**ADVANCED MACHINE LEARNING**

**ASSIGNMENT 3: TIME SERIES DATA**

**VIJAY CHARAN REDDY GOTTAM**

**KSU ID: 811292996**

**Summary:**

I have conducted a study utilizing the Jena Climate dataset, focusing on developing and evaluating various neural network architectures for time series prediction. The primary objective is to accurately forecast future temperature readings based on past climate data in an efficient manner. To obtain preliminary insights, the temperature time series is visualised and the data is carefully analysed. In our time series analysis investigation, we evaluated 14 models. A baseline model employing established techniques served as a benchmark, achieving a Mean Absolute Error (MAE) of 2.62. However, a basic machine learning model with a single dense layer exhibited inferior performance (MAE of 2.64). This suggests that flattening the time series data during model creation may have disrupted the inherent temporal dependencies, leading to decreased accuracy. We further explored a convolutional neural network approach. However, its treatment of all data segments uniformly, even with pooling techniques, appeared to disrupt the sequential nature of the data, resulting in unsatisfactory performance.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Dense Units | Dropout | Loss | Test MAE |
| Basic Machine Learning Model | 16 | No | 11.2405 | 2.64 |
| 1D convolutional model | 16 | No | 15.9922 | 3.16 |

Recognizing the limitations of flattening time series data, we shifted our focus to Recurrent Neural Networks (RNNs). RNNs excel at handling sequential data by leveraging past information to inform current predictions. This "memory" allows them to learn complex patterns within sequences of varying lengths. However, the basic Simple RNN proved to be the weakest performer. Despite its theoretical ability to remember all previous steps, the "vanishing gradient problem" hinders its practicality, especially in deeper networks. This problem makes training the network very difficult.

To address this limitation, advanced RNN variants like Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) were developed. We explored the GRU model within Keras and achieved the best results. This success can be attributed to GRU's ability to capture long-range dependencies in sequences while maintaining superior computational efficiency compared to LSTMs.

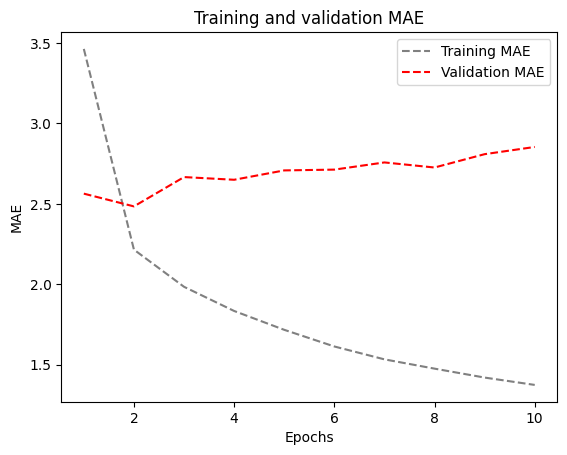
RNN Models

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LSTM Models | 16 | No | 10.9326 | 2.60 |
| LSTM Models | 16 | Yes | 10.6593 | 2.56 |
| GRU (later replaced with LSTM)- not needed but did for comparison | 16 | Yes | 10.1006 | 2.49 |
| Bi directional LSTM Model | 16 | No | 11.1661 | 2.65 |

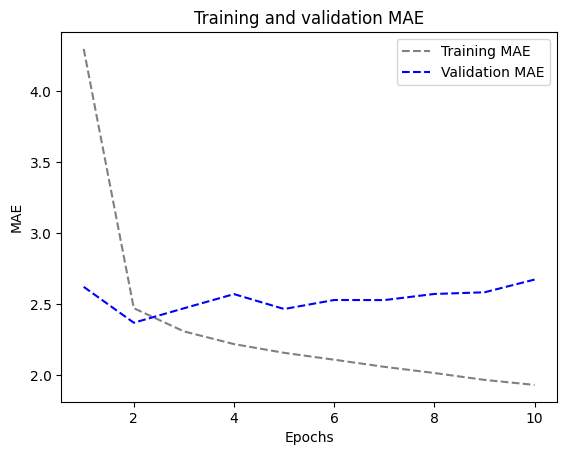
We explored the popular Long Short-Term Memory (LSTM) architecture, known for its effectiveness with time series data. We tested six LSTM models with varying numbers of units (8, 16, and 32) stacked in recurrent layers. Interestingly, the model with the fewest units (8) achieved the best performance. To further enhance the models, we incorporated recurrent dropout to combat overfitting and experimented with feeding data in both directions (bidirectional). This bidirectional approach aimed to improve accuracy and mitigate the "forgetting problem" inherent in some RNNs. Notably, all LSTM models delivered consistently lower MAE values compared to the baseline model, demonstrating their superiority in this task.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Dense units | Dropout | Loss | Test MAE |
| LSTM | 32 | No | 11.9443 | 2.73 |
| LSTM | 16 | No | 11.3917 | 2.63 |
| LSTM | 8 | No | 10.3211 | 2.50 |

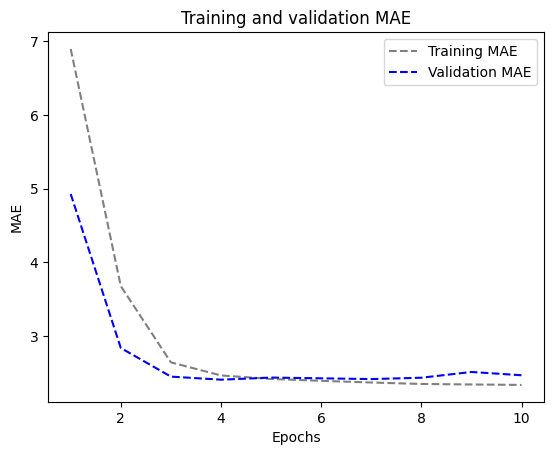
LSTM with 32 units



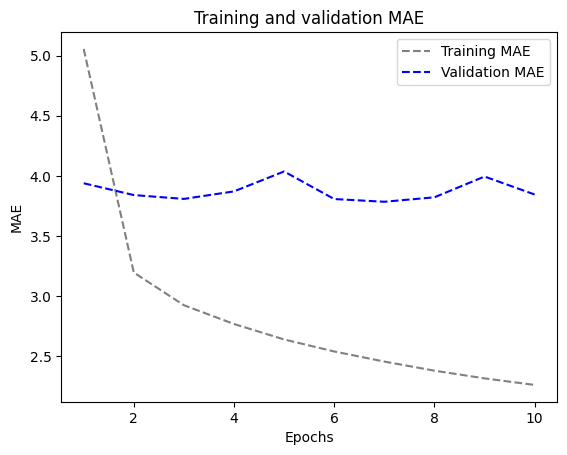
LSTM with 16 units



LSTM with 8 units



Our investigation culminated in exploring a hybrid model combining a 1D convolution with an RNN, but this resulted in a higher MAE 3.96. This suggests that convolutions might struggle to preserve the sequence information crucial for time series data.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Combination | 16 | Yes | 24.7571 | 3.96 |

Here are some key takeaways:

* Simple RNNs are best avoided: Their vanishing gradients make them unsuitable for capturing long-term dependencies in time series.
* Advanced RNNs like LSTM and GRU are preferred: These architectures are specifically designed to address the limitations of simpler RNNs.
* GRU can be an efficient alternative to LSTMs: Our experiments showed promising results with GRU, suggesting it might offer a good balance between performance and efficiency.
* Optimize GRU models: Experiment with hyperparameters like hidden units, dropout rates, and bidirectional processing to fine-tune your GRU model.
* Focus on RNN architectures: While we explored a 1D convolution and RNN combination, it's generally advisable to prioritize architectures specifically designed for sequential data like RNNs. Convolutions might disrupt the order of information, hindering their effectiveness in time series analysis.

Conclusion:

Among the various models i assessed, the stacked GRU and LSTM architectures stood out for their superior performance, boasting the lowest test MAE. This can be attributed to their adeptness at capturing long-term dependencies within the time series data, as well as the regularization provided by dropout. Through our exploration using the Jena Climate dataset as a practical case study, we present a methodical approach to the development and evaluation of multiple neural network designs for time series forecasting. my findings underscore the effectiveness of stacked GRU and LSTM models over alternative architectures in uncovering intricate patterns and connections within climate data.