Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2020) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

2020 International Conference on Identification, Information and Knowledge in the Internet of Things (IIKI2020)

Air Temperature Forecasting using Traditional and Deep Learning Algorithms

Chengsi Li[a*], Mengyisong Zhao[b*], Yilong Liu[c*], Fangzhou Xu[d*]

[a]*University of Toronto, Toronto, Canada*
chengsi.li@mail.utoronto.ca
[b]*University of Sheffield, Sheffield, The United Kingdom*
zhaomengyisong@gmail.com
[c]*Beijing University of Posts and Telecommunications, Beijing, China*
liuyilong@bupt.edu.cn
[d]*University of Manitoba, Winnipeg, Canada*
xianivian@gmail.com

***These authors have contributed equally to this work**

Abstract

This paper intends to find the appropriate model for air temperature forecasting on German Beutenberg area based on the previous hourly air temperature data. We gathered and preprocessed the data from Max Planck Institute for Biogeochemistry (MPIB). Then, we proposed and compared two traditional machine learning models XGBoost and Polynomial Regression against one deep learning model LSTM to achieve the prediction. Our experimental results indicate that LSTM can be accepted as the most appropriate model for predicting air temperature. This model has shortest running time and can even make predictions with discontinuous time features, it achieved 98.4% R-squared and 1.04 RMSE. In addition, the other two commonly used traditional machine learning models can also perform well in predicting continuous air temperature features.

Keywords: Artificial Intelligence, Machine Learning, Recurrent Neural Networks, Weather Forecasting, Temperature

1. Introduction

• Weather forecasting and inferencing have been an omnipresent challenge throughout our daily life. In meteorology, successful and accurate weather condition predictions can benefit farming, utility companies and various stakeholders around this life cycle. For instances, the weather condition is an important impact indicator

Peer-review under responsibility of the scientific committee of the 2020 International Conference on Identification, Information and Knowledge in the Internet of Things

of agriculture, accurate weather forecasts can let farmers prepare activities. As for utility companies, any sudden change of weather conditions, like heavy rain, snow, or wind, may seriously affect the companies' outdoor experiments. In those cases, if the future weather can be predicted, then the activities can be organized and prepared in advance, particularly on the occasions that may reflect in their survivals. [1].

- One of the main challenges in weather forecasting is that extreme weather events have shown an increasing trend and becoming more complex and changeable in time and space. Under this global circumstance, making accurate weather forecasts becomes even more difficult [2-3]. Moreover, with the spatial and temporal resolution of various meteorological observation data increases, this domain could still be facing the problem of insufficient ability to handle the massive data volume and inadequate uses of technology advancement.
- With the developments and prosperities of artificial intelligence and machine learning technologies, many researchers are now able to explore hidden patterns in the weather prediction area. In recent years, various data mining and deep learning model have been proposed to tackle the weather forecasting problem [1-21].
- The work of weather forecasting usually needs to deal with massive data as input parameters, such as wind direction, wind speed, humidity, rainfall, temperature etc. Specifically, the different machine learning methods have been used in this domain, which including the traditional model like Naïve Bayes [1] and support vector machines [4-5]. Moreover, time-serious analysis ARIMA model has also been conducted [6-8] There are also deep learning models been applied in weather forecasting. Quite a lot of researchers employed Artificial Neural Networks [9-17] and Recurrence Neural Networks [18-19] into this area. Some even used Genetic algorithms to make weather predictions [20].
- This paper compares the performance of Long-Short Term Memory (LSTM) Model of Recurrence Neural Networks (RNNs) with traditional machine learning methods XGBoost and Polynomial Regression for air temperature forecasting using the previous data at German Beutenberg weather station. Our objective is to predict specific hourly air temperature based on the prior three days of hourly air temperature data. The rest of this paper is structured as follows: Section 2 describes the data collection progress and preprocessing steps. Section 3 explains the three proposed model into detailed. Section 4 demonstrates the experiment results and discussion. Finally, Section 5 is the conclusion and the expectation of future work.

2. DATA COLLCETION AND PREPROCESSING

The raw data has been acquired from a German weather website, recorded by the Max Planck Institute for Biogeochemistry (MPIB). MPIB engage in "understanding of how organisms exchange basic resources with the living environment, and how does the environment influence and respond to global climate and environmental change [21]. The used dataset collected by Beutenberg weather station contains the data of recent ten years beginning on 1st January 2010 to 5th July 2020. This dataset contains 21 different features such as air temperature, relative humidity and atmospheric pressure, and these data were collected every 10 minutes. The raw data shows in Fig 1. The raw data include 555,782 rows and 21 columns.

Date Time	p (mbar)	T (degC)	Tpot (K)	Tdew (degC)	rh (%)	VPmax (mbar)	VPact (mbar)	VPdef (mbar)	sh (g/kg)	H2OC (mmol/mol)	...	wv (m/s)	max. wv (m/s)	wd (deg)	rain (mm)	raining (s)	SWDR (W/m*2)	PAR (µmol/m*2/s)
2018-01-01 00:10:00	975.22	9.59	284.79	4.85	72.20	11.96	8.63	3.32	5.53	8.85	...	3.40	5.29	180.4	0.0	0	0.00	0.00
2018-01-01 00:20:00	975.16	9.82	285.02	4.81	70.90	12.15	8.61	3.53	5.51	8.83	...	3.38	5.05	175.1	0.0	0	0.00	0.00
2018-01-01 00:30:00	975.03	9.91	285.12	4.75	70.20	12.22	8.58	3.64	5.49	8.80	...	3.76	6.97	179.1	0.0	0	0.00	0.00

Fig 1. Raw data directly collect from Beutenberg weather station

However, this row dataset includes some redundant and missing value. The missing values are the times that there is no record and they are hard to find. First, we find out if there are 10 minutes time difference between each row and mark the rows with a longer or shorter time difference. The rows with a difference larger than 10min are treated as missing rows, and the rows with difference smaller than 10min are treated as redundant rows.

Then, the redundant rows are removed, and the values in missing rows are calculated using the gradual average using iterative computations.

Then, the dataset has been split into two parts, the training set includes the air temperature from 2010.1.1 to 2018.1.1, and the test set includes the same information from 2018.1.1 to 2020.7.5.

Since we are interested in predicting the air temperature three days in the future at the specific hour based on looking at the trend of past seventy-two hours of data. The datasets are resampled into hours by grouping six 10 minutes rows together and calculate the mean value. Finally, we created a matrix with 73 features. The first 72 columns of data stand for the 72 continuous hours of air temperature and the final column is the target feature, which presents the temperature at the 73rd hour for each entry. Fig 2 shows the data after cleaning and preprocessing. After the data preprocessing is finished, there are 70129 rows (time start from 2020-01-01 00:00:00 to 2018-01-01 00:00:00) and 73 columns (the air temperature starting from time 2020-01-01 00:00:00 to 2020-01-03 00:00:00) in the training set. We will test the model performance by using the air temperature data start from 2018-01-01 01:00:00 to 2020-07-05 23:00:00, this includes 22007 rows. In short, the dataset split can be defined as 80% (training) and 20% (test).

Date Time	T+0	T+1	T+2	T+3	T+4	T+5	T+6	T+7	T+8	T+9	...	T+63	T+64	T+65
2010-01-07 00:00:00	17.815000	16.908333	16.213333	15.348333	14.790000	15.893333	18.090000	19.403333	21.596667	24.176667	...	13.903333	13.358333	15.055000
2010-01-07 01:00:00	16.908333	16.213333	15.348333	14.790000	15.893333	18.090000	19.403333	21.596667	24.176667	26.226667	...	13.358333	15.055000	13.195000
2010-01-07 02:00:00	16.213333	15.348333	14.790000	15.893333	18.090000	19.403333	21.596667	24.176667	26.226667	27.120000	...	15.055000	13.195000	12.726667

Fig 2. Final dataset has been used for model training..

3. Proposed Method

A. Time Series Predicting from History

The air temperature data we gathered is a time-series type of data. The simplest and straightforward way for time series prediction is modelling through historical data. So, the target temperature (predicting temperature) can be defined as the function shown below:

$$T_t = f(T_{t-1}, T_{t-2}, T_{t-3}, \ldots\ldots T_{t-n})$$

T stands for the temperature, and t is the time in an hourly presentation. We choose to predict the air temperature at the hour after 3 days, so n should be 72 in this case.

Each model used to conduct forecasting is explained in detail as follows.

B. eXtreme Gradient Boosting (XGBoost)

The full name of XGBoost is eXtreme Gradient Boosting, it is an optimized distributed gradient boosting model with more efficient and flexible ability. Through integrating many simple tree models, XGBoost becomes a stronger classifier. XGBoost is an efficient implementation of the GB algorithm, it displays a regularization term in the objective function. Fig 3 below shows the structure of XGBoost.

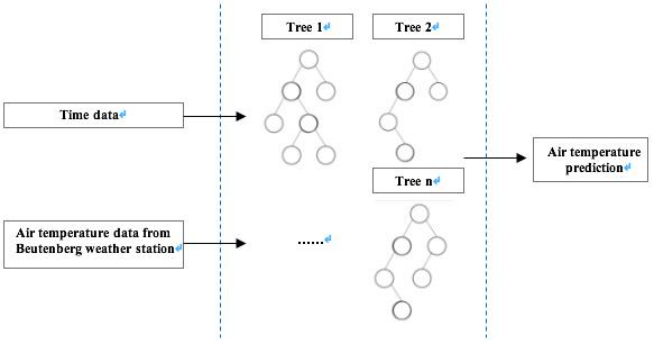

Fig 3. XGBoost Structure of n_estimators
Source from: [19]

The objective function of XGBoost model have further optimization, which shows below:

$$Objection\ Function: L(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k)$$

$$loss\ Function: \sum_i l(\hat{y}_i, y_i)$$

$$Regularisation: \sum_k \Omega(f_k)$$

y_i is true label, $\hat{y}_i$ is the predicted label, l is a loss function, f_k represents the structure of the kth tree, Ω is the regularization.

GB algorithm uses the first derivative of loss function to calculate the pseudo-residual to learn to generate fm(x), XGBoost not only involves the first derivative, but also uses the second derivative. The regularization of XGBoost model contains two parts: one is to increase the number of leaf nodes T of the tree, which is used to control the complexity of the tree and achieve the effect of pruning; The other part is the squared weight and regular term of each tree's leaf node w, which can avoid over-fitting as far as possible. Through the above target function, each subtree of XGBoost will tend to learn the simpler tree. In addition, when the parameter of the regular term is 0, the target expression will be reduced to the traditional Gradient Tree boosting model. XGBoost through Shrinkage can be predicted for each tree, acting as a learning rate, reducing the model's reliance on individual trees, and enhancing the generalization capacity of the model [22].

C. Polynomial Regression

Polynomial Regression is a method that the regression variable of its regression equation is polynomial. It is also a specific kind of Linear Regression because the process of using the regression equation to estimate the unknown parameter from the data can be seen as a linear regression. The general function of the Polynomial Regression is:

$$\hat{y} = b_0 + b_1 x + b_2 x^2 + ... + b_m x^m$$

Linear regression focuses on the problem between one independent variable and one dependent variable. In statistics, Linear Regression is a specific kind of regression analysis that uses the least square function called linear regression equation to build a model using the relationship between independent and dependent variables. In the application of Linear Regression, the model is built with the dataset using linear prediction functions, and other unknown parameters are also estimated from the data. The most commonly used method to build a linear regression model is the conditional mean of the dependent variable is an affine function of the independent variable, in which the y is given by x. Linear Regression mainly focuses on the conditional probability distribution of the dependent value of a given independent value, but not the joint probability distribution of the two values, just like many forms of other regression analysis.

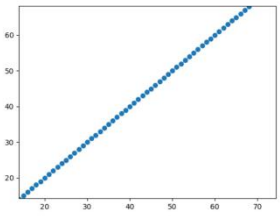

Fig. 4 linear relationship

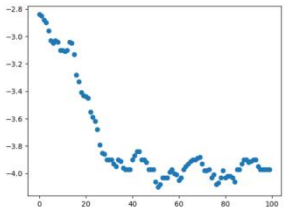

Fig.5 first one hundred air temperature data

When the dataset of the independent variable can be concluded as a linear function of the dependent variable (Fig.4, where the data points are generated as a linear relationship), we can use the Linear Regression.

Fig 5 is the plot of the first 100 temperature data in the dataset. It is apparent that the fitting curve of the plot is not a linear line, so we use the Polynomial Regression here. The basic logic of Polynomial Regression is similar to the Linear Regression, except that the independent variables are more than one and the weight of impact that different variables may have to the dependent variable are different.

Whereas in many real-life situations, the dependent variable is changing based on multiple independent variables. Just like the circumstance in this paper, the plot of the result cannot be concluded as a linear function. Polynomial Regression can be used to deal with the dataset that has no linear feature. The strength of this method is that it covers the weakness of the Linear Regression to fit the function which has no linear separability. It is more flexible to use and able to handle more complicated tasks. However, Polynomial Regression must have fully control of the element variables, including the exponential of different variables. The model also needs to have a thorough design and the acquirement of the better exponential of the variable requires some prior knowledge of the test set. It is possible that inappropriate choices of the exponential may cause over-fitting.

D. Long-Short Term Memory (LSTM) Model

LSTM is a type of RNN (Recurrent Neural Network). It is capable of handling gradient vanishing problem and gradient exploding problem when training with long sequence of data.

Below is the structure of LSTM:

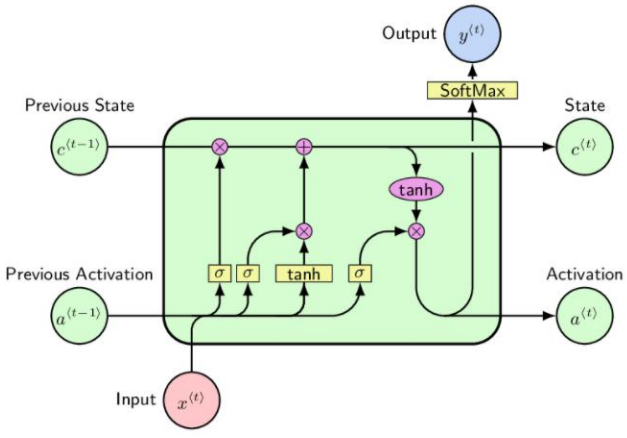

Fig. 6 LSTM Structure

Source from: https://www.researchgate.net/figure/The-LSTM-cell-internals_fig1_334360853

Fig 6 above shows the structure of LSTM model, Where:

x^t is the input

y^t is the output

c^{t-1} is the cell state input received from the previous cell

c^t is the cell state output that will be given to the next cell

a^{t-1} is the hidden state input received from the previous cell

a^t is the hidden state output that will be given to the next cell

Compared with normal RNNs which only uses hidden state as transmission state, LSTM has two transmission states, a cell state c^t and a hidden state a^t (cell state c^t in LSTM performs similar function as the hidden state in RNN). Usually, c^t in LSTM does not change a lot during transmission, such that a cell's output c^t is obtained by the combination of the original input c^{t-1} from the previous cell and some new input data from current cell. However, the hidden state output h^t changes faster, so its value may be very different between cells.

LSTM obtains result by going through three gates: forget gate, input gate and output gate. Each gate in LSTM is controlled by their own gate control signals. The network uses these gate control signals to control transmission, help the network to memorize long-term data and forget unnecessary information. Each gate control signal is obtained by first joining x^t and a^{t-1} together and multiply by the corresponding gate weight matrices, then put the result into a sigmoid layer to output a number between 0 and 1.

Forget Signal: $z^f = \sigma(W^f \cdot [a^{t-1}, x_t])$

Input Signal: $z^i = \sigma(W^i \cdot [a^{t-1}, x_t])$

Output Signal: $z^o = \sigma(W^o \cdot [a^{t-1}, x_t])$

These signals are multiplied with data every time they come out of the gates in order to control the proportion memorized or forgotten.

In the forget gate, the network forgets unnecessary parts of the inputs from the previous cell. Specifically, it uses the calculated signal z^f to decide which part of c^{t-1} needs to be forgotten.

In the input gate, the network selects which part of the input will be memorized and written into cell state output. The gate first uses a *tanh* layer to transform the input data into values between -1 and 1.

$$z = tanh(W \cdot [a^{t-1}, x_t])$$

Then it uses the gate control signal z^i to select the part that will be memorized. Next, the cell state output c^t can be calculated by combining the results from the previous two gates together.

$$c^t = z^f \odot c^{t-1} + z^i \odot z$$

Finally, the output gate decides which will be the current cell output. First, the gate puts the calculated c^t into a *tanh* layer to rescale the values. Then, it uses the control signal z^o to select which part will be the hidden state output a^t.

$$a^t = z^o \odot \tanh(c^t)$$

When it is needed, the cell output y^t can be obtained by transforming the hidden state output a^t, just like normal RNNs.

4. Experiments

The data of air temperature is periodically ordered. Fig 7 below shows the change in air temperature between 2010 and 2020. The fluctuant trends in temperature is caused by the variation of seasons.

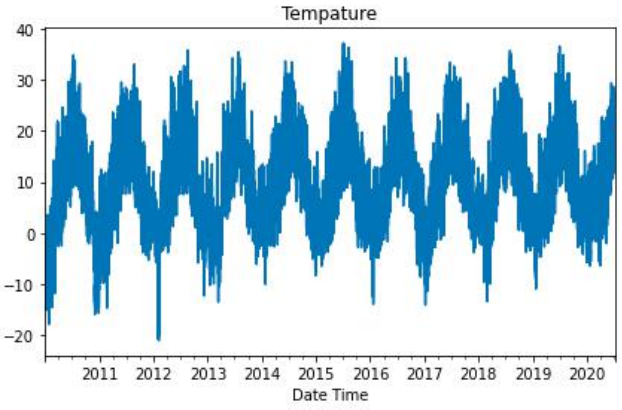

Fig. 7 Visualisation of air temperature from 2010-2020

The result of this project was carried out with python 3.7, where Sklearn and Keras were the main packages used. In order to find the model with best performance and efficiency, we compared the traditional commonly used machine learning and deep learning algorithms. The RMSE (rooted mean squared error) and R2 (R-squared) have been used to evaluate the model, and all parameters not shown in the tables are default. TABLE I shows the parameters and results obtained from XGBoost. The model achieved 98.07% R2, with 1.16 RMSE. And it took the model almost 12 minutes to calculate the result. TABLE II demonstrates the result obtained by Polynomial Regression. The best performance came when the degree of the polynomial features was set to 2. The model achieved 98.29% R2 and 1.09 RMSE, which is slightly better than the XGBoost model. Also, the model only took 3 minutes to get the results. Since the changes in air temperatures are more gentle and smooth instead of sharp and fluctuant within short time periods, the XGBoost model with high generalization ability was able to predict a quite successful result. Polynomial Regression is more straightforward. Through ascending dimensions and adding new features, it can better fit with high-dimensional data.

The model that has the best performance for predicting air temperature is LSTM. TABLE III shows its parameters and results. With the parameters in TABLE III, the model has achieved the best result with 98.40% R2 and 1.04 RMSE, which means that, on average, the predicted temperature error should be within 1.04. The deep learning model's ability to memorize long-term potential data pattern and the periodically fluctuant temperature trends could be the main reasons why the LSTM's prediction result was quite successful. More importantly, as all the models' results are quite close, LSTM model has the shortest running time, only 1m40s. Since it also obtained the highest accuracy and lowest error, LSTM can be treated as the best performance model for air temperature forecasting among these three models. Fig 8 below shows the comparison of the results between the model predicted air temperature and the actual air temperature. It could be seen that LSTM achieved fairly accurate forecasting.

TABLE I XGBOOST MODEL PARAMETER SETTING AND RESULT

XGBoost	
Parameters	**Values**
Estimators	500
Max Depth	4
Min Samples Split	5
Learning Rate	0.1
Loss	Least Squares Regression
Model Results	**Values**
RMSE	1.1556
R2	0.9807
Running time	11.27m

TABLE II POLYNOMINAL REGRESSION MODEL PARAMETER SETTING AND RESULT

Polynomial Regression	
Parameters	**Values**
Degree	2
Model Results	**Values**
RMSE	1.0867
R2	0.9829
Running Time	3m

TABLE III LSTM MODEL PARAMETER SETTING AND RESULT

LSTM	
Parameters	**Values**
Epochs	25
Batch Size	108
Dropout	20%
Input Shape	6 timestamps and 72 features
LSTM Layers	2
Neurons in LSTM Layer	96
Neurons in Output Layer	6
Model Results	**Values**
RMSE	1.0425
R2	0.9840
Running Time	1m40s

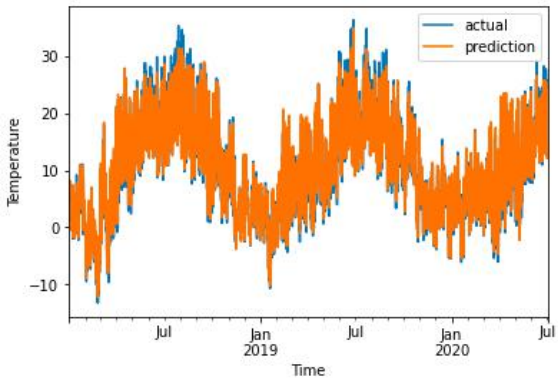

Fig 8. LSTM actual and prediction result comperasion

5. Conclusions

This paper examined three different machine learning algorithms for predicting air temperatures. The experiment result suggests that LSTM can be taken as the most appropriate algorithm when forecasting air temperature. LSTM model has achieved 98.40% R-squared score with 1.04 rooted mean squared error, a fairly accurate prediction. This model is more prominent at its running time, as it can produce the result much faster than the other two models. The paper also indicated that traditional machine learning models, particularly XGBoost and Polynomial Regression, can also perform well in predicting continuous air temperature features.

In the future, we would like to add more weather features such as relative humidity, air pressure, wind direction to examine whether these features could improve the prediction results. More interestingly, it is also worth to explore the performance of other various deep learning algorithms within this weather forecasting domain.

References

[1] M. Biswas, T. Dhoom, and S. Barua, “Weather Forecast Prediction: An Integrated Approach for Analyzing and Measuring Weather Data,” International Journal of Computer Applications, vol. 975, pp. 8887,2018.

[2] N. Krishnaveni, and A. Padma. "Weather forecast prediction and analysis using sprint algorithm." Journal of Ambient Intelligence and Humanized Computing, pp. 1-9, 2020.

[3] K. Almgren, S. Alshahrani, & J.Lee, “Weather Data Analysis using Hadoop to Mitigate Event Planning Disasters”, 2015.

[4] Y. Radhika and M. Shashi,"Atmospheric temperature prediction using support vector machines," International Journal of Computer Theory and Engineering, vol. 1, no. 1,pp. 1793-8201, 2009.

[5] N. I. Sapankevych and R. Sankar, “Time series prediction using support vector machines: A survey," Computational Intelligence Magazine, IEEE, vol. 4, no. 2, pp. 24-38, 2009.

[6] S. Kothapalli, and S. G. Totad. "A real-time weather forecasting and analysis." In 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), pp. 1567-1570. IEEE, 2017.

[7] M. Tektaş, “Weather forecasting using ANFIS and ARIMA models.” Environmental Research, Engineering and Management vol. 51, no. 1, pp. 5-10, 2010.

[8] M. Murat, I. Malinowska, M. Gos, and J. Krzyszczak, “Forecasting daily meteorological time series using ARIMA and regression models,” International agrophysics, vol. 32 no. 2, 2018.

[9] S. Ghimire, R. C. Deo , N. J. Downs, and N. Raj, “Global solar radiation prediction by ANN integrated with European Centre for medium range weather forecast fields in solar rich cities of Queensland Australia,” Journal of cleaner production, vol. 216, pp. 288-310, 2019.

[10] N. Do Hoai, K. Udo, and A. Mano, “Downscaling global weather forecast outputs using ANN for flood prediction, ”Journal of Applied Mathematics, 2011.

[11] C. Voyant, M. Muselli , C. Paoli , and M. L. Nivet, “Numerical weather prediction (NWP) and hybrid ARMA/ANN model to predict global radiation,” Energy, vol.39, no. 1, pp. 341-355, 2012.

[12] S. Traore, Y.Luo , and G. Fipps, “Deployment of artificial neural network for short-term forecasting of evapotranspiration using public weather forecast restricted messages,” Agricultural Water Management, vol. 163, pp. 363-379, 2016.

[13] M.Fuentes, C.Campos, and S. García-Loyola, “Application of artificial neural networks to frost detection in central Chile using the next day minimum air temperature forecast,” Chilean journal of agricultural research, vol. 78(3), pp. 327-338, 2018.

[14] A. A. Imran Maqsood Muhammad Riaz Khan, "An ensemble of neural networks for weather forecasting," Neural Computing & Applications, vol. 13, no. 2, pp. 112-122, 2004.

[15] B. A. Smith,"Air temperature prediction using artificial neural networks," PhD thesis, University of Georgia, 2006.

[16] K. Philippopoulos, D. Deligiorgi, and G, Kouroupetroglou. “An artificial neural network approach for the forecast of ambient air temperature,” EGUGA, 11446, 2014.

[17] S. Francik, and S. Kurpaska, “The Use of Artificial Neural Networks for Forecasting of Air Temperature inside a Heated Foil Tunnel,” Sensors, vol. 20 no.3, pp.652, 2020.

[18] M.A. Zaytar, and C. El Amrani, "Sequence to sequence weather forecasting with long short-term memory recurrent neural networks." International Journal of Computer Applications vol. 143, no. 11, pp. 7-11, 2016.

[19] M. Abdel-Nasser, & K. Mahmoud, “ Accurate photovoltaic power forecasting models using deep LSTM-RNN,” Neural Computing and Applications, vol. 31, no. 7, pp. 2727-2740, 2019.

[20] 1. P. Donate,X. Li,G. G. Sanchez,and A. S. de Miguel, "Time series forecasting by evolving artificial neural networks with genetic algorithms, differential evolution and estimation of distribution algorithm," Neural Computing and Applications, vol. 22,no. 1, pp. 11-20, 2013.

[21]MPIB official website: https://www.bgc-jena.mpg.de/index.php/Main/HomePage

[22] X. Ma, C. Fang , and J. Ji, “Prediction of outdoor air temperature and humidity using Xgboost,” In IOP Conference Series: Earth and Environmental Science Vol. 427, No. 1, p. 012013, IOP Publishing, 2020