the NN model to see how good it was at predicting traffic flow.[77]

The capacity to handle complex, nonlinear interactions in multivariate data is a benefit of the ANN family algorithm. The classic NN, on the other hand, is limited in terms of time-series prediction by short-term memory, which ignores the long-term trend that exists in a speed sequence. In addition, before GPU-accelerated computing, a typical ANN had a constrained structure because to the processing power constraint.

3.3 Traffic Flow Theory Based Models

The flow theory has been used by many transportation specialists to examine and anticipate different stages of traffic flow. The majority of them were focused on computing the network's origin-destination (OD) matrix to estimate volume.[24], [25] A variety of simulation studies based on flow theory have also been conducted. In addition to traffic volume estimation, researchers attempted to depict the full traffic evolution with volume, density, and speed. Daganzo (1994) presented a cell transmission model (CTM) to characterize and anticipate traffic evolution over time and location.[28]

Some academics have examined and modeled the stochastic properties in traffic flow to anticipate short-term speed by treating it as a stochastic process. Qi and Ishak (2013) studied urban freeways during rush hour and calculated speed transition probabilities, from which predicted values were retrieved and fitted using exponential models.[26]

3.4 Kalman Filter Based Models

Based on observed measurements and the present estimated state, the KF makes a prediction of a future state with statistical noises. This state space-based model has been employed in a number of traffic prediction studies due to its capacity to deal with multivariate data. Okutani and Stephanedes (1984) were the first to use KF to predict short-term traffic volume, and it outperformed the benchmark model UTCS-2.[85] The proposed wavelet KF outperformed the direct KF, according to the results. Guo et al. established a new breakthrough in volume forecasting.[48] They proposed an adaptive KF to fine-tune the variances and execute the Kalman recursion by using observation mistakes and state estimation errors to update the process variance.

Other researchers wanted to know how to predict journey times. They employed KF to anticipate freeway journey duration utilizing a variety of data sources, including toll tags, GPS data, and inductive loops.[90], [91], [94]

Kuchipudi and Chien also provided the flexibility of using path-based 11 or link-

based trip time computation depending on traffic conditions, based on their previous study.[90] Lint also introduced an extended KF (EKF) that may be used to anticipate freeway trip duration with the help of online learning.[93] Other applications have used KF embedded in network-level traffic prediction. [94], [62]

Despite the fact that KF has been used to predict traffic flow in numerous research, there are few applications in traffic speed prediction. Yang et al. (2004) proposed an adaptive recursive least-squares technique for predicting online traffic speed. [94] They employed an autoregressive (AR) model to obtain an offline estimate for the transition coefficient matrix and noise covariance matrix, then used KF to make predictions live. This study has a drawback in that it does not investigate the spatial relationship that happens in traffic propagation.

The Kalman Filter can handle multivariate time series and update the state on a continuous basis; nevertheless, it requires state transition and noise covariance knowledge, which are difficult to detect in traffic flow.

3.5 Time Series Based Models

Time-series modeling has been frequently utilized in traffic data analysis because of time-variant property of the traffic data. The autoregressive integrated moving average (ARIMA), which implies stationarity and constant variance, is a common model in traffic prediction studies. Hamed et al. (1995) and Lee and Fambro (1999) employed the ARIMA model to predict traffic volume.[27], [50] Williams et al. (1998) and Williams and Hoel (2003) investigated seasonality using the seasonal ARIMA (SARIMA) model. Also there are other researchers who coupled generalized autoregressive conditional heteroscedasticity (GARCH) with the ARIMA model to deal with traffic volatility. [84], [35]

There are also other ARIMA-based research which are focused on speed prediction. To deal with traffic regime shifts, Cetin and Comert (2006) proposed an adaptive ARIMA model (such as free flow regime, congested regime, etc.).[89] They employed the expectation maximization (EM) and cumulative sum (CUSUM) techniques, followed by the ARIMA model with adaptive parameters, to detect a change

in the mean of the process. Wang et al. (2014) established a hybrid EMD-ARIMA model by combining empirical model decomposition (EMD) with ARIMA.[59] The traffic data was decomposed into intrinsic model functions (IMFs), ARIMA models were applied to each of the IMFs, and the components were finally reconstructed to the projected traffic speed using empirical model decomposition.

To capture the spatial connection that occurs in traffic flow, several academics have been working on multivariate time-series analysis. Williams (2001) used the ARIMAX to add upstream volume as a transfer function input in his model.[80] Kamarianakis and Prastacos (2003) examined univariate and multivariate models for traffic prediction. They employed the ARIMA model, the vector autoregressive moving average (VARMA) model, and the space-time ARIMA (STARIMA) model to predict the speed on major arterials. [83] The multivariate model performed better in the forecasting results because it can deal with interdependencies between speed from surrounding detectors.

Pavlyuk (2017) compared VARMA, VECM, STARIMA, and the multivariate autoregressive space state (MARSS) model, as well as other multivariate models for speed prediction.[87] The multivariate model was proven to be capable of capturing the geographical and temporal connection in traffic flow, and it was proposed that forecasts be improved by using simultaneous modeling on volume, speed, and occupancy. Multivariate models have been used to forecast traffic speed in a similar way.

The vector autoregressive (VAR) model was employed by Chandra and Al-Deek (2008, 2009) to account for the spatial connection between two upstream and two downstream sites. [95], [96] With the assumption of the traffic speed has a daily periodic trend on work days of the week, Zou et al. (2015) developed a hybrid model to estimate the speed series via periodic trend and residual component.[97] The periodic component was estimated using the trigonometric regression function, and the residual was estimated using the space time (ST), VAR, and ARIMA models. The hybrid ST model with different prediction phases performed better, according to the data.

Other multivariate techniques were investigated in addition to the ARIMA model. With reference to multivariate data, Ghosh et al.(2009) introduced the structural time-series model (STM).[86] This model was used to estimate traffic volume at signalized

junctions by separating components in a time-series. Szeto et al. make another another advancement in multivariate time-series analysis (2009). To accomplish volume prediction on a signalized network, they included CTM into the SARIMA model.

Furthermore, with respect to traffic volume, traffic speed has fewer evident patterns and can be influenced by the latent variables, making standard time-series analysis difficult to forecast.

CHAPTER 4

IMPROVED LSTM WITH ANOMALY BASED RULES

Data description, analysis, preprocessing, and the building of the LSTM also anomaly rule-based improvements are all covered in this section.

4.1 Data Description

Dataset consists of 79 junctions of the Konya city of Turkey. There are sensors that count the vehicles at every junctions for every 4 directions. So for every hour we have 4 rows that specify vehicle count and green light duration at the junction. Every junction has a unique id and also data has a specifier that shows the direction of the road. It starts from the north with 1 and increases while going clockwise direction. While 1, 2, 3 and 4 corresponds to different roads of that junction 0 specifier includes the information that shows sum of the all cars on that junction and average of the green light durations.

There were 81 junctions but only 52 of them can be used in this thesis since the rest of the junctions' sensors were broken and they were not collecting any data. We have used 3 years of these data which consists of 2018, 2019 and 2020 years. A row of data was including date (year,month,day), time (hour, minute), vehicle count, green light duration. The sensors at the junctions were closed between 02:00 and 06:00 so every day has 20 hours that has sensor information. If a junction's data is complete this means that it has 4 rows for every hour, 4*20 rows for every day, 4*20*365*3 for 3 years data. At the end it means 4.555.200 rows of data for all 52 junctions.

4.2 Data Analysis and Preprocessing

First, we started by obtaining the list information of all junctions from the system. Thanks to the graphical interface that users can access, the system provides users with information on the location of each junction and whether there are sensors at these junctions on the map. By examining the requests sent from the system, it was necessary to analyze the information with which request it pulled all the junctions and how it colored these junctions. In the list that returned all junctions information, there was an information called status type for each junction. When we examined the system, we analyzed that this information includes information about whether the sensor is working at junctions or whether it is measuring. We filtered all the junctions information we received from the system according to this information. We used only two fields of this all junctions data.

After filtering the junctions we have noticed that filtering junction process was not successful. The reason of the failure was statusType information of the data wasn't giving the correct result for every junction. Therefore; we had to do create our own check mechanism in order to understand vehicle count measurement is valid or not. We have created a rule as "If the total vehicle count for the junctions is 0 for at least 10 days this measurement is invalid". As a result we have updated out filtering mechanism and we have obtained a new subset of junctions.

For this subset of junctions, we started to pull data daily because the system only allowed data for one day. Therefore, we had to make separate requests for each day. In each junction data, there is an indicator that tells which road it belongs to. This token increments by 1 for the northernmost road, 2 for the road to the right of the northernmost road, and other roads. The data includes the 0 token, and the data includes the total number of vehicles on that road and the average green light time. We used the junction id and road id information from the junction information.

The data for the 0, 1, 2, 3, 4 markers have the same structure, this data includes the number of vehicles measured on the roads during the day and the green light duration information that is changed at various intervals according to the number of vehicles. The year, month, day, hour and minute information, which shows when this

information occurred, is also included with this information.

We wanted to add some extra information to this data, since we were trying to guess vehicle counts at the junctions some extra information would help LSTM model to predict more correctly. First thing that comes to mind is adding day index to data. Let's think as human, when there is most traffic congestion on the roads? Most probably at the weekend days there are more vehicles in the traffic than the week days. Since people are not at work they are most probably outside and going some places. This information should be added to the data. By starting from Monday we give every day a index number. By doing this we added Monday, Tuesday, etc. information to the data. Since there can be some routines in that road that always happen on the Monday morning.

The second extra information was that adding the information whether that day is holiday or not to the data. This is also same as day index information. On the national or religious holidays there are more vehicles on the roads and this information can give a good idea to LSTM while it's guessing the relationship between vehicle count and day. We thought that not only day index(example: Monday:1, Tuesday:2) is enough, there should be extra column that states whether that day is holiday or not. This also includes the saturday and sunday informations. By holiday we are trying to specify the days that people are not going to their jobs actually. As a result all saturdays and sundays also the national/religious holidays are marked as holiday.

4.3 Model

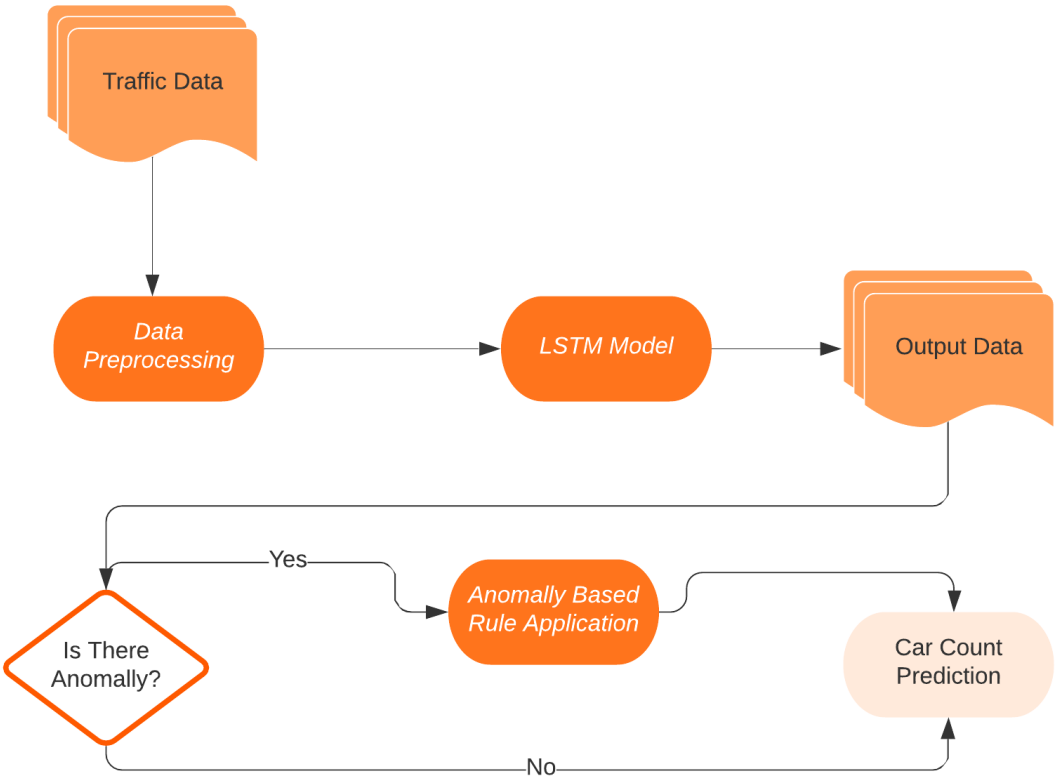


Figure 4.1: "Main Model Flow Chart"

Firstly data is downloaded and it goes through the preprocessing process described first and is made available as input to LSTM. Unlike Neural Networks in LSTM, a data sequence is given as input, not an instantaneous data. This is because LSTM uses this sequence when training itself, making the prediction by looking at a historical data sequence instead of looking at the last data of the sequence. When estimating the number of vehicles at the junction, we would be less likely to guess correctly if we only made an estimate based on the data 15 minutes ago. Instead, we used the LSTM structure, where the closest data affected the most, but the data 2 hours ago also had an impact on the forecast. In this way, we ensured that situations such as the increase or decrease in vehicles in the past had a direct impact on our vehicle forecast.

4.3.1 LSTM

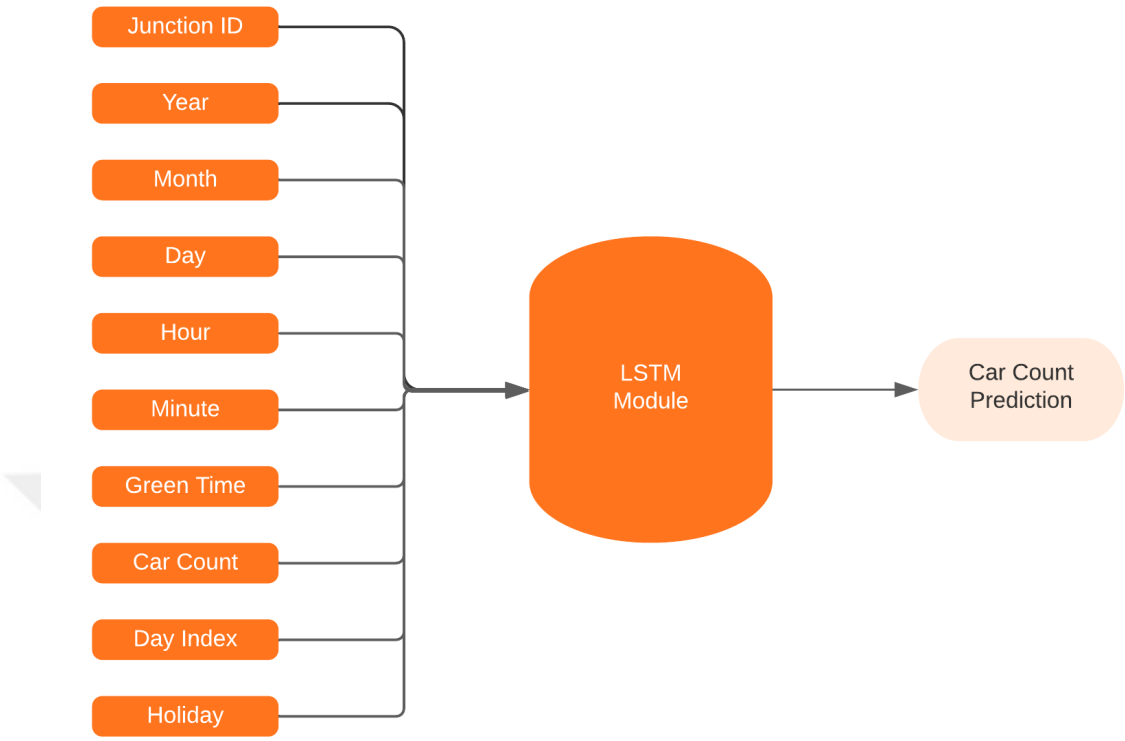


Figure 4.2: "LSTM Module"

LSTM module takes junction id, date, time informations, green light duration of the traffic light, vehicle count at the junction, information of which dat it is(day index), whether the day is holiday or work day as an input. It uses this information to train the model and produces an prediction of how many vehicles will be at this junction in the next time period as a result of the model.

4.3.2 Anomaly-Based Rule Extraction

When we tried to estimate the number of vehicles with the LSTM structure we prepared, we achieved good results. In general, there were slight differences between the number of vehicles we estimated and the actual number of vehicles. However, when there were big changes in the number of vehicles, the LSTM structure had difficulty

in estimating these vehicle numbers correctly. These big chances in the vehicle count can be described as anomalies in the junction since there is a unexpected changes in the vehicle count. We can call these changes unexpected because when we examine the data these big changes were not happening in a particular order always. Predicting these sudden increases/decreases was not easy job for LSTM model. In order to understanding data and predicting better results we wanted to examine and understand the data in a detailed way.

As a result of examining the data, we have seen that such situations generally appear as anomaly. To detect this, we did the following; we also looked at the number of vehicles on that road in the past weeks. For example, there is a sudden increase in the number of vehicles at 6 pm that week, but when we look at the previous weeks, if we can see that this is already happening regularly then we did not pay much attention. Since, LSTM does not have much difficulty on estimating these points because these sudden increases/decreases occur systematically on specific day at specific time from a specific reason. Therefore; LSTM structure can learn this by training with enough data and enough training time. The example we had real difficulty with is the traffic that occurred for no reason. Because there is no such example in the data it is trained on. While the number of vehicles was proceeding normally in previous weeks, this week there is a sudden increase/decrease at vehicle count for no reason. These were the examples that made LSTM's job difficult, and we wanted to detect these anomalies and make LSTM's job easier.

At first while making this determination, we first took the average of the previous 3 weeks. If the average number of vehicles for that road was in the previous 3 weeks, we expected results close to that average this week. In this way, we wanted to detect anomalies in the data beforehand and establish a mechanism that will help the LSTM system at the points where it is difficult. The 3-week average structure that we first established did not give proper results. Because if an anomaly occurred in the previous 3 weeks, this would directly affect the average we calculated and cause us to calculate an incorrect average.

Table 4.1: Road 39 - Anomaly Rule-1

ID	Year	Month	Day	Hour+Minute	vehicle count	Average	Diff(%)
39	2018	1	2	18:00	523	-	-
39	2018	1	9	18:00	459	-	-
39	2018	1	16	18:00	391	-	-
39	2018	1	23	18:00	438	457,6667	-4,4
39	2018	1	30	18:00	542	429,3333	20,7
39	2018	2	6	18:00	414	457	-10,3
39	2018	2	13	18:00	450	464,6667	-3,2
39	2018	2	20	18:00	472	468,6667	0,7
39	2018	2	27	18:00	400	445,3333	-11,3
39	2018	4	3	18:00	290	440,6667	**-51,9**
39	2018	4	10	18:00	530	387,3333	**26,9**
39	2018	4	17	18:00	490	406,6667	17,0

Let's examine the sample table above. The table includes the information of junction id, year, month, day, hour, minute, number of vehicles, average and how many percent the number of vehicles deviate from this average. Each line shows the number of vehicles on Tuesdays at junction 39, so the number of days is increasing by 7s.

From these lines, we can understand how many vehicles are at the junction every Tuesday at 18:00. We see that this junction did not deviate from its average in the first 3 months of 2018. The average number of vehicles in these 3 months is 433 and we can see that the values are generally around this value. The numbers in the average column are calculated by averaging the number of vehicles in the previous 3 weeks. The difference column next to it shows how much the number of vehicles in that week deviates as a percentage from the average of the previous 3 weeks. This method appears to work fine in general but it is not enough.

Table 4.2: False Finding of Anomaly Rule-1

ID	Year	Month	Day	Hour+Minute	vehicle count	Average	Diff(%)
39	2018	4	10	18:00	530	387,3333	**26,9**

The main reason why we are not satisfied with this method is the row you see in Table 4.2. We decide whether there is an anomaly at that moment at an junction, based on its deviation from the mean by a certain amount.

For example, let's set the threshold value as 25%. According to this table, we have two anomaly detections. We detect the 6^{th} and 13^{th} days of March as anomaly, but when we visually check, the 13^{th} day does not contain an anomaly. The number of 530 vehicles is a normal number for this junction. In fact, there was a vehicle number of 542 in the previous days and it was not marked as an anomaly that day. The reason why this day is marked as an anomaly is because an anomaly occurred in the previous week. There was a Tuesday evening with very few vehicles at this junction with 254 vehicles in the previous week, and since we calculate the average from this, we treat the next week as an anomaly. However, this week is actually close to the overall average of the junction, and treating this week as an anomaly is a misnomer.

For this reason, we decided that we should develop a new average calculation. There had to be a method that would not add to the average the extreme situations experienced in the previous weeks. As a result, we started to get the data from the previous 5 weeks and from these 5 weeks we disvehicleded the extreme values, the maximum and minimum value. We started to calculate much more reasonable averages by taking the average of the 3 values in the middle. You can see how the averages changed and rows which were marked as anomalies in the table below.

Table 4.3: Road 39 - Anomaly Rule-2

ID	Year	Month	Day	Hour+Minute	vehicle count	Average	Diff(%)
39	2018	1	2	18:00	523	-	-
39	2018	1	9	18:00	459	-	-
39	2018	1	16	18:00	391	-	-
39	2018	1	23	18:00	438	-	-
39	2018	1	30	18:00	542	-	-
39	2018	2	6	18:00	414	473,3	-14,3
39	2018	2	13	18:00	450	437	2,8
39	2018	2	20	18:00	472	434	8,0
39	2018	2	27	18:00	400	453,3	-13,3
39	2018	4	3	18:00	290	440,6	**-51,9**
39	2018	4	10	18:00	530	421,3	20,5
39	2018	4	17	18:00	490	440,6	10,0

As seen in the table, the 10^{th} day of April is no longer marked as an anomaly. We ensured that the anomaly in the previous week was not a factor in calculating this week's anomaly and we were successful in this. The week before that was marked as an anomaly as usual, and the number of vehicles in no week other than this week is 25% above or below the average number of vehicles.

Our application of this improved structure to the LSTM model was more beneficial than the original version. Our main goal was to improve the difficulty in estimating the number of vehicles LSTM experienced in anomaly situations, thanks to these rules. In fact, we did not make an update on the number of vehicles estimation with LSTM, we also estimated the number of vehicles with LSTM. For the structure that we will add on the LSTM, we first determined the anomaly situations that occur at the junctions. We have determined what day and what time an anomaly occurs at these junctions.

Next, we wanted to find out whether these anomalies trigger each other. In other

words, we wanted to determine whether an anomaly that occurred at the 39th junction at 8 pm would cause an anomaly to occur at a neighboring junction in the future. We found a subset of 4 junction in order to find out if the data contains such a relationships. Junctions with id 39, 41, 45, 46 were 4 junctions adjacent to each other on the map.

First, we detected the anomalies occurring at these junctions separately, and then we checked whether these anomalies triggered each other within 15 to 30 minutes. As a result of our controls, we came to the conclusion that the anomalies at these junctions really affect each other, and we thought that it would be better for us to divide the situations at the junction into three as normal, anomaly+, anomaly- instead of dividing them into two as normal and abnormal. Because the decrease in the number of vehicles that occur more than normal at one junction might have an effect on another junction as an increase. Likewise, an unexpected increase in the number of vehicles at another junction could cause a sudden decrease in the number of vehicles at another junction. Since we would make this estimation without knowing what kind of relationship there is between these junctions, separating these cases would allow us to obtain more accurate results.

With the method we mentioned, we found the anomalies at each junction and determined the relationships between the anomalies occurring at these junctions. One of the rule examples we are trying to find is that anomaly+, which occurs at the 39^{th} junction on Tuesday at 8pm, causes anomaly+ to occur within half an hour at the 41^{st} junction. So after we found this relationship, how did we integrate it into the system is need to be explained. LSTM has always estimated the number of vehicles, but we wanted to add improvement in case of anomalies. It will be more understandable if we go through the example scenario.

4.3.3 Applying Anomaly-Based Rule To LSTM

In order to understand when the anomaly based rule layer updates or does not update the lstm results, we have created a flow chart of the mechanism we have established.

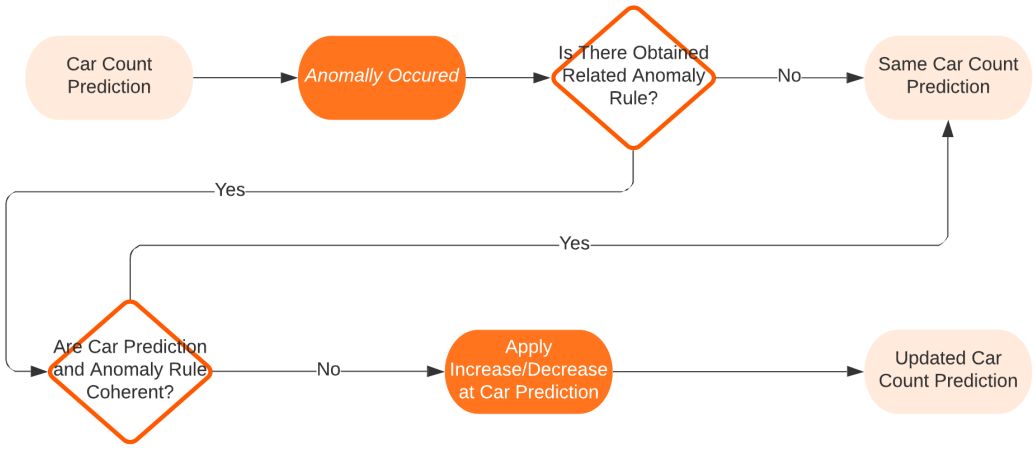


Figure 4.3: "Anomaly-Based Rule Flow Chart"

We have also constructed a pseudocode in order to create better understand on how and when we apply these anomaly based rules on the results.

Algorithm 1 Applying Anomaly Based Algorithm To LSTM Result

```
if there is obtained related anomaly rule then
    if anomaly rule and vehicle prediction are coherent then
        result = lstmPrediction
    else
        if anomaly rule predicts increase in volume then
            result = lstmPrediction + lstmPrediction * 0.20
        else if anomaly rule predicts decrease in volume then
            result = lstmPrediction - lstmPrediction * 0.20
        end if
    end if
else if no rule obtained then
    result = lstmPrediction
end if
```

Let's assume we have find a rule like this:

- Anomaly+ situations occurring at between 8am and 9am on Monday at the 39^{th}

junction cause Anomaly+ to occur at the 41^{st} junction within half an hour.

Table 4.4: Example Anomaly Rule

Triggering Junction	Triggered Junction	Day-Hour
39(Anomaly+)	41(Anomaly+)	Monday - 08:00-09:00

We can update LSTM's estimate for Junction 41 in weeks of anomaly+ occurring for Junction 39. We decided to apply this update when as follows; we see that anomaly+ has formed at the 39^{th} junction, then we expect anomaly+ to occur at the 41^{st} junction. But the estimate made by LSTM is not an anomaly+ estimate (that is, a low estimate close to the average of the junction), then we intervene in this estimate and add 20% to the estimated number of vehicles. In this way, we make it better with the rule that LSTM estimates for junction 41.

CHAPTER 5

EXPERIMENTS

5.1 Environment and Configurations

Table 5.1: Configuration Settings For LSTM Model

Parameters	Values
Train-Test Split	0.8
Epochs	200
Batch Size	1024
Loss Function	Mean Squared Error
Optimizer	Adam
Dropout	0.2
Output Size	1

We have used a computer which has following specifications:

- Intel Xeon(R) CPU E5-2620 v4 @ 2.10GHz 2.10GHz (2 processors)
- Nvidia 64 GB GPU
- 128 GB SSD

We have created a LSTM model which has following layer structure:

- 12 input node

- 128 hidden node - 0.2 dropout
- 128 hidden node
- 128 hidden node - 0.2 dropout
- 1 output node

The data is divided into two and 80% of the data is given to LSTM at the training part. After completing this training, the resulting model structure is saved in a file. In the remaining 20% of the data, we test how accurately this model works. We have trained our LSTM module for 160 epoch and this training part was taking approximately 8 hours. We have tried different epoch counts and decided what to use with elbow method since after 160 epoch there was only a slight decrease on the Mean Absolute Error.

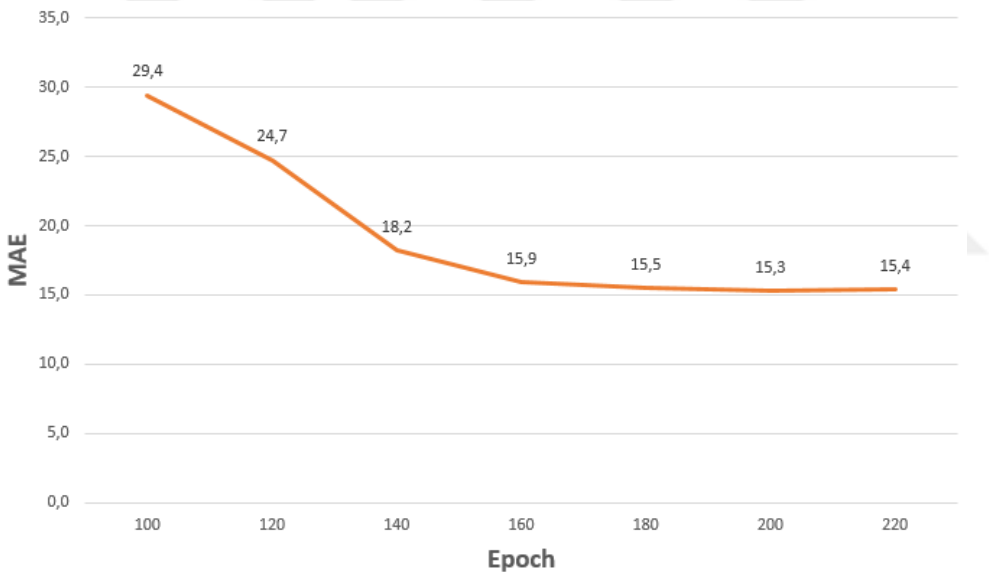


Figure 5.1: "MAE vs Epoch Graph"

5.2 Error Functions

One of the main factors limiting the use of neural networks is the difficulty of demonstrating that a trained network will continue to produce reliable results in real-life use [4]. It is observed that a network gives reliable results when data similar to the

data used during training are presented, but when new data is presented, there is a significant difference between the expected value and the output value.

Error functions are an important part of the network used to measure inconsistencies. The robustness of the network model takes a non-negative value that varies inversely with the result of the error function. The most commonly used error functions, the choices of which change according to the problem that the network seeks for a solution, are explained below.

In our study, we chose the values obtained with Moving Average and Three Weeks Average as our base. We determined the amount of error in the results obtained with these methods by the MSE, MAE and RMSE error functions. We tried to reach the best result by comparing the error rates we found in our own study with the error rates of these methods.

5.2.1 MSE

The performance of the network is observed by summing the squares of the difference between the expected output value(x_i) of the network and the obtained output value(y_i). This method is called MSE [3] and is defined by the formula in Equation 5.1:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2 \tag{5.1}$$

In Equation 5.1 [3], the main goal of MSE is to minimize the sum of the differences between the expected value and the obtained output values [5].

5.2.2 MAE

In order to measure the success of a trained NN, the predicted result should be compared with the actual result [1]. In Equation 5.2, y_i represents the expected value and x_i represents the actual measurement value. For each measurement value in the

test data set, the absolute values of the difference between the actual and suggested values are added and the MAE value is calculated by dividing it by the number of measurements in the dataset.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - x_i| \tag{5.2}$$

5.2.3 RMSE

The root mean square error (RMSE) is the standard deviation of the residuals (prediction errors). The distance between the data points and the regression line is measured by residuals, and the RMSE is a measure of how spread out these residuals are. To put it another way, it shows how closely the data is grouped around the line of greatest fit.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2} \tag{5.3}$$

5.3 Model Steps

While training our LSTM model, we used Keras as a backend in Python. The following procedures were used to create our model:

• Step1: Obtain the dataset and loaded all of the relevant Python libraries.

• Step2: The next stage in our preprocessing is to find and replace null values in our dataset, which make up a small percentage of the total data. To provide an accurate result, blank records have been replaced with the historical average value of the same day of the past three weeks.

• Step3: Normalization is a crucial step that allows you to scale your data between 0 and 1 when training and analyzing it. It will be tough to compare statistics if the data is really large.

• Step4: We divided our dataset into two halves (training and test), each with 80% and 20% of the data, and calculated the model input and output values.

• Step5: The following step is to create a model by defining all of the parameters, such as defining the number of layers.

• Step6: LSTM model has been trained and model is ready to give predictions abour vehicle counts.

• Step7: We create anomally-based rule definitions and used them to improve LSTM's predictions.

5.4 Experiment Results

When we started the tests, we first started with a simple LSTM structure. We had data on the number of vehicles measured at 15-minute intervals at 52 junctions. Measurements were not taken between 2 and 6 am, this part of the data was blank. Apart from that, we had 4 data for 20 hours. Therefore, the number of vehicles was measured 80 times per day.

When starting the tests, we had to decide whether we would build a model for each intersection or a single LSTM model that did the overall prediction. We made this decision in favor of creating a single model that makes predictions, and we gave the intersection ids as input to our model and enabled the model to learn information specific to junctions while learning.

We used 3 error functions to test how well the model performed; MAE, MSE, RMSE. We determined the differences between the estimated number of cars and the actual number of cars using these 3 methods. We compared the results obtained from these methods with 2 methods. We used the two simplest methods, averaging and moving average, when estimating a data sequence. When calculating the average, we used the number of vehicles on this road this day and at this hour in the previous 3 weeks. When calculating the moving average, we used the last 3 data from that day.

5.4.1 Base LSTM Experiment

When we made an estimation with the LSTM structure we established, we extracted the error results with MSE, MAE and RMSE, and we also made these error calculations for the results obtained when we estimated with the Moving Average and Three Weeks Average. We have obtained MSE, MAE and RMSE results for all junctions, took average of them and put in the Table 5.2. In order to understand better these results, the information that average vehicle count on these roads which is 174.2 is a crucial point. You can find the estimation of the number of vehicles in a day for the 39th intersection in graph 5.2.

Table 5.2: Experiment-1 Results

LSTM			Moving Average			Three Weeks Average		
MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
35.5	2561.2	276.9	276.9	2978.0	304.3	27.5	1832.3	207.3

5.4.2 Training LSTM With Updated Input Experiment

We had obtained these results only with the information we received from the sensor, and we thought that there was information we could add to this information. When estimating the number of vehicles to be found at the intersection, the number of vehicles at this intersection only 1-2 hours ago was not effective. We thought that if we enriched the given inputs, we would increase the sensitivity of the results we received. For this reason, we thought about what information we could add and decided to add the day information first. There was information about the day of the month in the data, but this was not enough. Because the traffic at an intersection was related to the day of the week rather than the day of the month. We planned to input this information into LSTM by numbering the days like an enum. Starting from Monday, we gave the days numbers 1, 2, 3... In this way, we would help the LSTM structure to distinguish Friday from Wednesday. Because there could be a traffic jam specific to Friday at the junctions and missing this information could make our prediction for

Fridays incorrect.

In addition, we thought that it would be valuable to provide information on which days are holidays as input. Holiday information should include not only weekday and weekend information, but also public holiday information. We have divided this information into 3 enums. We marked non-holiday days as 1, weekends as 2, and public holidays as 3 and fed the system in this way.

Thanks to the extra information we added, we expected our LSTM model to be able to better predict the increase in city traffic specific to the start of the week or the decrease in urban traffic specific to the Ramadan holiday. The numbers we obtained proved this situation and they are shown on Table 5.3. We have seen that the more accurate inputs we feed the LSTM structure, the more sensitive and accurate information we will obtain.

Table 5.3: Experiment-2 Results

LSTM			Moving Average			Three Weeks Average		
MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
24.7	1972.3	209.1	276.9	2978.0	304.3	27.5	1832.3	207.3

5.4.3 Training LSTM With Similar Junctions Together Experiment

In these studies we have done, we have created a model structure that tries to train itself separately for each intersection. However, each of these junctions did not contain separate structures from each other. For example, there should have been more than one intersection where the number of vehicles generally increased at 8 on weekdays or 6 in the afternoon, that is, during working hours, and with a generally stable number of vehicles at other times. Apart from this overtime example, we thought that there should be intersection clusters that resemble each other in terms of intersection structure and where the increase/decrease in the number of vehicles at these junctions is similar to each other. Also, it wouldn't make sense to continue this study on so many junctions, creating a smaller cluster and advancing the tests on these would

give us more accurate information in terms of whether our studies were useful or not. For this reason, we decided that it would be more beneficial to train LSTM in this way by grouping these junctions.

After researching various methods to group the junctions within themselves, we decided that the Hierarchical Clustering algorithm would be the most suitable for this process. Because our main goal was to create a subset of junctions that are similar to ourselves, rather than dividing all junctions into separate clusters. In this way, we would have more than one intersection data that we used for training a junction and we would have an LSTM structure trained with a larger data.

While doing this clustering, we obtained the following hierarchical clustering scheme.

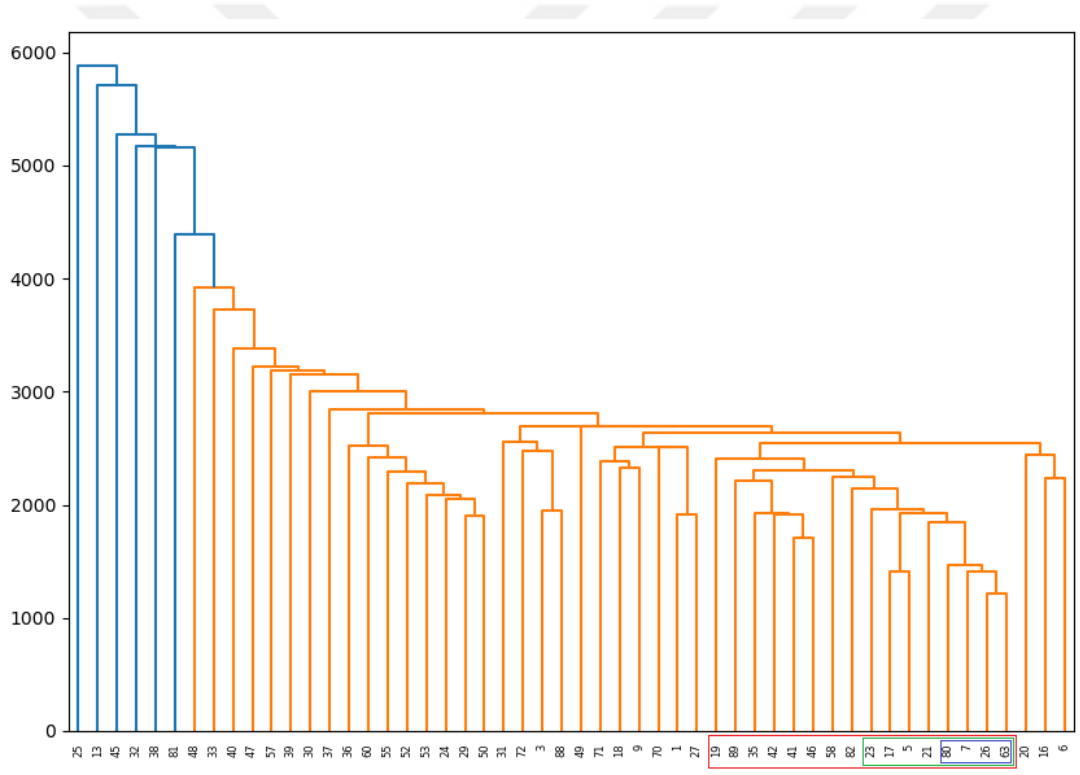


Figure 5.2: "Hierarchical Clustering"

By looking at this diagram, we started from the bottom intersection and we created 3 subsets. First of all, there is a subset that we created by combining the data of 4 junctions and it consists of junctions 80, 7, 26 , 63. The 2nd sub-cluster is formed by the merging of 8 junctions and is formed by adding the id 23, 17, 5 and 21 junctions

on top of the 1st cluster. Finally, our largest cluster is formed by adding the 19, 89, 35, 42, 41, 46, 58, and 82 junctions on top of the 2nd cluster, and it is a subset formed by the combination of 16 junctions in total. These clusters are painted with different colors on the figure also you can find the all the subclusters' elements on Table 5.4.

Table 5.4: Subclusters

Cluster Id	Color	Junctions
1	Blue	80, 7, 26, 63
2	Green	80, 7, 26, 63, 23, 17, 5, 21
3	Red	80, 7, 26, 63, 23, 17, 5, 21, 19, 89, 35, 42, 41, 46, 58, 82

We performed tests to see if the LSTM models trained with these 3 subsets we created give better results. We had 4 trials in our test. In our model, where we give each intersection with its own ID as input to LSTM, the accuracy rate we obtained from the 26^{th} intersection is 1. ,2. and our 3rd subsets. We present these subsets as S1, S2 and S3 on the Table 5.4. You can find the results of all these in table 5.5 below.

Table 5.5: Experiment-3 Results

Exp. Set	LSTM			Moving Avg			Three Weeks Avg		
	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
26	24.7	1872.3	209.1	276.9	2978.0	304.3	27.5	1832.3	207.3
S1	24.1	2364.8	213.8	249.1	2465.7	308.7	31.2	1653.1	197.1
S2	19.8	1249.3	174.9	212.3	1979.1	283.5	22.3	1597.7	167.3
S3	18.7	1354.2	168.2	255.7	2378.2	277.6	24.7	1324.8	187.4

By adding new information to the inputs we gave to the LSTM and growing our data by grouping the junctions, we started to get better results than the LSTM structure

we established at the beginning. Thanks to the clustered junctions, we have ensured that a situation that occurs less at one intersection and more at the other intersection is learned more properly by LSTM. The model we built has become more familiar with possible traffic situations because it has been trained not only with its own data, but also with the data of other junctions that are structurally similar to it.

5.4.4 Improved LSTM With Anomaly Based Rules Experiment

After all these studies, our LSTM structure had reached a sufficient level of accuracy, but the only shortcoming was that we could not give examples from the past. The situations I mentioned as situations that we cannot give examples from the past are actually anomaly situations. I'm talking about situations that are unique to that moment without a reason for that situation to occur. These can be caused by traffic accidents, road works or various other reasons.

Of course, it is not possible to detect anomaly situations, but it is possible to predict the effects of these anomaly situations in the future. For example, we can estimate how a traffic jam at junction 39 might have an impact on other junctions.

After this study, we tried to improve the predictions we made with LSTM by using the knowledge of how junctions affect each other during anomaly times. For this, we again chose a subset for ourselves. We started to work on a subset of 4 adjacent junctions.

We examined each intersection hourly and divided the anomaly cases into Anomaly+, Anomaly-. While making this determination, we looked at the average number of vehicles found at these junctions on the same days in the previous 5 weeks and took the average of the 3 in the middle without taking into account the highest and lowest number of vehicles in these weeks. In this way, we tried to find the average vehicle average of this intersection without adding extreme values. If the number of vehicles formed this weekday is more than 30% than this average, we marked this situation as anomaly+, if less than 30%, as anomaly-, if it is within -30% - +30% as normal. All junctions and hours were hard to put here but we put a part of the table in order to be an example of our work. You can see 39^{th} junction's anomaly counts for the

07:00-11:00(Table 5.6) and 16:00-20:00(Table 5.7) intervals.

Table 5.6: Anomaly Counts For Junction 39 (07:00-11:00)

Days	07:00-08:00	08:00-09:00	09:00-10:00	10:00-11:00
M	3	2	18	5
T	5	6	12	4
W	3	1	4	7
T	6	2	12	6
F	7	5	16	8
S	11	2	2	10
S	4	3	5	3

Table 5.7: Anomaly Counts For Junction 39 (16:00-18:00)

Days	16:00-17:00	17:00-18:00	18:00-19:00	19:00-20:00
M	3	8	7	8
T	6	8	12	6
W	4	7	4	5
T	3	4	7	4
F	7	5	11	7
S	3	4	5	6
S	4	3	5	8

After obtaining these tables, it was time to find the relationship between these tables. The decrease/increase in traffic at which intersection causes the decrease/increase in traffic at which intersection. Our aim was to find out on which days and at what times this relationship occurred. Because the increase in traffic on one road would not always affect the other in the same way. Where in the city people moved on that day or time would affect the direction of this effect. For this reason, we have divided these relationship tables into 4:

- Anomally+ → Anomally+

- Anomally+ → Anomally-

- Anomally- → Anomally+

- Anomally- → Anomally-

We have found the total anomaly counts and now it's time to find if these anomalies are triggered by other junctions' anomalies. You can find the anomaly+ situations in the junction 46 because of anomaly+ situation occured in junction 45 on the Table 5.8 and 5.9. There is a highlighted cell in the Table 5.8. It shows that 9(caused by 45) of the 16(total anomally) anomaly+ situtations are occured because of the junction 45's anomaly+ situation.

Table 5.8: Anomaly Counts For Junction 46 - Triggering by 45(07:00-11:00)

Days	07:00-08:00	08:00-09:00	09:00-10:00	10:00-11:00
M	0	2	6	0
T	0	0	5	0
W	3	1	4	1
T	0	2	1	1
F	0	0	**9(16)**	0
S	1	2	2	1
S	4	3	5	1

Table 5.9: Anomaly Counts For Junction 46 - Triggering by 45 (16:00-18:00)

Days	16:00-17:00	17:00-18:00	18:00-19:00	19:00-20:00
M	0	0	0	0
T	0	1	0	0
W	1	1	1	1
T	0	0	0	0
F	0	0	0	0
S	1	0	0	1
S	0	0	1	1

We have identified exactly 6 rules that affect this data as above. You can find out which rules can be applied to which junctions and effects of them on the vehicle counts, which days and hours and their probability of happening in the Table 5.10 below.

Table 5.10: Anomaly Rules Found in Neighbour Junctions

Triggering Junction	Triggered Junction	Day-Hour	Probability
39(Anomaly+)	41(Anomaly+)	Tuesday - 08:00-09:00	%67
45(Anomaly-)	39(Anomaly+)	Friday - 08:00-09:00	%62
45(Anomaly+)	46(Anomaly-)	Friday - 09:00-10:00	%56
39(Anomaly+)	41(Anomaly+)	Sunday - 18:00-19:00	%55
45(Anomaly+)	46(Anomaly+)	Monday - 17:00-18:00	%54
41(Anomaly-)	39(Anomaly-)	Tuesday - 19:00-20:00	%51

After finding these rules we have specified a threshold in order to apply them. If we continue from an example, let's take first row. When there is a Anomaly+ situation on the 39^{th} junction on Tuesday at 08:15 we look for predictions at 08:30 and 08:45 for the 41^{st} junction. If it doesn't predict a vehicle count which can be evaluated as Anomaly+ then the mechanism steps in and adds %20 to the LSTM prediction. By doing this we have improved our prediction accuracies. You can find the updated results in terms of error parameters in Table 5.11.

Table 5.11: Experiment-4 Results

Exp. Set	LSTM			LSTM+Anomaly Based Rule		
	MAE	MSE	RMSE	MAE	MSE	RMSE
26	24.7	1872.3	209.1	23.6	1692.1	203.8
S1	24.1	2364.8	213.8	23.8	2125.0	215.4
S2	19.8	1249.3	174.9	18.2	1342.2	168.2
S3	18.7	1354.2	168.2	15.9	1476.9	172.1

After obtaining these results, we wanted to see how the results would change when we changed the threshold value we tried. We found the results by setting the 30% threshold. We wanted to find out how the 20% and 40% thresholds would contribute to the mechanism we established, and we obtained the following table.

Table 5.12: Anomaly Based Rule Threshold Experiment Results

Exp. Set	LSTM+Anomaly Based Rule (Threshold = %20)			LSTM+Anomaly Based Rule (Threshold = %30)			LSTM+Anomaly Based Rule (Threshold = %40)		
	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
26	25.3	1702.7	208.2	23.6	1692.1	203.8	24.3	1781.4	205.7
S1	27.1	2495.0	271.2	23.8	2125.0	215.4	24.5	2164.2	203.9
S2	16.2	1387.9	154.2	18.2	1342.2	168.2	18.9	1349.1	172.1
S3	18.6	1561.9	182.8	15.9	1476.9	172.1	18.4	1314.7	173.7

As can be seen from the table, the results we obtained with 30% seem to be the results that improved the LSTM model the most. When we set the Threshold to 40%, we were able to find only 2 rules that could update the prediction of the lstm model, and this caused us not to contribute much to the lstm model. When we made the

Threshold 20%, although we found 11 rules in total, the contribution of these rules to our lstm model was not always good, sometimes it made the predictions worser. So we decided it would be best to use this threshold as 30%.

In order to measure how successful our predictions were, we calculated the success percentages of the predictions made with ARIMA, SVR, LSTM and LSTM+Anomaly Based Rule Mechanism. While converting the estimations to accuracy, we converted the error differences found with the MAE method to accuracy. While doing this, we calculated the average number of vehicles found at the intersections during the day and the result was 174.7. We saw that we achieved a success rate of %90.9 when we compared the error with the MAE method to this average number. You can find detailed information in table 5.13.

Table 5.13: Prediction Performances

Models	Accuracy
ARIMA	%65.2
SVR	%78.4
LSTM	%88.7
LSTM+Anomaly Based Rule Mechanism	%90.9

CHAPTER 6

CONCLUSION

Various studies on traffic forecasting have been conducted so far, and useful implications for this area have been obtained from each. Studies with LSTM are also proof that useful results can be obtained with this method in this field. When we started this study, we set out with the aim of taking these studies one step further by putting a new layer on these works done so far. It was important at the beginning of this study to understand what LSTM has good and what can be improved sides and to direct the work accordingly. We knew that in order for LSTM to work properly, it had to take the necessary inputs and be fed as rich of information as possible. At the beginning of our work, we expanded the information that could improve our model in order to get the maximum efficiency we could get from LSTM and ensured that the dataset we gave was varied and rich enough to enable the model to work properly. After seeing that a well-fed model produced successful results, we put the rule-base anomally rules on top of this model, which is the improvement we actually want. We created these rules by examining the anomally occurring situations at neighboring intersections and detecting the relationship between these situations. Thanks to these rules, we have determined how the anomaly situation at an intersection will affect which intersections in the future. In this way, we have ensured that this anomaly situation does not overturn the traffic of the entire city, and we have determined in advance what will cause the continuation of the realized anomaly situation. When we added these anomally cases where it is not possible to teach the model to the predictions we made with our LSTM model, the resulting improvements were satisfactory.

The methods that can take this study further can be divided into two; to improve the LSTM model we created and to improve the anomaly rule-based development mod-

ule we added on it. More detailed information about intersections can be given as input for the development of LSTM. Instant information such as meteorological information about the road, rainy/snowy weather information can improve the LSTM's forecast. In the Anomally rule-based part, more detailed rules can be extracted. More sensitive rules can be defined if it is determined that the traffic increase/decrease on which road will cause the traffic increase/decrease at other intersections if the number of vehicles at the intersections is not used as a total but by separating the road. Thanks to these methods, both a better LSTM model and a better anomally-based rule set can be created and more accurate results can be obtained. As a future work LSTM part of the mechanism can be replaced with Gated Recurrent Units which a new concept that is interchangeable with LSTM.

REFERENCES

[1] Wang, J., Gu, Q., Wu, J., Liu, G., and Xiong, Z., "Traffic speed prediction and congestion source exploration: A deep learning method", Data Mining (ICDM), IEEE 16th International Conference, 499–508 (2016).

[2] Patterson, J. and Gibson, A., "Deep Learning: A practitioner's Approach", O'Reilly Press, ISBN:978-1-491-91425-0, 41-114, 125-165 (2017).

[3] Demuth, H., Beale, M. and Hagan, M., "User Guide, Neural Networks ToolboxTM 6", The MathWorks, Inc., ISBN:0-9717321-0-8, 2-25, 3-5, 8-6 (2009).

[4] Bishop, C., "Novelty detection and neural network validation", Proceedings of the IEEE Conference on Vision, Image and Signal Processing, 141, IET, 217–222 (1994).

[5] Hagan, M.T., Demuth, H.B. and Beale, M., "Neural Network Design", Boston, PWS Publishing Co., 10-04 (1996).

[6] Goyal, P., Pandey, S., and Jain, K., "Deep Learning for Natural Language Processing", Apress, 35 -75, 119-168 (2018).

[7] Buduma, N. and Locascio, N., "Fundamentals of Deep Learning. Designing Next-Generation Machine Intelligence Algorithms", O'Reilly Media, 172-217 (2017).

[8] Hochreiter, S. and Schmidhuber J., "Long short-term memory", Neural computation, 9(8):1735–1780 (1997).

[9] Pattanayak, S., "Pro Deep Learning with TensorFlow", Apress, New York, ISBN 978-1-4842-3095-4, 153-278 (2017).

[10] Krizhevsky, A., Sutskever, I. and Hinton, G., "Imagenet classification with deep convolutional neural networks", NIPS (2012).

[11] Chung, J., Gülçehre, Ç., Cho, K., and Bengio, Y., "Empirical evaluation of gated recurrent neural networks on sequence modeling", CoRR, abs/1412.3555 (2014).

[12] I. Witten, E. Frank, M. Kaufmann, and J. Geller, "Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations," 2000. Accessed: Jul. 14, 2021. [Online]. Available: https://web.archive.org/web/20170813015421id_/https://sigmodrecord.org/publications/sigmodRecord/0203/bookreview2-geller.pdf.

[13] Han, J., Kamber, M. and Pei,J., 2011, Data Mining Concept and Techniques (Third Edition), Morgan Kaufmann Publishers, Waltham 703p

[14] Halkidi, M., Batistakis, Y. and Vazirgiannis, M., 2001, On Clustering Validation Techniques, Journal of Intelligent Information Systems, 17 : 107–145 pp.

[15] Jain, A.K. and Dubes, R.C., 1988, Algorithms for Clustering Data, Prentice Hall, New Jersey, 320 p.

[16] Xu, R., Wunsch, D., 2005, Survey of Clustering Algorithms. IEEE Transactions on Neural Networks, 16(3), 645–678 pp.

[17] Zaki, M.J. and Meira, W.J., 2014, Data Mining and Analysis: Fundamental Concept and Algorithms, Cambridge University Press, New York, 562p.

[18] Berkhin, P., 2006, Survey Of Clustering Data Mining Techniques, Accure Software, San Jose CA, 56p.

[19] Everitt, B.S., Landau, S., Leese, M. And Stahl, D., 2011, Cluster Analysis 5th Edition, John Wileys & Sons, West Sussex, 330p.

[20] Tan, P., Steinbach, M. and Kumar, V., 2005, Introduction to Data Mining Chapter 8: Cluster Analysis: Basic Concepts and Algorithms, Pearson Addison-Wesley, Boston, 487-568 pp.

[21] Viswanath, P. And Babu, S., 2009, Rough-DBSCAN: A fast hybrid density based clustering method for large data sets, Pattern Recognation Letters, 30(16): 1477-1488 pp.

[22] Larose, D.T., 2005, Discovering Knowledge In Data: An Introduction to Data Mining, John Wileys & Sons, New Jersey, 222p.

[23] Gorunescu, F., 2010, Data Mining: Concepts, Models and Techniques, Springer, Berlin, 357p.

[24] Camus, R., Cantarella, G. E., & Inaudi, D. (1994). O-D prediction for real-time traffic management. IFAC Proceedings Volumes, 27(12), 707–711. https://doi.org/10.1016/S1474-6670(17)47555-0

[25] Crittin, F., & Bierlaire, M. (2002). An efficient algorithm for real-time estimation and prediction of dynamic OD table. In Swiss Transport Research Conference.

[26] Qi, Y., & Ishak, S. (2013). Stochastic approach for short-term freeway traffic prediction during peak periods. IEEE Transactions on Intelligent Transportation Systems, 14(2), 660–672. https://doi.org/10.1109/TITS.2012.2227475

[27] Hamed, M. M., Al-Masaeid, H. R., & Said, Z. M. B. (1995). Short-term prediction of traffic volume in urban arterials. Journal of Transportation Engineering, 121(3), 249–254.

[28] Daganzo, C. F. (1994). The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. Transportation Research Part B: Methodological, 28(4), 269–287.

[29] Smith, B. L., & Demetsky, M. J. (1994). Short-term traffic flow prediction: Neural network approach. Transportation Research Record, (1453). https://trid.trb.org/view/424677

[30] Smith, B., & Demetsky, M. (1996). Multiple-interval freeway traffic flow forecasting. Transportation Research Record: Journal of the Transportation Research Board, (1554), 136–141.

[31] Smith, B. L., & Oswald, R. K. (2003). Meeting real-time traffic flow forecasting requirements with imprecise computations. Computer-Aided Civil and Infrastructure Engineering, 18(3), 201–213. https://doi.org/10.1111/1467-8667.00310

[32] Smith, B. L., Williams, B. M., & Oswald, R. K. (2002). Comparison of parametric and nonparametric models for traffic flow forecasting.

[33] You, J., & Kim, T. J. (2000). Development and evaluation of a hybrid travel time forecasting model. Transportation Research Part C: Emerging Technologies, 8(1–6), 231–256.

[34] Clark, S. (2003). Traffic prediction using multivariate nonparametric regression. Journal of Transportation Engineering, 129(2), 161–168.

[35] Chen, C., Hu, J., Meng, Q., & Zhang, Y. (2011). Short-time traffic flow prediction with ARIMA-GARCH model. In 2011 IEEE Intelligent Vehicles Symposium (IV) (pp. 607–612). https://doi.org/10.1109/IVS.2011.5940418

[36] Zeng, X., & Zhang, Y. (2013). Development of recurrent neural network considering temporal-spatial input dynamics for freeway travel time modeling. Computer-Aided Civil and Infrastructure Engineering, 28(5), 359–371. https://doi.org/10.1111/mice.12000

[37] Zhong, J., & Ling, S. (2015). Key factors of k-nearest neighbours nonparametric regression in short-time traffic flow forecasting. In Proceedings of the 21st International Conference on Industrial Engineering and Engineering Management 2014 (pp. 9–12). Paris: Atlantis Press. https://doi.org/10.2991/978-94-6239-102-4_2

[38] Oswald, R. K., Scherer, W. T., & Smith, B. L. (2000). Traffic flow forecasting using approximate nearest neighbor nonparametric regression. Retrieved from https://trid.trb.org/view/661764

[39] Kindzerske, M., & Ni, D. (2007). Composite nearest neighbor nonparametric regression to improve traffic prediction. Transportation Research Record: Journal of the Transportation Research Board, 1993, 30–35. https://doi.org/10.3141/1993-05

[40] Yildirim, U., & Cataltepe, Z. (2008). Short time traffic speed prediction using data from a number of different sensor locations. In 2008 23rd International Symposium on Computer and Information Sciences (pp. 1–6). https://doi.org/10.1109/ISCIS.2008.4717955

[41] Chen, C., Hu, J., Meng, Q., & Zhang, Y. (2011). Short-time traffic flow predic-

tion with ARIMA-GARCH model. In 2011 IEEE Intelligent Vehicles Symposium (IV) (pp. 607–612). https://doi.org/10.1109/IVS.2011.5940418

[42] Chen, D. (2017). Research on traffic flow prediction in the big data environment based on the improved RBF neural network. IEEE Transactions on Industrial Informatics, 13(4), 2000–2008. https://doi.org/10.1109/TII.2017.2682855

[43] Chen, H., Grant-Muller, S., Mussone, L., & Montgomery, F. (2001). A study of hybrid neural network approaches and the effects of missing data on traffic forecasting. Neural Computing & Applications, 10(3), 277–286. https://doi.org/10.1007/s521-001-8054-3

[44] Chen, X. Y., Pao, H. K., & Lee, Y. J. (2014). Efficient traffic speed forecasting based on massive heterogeneous historical data. In 2014 IEEE International Conference on Big Data (Big Data) (pp. 10–17). https://doi.org/10.1109/BigData.2014.7004425

[45] Chen, Y., Lv, Y., Li, Z., & Wang, F. (2016). Long short-term memory model for traffic congestion prediction with online open data. In 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 132–137. IEEE.

[46] Kwon, E., & Stephanedes, Y. J. (1994). Comparative evaluation of adaptive and neuralnetwork exit demand prediction for freeway control. Transportation Research Record, 66–66.

[47] Yun, S.-Y., Namkoong, S., Rho, J.-H., Shin, S.-W., & Choi, J.-U. (1998). A performance evaluation of neural network models in traffic volume forecasting. Mathematical and Computer Modelling, 27(9–11), 293–310.

[48] Guo, J., Huang, W., & Williams, B. M. (2014). Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification. Transportation Research Part C: Emerging Technologies, 43, 50–64. https://doi.org/10.1016/j.trc.2014.02.006

[49] Lee, E.-M., Kim, J.-H., & Yoon, W.-S. (2007). Traffic speed prediction under weekday, time, and neighboring links' speed: Back propagation neural network approach. In Advanced Intelligent Computing Theories and Applications.

With Aspects of Theoretical and Methodological Issues (pp. 626–635). Springer, Berlin, Heidelberg.https://doi.org/10.1007/978-3-540-74171-8_62

[50] Lee, S., & Fambro, D. (1999). Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting. Transportation Research Record: Journal of the Transportation Research Board, (1678), 179–188.

[51] Lee, S., Lee, Y.-I., & Cho, B. (2006). Short-term travel speed prediction models in car navigation systems. Journal of Advanced Transportation, 40(2), 122–139. https://doi.org/10.1002/atr.5670400203

[52] Lee, Y. (2009). Freeway travel time forecast using artificial neural networks with cluster method. In 2009 12th International Conference on Information Fusion (pp. 1331–1338).

[53] Fabritiis, C. de, Ragona, R., & Valenti, G. (2008). Traffic estimation and prediction based on real time floating car data. In 2008 11th International IEEE Conference on Intelligent Transportation Systems (pp. 197–203). https://doi.org/10.1109/ITSC.2008.4732534

[54] Park, J., Li, D., Murphey, Y. L., Kristinsson, J., McGee, R., Kuang, M., & Phillips, T. (2011). Real time vehicle speed prediction using a neural network traffic model. In The 2011 International Joint Conference on Neural Networks (pp. 2991–2996). https://doi.org/10.1109/IJCNN.2011.6033614

[55] Ye, Q., Szeto, W. Y., & Wong, S. C. (2012). Short-term traffic speed forecasting based on data recorded at irregular intervals. IEEE Transactions on Intelligent Transportation Systems, 13(4), 1727–1737. https://doi.org/10.1109/TITS.2012.2203122

[56] Habtie, A. B., Abraham, A., & Midekso, D. (2017). Artificial neural network based real-time urban road traffic state estimation framework. In Computational Intelligence in Wireless Sensor Networks (pp. 73–97). Springer, Cham. https://doi.org/10.1007/978-3-319-47715-2_4

[57] Abdulhai, B., Porwal, H., & Recker, W. (1999). Short term freeway traffic

flow prediction using genetically-optimized time-delay-based neural networks. https://escholarship.org/uc/item/4t05p2mp

[58] Lingras, P., & Mountford, P. (2001). Time delay neural networks designed using genetic algorithms for short term inter-city traffic forecasting. In Engineering of Intelligent Systems (pp. 290–299). Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-45517-5_33

[59] Wang, H., Liu, L., Qian, Z., Wei, H., & Dong, S. (2014). Empirical mode decompositionautoregressive integrated moving average: Hybrid short-term traffic speed prediction model. Transportation Research Record: Journal of the Transportation Research Board, 2460, 66–76. https://doi.org/10.3141/2460-08

[60] Wang, J., & Shi, Q. (2013). Short-term traffic speed forecasting hybrid model based on Chaos–Wavelet Analysis-Support Vector Machine theory. Transportation Research Part C: Emerging Technologies, 27, 219–232. https://doi.org/10.1016/j.trc.2012.08.004

[61] Wang, J., Gu, Q., Wu, J., Liu, G., & Xiong, Z. (2016). Traffic speed prediction and congestion source exploration: A deep learning method. In 2016 IEEE 16th International Conference on Data Mining (ICDM), pp. 499–508. IEEE.

[62] Wang, Y., Papageorgiou, M., & Messmer, A. (2006). RENAISSANCE—A unified macroscopic model-based approach to real-time freeway network traffic surveillance. Transportation Research Part C: Emerging Technologies, 14(3), 190–212. https://doi.org/10.1016/j.trc.2006.06.001

[62] Wang, Y., Wang, H., & Xia, L. (2005). Highway traffic prediction with neural network and genetic algorithms. In IEEE International Conference on Vehicular Electronics and Safety, pp. 211–216. IEEE.

[63] Vlahogianni, E. I., Karlaftis, M. G., & Golias, J. C. (2005). Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach. Transportation Research Part C: Emerging Technologies, 13(3), 211–234. https://doi.org/10.1016/j.trc.2005.04.007

[64] Lingras, P., & Mountford, P. (2001). Time delay neural networks designed using genetic algorithms for short term inter-city traffic forecasting. In En-

gineering of Intelligent Systems (pp. 290–299). Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-45517-5_33

[65] Park, D., & Rilett, L. (1998). Forecasting multiple-period freeway link travel times using modular neural networks. Transportation Research Record: Journal of the Transportation Research Board, (1617), 163–170.

[65] Jiang, X., & Adeli, H. (2005). Dynamic wavelet neural network model for traffic flow forecasting. Journal of Transportation Engineering, 131(10), 771–779.

[66] Jiang, H., Zou, Y., Zhang, S., Tang, J., & Wang, Y. (2016). Short-term speed prediction using remote microwave sensor data: Machine learning versus statistical model. Mathematical Problems in Engineering, vol. 2016. https://doi:10.1155/2016/9236156

[67] Xie, Y., & Zhang, Y. (2006). A wavelet network model for short-term traffic volume forecasting. Journal of Intelligent Transportation Systems, 10(3), 141–150. https://doi.org/10.1080/15472450600798551

[68] Xie, Y., Zhang, Y., & Ye, Z. (2007). Short-term traffic volume forecasting using Kalman filter with discrete wavelet decomposition. Computer-Aided Civil and Infrastructure

[69] Boto-Giralda, D., Díaz-Pernas, F. J., González-Ortega, D., Díez-Higuera, J. F., Antón-Rodríguez, M., Martínez-Zarzuela, M., & Torre-Díez, I. (2010). Wavelet-based denoising for traffic volume time series forecasting with self-organizing neural networks. Computer-Aided Civil and Infrastructure Engineering, 25(7), 530–545. https://doi.org/10.1111/j.1467-8667.2010.00668.x

[70] Amin, S. M., Rodin, E. Y., Liu, A.-P., Rink, K., & García-Ortiz, A. (1998). Traffic prediction and management via RBF neural nets and semantic control. Computer-Aided Civil and Infrastructure Engineering, 13(5), 315–327. https://doi.org/10.1111/0885-9507.00110

[71] Park, B., Messer, C., & Urbanik II, T. (1998). Short-term freeway traffic volume forecasting using radial basis function neural network. Transportation Research Record: Journal of the Transportation Research Board, (1651), 39–47.

[72] Zheng, W., Lee, D.-H., & Shi, Q. (2006). Short-term freeway traffic flow prediction: Bayesian combined neural network approach. Journal of Transportation Engineering, 132(2), 114–121.

[73] Yin, H., Wong, S. C., Xu, J., & Wong, C. K. (2002). Urban traffic flow prediction using a fuzzy-neural approach. Transportation Research Part C: Emerging Technologies, 10(2), 85–98. https://doi.org/10.1016/S0968-090X(01)00004-3

[74] Coufal, D., & Turunen, E. (2003). Short term prediction of highway travel time using data mining and neuro-fuzzy methods. In Proc. IFSA (pp. 175–182).

[75] Jiang, H., Zou, Y., Zhang, S., Tang, J., & Wang, Y. (2016). Short-term speed prediction using remote microwave sensor data: Machine learning versus statistical model. Mathematical Problems in Engineering, vol. 2016. https://doi:10.1155/2016/9236156

[76] Yasdi, R. (1999). Prediction of road traffic using a neural network approach. Neural Computing & Applications, 8(2), 135–142. https://doi.org/10.1007/s005210050015

[77] Dia, H. (2001). An object-oriented neural network approach to short-term traffic forecasting. European Journal of Operational Research, 131(2), 253–261. https://doi.org/10.1016/S0377-2217(00)00125-9

[77] Ishak, S., Kotha, P., & Alecsandru, C. (2003). Optimization of dynamic neural network performance for short-term traffic prediction. Transportation Research Record: Journal of the Transportation Research Board, (1836), 45–56.

[78] Moretti, F., Pizzuti, S., Panzieri, S., & Annunziato, M. (2015). Urban traffic flow forecasting through statistical and neural network bagging ensemble hybrid modeling. Neurocomputing, 167, 3–7. https://doi.org/10.1016/j.neucom.2014.08.100

[79] Alecsandru, C., & Ishak, S. (2004). Hybrid model-based and memory-based traffic prediction system. Transportation Research Record: Journal of the Transportation Research Board, (1879), 59–70.

[80] Williams, B. (2001). Multivariate vehicular traffic flow prediction: Evaluation of ARIMAX modeling. Transportation Research Record: Journal of the Transportation Research Board, (1776), 194–200.

[81] Williams, B. M., & Hoel, L. A. (2003). Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. Journal of Transportation Engineering, 129(6), 664–672.

[82] Williams, B., Durvasula, P., & Brown, D. (1998). Urban freeway traffic flow prediction: Application of seasonal autoregressive integrated moving average and exponential smoothing models. Transportation Research Record: Journal of the Transportation Research Board, (1644), 132–141.

[83] Kamarianakis, Y., & Prastacos, P. (2003). Forecasting traffic flow conditions in an urban network: Comparison of multivariate and univariate approaches. Transportation Research Record: Journal of the Transportation Research Board, (1857), 74–84.

[84] Kamarianakis, Y., Kanas, A., & Prastacos, P. (2005). Modeling traffic volatility dynamics in an urban network. Transportation Research Record: Journal of the Transportation Research Board, (1923), 18–27.

[85] Okutani, I., & Stephanedes, Y. J. (1984). Dynamic prediction of traffic volume through Kalman filtering theory. Transportation Research Part B: Methodological, 18(1), 1–11.

[86] Szeto, W. Y., Ghosh, B., Basu, B., & O'Mahony, M. (2009). Multivariate traffic forecasting technique using cell transmission model and SARIMA model. Journal of Transportation Engineering, 135(9), 658–667.

[87] Pavlyuk, D. (2017). Short-term traffic forecasting using multivariate autoregressive models. Procedia Engineering, 178, 57–66. https://doi.org/10.1016/j.proeng.2017.01.062

[88] Ghosh, B., Basu, B., & O'Mahony, M. (2009). Multivariate short-term traffic flow forecasting using time-series analysis. IEEE Transactions on Intelligent Transportation Systems, 10(2), 246–254. https://doi.org/10.1109/TITS.2009.2021448

[89] Cetin, M., & Comert, G. (2006). Short-term traffic flow prediction with regime switching models. Transportation Research Record: Journal of the Transportation Research Board, (1965), 23–31.

[90] Chien, S. I.-J., & Kuchipudi, C. M. (2003). Dynamic travel time prediction with real-time and historic data. Journal of Transportation Engineering, 129(6), 608–616.

[91] Chien, S., Liu, X., & Ozbay, K. (2003). Predicting travel times for the South Jersey real-time motorist information system. Transportation Research Record: Journal of the Transportation Research Board, (1855), 32–40.

[92] Yang, J.-S. (2005). Travel time prediction using the GPS test vehicle and Kalman filtering techniques. In Proceedings of the 2005, American Control Conference, 2005. (pp. 2128–2133 vol. 3). https://doi.org/10.1109/ACC.2005.1470285

[93] Lint, J. W. C. van. (2008). Online learning solutions for freeway travel time prediction. IEEE Transactions on Intelligent Transportation Systems, 9(1), 38–47. https://doi.org/10.1109/TITS.2008.915649

[94] Whittaker, J., Garside, S., & Lindveld, K. (1997). Tracking and predicting a network traffic process. International Journal of Forecasting, 13(1), 51–61.

[94] Yang, F., Yin, Z., Liu, H., & Ran, B. (2004). Online recursive algorithm for short-term traffic prediction. Transportation Research Record: Journal of the Transportation Research Board, (1879), 1–8.

[95] Chandra, S., & Al-Deek, H. (2008). Cross-correlation analysis and multivariate prediction of spatial time series of freeway traffic speeds. Transportation Research Record: Journal of the Transportation Research Board, 2061, 64–76. https://doi.org/10.3141/2061-08

[96] Chandra, S. R., & Al-Deek, H. (2009). Predictions of freeway traffic speeds and volumes using vector autoregressive models. Journal of Intelligent Transportation Systems, 13(2), 53–72. https://doi.org/10.1080/15472450902858368

[97] Zou, Y., Hua, X., Zhang, Y., & Wang, Y. (2015). Hybrid short-term freeway speed prediction methods based on periodic analysis. Canadian Journal of Civil Engineering, 42(8), 570–582. https://doi.org/10.1139/cjce-2014-0447

[98] AyonDey , “Machine Learning Algorithms: A Review”, (IJCSIT) International Journal of Computer Science

[99] Mitchell, T. (1997). Machine Learning. McGraw Hill. p. 2. ISBN 978-0-07-042807-2.