**Predicting weather patterns over a period of time.**

**REPORT**

**Likitha Sree Yarabarla**

**Sai Mithra Vaddepalli**

In our time series data analysis, we constructed a total of 14 models. The initial model served as a reference point, utilizing straightforward techniques, and achieving a Mean Absolute Error (MAE) of 2.62. Following this, we developed a basic machine learning model featuring a dense layer, resulting in a slightly elevated MAE of 2.70. However, the performance of the dense layer model was unsatisfactory due to the loss of temporal context caused by the flattening of the time series data. Additionally, we experimented with a convolutional model, which yielded inadequate outcomes as it treated all data segments uniformly, even after pooling, thereby disrupting the sequential order of the data.

Henceforth, we acknowledged that Recurrent Neural Networks (RNNs) offer superior suitability for time series data. A pivotal attribute of RNNs lies in their capability to incorporate past information into current decision-making processes. This empowers the network to discern dependencies and patterns within sequential data. The RNN's internal state effectively serves as a memory, preserving past input information and facilitating the modeling of sequences with varying lengths. Nonetheless, the fundamental Simple RNN often proves too rudimentary for practical applications. Notably, Simple RNN exhibits a significant limitation: depicted in graphical representations, it consistently registers the poorest performance among all models. Despite theoretically being capable of retaining information from all preceding time steps, Simple RNN tends to struggle in practice, particularly within deep networks, owing to the notorious "vanishing gradient problem". This issue renders the network nearly intractable. To address this challenge, more sophisticated RNN variants such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) were developed and seamlessly integrated into Keras. Our experimentation with the basic GRU model showcased the most promising results among all models, primarily due to its adeptness at capturing long-range dependencies in sequential data while also being more computationally efficient compared to LSTMs.

The renowned architecture renowned for proficiently managing time series data comprises Long Short-Term Memory (LSTM) networks. We conducted trials with six distinct LSTM models, each featuring different unit configurations in stacked recurrent layers (8, 16, and 32). Surprisingly, the model with 8 units showcased the most superior performance. Furthermore, to mitigate overfitting, we implemented recurrent dropout and explored bidirectional data presentation to improve accuracy and tackle the forgetting issue. Remarkably, all LSTM models exhibited comparable Mean Absolute Error (MAE) values, consistently outperforming the common-sense model.

In conclusion, our attempt to merge a 1D convolution model with an RNN resulted in a higher Mean Absolute Error (MAE) of 3.79, likely due to the convolution's difficulty in preserving the sequential order of information. From my observations, it is advisable to steer clear of simple RNNs for time series analysis, given their susceptibility to the vanishing gradient problem and their inability to effectively capture long-term dependencies. Instead, opting for more advanced RNN architectures like LSTM and GRU, designed to address these issues, is recommended. Although LSTM is widely favored for handling time series data, our experiments indicate that GRU might offer more efficient outcomes. To enhance GRU models, focus on adjusting hyperparameters such as the number of units in stacked recurrent layers, recurrent dropout rates, and the utilization of bidirectional data presentation. Moreover, prioritize RNN architectures specifically tailored for sequential data, as our exploration found that combining 1D convolution with RNN did not yield optimal results. Convolutional methodologies tend to disrupt the sequential nature of data, making them less suitable for time series data analysis.