Name: Hithesh Shanmugam

Subject code: CSC 578

Final Project

Kaggle username: Hithesh Shanmugam

Rank: 11 (public score: 250.44468)

Journey of the Final project:

The final project was tougher than the other assignments from this course. Starting from the dataset, first there was numerical variables-5: ("temp", "rain_1h", "snow_1h", "clouds_all", and "traffic_volume"), Categorical variables-3: ("holiday", "weather_main", and "weather_description") and Ordinal variable-1: ("date_time"). So, I started splitting the dataset, dropping the categorical variables and normalizing it with the one method in tensor flow. After constructing some models, I tried to predict the output. For denormalize first I used the standard and mean technique. The predictions were not good. Then after researching I realized that I was doing the pre-process wrong and started to analyse the data from the beginning. There was a lot of things I came up with. Before splitting the dataset, I had to convert the ordinal variable in this dataset to a correct form and I kept only the "hour" and dropped everything from the "date_time" variable next the categorical variables I dropped before are seemed to be important in the data so I got dummies for two categorical variables "holiday" and "weather_main". Specifically, in" holiday" before getting dummies I replaced the holidays as holiday and none as workday because to give more importance than the none values. From tutorial the sin and cos of both day and year was also added. Now I dropped only two variables and they are: "weather_description" and "date_time". The normalization was not good before and I ended up using minmax normalizing technique. But I also noticed there was an outlier in "rain_1h" which was also taken care. For denormalizing this time, I used the min max algorithm which gave me the better results.

Note: Shuffle is not needed in this dataset so I made sure to give False in the shuffle in two functions one is make_dataset and other is compile_and_fit. This was an important part in prediction.

Baseline Model:

Architecture: My baseline model had two LSTM layers two dropout layers and three dense layers. The reason I chose this architecture is that from previous assignment in CNN architecture the dropout helped me from overfitting and more dense layers gave me a better result.

Every model I have constructed was compiled and fitted for 50 epochs. The reason I have gone for 50 epochs is to train the model at a better performing state.

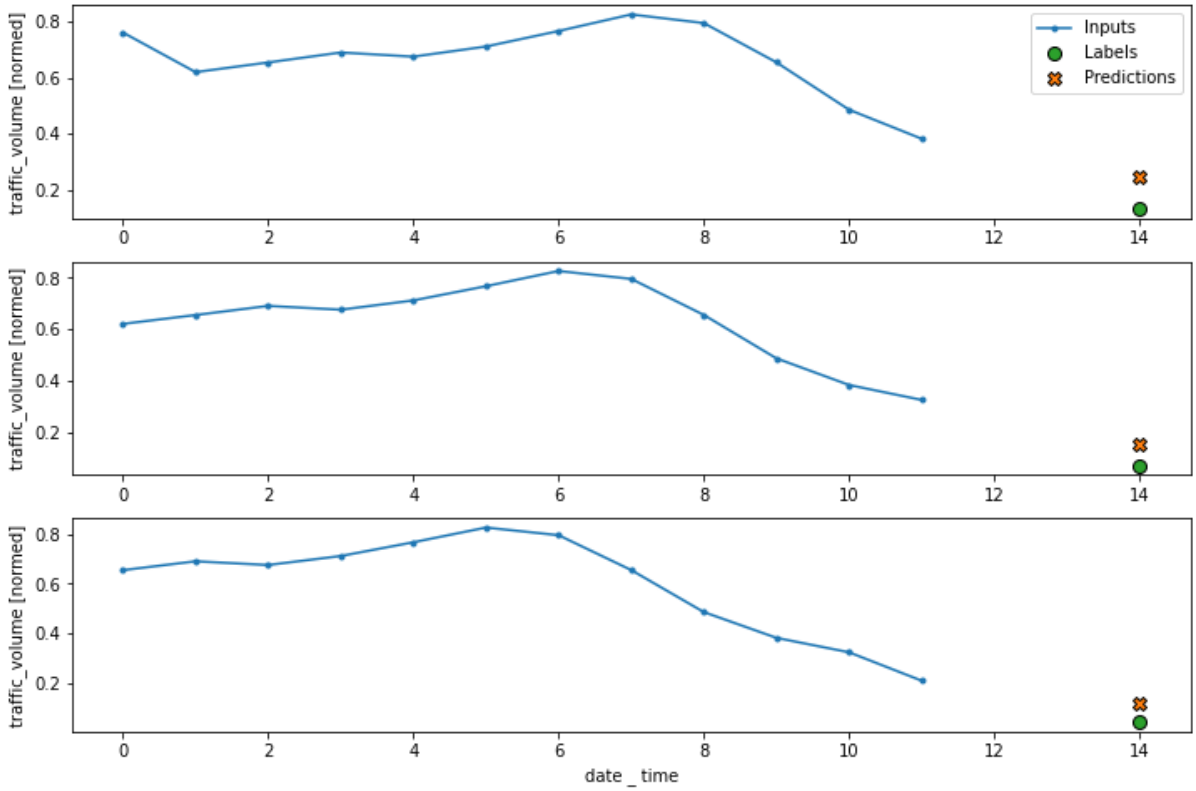
Expectation: Since the model had only few LSTM layers, I was expecting the model to perform not as good as some models with more layers and more hyperparameters.

Results: As my expectation the result was not that good but it was not my worst model. The mean absolute error was better than some other models.

After running for 50 epochs the final losses are as follows

```
Epoch 50/50
955/955 [==============================] - 12s 13ms/step - loss: 0.0147 - mean_absolute_error: 0.0783 - val_loss: 0.0092
- val_mean_absolute_error: 0.0659
156/156 [==============================] - 1s 7ms/step - loss: 0.0092 - mean_absolute_error: 0.0659
```

The window plot can be represented as



The loss plot of this model is shown in the below picture

Worst Model:

Architecture: After the baseline model this was my second model and turned out to be the worst model I have constructed. The architecture of this model is I added two additional LSTM layers with more units.

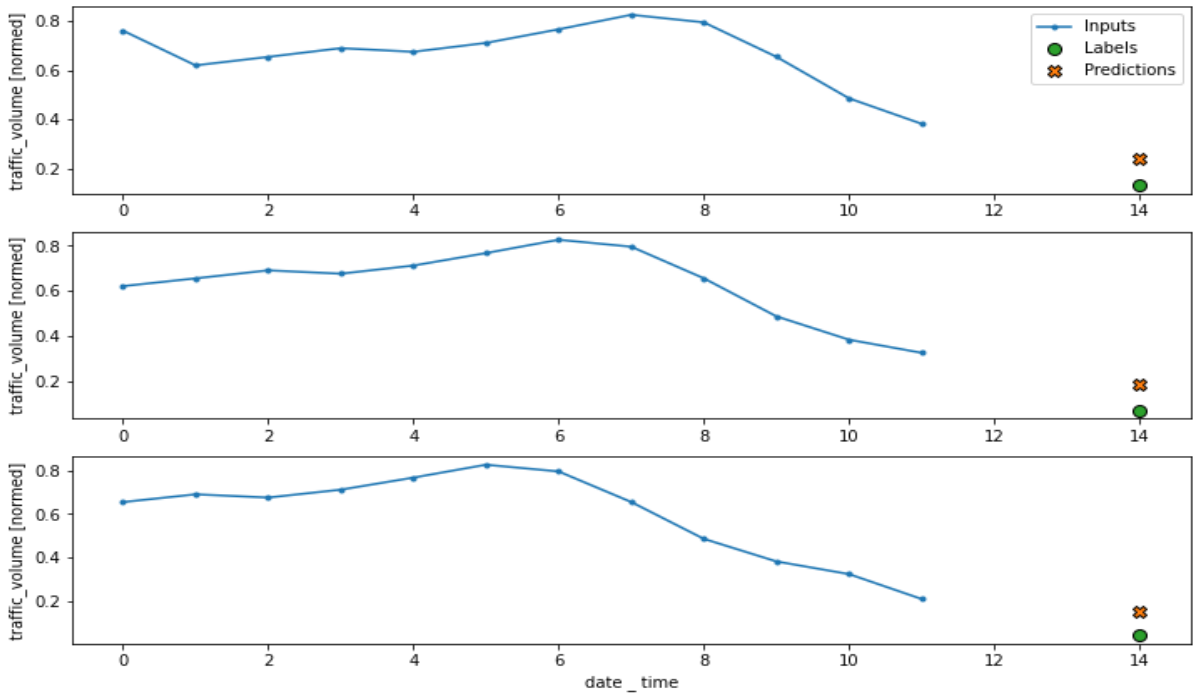
Expectation: I had extra layers in the model so in return I thought that the model would perform better than my baseline model. Even though with the same number of dense layers as previous model I was expecting for a better performance.

Result: My result turned out to be not I was expecting rather it gave me as the worst model that has been constructed in this project.

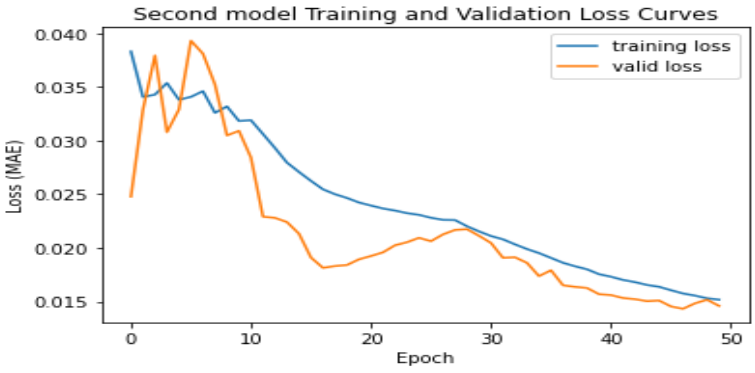
After running for 50 epochs the final losses are as follows

```
Epoch 50/50
955/955 [==============================] - 14s 15ms/step - loss: 0.0152 - mean_absolute_error: 0.0801 - val_loss: 0.0146
- val_mean_absolute_error: 0.0958
156/156 [==============================] - 1s 8ms/step - loss: 0.0146 - mean_absolute_error: 0.0958
```

The window plot can be represented as



The loss plot of this model is shown in the below picture



Best Model: (Kaggle Best Model also)

Architecture: For my best model I had one bidirectional LSTM layer with 512 units and two LSTM layers and three Dense layers. The reason I chose this architecture is that out of my previous models my baseline performed well so I chose that model and added an additional Bidirectional LSTM layer. I took out the dropouts to check if there was any overfitting.

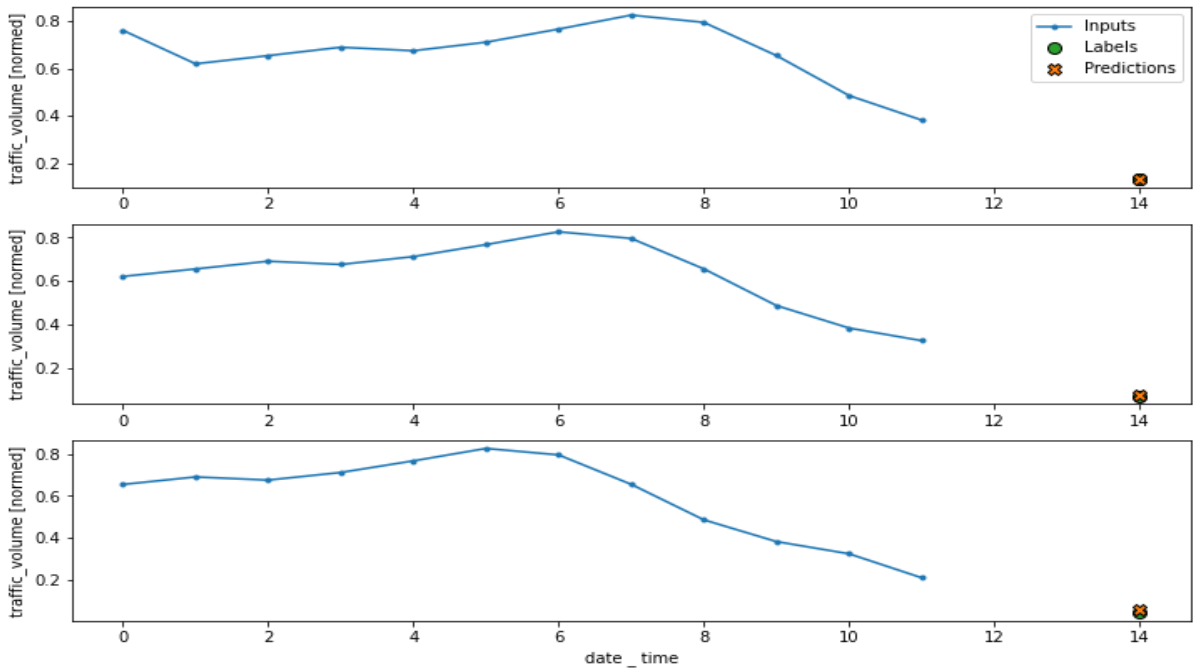
Expectation: Since the bidirectional LSTM has the advantage of enabling it for additional training and also, I was using the same number layers in the previous best model until this point. Hence, I was expecting the model to perform better than my previous models.

Result: As I was expecting the result was better and also the best among the other models I constructed. This was because of the bidirectional LSTM layer.

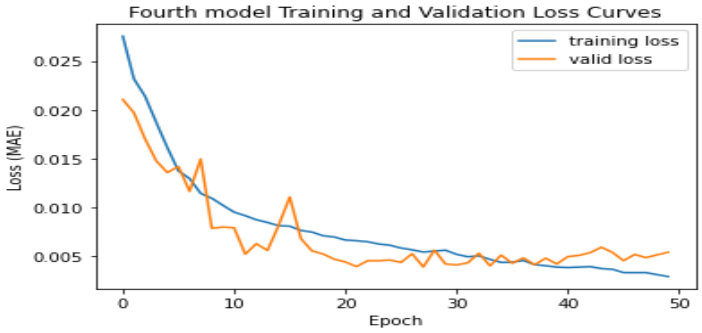
After running for 50 epochs the final losses are as follows

```
Epoch 50/50
955/955 [==============================] - 16s 17ms/step - loss: 0.0029 - mean_absolute_error: 0.0340 - val_loss: 0.0054
- val_mean_absolute_error: 0.0443
156/156 [==============================] - 1s 8ms/step - loss: 0.0054 - mean_absolute_error: 0.0443
```

The window plot can be represented as



The loss plot of this model is shown in the below picture



Second Best Model:

Architecture: Since the bidirectional layer gave me a better result, I tried to put more bidirectional layers. This model had three LSTM layers and three bidirectional LSTM layers and three dropout layers with 0.2 dropout rate. I also used batch normalization in the model and an additional dense layer.

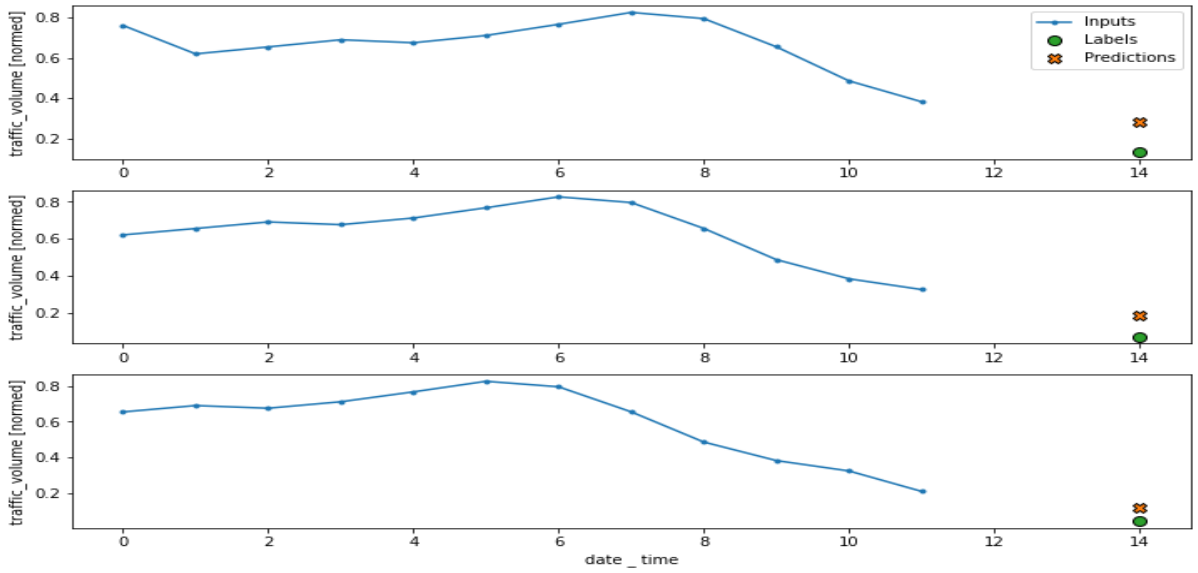
Expectation: With the increased number of layers in LSTM and also bidirectional LSTM layers and also batch normalization layers I was expecting this model to outperform the best model.

Result: The result I got was not better than the best model but it turned out to be my second-best model. The reason I was hoping it to perform better was because of batch normalization layers and also more bidirectional layers but it was not the worst though.

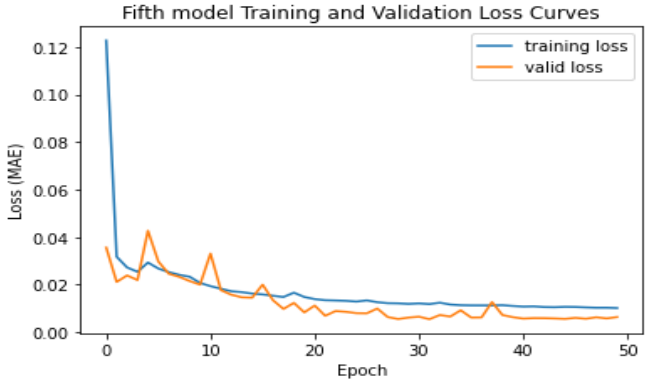
After running for 50 epochs the final losses are as follows

```
Epoch 50/50
955/955 [==============================] - 27s 28ms/step - loss: 0.0101 - mean_absolute_error: 0.0781 - val_loss: 0.0063
- val_mean_absolute_error: 0.0573
156/156 [==============================] - 2s 11ms/step - loss: 0.0063 - mean_absolute_error: 0.0573
```

The window plot can be represented as



The loss plot of this model is shown in the below picture



Note: Every model I have constructed had a reshaping layer at the end of every model. This is to window plot the results.

Discussion:

There was a total of six models I have constructed and every model was running on 50 epochs, Adam optimizer with 0.001 learning rate. The best model did not have more layers than some of the other models. In my point of view, I think this was better than CNN models because CNN needed more layers but here it didn't need to have more layers to perform better. I was expecting that these models would also be like the CNNs which would need more layers but it turned out to be not bad.

Reaction and reflection of result and competition:

The results I got was quite surprising and the competition was fun because it enabled us to perform better than the previous result we have got. For me the competition motivated me to perform better than the models I was creating. Rather than competing with classmates I enjoyed competing with my previous result.

Reaction and reflection of Project and overall course:

The final project was the toughest among the previous works I have done in this course and also among the other works in other courses I have taken, because this was my first advanced class. As a course I learned a lot by performing all the assignments and quiz help me improved my python coding specifically with TensorFlow. It was the toughest course I have taken until this point.

Additional Comments (As advised by professor)

One of my friends and I tried to code a function which would give the MAE close to Kaggle. For that my friend needed my csv so I shared it with him but accidentally while submitting the file in Kaggle he dropped mine. This was not intentional but this was my mistake too but my score was I mentioned as before (250.44468)