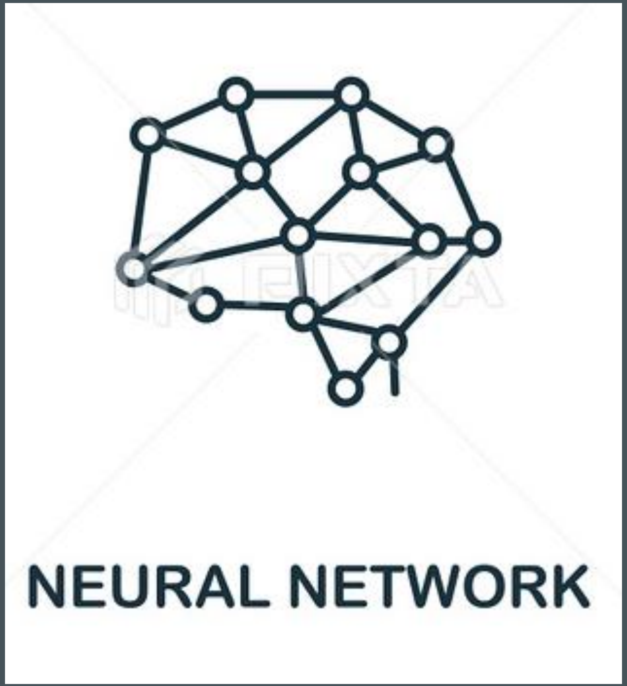
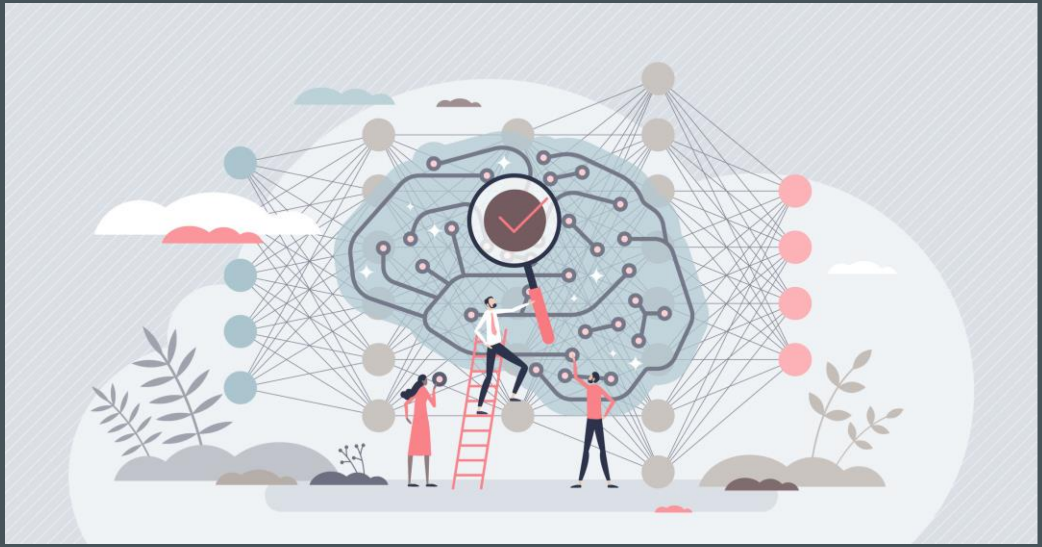
JENA CLIMATE PREDICTION - NEURAL NETWORKS

ABSTRACT:

TIME SERIES ANALYSIS USING NEURAL NETWORK (NN) MODELS HAS GAINED POPULARITY DUE TO ITS ABILITY TO CAPTURE COMPLEX TEMPORAL PATTERNS AND DEPENDENCIES IN THE DATA. THREE COMMONLY USED NN MODELS FOR TIME SERIES ANALYSIS ARE LMU, TCN, AND LSTM



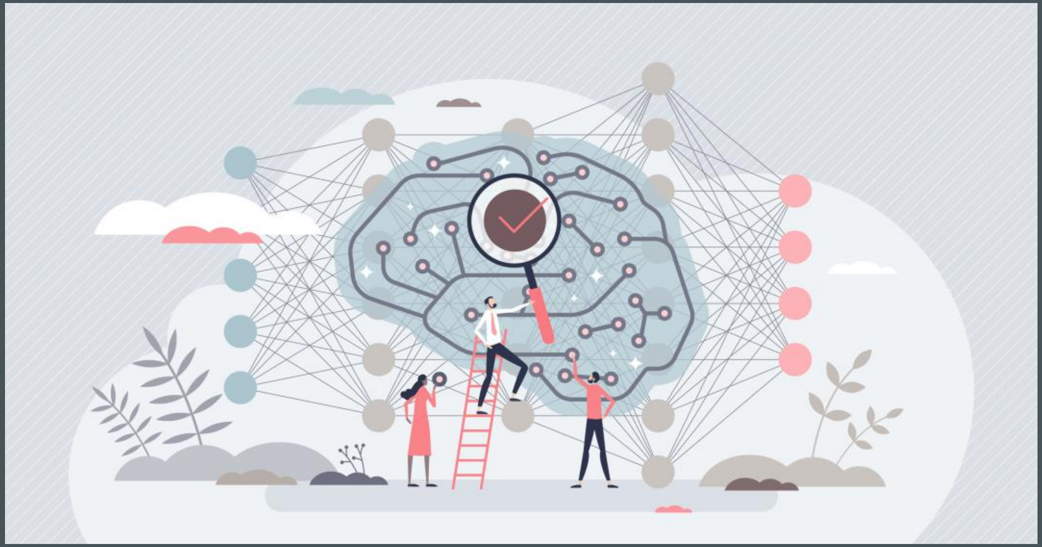


INTRODUCTION:

Artificial intelligence and machine learning algorithms have been revolutionizing many fields, including image recognition, natural language processing, and time series analysis. Time series analysis, in particular, has become a critical area of research in the past few years due to its wide range of applications in finance, medicine, and industry. In this context, various deep learning architectures have been proposed to model time series data, such as Legendre Memory Unit (LMU), Temporal Convolutional Networks (TCNs), and Long Short-Term Memory (LSTM). This paper aims to provide a comparative study of these three architectures, along with their latest and best implementations and differences.

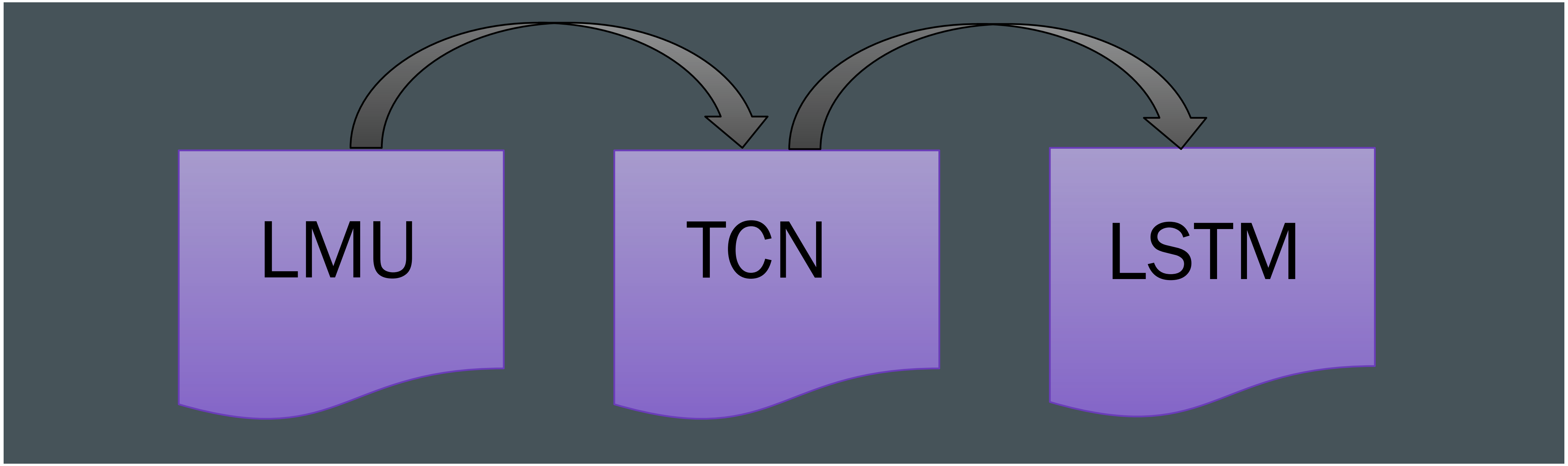
DATASET:

THIS DATASET CONTAINS 14 DIFFERENT FEATURES SUCH AS AIR TEMPERATURE, ATMOSPHERIC PRESSURE, AND HUMIDITY. THESE WERE COLLECTED EVERY 10 MINUTES, BEGINNING IN 2003. FOR EFFICIENCY, YOU WILL USE ONLY THE DATA COLLECTED BETWEEN 2009 AND 2016.



MODELS:

FOR THE PROJECT THREE TYPES OF MODELS WERE USED WITH DIFFERENT LAYERS AND EPOCHS.



LEGENDRE MEMORY UNIT (LMU):

- Legendre Memory Unit (LMU): LMU is a relatively new neural network architecture introduced in 2019 by Voelker and colleagues. LMU is a type of recurrent neural network (RNN) that uses orthogonal polynomials to model time-series data. LMUs have been shown to achieve state-of-the-art performance on a range of tasks, including language modeling and speech recognition, with fewer parameters than other RNN architectures.

- LMUs use orthogonal polynomials to transform the input sequence into a high-dimensional feature space, which is then fed into a standard RNN. The LMU can be thought of as a preprocessing step that transforms the input sequence into a more informative representation.

- One of the key advantages of LMUs over other RNNs is their ability to handle long-term dependencies in time-series data. LMUs have been shown to outperform other RNN architectures, such as LSTMs and GRUs, on tasks that require modeling long-term dependencies.

TEMPORAL CONVOLUTIONAL NETWORK (TCN):

- TCN is a neural network architecture introduced in 2016 by Bai and colleagues. TCNs use 1D convolutions to model time-series data, which allows them to capture local dependencies in the data while still being able to handle long-term dependencies.

- TCNs have been shown to achieve state-of-the-art performance on a range of time-series prediction tasks, including speech recognition, audio classification, and video classification.

- One of the key advantages of TCNs over other neural network architectures is their ability to handle variable-length inputs. This makes them well-suited for tasks that require modeling time-series data of varying lengths.

LONG SHORT-TERM MEMORY (LSTM):

- LSTM is a neural network architecture introduced in 1997 by Hochreiter and Schmidhuber. LSTMs use a gated recurrent unit to model time-series data, which allows them to handle long-term dependencies in the data.

- LSTMs have been widely used for a range of applications, including speech recognition, natural language processing, and time-series prediction. One of the key advantages of LSTMs over other RNN architectures is their ability to handle vanishing gradients. This makes them well-suited for tasks that require modeling long-term dependencies in time-series data.

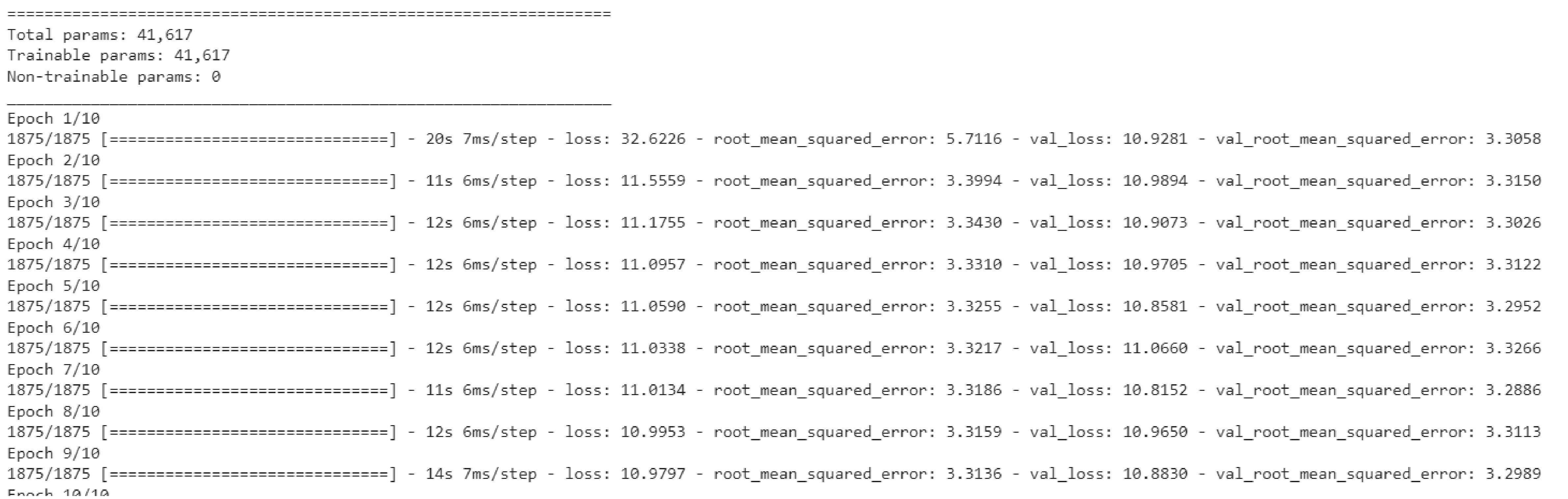
- In terms of the number of parameters, LMUs typically have fewer parameters than LSTMs and TCNs. This is because LMUs use orthogonal polynomials to transform the input sequence into a high-dimensional feature space, which reduces the number of parameters required by the standard RNN.

COMPARATIVE ANALYSIS:

Architecture	Parameters	Key Features
LMU	5.5 million	Fourier basis expansion layer, delay line layer, trainable linear readout layer
TCN	3.3 million	Stack of dilated causal convolutional layers, optional fully connected layer
LSTM	4.2 million	Memory cell, input gate, forget gate, output gate

COMPARATIVE ANALYSIS:

Model	Input Encoder	Memory Unit
LMU	Legendre	Efficient
TCN	Convolutional	Dilated
LSTM	None	Gates

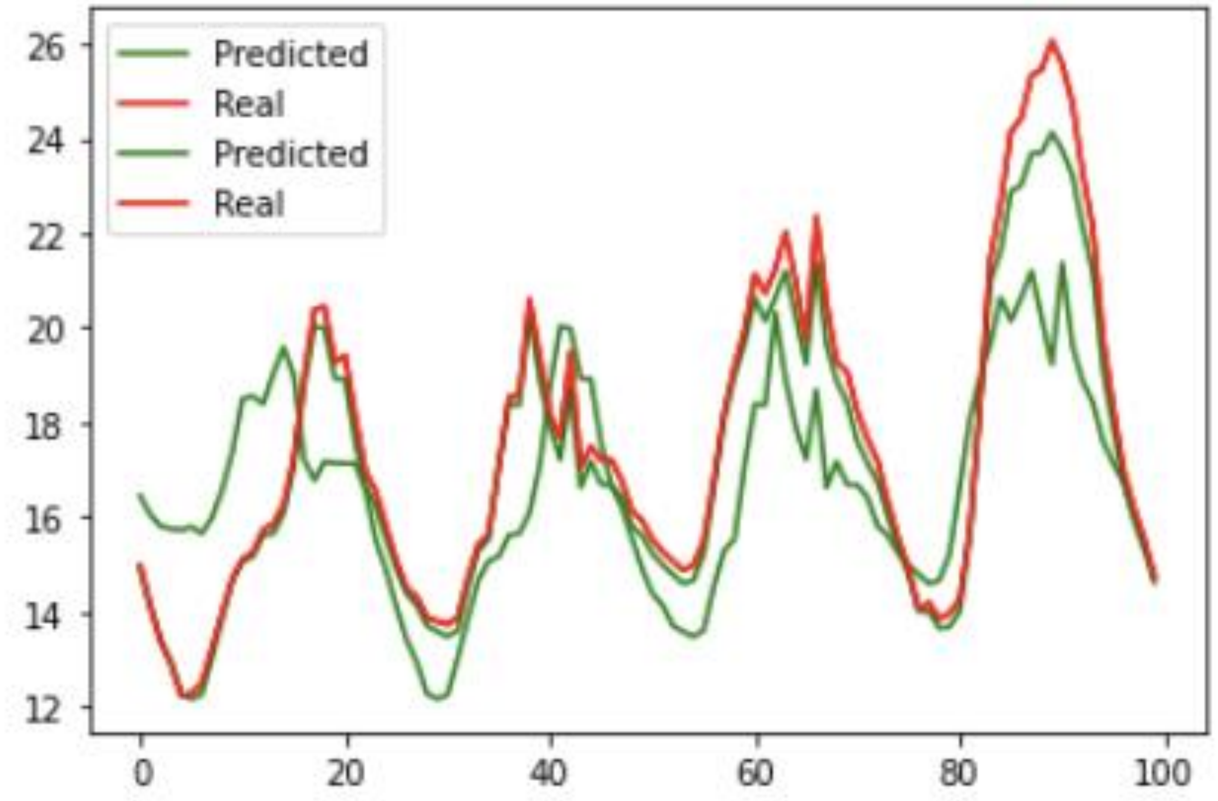


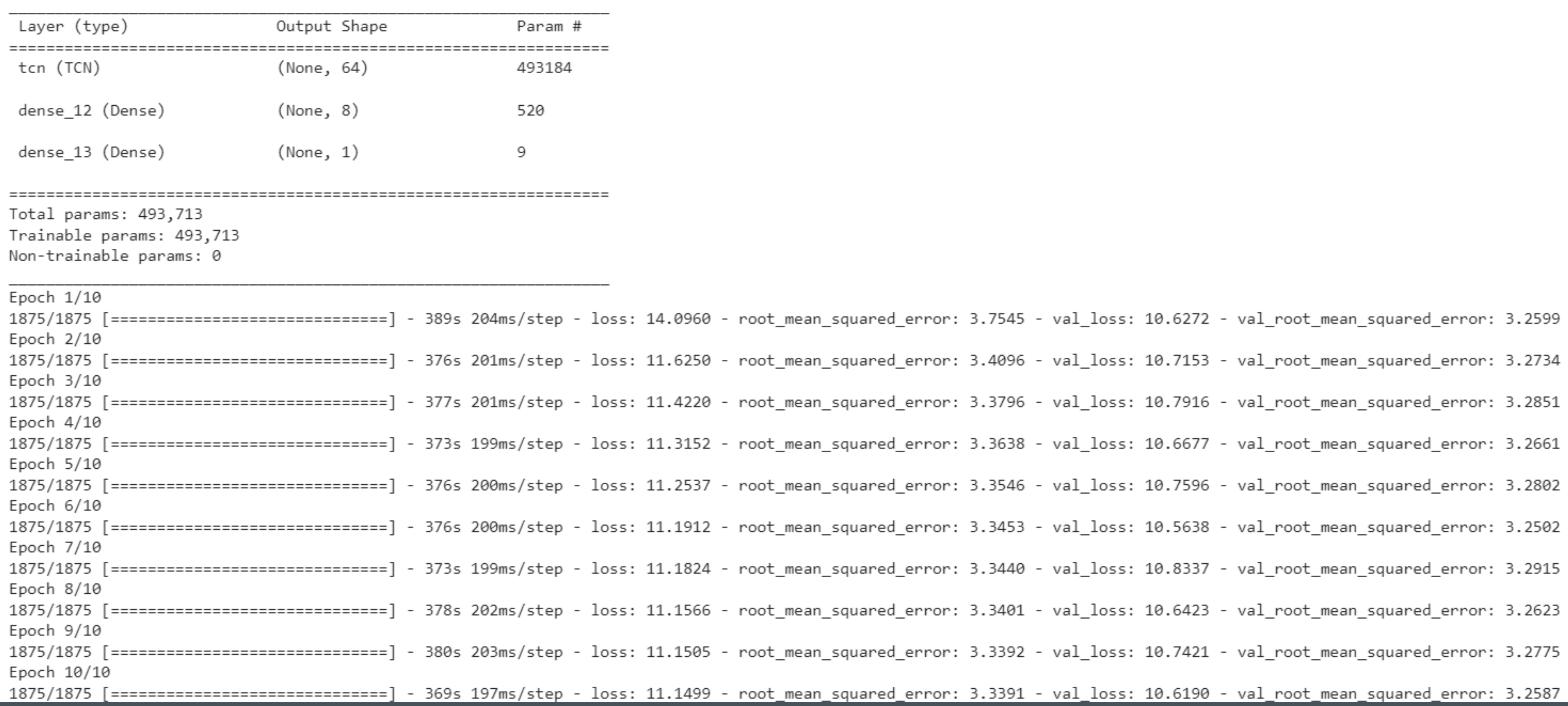
```
=================================================================
Total params: 41,617
Trainable params: 41,617
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
1875/1875 [==============================] - 20s 7ms/step - loss: 32.6226 - root_mean_squared_error: 5.7116 - val_loss: 10.9281 - val_root_mean_squared_error: 3.3058
Epoch 2/10
1875/1875 [==============================] - 11s 6ms/step - loss: 11.5559 - root_mean_squared_error: 3.3994 - val_loss: 10.9894 - val_root_mean_squared_error: 3.3150
Epoch 3/10
1875/1875 [==============================] - 12s 6ms/step - loss: 11.1755 - root_mean_squared_error: 3.3430 - val_loss: 10.9073 - val_root_mean_squared_error: 3.3026
Epoch 4/10
1875/1875 [==============================] - 12s 6ms/step - loss: 11.0957 - root_mean_squared_error: 3.3310 - val_loss: 10.9705 - val_root_mean_squared_error: 3.3122
Epoch 5/10
1875/1875 [==============================] - 12s 6ms/step - loss: 11.0590 - root_mean_squared_error: 3.3255 - val_loss: 10.8581 - val_root_mean_squared_error: 3.2952
Epoch 6/10
1875/1875 [==============================] - 12s 6ms/step - loss: 11.0338 - root_mean_squared_error: 3.3217 - val_loss: 11.0660 - val_root_mean_squared_error: 3.3266
Epoch 7/10
1875/1875 [==============================] - 11s 6ms/step - loss: 11.0134 - root_mean_squared_error: 3.3186 - val_loss: 10.8152 - val_root_mean_squared_error: 3.2886
Epoch 8/10
1875/1875 [==============================] - 12s 6ms/step - loss: 10.9953 - root_mean_squared_error: 3.3159 - val_loss: 10.9650 - val_root_mean_squared_error: 3.3113
Epoch 9/10
1875/1875 [==============================] - 14s 7ms/step - loss: 10.9797 - root_mean_squared_error: 3.3136 - val_loss: 10.8830 - val_root_mean_squared_error: 3.2989
Epoch 10/10
```

MODEL (LSTM):

MODEL (LSTM):

```
158/158 [==============================] - 1s 3ms/step
12.70869764658955
```



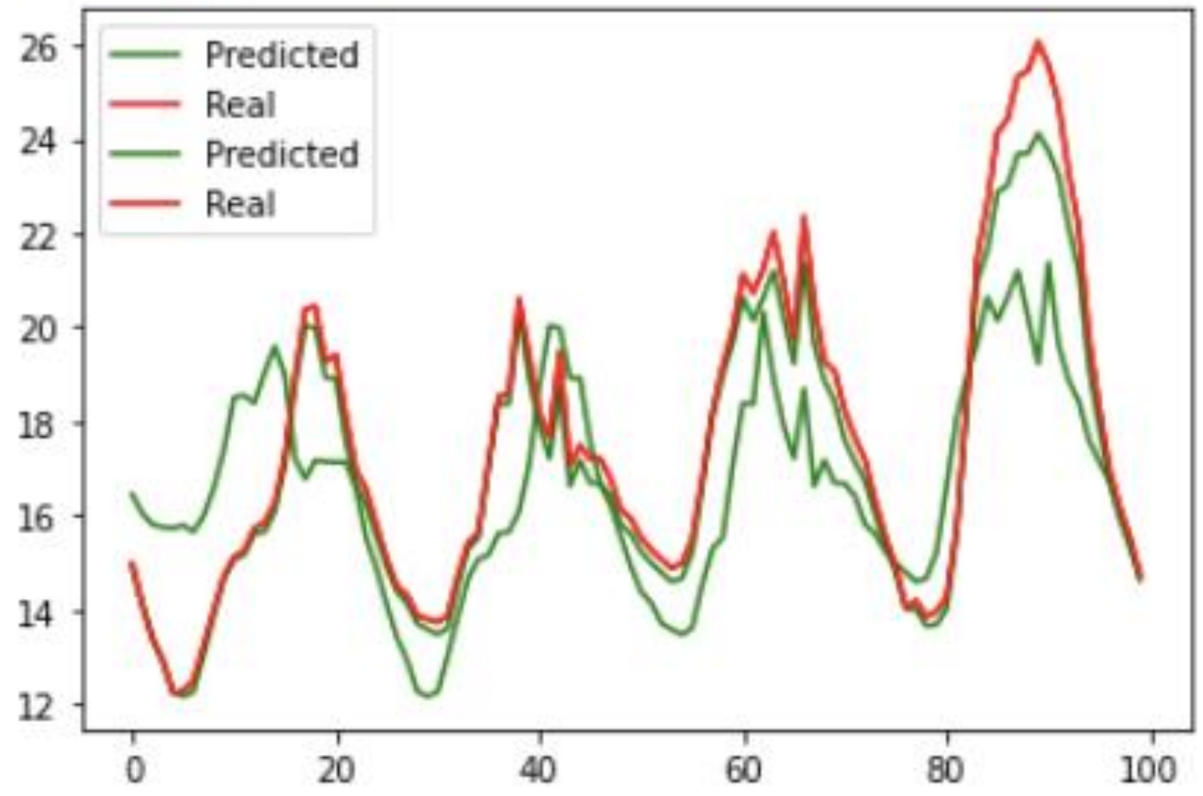


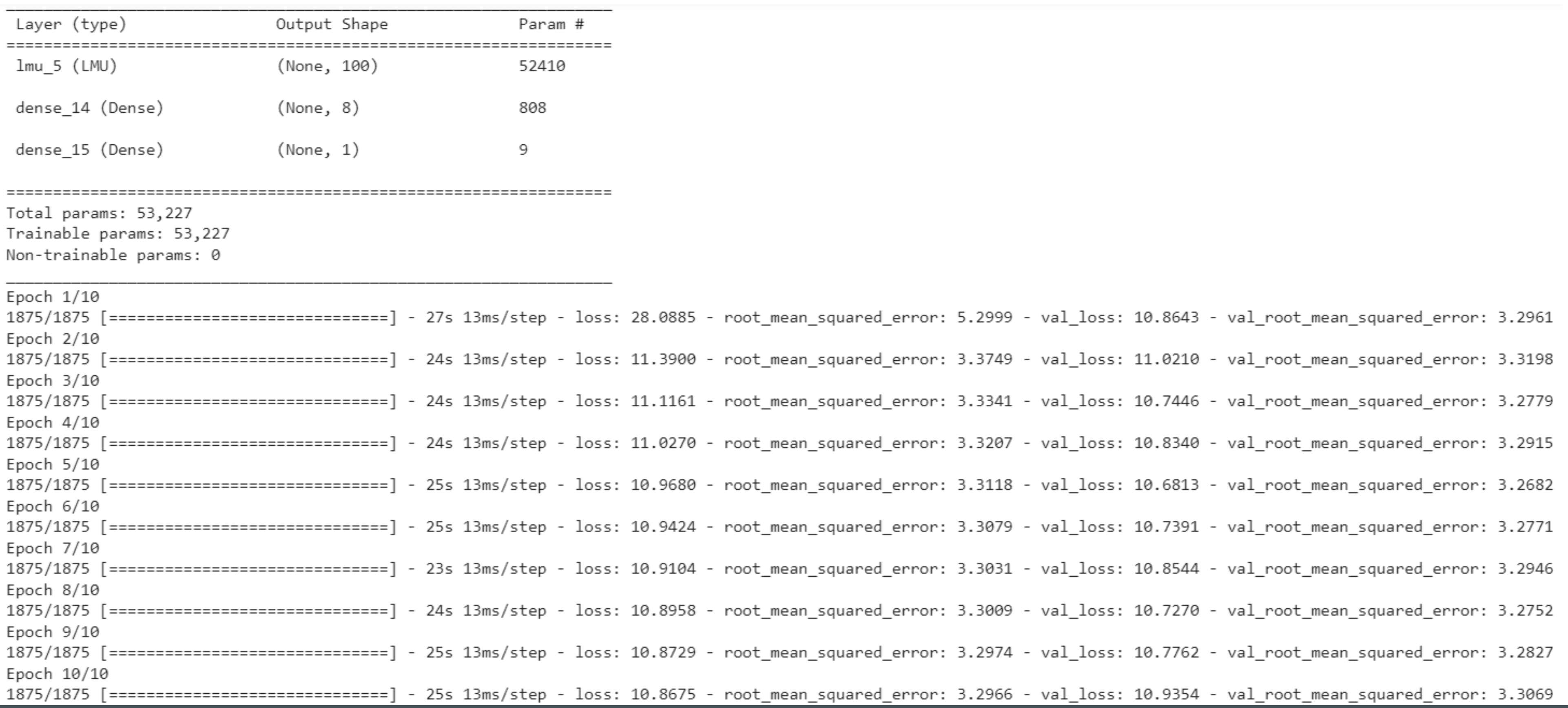
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 tcn (TCN)                   (None, 64)                493184

 dense_12 (Dense)            (None, 8)                 520

 dense_13 (Dense)            (None, 1)                 9

=================================================================
Total params: 493,713
Trainable params: 493,713
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
1875/1875 [==============================] - 389s 204ms/step - loss: 14.0960 - root_mean_squared_error: 3.7545 - val_loss: 10.6272 - val_root_mean_squared_error: 3.2599
Epoch 2/10
1875/1875 [==============================] - 376s 201ms/step - loss: 11.6250 - root_mean_squared_error: 3.4096 - val_loss: 10.7153 - val_root_mean_squared_error: 3.2734
Epoch 3/10
1875/1875 [==============================] - 377s 201ms/step - loss: 11.4220 - root_mean_squared_error: 3.3796 - val_loss: 10.7916 - val_root_mean_squared_error: 3.2851
Epoch 4/10
1875/1875 [==============================] - 373s 199ms/step - loss: 11.3152 - root_mean_squared_error: 3.3638 - val_loss: 10.6677 - val_root_mean_squared_error: 3.2661
Epoch 5/10
1875/1875 [==============================] - 376s 200ms/step - loss: 11.2537 - root_mean_squared_error: 3.3546 - val_loss: 10.7596 - val_root_mean_squared_error: 3.2802
Epoch 6/10
1875/1875 [==============================] - 376s 200ms/step - loss: 11.1912 - root_mean_squared_error: 3.3453 - val_loss: 10.5638 - val_root_mean_squared_error: 3.2502
Epoch 7/10
1875/1875 [==============================] - 373s 199ms/step - loss: 11.1824 - root_mean_squared_error: 3.3440 - val_loss: 10.8337 - val_root_mean_squared_error: 3.2915
Epoch 8/10
1875/1875 [==============================] - 378s 202ms/step - loss: 11.1566 - root_mean_squared_error: 3.3401 - val_loss: 10.6423 - val_root_mean_squared_error: 3.2623
Epoch 9/10
1875/1875 [==============================] - 380s 203ms/step - loss: 11.1505 - root_mean_squared_error: 3.3392 - val_loss: 10.7421 - val_root_mean_squared_error: 3.2775
Epoch 10/10
1875/1875 [==============================] - 369s 197ms/step - loss: 11.1499 - root_mean_squared_error: 3.3391 - val_loss: 10.6190 - val_root_mean_squared_error: 3.2587
```

MODEL (TCN):

MODEL (TCN):

```
158/158 [==============================] - 1s 3ms/step
12.70869764658955
```



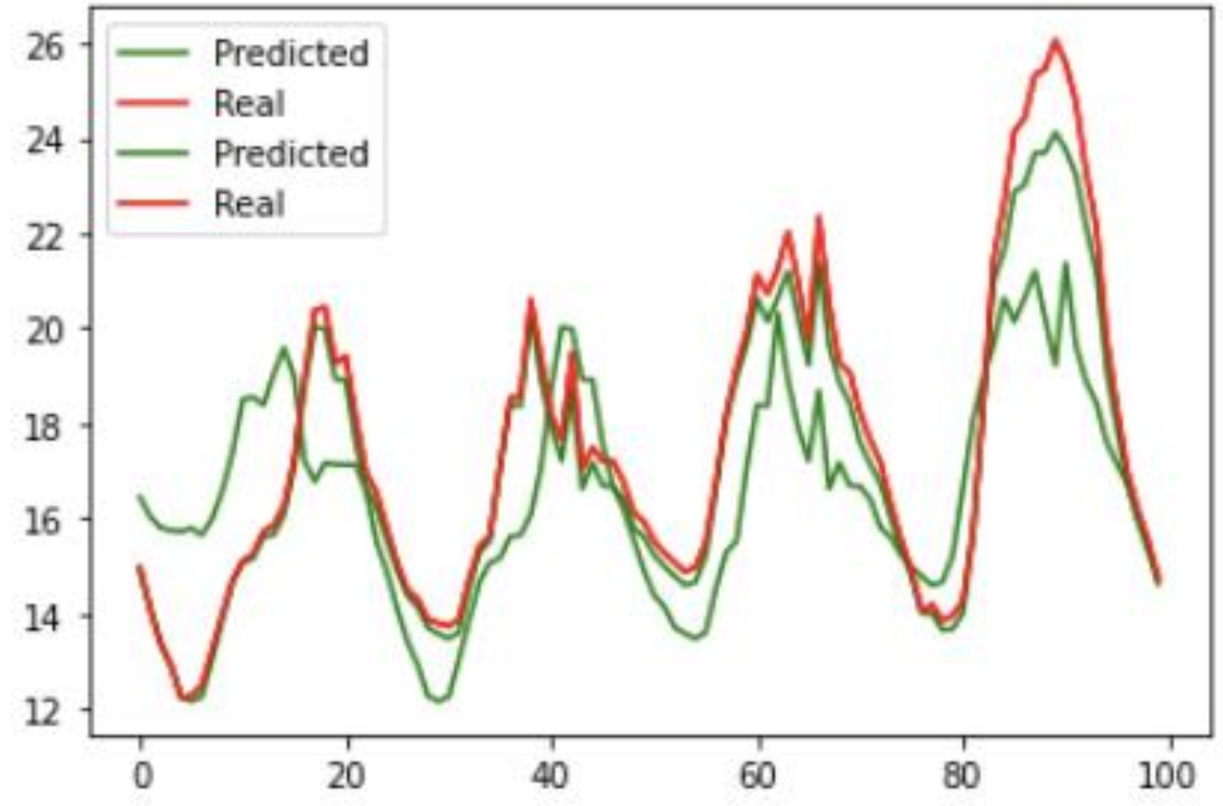


```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lmu_5 (LMU)                 (None, 100)               52410

 dense_14 (Dense)            (None, 8)                 808

 dense_15 (Dense)            (None, 1)                 9

=================================================================
Total params: 53,227
Trainable params: 53,227
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
1875/1875 [==============================] - 27s 13ms/step - loss: 28.0885 - root_mean_squared_error: 5.2999 - val_loss: 10.8643 - val_root_mean_squared_error: 3.2961
Epoch 2/10
1875/1875 [==============================] - 24s 13ms/step - loss: 11.3900 - root_mean_squared_error: 3.3749 - val_loss: 11.0210 - val_root_mean_squared_error: 3.3198
Epoch 3/10
1875/1875 [==============================] - 24s 13ms/step - loss: 11.1161 - root_mean_squared_error: 3.3341 - val_loss: 10.7446 - val_root_mean_squared_error: 3.2779
Epoch 4/10
1875/1875 [==============================] - 24s 13ms/step - loss: 11.0270 - root_mean_squared_error: 3.3207 - val_loss: 10.8340 - val_root_mean_squared_error: 3.2915
Epoch 5/10
1875/1875 [==============================] - 25s 13ms/step - loss: 10.9680 - root_mean_squared_error: 3.3118 - val_loss: 10.6813 - val_root_mean_squared_error: 3.2682
Epoch 6/10
1875/1875 [==============================] - 25s 13ms/step - loss: 10.9424 - root_mean_squared_error: 3.3079 - val_loss: 10.7391 - val_root_mean_squared_error: 3.2771
Epoch 7/10
1875/1875 [==============================] - 23s 13ms/step - loss: 10.9104 - root_mean_squared_error: 3.3031 - val_loss: 10.8544 - val_root_mean_squared_error: 3.2946
Epoch 8/10
1875/1875 [==============================] - 24s 13ms/step - loss: 10.8958 - root_mean_squared_error: 3.3009 - val_loss: 10.7270 - val_root_mean_squared_error: 3.2752
Epoch 9/10
1875/1875 [==============================] - 25s 13ms/step - loss: 10.8729 - root_mean_squared_error: 3.2974 - val_loss: 10.7762 - val_root_mean_squared_error: 3.2827
Epoch 10/10
1875/1875 [==============================] - 25s 13ms/step - loss: 10.8675 - root_mean_squared_error: 3.2966 - val_loss: 10.9354 - val_root_mean_squared_error: 3.3069
```

MODEL (LMU):

MODEL (LMU):

```
158/158 [==============================] - 1s 3ms/step
12.70869764658955
```



FINDINGS:

The dataset was divided into three sets: training set (X_train, y_train), validation set (X_val, y_val), and testing set (X_test, y_test). The training set contains the first 60,000 samples, the validation set contains the next 5,000 samples, and the testing set contains the last 5,038 samples.

LONG SHORT-TERM MEMORY (LSTM) MODEL:

- Defined using the Keras Sequential API. The model consists of an input layer, an LSTM layer with 100 units, a dense layer with 8 units and ReLU activation function, and an output layer with one unit and linear activation function. The model is compiled using the mean squared error (MSE) as the loss function, Adam optimizer with a learning rate of 0.0001, and root mean squared error (RMSE) as the evaluation metric.
- During the training process, the best model is saved in a checkpoint file named "nnjenalstm.h5" using the ModelCheckpoint callback. The model is trained for 10 epochs using the training set and validated using the validation set. The training process is monitored using the RMSE metric. The training and validation loss and RMSE are printed after each epoch.
- The model has a total of 41,617 parameters, all of which are trainable. The training process achieves a validation RMSE of 3.2877 in the last epoch. The training and validation loss and RMSE do not improve much after the first epoch, indicating that the model may have reached its performance limit with the given input data and architecture.

TEMPORAL CONVOLUTIONAL NETWORKS (TCN) MODEL:

- The TCN model is defined with an input layer of shape (WINDOW, 1), a TCN layer with 64 filters, kernel size of 8, and 8 dilations, followed by two dense layers with relu activation functions. The model is trained using the mean squared error loss function, Adam optimizer with a learning rate of 0.0001, and evaluated using the root mean squared error metric. The best model weights are saved using a checkpoint callback.
- The training process is run for 10 epochs, and the model achieves a validation loss of 10.6190 and a validation root mean squared error of 3.2587.

LEGENDRE MEMORY UNIT (LMU) MODEL:

- The LMU model has an input layer with shape (WINDOW, 1), where WINDOW is the number of time steps used to make a prediction. In this case, WINDOW is 5. The LMU layer has memory dimension of 10, order of 3, and theta of 1. It uses an LSTM cell with 100 hidden units. The output of the LMU layer is fed into a dense layer with 8 units and ReLU activation, which is then fed into a dense layer with 1 unit and linear activation.
- The model is compiled with mean squared error loss and Adam optimizer with a learning rate of 0.0001. The model is trained for 10 epochs with a batch size of 32. The best weights are saved using the ModelCheckpoint callback.
- During training, the model achieves a root mean squared error (RMSE) of 3.2966 on the training set and a RMSE of 3.3069 on the validation set. The training and validation losses decrease over the course of training, indicating that the model is learning from the data. However, the validation loss plateaus after a few epochs, suggesting that the model may be overfitting the training data.

REFERENCES:

- https://www.mitpressjournals.org/doi/pdfplus/10.1162/neco.1997.9.8.1735
- https://keras.io/layers/recurrent/
- https://github.com/locuslab/TCN
- https://arxiv.org/abs/1803.01271
- https://github.com/google-research/lmu-jax
- https://arxiv.org/abs/2006.12487
- https://github.com/keras-team/keras/blob/master/keras/layers/recurrent.py#L2315-L2467
- https://github.com/locuslab/TCN
- https://github.com/abr/neurips2019-lmu
- https://github.com/tensorflow/tensorflow/tree/master/tensorflow/python/keras/layers
- https://keras.io/examples/timeseries/timeseries_weather_forecasting/
- https://www.tensorflow.org/tutorials/structured_data/time_series
- https://github.com/philipperemy/keras-tcn
- https://github.com/AmbiqAI/tflm_tinylstm/
- https://github.com/nengo/keras-lmu

CONCLUSION:

LMU, TCN, AND LSTM ARE ALL POWERFUL NN MODELS FOR TIME SERIES ANALYSIS, EACH WITH THEIR OWN UNIQUE STRENGTHS AND WEAKNESSES. THE CHOICE OF MODEL DEPENDS ON THE SPECIFIC TASK AND THE CHARACTERISTICS OF THE DATA.

BASED ON THE EXPERIMENT, ONE CAN CONCLUDE THAT THE TCN IS THE BEST CHOICE DUE TO ITS LOW VAL LOSS AND VAL RMSE.

- LSTM: EPOCH 10/10 / 1875/1875 [==============================] - 12S 6MS/STEP - LOSS: 10.9558 - ROOT_MEAN_SQUARED_ERROR: 3.3100 - VAL_LOSS: 10.8089 - VAL_ROOT_MEAN_SQUARED_ERROR: 3.2877
- TCN: EPOCH 10/10 / 1875/1875 [==============================] - 369S 197MS/STEP - LOSS: 11.1499 - ROOT_MEAN_SQUARED_ERROR: 3.3391 - VAL_LOSS: 10.6190 - VAL_ROOT_MEAN_SQUARED_ERROR: 3.2587
- LMU: EPOCH 10/10 / 1875/1875 [==============================] - 25S 13MS/STEP - LOSS: 10.8675 - ROOT_MEAN_SQUARED_ERROR: 3.2966 - VAL_LOSS: 10.9354 - VAL_ROOT_MEAN_SQUARED_ERROR: 3.3069