ARTIFICIAL INTELLIGENCE

COSC 6368 SUMMER 2022

Final Project

Analysis of Air Quality Prediction using Autoregressive LSTM VS GRU

Team:

Sindhuja Thogarrati(1857970)

Sainadh Chebrolu(2151564)

Akhila Gaddam(2153343)

1. INTRODUCTION:

With the development of industry, air pollution has become a serious problem. It is very important to create an air quality prediction model with high accuracy and good performance. Globally, the problem of poor air quality as an environmental risk has grown significantly. As a result, predicting and assessing ambient air quality has grown in importance. In general, the term "air quality" refers to the measurement of clean air in a certain area. It is calculated by measuring several atmospheric pollution parameters. The concentration of air pollutants is anticipated using traditional methods, which are computationally intensive. Additionally, these approaches are unable to make sense of the wealth of information at hand. The suggested work offers a deep learning method combined with auto regression for quantifying and forecasting air quality in order to address this problem. Long Short-Term Memory (LSTM) is a unique type of structured memory cell that is used in a framework based on recurrent neural networks (RNNs) to do air quality prediction. In this work, we propose LSTM and BiGRU (Bidirectional Gated Recurrent Unit) integrated with Auto-regressor to capture the dependencies in various pollutants and to perform air quality predictions. Further, we have analyzed the forecasts on the state-of-the-art DeepAR model as well.

2. SURVEY:

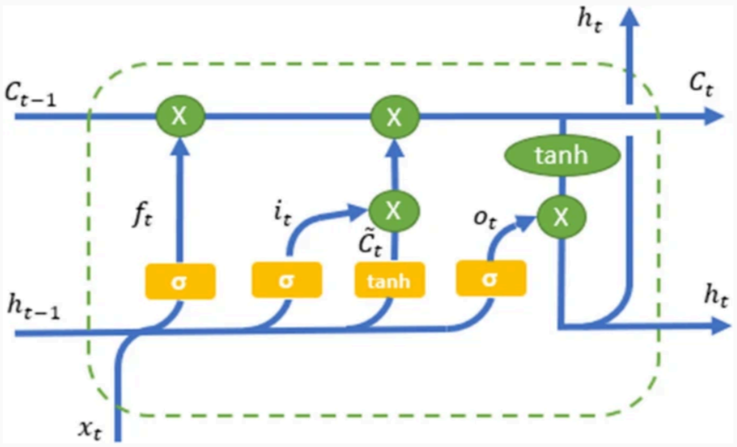
Numerous statistical methods are used to predict the air quality. In this study, we look into the abilities of different deep learning models to predict the concentration of PM2.5. As a result, we choose to employ the AR-LSTM, GRU and DeepAR. Then, we succinctly outline each network:

2.1. LSTM [Long Short-Term Memory]

LSTM is an improved approach to conventional RNN. By including a memory block, LSTM resolves the RNN's vanishing gradient issue. With a constant error carousel (CEC) value of 1, a memory block is a complicated processing unit that has at least one memory cell as well as a few multiplicative gates serving as its input and output. The memory block does not receive any outside signal values, which causes the error value to become active for the duration of the time step. The entire operation of the memory block is under the control of the multiplicative gates. An input gate regulates the flow of input into a memory cell by controlling the activation of the cell. Three gates make up an LSTM: an input gate that decides whether to accept fresh data, a forget gate that eliminates unimportant information, and an output gate that selects the information to be produced. These three gates operate in the 0 to 1 range and are analogical gates based on the sigmoid function.

Fig. Below shows these three sigmoid gates. The cell state is represented by a horizontal line that passes through it.

Fig. 1



Architecture of the LSTM cell

Fig 1. Architecture of an LSTM cell

The input feature from the previous time step is sent into the LSTM cell along with cell state C_{t-1} and hidden state h_{t-1}. Depending on the input feature and the hidden state, the forget gate will generate an output. The result will range from 0 to 1 since the sigmoid activation function will be applied. The output of the forget gate determines how much data will be forgotten by the cell.

The input feature will choose how much of the most recent data should be saved after receiving the hidden state of the previous time step from the input gate. The output from the input gate is multiplied by the new values produced by the tanh function, which are then added to the cell state. Tanh function and an output gate produce output. The tanh function keeps the values in the range -1 to 1. The following cell will get it as output along with the state of the previous cell.

The LSTM architecture estimates an output $o_t=(o_{t\ 1},o_{t\ 2},o_{t3},\ldots,o_{t(T-1)},o_{t(T)})$ by updating three multiplicative units (input I output (op), and forget gate (fr)) on the memory cell with continuous write, read, and reset operations on the memory cell (mc) from time t=1 to T.

LSTM formulas are listed below:

Inputgate : $i_t = \sigma(W^{(it)}\overline{x}_t + W^{(it)}h_{t-1})$ (1)

Forgetgate : $f_t = \sigma(W^{(if)}\overline{x}_t + W^{(hf)}h_{t-1})$ (2)

Outputgate : $i_t = \sigma(W^{(io)}\overline{x}_t + W^{(ho)}h_{t-1})$ (3)

ProcessInput : $\widetilde{C}_t = tanh(W^{(i\tilde{c})}\overline{x}_t + W^{(h\tilde{c})}h_{t-1})$ (4)

Cellupdate : $C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t$ (5)

Output : $y_t = h_t = O_t + i_t * tanh(C_t)$ (6)

Since LSTMs are frequently used for sequential analysis, they can be trained to forecast air quality index levels for the upcoming hour or even the upcoming month using the historical data collected by sensors at various weather stations.

2.2. Gated Recurrent Unit [GRU]:

An addition to the LSTM network is GRU. They all include balancing the data flow within the unit. The GRU, or gated recurrent unit, is an improvement over the traditional RNN and is incorporated into RNN. It is comparable to an LSTM unit. The reset and update gates make up the GRU unit. The GRU architecture is shown in Figure below. While the update gate is used to choose the number of the candidate activation that updates the cell state, the reset gate is intended to erase the previous state between the previous activation and the next candidate activation.

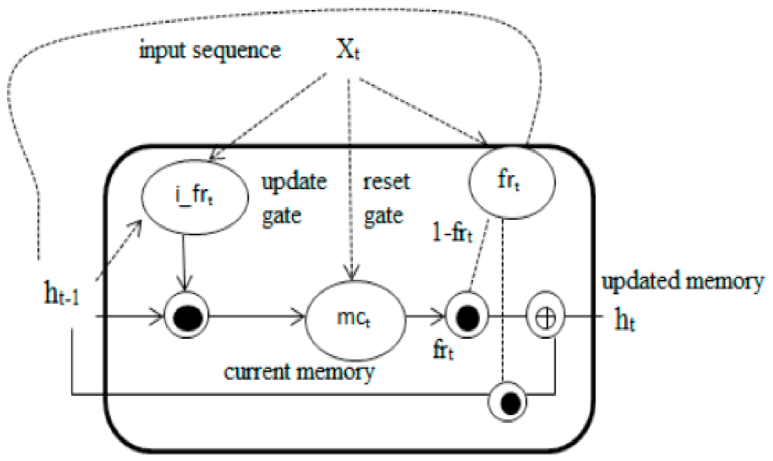
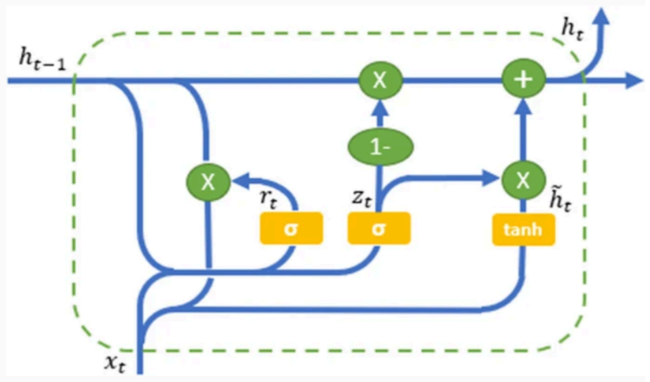


Fig 2. Internal architecture of GRU cell

The update gate is used to regulate how much data from hidden states in the past is carried over into the current state. More information about past states is brought in as the update gate value increases. The amount to which the information from earlier stages is discarded is controlled by the reset gate, and the smaller the value of the reset gate, the more it is ignored. As a result, long-term dependencies are accompanied by the activation of update gates, but short-term relationships are typically captured with frequent activation of reset gates.

Fig. 2



Architecture of the GRU cell

Fig 3. Architecture of a GRU cell

2.3. Bidirectional RNN models

A bidirectional LSTM applies two LSTM layers on data; one in the forward direction and the other in the backward direction. The results from these two LSTM layers can be combined to give the final output of Bidirectional LSTM models, both GRU and LSTM. Bidirectional methods are preferred as they provide extended LSTM capabilities on training data, since they train it twice, both forwards and backwards. These models have proven to achieve a higher predictive performance in time series data.

2.4. DeepAR:

DeepAR is a forecasting algorithm used for forecasting scalar one dimensional time-series data. It's a recursive neural network designed by Amazon research group. Classic forecasting methods like ARIMA and ETS can only be trained on individual time-series data and forecast it. However, DeepAR is designed to learn multiple related time series and forecasts using the combined knowledge of all the related data. When a dataset contains multiple related time series, DeepAR outperforms the traditional ARIMA and ETS algorithms.

The model has multiple tuneable hyperparameters like context_length, prediction_length, learning_rate, dropout_rate, embedding_dimension, num_cells and num_layers. The context_length decides the number of past records which the model has visibility on. And prediction_length decides the number of future records the model can predict. Amazon suggests keeping prediction_length less than or equal to context_length to ensure model's predictions are close to real values.

3. IMPLEMENTATION:

3.1. Auto-Regression:

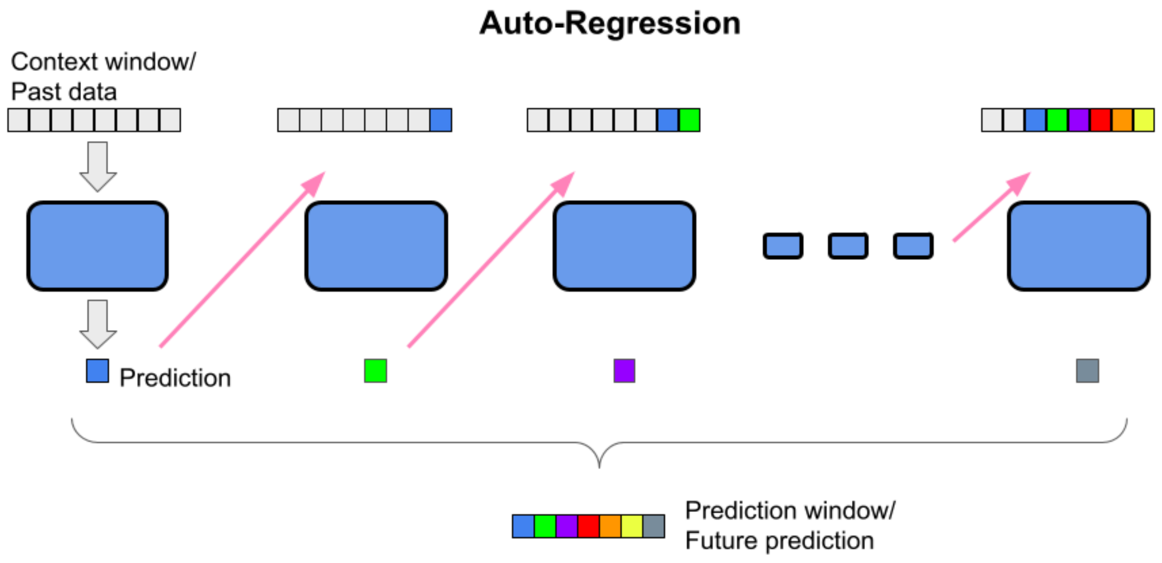

Auto-Regression
Context window/ Past data
Prediction
Prediction window/ Future prediction

Fig 4. Proposed AR-Based architecture

The blue rectangle represents the single layer LSTM, BiLSTM or GRU model.

Auto-regression is the method in which a linear combination of past values of the variable are used for prediction/forecasting. An autoregressive model of an order 'p' can be written as:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} \cdot \cdot \cdot \cdot + \phi_p y_{t-p} + \varepsilon_t$$

Here y_t is the prediction value at time stem t. ε_t is the white noise and ϕ_p is the parameter which is multiplied with y_t. This is like a multiple regression but with a lagged values of y_t. We can refer to this model as **AR(p)** which represents an Auto-regressive model of order **'p'**.

Few things we need to consider while building an auto-regressive model are the length of the context window and the length of the prediction window. An auto-regressive model uses its previous prediction as input for its future prediction. i.e. prediction y_{t-1} is used as input for predicting y_t.

The context window defines the length of the linear combination of $(y_{t-1}, y_{t-2}, \ldots, y_{t-c})$ previous predictions to be used for new prediction y_t. And the prediction window defines the number of time steps the auto-regressive model can be called to predict the future. An auto regressor with prediction window 10 can predict up-to 10 days in the future considering each time step accounting for 1 day.

3.2. Visualizing of Time Series Data

When evaluating a time series model's stability over time, rolling analyses are frequently used. An important presumption when using a statistical model to analyse financial time series data is that the model's parameters won't change over time. To illustrate the outcome of the anticipated air quality plot, we use matplotlib. The following graphs show the Air quality of India (2017-2022) and Air quality of 1 year (2017-2018). We have chosen to average the data from hourly basis to daily basis. This illustration gives us an idea of how the patterns in data change over time.

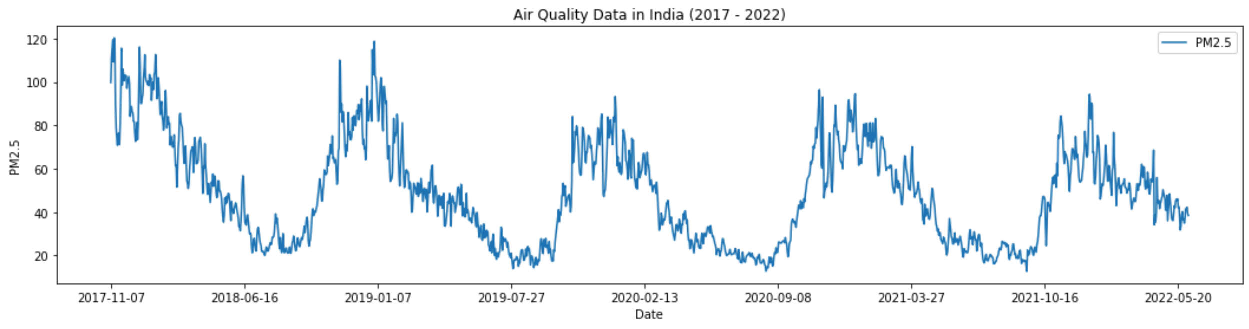


Fig 5. Daily trend of Air Quality Data

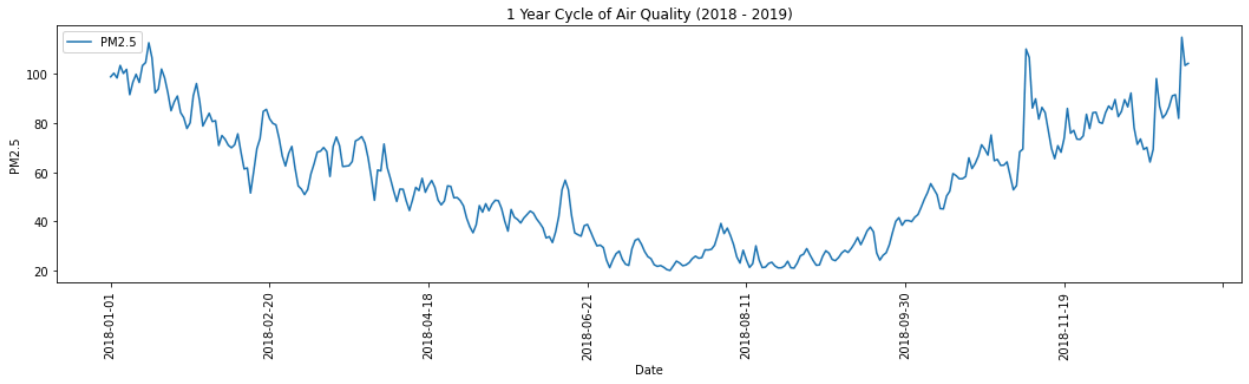


Fig 6. Monthly trend of Air Quality Data

3.3. Prediction using AR-based models:

A general LSTM is a sequential model, which is made up of a linear stack of layers. Then we introduced dense. which is the standard deeply coupled neural network layer. It is used to change the output vector's dimensions during backpropagation. Next, we establish our RNN using a regression model, that reads the sequential data in and adds it to the regressor model to achieve this.

The neural network receives the input and is trained for prediction using random biases and weights. A sequential input layer, three LSTM layers, and a dense output layer with a linear activation function make up the LSTM model. The output value produced by the RNN's output layer is contrasted with the desired value. The backpropagation algorithm reduces error, or the discrepancy between the desired output value and what is actually produced.

We employ auto-regression technique over the existing LSTM model with ADAM optimizer and length of **context window as 100** and length of **prediction window to be 365**. This model takes 100 days data as input and can predict up-to 365 days in the future which is a full 1 year prediction based on past 3 months data.

As the context window is 100, the model's predictions are reliable up-to 100 days into the future. i.e., a prediction window with 100 length is reliable. However further predictions are made based on the previous predictions which will result in higher deviation from the original data.

The following graphs show the output of the autoregressive model trying to predict 365 days into the future using past data.

3.4. Prediction using AR-LSTM Model:

We have used matplotlib for visualisation. The graph obtained below is plotted against models predictions over training data, testing data and actual data.

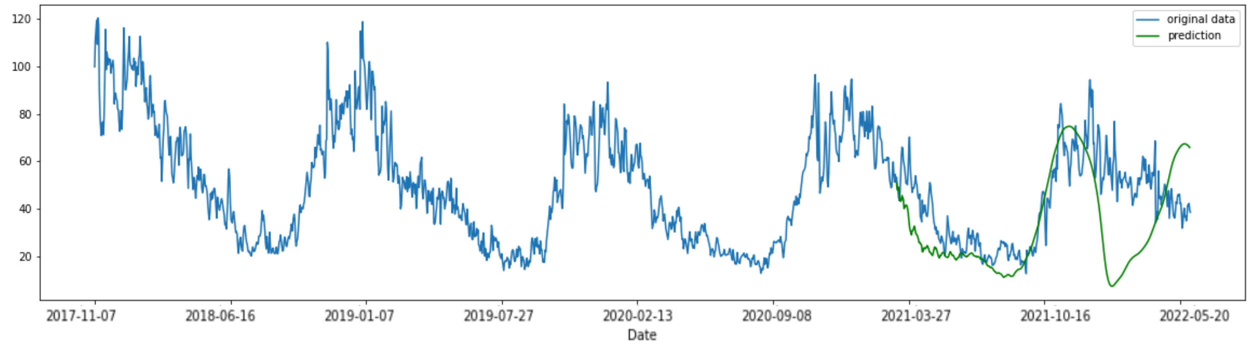


Fig 7. Prediction using AR-LSTM model

3.5. Prediction using AR-Bi-directional LSTM:

The graph obtained below is plotted against predictions over training and testing data overlapped over original values.

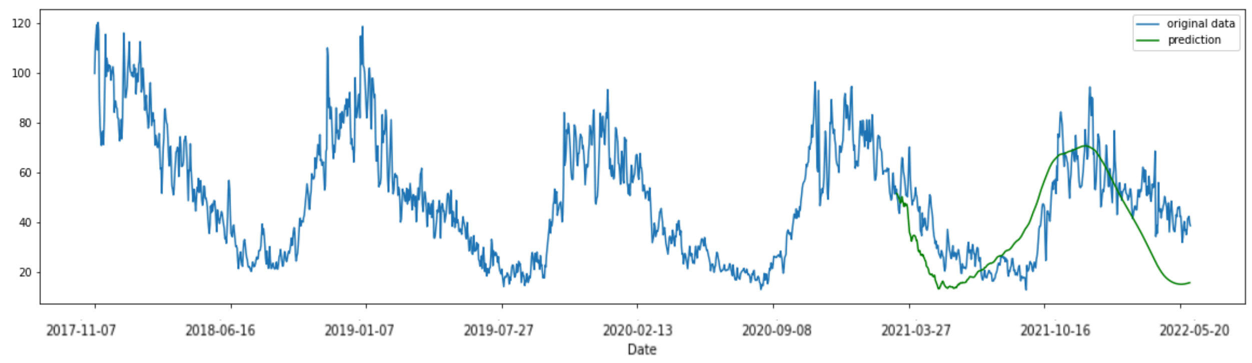


Fig 8. Prediction using AR-BiLSTM model

3.6. Prediction using AR-GRU Model:

The graph obtained below is plotted against predictions over training and testing data overlapped over original values.

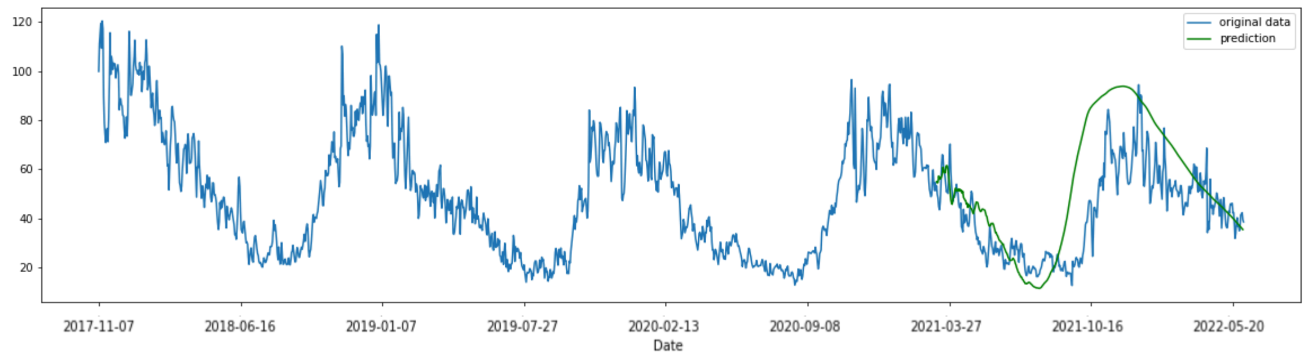


Fig 9. Prediction using AR-GRU model

3.7. Prediction using AR Bi-directional GRU:

The graph obtained below is plotted against predictions over training and testing data overlapped over original values.

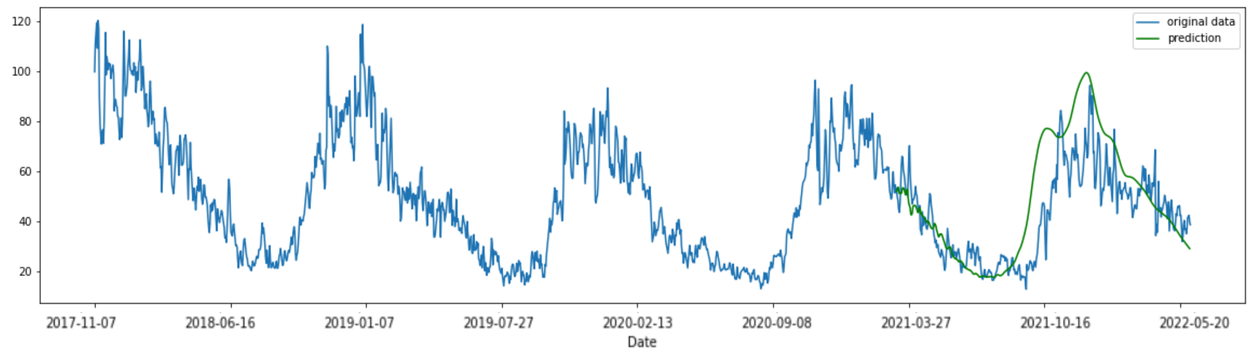


Fig 10. Prediction using AR-BiGRU model

3.8. Prediction using DeepAR:

DeepAR algorithm has also been implemented over the data. The experiment was to understand and compare the performance of various models vs the state of the art DeepAR model. The model performed well over the data but the training time was relatively huge and was increasing exponentially with increase in prediction_window. By using a prediction window of 100 days, the model was trained for 10 epochs and 3minutes per epoch. The results of the DeepAR model are provided in the graph below.

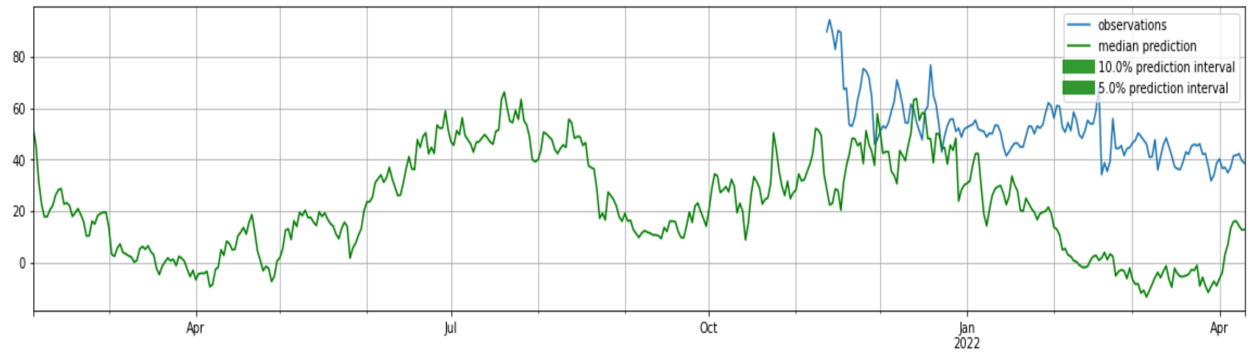


Fig 11. Prediction using DeepAR model

4. CONCLUSION:

A combination of statistical methods and deep learning methods have proven to be successful in time series forecasts. LSTM networks have been able to outperform the traditional forecasting methods. In our analysis we have combined auto regression with LSTM, BiLSTM, GRU and BiGRU. The highest score has been reached by the bi-directional GRU model. According to the findings, the AR-Bi-GRU network performs marginally better than the AR-LSTM network. Additionally, AR-BiGRU has a reduced error value when compared to any other model, which suggests that its adoption can enhance prediction accuracy. This is because the bidirectional GRU processes the time series both chronologically and anti chronologically, capturing patterns that one-direction GRUs could overlook and enhancing the ability of time series to learn features.

5. FUTURE WORK:

In the future, we can improve our model even further and assess it using a larger dataset. We can also use the programme to forecast the concentration of other pollutants. By using techniques like Convolution Neural Network (CNN), the analysis can be expanded further in order to detect the unequal changes occurring in the air pollution data. The relationship between several characteristics can also be evaluated, allowing us to determine whether a concealed parameter will correlate the performance of features that seem to perform differently from the initial glimpse. As both LSTM and GRU demonstrate their significance in prediction, there may as well be a chance that their combined model will be more effective than LSTM and GRU used alone.

6. REFERENCES:

1. Wang, Jingyang, Jiazheng Li, Xiaoxiao Wang, Jue Wang, and Min Huang. "Air quality prediction using CT-LSTM." Neural Computing and Applications 33, no. 10 (2021): 4779-4792.

2. Tiwari, Animesh, Rishabh Gupta, and Rohitash Chandra. "Delhi air quality prediction using LSTM deep learning models with a focus on COVID-19 lockdown." arXiv preprint arXiv:2102.10551 (2021).

3. Athira, V., P. Geetha, Rab Vinayakumar, and K. P. Soman. "Deepairnet: Applying recurrent networks for air quality prediction." Procedia computer science 132 (2018): 1394-1403.

4. G. Dantas, B. Siciliano, B. B. Franc ̧a, C. M. da Silva, and G. Arbilla,“The impact of COVID-19 partial lockdown on the air quality of the city of rio de janeiro, brazil,” Science of the Total Environment, vol. 729,p. 139085, 2020

5. Zhang, Luo, Peng Liu, Lei Zhao, Guizhou Wang, Wangfeng Zhang, and Jianbo Liu. "Air quality predictions with a semi-supervised bidirectional LSTM neural network." Atmospheric Pollution Research 12, no. 1 (2021): 328-339.

6. Song, Xijuan, Jijiang Huang, and Dawei Song. "Air quality prediction based on the LSTM-Kalman model." In 2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), pp. 695-699. IEEE, 2019.

7. Zhou, Xinxing, Jianjun Xu, Ping Zeng, and Xiankai Meng. "Air pollutant concentration prediction based on GRU method." In Journal of Physics: Conference Series, vol. 1168, no. 3, p. 032058. IOP Publishing, 2019.

8. Ahn, Jaehyun, Dongil Shin, Kyuho Kim, and Jihoon Yang. "Indoor air quality analysis using deep learning with sensor data." Sensors 17, no. 11 (2017): 2476.

9. Bekkar, Abdellatif, Badr Hssina, Samira Douzi, and Khadija Douzi. "Air-pollution prediction in smart city, deep learning approach." Journal of Big Data 8, no. 1 (2021): 1-21.

10. Du, Shengdong, Tianrui Li, Yan Yang, and Shi-Jinn Horng. "Deep air quality forecasting using a hybrid deep learning framework." IEEE Transactions on Knowledge and Data Engineering 33, no. 6 (2019): 2412-2424