CLIMATE CHANGE
PREDICTOR USING TIME SERIES FORECASTING

An Engineering Project in Community Service

Phase – II Report

Submitted by

1. Aabir Datta - 19BCE10062
2. Uday Agarwal - 19BAI10147
3. Abdul Raziq Khan - 19BCE10135
4. Parv Bhargava - 19BAI10116
5. Aryan Tandon - 19BAI10148
6. Vimal Tiwari - 19BCG10054
7. Govit Khasare - 19MIM10094
8. Mriganka Shekhar Das - 19BOE10053

in partial fulfillment of the requirements for the degree of

Bachelor of Engineering and Technology

VIT Bhopal University
Bhopal
Madhya Pradesh

April, 2022

Bonafide Certificate

Certified that this project report titled **"CLIMATE CHANGE PREDICTOR USING TIME SERIES FORECASTING"** is the bonafide work of "19BCE10062 Aabir Datta, 19BAI10147 Uday Agarwal, 19BCE10135 Abdul Raziq Khan, 19BAI10116 Parv Bhargava, 19BAI10148 Aryan Tandon, 19BCG10054 Vimal Tiwari, 19MIM10094 Govit Khasare, 19BOE10053 Mriganka Shekhar Das**"** who carried out the project work under my supervision.

This project report (Phase II) is submitted for the Project Viva-Voce examination held on …………..

PAVAN KUMAR

Supervisor

CONTENTS

INTRODUCTION

One of the major problems of the 21st century is undoubtedly **Climate Change** and with the sudden advent and breakout of disastrous climate hazards in India and many such countries, it has become a growing public concern and a societal affair posing serious threat. Hence we take steps in understanding climate change and make certain vital predictions which are both effective and efficient. We understand and explore the performance of climate through the dataset of quite a sizeable quantity to make predictions using a neural network via a top down approach model and big data of global mean monthly temperature and rainfall. We generate graphical images of the monthly temperature and rainfall for over 30 years. The neural network model successfully works to predict the rise and fall of temperature for the upcoming 10 years, and also estimates the amount of rainfall for the next 5 years. In addition to this the prediction accuracy differs among climatic zones and temporal ranges. Thus we introduce a forecasting method that works along with conventional physics and mathematical models to construct a highly precise model for making predictions.

1.1 & 1.2 MOTIVATION & OBJECTIVE

The recent reports from the Intergovernmental Panel on Climate Change throw light on the issue of extreme weather events which includes several hazards that contribute to environmental and societal risks. Some of them include severe heat waves, droughts, flooding caused due to heavy rainfall followed by storm surge, compound fire conditions i.e. a combination of hot, dry and windy conditions in varying locations causing immense amounts of distress leading to uncanny and helpless situations. Thus to produce a better understanding of the current climatic scenario and also being well ahead of the future situations we decided to work on a climate change predictor that not just understands, visualizes and analyzes the current situation but also lets us look forward to predict the upcoming years and how variations in climate will indeed affect temperature and precipitation. This will be assisting in setting up precautionary measures for the future.

LITERATURE REVIEW

We have previously observed concerns regarding climate change and also the need for predictions in climate change for over more than two decades. With the advent of artificial intelligence and it being the path to all kinds of solutions these days there have been a number of prediction systems for climate change over the due course of time. Hence we could see how well these predictions are currently being used up by meteorological departments but all models are subject to improvements. Some of the models on these systems which we referred upon to build our enhanced version of the same are discussed below.

Peter Turner [1] has discussed how time is useful in comparing things and how it is the basic building block in prediction of certain outcomes. Another major concept that is discussed is that of the time series which is a set of repeated measurements of the same phenomenon taken sequentially over a specific period of time. Four main components of time series namely 'Trend', 'Seasonality', 'Cyclicity', 'Irregularity'. Climate change refers to the long term shifts in temperature and weather patterns. In this article the climate change has been predicted using time series.

Two datasets, one from NASA giving an estimate of global surface temperature change ,the other from the World Bank giving an estimate of CO2 emissions in metric tons per capita. The temperature data represents temperature anomalies per month and season and the CO2 data gives the average emission per person. Firstly the data of temperature is wrangled. Data wrangling is the process of transforming and mapping data from the raw dataset and pushing it into another data frame. In doing so we get all the intecracy of the details for our data for making the necessary setups for the model. With this he could also determine the missing data and worked on getting the suited technique to manipulate the data and fill in the missing values. Then he worked on resampling the data to a different frequency to check the best possible variation for making the predictions. He also worked on CO2 emission calculations to get the predictions to a higher accuracy. He carried forward with slicing the data and keeping it ready for plotting and visualization of all sorts to obtain some visible outputs of the same. Finally using Granger Causality he performed the time series correlation to develop the trend and create the prediction system delivering accurate results. The functional ARIMA model creates a completely clear picture of the current scenario of the climate and also can allow the meteorological departments to take up necessary measures for the same. We too could get some critical insights about SARIMA from his functioning model.

Next up, in Trend analysis of climate time series [2], a review of the methods research paper, we observed that the authors have discussed how statistics has developed methods to quantify the warming trend and detect change points. Statistics serve to place error bars and other measures of uncertainty to the estimated trend parameters.

The application of the state-of-the-art statistical methods to the GISTEMP time series of global surface temperature reveals an accelerated warming since the year 1974. It shows that a relative peak in warming for the years around World War II may not be a real feature but a product of inferior data quality for that time interval.

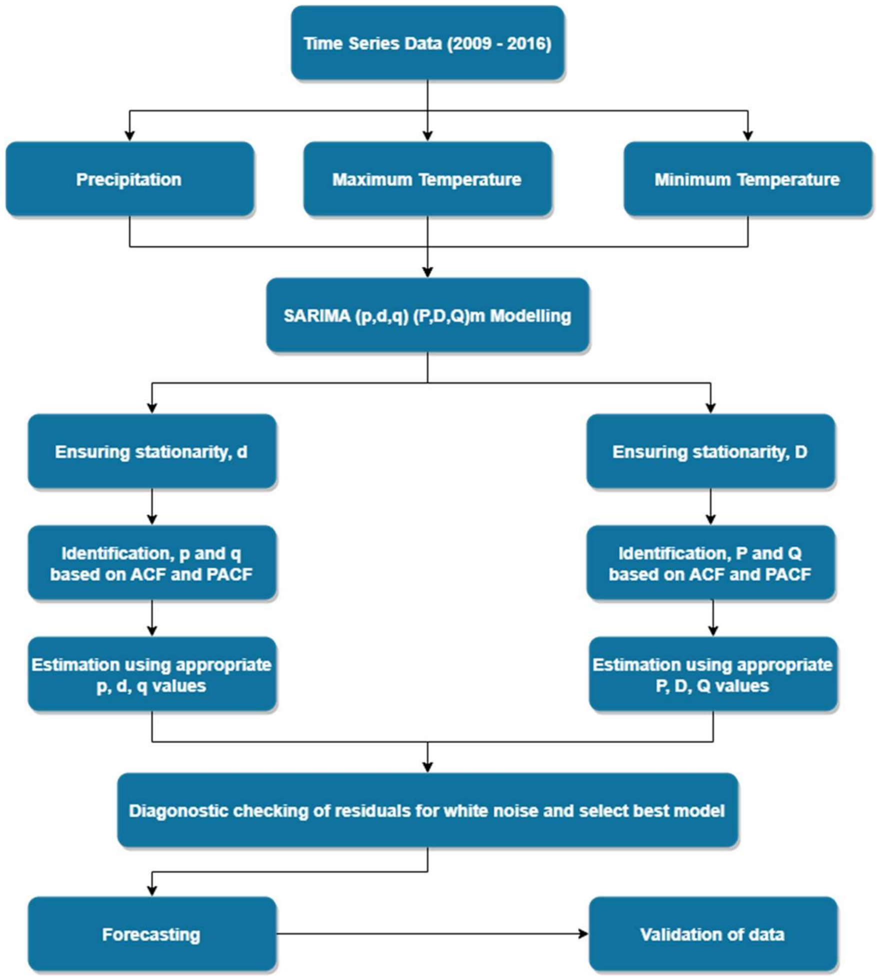
The recommendation therefore is that interval selection should be objective and oriented on general principles. The application of statistical methods to data has also a moral aspect.

The concept of linear regression tends to guide the flow of paper. The linear regression describes Xtrend(i) by means of two parameters, namely the intercept, $\beta 0$, and the slope, $\beta 1$. The model is "on the process level".
The ordinary least-squares (OLS) estimation minimizes the sum of squares of differences between data and the linear fit.
But in this paper, the linear model is not well suited to describe the trend for the Surface temperature analysis (GISTEMP) time series. The intercept and slope estimates should therefore be interpreted with caution.

Hence although the paper addresses regression to the core level and covers statistical concepts to draw conclusions for the predictions, it lacks a constructive model for making accurate predictions as we know that linear regression is graph dependent and to get accurate level results the plotting of the graph needs to be super accurate which theoretically might sound feasible but in the practical grounds does show some faults. On the contrary non linear regression includes too many calculations and the system needs to undergo heavy mathematical load to produce results. Thus predictions take immense amounts of time and after a certain point in time the system is unable to make those calculations and therefore inaccurate results is what we obtain.

WORKING

3.1 SYSTEM ARCHITECTURE



3.2 WORKING PRINCIPLE

The precipitation and temperatures (maximum and minimum) data are considered for the study and prepared for the analysis as monthly means. Once the data files are prepared, the ARIMA model needs to be identified. In this study, we have tried to fit separate SARIMA models to precipitation and temperature time series. So, we have one SARIMA model that fits the precipitation time series and one that fits the temperature time series.

Seasonal Autoregressive Integrated Moving Average, SARIMA or Seasonal ARIMA, is an extension of ARIMA that explicitly supports univariate time series data with a seasonal component. It adds three new hyperparameters to specify the autoregression (AR), differencing (I) and moving average (MA) for the seasonal component of the series, as well as an additional parameter for the period of the seasonality. The difference between ARIMA and SARIMA is about the seasonality of the dataset. If the data used is seasonal, like it happens after a certain period of time, then we will use SARIMA. Just as we know the weather datasets are seasonal in nature.

In addition to the working SARIMA model we have added an enhanced library named JAX in place of NumPy. This is the one which improves and betters any Machine Learning model. Its improved features help carry out the same actions at a relatively very high speed. Thus the model which was initially being trained for over 30 minutes now got working in less than 3 minutes.

3.3 RESULTS AND DISCUSSIONS

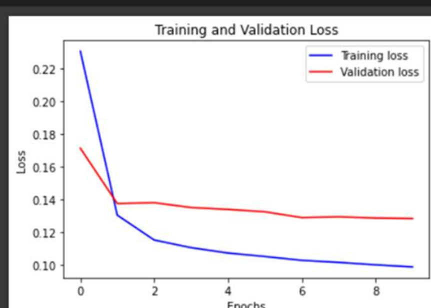
The following results show a detailed output of the trained data.

```
#Training
inputs = keras.layers.Input(shape=(inputs.shape[1], inputs.shape[2]))
lstm_out = keras.layers.LSTM(32)(inputs)
outputs = keras.layers.Dense(1)(lstm_out)

model = keras.Model(inputs=inputs, outputs=outputs)
model.compile(optimizer=keras.optimizers.Adam(learning_rate=learning_rate), loss="mse")
model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 120, 7)]          0

 lstm (LSTM)                 (None, 32)                5120

 dense (Dense)               (None, 1)                 33

=================================================================
Total params: 5,153
Trainable params: 5,153
Non-trainable params: 0
_________________________________________________________________
```

We visualized the loss with the function below. After one point, the loss stops decreasing and the graph become flat

```
def visualize_loss(history, title):
    loss = history.history["loss"]
    val_loss = history.history["val_loss"]
    epochs = range(len(loss))
    plt.figure()
    plt.plot(epochs, loss, "b", label="Training loss")
    plt.plot(epochs, val_loss, "r", label="Validation loss")
    plt.title(title)
    plt.xlabel("Epochs")
    plt.ylabel("Loss")
    plt.legend()
    plt.show()


visualize_loss(history, "Training and Validation Loss")
```



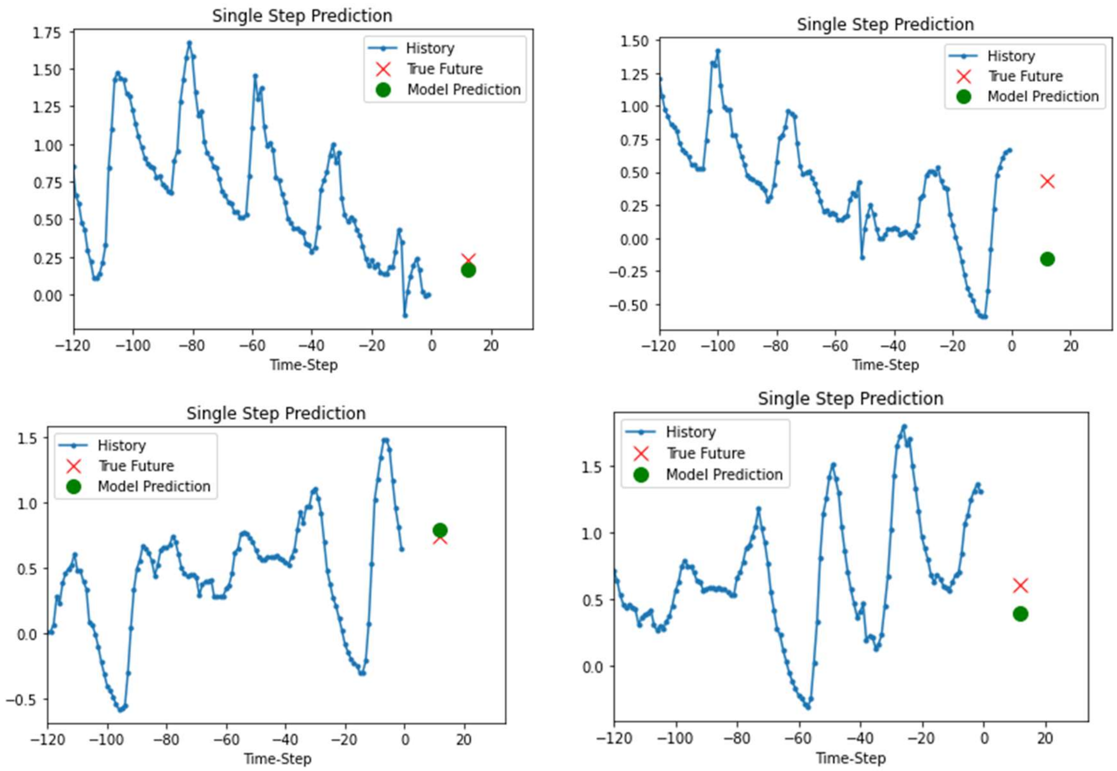


Figure: Prediction using First model

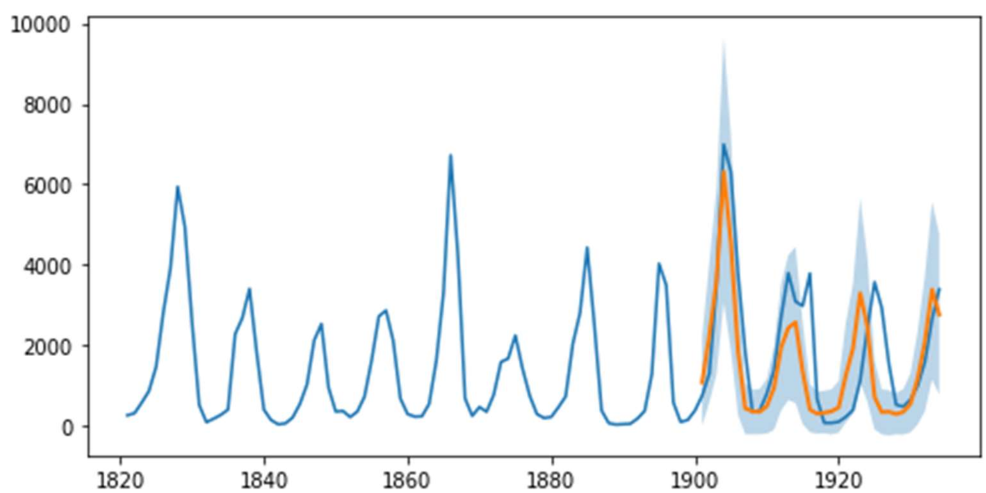


Figure: Prediction using Enhanced model

3.4 MY CONTRIBUTIONS

- I have worked on Validation of the Forecasts and got to the following observation
  We could check if a better representation of our true predictive power can be obtained using dynamic forecasts. In this case, we only use information from the time series up to a certain point, and after that, forecasts are generated using values from previous forecasted time points.
  In the code chunk below, we specify to start computing the dynamic forecasts and confidence intervals from May 2017 onwards.

```
In [25]: pred_dynamic = results.get_prediction(start=pd.to_datetime('2017-05-19'), dynamic=True, full_results=True)
         pred_dynamic_ci = pred_dynamic.conf_int()
```

Figure: Make the Forecasts

Once again, we plot the real and forecasted values of the average daily temperature to assess how well we did:

```
In [26]: ax = one_step_df.T_mu_actual['2015':].plot(label='observed', figsize=(20, 15))
         pred_dynamic.predicted_mean.plot(label='Dynamic Forecast', ax=ax)

         ax.fill_between(pred_dynamic_ci.index,
                         pred_dynamic_ci.iloc[:, 0],
                         pred_dynamic_ci.iloc[:, 1], color='k', alpha=.25)


         ax.set_xlabel('Date')
         ax.set_ylabel('Temperature (in Celsius)')
         plt.ylim([-20,30])
         plt.legend()
         plt.show()
```

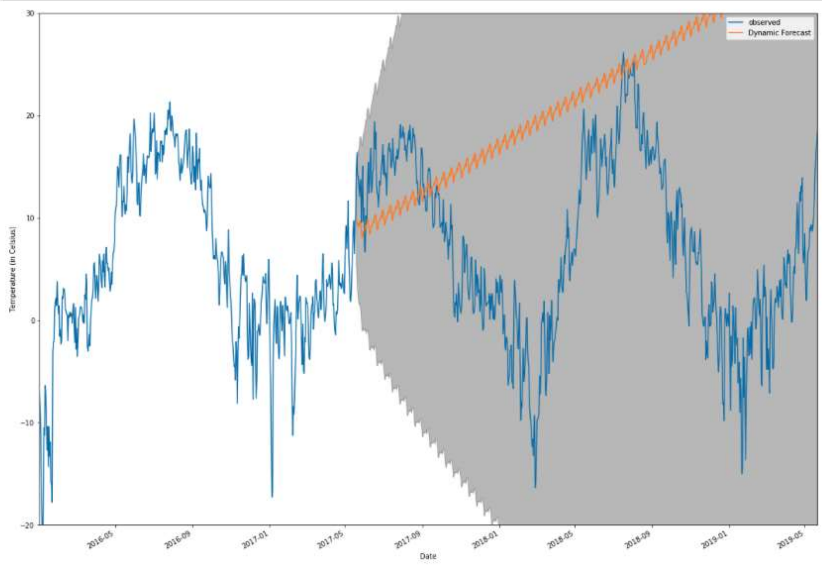


Figure: Observed vs. Forecast

Once again, we quantify the predictive performance of our forecasts by computing the RMSE:

```
In [21]: # Extract the predicted and true values of our time series
         y_forecasted = pred_dynamic.predicted_mean
         y_truth = one_step_df.T_mu_actual['2017-05-19':]

         # Compute the mean square error
         mse = sqrt(MSE(y_truth, y_forecasted).mean())
         print('The Root Mean Squared Error of our forecasts is {}'.format(round(mse, 2)))

         The Root Mean Squared Error of our forecasts is 20.04
```

Figure: The RMSE value

The predicted values obtained from the dynamic forecasts yield an RMSE of 20.04. This is significantly higher than the one-step ahead, which is to be expected given that we are relying on less historical data from the time series.

- I also have contributed to core code for JAX implantation of new models. JAX is an automatic differentiation (AD) toolbox developed by a group of people at Google Brain and the open source community. It aims to bring differentiable programming in NumPy-style onto TPUs. On the highest level JAX combines the previous projects XLA & Autograd to accelerate linear algebra-based projects.
- In our Model we decreased the training time from 30 mins to 3 mins. That's a great improvement in training the model.
- The implementation of our Time Series and related materials can be found here at my GitHub Repository
- I have previously worked with JAX which you can find here, and I found it to be the future of machine learning as an API over Numpy as JAX provide DeviceArrays which are immutable over the traditional NDArrays provided by Numpy which make the calculation here much faster as compared to Numpy and hence can be used to accelerate our model performance significantly as our project here require continuous crunching of numbers for the results.
- I have also worked hard on preparing the overall presentation for our project and on the collection of content and write ups for the presentation
- I have also helped on creating this report and worked on content and structuring of this report.
- Overall it was a group effort and I am happy that each and every one of us contributed to the project as a team.

CONCLUSION

Thus, we could implement a seasonal SARIMA model in Python and worked on the enhancement of the code using JAX. We made extensive use of the pandas and statsmodels libraries and showed how to run model diagnostics, as well as how to produce forecasts of the temperature. We could also reduce the training time of the model to a great extent and worked on increasing the efficiency of the SARIMA model with highly accurate results.

Here are a few other things we can try additionally to our coded model:

- Change the start date of our dynamic forecasts to see how this affects the overall quality of your forecasts.
- Try more combinations of parameters to see if we can improve the goodness-of-fit of the model.
- Select a different metric to select the best model. For example, we used the AIC measure to find the best model, but we could seek to optimize the out-of-sample mean square error instead.

REFERENCES

[1]https://towardsdatascience.com/time-series-analysis-and-climate-change-7bb4371021e

[2]https://www.sciencedirect.com/science/article/pii/S0012825218303726

[3]https://medium.com/@llmkhoa511/time-series-analysis-and-weather-forecast-in-python-e80b664c7f71#:~:text=Time%20series%20data%20are%20simply,price%2C%20weather%20data%2C%20etc.&text=Make%20Weather%20Forecasts%20using%20the%20SARIMAX%20model

[4]https://keras.io/examples/timeseries/timeseries_weather_forecasting/