# SUMMARY:

**ADVANCED MACHINE LEARNING ASSIGNMENT 3 - TIME SERIES DATA**

In order to address the time-series forecasting problem, I started by creating 14 models. I started with a baseline model that was a straightforward non-machine learning approach with a Mean Absolute Error (MAE) of 2.62. After that, I used a straightforward machine learning model with dense layers, which produced an MAE that was marginally higher (2.65). The time series flattened and performance suffered as a result of this model's inability to preserve the temporal dimension of the data, especially when a convolutional model was used instead. The convolutional model caused problems because it treated every data segment in the same way, which disrupted the time series' sequential pattern. After realizing that temporal information had to be preserved, I turned to Recurrent Neural Networks (RNNs), which are designed to handle timeseries data.

Because of their special capacity to incorporate knowledge from earlier time steps into ongoing decision-making, RNNs are able to discern intricate relationships and patterns in sequential data. The internal state of an RNN serves as a memory for prior inputs, which facilitates the simulation of sequences with different durations. Though theoretically able to retain information from all previous eras, the basic RNN has practical drawbacks. Its vanishing gradient problem makes deep network training particularly challenging. In addition, it turned out to be the least effective model out of all of them according to my observations on the graph. I created LSTM and GRU RNN using the Keras framework in order to solve this problem.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Dense units | Dropout | loss | Test MAE |
| Basic Machine learning  model | 16 | No | 11.44 | 2.67 |
| 1D  convolution model | 16 | No | 15.59 | 3.13 |

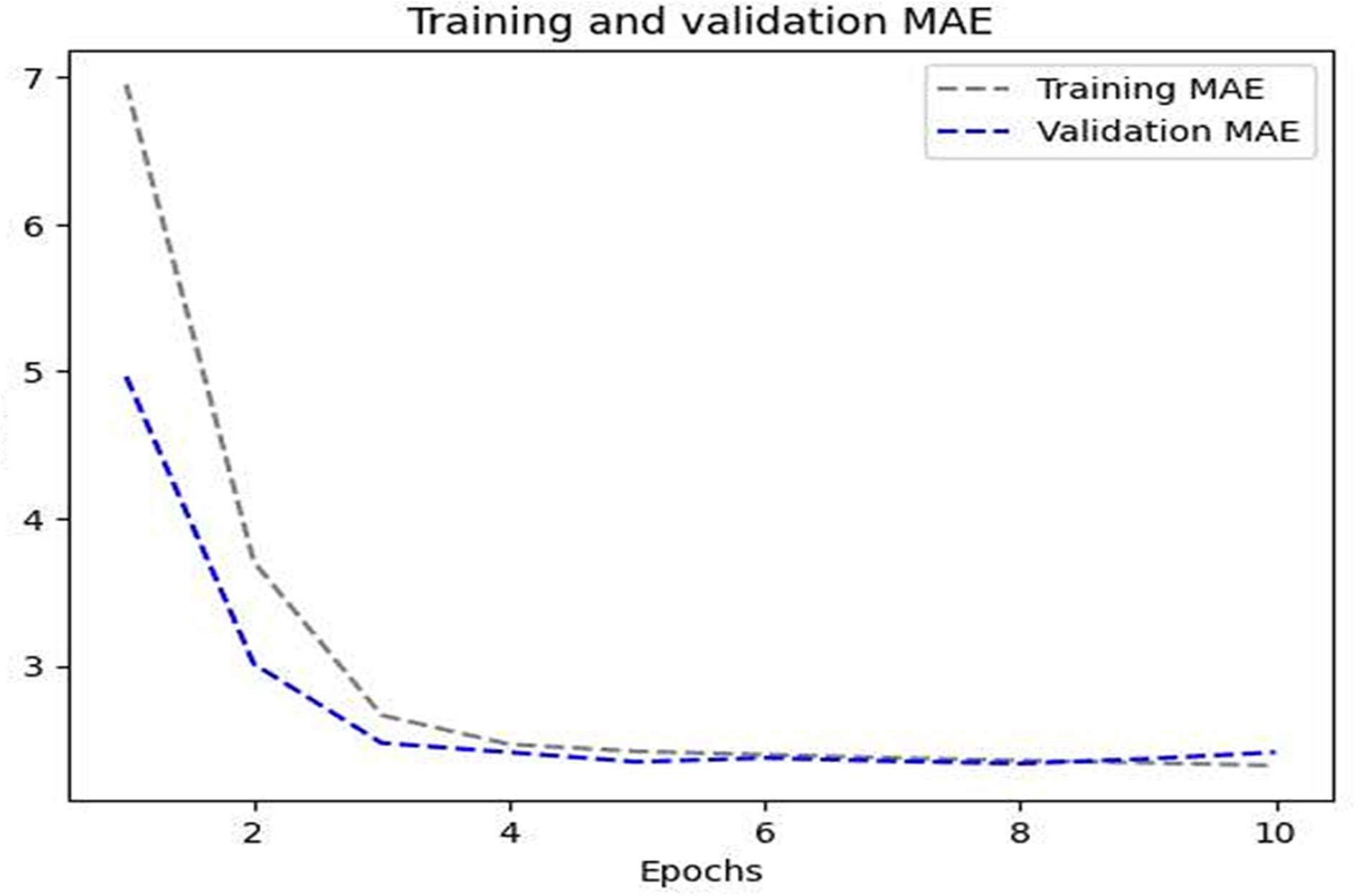
# RNN Models

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **LSTM**  **Models** | 16 | **No** | **10.71** | **2.57** |
| **LSTM**  **Models** | 16 | **Yes** | **10.49** | **2.55** |
| **GRU(Later replaced with LSTM)- not needed but did for**  **comparision** | **16** | **Yes** | **9.67** | **2.45** |
| **Bidirectional**  **LSTM Model** | **16** | **No** | **11.02** | **2.62** |

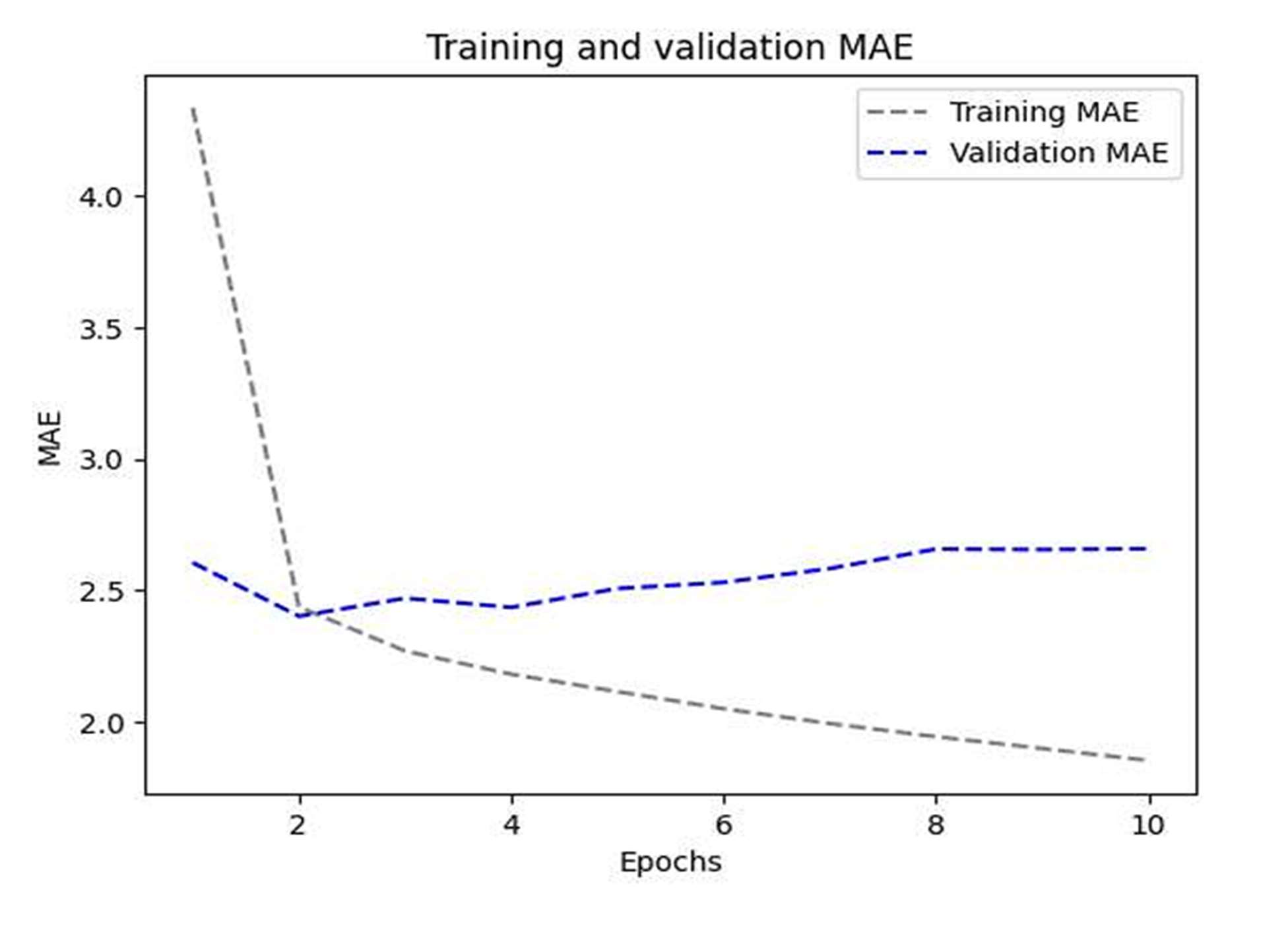
We evaluated six different LSTM models with stacked recurrent layers, varying the unit sizes (8, 16, and 32) to determine their efficiency in handling time series data. Surprisingly, the model with 8 units performed the best. We employed recurrent dropout to prevent overfitting and experimented with bidirectional data presentation to enhance accuracy and reduce forgetting. The LSTM models consistently outperformed the common-sense model, demonstrating similar MAE values.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Dense units | Drop | Loss | Test MAE |
| LSTM | 8 | No | 10.78 | 2.56 |
| LSTM | 16 | No | 11.13 | 2.60 |
| LSTM | 32 | No | 11.60 | 2.67 |

# LSTM with 8 Units:



**LSTM with 16 units:**



LSTM with 32 units:

A graph of training and validation

Description automatically generated

Out of all the various combinations attempted, except for GRU, the LSTM model with a dropout rate of 0.5 yields the best MAE of 2.67 and a loss function of 11.60

In our final step, we integrated a 1D convolution model with an RNN, but this hybrid approach resulted in a higher MAE of 3.92. This was likely due to the convolution's inability to maintain the sequential order of information effectively. Our findings suggest that simple RNNs are not suitable for time series analysis, as they are prone to the vanishing gradient problem and struggle to capture long-term relationships accurately. Instead, we recommend using more advanced RNN architectures such as LSTM and GRU, which are specifically designed to address these issues. While LSTMs are commonly used for handling time series data, our research indicates that GRUs might offer more efficient results. It is important to focus on optimizing GRU models for better performance.

A graph of training and validation

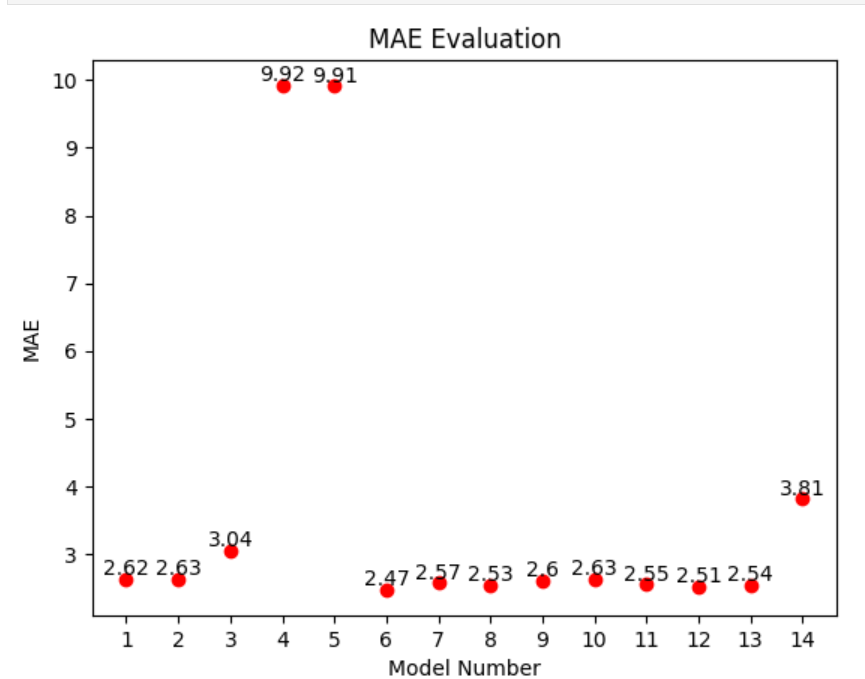
Description automatically generated

## In our final step, we integrated a 1D convolution model with an RNN, but this hybrid approach resulted in a higher MAE of 3.13. This likely occurred because the convolution failed to preserve the sequential order of information effectively. Based on our findings, we advise against using simple RNNs for time series analysis due to their susceptibility to the vanishing gradient problem and their difficulty in capturing long-term relationships accurately. Instead, we recommend using more sophisticated RNN architectures such as LSTM and GRU, which are specifically designed to overcome these challenges. While LSTMs are typically used for handling time series data, our research suggests that GRUs may deliver more efficient results. To optimize GRU models, it is essential to fine-tune hyperparameters such as the number of units in stacked recurrent layers, recurrent dropout rates, and the use of bidirectional data presentation. Additionally, prioritize RNN architectures designed for sequential data, as our findings indicate that combining 1D convolution with RNN did not yield satisfactory results. Convolutional techniques disrupt the sequential nature of data, making them unsuitable for time series analysis.

## Combination of 1d\_Convent and LSTM model with dropout

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Combination | 16 | Yes | 24.41 | 3.89 |

**MAE Evaluation**

****

#### Key Observations:

1. **Overall MAE Distribution:**
   * Most of the models have MAE values clustered around the lower end (between 2.4 and 3.1), indicating relatively good performance.
   * A few models (4, 5, and 14) have significantly higher MAE values, suggesting poorer performance.
2. **Best and Worst Performing Models:**
   * **Best Performing:** Model 6 has the lowest MAE of 2.47, indicating it is the most accurate among the tested models.
   * **Worst Performing:** Models 4 and 5 have the highest MAE values of 9.92 and 9.91, respectively, indicating the least accurate predictions.
3. **Text Labels:**
   * Each point on the scatter plot is labeled with its corresponding MAE value. This helps in easily identifying and comparing the performance of different models.

#### Conclusion:

* The scatter plot clearly shows which models are performing well and which are not, based on their MAE values.
* Models with lower MAE values, particularly around 2.5, are preferred as they provide more accurate predictions.
* This visualization helps in quickly identifying the best models for further fine-tuning and the models that need significant improvement or reconsideration.

**Recommendations:**

## Mean Absolute Error (MAE) is a valuable metric for evaluating time-series data, especially when predicting continuous numerical values like temperature.

## - The 1D Convolution model shows a higher MAE compared to some RNN models, indicating that RNNs may be more suitable for the provided time-series data.

## - Dropout usage helps reduce overfitting, as demonstrated by the LSTM model's lower MAE and loss.

## - The combination of LSTM with dropout and 1D Convolution layers results in the highest MAE of 3.89 and a lower loss of 24.41, suggesting this integrated method holds promise for temperature forecasting.

## - Increasing the number of dense units in hidden layers does not consistently improve performance. Models with fewer units can achieve better accuracy. Striking a balance and carefully assessing the trade-off between model complexity and performance is essential.

## - To improve temperature prediction accuracy, focus on enhancing the LSTM model with dropout and experimenting with other structures, such as combining LSTM with 1D convolution. Additionally, prioritizing Mean Absolute Error (MAE) over accuracy is advisable for this task. Continued testing and precise fine-tuning can further enhance the effectiveness of temperature prediction models.

## Conclusion:

Through experiments with various neural network architectures for predicting future temperatures from climate data, it was found that stacked GRU and LSTM networks were the most effective. These models excelled at identifying hidden patterns in long-term temperature trends, and the use of dropout successfully prevented overfitting. This comprehensive study, leveraging real climate data, details a systematic approach to developing and evaluating neural networks for time series forecasting. The results show that stacked GRU and LSTM models outperform other tested models in capturing detailed climate data patterns