AML ASSIGNMENT-3
TIME-SERIES DATA

Submitted by: Nikitha Begari & Kavyasree Bommakanti
Group:4

Summary:

This assignment uses the Jena weather dataset, which includes 420,451 samples and 15 weather-related parameters, to apply several deep-learning models to temperature prediction. The main aim is to compare different neural network topologies for forecasting time series data.

We built 14 different models to evaluate time series data. The first model generated a baseline and a Mean Absolute Error (MAE) of 2.44 using common sense methods. Subsequently, we created a simple machine learning model using a dense layer, yielding a slightly higher MAE of 2.67. The dense layer model performed poorly because of the time series data being flattened, which removed the temporal context.

A convolutional model was also used, however it yielded poor results because it treated each data segment equally, even after pooling, which interfered with the data's sequential order.

It is for this reason that we came to the conclusion that Recurrent Neural Networks (RNNs) work better with time series data. An essential feature of recurrent neural networks (RNNs) is their capacity to incorporate information from earlier stages into the decision-making process at hand. This can then be used by the network to identify dependencies and patterns in the sequential data. The RNN's internal state, which serves as a sort of memory and stores information from prior inputs, allows it to represent sequences of varying lengths. However, the fundamental Simple RNN is frequently too simple to be extremely useful.

One significant drawback of Simple RNN is that it constantly performs the worst of all the models, as seen by the graphical representation. Theoretically, Simple RNN should be able to preserve information from all previous time steps, despite the well-known "vanishing gradient problem" making it difficult to utilize in deep networks. This problem effectively renders the network untrainable. In response to this challenge, more advanced RNN variants were developed and added to Keras as the Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). Out of all the models, the simplest GRU model yielded the best results from our testing, primarily because it can capture long-range dependencies in sequential data and is more computationally efficient than LSTMs.

In stacking recurrent layers, we tried six different LSTM models with varying units (8, 16, and 32), and the model with eight units performed the best. One well-known design for effectively managing time series data is LSTMs.

In order to address the forgetting issue and increase accuracy, we also experimented with bidirectional data presentation. Recurrent dropout was employed to avoid overfitting. All of these LSTM models showed similar MAE values, which were consistently lower than the common-sense model.

Lastly, we attempted to incorporate a 1D convolution model with an RNN. The higher mean absolute error (MAE) of 3.25 obtained by the hybrid model is explained by the convolution's limitations in maintaining the information order. According to my research, simple RNNs should be avoided when doing time series analysis because they can't reliably capture long-term associations and have problems with the vanishing gradient problem. Consider more advanced RNN designs like LSTM and GRU instead, which are designed to overcome these challenges.

As our trials show, LSTM is a popular choice for processing time series data, even if GRU might yield better results.

To enhance GRU models, one can adjust hyperparameters such as the quantity of units in stacked recurrent layers, recurrent dropout rates, and the use of bidirectional data presentation. Furthermore, it is recommended to focus on RNN designs intended for sequential data, as the coupling of RNN with 1D convolution did not yield the best results. Because convolutional approaches sometimes result in information being out of order, they are less suitable for processing time series data.

Introduction:

Recurrent neural networks (RNNs) are applied to time-series data in this report, particularly on weather forecasting issues. The assignment aims to investigate several approaches for enhancing RNN models' performance in weather pattern prediction. The strategies involve modifying the RNN model's architecture, experimenting with other recurrent layers (e.g., LSTM, GRU), and adding 1D convolutions alongside RNN layers. The study describes how these techniques were implemented, assesses how well they worked using validation datasets and shows which models performed the best when tested on the test set. In summary, the research attempts to demonstrate how well RNNs handle time-series data and to highlight methods for enhancing their predicting accuracy.

Problem Overview:

The issue of anticipating fluctuations in temperature over a 24-hour horizon based on hourly assessments of atmospheric variables is the focus of this study. The Max Planck Institute for Biogeochemistry in Jena, Germany is the source of our dataset, which includes a complete set

of meteorological measurements from 2009 to 2016. Recorded at 10-minute intervals, these measurements include temperature, humidity, wind direction, and atmospheric pressure—all of which are vital.
The main goal is to create a prediction model that can accurately estimate temperature by capturing all of the complex temporal correlations present in weather information. We want to investigate the effectiveness of Recurrent Neural Networks (RNNs) in this situation by making use of their capacity to pick up long-term dependencies and sequential patterns.

Methodology:

- To ensure consistency , we normalize the remaining features after preprocessing the dataset and extracting temperature values. The data is then divided into training, validation, and test sets while maintaining temporal continuity to appropriately reflect the forecasting task's forward-looking nature.
- We use a customized RNN architecture intended for time-series forecasting for modeling. The timeseries_dataset_from_array() tool in Keras is used to create datasets that are suitable for testing, validation, and training. The model is trained to forecast the temperature 24 hours in advance using a series of historical data.
- We used a weather time series dataset that was gathered from Jena, Germany's weather station. This dataset contains 14 different metrics that were recorded at 10-minute intervals throughout several years, such as temperature, pressure, humidity, wind direction, etc.
- The subset we will obtain is limited to the years 2009 to 2016, even though the entire dataset dates back to 2003.
- we used the first 50% of the data for training, the next 25% for validation, and the last 25% for testing when preparing our model.
- We built 14 models, listed below with results.

Data Preprocessing:

Data is normalized by subtracting the mean and dividing by the standard deviation of the training set. The dataset is divided into multiple sets for training, validation, and testing purposes.

RNN Application to the Time-series data:

➔ The findings show that the stacked SimpleRNN and simple RNN models perform poorly on the test set, with MAE values much greater than those of the other models. This implies that basic RNNs might not be appropriate for this specific time-series forecasting purpose.

- ➔ In contrast, GRU and LSTM models perform better. The Simple GRU and Bidirectional LSTM models are the most successful in capturing the temporal patterns in the data, as seen by their lowest test MAE values.
- ➔ LSTM models with diverse configurations (e.g., dropout regularization, stacked arrangement with varying units) perform reasonably, but not as well as top performers.

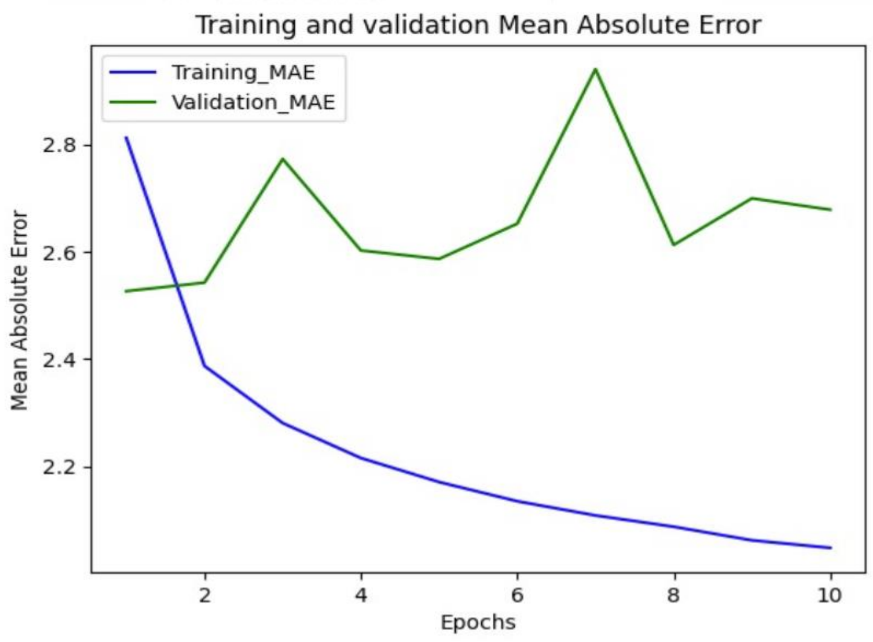
Results:

Model Number	**Name of the Model Description**	**Validation MAE**	**Test MAE**
1	Common-sense, non-machine-learning baseline	2.44	2.62
2	A basic machine-learning model(Dense layer)	2.67	2.64
3	1D convolutional model	3.10	3.25
4	Simple RNN layer that can process sequences of any length	9.85	9.92
5	Stacking RNN layers using simple RNN	9.83	9.91
6	GRU (Gated Recurrent Unit)-Simple Method	2.38	2.47
7	LSTM-Simple	2.48	2.58
8	LSTM(dropout Regularization)	2.29	2.51
9	LSTM Stacked(16 units)	2.69	2.65
10	LSTM Stacked(32 units)	2.99	2.62
11	LSTM Stacked(8 units)	2.40	2.62
12	LSTM(dropout-regularized and stacked)	2.36	2.57
13	LSTM(Bidirectional)	2.60	2.55
14	LSTM and 1D Convnets Combination	3.85	3.82

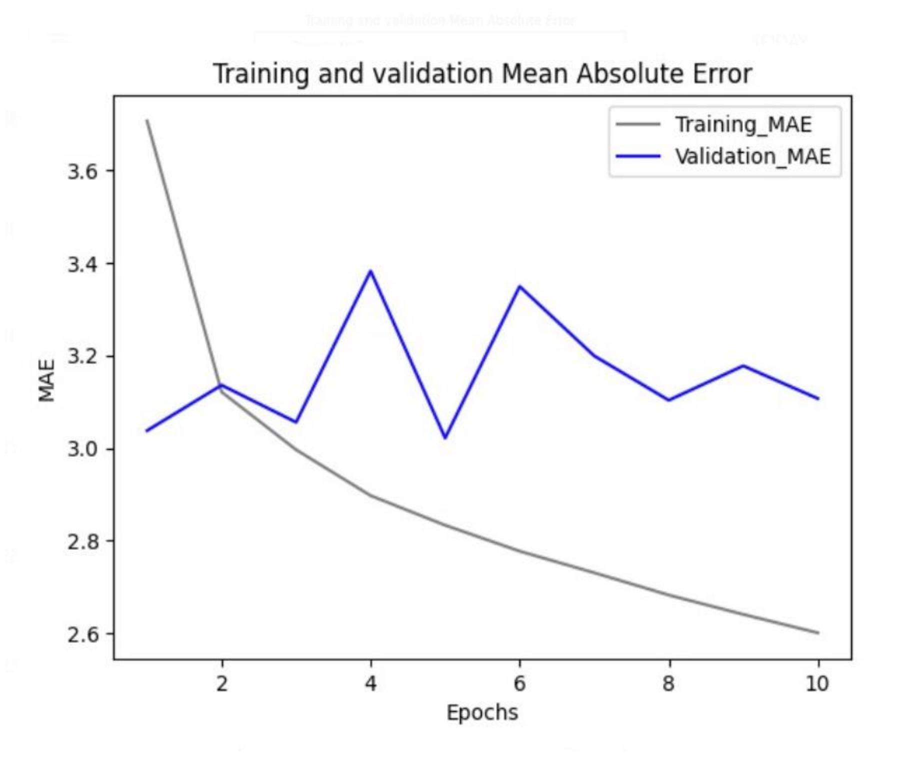
From above table, we observe that the validation MAE of the other LSTM models and the basic GRU model is lower. Because the ordered information is hampered by the max pooling and global average pooling layers, the 1D convolution model performs worse than this. When compared to the 1-D convolution model, a straightforward LSTM model produced superior outcomes. The LSTM models are among the finest, as far as we are aware, for handling the vanishing gradient descent problem.

In order to reduce the overfitting, we employed dropout regularization. The model's capacity to represent complicated information is improved by the use of stacked recurrent layers, although this has the disadvantage of increasing computing requirements.
In the stacked configuration, we also employed 8, 16, and 32 units in each recurrent layer. We obtained a very good validation MAE 2.4 low value.

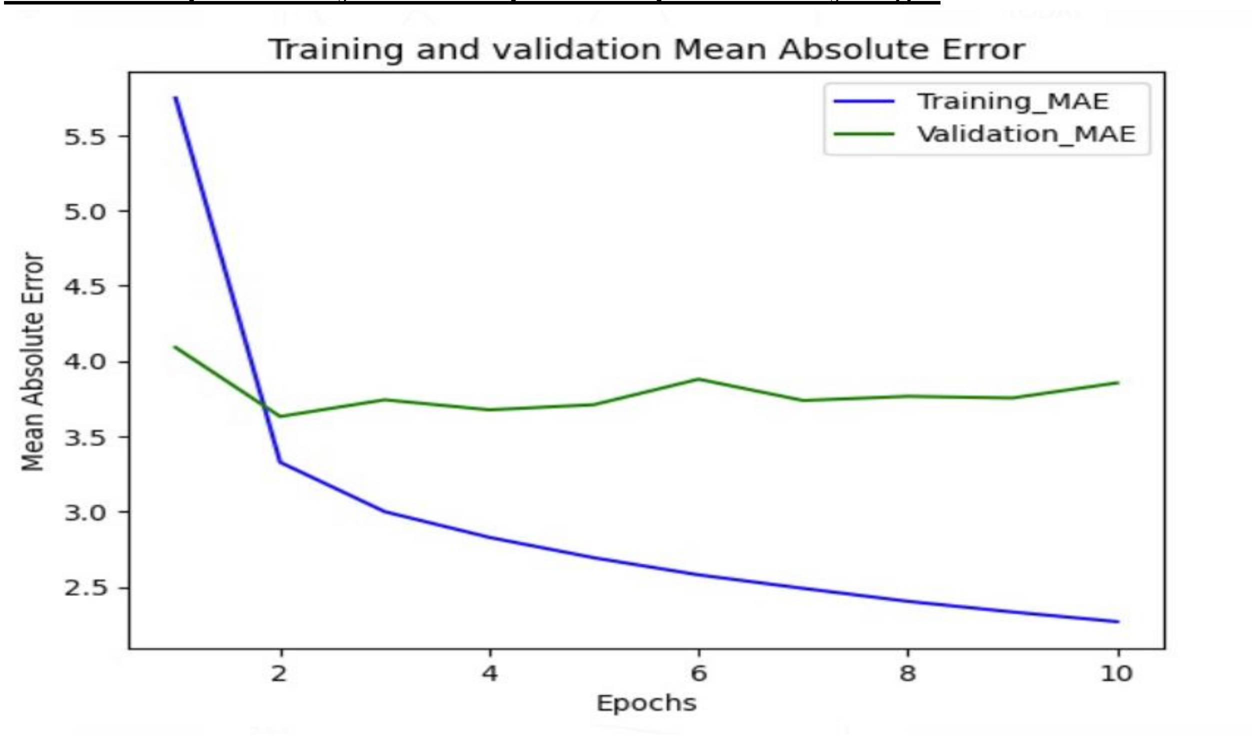
Model 2: A basic machine-learning model(Dense Layer)



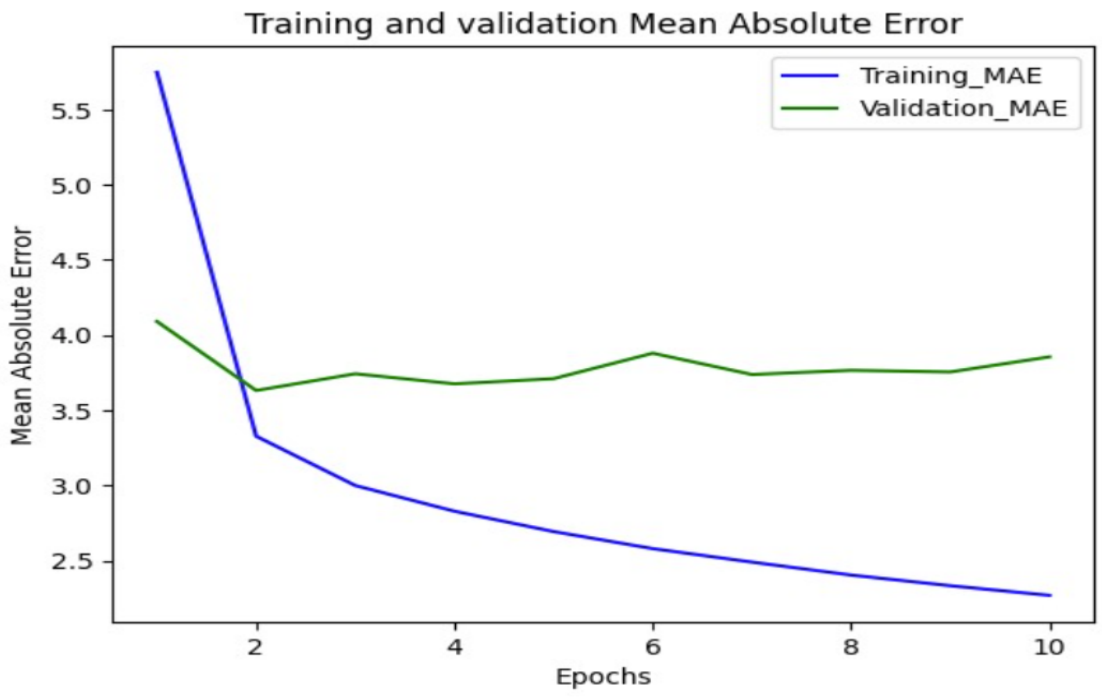
Model 3: 1D convolutional model



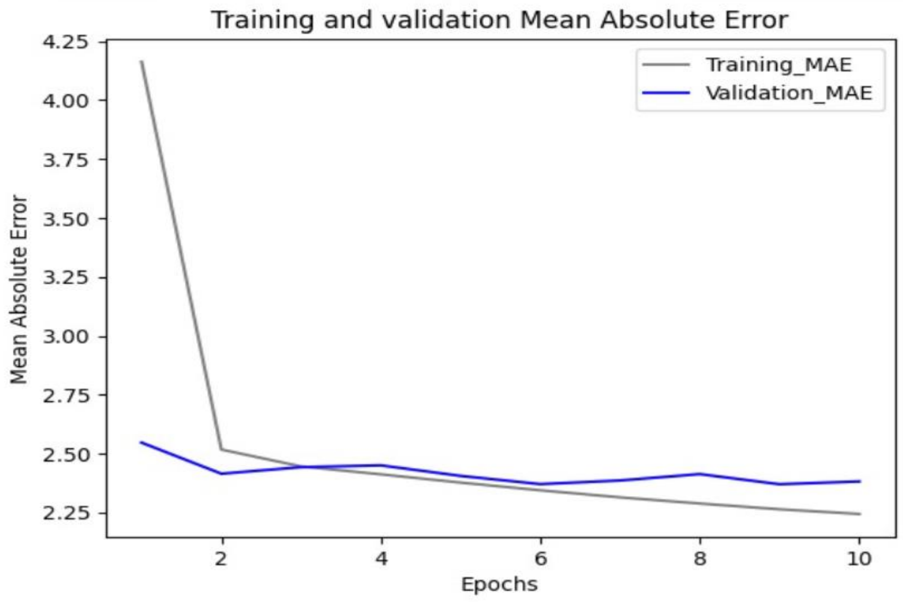
Model 4: Simple RNN layer that can process sequences of any length



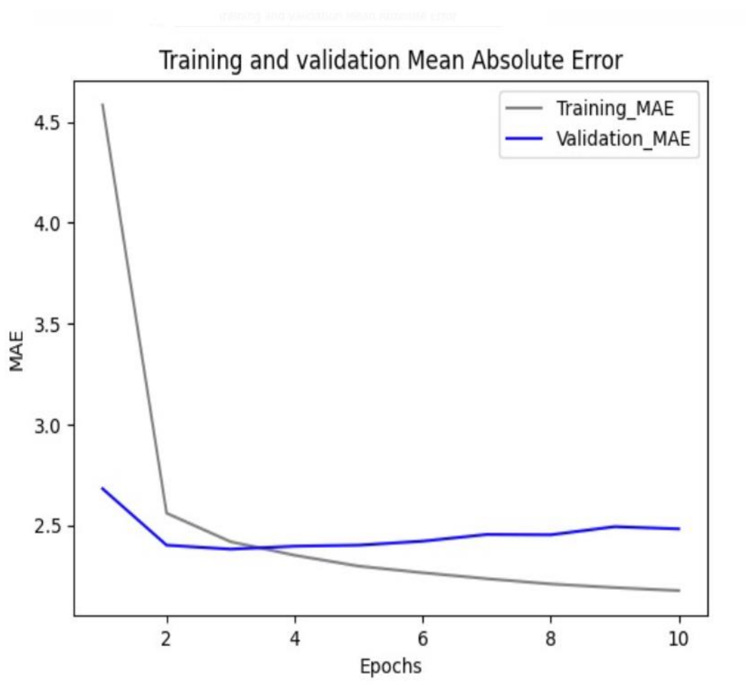
Model 5: Stacking RNN layers using simple RNN



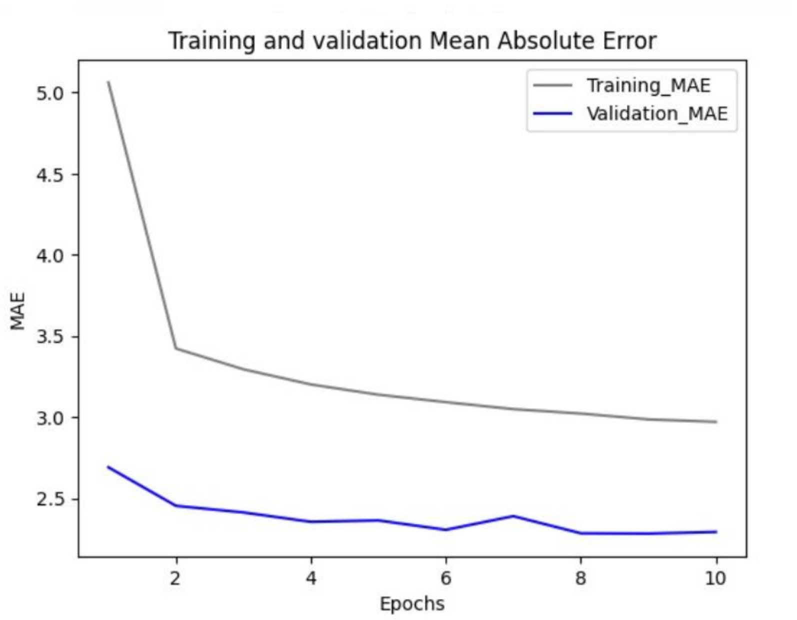
Model 6:GRU (Gated Recurrent Unit)-Simple Method



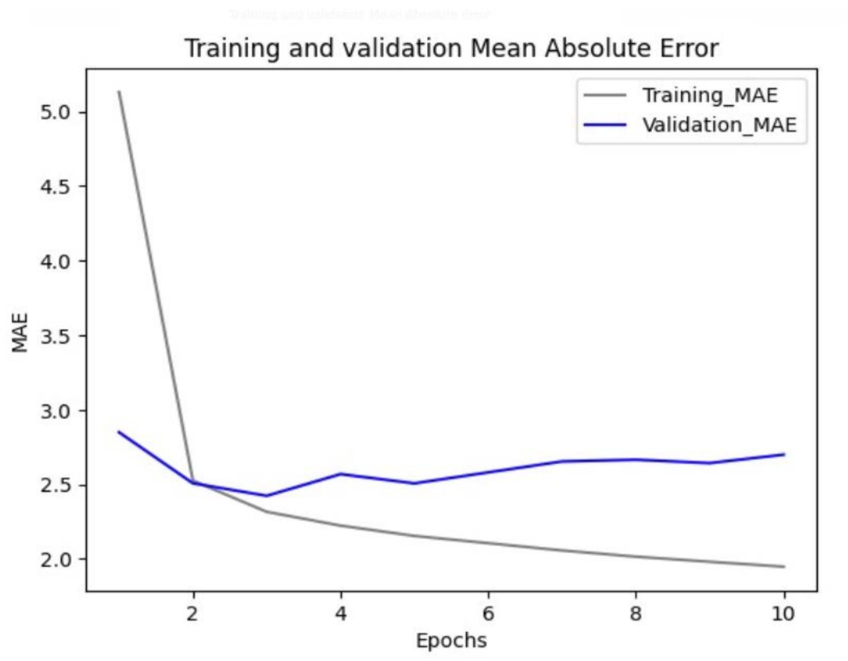
Model 7: LSTM-Simple



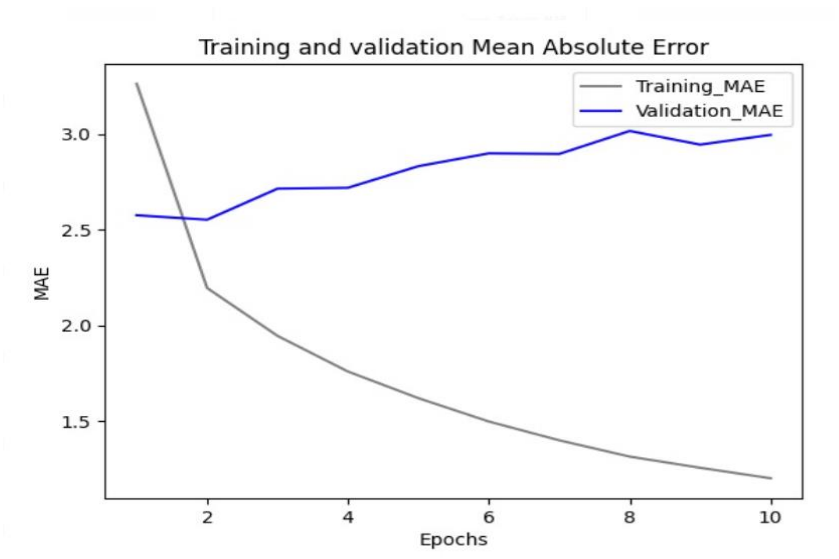
Model 8: LSTM (dropout Regularization)



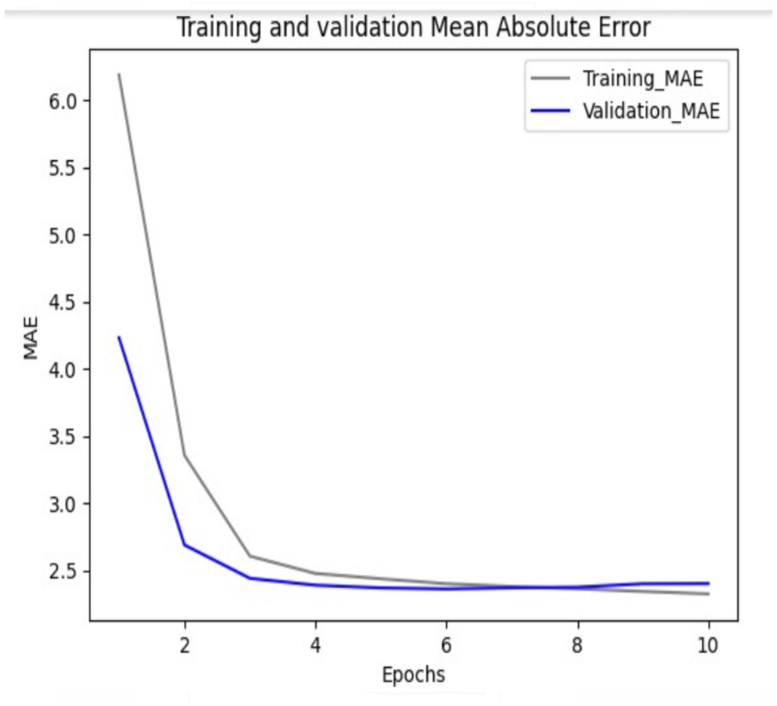
Model 9: LSTM Stacked(16 units)



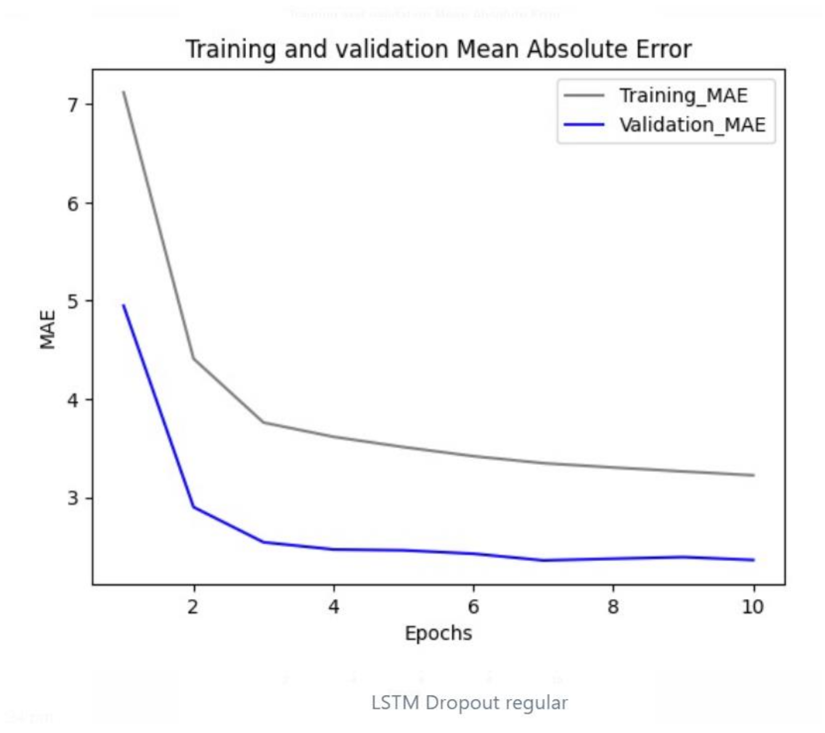
Model 10: LSTM Stacked(32 units)



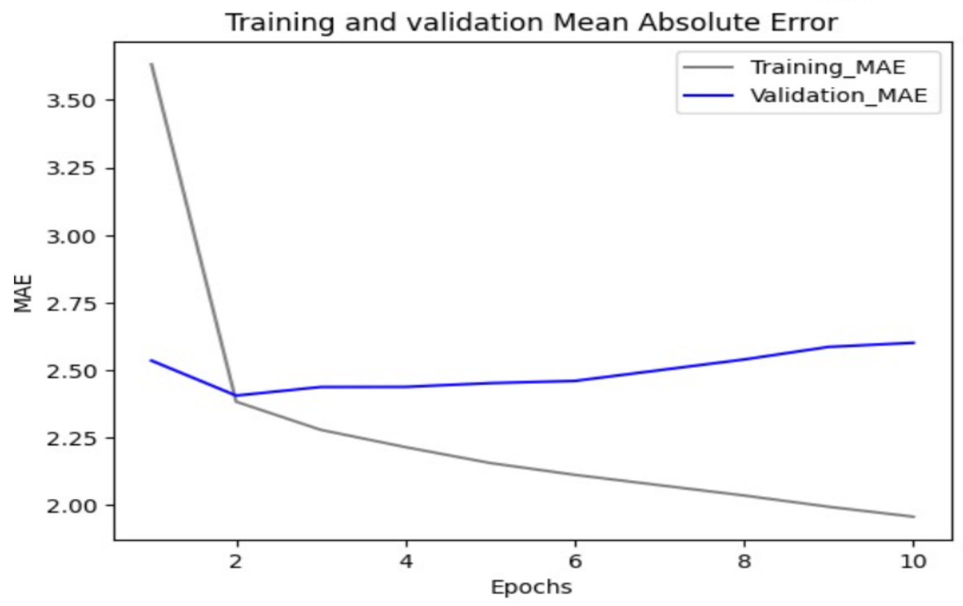
Model 11: LSTM Stacked (8 units)



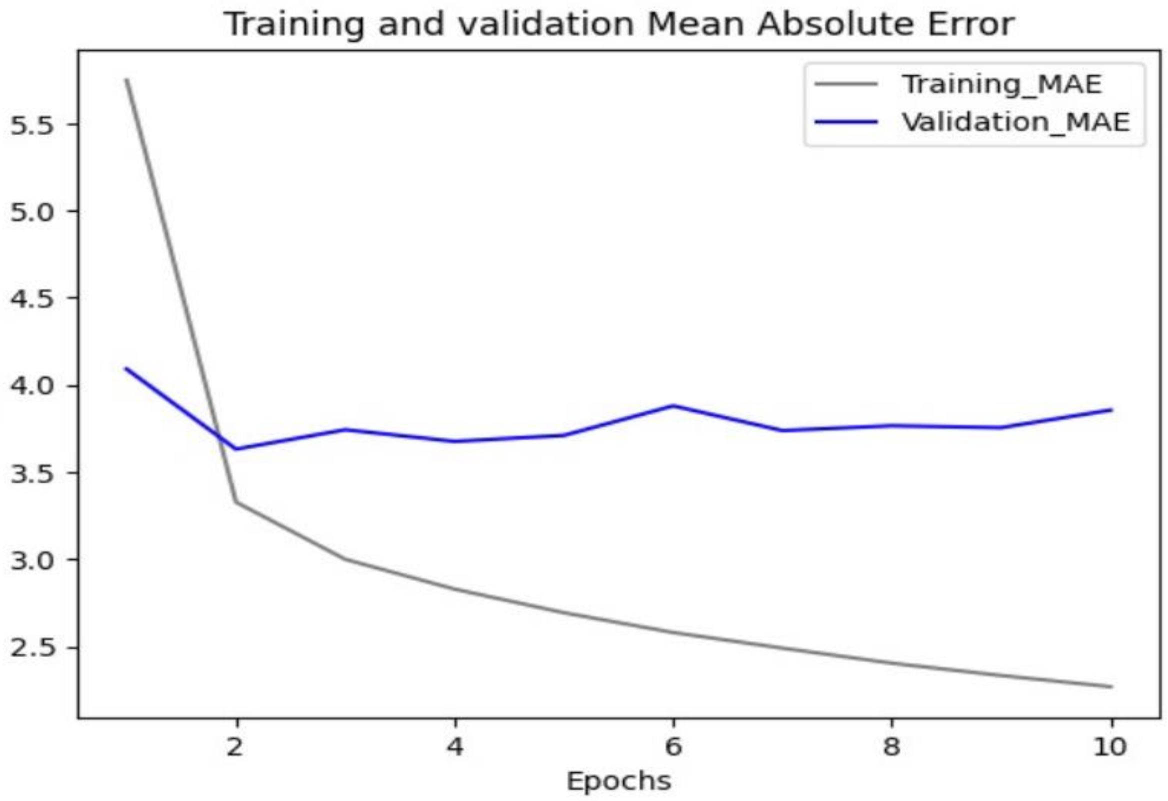
Model 12: LSTM (dropout-regularized and stacked)



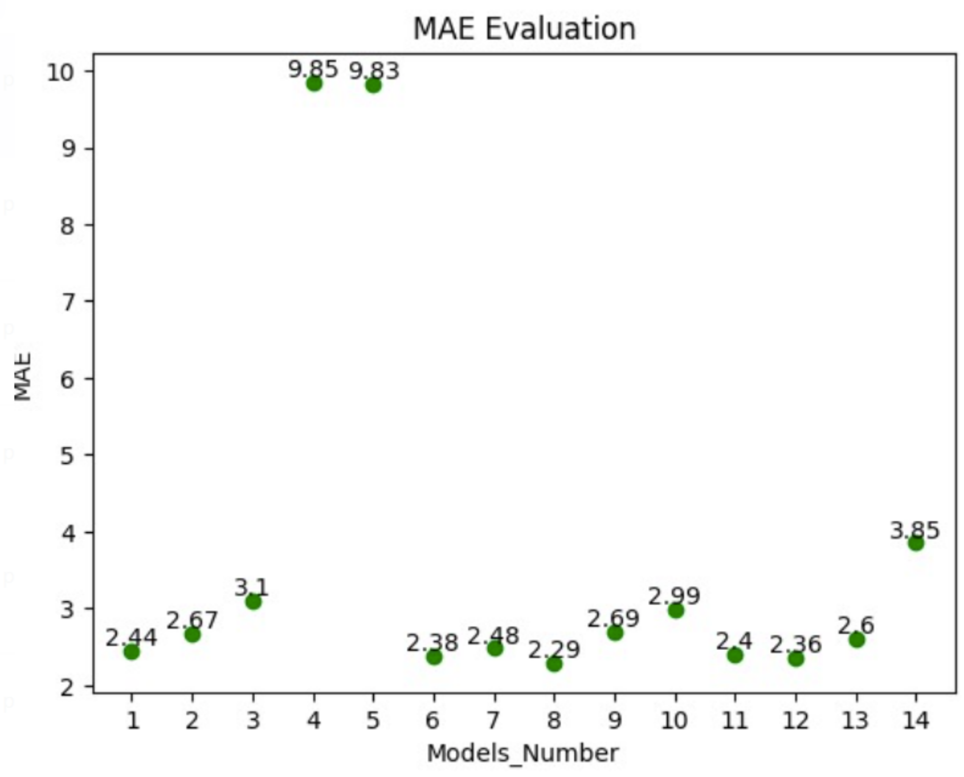
Model 13: LSTM (Bidirectional)



Model 14: LSTM and 1D Convnet combined



MAE Evaluation:



Enhancing Network Performance for Time-Series Data:

➔Some LSTM models use dropout regularization to prevent overfitting. On the other hand, dropout regularization does not always translate into better performance for LSTM models.
➔Better performance is not always achieved by stacking more LSTM units in the configuration. For example, the LSTM stacked arrangement with 32 units has a slightly higher test MAE than the setup with 16 units.
➔Bidirectional LSTM models outperform unidirectional LSTM models in terms of capturing information from past and future time steps.

Recommendations:

➔Select models that perform well on the test set, such as Simple GRU and Bidirectional LSTM.

➔Try out various hyperparameters and architectures to determine which model performs the best for the particular dataset.
➔Consider adding or developing new features to enhance the model's capacity to detect meaningful patterns in data.
➔Evaluate models on validation and test sets to ensure performance improvements are applicable beyond training data.
➔Consider specialized deep learning techniques for time-series forecasting, such as attention mechanisms or hybrid models that combine classic statistical methods with deep learning.

Insight & Observations:

A validation Mean Absolute Error (MAE) of 2.44 degrees Celsius and a test MAE of 2.62 degrees Celsius are obtained from an initial evaluation utilizing a simple baseline technique, where the future temperature is expected to reflect the current temperature. This baseline, albeit useful as a basic guide, emphasizes the necessity for more advanced forecasting methods.

The training and validation MAE for the basic GRU model is displayed in the above graph. We aim to improve upon the performance of the baseline technique by implementing the RNN-based forecasting model. We expect improved temperature predictions by utilizing the temporal dynamics recorded by RNNs, which will advance the state-of-the-art in weather forecasting technology.

Conclusion:

The training and validation MAE for the basic GRU model is displayed in the above graph. We aim to improve upon the performance of the baseline technique by implementing the RNN-based forecasting model. We expect improved temperature predictions by utilizing the temporal dynamics recorded by RNNs, which will advance the state-of-the-art in weather forecasting technology.

In summary, the more complex RNN models, such as the bidirectional LSTM and the stacked LSTM with dropout perform better than the more basic ones, like the 1D convolutional network and the dense neural network. The bidirectional LSTM performs optimally, with a validation MAE of 2.60 and a test MAE of 2.55. This assignment highlights the efficiency of recurrent neural networks and their modifications for time series forecasting challenges.