**TIME SERIES REPORT**

**Assignment-3**

Harini Padmaja Solleti

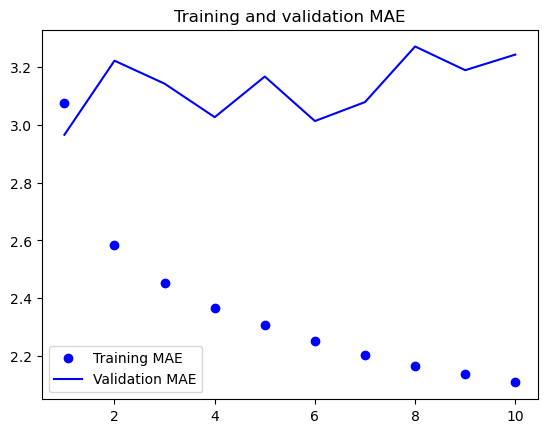
Priyanka Jonnala

Using the temperature-forecasting dataset, we are going illustrate the key qualifications between timeseries information and the other dataset types we've as of now worked with. We will see that convolutional and profoundly associated systems are deficient to manage this kind of dataset, whereas repetitive neural systems (RNNs), an unused frame of machine learning approach, completely flourish at understanding this kind of issue.

In all our studies, a total of 50% of the data is used for training, 25% for validation, and the remaining 25% for testing. When working with time series data, it is important to use validation and test data that are more recent than training data because our goal is to predict the future based on the past. The validation/testing split should reflect this. The exact formulation of the problem is: Is it possible to estimate daily temperature using data collected once per hour for the past 5 days.

We created a total of 14 models to analyze time series data. The first model relied on common sense methods and produced a baseline mean absolute error (MAE) of 2.44. When building the basic machine learning model with thick layer models, the MAE was slightly higher than 2.62. Thick film models performed poorly because the time series data were flattened, and temporal context was lost. Although not consistent, some of the validation losses are close to the untrained baseline.

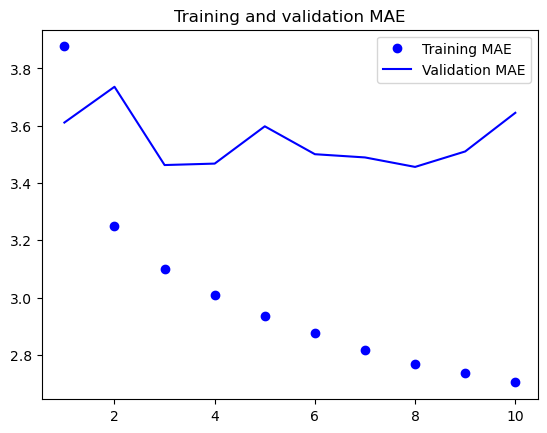
**Basic Machine Learning Model: Dense Layer**



**1D Convolutional Model:**

In terms of using appropriate architectural priorities, a convolutional model is appropriate since the input sequence consists of daily cycles. Just as a spatial convolutional network can reuse the same representation at multiple locations in an image, a temporal convolutional network can also use the same representation across different days.

In fact, the performance of this model is significantly worse than that of the tightly connected model. Since not all meteorological data meet the translational invariance assumption, the validation MAE can only be achieved around 3.70 degrees, which is far from a practical baseline.



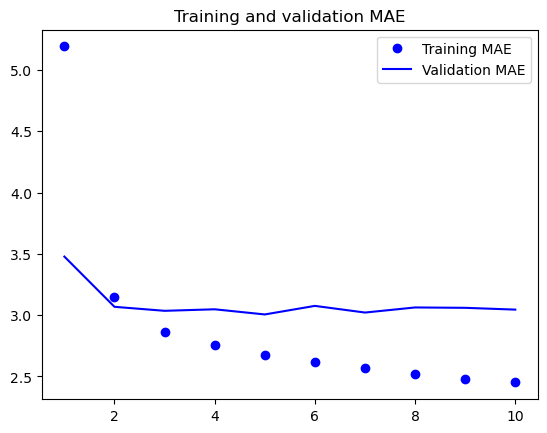
The convolutional data did not perform as well as the dense model or common sense. This might be the case because for meteorological data, the translation invariance assumption is not very strong. The data's order is very important. When it comes to forecasting the temperature for the next day, recent historical data is noticeably more useful than data collected several days prior. Sadly, this crucial temporal order is beyond the reach of a 1D convolutional neural network.

**Simple RNN:**

Recurrent neural networks (RNNs) have a remarkable ability to integrate past time step information into today's decision-making processes, thereby identifying complex relationships and trends in continuous data. Since the internal state of an RNN acts as a memory of previous inputs, it is possible to describe sequences of different lengths. Although a simple RNN could theoretically store data from any point in time in the past, difficulties arise in practice. This makes training deep networks difficult due to the vanishing gradient problem. Additionally, the graph shows that the simplest RNN has the worst performance of all.

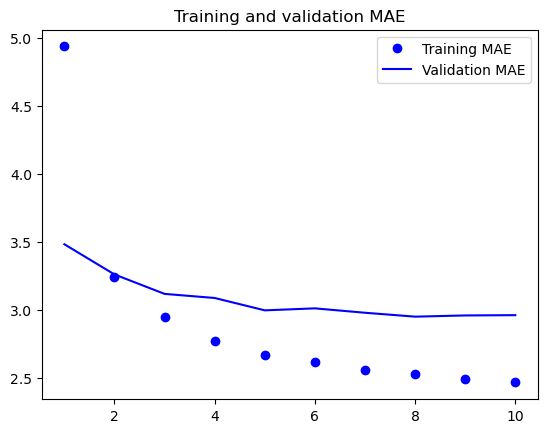
As a solution to this problem, we must create GRU’s and RNN’s.

**GRU:** Instead of LSTM layers, we will use Gated Recurrent Unit (GRU) layers. GRU and LSTM are quite similar; think of it as a shortened, simpler form of the LSTM architecture.



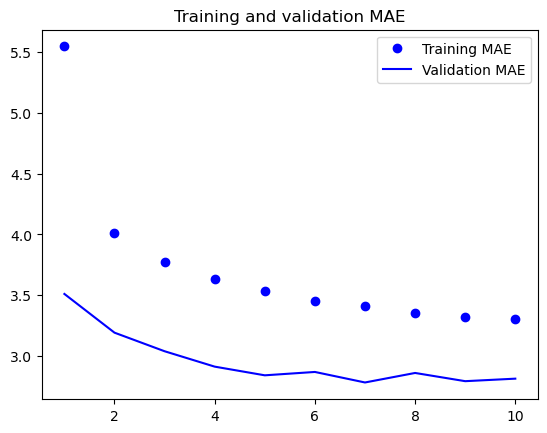
We found that the MAE – 3.21 test is found to be the most efficient model, it is less computationally expensive than long and short-term memory (LSTM) models, and it captures an efficiently separate long-range dependencies sequentially data compared to other models.

**LSTM:** Recurrent neural networks (RNN) provide a class of neural network architecture specifically designed for sequential data processing tasks. Among them, the LSTM (Long Short-Term Memory) class is very popular. We will start by exploring the LSTM layer to understand its functions.



We achieved a test MAE of 3.06 degrees and a confirmation MAE as low as 2.39 degrees. The LSTM-based model outperformed the conventional baseline, highlighting the effectiveness of machine learning in this effort.

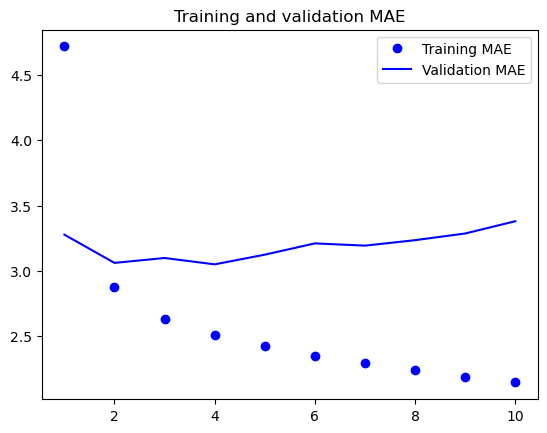
**LSTM- Dropout Regularization**



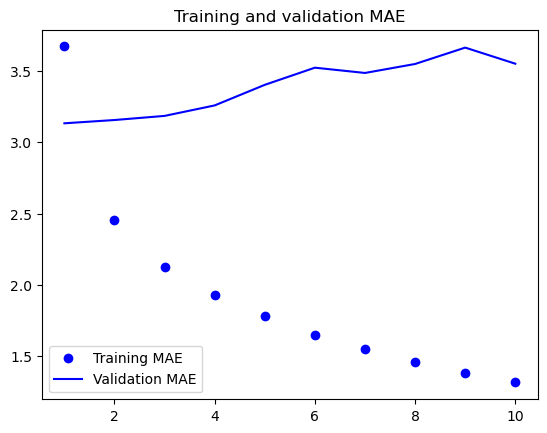
We achieved a test MAE of 2.99 which is not too bad.

Using 8, 16, and 32 units as the number of units varying within the stacked repetition layers, we constructed six distinct LSTM models.

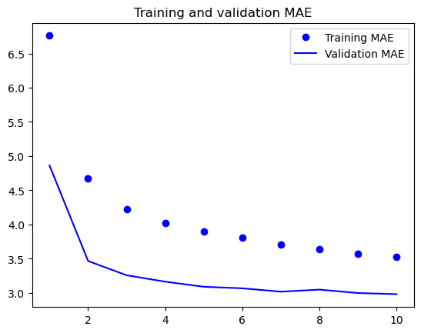
**LSTM stacked with 16 units:**



**LSTM stacked with 32 units:**

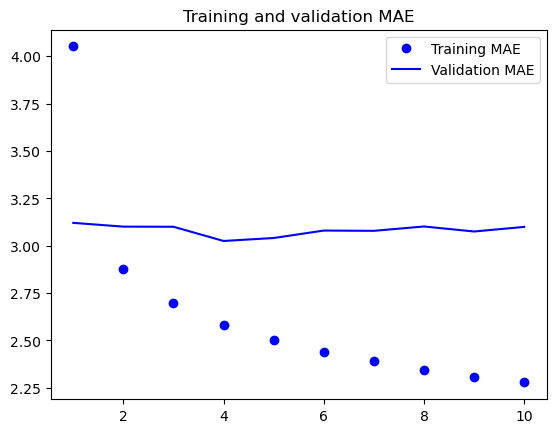


**LSTM stacked with 8 units:**



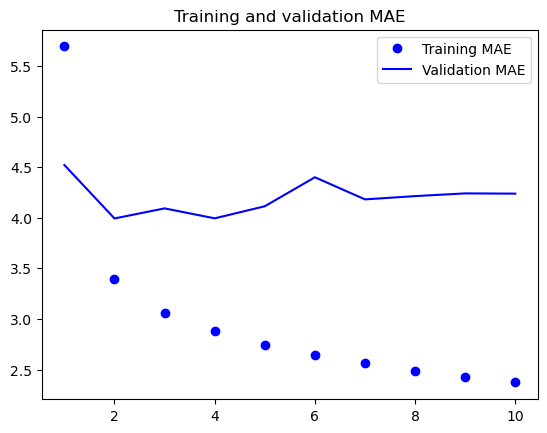
With an MAE score of 3.11, the 8-unit configuration among these alternatives showed the best performance.

**LSTM: Bidirectional**

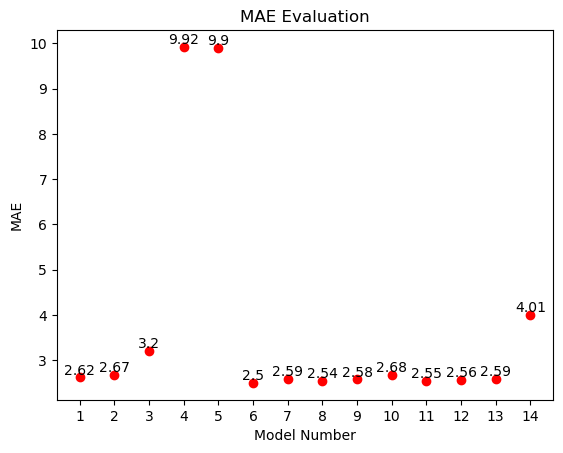


**1D Convnets and LSTM together:**

The model I developed using both RNN and 1D convolution, produced incomplete results with 3.78 MAE. The order of information can be destroyed by the convolution limit, which can be a cause of poor performance.



**The Complete Model’s Performance:**



In summary, my observations show that using LSTM and GRU (advanced RNN architecture) is the best choice, while combining RNN with 1D convolution produces unsatisfactory results. After some testing, I think GRU is a more efficient choice for processing time series data, although bidirectional LSTM is still a popular choice. Hyperparameters that need to be tuned to maximize GRU include the number of units in stacked iteration layers, iteration dropout rate, and the use of bidirectional data.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_