**TRAFFIC FORECASTING**

**by**

**Capturing the continuity and periodicity of time series through Traffic transformer**

**A PROJECT REPORT**

***Submitted by***

**Nallendiran, A P**

**(2019272026)**

s*ubmitted to the Faculty of*

**DR.M.VIJAYALAKSHMI**

**Associate Professor**

**INFORMATION SCIENCE AND TECHNOLOGY**

*in partial fulfillment for the award of the degree*  *of*

**MASTER OF COMPUTER APPLICATIONS**



**DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY**

**COLLEGE OF ENGINEERING, GUINDY**

**ANNA UNIVERSITY CHENNAI**

**600025**

**MAY 2022**

# BONA FIDE CERTIFICATE

Certified that this project report **TRAFFIC FORECASTING by Capturing the continuity and periodicity of time series through Traffic transformer** is the bona fide work of Nallendiran A P(2019272026) who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

**PLACE: Chennai**

**DATE:**

**DR.M.VIJAYALAKSHMI**

ASSOCIATE PROFESSOR

PROJECT GUIDE

DEPARTMENT OF IST,

CEG, ANNA UNIVERSITY,

CHENNAI - 600025

COUNTER SIGNED

## Dr. S. SRIDHAR

HEAD OF THE DEPARTMENT INFORMATION SCIENCE AND TECHNOLOGY COLLEGE OF ENGINEERING, GUINDY, ANNA UNIVERSITY CHENNAI-600025

**ABSTRACT :**

Traffic forecasting is a challenging problem due to the complexity of jointly modeling spatio-temporal dependencies at different scales. Recently, several hybrid deep learning models have been developed to capture such dependencies. These approaches utilize convolutional neural networks or graph neural networks (GNNs) to model spatial dependency and leverage recurrent neural networks (RNNs) to learn temporal dependency.

However, RNNs are only able to capture sequential information in the time series, while being incapable of modeling their periodicity (e.g., weekly patterns). Moreover, RNNs are difficult to parallelize, making training and prediction less efficient. In this work we propose a novel deep learning architecture called Traffic Transformer to capture the continuity and periodicity of time series and to model spatial dependency.

**TABLE OF CONTENTS**

**ABSTRACT**

**CHAPTER 1: INTRODUCTION PAGE NO**

* 1. INTRODUCTION 1
  2. PROBLEM STATEMENT 1
  3. MOTIVATION AND OBJECTIVE 1
  4. CHALLENGE AND PROPOSAL METHOD 1

1.5 TECHNOLOGIES USED 1

**CHAPTER 2: LITERATURE SURVEY** 2

**CHAPTER 3: SYSTEM DESIGN**

3.1 SYSTEM ARCHITECTURE 3

3.2 MODULE DESCRIPTION

3.2.1 CONTINUITY OF TIME SERIES 4

3.2.2 PERIODICTY OF TIME SERIES 4

3.2.3 TRAFFIC TRANSFORMER ARCHITECTURE 4

3.2.4 DATA SET DESCRIPTION 5

**CHAPTER 4: MODULE IMPLEMENTATION**

4.1 VIDEO PROCESSING 6

4.2 LOADING DATASET 6

4.3 PREPROCESSING & DATA TRANSFORMATION 11

4.4 TRANSFORMED SERIES FOR STATIONARITY 13

4.5 MODEL TRAINING & RESULT 14

4.6 REFERENCES 16

**CHAPTER 1**

**INTRODUCTION:**

Traffic forecasting is concerned with estimating future traffic conditions (such as the density of vehicles and their speed) to enable the prediction of future events (such as congestion or travel duration) by analyzing historical traffic conditions and patterns. Highly accurate forecasts provide guidance to decision-makers, provide safety and convenience for citizens, and reduce environmental Traffic forecasting, however, is challenging due to the complexity of modeling spatio-temporal dependencies of traffic conditions at varying.

For instance, the traffic flow on a road is influenced by both its historical traffic conditions and the conditions of upstream roads. Due to the increasing availability of massive traffic data, high-performance computing, and novel deep learning models, recent work has pushed the envelope on learning spatio-temporal dependency models for accurate traffic forecasting.

**PROBLEM STATEMENT:**

Traffic forecasting is concerned with estimating future traffic conditions (such as the density of vehicles and their speed) to enable the prediction of future events (such as congestion or travel duration) by analyzing historical traffic conditions and patterns. Highly accurate forecasts provide guidance to decision-makers, provide safety and convenience for citizens, and reduce environmental impacts.

**CHALLENGE AND PROPOSAL METHOD:**

**Challenge:** Due to the complexity of jointly modeling spatio-temporal dependencies at different scales.

**Existing Solution:** Convolutional neural networks or graph neural networks (GNNs) to model spatial dependency and leverage recurrent neural networks (RNNs) to learn temporal dependency.

**Disadvantage:** RNNs are only able to capture sequential information in the time series, while being incapable of modeling their periodicity.

**Proposed Method:** Traffic Transformer to capture the continuity and periodicity of time series and to model spatial dependency.

**TECHNOLOGIES USED:**

* **Requirements: TensorFlow, NumPy**
* **LANGUAGE: Python**
* **Models: GCNN, Transformer, FFNN**

1

**CHAPTER 2**

There are a lot of approaches which are adopted for this prototype. Among those the following have been adopted and researched.

**Cui et al., 2018; Jin et al., 2018[1]** has proposed a models based on recurrent neural networks (RNNs), such as Gated Recurrent Unit (GRU) and Long Short Term

Memory (LSTM), can be used effectively to capture temporal dependencies.

**Shi et al. (2015)[2]** proposed a convolutional LSTM model for traffic forecasting, in which the traffic flow at each time-step was recursively fed into an LSTM architecture.

**Li, Yu, Shahabi, and Liu (2017)[3]** proposed a convolutional RNN model where graph diffusion convolutional operators were used to model spatial dependencies and GRU was employed instead of LSTM to capture temporal dependencies.

**Vaswani et al., 2017[4]** have introduced the Transformer architecture to replace RNNs for machine translation. It replaces convolutional neural networks (CNNs) and RNNs and is solely built on attention mechanisms to model sequential data.

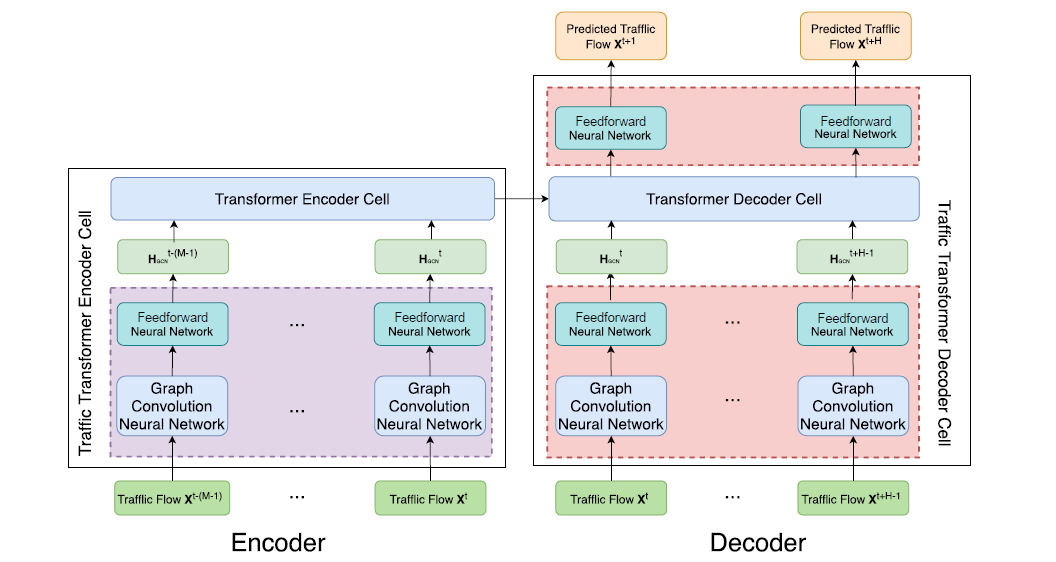
**Lv et al. (2014)[5]** proposed a novel deep learning architecture to inherently consider temporal and spatial dependencies, where autoencoders were first introduced to serve as the building block to learn latent features.

**Zhang, Zheng, Qi, Li, and Yi (2016)[6]** designed a deep learning model which takes into account both temporal aspects (temporal closeness, periods, and trends of crowd traffic) and spatial proximity.

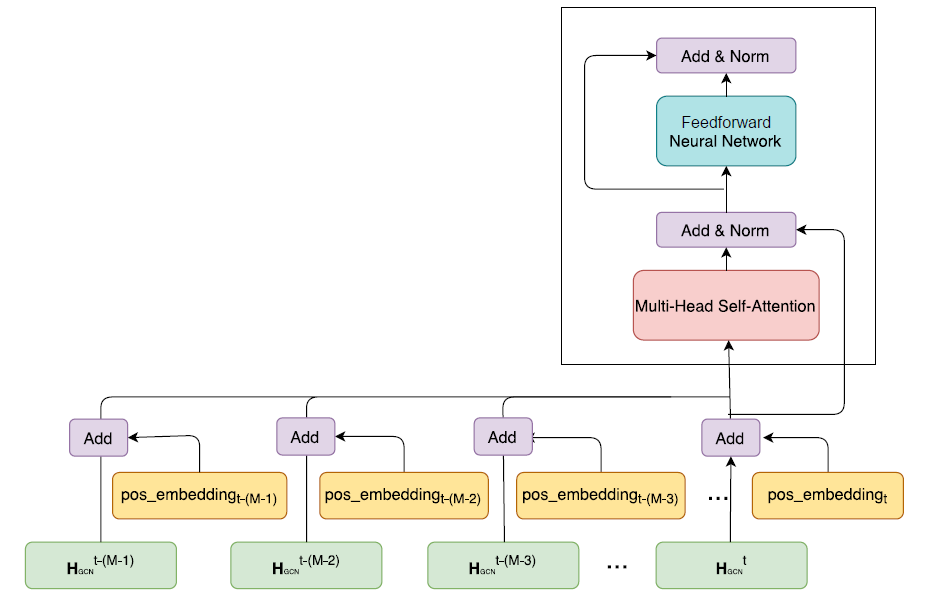
2

**CHAPTER 3**

**OVERALL ARCHITECTURE:**



**ENCODER CELL ARCHITECTURE:**



3

**MODULES**

**Continuity of time series:**

The aim of this strategy is to encode relative continuity, which means the continuity of time in the window of the source–target sequence regardless of the *position* of one time-step in the whole time series under consideration.

**Relative position encoding:**

The aim is to encode relative continuity, which means that we care about the continuity of time in the window of the source–target sequence regardless of the position of one time-step in the whole time series under consideration.

This can be achieved by indexing the time-step (t − (M − 1)) with 0 as the starting position and raising the index position by 1 per time-step.

**Global position encoding:**

The success of relative position encoding in preserving the local continuity of time. It ignores the fact that most time‐steps in two consecutive source–target sequence pairs are common.

**Periodicity of time series:**

Time series also convey periodicity (i.e., weekly and daily patterns). There are two potential ways to go about this. One is centered around the position encoding design: the position embedding for each time-step is enriched with periodic patterns. The other is by using different time series segments corresponding to different temporal features.

**Periodic position encoding:**

The relative/global position encoding strategy to position indexes in terms of weeks and days

**Time series segments**

Periodic position encoding employs different flags (e.g., flags to differentiate different time-steps on one day and flags to denote days with distinct week attributes) to explicitly differentiate sequences.

**Traffic transformer architecture:**

Since Transformer itself can only deal with sequential information, it is unable to deal with complicated spatiotemporal dependencies. In order to address this problem, here we extend Transformer to Traffic Transformer by introducing a GNN. Our proposed architecture conforms to an end-to-end sequence framework, composed of an encoder & 4

decoder. In the encoder, a sequence of traffic t−(M−1) is first passed through a GCN to aggregate the neighborhood information on a road network, which captures the spatial dependency over nearby nodes.

Then a fully connected neural network is employed to strengthen the expressiveness of the model. These features then are fed into the Transformer encoder cell to learn temporal features. A typical Transformer encoder cell is depicted. Note that this figure adopts elementwise addition-based combination. That is, each Ht GCN is added up with its temporal embedding so as to incorporate temporal information. We can also use similarity-based combination to account for temporal information.Then the results are fed through multi-head self-attention to aggregate the impact of the traffic at other timesteps

on that at time-step t. Again, fully connected neural networks are employed to improve the expressiveness of the model.

As for the decoder, it has a similar structure except that the Transformer decoder cell has one more encoder–decoder attention block and another dense layer is leveraged before outputting the forecasts. During the training process, a sequence of traffic t+H−1

t is fed into the decoder through the same series of neural networks as the input for the encoder and then is passed through the additional dense layer to map the outputs of the

Transformer decoder cell to the predicted traffic t+H t+1 . Thus, the architecture is more computationally efficient during the training process.

However, in the prediction phase, this decoder functions a little differently as any other model for time series prediction does, because in practice the input sequence for the decoder is unknown. In order to yield the predicted traffic at future time-steps, we use X t as the first input for the decoder to output the predicted traffic X t+1, which then serves as the next input for the decoder to gain X t+2. This whole process continues until the predicted horizon is met. Note that here X t rather than a zero matrix, which is widely adopted by other models, is used as the first input for the decoder, since it is supposed to provide more useful information than zeros.

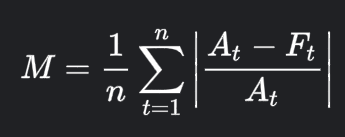
**Data set description**

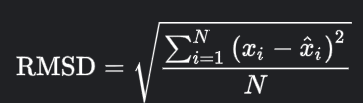
Two real-world benchmark data sets are used for evaluation: METR-LA from loop detectors in the highway of Los Angeles County and California Transportation Agencies Performance Measurement System (PeMS), respectively. Both data sets are sorted by time in ascending order (from the past to the present) and are split into three parts for training (70%), validation (10%), and testing (20%). Z-score normalization with the mean and standard derivation of the training data is applied to these three sets. To retain comparability with baselines, we use the sensor graphs of both data sets.

5

**Evaluation metrics:**

To evaluate and compare the performance of different models, we adopt the following three metrics: the MAE , the root mean squared error (RMSE) and the mean absolute percentage error (MAPE).

****

****

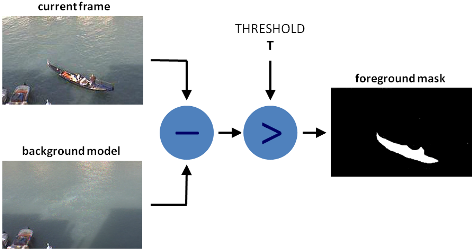
**Module Implementation:**

**Video Preprocessing:**

Extracting key features from traffic footage using computer vision and OpenCV.

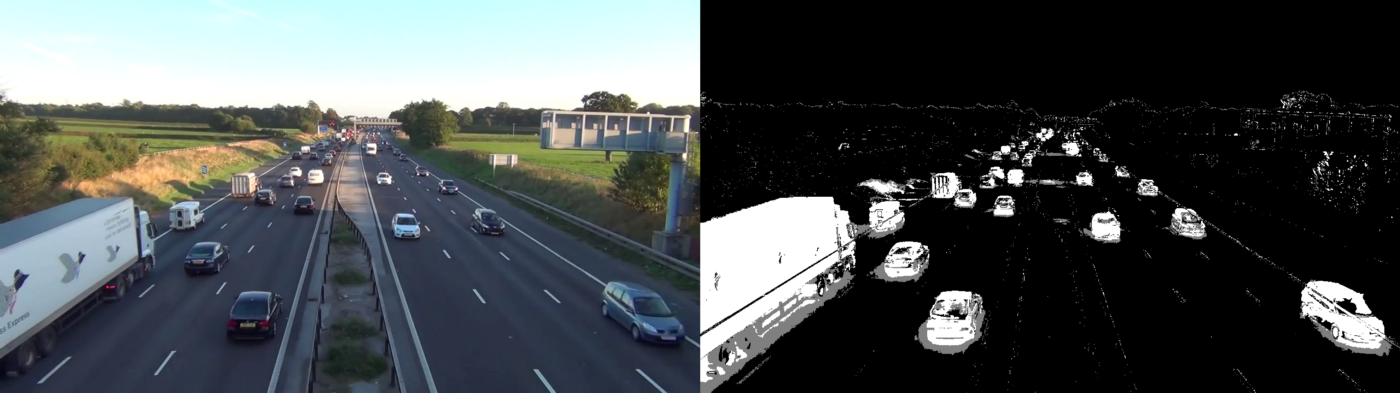
Background subtraction algorithms used for foreground detection. OpenCV image filters. Object detection by contours. Building processing pipeline for further data manipulation.

**Background subtraction:**

****

6

I used MOG algorithm for background subtraction and after processing, it looks like this:

****

**Filters:**

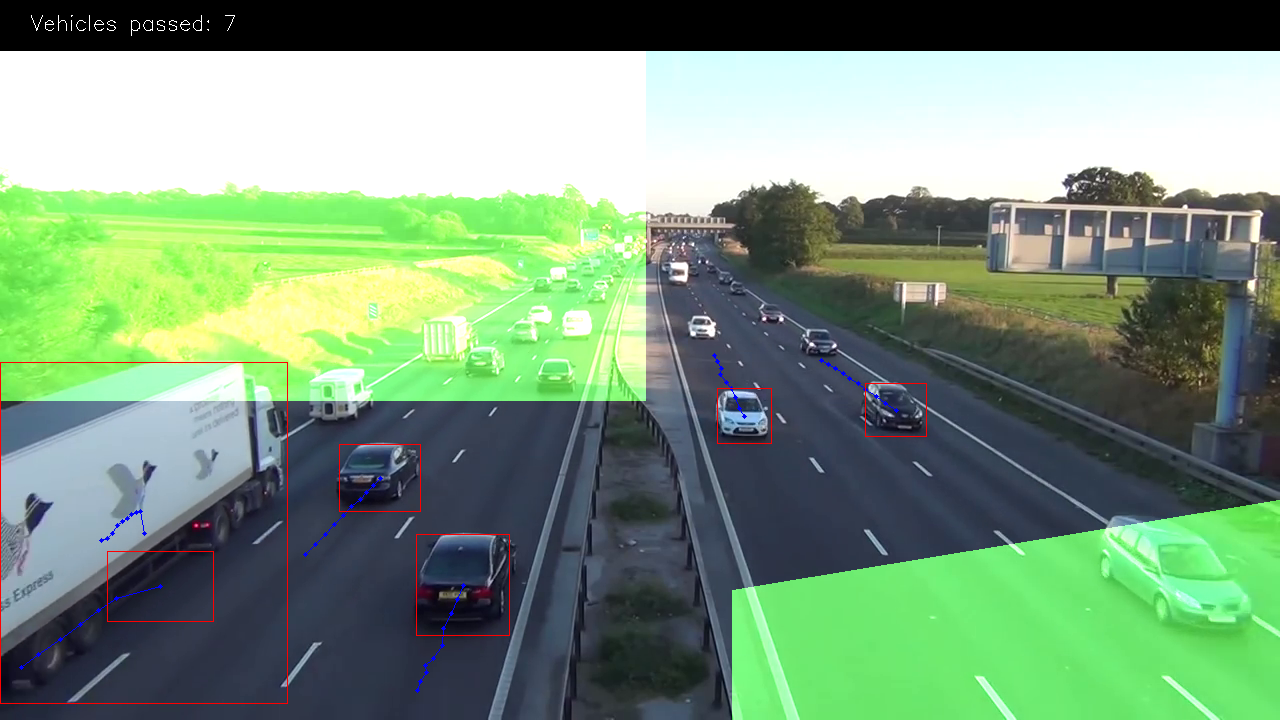
Used filters to remove some noise on foreground mask. First, I used Closing to remove gaps in areas, then Opening to remove 1–2 px points, and after that dilation to make object bolder.

****

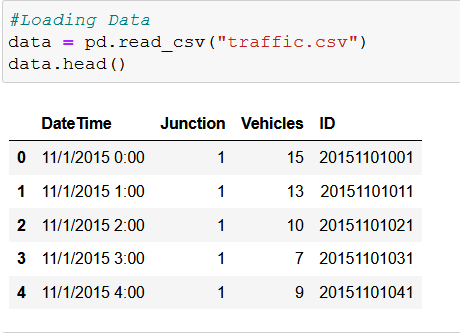
7

**Object Detection & Building processing pipeline:**

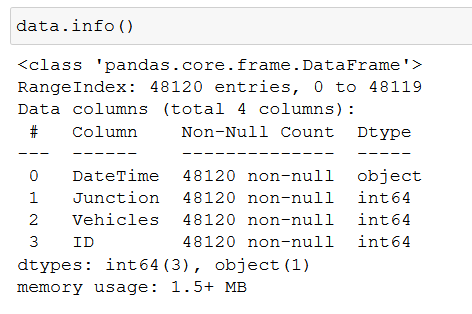
Used the standard cv2.findContours method with params

****

**Loading Data:**

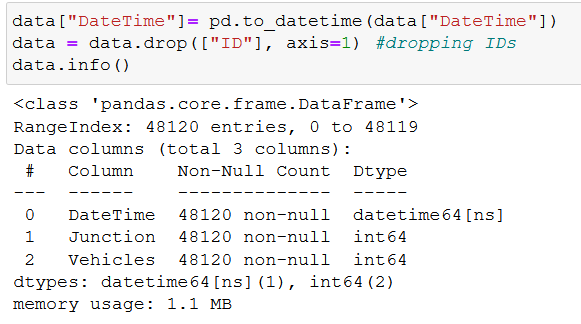
****

8

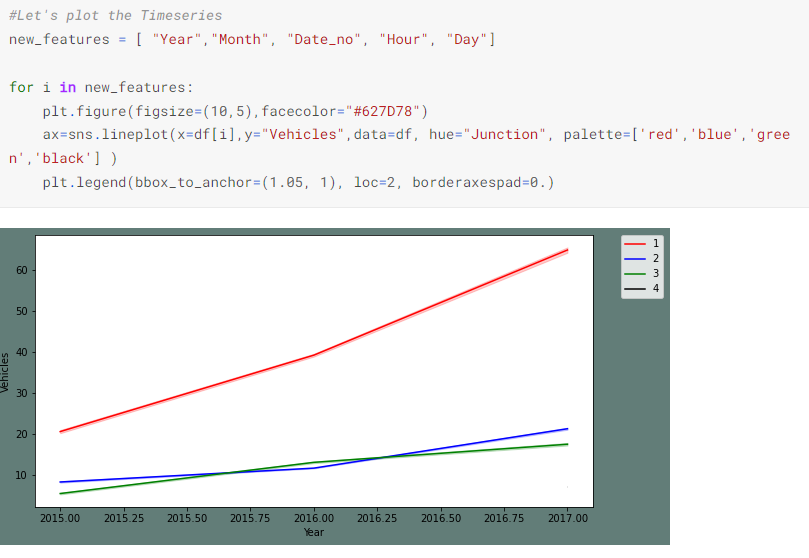
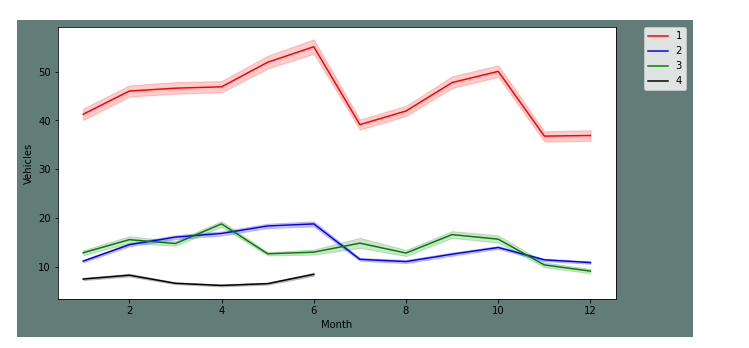
****

**Data Exploration:**

Pharsing dates, Ploting timeseris, Feature engineering for EDA

****

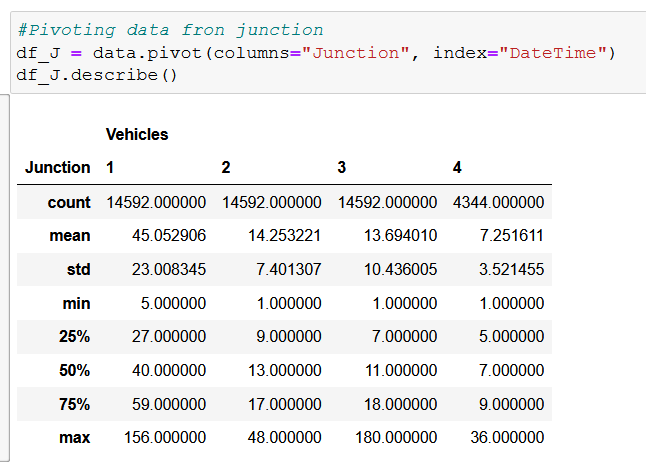
9

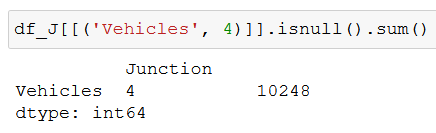
****

10

**Data Transformation and Preprocessing:**

Creating different frames for each Junction and plotting them. Transforming the series and plotting them. To check the seasonality of transformed series. Creating test and train sets.

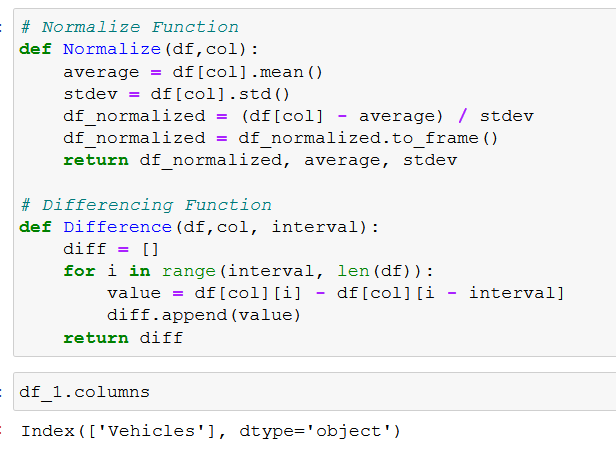


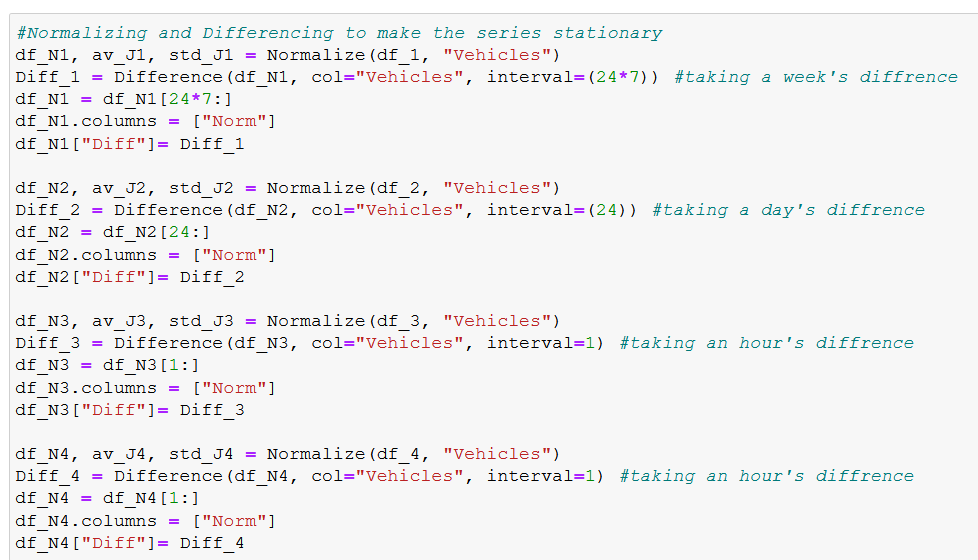


**Transforming:**

A time series is stationary if it does not have a trend or seasonality. However, in the EDA, I observed a weekly seasonality and an upwards trend over the years. In the above plot, it is again established that Junctions one and two have an upward trend. If I limit the span I will be able to further see the weekly seasonality. I will be spairing that step at this point and moving on with the respective transforms on datasets.

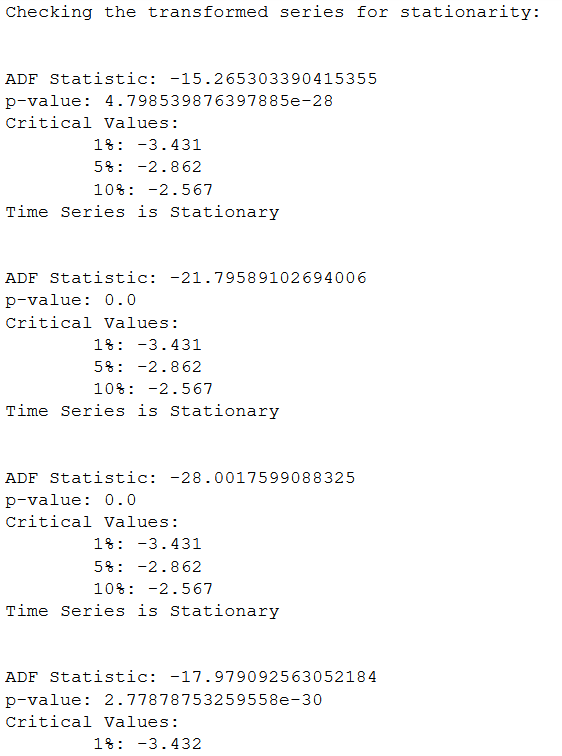
11





12

**Transformed series for stationarity:**

****

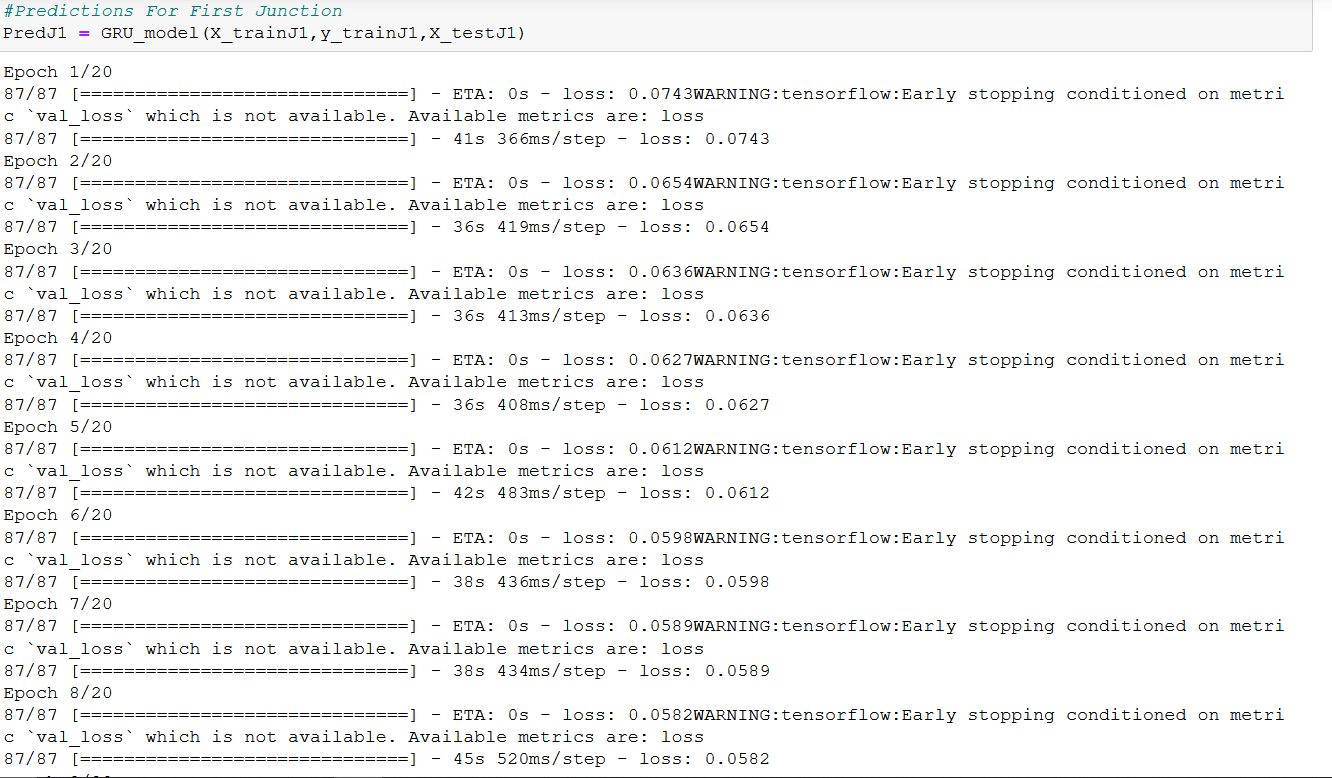
13

**Model Training:**

I have settled to use Gated Recurrent Unit (GRU). In this section, I am creating a function for the neural net to call on and fit the data frames for all four junctions.

****

Now, I will be fitting the transformed training sets of four junctions to the model created and compare them to the transformed test sets.



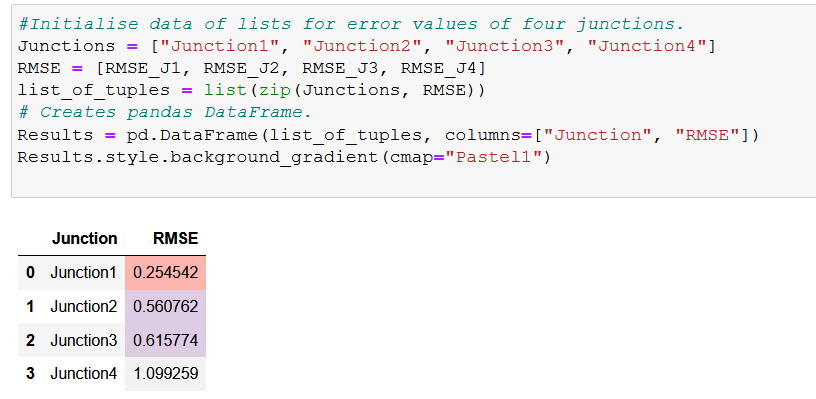
14

**Inversing the Transformation Of Data:**

I will be inversing transforms that I applied to the datasets to remove the seasonality and trends. Performing this step will make the predictions get back on the accurate scale.



**Results for Model:**



15

**References:**

Li, Y., & Moura, J. M. (2019). Forecaster: A graph transformer for forecasting spatial and time dependent data.

Liu, W., Zheng, Y., Chawla, S., Yuan, J., & Xing, X. (2011). Discovering spatio-temporal causal interactions in traffic data

streams. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining

<https://arxiv.org/abs/2001.02908>

<https://ecai2020.eu/papers/274_paper.pdf>

<https://towardsdatascience.com/multivariate-time-series-forecasting-with-transformers-384dc6ce989b>

<https://ieeexplore.ieee.org/document/9491035>

16