Assignment – 3

Time Series Data

Report

Rana

In this temperature-forecasting exercise, we aim to illustrate the main differences between time-series data and other datasets we've encountered before. It will become clear that recurrent neural networks (RNNs), a specialized type of machine learning, are particularly well-suited for this kind of problem, whereas convolutional and densely connected networks have difficulty handling such data.

Throughout our studies, we will divide the data into three parts: 50% for training, 25% for validation, and 25% for testing. When dealing with time-series data, it is essential to ensure that the validation and testing datasets consist of more recent information than the training dataset. This approach is necessary because our objective is to forecast future events using historical data, rather than the reverse.

This is what the test and validation splits should demonstrate. The problem will be formulated as follows: Is it possible to predict the temperature for a given day using hourly data collected over the previous five days?

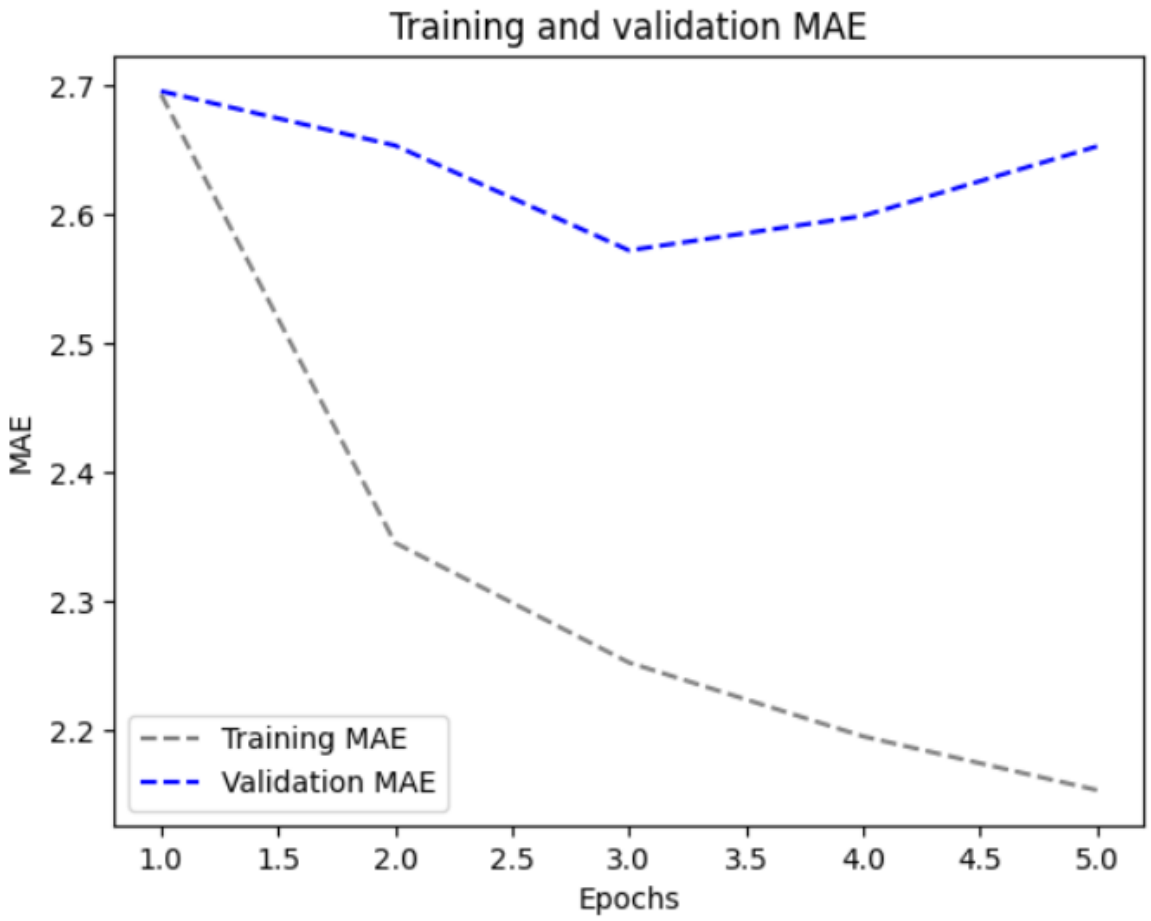
Initially, the data undergoes preprocessing to prepare it for training a neural network. Since the data is already in numerical format, vectorization is unnecessary. We developed 14 models to evaluate the time series data. The first model, which used common-sense techniques, achieved a Mean Absolute Error (MAE) of 2.44, serving as a baseline. In contrast, our dense machine learning model recorded a slightly higher MAE of 2.62. Flattening the time series data led to a loss of temporal context, which adversely affected the performance of the dense model. While some validation losses approached the no-learning baseline, this was not consistent across all models.

Models used in the assignment and the result plots:

Basic Dense Layer: A basic dense layer, or fully connected layer, is a key element of neural networks in which each neuron connects to every neuron in the preceding layer. This layer

computes a weighted sum of its inputs and uses an activation function to generate its output. This procedure illustrates how to create a straightforward neural network for time series forecasting, highlighting the significance of validation during the training process and assessing the final model on a distinct test set.

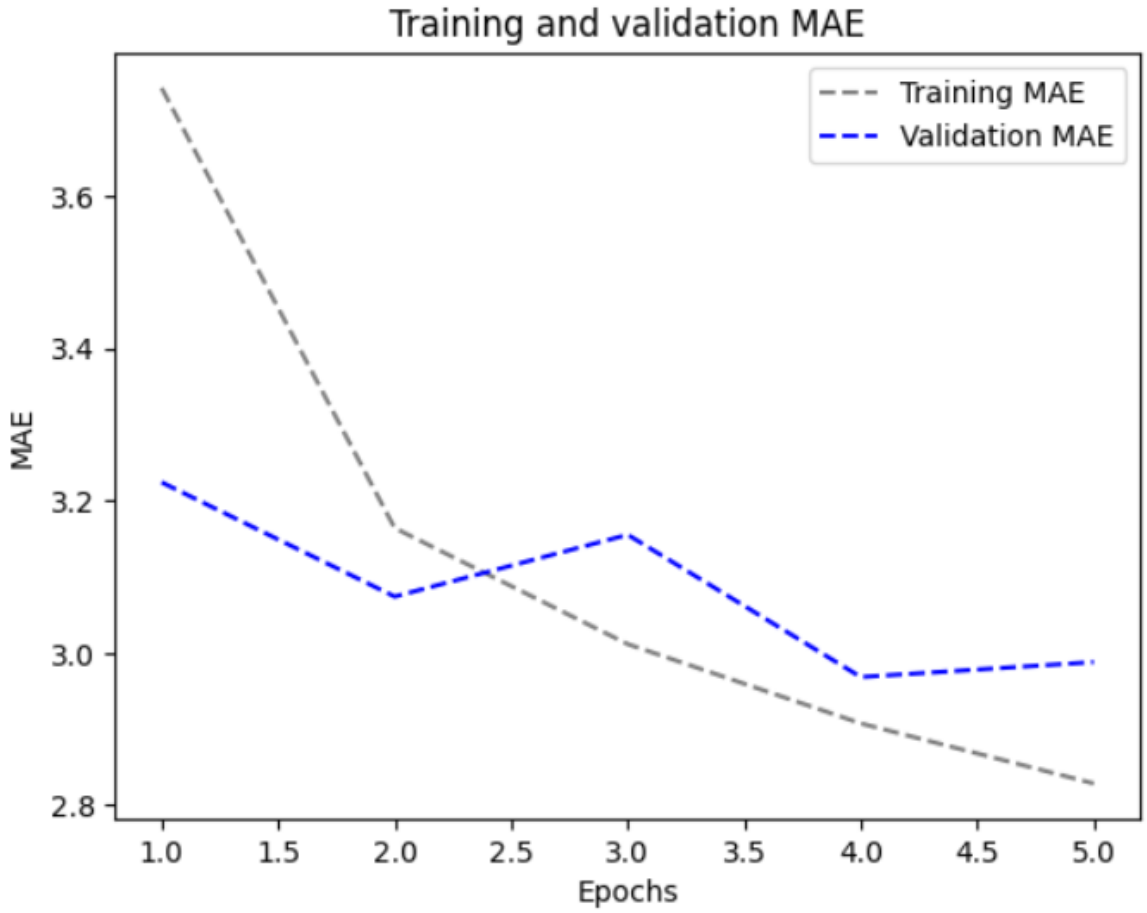
Training and Validation MAE Plot:



1D Convolutional Model:

By leveraging suitable architectural priors, a convolutional model is effective for input sequences that exhibit daily cycles. Just as a temporal convolutional network can apply the same representations across multiple days, a spatial convolutional network utilizes representations in various locations within an image. However, the convolutional model underperforms significantly compared to the densely connected model because not all meteorological data

fulfills the translation invariance assumption. As a result, the convolutional model attains a validation Mean Absolute Error (MAE) of approximately 3.15 degrees, which is notably higher than an acceptable baseline.



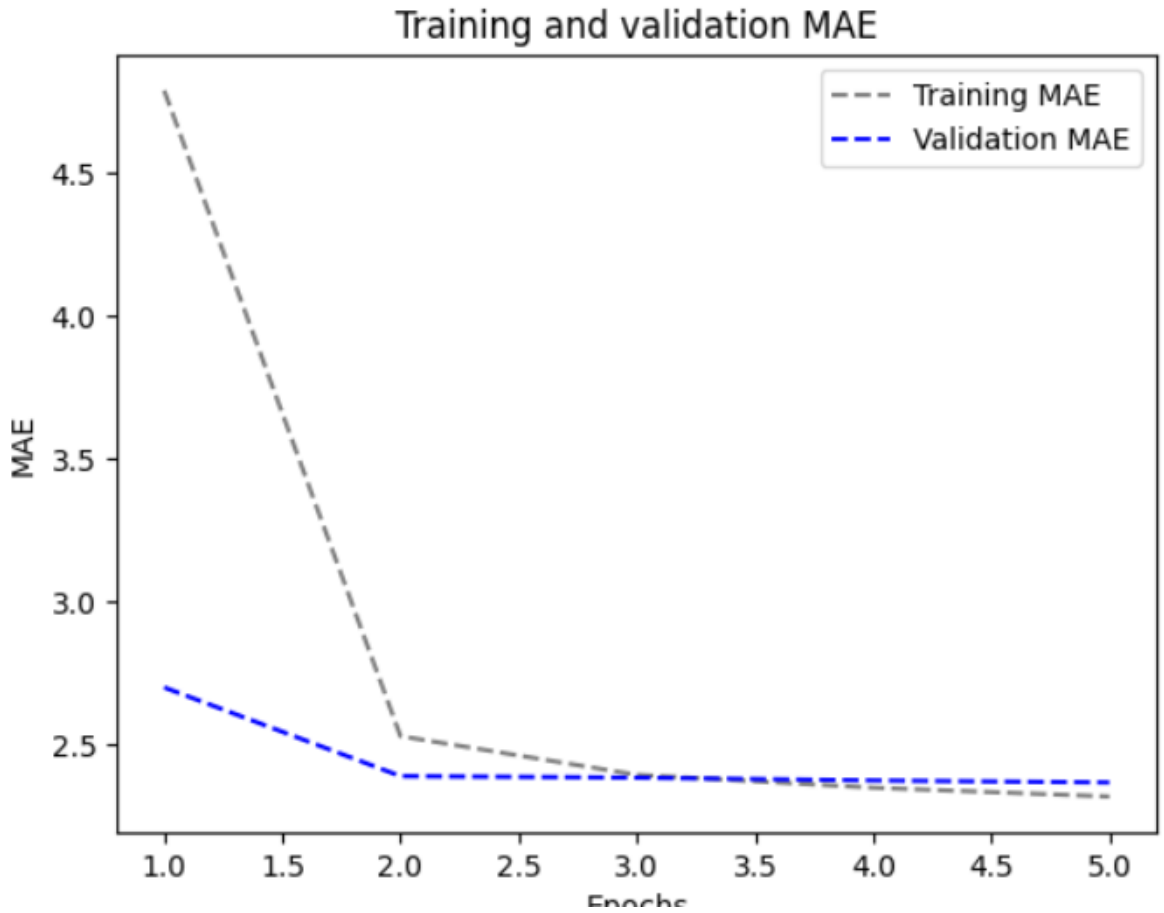
A Simple RNN:

Recurrent neural networks (RNNs) are particularly adept at integrating information from prior time steps during decision-making, which allows them to identify intricate patterns and relationships in sequential data. The internal state of an RNN serves as a memory for earlier inputs, enabling it to manage sequences of varying lengths. While a basic RNN can theoretically remember information from all previous time steps, practical issues such as the vanishing gradient problem hinder the training of deeper networks. As demonstrated in the graph, the simplest RNN shows the least effectiveness.

Gated Recurrent Unit (GRU):

To mitigate this challenge, we will implement LSTM and GRU RNNs using Keras. The Gated Recurrent Unit (GRU) layers will be used instead of LSTM layers, as GRU and LSTM architectures are quite similar, with GRU being a more streamlined and simplified variant of LSTM.

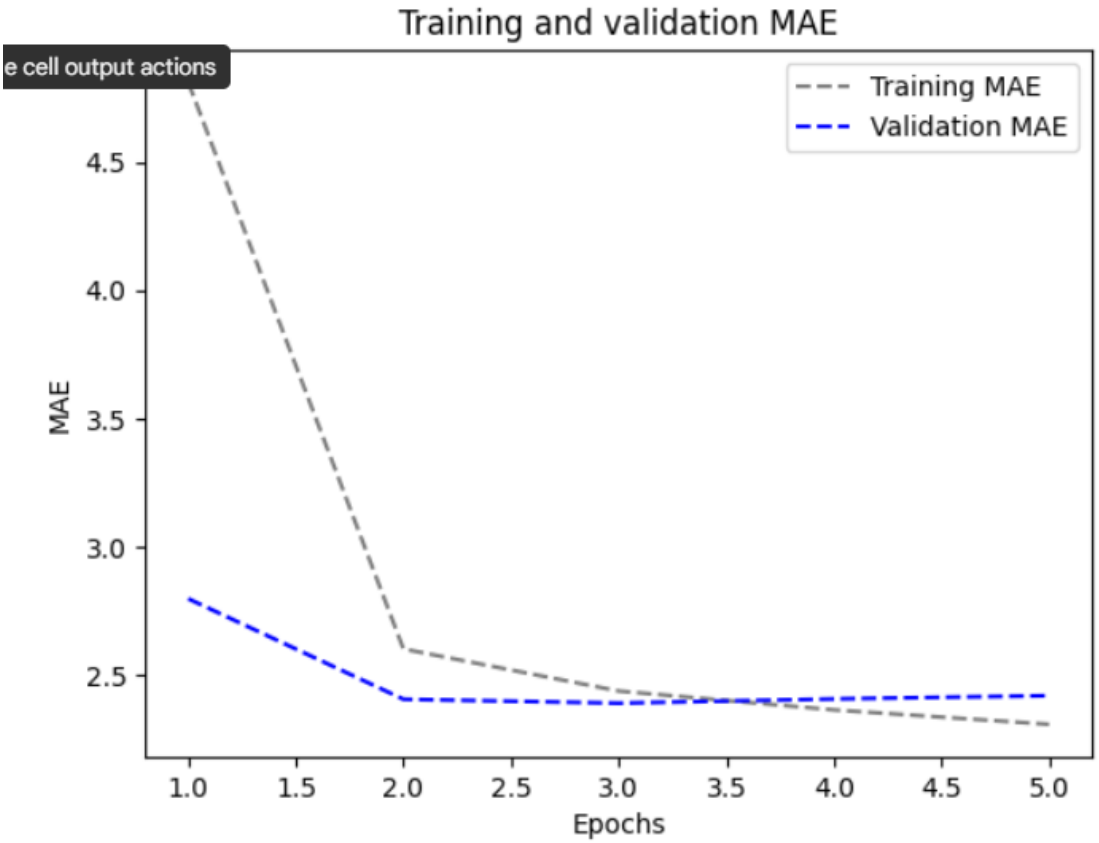
The model that achieved a Test Mean Absolute Error (MAE) of 2.51 was the most effective, proving to be less computationally intensive than Long Short-Term Memory (LSTM) models while successfully capturing long-range dependencies in sequential data, thus outperforming the other models.



Long Short-Term Memory (LSTM):

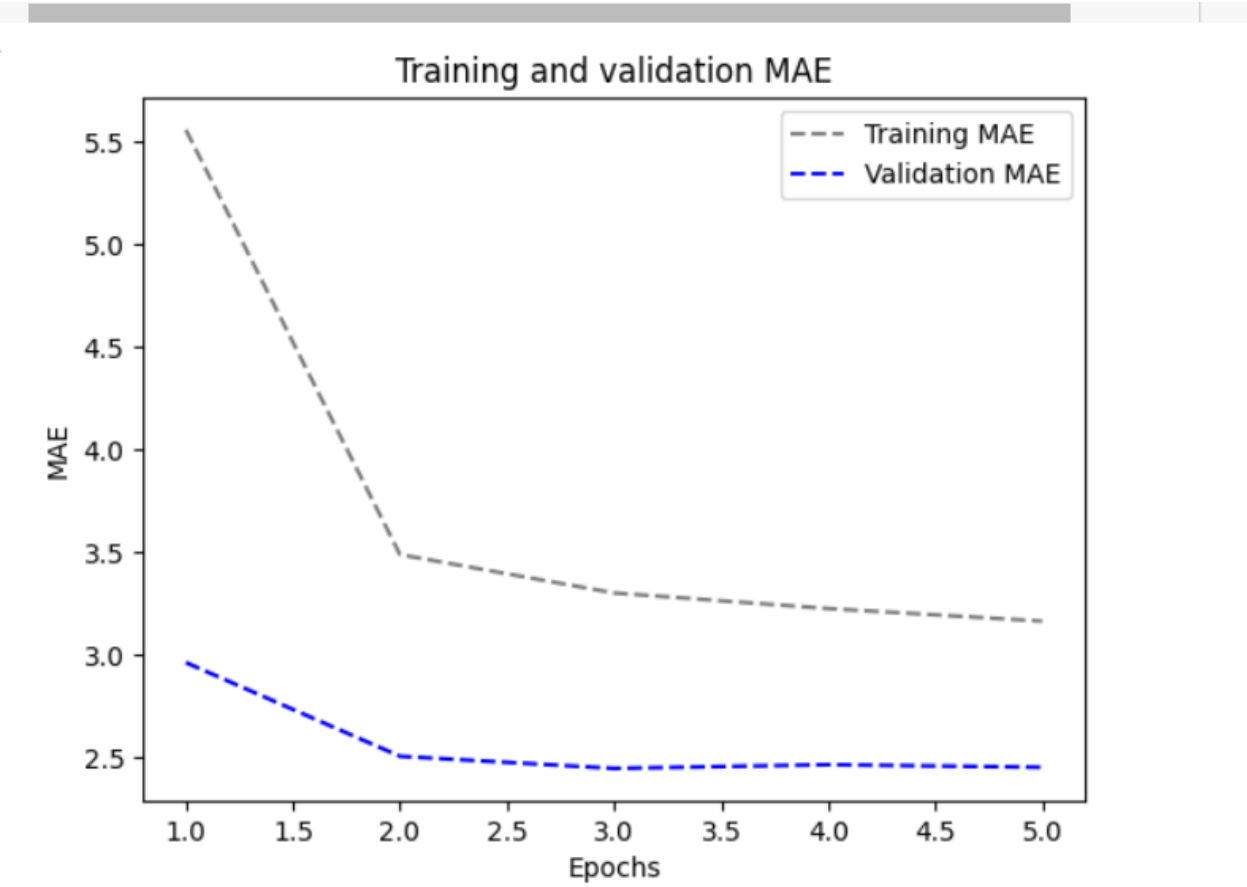
Recurrent neural networks offer architectures that are particularly designed for this application, with the Long Short-Term Memory (LSTM) layer being the most utilized. We will soon assess the performance of these models, beginning with the LSTM layer. The outcomes show

significant improvement, achieving a validation Mean Absolute Error (MAE) of just 2.41 degrees and a test MAE of 2.57 degrees. Overall, the LSTM-based model illustrates the capability of machine learning in this context by exceeding the common-sense baseline.



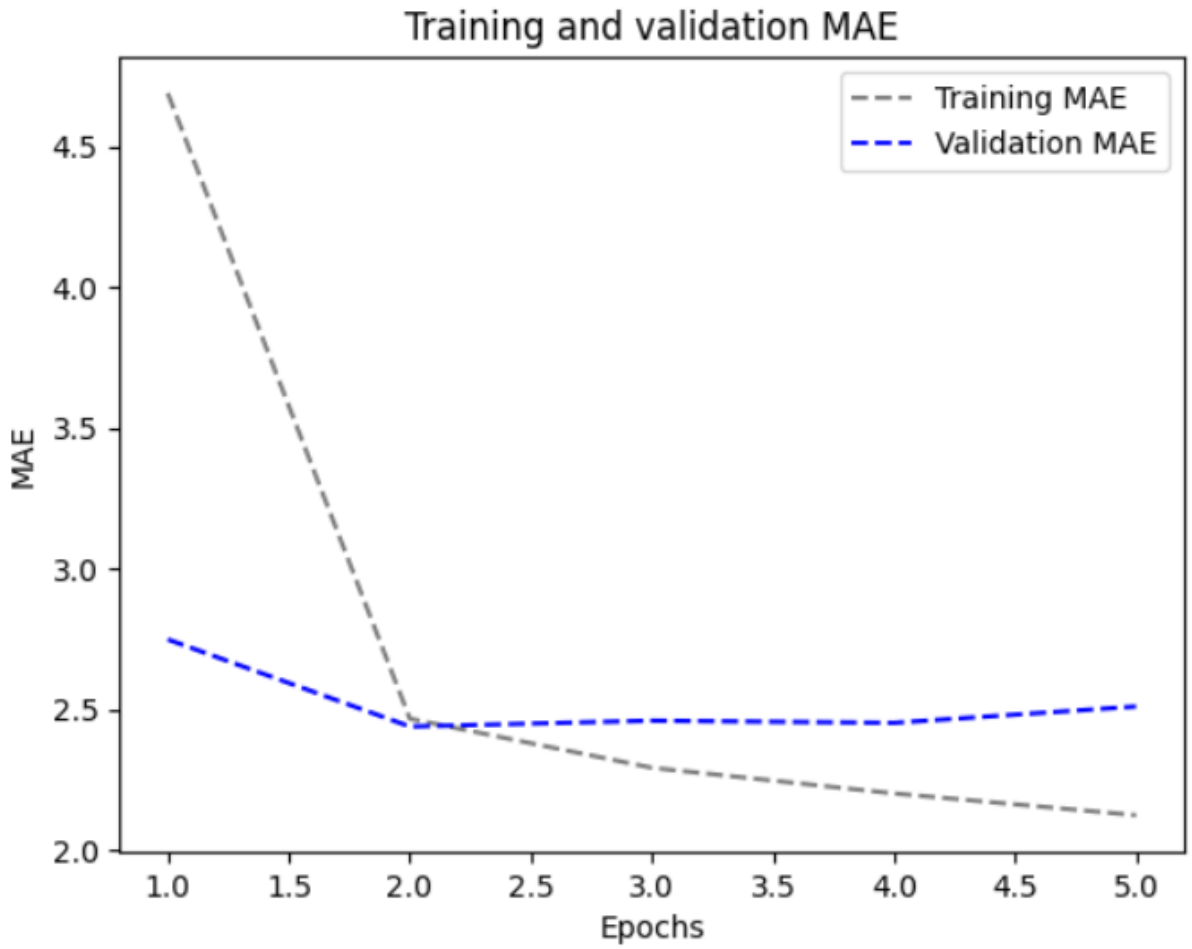
LSTM – Dropout Regularization:

After five epochs, overfitting was not a concern anymore. We reached a test Mean Absolute Error (MAE) of 2.6 degrees and a validation MAE as low as 2.449 degrees, which is quite encouraging. I created six distinct LSTM models by adjusting the number of units in the stacked recurrent layers to 8, 16, and 32.



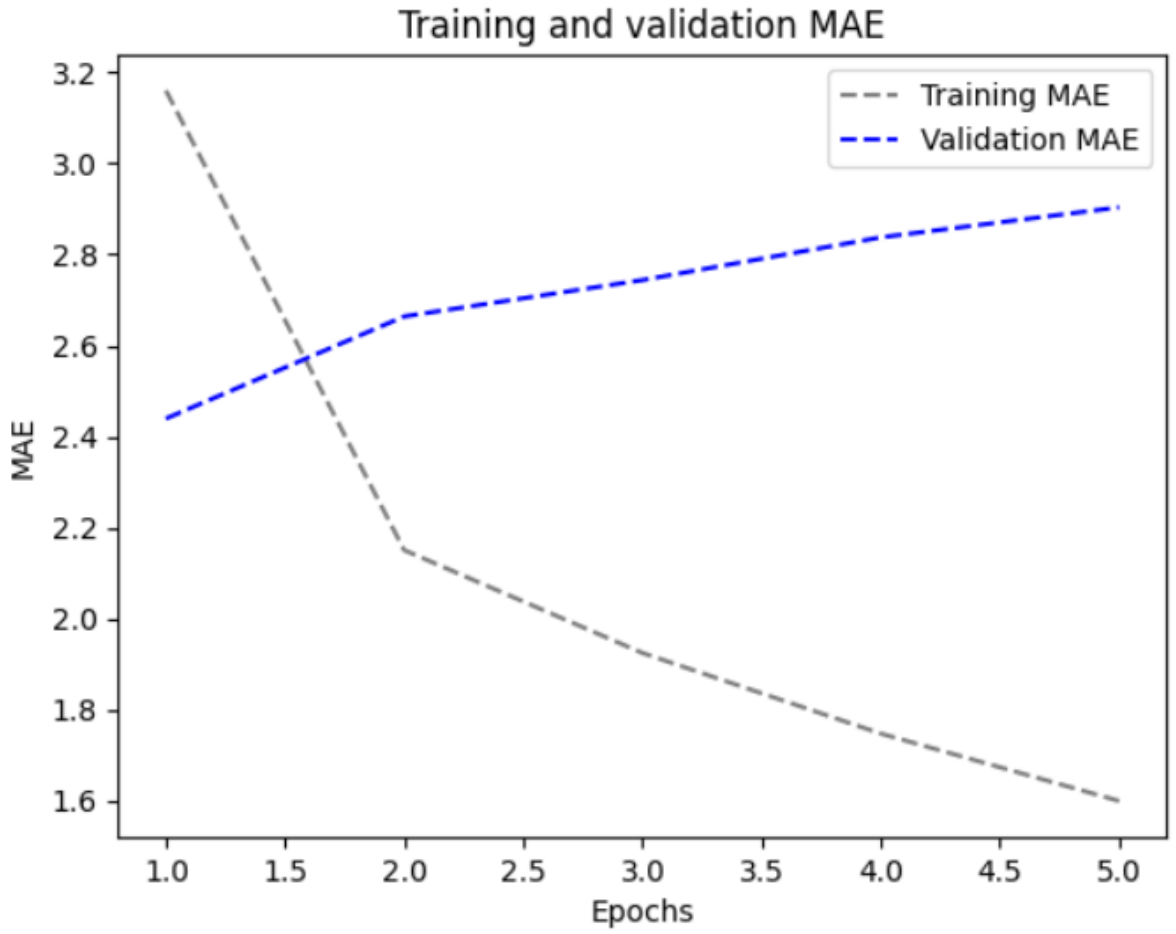
LSTM - Stacked setup with 16 units:

The LSTM stacked setup with 16 units underwent training for five epochs, showing a progressive improvement in performance. By the final epoch, the training loss was reduced to 7.41, with a corresponding training Mean Absolute Error (MAE) of 2.12. The validation loss and validation MAE were recorded at 10.39 and 2.51, respectively, indicating a slight increase in validation error compared to previous epochs, suggesting potential overfitting. Upon evaluating the model on the test dataset, the results indicated a test loss of 11.12 and a test MAE of 2.61. This demonstrates that while the model has learned effectively from the training data, it exhibits some overfitting, as evidenced by the difference in performance between the training and test sets, yet it maintains a satisfactory level of accuracy for time-series forecasting tasks.

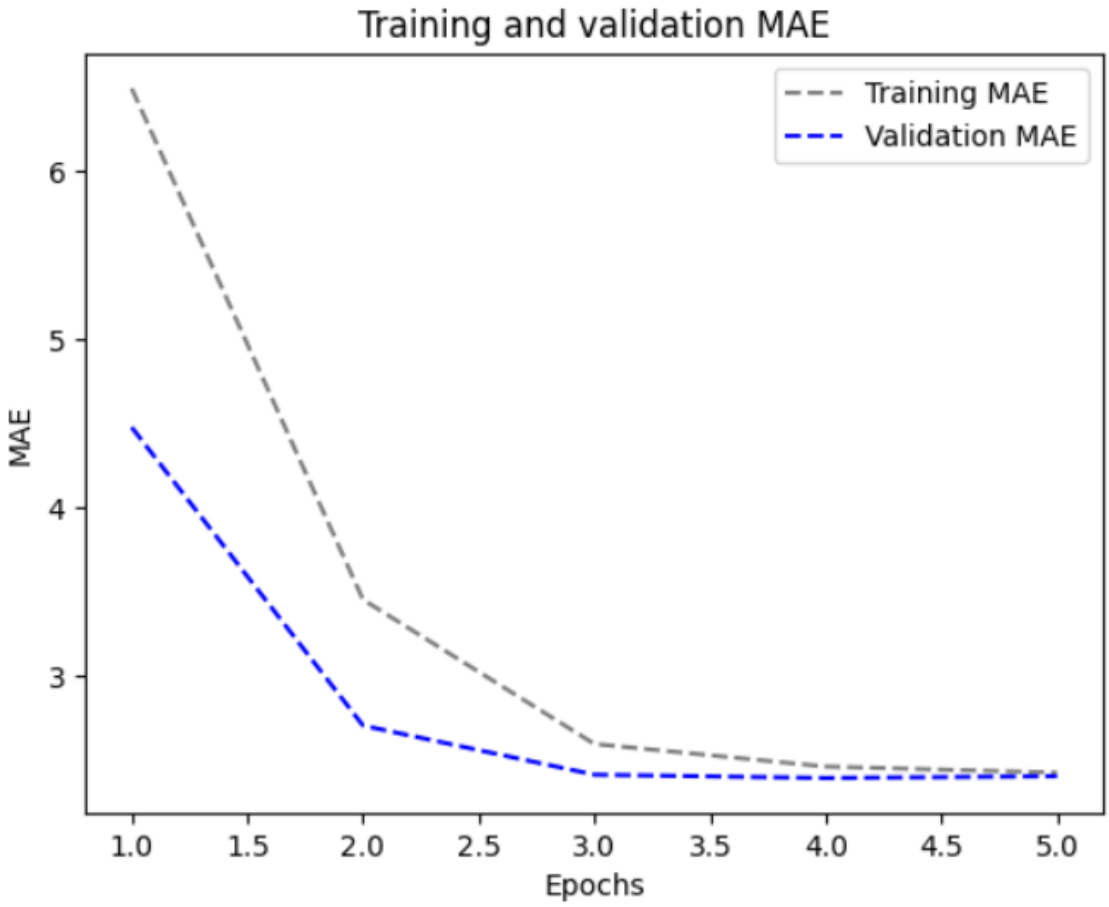


LSTM - Stacked setup with 32 units:

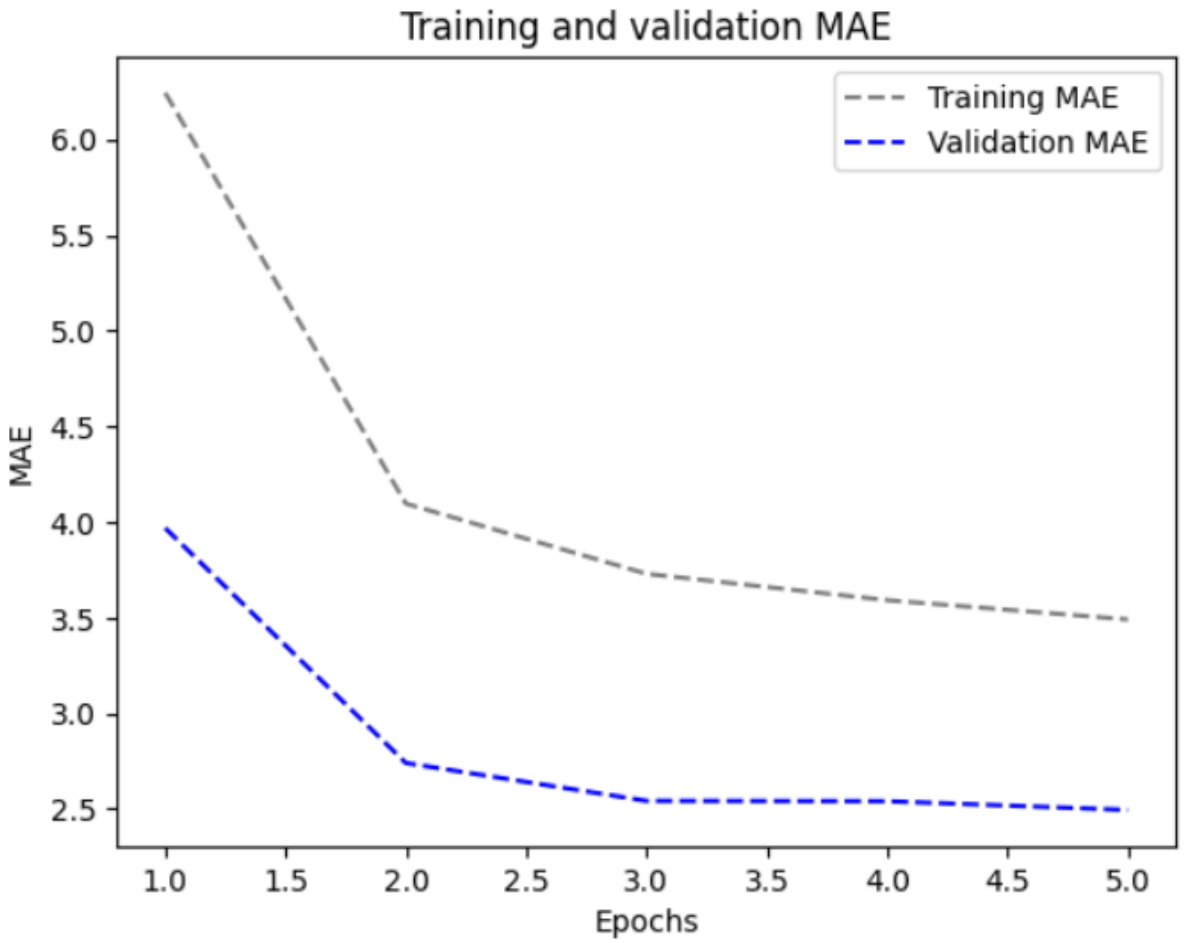
In the LSTM stacked setup with 8 units, the model was trained over five epochs, showing a trend of decreasing loss but an increasing validation loss, indicating possible overfitting. By the end of the training, the training loss was reduced to 4.25, with a training Mean Absolute Error (MAE) of 1.60. The validation loss increased to 13.68, and the corresponding validation MAE was 2.90, reflecting a deterioration in the model's performance on the validation set as training progressed. When the model was evaluated on the test dataset, it yielded a test loss of 11.39 and a test MAE of 2.64, indicating that while the model has effectively learned from the training data, its performance on unseen data is slightly less satisfactory, suggesting the need for adjustments to improve generalization.



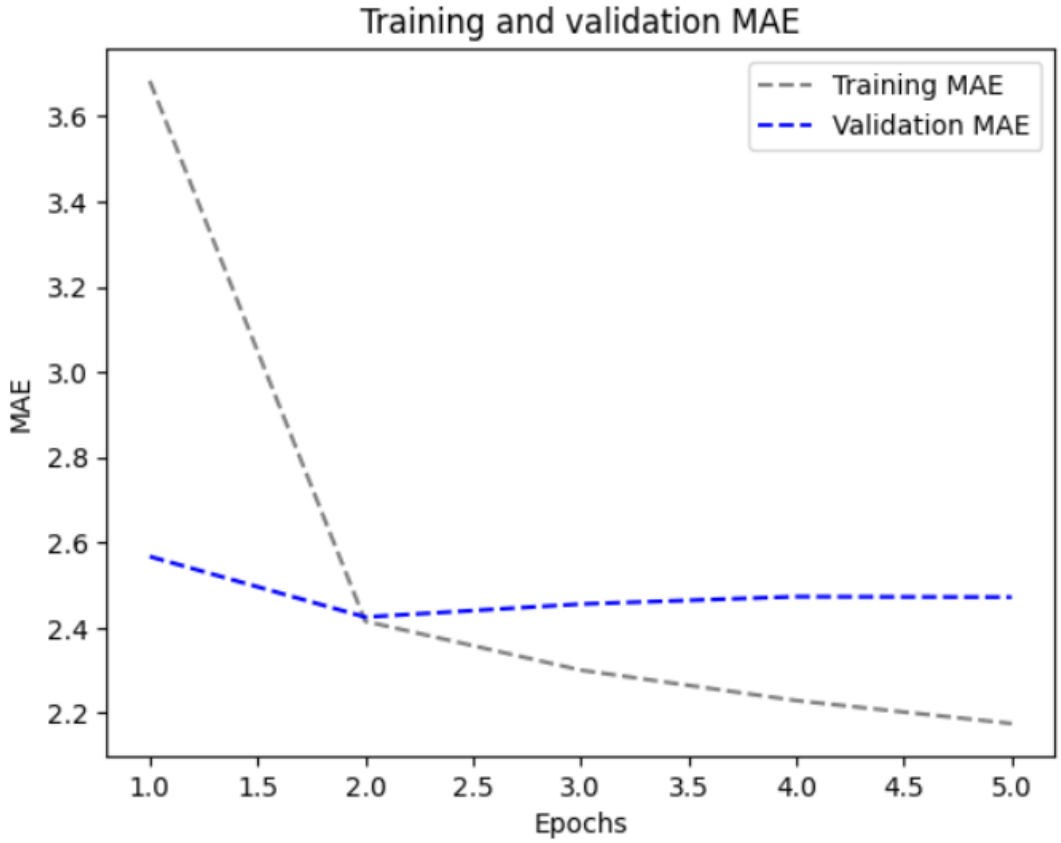
5.LSTM - Stacked setup with 8 units



6.LSTM - dropout-regularized, stacked model

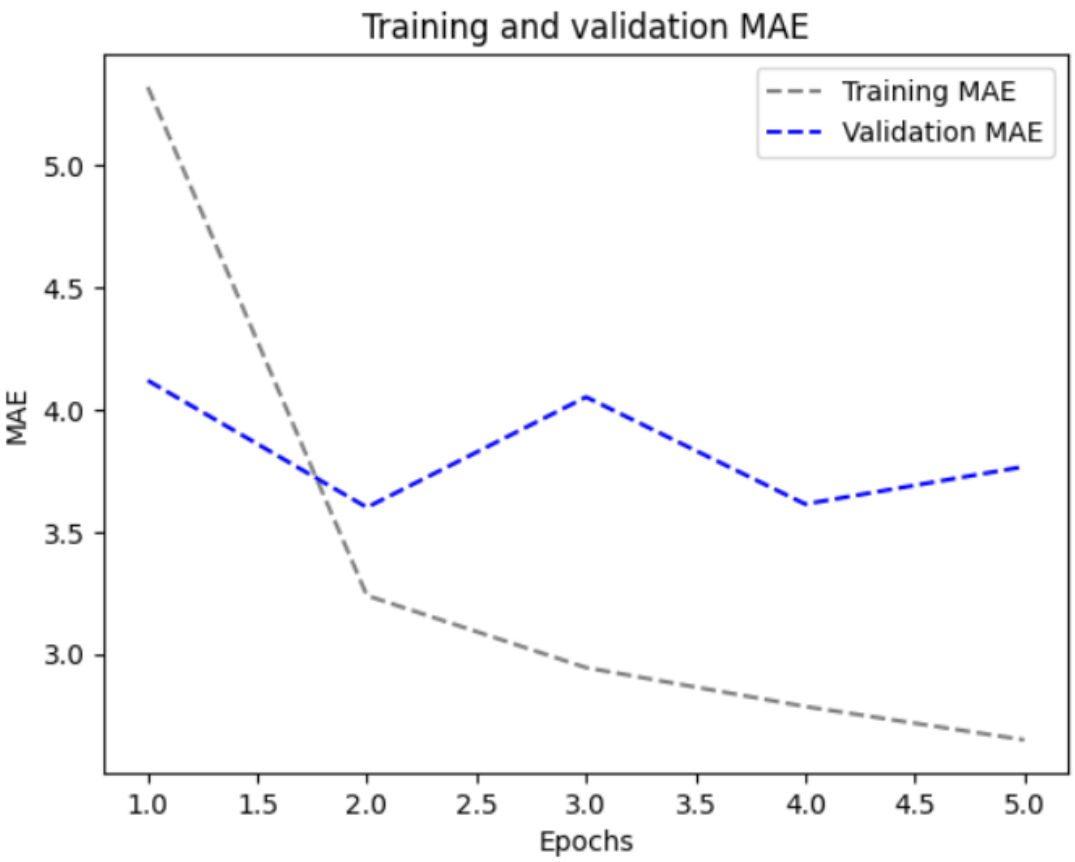


Bidirectional LSTM:

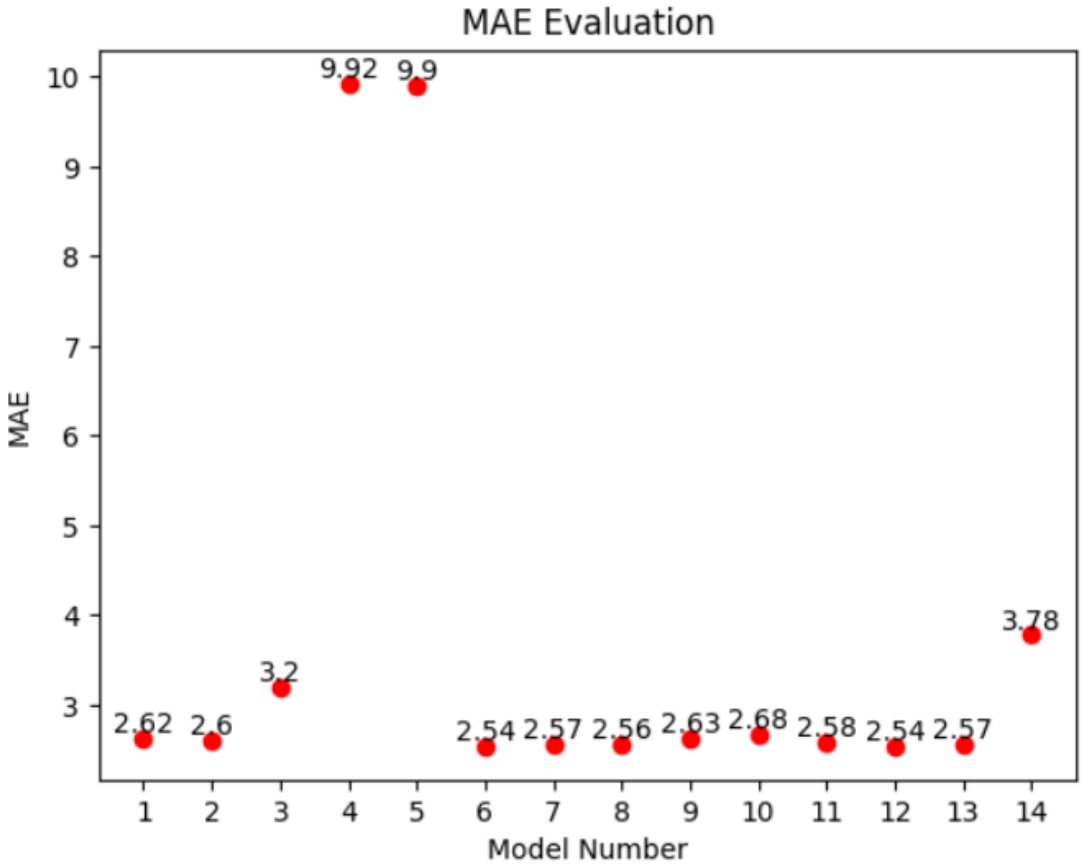


1D Convnets and LSTM together:

A Mean Absolute Error (MAE) of 3.85 was not a satisfactory outcome for the model I created, which integrated RNNs with 1D convolution. One potential explanation for this subpar performance is that the convolutional layer is disrupting the sequential order of the information.



Performance of Every Model:



Model	Validation MAE	Test MAE
Dense Model	2.6525	2.69
1D Convolutional Model	2.988	3.15
Simple RNN	9.85	9.93
Stacked Simple RNN	9.83	9.92
GRU	2.36	2.52
LSTM Simple	2.41	2.57
LSTM – Dropout Regularization	2.44	2.6
LSTM – Stacked 16 Units	2.5	2.6
LSTM – Stacked 32 Units	2.9	2.64
LSTM Stacked 8 Units	2.4	2.55
LSTM – Dropout regularization, stacked model	2.49	2

Bidirectional LSTM	2.47	2.5
1D Convolutional LSTM	3.76	3.85

Key Findings Summary:

Best Performing Models:

GRU: This model achieved the lowest validation MAE of 2.36 and a test MAE of 2.52, indicating its effectiveness in capturing patterns within the time-series data.

Bidirectional LSTM: Also performed strongly, with a validation MAE of 2.47 and a test MAE of 2.5, proving it to be effective in processing sequential data from both directions.

Good Performance:

LSTM Models: The simple LSTM yielded a validation MAE of 2.41 and a test MAE of 2.57, while the stacked LSTM with 8 units also showed good results with a validation MAE of 2.4 and a test MAE of 2.55. Adding dropout regularization to LSTM further improved results to a validation MAE of 2.44 and test MAE of 2.6.

Dense Model: Performed reasonably well with a validation MAE of 2.6525 and a test MAE of 2.69, though slightly less effective than advanced RNNs.

Moderate Performance:

Stacked LSTM Models: The LSTM with 16 units had a validation MAE of 2.5 and test MAE of 2.6, while the one with 32 units showed slightly higher MAE values, indicating diminishing returns with more units in certain cases.

Poor Performance:

1D Convolutional Model: Displayed a validation MAE of 2.988 and a test MAE of 3.15, suggesting that convolution alone does not capture the sequential patterns effectively.

1D Convolutional and LSTM Combination: With a validation MAE of 3.76 and test MAE of 3.85, this model struggled to maintain the sequence order, leading to suboptimal performance.

Simple and Stacked RNNs: Both showed very high MAE scores around 9.9, indicating significant underperformance due to difficulties in retaining temporal patterns.

Overall, these findings indicate that advanced RNN architectures, such as GRU and Bidirectional LSTM, are well-suited for time-series forecasting. Incorporating dropout regularization and optimizing hyperparameters can further enhance model performance. Convolutional models and simple RNNs are less effective for this data type, reinforcing the suitability of advanced RNNs, particularly GRU, for weather prediction tasks.