Summary - Assignment 3

A total of 14 models are created to address the time-series forecasting problem. Initially, I began with a non-machine learning common sense baseline model, where I got a Mean Absolute Error (MAE) value of 2.62. Then, I created a basic machine-learning model using dense layers which resulted in a slightly higher MAE of 2.63. However, this model had challenges due to the flattening of the time series, which removed the idea of time from the original data. Later, it gave poor results with a convolutional model. It was difficult to experiment with a convolutional model because it applied pooling and treated all data segments equally, which disturbed the information's time order. After realizing the need to preserve time information, I used Recurrent Neural Networks (RNNs), which are developed for time series data.

RNNs have the unique capability of utilizing information from past time steps in current decision-making processes, allowing them to capture intricate dependencies and patterns within sequential data. The internal state of an RNN acts as a memory of past inputs, enabling the modeling of sequences of varying lengths. Simple RNN, while theoretically able to retain information from all previous time periods, faces practical challenges. It suffers from the vanishing gradient problem, making it difficult to train for deep networks. Also, I observed from the graph that the simple RNN is the worst performer among all. To overcome this problem, as a part of Keras, I created LSTM and GRU RNNs.

Simple Gated Recurrent Unit (GRU) is found to be the best-performing model with MAE OF 2.47 compared to all models as it efficiently captures long-range dependencies in sequential data and is computationally less expensive than Long Short-Term Memory (LSTM) models. Next, I developed Long Short-Term Memory (LSTM) models, which are very famous for managing time series data. I created six different LSTM models, with variations in the number of units within the stacked recurrent layers, including 8, 16, and 32 units. Among these options, the 8-unit configuration demonstrated the best performance with an MAE value of 2.55.

For model improvement, I also tried techniques such as recurrent dropout to counter overfitting and leveraged bidirectional data (leading to lower MAE values) which presents information to a recurrent network in diverse ways. These enhancements resulted in similar MAE values for all models, and notably, these values were consistently lower than those of the common sense model, I can also verify the same from the evaluation graph of MAE. Then, I created a model using both RNN and 1D convolution model which gave poor results of 3.81 MAE. This poor performance might be due to the limitation of convolution which is destroying the information order.

Finally, based on the results I observed that using LSTM and GRU (advanced RNN architectures) is preferred and RNN and 1D convolution combination resulted in poor performance. While LSTM stands as a popular choice for handling time series data, after experimenting, I think that GRU is a more efficient choice. The number of units in the stacked recurrent layers, the recurrent dropout rate, and the utilization of bidirectional data are examples of hyperparameters that should be tuned to optimize GRU.