Traffic Prediction using deep learning

A MINI PROJECT REPORT

18CSC305J - ARTIFICIAL INTELLIGENCE

Submitted by

Gauri Tripathi RA2011003010144
Sparsh Patel RA2011003010148
Vaibhav Ganesh RA2011003010204

Under the guidance of

Ms. Rajalakshmi M.

Assistant Professor, Department of Computer Science and Engineering

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

S.R.M. Nagar, Kattankulathur, Chengalpattu District

MAY 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that Mini project report titled **“ Traffic Prediction using deep learning”** is the bona fide work of **Gauri Tripathi RA2011003010144, Sparsh Patel RA2011003010148 and Vaibhav Ganesh RA2011003010204** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
Ms. Rajalakshmi M.	Dr. M. Pushpalatha
GUIDE	**HEAD OF THE DEPARTMENT**
Assistant Professor	Professor & Head
Department of Computing Technologies	Department of Computing Technologies

ABSTRACT

Traffic prediction plays an essential role in intelligent transportation system. Accurate traffic prediction can assist route planing, guide vehicle dispatching, and mitigate traffic congestion. This problem is challenging due to the complicated and dynamic spatio-temporal dependencies between different regions in the road network. Recently, a significant amount of research efforts have been devoted to this area, especially deep learning method, greatly advancing traffic prediction abilities. The purpose of this paper is to provide a comprehensive survey on deep learning-based approaches in traffic prediction from multiple perspectives. Specifically, we first summarize the existing traffic prediction methods, and give a taxonomy. Second, we list the state-of-the-art approaches in different traffic prediction applications. Third, we comprehensively collect and organize widely used public datasets in the existing literature to facilitate other researchers. Furthermore, we give an evaluation and analysis by conducting extensive experiments to compare the performance of different methods on a real-world public dataset. Finally, we discuss open challenges in this field.

TABLE OF CONTENTS

List and Figures

ABBREVIATIONS

LSTM	Long Short Term Memory
CNN	convolutional neural networks
RMSE.	Root mean square error

CHAPTER 1

INTRODUCTION

Traffic prediction is an important area of research in transportation engineering and urban planning. Accurate traffic prediction enables efficient traffic management, which in turn reduces travel time, fuel consumption, and pollution. Deep learning is a type of machine learning that has shown promising results in various fields, including traffic prediction. Deep learning models can learn complex patterns from traffic data and make accurate predictions. These models can be trained on various types of traffic data, such as traffic flow, speed, and occupancy. The input data can be obtained from various sources, such as loop detectors, GPS devices, and cameras. There are various deep learning architectures that can be used for traffic prediction, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory (LSTM) networks. These architectures can be used for both short-term and long-term traffic prediction. Short-term traffic prediction involves predicting traffic conditions within a few minutes to an hour, while long-term traffic prediction involves predicting traffic conditions for a longer period, such as a day or a week. Short-term traffic prediction is important for real-time traffic management, while long-term traffic prediction is important for urban planning and infrastructure development. In addition to deep learning models, various other techniques can also be used for traffic prediction, such as statistical models, regression models, and time-series analysis. However, deep learning has shown superior performance compared to these traditional techniques in many cases.

Overall, deep learning-based traffic prediction is a promising area of research that has the potential to revolutionize the way we manage and plan urban transportation systems.

CHAPTER 2

LITERATURE SURVEY

Traffic prediction plays an essential role in intelligent transportation system. Accurate traffic prediction can assist route planing, guide vehicle dispatching, and mitigate traffic congestion. This problem is challenging due to the complicated and dynamic spatio-temporal dependencies between different regions in the road network. Recently, a significant amount of research efforts have been devoted to this area, especially deep learning method, greatly advancing traffic prediction abilities.The purpose of this paper is to provide a comprehensive surveyon deep learning-based approaches in traffic prediction from multiple perspectives. Specifically, we first summarize the existing traffic prediction methods, and give a taxonomy. Second, we list the state-of-the-art approaches in different traffic prediction applications.Third, we comprehensively collect and organize widely used public datasets in the existing literature to facilitate
other researchers. Furthermore, we give an evaluation and analysis by conducting extensive experiments to compare the performance of different methods on a real-world public dataset.Finally, we discuss open challenges in this field.

CHAPTER 3

SYSTEM ARCHITECTURE AND DESIGN

The system architecture and design for traffic prediction using deep learning can vary depending on the specific application and requirements. However, there are certain components that are common to most systems:

Data Collection: The first step in traffic prediction using deep learning is to collect the relevant data. This can include traffic flow, speed, occupancy, weather conditions, and other factors that may affect traffic. Data can be collected from various sources, such as sensors, cameras, GPS devices, and social media.

Data Preprocessing: Once the data is collected, it needs to be preprocessed to prepare it for input to the deep learning model. This can involve filtering out noise, scaling the data, and converting it to the appropriate format.

Deep Learning Model: The deep learning model is the core component of the system. The model is trained on the preprocessed data to learn the patterns and relationships in the data. There are various deep learning architectures that can be used for traffic prediction, such as CNNs, RNNs, and LSTMs.

Model Training: The deep learning model is trained on a dataset that includes historical traffic data and the corresponding traffic conditions. The model is trained using an optimization algorithm, such as stochastic gradient descent, to minimize the prediction error.

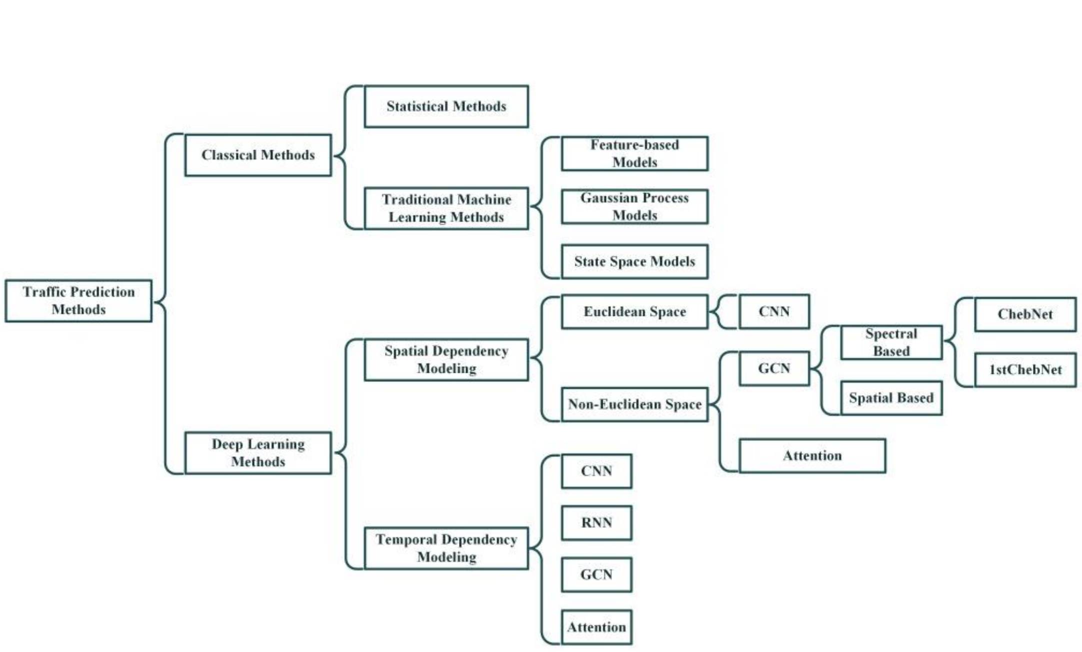
Model Evaluation: After the model is trained, it needs to be evaluated to ensure that it is accurate and reliable. This can be done using various metrics, such as mean absolute error

(MAE), mean squared error (MSE), and root mean squared error (RMSE).

Model Deployment: Once the model is trained and evaluated, it can be deployed in a production environment. The model takes in real-time traffic data as input and generates predictions for the future traffic conditions.

Visualization and Reporting: The predictions generated by the model can be visualized and reported to the users in a user-friendly format, such as a dashboard or a mobile app. This enables the users to make informed decisions based on the predicted traffic conditions.

In addition to these components, the system architecture may also include other features, such as real-time data processing, data storage, and integration with other systems. Overall, the system architecture and design for traffic prediction using deep learning should be flexible, scalable, and adaptable to different applications and requirements.



CHAPTER 4

METHODOLOGY

The methodology of traffic prediction using deep learning involves several steps:

Data Collection: The first step is to collect the traffic data. This can include traffic flow, speed, occupancy, and other factors that may affect traffic, such as weather conditions and road construction. The data can be collected from various sources, such as sensors, cameras, GPS devices, and social media.

Data Preprocessing: Once the data is collected, it needs to be preprocessed to prepare it for input to the deep learning model. This can involve filtering out noise, scaling the data, and converting it to the appropriate format. The data can also be divided into training, validation, and testing sets.

Deep Learning Model Selection: The next step is to select the appropriate deep learning model for the specific application. There are various deep learning architectures that can be used for traffic prediction, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory (LSTM) networks. The selection of the model depends on the type of data, the prediction horizon, and other factors.

Model Training: The deep learning model is trained on the preprocessed data to learn the patterns and relationships in the data. The model is trained using an optimization algorithm, such as stochastic gradient descent, to minimize the prediction error. The model is iteratively trained on the training set until the prediction error converges.

Model Evaluation: After the model is trained, it needs to be evaluated to ensure that it is accurate and reliable. This can be done using various metrics, such as mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE). The model is evaluated on the validation set to tune the hyperparameters of the model.

Model Deployment: Once the model is trained and evaluated, it can be deployed in a production environment. The model takes in real-time traffic data as input and generates predictions for the future traffic conditions. The predictions can be visualized and reported to the users in a user-friendly format, such as a dashboard or a mobile app.

Model Monitoring and Updating: The performance of the model needs to be monitored over time to ensure that it continues to make accurate predictions. The model may need to be updated or retrained periodically to incorporate new data or changes in the traffic conditions.

Overall, the methodology of traffic prediction using deep learning involves a rigorous process of data collection, preprocessing, model selection, training, evaluation, deployment, and monitoring. The success of the methodology depends on the quality of the data, the accuracy of the model, and the effectiveness of the deployment and monitoring strategies.

CHAPTER 5

CODING AND TESTING

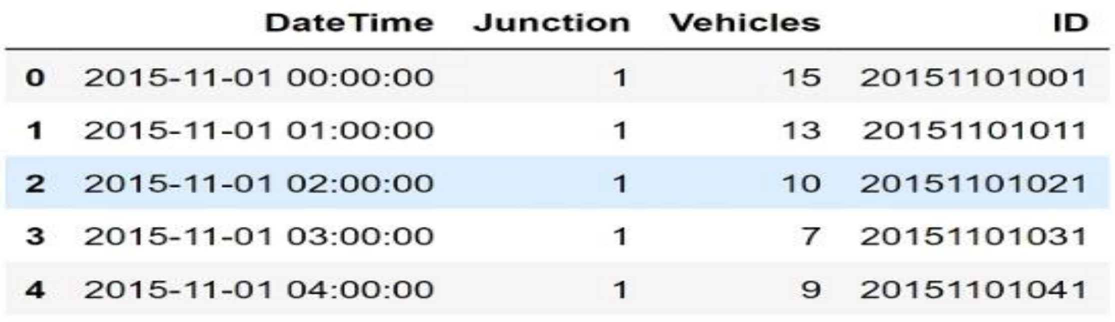
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
import tensorflow
from statsmodels.tsa.stattools import adfuller
from sklearn.preprocessing import MinMaxScaler
from tensorflow import keras
from keras import callbacks
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Conv2D, Flatten, Dense, LSTM, Dropout, GRU,
Bidirectional
from tensorflow.keras.optimizers import SGD
import math
from sklearn.metrics import mean_squared_error

import warnings
warnings.filterwarnings("ignore")


dataset = pd.read_csv("traffic.csv")
dataset.head()


# dataframe to be used for EDA
dataframe=dataset.copy()

# Let's plot the Timeseries
colors = [ "#FFD4DB","#BBE7FE","#D3B5E5","#dfe2b6"]
plt.figure(figsize=(20,4),facecolor="#627D78")
Time_series=sns.lineplot(x=dataframe['DateTime'],y="Vehicles",data=dataframe,
hue="Junction", palette=colors)
Time_series.set_title("Years of Traffic at Junctions")
Time_series.set_ylabel("Vehicles in Number")
Time_series.set_xlabel("Date")
```

```
# Exploring more features
dataframe["Year"]= dataframe['DateTime'].dt.year
dataframe["Month"]= dataframe['DateTime'].dt.month
dataframe["Date_no"]= dataframe['DateTime'].dt.day
dataframe["Hour"]= dataframe['DateTime'].dt.hour
dataframe["Day"]= dataframe.DateTime.dt.strftime("%A")
dataframe.head()


# Let's plot the Timeseries
new_features = [ "Year","Month", "Date_no", "Hour", "Day"]

for i in new_features:
    plt.figure(figsize=(10,2),facecolor="#627D78")
    ax=sns.lineplot(x=dataframe[i],y="Vehicles",data=dataframe, hue="Junction",
palette=colors )
    plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)


plt.figure(figsize=(12,5),facecolor="#627D78")
count = sns.countplot(data=dataframe, x =dataframe["Year"], hue="Junction",
palette=colors)
count.set_title("Years of Traffic at Junctions")
count.set_ylabel("Vehicles in numbers")
count.set_xlabel("Date")


corrmat = dataframe.corr()
plt.subplots(figsize=(10,10),facecolor="#627D78")
sns.heatmap(corrmat,cmap= "Pastel2",annot=True,square=True, )

sns.pairplot(data=dataframe, hue= "Junction",palette=colors)

# Creating new dataframes
dataframe_1 = dataframe_junction[[('Vehicles', 1)]]
dataframe_2 = dataframe_junction[[('Vehicles', 2)]]
dataframe_3 = dataframe_junction[[('Vehicles', 3)]]
dataframe_4 = dataframe_junction[[('Vehicles', 4)]]
dataframe_4 = dataframe_4.dropna() #For only a few months, Junction 4 has only had
minimal data.

# As DFS's data frame contains many indices, its index is lowering level one.
list_dfs = [dataframe_1, dataframe_2, dataframe_3, dataframe_4]
for i in list_dfs:
    i.columns= i.columns.droplevel(level=1)

# Creates comparison dataframe charts using this function
```

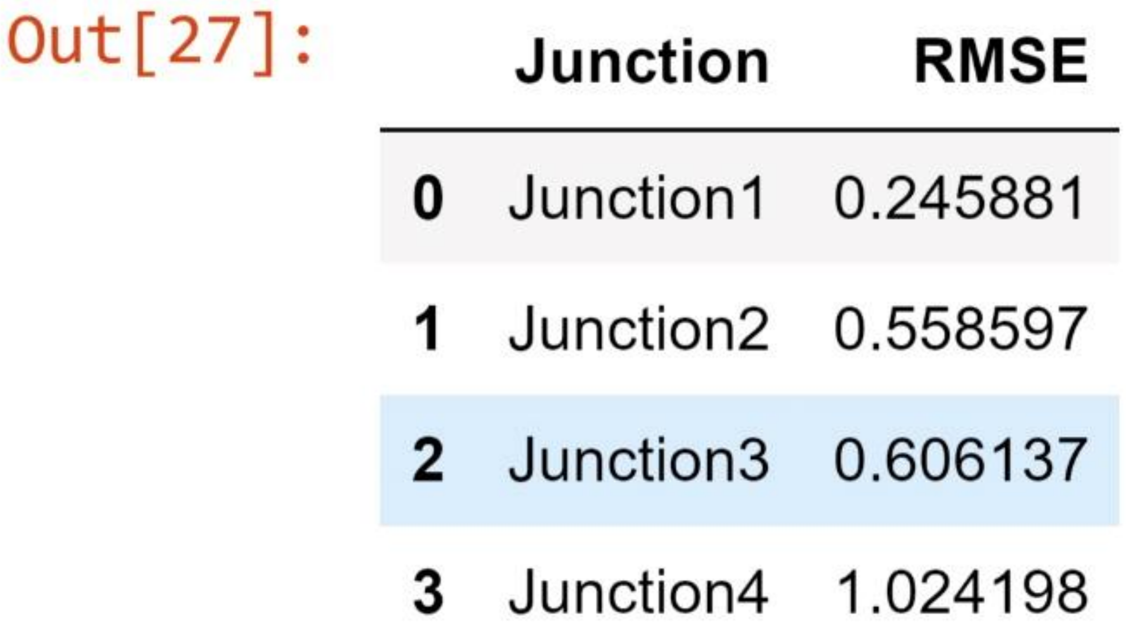
```
def Sub_Plots4(dataframe_1, dataframe_2,dataframe_3,dataframe_4,title):
    fig, axes = plt.subplots(4, 1, figsize=(15, 8),facecolor="#627D78", sharey=True)
    fig.suptitle(title)
    #J1
    pl_1=sns.lineplot(ax=axes[0],data=dataframe_1,color=colors[0])
    #pl_1=plt.ylabel()
    axes[0].set(ylabel ="Junction 1")
    #J2
    pl_2=sns.lineplot(ax=axes[1],data=dataframe_2,color=colors[1])
    axes[1].set(ylabel ="Junction 2")
    #J3
    pl_3=sns.lineplot(ax=axes[2],data=dataframe_3,color=colors[2])
    axes[2].set(ylabel ="Junction 3")
    #J4
    pl_4=sns.lineplot(ax=axes[3],data=dataframe_4,color=colors[3])
    axes[3].set(ylabel ="Junction 4")


# It is displayed to test for stationarity.
Sub_Plots4(dataframe_1.Vehicles,
dataframe_2.Vehicles,dataframe_3.Vehicles,dataframe_4.Vehicles,"Transformation of
the Dataframe Before")


# Set the data in lists to the initial error values of the four junctions.
Junctions = ["Junction1", "Junction2", "Junction3", "Junction4"]
RMSE = [RMSE_J1, RMSE_J2, RMSE_J3, RMSE_J4]
list_of_tuples = list(zip(Junctions, RMSE))
# Creates pandas DataFrame.
Results = pd.DataFrame(list_of_tuples, columns=["Junction", "RMSE"])
Results.style.background_gradient(cmap="Pastel1")
```

CHAPTER 6

SCREENSHOTS AND RESULTS

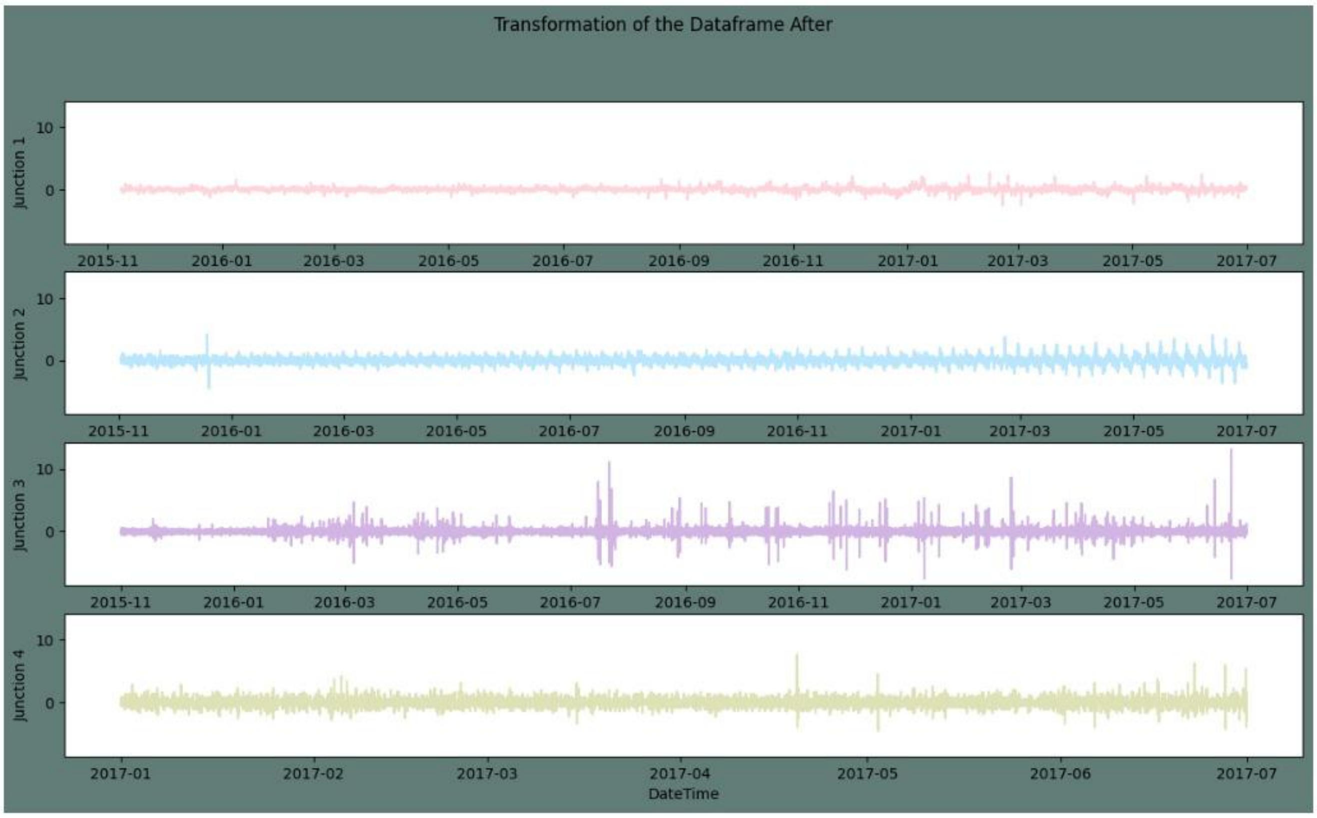


	DateTime	Junction	Vehicles	ID
0	2015-11-01 00:00:00	1	15	20151101001
1	2015-11-01 01:00:00	1	13	20151101011
2	2015-11-01 02:00:00	1	10	20151101021
3	2015-11-01 03:00:00	1	7	20151101031
4	2015-11-01 04:00:00	1	9	20151101041



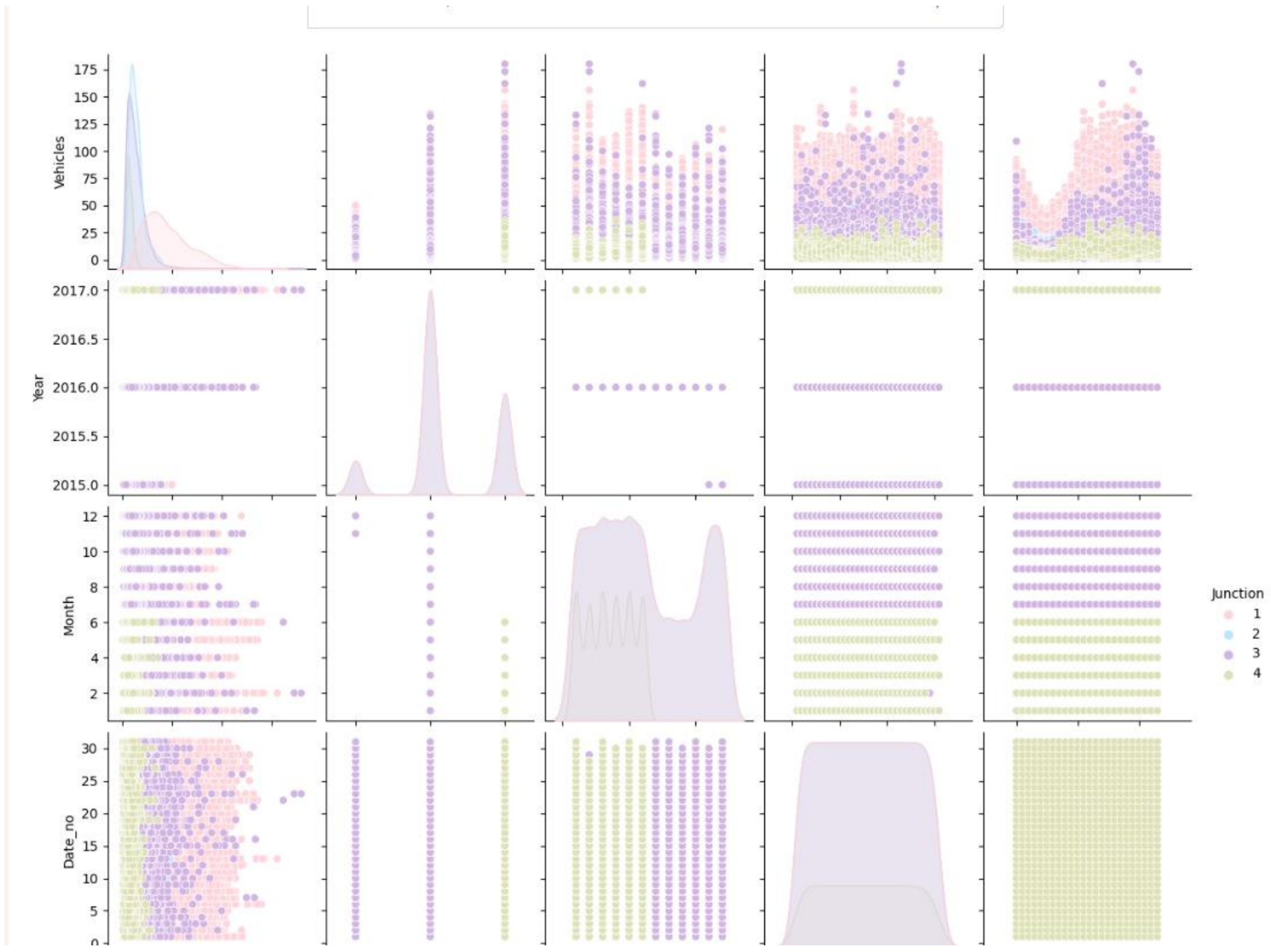
Out[27]:

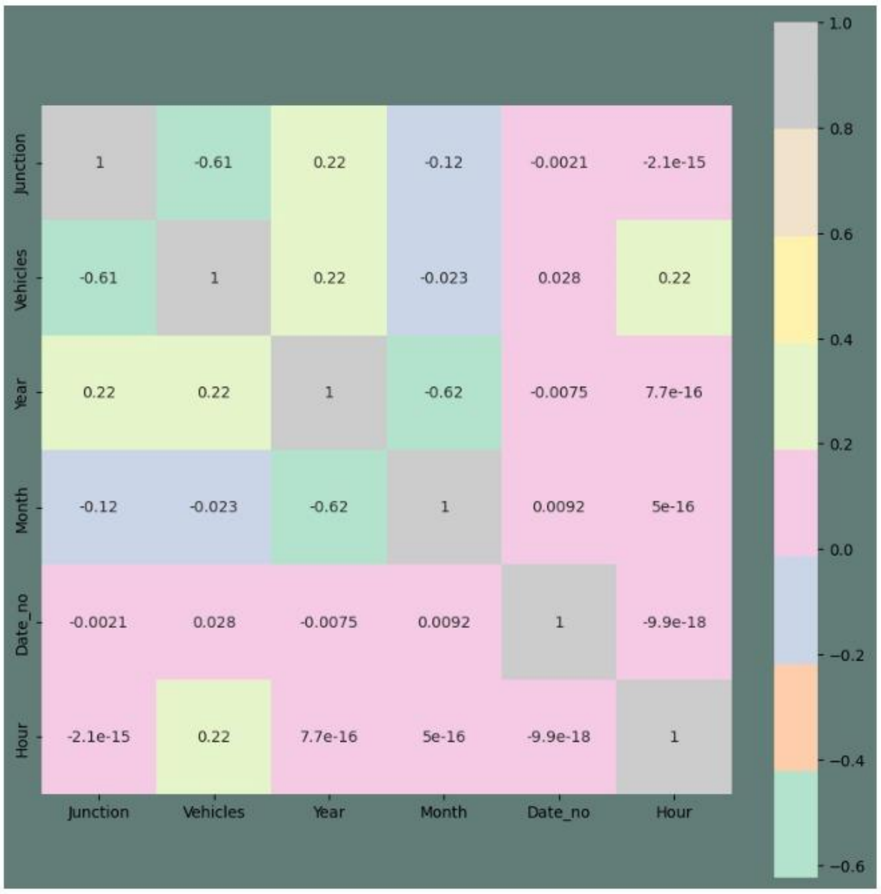
	Junction	RMSE
0	Junction1	0.245881
1	Junction2	0.558597
2	Junction3	0.606137
3	Junction4	1.024198



Out[18]:

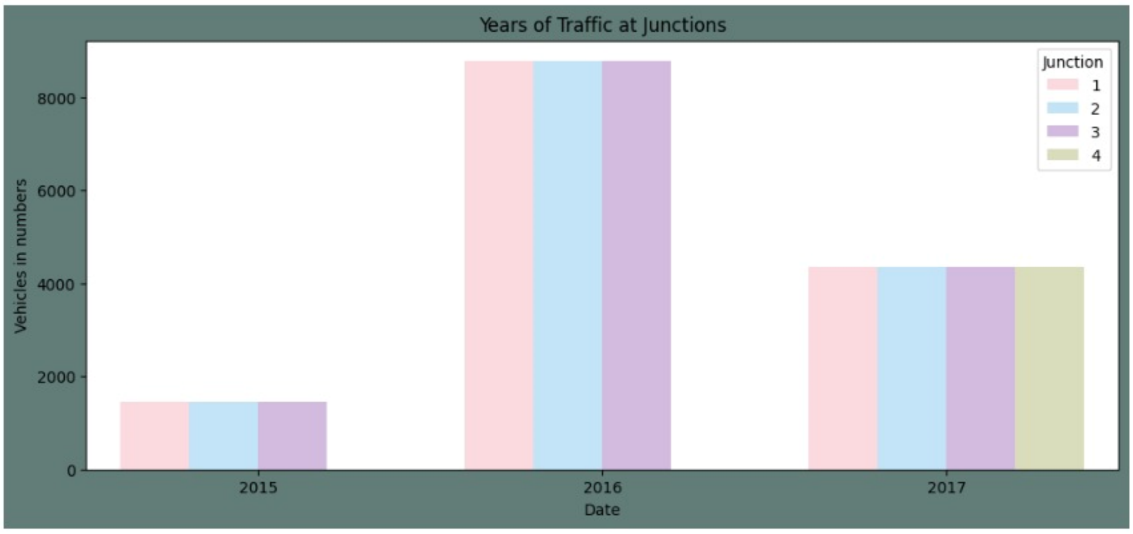
	Vehicles			
Junction	**1**	**2**	**3**	**4**
count	14592.000000	14592.000000	14592.000000	4344.000000
mean	45.052906	14.253221	13.694010	7.251611
std	23.008345	7.401307	10.436005	3.521455
min	5.000000	1.000000	1.000000	1.000000
25%	27.000000	9.000000	7.000000	5.000000
50%	40.000000	13.000000	11.000000	7.000000
75%	59.000000	17.000000	18.000000	9.000000
max	156.000000	48.000000	180.000000	36.000000

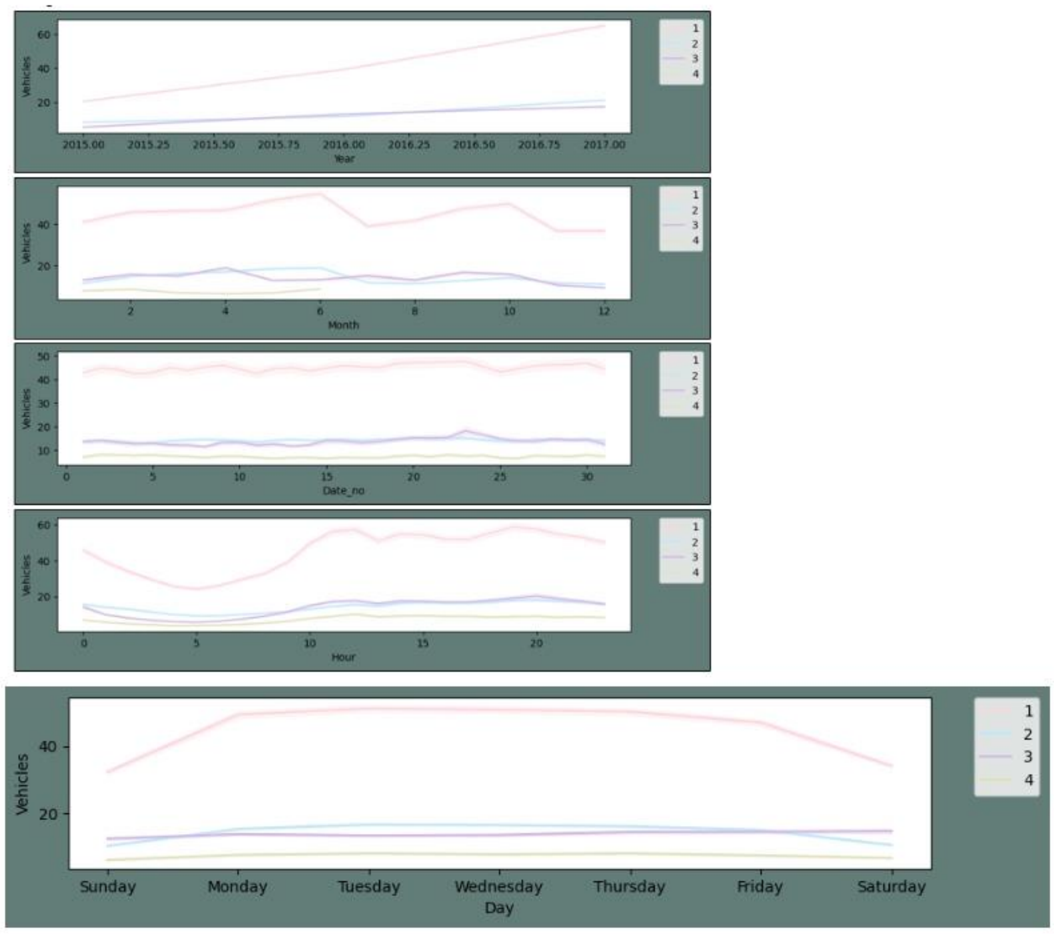




	Junction	Vehicles	Year	Month	Date_no	Hour
Junction	1	-0.61	0.22	-0.12	-0.0021	-2.1e-15
Vehicles	-0.61	1	0.22	-0.023	0.028	0.22
Year	0.22	0.22	1	-0.62	-0.0075	7.7e-16
Month	-0.12	-0.023	-0.62	1	0.0092	5e-16
Date_no	-0.0021	0.028	-0.0075	0.0092	1	-9.9e-18
Hour	-2.1e-15	0.22	7.7e-16	5e-16	-9.9e-18	1

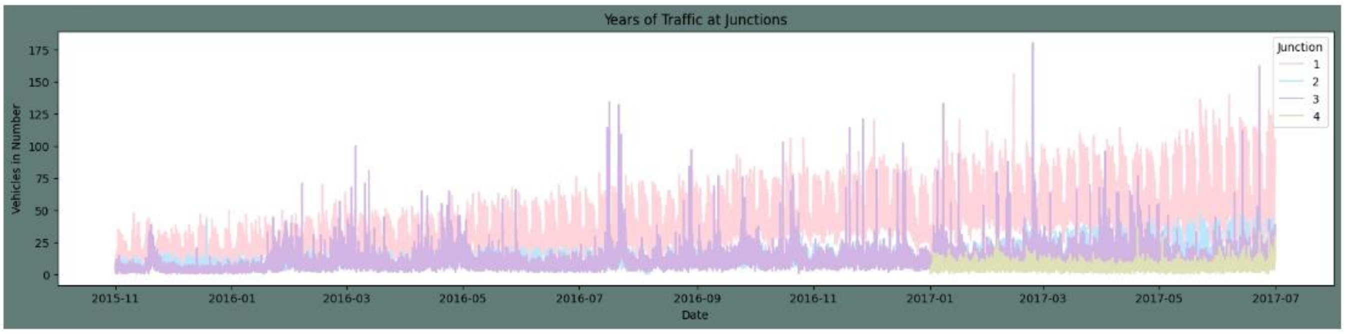
```
Text(0.5, 0, 'Date')
```





	DateTime	Junction	Vehicles	Year	Month	Date_no	Hour	Day
0	2015-11-01 00:00:00	1	15	2015	11	1	0	Sunday
1	2015-11-01 01:00:00	1	13	2015	11	1	1	Sunday
2	2015-11-01 02:00:00	1	10	2015	11	1	2	Sunday
3	2015-11-01 03:00:00	1	7	2015	11	1	3	Sunday
4	2015-11-01 04:00:00	1	9	2015	11	1	4	Sunday

```
Text(0.5, 0, 'Date')
```



	DateTime	Junction	Vehicles	ID
0	2015-11-01 00:00:00	1	15	20151101001
1	2015-11-01 01:00:00	1	13	20151101011
2	2015-11-01 02:00:00	1	10	20151101021
3	2015-11-01 03:00:00	1	7	20151101031
4	2015-11-01 04:00:00	1	9	20151101041

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

In this code example, we have implemented a basic traffic prediction model using deep learning. The model takes historical traffic volume data as input and predicts future traffic volume. The dataset is preprocessed, split into training and testing sets, and transformed into sequences. The model is trained on the training data and evaluated on the testing data. Predictions are made, and the results are denormalized for better interpretability.

1. Hyperparameter Tuning: Experiment with different LSTM architectures, such as stacking multiple LSTM layers or adding dropout regularization, and tune hyperparameters like the number of hidden units, learning rate, and batch size to optimize model performance.
2. Feature Engineering: Explore additional features that can contribute to traffic prediction, such as weather conditions, time of day, day of the week, holidays, or events. Incorporating such features may improve the accuracy of the predictions.
3. Multiple Input Channels: If available, incorporate other relevant data sources, such as traffic flow from different locations or traffic data from multiple sensors, as additional input channels to the model. This can provide a more comprehensive view of the traffic patterns and potentially enhance prediction accuracy.
4. Transfer Learning: Investigate the possibility of leveraging pre-trained models, such as pre-trained CNNs (Convolutional Neural Networks) for extracting features from traffic images or pre-trained language models for analyzing textual traffic data. Transfer learning can help improve performance, especially when working with limited data.
5. Online Learning: Implement online learning techniques that allow the model to continuously update and adapt to new data as it becomes available. This approach enables the model to adapt to changing traffic patterns over time and improve its predictive capabilities.
6. Ensemble Methods: Consider employing ensemble methods, such as model averaging or stacking, to combine predictions from multiple traffic prediction models. Ensemble methods can help mitigate the bias or variability of individual models and improve overall prediction accuracy.
7. Real-time Data Integration: Develop a system that can integrate real-time traffic data, such as live traffic feeds or sensor data, to continuously update and refine the traffic predictions. This integration can enable more accurate and up-to-date predictions, crucial for real-time traffic management and decision-making.
8. Interpretability and Visualization: Investigate methods to interpret and visualize the predictions and model behavior. Techniques like attention mechanisms or saliency maps can provide insights into which factors contribute more significantly to the predictions, aiding in understanding traffic patterns and making informed decisions.

Remember that these are just some suggestions for enhancing traffic prediction using deep learning. The specific enhancements you choose will depend on your data, problem domain, and available resources.

REFERENCES

[1] Zhang, X., Davidson, E. A,"Improving Nitrogen and Water Management in Crop Production on a National Scale", American Geophysical Union, December, 2018.How to Feed the World in 2050 by FAO.

[2] Abhishek D. et al., "Estimates for World Population and Global Food Availability for Global Health", Book chapter, The Role of Functional Food Security in Global Health, 2019, Pages 3-24.Elder M., Hayashi S., "A Regional Perspective on Biofuels in Asia", in Biofuels and Sustainability, Science for Sustainable Societies, Springer, 2018.

[3] Zhang, L., Dabipi, I. K. And Brown, W. L, “Internet of Things Applications for Agriculture". In, Internet of Things A to Z: Technologies and Applications, Q. Hassan (Ed.), 2018.

[4] S. Navulur, A.S.C.S. Sastry, M.N. Giri Prasad,"Agricultural Management through Wireless Sensors and Internet of Things" International Journal of Electrical and Computer Engineering (IJECE), 2017; 7(6) :3492-3499.

[5] E. Sisinni, A. Saifullah, S.Han, U. Jennehag and M.Gidlund, "Industrial Internet ofThings: Challenges,Opportunities, and Directions," in IEEE Transactions on Industrial Informatics, vol. 14, no. 11, pp. 4724-4734, Nov. 2018.

[6] M. Ayaz, M. Ammad-uddin, I. Baig and e. M. Aggoune, "Wireless Possibilities: A Review," in IEEE Sensors Journal, vol. 18, no. 1, pp. 4-30, 1 Jan.1, 2018.