**Time Series RNN**

**Methodology**

The procedure for the time series RNN project includes a framework for preparing data, developing multiple sorts of neural networks, training the established networks, and assessing the performance of the networks when applied to time series in general. Every stage of the approach is thought out in order to consider the temporal relations and to enhance the performance of the model.

**Data Preparation**

The first of the steps being to clean and format the time series data for analysis using the RNN approach. A function to create sequences is formed and it rearranges the data into sequences of a certain number of time steps and their corresponding features. Of this structuring, the ability to learn patterns in successive time intervals enables the model to be made. When the sequences are formed, the entire set of data is separated between the training and testing sets with 80% data used for training and 20% data reserved for testing. Such division helps to ensure that the model is trained on most part of the data and yet there will be data set aside for model testing to check the generality of the model and its accuracy.

**Model Design**

Three distinct neural network models are designed to analyze and compare their effectiveness on the time series data:

**Simple RNN Model:** The Simple RNN model proposed here consists of two recurrent layers with fifty and twenty and five units respectively and a dense layer for output. Its architecture utilizes essence of basic RNN layers which might be capable of capturing temporal connections but weak in terms of detecting long ranged temporal patterns. As of the objective function used in this work, MSE is employed and the model is optimized using Adam as it converges faster.

**LSTM Model:** The LSTM model extends the RNN model to utilise LSTM layers because these are more capable of managing the long-term dependency of data. This model has three LSTM layers with 100, 50 and 25 cells and a dense layer for the output layer. The opposed directions of the LSTM layers allow providing long-term memory which is an essential characteristic in time series users for the use of the model. It uses even simpler RNN and the same loss function as MSE and the same optimizer as Adam.

**CNN-LSTM Model:** Most of the models used in this research are CNN-LSTM which combines CNN to extract features then LSTM to parse the data serially. This design should enable the current model to retain spatial patterns due to CNN layers and temporal features due to LSTM layers and might be beneficial for time series data with a rich structure. The last layer is an output layer which is also dense layer gives out the final prediction.

**Model Training and Evaluation**

Each model is trained with MSE as the loss function and the Mean Absolute Error (MAE) is used as the evaluation measure, which begets the averages of the magnitudes the errors made during all predictions, which is less likely to be skewed by very large errors. The models are trained with batch size of 32 and for a total of 10 cycles of training. Division of training data: During the training of the model, 20% of the training data is kept aside for validation purpose to avoid over training.

In the test set, the predictions of each are made to determine the accuracy of each model and to further compare predicted energy consumption with actual targets. Usually, data points are plotted on scatter plots to compare them to the predictive power of each of the models. This evaluation assists in understanding to what extent each model extended to unseen data and to identify the temporal pattern model.

**Results Interpretation**

**RNN Model**

The findings of the time series model using RNN are positive in terms of predictive accuracy showing significant successful prediction on a test set. The obtained Mean Squared Error (MSE) and Mean Absolute Error (MAE) point to minimal prediction error, with the validation loss of 0.0702 and the validation MAE of 0.1820 indicating good generalising performance on new data. While training, they achieved a train loss of approximately 0.0837 and MAE of 0.1958, and during evaluation the test loss was about 0.0940 and MAE of 0.2027 which maintains the consistency between the train and test. Further, checking true values against predicted ones using the scatter plot exhibit a high correspondence suggesting that the model promotes the identification of temporal relationships within the data. These outcomes indicate that the RNN model is effective for forecasting tasks in this dataset, extraordinarily adaptable to changes in data split training.

**LSTM Model:**

From the project “Time series RNN”, the LSTM model shows the way of accurate representation of time-dependent features of the data. During the training procedure MSE and MAE of the model is observed to decrease with the increase in epochs which shows that the performance of the model is enhancing. In particular, after training, the obtained validation loss equals 53.11 and the validation MAE is about 5.99. When using the similar evaluation on the same test set, the LSTM model gives out a loss of 72.73 and an MAE of 6.91 which revealed the fact that it was performing in a same way on unseen data as well.

The plot of true y values and the predicted ones represents a scattered figure that indicates very high correlation, thereby implying the competence of LSTM in detecting sequential relations. The lower validation error rates and similar performance with the training and test data support the good generality of the current model. This makes the LSTM model ideal for the use in application forecasting that highly rely on time stamped data stream.

**CNN-LSTM Model:**

In the project “Time series RNN”, the CNN-LSTM model was built in order to use both CNN and LSTM layers used to extract spatial and temporal dependencies in the time series. The model starts with the 1D convolutional layers with subsequent max-pooling layers improve the subsequent LSTM processing of the time series data by extracting the most relevant features from it. This is a perfect architecture for the problem at hand since it embeds the CNN layers to learn localized patterns, while the LSTM layers learn the sequential dependencies of the problem.

Evaluation of the CNN-LSTM model yielded a validation loss of about 52.77 and validation Mean Absolute Error, (MAE) of 5.98 only. In the test set, the model had a test loss of approximately 70.78 and the test set MAE was 6.83 thereby showing that the model is operational to generalize in unseen data relatively well. The plotting of the actual and those of the predicted values as a scatter plot indicated significant degree of parallelism, suggesting that the model has the potential for successful identification of temporal patterns.

These results indicate that the proposed CNN-LSTM model is insensitive to the dynamic change of the features and has a good ability of analyzing time series data, which is more suitable for complex time series tasks where the traditional RNNs are incapable of.