Technium Vol. 25, pp.24-33 (2024)
ISSN: 2668-778X
www.techniumscience.com

Building an emulator for sea surface temperature

Donghee Kim[1,3,*], Dongwook Kim[2,4,*]

[1] School of Software, Chung-Ang University, Republic of Korea

[2] Wheaton Academy, United States

[3] donghee0815@gmail.com

[4] vision.dwk@email.com

*Corresponding author

Abstract. We are interested in the problem of predicting sea surface temperature with machine learning methods using only the relevant climate factors. We tried to resolve the cost inefficiency of the temperature predicting model using over a million of data[1] that is observed by about 6300[2] NASA weather stations, a typical model that prominent agencies like NASA uses. First of all, we minimized the required computing power by using only velocity and pressure variables. Moreover, we minimized the usage of data by using only 14GB of NASA-GISS data and supplemented the minimized data by comparing four different machine learning models to predict global sea surface temperature. In addition, beyond predicting the global sea surface temperature, we also wanted to delve into predicting each region's temperature. Thus, we gridded the earth into 64,800 regions, based on latitudes and longitudes. However, the accuracy drastically decreased since the data was also divided into 64,800 parts. Therefore, we applied the Gaussian filter to our predicted outcomes, and it made a breakthrough in accuracy. This will have a huge impact on temperature prediction because it will astoundingly decrease the expense and the amount of data needed for predicting temperature.

Keywords: Temperature Measurement, Climate Change, Machine Learning

1. Introduction

Recently, we watched a video describing how global climate research groups predict sea surface temperatures using very large and computationally expensive numerical models on huge supercomputers. One fact that surprised us was that an organization

needs more than a thousand computers, a tremendous amount of money to accommodate countless data, and core resources for predicting the seawater temperature. For example, even though the National Aeronautics and Space Administration (NASA) works on various important tasks, we found that they spend prodigal amounts of money to operate hundreds of satellite missions that monitor Earth's climate and link to 6300 weather stations around the world for measuring temperatures. Sea surface temperature affects the climate in many different parts of the world, for example, near ocean currents such as the Gulf Stream, the North Atlantic Current, the Kuroshio, and the Southern Ocean Circumpolar Current. Temperature is also a significant index in geological research because it helps predict different climate states. Suppose we can backtrack the temperature by other seawater data. In that case, it will provide a novel contribution to science that provides a method to accurately estimate temperature, and be the predominant element for predicting climate.

In this paper, we seek to develop an emulator of a climate model that aims to accurately predict sea surface temperature by training machine learning models with seawater data. Applying machine learning methods to climate prediction have been studied by various researchers. For example, Watson-Parris (2021) utilized the linear regression model and the neural network model to facilitate the development of the machine learning model for climate predictions. Moreover, Seokhyun Chin, and Victoria Lloyd (2024) utilized an autoregressive long short-term memory model to predict climate change. Our objective in this paper is to utilize the surface ocean current speed and sea level pressure to predict the earth's average ocean surface temperature, both in global average time evolution as well as the regional pattern prediction.

In this work, we aim to use for the first time NASA-Goddard Institute for Space Studies (GISS) data to emulate sea surface temperatures using surface currents and sea level pressure by deploying a variety of machine learning methods.

The paper is structured as follows. In section 2, we describe the data preprocessing procedure and describe the different algorithms we use: multilayer perceptron (MLP), Random Forest, Deep Neural Network, and XGBoost. Then, in section 3, we describe the results of both global and regional temperature. Finally, in section 4, we offer some discussion and conclusions.

2. Methodology

The NASA Center for Climate Simulation[3] provides extensive climate data that NASA has measured. We downloaded the VO and UO data, which provide the velocity of the seawater across the latitude and longitude, PSL data, which indicates the pressure, and the T data, which indicates the temperature. We acquired monthly data from 1951 to 2014, and stored in NetCDF files, which we later integrated and preprocessed using Python programming language. NASA plays a significant role in geographical research, providing massive and reliable data relevant to our topic so that we can build steadfast algorithms with it. The data can be downloaded from this link(https://portal.nccs.nasa.gov/datashare/giss_cmip6/CMIP/NASA-GISS/GISS-E2-1-G/historical/r101i1p1f1/Omon).

Our goal is to build a model that can show high performance in predicting temperature with a minimum batch of climate variables. We use VO and UO, the speed of ocean surfacecurrent, and PSL, the pressure, as independent variables. We set T as the dependent variable. We extracted the surface level data of UO and VO, which is the region of our interest. Moreover, we normalized the data by subtracting the average value of each column and dividing it by the standard deviation. The resulting array has 64,800 samples indicating each grid across the globe, and 4 variables.

In order to predict the temperature, we fit our data with MLP, Random Forest, Deep Neural Network, and XGBOOST and compare those models to identify which model has the best performance as well as visualize the results. The Multilayer Perceptron (MLP) is a type of a neural network, consisting of more than three layers of nodes which are an input layer, various hidden layers, and an output layer. MLP is a subset of the Neural Network model, the method that mimics how neurons work. It consists of interconnected nodes, called neurons or units, arranged in layers. Each neuron receives the input, processes it, and sends the outputs to the next layer. Random forests are a complicated learning method, especially used for classification and regression. They work by constructing multiple decision trees during training and outputting the class mode (classification) or average prediction (regression) of each tree. It maximizes the efficacy of decision trees since it eliminates the sole shortcoming of decision trees, which are extremely vulnerable to the slight error, by interconnecting several trees so that the solution can be more stable. Deep Neural Network plays a role in building and training machine learning models, especially in deep neural networks. Deep Neural Network offers comprehensive things that can help researchers efficiently make machine learning and distribute it. XGBoost stands for Extreme Gradient Boosting, which is one kind of boosting model. It minimizes the error by using gradient descent optimization at each boosting iteration. XGBoost is a subset of boosting, an ensemble learning method that combines multiple decision trees to improve predictive performance. The core idea is to train each new model and pay attention to the training instances that previous models have misunderstood or predicted incorrectly. To identify which machine learning mode

Technium Vol. 25, pp.24-33 (2024)
ISSN: 2668-778X
www.techniumscience.com

l has the best performance, we use mean squared error (MSE), which is a popular me tric of error in predictions by calculating the average squared difference between the estimated values and the actual values, used in prediction tasks.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

In this equation, n is the sample size, yi' s are the predicted values and yi' s a re the true values.

We ran the model two times each, once before fine-tuning the machine learning para meters (see Table 2) and once after the fine-tuning of the parameters. Then, we te sted the sensitivity of our emulator to the size of the testing data by exploring the range from 10% to 75% of the total dataset. In addition, we use the LSTM model to sort the data by time. We applied a time sequence model, enabling our emulator to use every previous data, not the average data, in our model as the new data lis ting method as recklessness on data sorting may cause considerable error.

Beyond predicting the world' s global temperature, we gridded the world into 64,80 0 regions and predicted each area' s temperature since we wanted to delve into our research topic more elaborately. We separtely measured the monthly average tempera ture data of the world for the summer season and winter season to decrease the pos sibility of error caused by the intermingling of different seasonal temperature da ta. However, each region does not have enough data since the data would be divide d into 64,800 pieces. To solve this, we applied the Gaussian filter to our predict ed temperatures. Gaussian filter is a type of image processing filter that minimiz es noisy data by using a Gaussian function. It applies an oval-shaped weight distr ibution to each portion of the image, adding more weight to portions closer to the center of the filter window and less weight to those further away. The degree of s moothing is controlled by the standard deviation () of the Gaussian function, with larger values resulting in more blurring.

$$G(x,y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

In this equation, stands for standard deviation, G(x,y) is the value of the Gaussian function at the point (x,y), and x and y are the horizontal and vertical distances from the center of the Gaussian kernel, respectively. We ran the four models on each area's climate data and applied a Gaussian filter outputting the final predictions for each area.

3. Results

There are several parameters to proceed the fine-tuning. Random Forest(n_estimators [100, 200, 300], max_depth [10, 20, 30], min_samples_split). MLP(hidden_layer_size[50, 50], [100,50], [100, 100], activation [tanh, relu], solver[sgd, adam], alpha[0.0001,0.05], learning_rate[constant, adapt]. XGBoost(n_estimators: 1000,learning_rate: 0.05 ,max_depth:Value: 10, min_child_weight: 4, subsample: 0.7 ,colsample_bytree: 0.7,gamma: 0.1 ,Reg_alpha:0.3,,reg_lambda:0.8). DNN(epochs: 50, batch_size:32).

We could get four results from each model that we operated.

Test Size	Random Forest	MLP	XGBoost	DNN
0.1	0.614535	0.637807	0.657734	0.6024
0.15	0.592913	0.638027	0.780081	0.662303
0.2	0.53985	0.641939	0.750039	0.667402
0.25	0.642555	0.664077	0.769042	0.69102
0.3	0.66637	0.66041	0.806797	0.704548
0.35	0.653371	0.656446	0.813929	0.694012
0.4	0.625519	0.664079	0.745538	0.697372
0.45	0.658101	0.660779	0.730277	0.690644
0.5	0.683078	0.662879	0.760572	0.698112
0.55	0.738885	0.663955	0.880769	0.698581
0.6	0.717057	0.683335	0.827589	0.700851
0.65	0.720739	0.678119	0.817318	0.708531
0.7	0.70822	0.665662	0.815233	0.69624
0.75	0.712913	0.665562	0.855918	0.69624

Technium Vol. 25, pp.24-33 (2024)
ISSN: 2668-778X
www.techniumscience.com

Table 1. It shows the MSE of 4 learning machines in a data range of 10~75% before fine-tuning.

Test Size	Random Forest	MLP	XGBoost	DNN
0.1	0.583698	0.605175	0.61887	0.704123
0.15	0.606152	0.63722	0.65418	0.632139
0.2	0.60971	0.659984	0.671705	0.616439
0.25	0.620138	0.661568	0.715186	0.656955
0.3	0.633748	0.66828	0.770989	0.657813
0.35	0.617581	0.660996	0.738796	0.624277
0.4	0.604703	0.660166	0.726681	0.64472
0.45	0.622685	0.660619	0.693693	0.691198
0.5	0.647979	0.65616	0.724447	0.662071
0.55	0.698409	0.698434	0.737041	0.709595
0.6	0.684068	0.677077	0.74281	0.691302
0.65	0.689686	0.734816	0.750859	0.698495
0.7	0.681804	0.693054	0.73531	0.685387
0.75	0.677766	0.727168	0.751077	0.660445

Table 2. It shows the MSE of 4 learning machines in a data range 10~75% after fine-tuning.

The result showed splitting data into 3 parts with 70% of data for training, 20% of data for validation, and 10% of data for testing makes the best performance on every model. Moreover, fine-tuning affected the accuracy a lot since the result after fine-tuning shows the accuracy has increased by 6% overall.

Technium Vol. 25, pp.24-33 (2024)
ISSN: 2668-778X
www.techniumscience.com

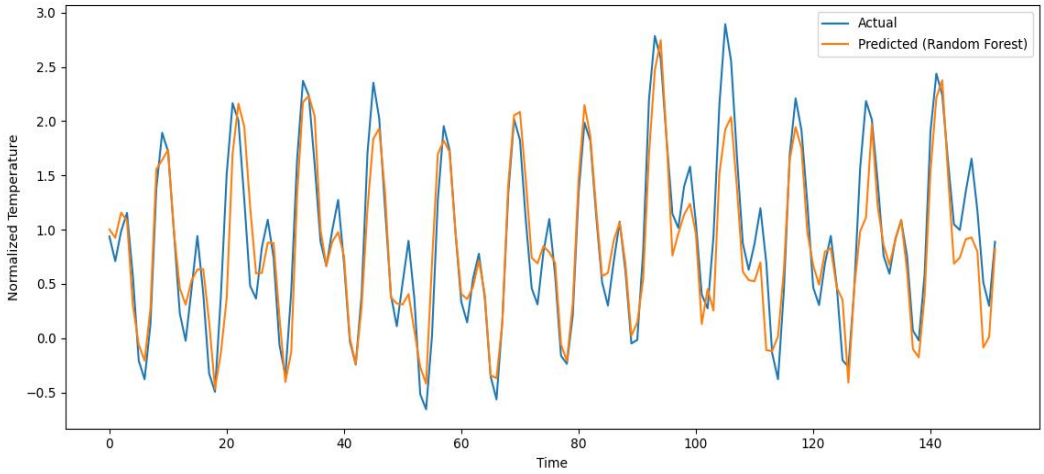

Figure 1. It shows the actual and the predicted global temperature with Random Forest.

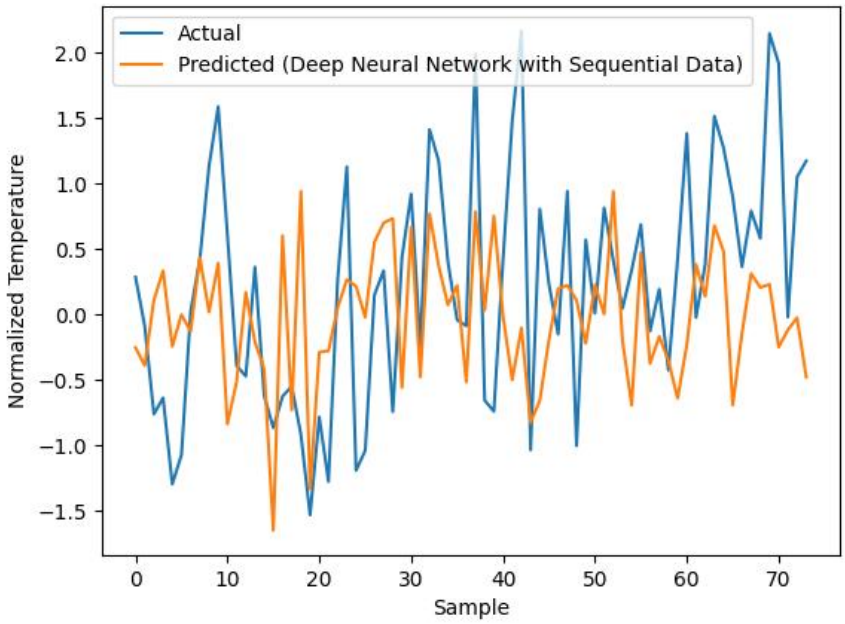

Figure 2. It shows the actual and the predicted global temperature with Random Forest.

Among the four models, Random Forest model's result had the best performance by Mean Squared Error around 0.5, yet others had average 0.7. So we applied the LSTM model in the Random Forest model, and got the result that only has 0.1 MSE, which means the difference between the actual data and the predicted data is incredibly small over the whole period. On the other hand, when we applied the LSTM model to t

he DNN model, even though the result between the sample 40~60 was pretty accurate, other parts had a big error.

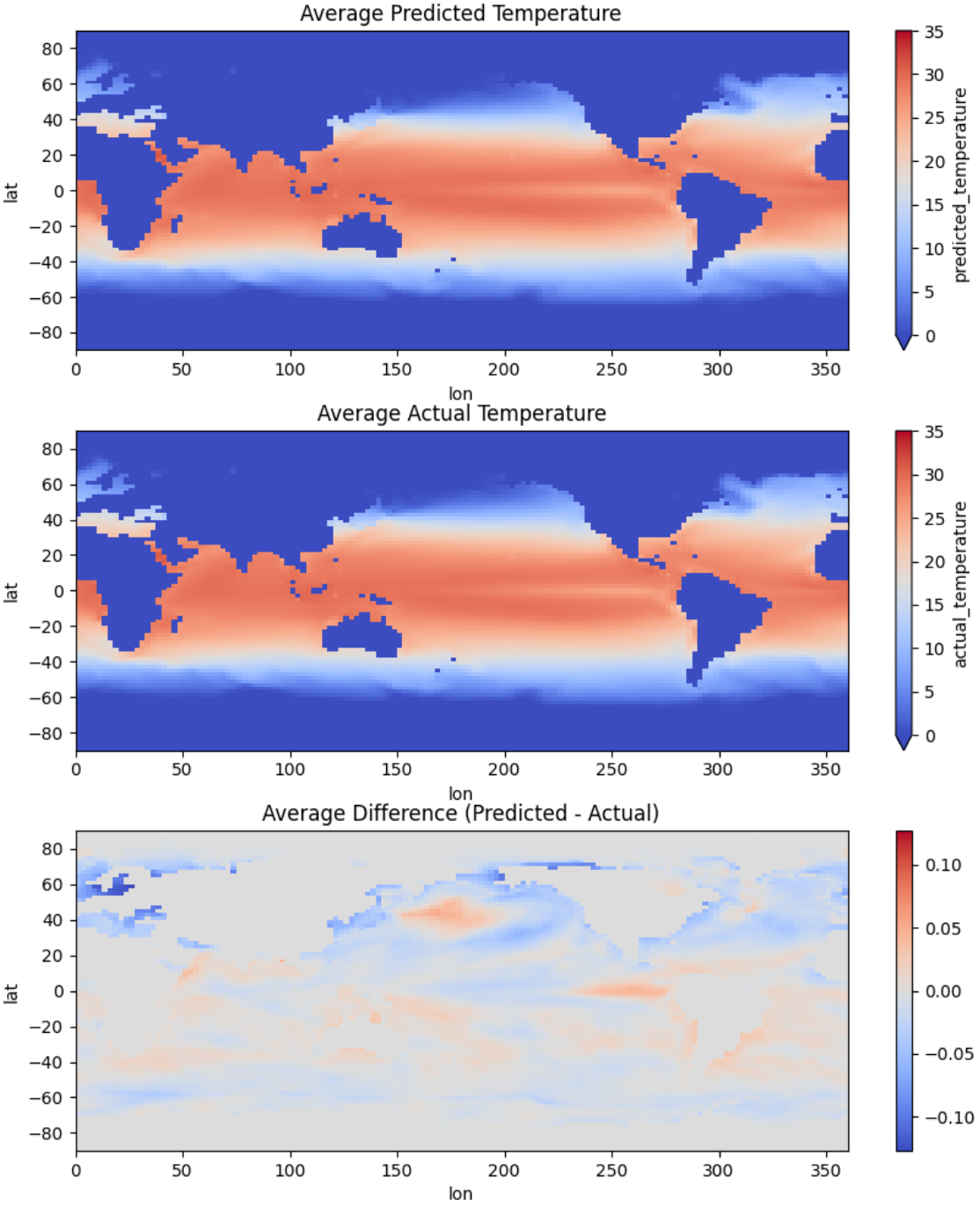

Figure 3. It shows average values of the predicted and actual temperature, and the difference between those two values.

We subtracted two models, average predicted temperature model and average actual m odel, and specified the outcome with the range of color. Gray part shows that the

difference between predicted value and the actual value is nearly 0, and the blue or red parts show that there is some difference between the two values.

4. Discussion and Conclusion

Our project will have an amazing influence on temperature-predicting areas since it will astoundingly reduce the money needed for predicting seawater temperature. Moreover, it will also reduce time since our seawater temperature-predicting model only uses a few variables. We are planning to develop our model to be able to predict the abnormal climate that has sparked recently by the successive airplane accidents caused by that. We'll add the direction and the power of wind, pressure, and abnormal climate data to evolve our model. This research will detect the area that abnormal climate occurs frequently, alerting airplanes to avoid that area, decreasing the accident.

Acknowledgements

This endeavor would not have been possible without Professor Anastasia Romanou of the Columbia University and National Aeronautics and Space Administration (NASA)-Goddard Institute for Space Studies (GISS), who supervised the entire project and generously provided knowledge and expertise.

We would like to extend our sincere thanks to our Research Assistant Yeojin Jung, M.Sc of the Yonsei University, for her invaluable patience and feedback.

References

1. *How NASA Scientists Measure Global Temperatures. (2017). [YouTube Video]. In YouTube. https://www.youtube.com/watch?v=3Uv26dIgaKs*
2. *Introduction to XGBoost Algorithm | by Nadeem | Analytics Vidhya.* (2021, March 5). Medium. Retrieved July 13, 2024, from https://medium.com/analytics-vidhya/introduction-to-xgboost-algorithm-d2e7fad76b04
3. Lloyd, V. (n.d.). *Predicting climate change using an autoregressive long short-term memory model.* Frontiers. Retrieved July 20, 2024, from https://www.frontiersin.org/journals/environmental-science/articles/10.3389/fenvs.2024.1301343/full
4. Lyashenko, V. (n.d.). *Random Forest Regression - The Definitive Guide.* cnvrg.io. Retrieved July 20, 2024, from https://cnvrg.io/random-forest-regression/
5. Singh, S. M. (2024, January 23). *What is Multilayer Perceptron (MLP) Neural Networks?* Shiksha. Retrieved July 20, 2024, from https://www.shiksha.com/on

line-courses/articles/understanding-multilayer-perceptron-mlp-neural-networks/
5.1.

6. Watson‐Parris, D., Rao, Y., Olivié, D., Seland, Ø., Nowack, P., Camps‐Valls, G., Stier, P., Bouabid, S., Dewey, M., Fons, E., Gonzalez, J., Harder, P., Jeggle, K., Lenhardt, J., Manshausen, P., Novitasari, M., Ricard, L., & Roesch, C. (2022). ClimateBench v1.0: A Benchmark for Data‐Driven Climate Projections. Journal of Advances in Modeling Earth Systems, 14(10). https://doi.org/10.1029/2021ms002954
7. (2020). Nasa.gov. https://portal.nccs.nasa.gov/datashare/giss_cmip6/CMIP/NASA-GISS/GISS-E2-1-G/historical/r101i1p1f1/Omon/
8. *Deep Neural Network - an overview | ScienceDirect Topics*. (n.d.). Www.sciencedirect.com. https://www.sciencedirect.com/topics/computer-science/deep-neural-network