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.

This is a Simple Recurrent Neural Network model where the network was trained over 10 epochs to peg the dependency of the target, the variable being temperature. The training process and performance metrics for each epoch are summarized as follows:

Training and Validation Loss

From Epoch 1 to Epoch 10, the value of training loss of the selected RNN model seemed to decrease consistently; from 5.0530 to 0.2698. This also shows that there is a continuous change of weights in the model to enhance the capability of making the right prediction on the temperature. However, the validation loss was somewhat noisy, which saw it increase at Epoch 5, in which the validation loss was 0.3381 before dropping at Epoch 6 and thereafter.

Epoch 1 Loss: 5.0530 (training), 0.1846 (validation).

Epoch 10 Loss: 0.This is for the training dataset; 2698 and for the validation dataset; 0.1458.

Significantly, whereas the training loss continued to decrease, the validation loss increased during the epochs, and this may have caused the model to generalise during Epoch 5 and Epoch 7.

Training and validation MAE is the mean absolute error of a network which indicates an aspect of the capability of a network on how well it can predict target variables.

The MAE, which quantifies the total sum average of the errors for each prediction at each epoch, follow the same trend with increasing epoch number. Firstly, the training MAE was set at 1.1528 at epoch 1ToFile: A Deep Imitation Learning Technique for Lane Keeping Assistance System.doc and was reduced to 0.3823 at epoch 10. The same holds true for the validation MAE, they decreases from 0.3161 in Epoch 1 to 0.3047 in Epoch 10.

Epoch 1 MAE: 1.Training set is 1528, while the validation set is 0.3161.

Epoch 10 MAE: 0.Training accuracy = 3823, Validation accuracy = 0.3047

While both training and validation MAE reduced with the new training, the overall estimation was still higher when compared to more complex models such as LSTM or CNN-LSTM, which showed the simplicity of the chosen RNN model when addressing sequential data, and particularly long-term dependencies.

In this **evaluation**, the RNN model's performance on a test or validation dataset is summarized, based on the final epoch. The results are as follows:

Loss and Mean Absolute Error (MAE)

Loss: 0.1796

Mean Absolute Error (MAE): 0.3130

The loss value indicates that the model performed reasonably well in minimizing its objective function during training. A loss of 0.1796 suggests that the model was able to learn the underlying patterns in the data effectively, although there may still be room for improvement in terms of reducing the error further.

The MAE metric, which represents the average magnitude of the errors in the model's predictions, is 0.3130. This shows that on average, the model's predictions were off by approximately 0.3130 units from the actual values. Although this error is relatively low, further improvements could be achieved by exploring advanced models or fine-tuning the current model.

Time Efficiency

The model completed the evaluation on the test data in 14 seconds with a processing time of 5 milliseconds per step, suggesting that the RNN model is computationally efficient. This is beneficial when working with large datasets or in real-time prediction applications where time constraints are important.

Model Evaluation Insights

While the loss and MAE are reasonable for an RNN model, the relatively modest performance in comparison to more advanced architectures (such as LSTM or CNN-LSTM) highlights the limitations of the simple RNN. The model performed well in

learning patterns in the data, but further tuning or switching to a more complex architecture might yield better results for capturing longer-term dependencies.

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.

The loss value of 0.0940 and MAE of 0.2027 indicate that the model's predictions are relatively close to the actual values. A lower loss and MAE typically suggest a well-trained model that can make accurate predictions with minimal error, especially for regression tasks.

The training process achieved this performance in a relatively short time (19 seconds per epoch, 7 ms per step). This suggests the model is computationally efficient, which is particularly beneficial for real-time or resource-constrained applications.

Since the loss and MAE values are relatively low, it suggests that the model has converged well after training, and no major overfitting or underfitting is likely occurring. The model is generalizing well to unseen data, which is crucial for its performance on test or production data.

The MAE of 0.2027 suggests that, on average, the model's predictions deviate by approximately 0.20 units from the actual values. Depending on the range of the output

variable, this might be an acceptable level of error, indicating that further improvements in model tuning (such as adjusting the learning rate or experimenting with deeper architectures) could potentially lead to marginal gains in accuracy.

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.

The model described is a hybrid deep learning architecture combining 1D Convolutional Neural Networks (Conv1D), MaxPooling layers, and Long Short-Term Memory (LSTM) layers. This structure is designed to effectively process sequential data by capturing both local patterns through convolution and long-term dependencies through LSTM layers. The first layer is a Conv1D layer with 64 filters, which extracts local features from the input sequence. This is followed by a MaxPooling1D layer that reduces the sequence length, retaining the most prominent features. The second Conv1D layer, with 128 filters, captures deeper patterns, while another MaxPooling1D layer further downsamples the sequence. Two LSTM layers follow, the first capturing long-term dependencies in the sequence and the second refining the features to a fixed-size representation. The final dense layer produces a scalar output, typically used for regression tasks. The model contains a total of 68,770 parameters, all of which are trainable, reflecting its substantial complexity and capacity to learn intricate patterns in the data. This architecture is well-suited for tasks that require both local feature

extraction and sequential learning, such as time series analysis or other sequence-based problems, though training may require significant computational resources.

According to the results, in the model the loss is 72.7980 and MAE = 6.9147. From these values we can deduce that the performance of the model is not at its best given by the fact that both measures provide an indication of variation especially in reference to the actual values. The high loss indicates that scale of the error is high and the high MAE reveals that on an average the predictions are off by 6.91 units which in certain circumstances might not be quite acceptable.

It takes 9 seconds to finish each epoch, and each step has only 3 millisecond of latency. This means that during training the model is computationally efficient, thus one can be able to do many iterations in a short span. However, the efficiency that has been seen is reflected by the performance measurements indicators showing that the model is not properly learning the Samples seldom if at all, therefore indicating that there might be a problem in the architectural design of the model or data preparation or even some of the hyperparameters set.

It is the CNN layers followed by LSTM layers for the time series or sequential data standard configuration because CNNs are good at feature extraction and LSTMs deal with sequential dependencies. A high value of loss and MAE can also mean that this specific architecture is not suitable for the given dataset or task, or that the model should be improved through, for example, the increased number of layers, the optimization of the number of neurons in each layer, or trying to use various techniques to reduce overfitting.

Due to high loss and MAE, there is a very high chance of underfitting which means the model created is too simple to analyze the data. It is also may be caused by the overfitting of the model and that is may not generalize well between the training set and the validation set.

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.