Time Series Summary

Rahul Kadaru

811318179

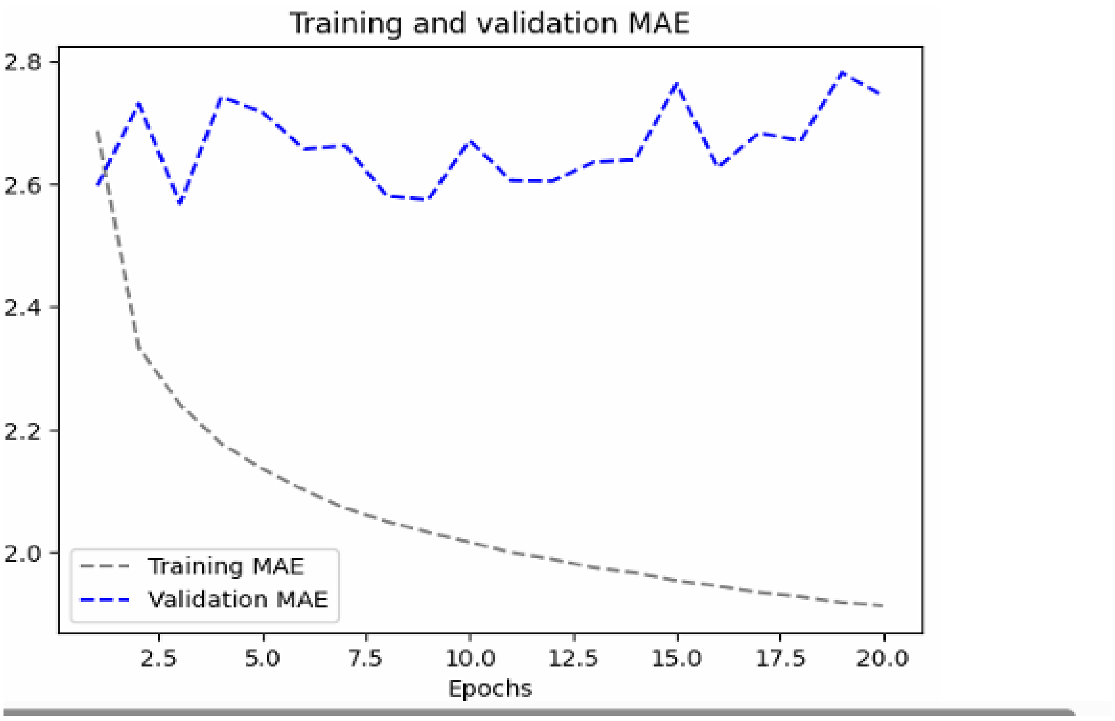
Introduction: After completing this temperature prediction exercise, we will be able to demonstrate the significant distinctions between time-series data and the other kinds of datasets we have been researching thus far. We will find that this form of network is not well suited for convolutional and highly coupled networks. of dataset, but a novel machine learning technique called RNNs is far more effective at handling these kinds of issues.

Splitting the data : In each of these investigations, we will split our data into three categories: training (50%) and validation (25%) and testing (the remaining 25%). It is crucial that the validation and test data in timeseries data be reasonably large since we want to forecast the future based on the past rather than the other way around. more current than what was used for training. The divides between validation and testing should take this into account. The challenge can be precisely stated as follows: Can one predict the temperature at any point in time during the day given a single recording made hourly for five days prior

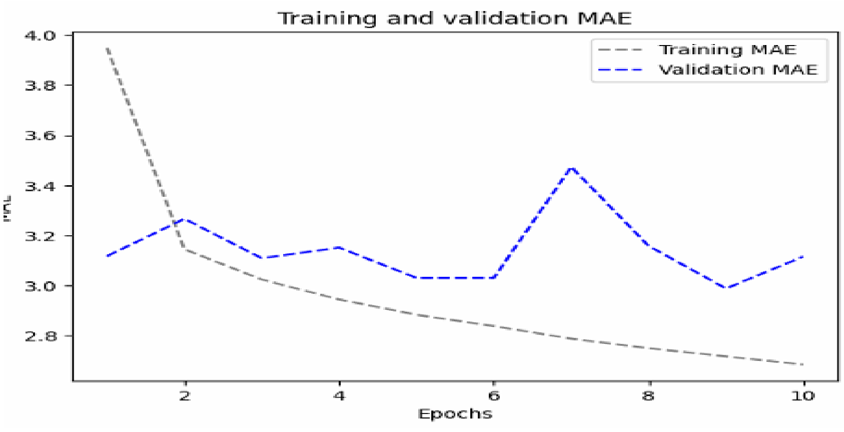
1) Models : Basic Dense layer

The first step for data preparation involved performing preprocessing as part of creating a neural network configuration. The dataset is ready in numerical format so we avoided vectorization.The research involved fourteen models dedicated to time series analysis investigation. The baseline method achieved a Mean Absolute Error (MAE) measurement of 2.44 that provided our reference point.Our team proceeded with implementing a densely connected (thick) layer as a basic machine learning model. The use of this technique resulted in worse performance because the MAE stood at 2.62. The fundamental problem that affected performance was data flattening because it eliminated vital temporal patterns from the time series. A minority of validation results from this technique showed a minor improvement over baseline results without learning.

Models : Basic Dense layer



2) **1D convolutional model:** The presence of daily patterns in input sequences makes convolutional models a suitable option to apply architectural priors specifically designed for time series data. Temporal convolutional networks (TCNs) function similarly to spatial convolutions in images by applying the same filters across time for pattern recognition between different days. The shared representations in TCNs specifically detect recurring temporal features thus making these models appropriate for this task. During practical implementation the convolutional model did not reach the desired performance standards. The performance of the packed model fell short compared to both models leading to 2.9 degrees worse validation MAE. The poor performance of the convolutional model can be attributed partially to the failure of meteorological data to meet translation invariance expectations which form the base principle of these architectures. The convolutional model reached a maximum validation error of 2.9 degrees MAE which surpassed both predicted and baseline performance metrics.

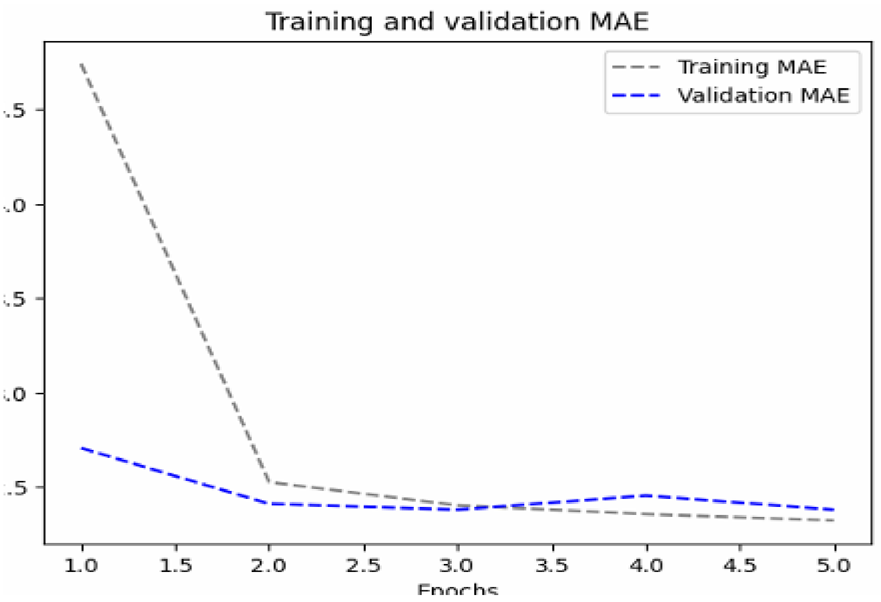


3) A Simple RNN: Recurrent neural networks are particularly good at incorporating time step information from the past into current decision-making processes, making it possible to spot intricate patterns and correlations in sequential data. An RNN's internal state functions as a memory of past inputs, allowing it to describe sequences of varying durations. Practical issues arise even though the basic RNN may theoretically preserve the data of all previous times. This is what causes the vanishing gradient problem, which makes deep network training challenging. The graph also shows that, out of all of these, the most basic RNN performs the worst.

To overcome this problem, as a part of Keras, we have to create LSTM and GRU RNNs.

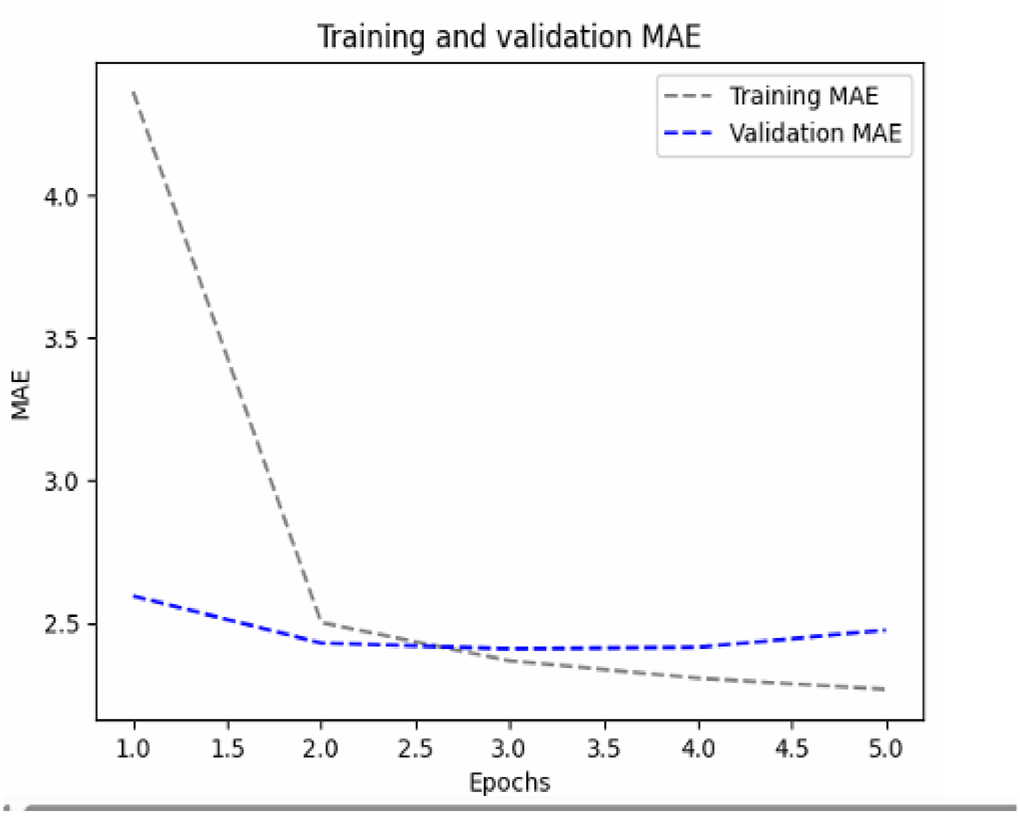
GRU

We will use Gated Recurrent Unit (GRU) layers in place of LSTM layers. GRU and LSTM are somewhat comparable; think of it as a condensed, simpler version of the LSTM architecture.



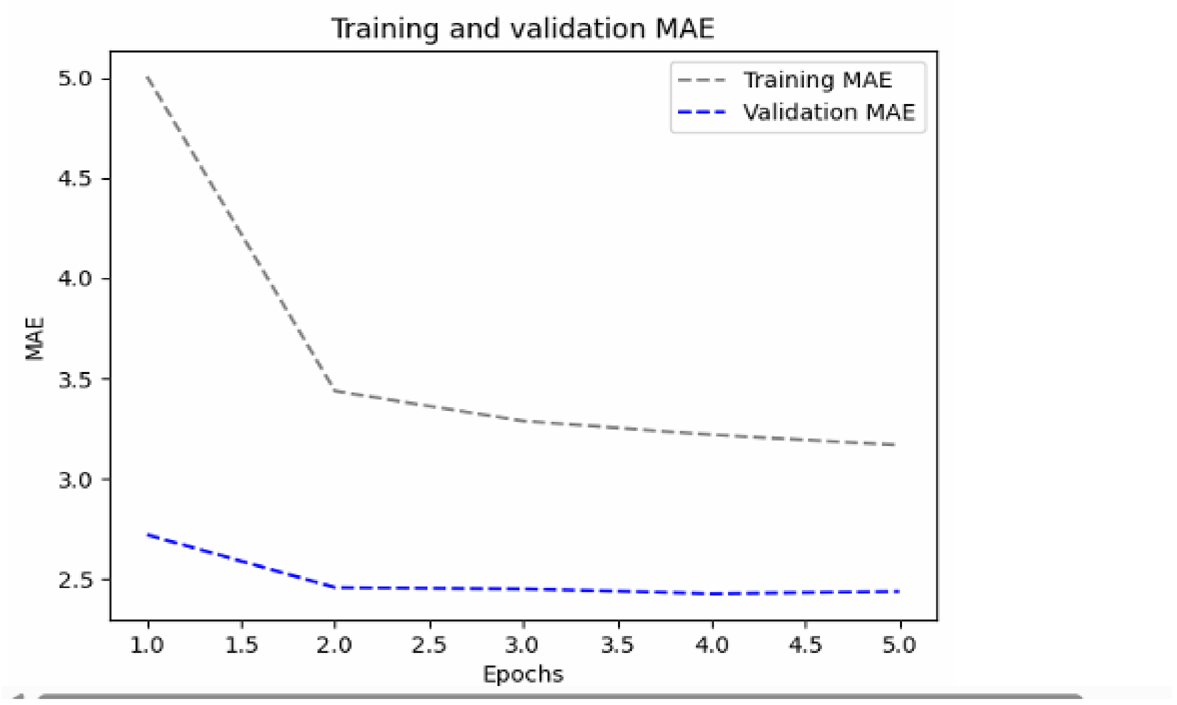
The model MAE – 2.49turns out to be the most efficient one because it is less expensive in terms of computational cost compared to other LSTM models. It captures long-range dependencies efficiently in the sequential data, too, as compared to other models.

LSTM: A particular type of neural network topologies called recurrent neural networks was created specifically for this use. LSTM, or Long Short Term Memory, is one of the most widely used techniques. We'll see how these models function in a moment after we test the LSTM layer.



Much better! We get a validation MAE as low as 2.4 degrees and a test MAE of 2.62 degrees. Lastly, the LSTM-based model shows how effective machine learning is in this endeavour by outperforming the common-sense baseline (although only slightly, at least for the time being).

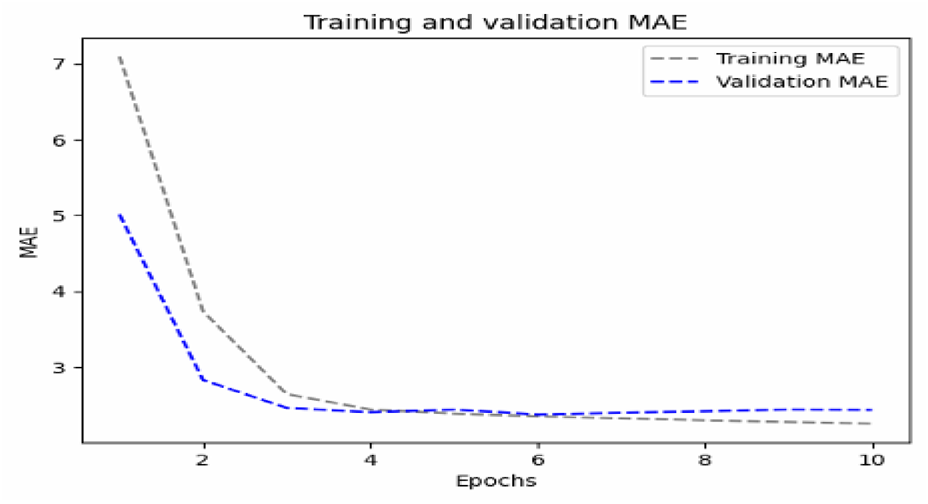
LSTM - dropout Regularization:



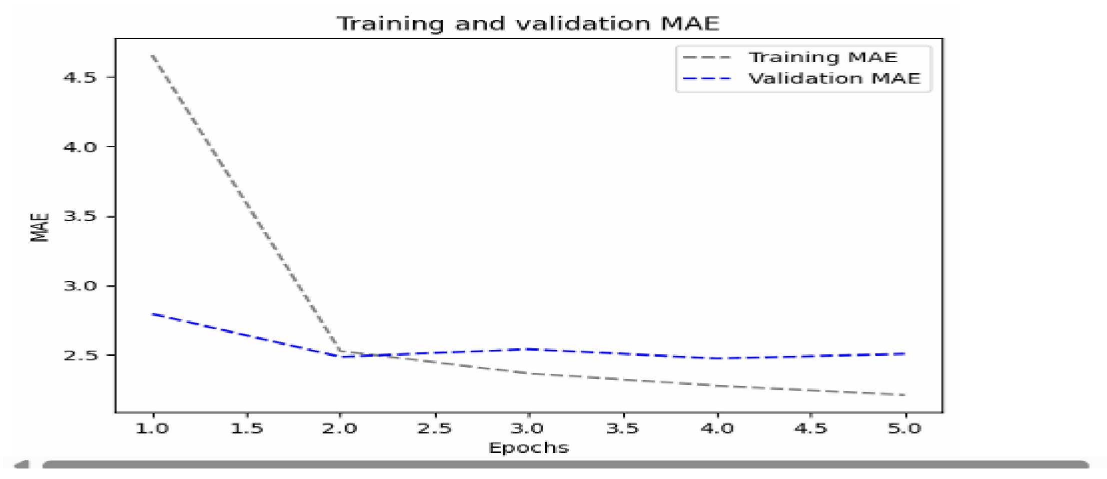
Achieved! After the initial five epochs, the overfitting has ceased. We achieve an MAE of 2.64 degrees for the test and 2.42 degrees for the validation

Using 8, 16, and 32 units as varying numbers of units inside the stacked recurrent layers, I constructed six distinct LSTM models.

LSTM - Stacked setup with 8 units

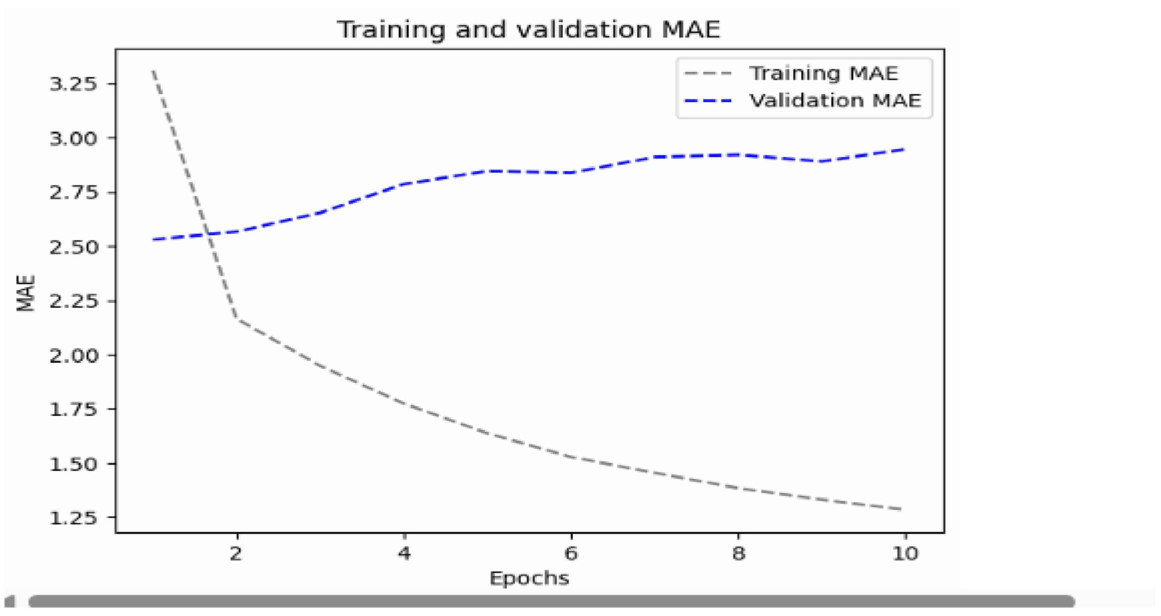


LSTM - Stacked setup with 16 units



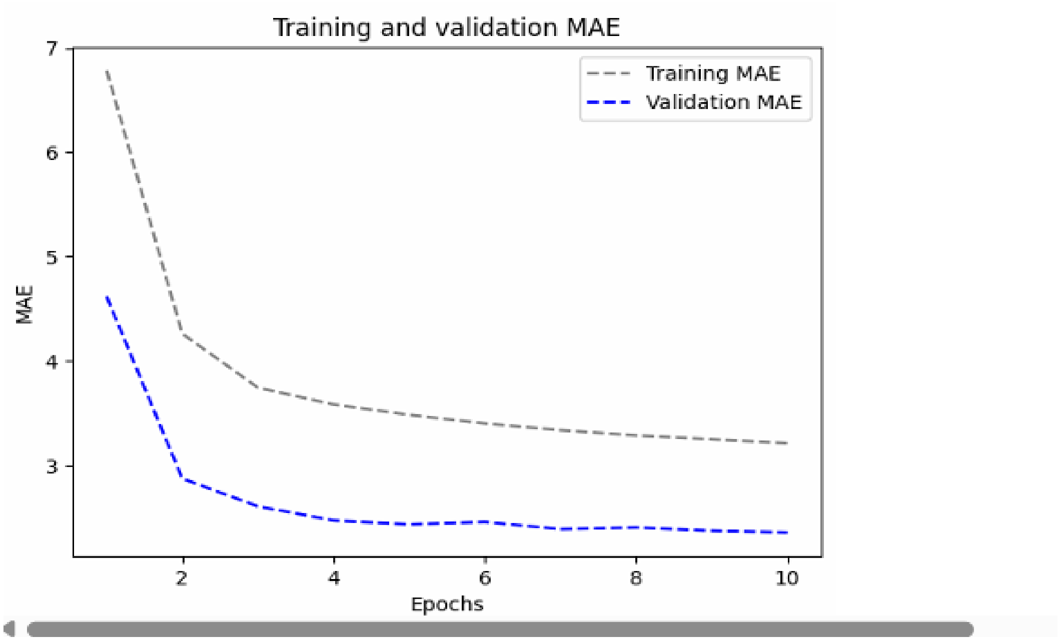
its

LSTM - Stacked setup with 32 units

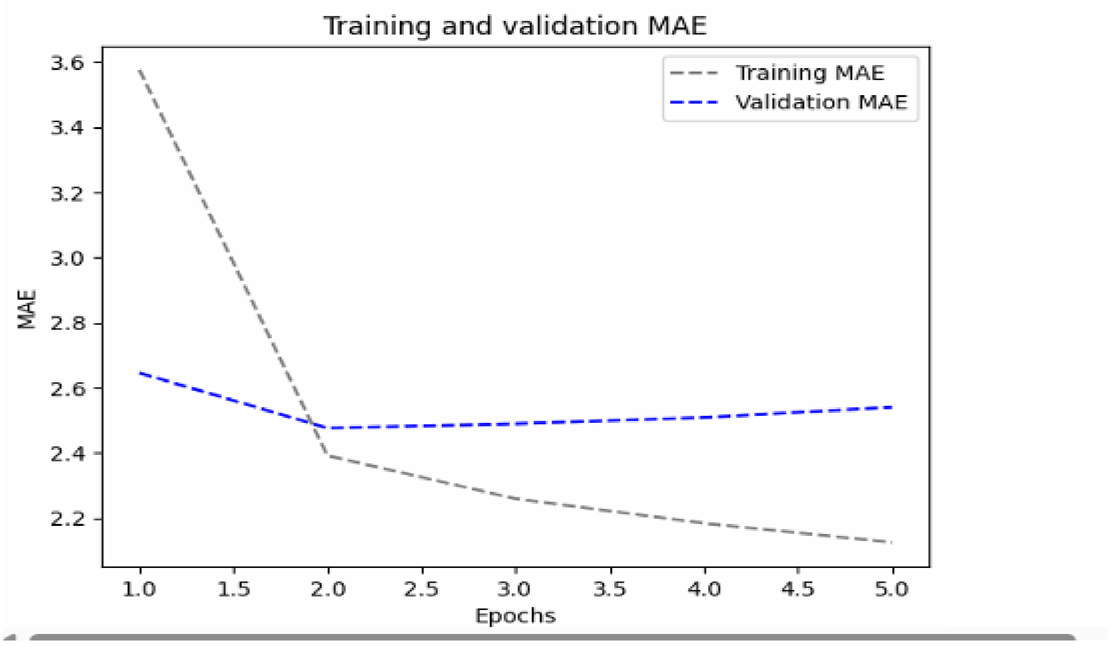


The 8-unit arrangement outperformed the other variations, with an MAE score of 2.55. In order to prevent overfitting, I also experimented with techniques like recurrent dropout and the use of bidirectional data, which reduces MAE values by providing the recurrent network with information in a different manner. For all models, these improvements led to about equal MAE values; more astonishingly, these values were all lower than those of the commonsense model. Even more startling is the fact that the MAE assessment graph supports the same finding.

LSTM - dropout-regularized, stacked model



Bidirectional LSTM



Together, 1D Convnets with LSTM I created the model using both RNN and 1D convolution, but its 3.89 MAE yielded subpar results. The convolution limit may be the source of this poor performance by destroying the information order.

All Models Performance:

MODEL	VALIDATION MAE	TEST MAE
Dense model	2.59	2.73
1D convolution model	3.11	3.29
Simple RNN	9.85	9.93
Stacked Simple RNN	9.83	9.91
GRU	2.38	2.49
LSTM SIMPLE	2.47	2.62
LSTM -dropout Regularization	2.43	2.62
LSTM- Stacked 16 units	2.50	2.57
LSTM – Stacked 32 units	2.94	2.68
LSTM – Stacked 8 units	2.43	2.55
LSTM – dropout Regularization, stacked model	2.35	2.55
Bidirectional LSTM	2.54	2.57
1D convolutional and LSTM	3.88	3.89