Assignment 3 – Time Series

Objective - The objective is to explore the effectiveness of Recurrent Neural Networks (RNNs) in analyzing time-series data, aiming to enhance their performance and efficiency in forecasting tasks. This involves experimenting with various deep learning techniques to improve the analysis of time-series data.

Model Building – We've constructed a total of 15 models by partitioning the data into training, validation, and testing sets with proportions of 50%, 25%, and 25%, respectively. Currently, our focus is on developing and evaluating a densely connected neural network model for forecasting using the Jena temperature dataset. We assess the model's performance using Mean Absolute Error (MAE).

- The current model architecture comprises one input layer with 14 features representing the Jena temperature dataset, one hidden layer with 16 units, and one output layer predicting the temperature (the target variable).

- The validation Mean Absolute Error (MAE) stands at 2.44, with the test MAE at 2.62. MAE measures the average absolute difference between predicted and actual values. These results serve as a baseline for performance comparison. A more advanced model achieving a lower MAE on both validation and test sets would indicate an improvement over this initial baseline.

- The densely connected model might be overfitting the training data due to its relatively large number of parameters (16 units in the hidden layer) compared to the available training data (819 samples). To address overfitting, regularization techniques such as dropout or L1/L2 regularization need to be applied.

- The 1D Convolutional Neural Network (CNN) model achieved a test MAE of 3.14 network model. One possible reason for this difference could be the presence of pooling layers in CNNs. These layers, intended to reduce data complexity and capture spatial features, might inadvertently remove important long-term dependencies present in time series data. Consequently, the densely connected neural network, which doesn't use pooling layers, might be better at preserving these temporal relationships, leading to its lower test MAE.

- Simple RNN model performance yielded a test MAE of 9.92, significantly higher than both the densely connected neural network model (2.64) and the CNN model (3.14). This poor performance is likely due to the "vanishing gradient" problem inherent in simple RNNs, making them less effective at capturing patterns in sequential data compared to more advanced RNN variants like LSTM and GRU.

- Further attempts to enhance model performance, such as stacking multiple Simple RNN layers, did not yield significant improvements. The test MAE remained around 9.92, indicating difficulty in capturing underlying patterns effectively.

Comparing different RNN variants, the Simple GRU model and the LSTM-Simple model, despite their architectural differences, both exhibited consistently high-test MAE scores. Although the LSTM (RNN) - dropout model required more training time, it achieved the best test MAE score among the three.

Exploring the impact of unit counts in stacked LSTM layers revealed that models with higher unit counts (32 and 64) had higher Mean Absolute Errors (MAEs) compared to models with fewer units (8 and 16), indicating sensitivity to the number of units in the LSTM layers.

Bidirectional RNNs, capturing dependencies in both forward and backward directions of the input sequence, consistently outperformed basic baseline models, demonstrating their effectiveness in time series forecasting.

However, hybrid approaches combining convolutional and RNN models led to unsatisfactory performance due to convolutional methods potentially disrupting the sequential order of information in the data. This underscores the importance of selecting architectures tailored for sequential data, such as RNNs.

- Another hybrid model combining 1D convolutional layers with an LSTM layer aimed to capture both local patterns and long-term dependencies. Despite this approach, the test MAE of 3.77 was higher compared to the bidirectional LSTM model, possibly due to increased complexity introduced by combining convolutional layers with LSTM.

In addition to experimenting with different neural network structures, the way we prepare our data plays a crucial role in improving time series forecasting models. Techniques like scaling, differencing, and windowing can significantly affect how well our models perform. Properly formatting and scaling our input data through preprocessing ensures that our models can effectively capture the underlying patterns. Additionally, thoughtful feature engineering, such as including lagged variables or external factors, can enhance our model's ability to understand the complexities within the data. Therefore, alongside considering the architecture of our models, paying attention to how we preprocess our data and engineer features is vital for achieving the best results in time series forecasting tasks.

In summary, LSTM and GRU models outperformed simple RNN models in time series forecasting tasks due to their ability to capture long-term dependencies. The unsuccessful performance of hybrid models combining convolutional and RNN approaches may stem from the disruptive impact of convolutional methods on the sequential order of information in time series data.