Investigation of Mixed Layer Depth through the lens of Artificial Intelligence

A thesis submitted to

Indian Institute of Science Education and Research, Pune
in partial fulfilment of the requirements for the
BS-MS Dual Degree Programme

Submitted by

Ankit Bhaskar
Indian Institute of Science Education and Research, Pune

Under the supervision of

Dr. Bipin Kumar (IITM Pune)
Co-Supervisor: Dr. Bishakhdatta Gayen (University of Melbourne)
Co-Supervisor: Dr. Manmeet Singh (IITM Pune)
TAC Member: Dr. Anupam Kumar Singh (IISER Pune)

Certificate

This is to certify that this dissertation entitled **Investigation of Mixed Layer Depth through the lens of Artificial Intelligence** towards the partial fulfilment of the BS-MS dual degree programme at the Indian Institute of Science Education and Research, Pune represents the work carried out by Ankit Bhaskar during the academic year 2021-2022 and comprises of original work with due citations to all other material used. The dissertation has not been submitted in any form for another degree or diploma at any other institute/university.

Bipin

Dr. Bipin Kumar
Scientist E, HPCS
IITM Pune

Supervisor: Dr. Bipin Kumar (IITM Pune)
Co-Supervisor: Dr. Bishakhdatta Gayen (University of Melbourne)
Co-Supervisor: Dr. Manmeet Singh (IITM Pune)
TAC Member: Dr. Anupam Kumar Singh (IISER Pune)

This thesis is dedicated to my Mother
I pray for her joyful and healthy life ahead.

Declaration

I hereby declare that the matter embodied in the report entitled **Investigation of Mixed Layer Depth through the lens of Artificial Intelligence** is the result of the work carried out by me at the Indian Institute of Tropical Meteorology, Pune and the University of Melbourne under the supervision of Dr. Bipin Kumar, Dr. Bishakhdatta Gayen and Dr. Manmeet Singh and the same has not been submitted elsewhere for any other degree.

[illegible]

Ankit Bhaskar

Acknowledgments

I would like to express my sincere gratitude to Dr. Bipin Kumar, Scientist 'E', IITM Pune for providing me with the opportunity to take up this project and most importantly, for his precious guidance and motivation, which helped me to conquer the difficulties in research and development of this project. I would like to thank him for going unconventional ways for helping me out throughout my thesis and for having patience with me.

Further, I am very grateful to Dr. Bishakhdatta Gayen, Lecturer, University of Melbourne, for conceptualising this project and guiding me throughout the journey. I would also like to thank Dr. Manmeet Singh, Scientist 'C', IITM Pune, for being available all the time and providing me with his expert support whenever needed.

Additionally, I would like to thank Dr Anupam Kumar Singh for always showing a keen interest in the project and giving valuable suggestions and comments.

Next, I would to acknowledge my friends Rithvika, Kaustubh, Vaisakh, and everyone whose name I have not mentioned for having my back. I owe them a debt of gratitude for taking care and looking after me in my tough times. I wish you all the best in all your future endeavours.

Abstract

Predicting the various oceanic parameters responsible for air-sea coupling is crucial to understanding how the climate and weather systems can affect the ecosphere. One of the most important among these oceanic parameters is the Mixed Layer.

In this study, the convolutional long short-term memory(ConvLSTM) based Neural Network(NN) architecture is used for monthly forecasting of Mixed Layer Depth(MLD) in the Bay of Bengal(BOB) region. The study uses multi-variables corresponding to other prominent ocean surface phenomena as input and the AI model is used to learn and understand the link between these input variables and the output variable MLD. This study forecasts the MLD with a correlation better than the operational dynamical Hindcast model and the ablation study suggests a decline in performance when any of these 5 input variables were removed from the training. The study not only deciphers the relationship between these variables and the MLD but also opens an interesting field to explore the forecasting of other ocean phenomena which directly or indirectly depend on the MLD.

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

In recent years, climate science and meteorology have become highly significant as it has a wide range of effects on all living things, the evidence of which we can see in floods and heat waves prevailing in the current time. As a result, it is critical to study meteorological conditions, and more accurate forecasting plays a significant role in averting extreme weather and disasters. Current Numerical weather prediction models(NWP) are prone to significant error levels and even though We will not be able to completely eliminate this error range, it should be lowered in order to improve forecasts [4].

With the effective use of data, learning algorithms, and sensing devices, Artificial Intelligence (AI) is a disruptive paradigm that has increased the ability to analyze, anticipate and reduce the danger of climate change. These algorithms are known to decipher the non-linear patterns when predicting any phenomenon and it has enabled the prediction of MLD values in the study [5]. Machine Learning algorithms have a lot of potential applications in climatology as it does calculations, forecasts, and takes decisions to help reduce the effects of climate change. AI helps us better comprehend the effects of climate change across different geographical places by generating effective models for weather forecasting and environmental monitoring. It analyses climatic data and forecasts weather occurrences, extreme weather conditions, and other socioeconomic consequences of climate change and precipitation [6][7]. From a technological standpoint, AI improves climate projections, demonstrates the effects of extreme weather, identifies the true source of carbon emissions, and makes countless other useful contributions. This empowers policymakers to be conscious of increasing sea levels, natural catastrophes, storms, climate change, habitat degradation, and species extinction. Nonetheless, the scientific community and specialists have begun to concentrate on climate informatics utilizing AI paradigms[8]. Machine Learning, a subclass of AI, has progressed to the point that it is being used to discover anomalies, as reported by more than 200 articles since 2000[9]. Deep Learning is known as a data-hungry approach since it requires a huge dataset to analyze and learn the mapping between input and target variables. This technique can also be applied to forecasting. Open-source frameworks like TensorFlow, Keras, and others, together with high-performance computers, have decreased the barriers to sophisticated calculations. We can now import greater datasets than ever before because of high-performance computers and parallel processing.

1.2 Definition

Mixed Layer is the uppermost part of the marine surface waters that is characterized by its homogenous density (temperature/salinity) distribution. It also directly impacts other essential air-sea variables like Surface Sea Temperature (SST) and Surface Air Temperature [10]. It regulates the air-sea interactions (heat, moisture, CO2, etc.). Throughout the world's oceans, the depth of this mixed layer is maintained/controlled by various processes such as Langmuir turbulence, shear wind-driven mixing, and upper ocean buoyancy fluxes[11][12]. The MLD or depth of turbulent mixing is very sensitive to density and temperature stratification and can change with the seasons and small climatic perturbations. Further, although the diminution of the mixed layer due to turbulent mixing is more rapid with a decrease in turbulent mixing when compared with the influence of temperature and density factors, data shows that the value of MLD is not consistent with the value of MLD derived from such mixing when the wind is weak. It is a fact that direct observation is a good way to obtain MLD values and so Bhaskar et al.[13] inferred MLD variation in the western Indian Ocean by Argo observations. Another definition proposed by Monterey et al.[14] suggests that MLD is the depth of a layer which has a certain density calculated based on salinity and temperature profiles and temperature is taken to be 0.5°C lower than the surface temperature. We can find different definitions of MLD but they suggest similar spatial distribution. The estimated MLD from these definitions is mostly consistent irrespective of the definitions being seasonal or spatial. An important purpose of this report is to analyze the playing factors that impact the formation of the mixed layer. Some of the studies seen in the articles like Fan et al.[15] suggests that the MLD is the most sensitive to Sea Surface Wind Stress. The absence of net sea surface heat flux forcing and evaporation/precipitation difference can further change the MLD by 13% and 2%. However, a study to determine the effect of all the phenomena at the sea surface is absent and this project would try to fill the gap. Figure 1.1 shows the trend of MLD values at the point 7.5°N and 83°E in the central region of BOB.

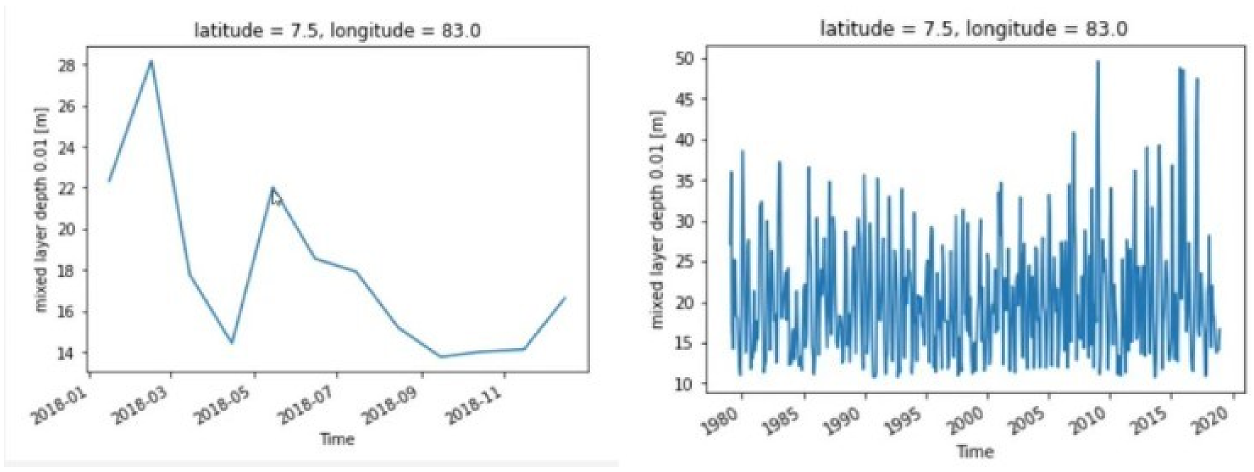


Figure 1.1: The figure shows the MLD values for the year 2018 (left) and for years 1979-2018 (right) for 7.5°N and 83°E

1.3 Thesis structure

The thesis is organised as follows: Chapter 2 provides the theoretical background of the DL algorithms and metrics which are used in this study. Chapter 3 discusses the details of the data with their sourcing and pre-processing steps. Chapter 4 highlights the model architecture and the steps in its training and testing. Chapter 5 discusses the result and ablation studies done on the model with further validations. Finally, Chapter 6 provides the conclusion of our work and ho this study creates a future impact.

Chapter 2

Theoretical Background

2.1 NN Fundamentals

Artificial Intelligence (AI) is the utilization of machines to perform complex computational tasks. Previously, during the age of Symbolic AI, problems were first solved by humans before being converted to machine-readable code, in essence, “teaching’ the machine to solve the problem. This was done primarily because the popular opinion of the time was that people would need to painstakingly hand-write a detailed, extensive and expansive set of rules which would dictate how a machine would behave.

What soon followed was a method by which a set of data (supervised input and output) would be provided to the program and encoded with updated parameters to calculate a new output when given new input. This was a drastic shift as previously in Symbolic AI, the original instructions could only be changed by the programmer. Currently, machine learning involves the use of computational algorithms that teach the machine a set of rules of a problem and solve it when a set of data is supplied, i.e., machine learning from a data-driven approach. This section will primarily explain the types of neural networks and the framework of machine learning algorithms used in this experiment

Machine Learning models based on how an algorithm learns a pattern as simply categorized as:

- Supervised
- Unsupervised

- Semi-supervised
- Reinforcement Learning

The model used in this work is a supervised machine learning model and uses pre-labelled meteorological data in training. The input data is of $(X_i - X_n, Y)$ structure in this supervised learning. A function f that can transform X to Y via representational learning will be found. Y plays an important role in the working of the model as f keeps getting updated through back-propagation of gradient-based loss function defined by L(Y, f(X)) taking into account the optimizers and evaluation metrics. The model used is a supervised deep machine learning model for non-linear time series regression, it is referred to as "deep" because of the use of multiple layers of neural networks.

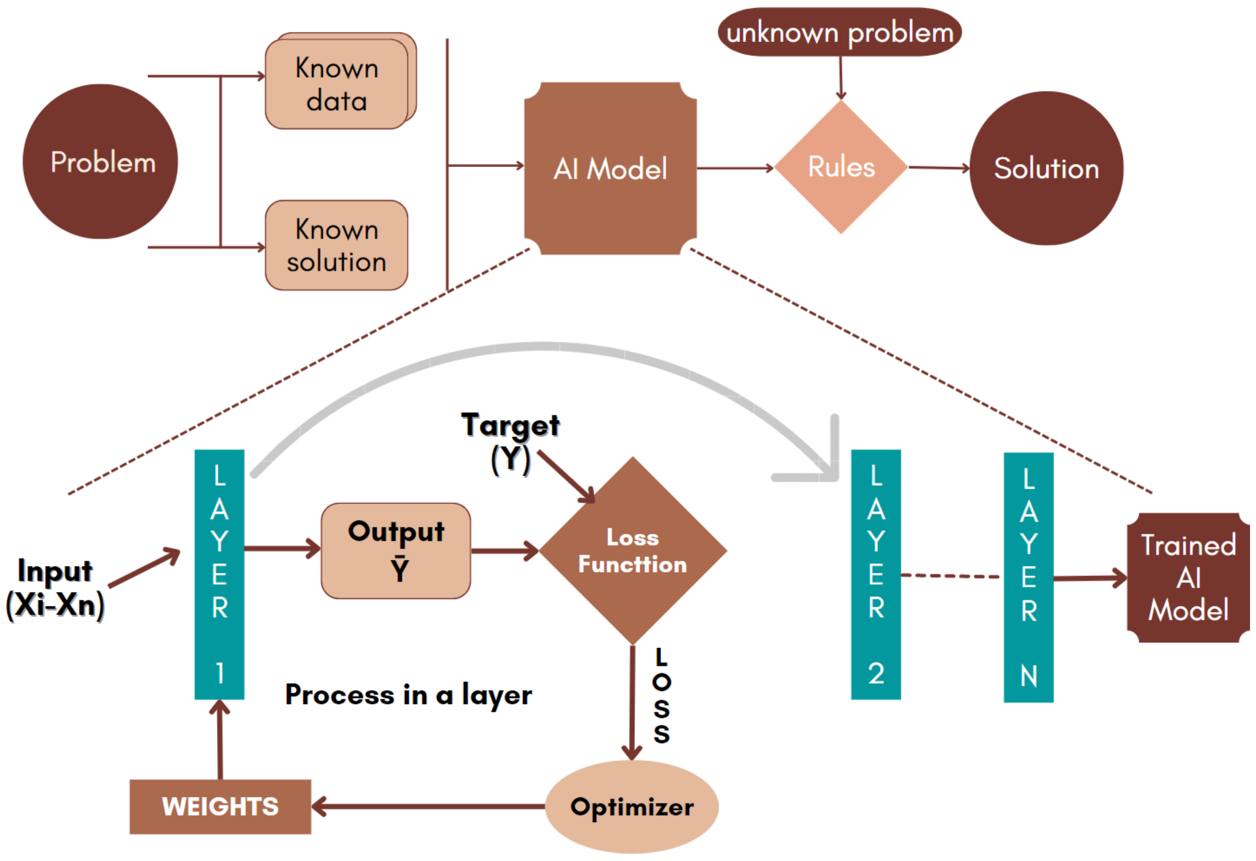


Figure 2.1: Flowchart of a DL algorithm

Deep learning models have recently become popular as they have been shown to be universal nonlinear function approximations [5]. There have also been significant advances in computational efficiency to train such large models. These models are expected to learn meaningful representations of the high-dimensional input through successive layers of transformations.

Figure 2.1 shows the general workflow of a supervised deep learning model. It depicts how the supervised deep learning algorithm loops through the process under a conditional statement. This statement can be as simple as a certain fixed number of loops, i.e., Epochs or could be a continuous loop till a certain metric of interest has stopped improving after a certain number of epochs, such as the loss value or a scalar function.

2.2 NN Layers

Neural networks are made from layers which perform transformations on the inputs to the layer to make it into a higher or lower dimensional representation. Layer transformation can be mathematically represented as, for a given n-dimensional input matrix $X \in \mathbb{R}^{j_1-j_n}$ & an m-dimensional weight matrix $w \in \mathbb{R}^{l_1-l_m}$, the layers perform optimization $o : \mathbb{R}^{j_1-j_n} \to \mathbb{R}^{l_1-l_m}$ that transforms X to a p-dimensional output matrix $\bar{X}$.

Here, the operation, o, can be an iterative or single-step function working to feed the activation function (ReLU in this study). The layers can be of several kinds as per the user and are followed by a linear or non-linear activation function on Y, the output matrix, in-built a layer or forming a separate layer. Moreover, hyperparameters are pre-defined during the compilation of the layer and are responsible for deciding the learning rate.

2.3 Convolutional Layer

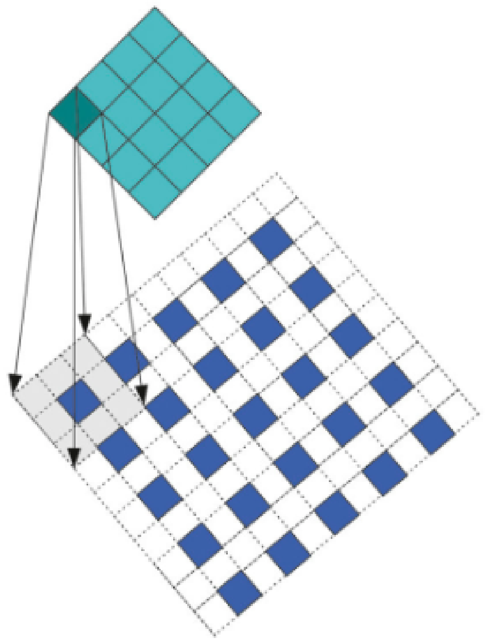


Figure 2.2: Deconvolution (Image taken from Jia, Chao[1])

As the name sounds, the layer does the convolutional operation on its inputs. A conv2D or conv3D layer can be used depending on the dimension of the data in the study. In its initial step, a $m{\times}n$ input matrix is taken, given a $k{\times}k$ kernel matrix that moves over the input matrix referred by 'stride'. It takes the element viz dot product with the element of the input matrix and itself 's' times to return $(m-k)/(s+1)\times(n-k)/(s+1)$ output matrix as shown in figure 2.2. The dataset is usually padded with data for the functioning of the kernel on the edges and this is a common practice in computer vision problems. Similarly, there is an option of padding by default in the deep learning system

It is important to note that in order to incorporate the RGB images, a conv2D layer has three 2D matrices considering each channel that describes image. Similarly, for our dataset, each of the three channels was filled in with latitude, longitude and time parameter. When the multiple lead time prediction is to be calculated, more than 3 channels are required and for such cases, conv3D layer, capable of incorporating up to 5 channels, is used.

2.4 Recurrence Layer

These layers are used in the transformation of the sequential data and their simplest form is 2D matrix with sample and feature as the dimensions. These are generally delineated by 2 components including single transformation cell and a function to repeat this transformation further in every time step with remembering some information in the distant run in the form of weight matrices.

Hochreiter et al.[16] introduced LSTM or 'Long Short-Term Memory' with multiple advantages on RNNS and will be used in this study in a modified form. LSTM not only contains specific memory cell matrices which are carried till the end of learning but also helps in remembering middle-distant temporal patterns.

The single-time step transformation into any LSTM cell is represented by the following equations.

Assumptions:

- $x_t \in \mathbb{R}^a$ is a 1-D vector with input length 'a', to the LSTM
- $h_t \in \mathbb{R}^b$ is the hidden-state vector with 'b' hidden units
- $i_t \in \mathbb{R}^b$ is the activation vector at input gate
- $f_t \in \mathbb{R}^b$ is the activation vector at forget gate
- $o_t \in \mathbb{R}^b$ is the activation vector at output gate
- $\bar{c}_t \in \mathbb{R}^b$ is the activation vector at memory cell input
- $c_t \in \mathbb{R}_b$ is the memory cell state vector
- $b \in \mathbb{R}^b$ and $w \in \mathbb{R}^{a*b}/\mathbb{R}^{b*b}$ based on multiplication with x_t or h_t
- $\sigma : \mathbb{R} \to \mathbb{R}$ represents sigmoid activation function (for reference)
- $\sigma_h : \mathbb{R} \to \mathbb{R}$ represents the hyperbolic activation function (default)
- $\circ : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ represents the Hadamard product

Equation:

$$i_t = \sigma(w_{ix}x_t + w_{ih}h_t + b_i) \tag{2.1}$$

$$f_t = \sigma(w_{fx}x_t + w_{fh}h_t + b_f) \tag{2.2}$$

$$o_t = \sigma(w_{ox}x_t + w_{oh}h_t + b_o) \tag{2.3}$$

$$\bar{c} = \sigma_h(w_{cx}x_t + w_{ch}h_t + b_c) \tag{2.4}$$

$$c_t = f_t \circ c_{t1} + i_t \circ \bar{c}_t \tag{2.5}$$

$$h_t = o_t \circ \sigma_h(c_t) \tag{2.6}$$

Consequently, an RNN can be customized to perform a transformation on more complex input matrices by using cells suitable for such inputs. We next present one such layer architecture which takes 5-dimensional inputs of the type (samples, timesteps, height, width, channels)

2.5 Convolutional Recurrence Layer or ConvLSTM

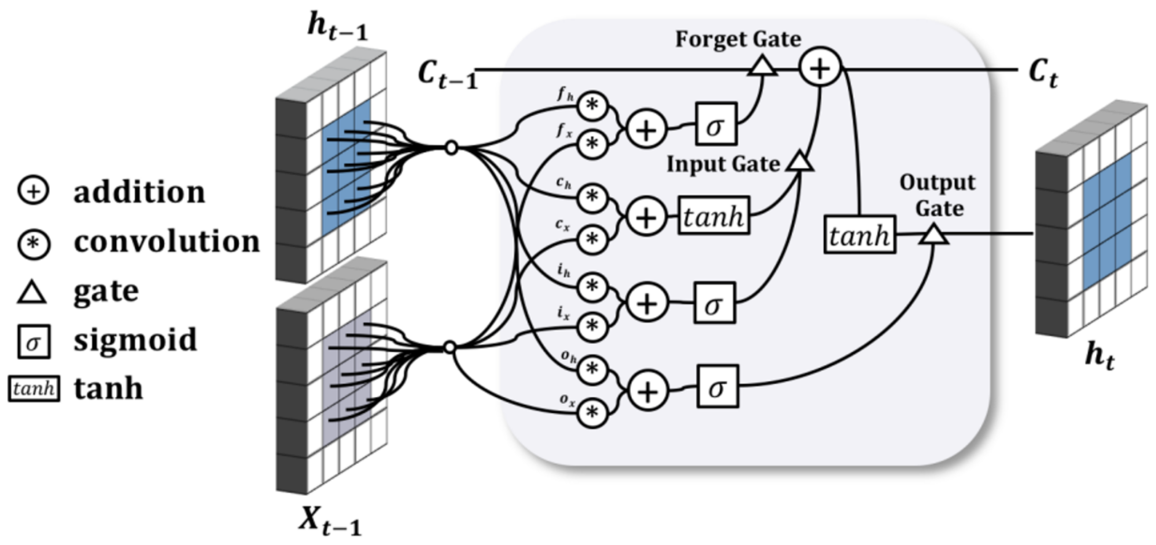


Figure 2.3: Inner structure of a ConvLSTM layer (Figure taken from Yuan et al.[2])

The model used in our study was first introduced by Shi et al.[17] and showed the implementation of the ConvLSTM2D layer. This model functions like LSTM networks and has images in time series. Their transformation equation is similar to LSTMs discussed in the previous section and the convolutional operation is used instead of the matrix multiplication when extracting spatial features. Figure 2.1 shows the structure of a single ConvLSTM2D layer. The mathematical representation of the operations happening inside a ConvLSTM2D cell with the variable description being the same as described in LSTMs section 2.4 unless specified are as follows:

$$i_t = \sigma(w_{ix} * x_t + w_{ih} * h_t + b_i) \tag{2.7}$$

$$f_t = \sigma(w_{fx} * x_t + w_{fh} * h_t + b_f) \tag{2.8}$$

$$o_t = \sigma(w_{ox} * x_t + w_{oh} * h_t + b_o) \tag{2.9}$$

$$\bar{c} = \sigma_h(w_{cx} x_t + w_{ch} h_t + b_c) \tag{2.10}$$

$$c_t = f_t \circ c_{t1} + i_t \circ \bar{c}_t \tag{2.11}$$

$$h_t = o_t \circ \sigma_h(c_t) \tag{2.12}$$

Assumptions: The symbol $'*'$ indicates the convolutional operation instead plain matrix multiplication as in LSTMs case. Other considerations different that LSTMs are as follows:

- x takes 5D input corresponding to samples, timesteps, height, width and channels
- h_t and c_t are 4D matrices and is zero by default
- w are convolutional filters (k, k).

2.6 Activation Functions

As it sounds, the activation function decides whether a neuron will be activated or not with its sole purpose being the addition of non-linearity to the NN. In every forward propagation, these activation functions add an extra step which when absent, the layers will give out a linear out independent of their numbers and the model would essentially behave like a linear regression model. The activation function can be divided into Binary Step Function, Linear Step Function and Non-Linear Activation functions as discussed below.

2.6.1 Binary Step Function

The binary step function is reliant on a predefined threshold value which determines whether or not a neuron should be engaged. If the input given to the activation function is greater than a specific threshold, the neuron is activated; otherwise, it is deactivated, which means that its output is not transmitted onto the following hidden layer. It can be represented mathematically as

$$f(x) = \begin{cases} 0 \; for \; x < 0 \\ 1 \; for \; x \geq 0 \end{cases} \tag{2.13}$$

However, the binary function has many limitations such as it is not useful if multi-class classification as it can only return 2 values. Further, it cannot be used for the backpropagation process as the gradient of the step is zero.

2.6.2 Linear Activation Function

Linear Activation Function does contribute to the weighted input sum and returns the input value directly making it insignificant as an activation function. It can be simply represented mathematically as:

$$f(x) = x \tag{2.14}$$

As trivial as it looks, it cannot be used in most of the deep learning cases as neither can it be used in the backpropagation cases nor it returns any difference in forward propagation as all layers will return the same output making the NN collapse and behave like a single layer.

2.6.3 Non-Linear Activation Function

The non-linear Activation function forms the base of any deep earning model and can create any desired complex mappings throughout the input and output of NN layers. In this case, not only backpropagation but also stacking of the NN makes sense as their non-linear combinations could give out a functional output. There are numerous non-linear activation functions used in deep learning models and some prominent among them are mentioned here.

Sigmoid/Logistic Activation Function

The sigmoid or logistic functions return output between 1.0 and 0.0 with any input in the range between 0 to 1 in a directly proportional relation. The mathematical representation of the function is as follows:

$$f(x) = \frac{1}{1 + e^{-x}}$$

This activation function is mostly used in the cases where we need probability value and as these provide a smooth gradient, it prevents jumps in the output values. However, the sigmoid activation functions suffer from the vanishing gradient problem and so their use is subject to the case in the picture.

Hyperbolic Tangent Function (Tanh)

The functions shares similarity with the sigmoid function and the shape resembles 'S' like the former with output ranging between -1 to 1. The function returns a value closer to 1 when the output is larger and vice versa. The mathematical representation is as follows:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Similar to the sigmoid activation function, this too faces the problem of vanishing gradient as when the input gets towards extremes, the output becomes relatively flat. However, as it returns values between -1 to 1, the mean of the layers lies near 0 and data is centred making it easy for the learning process of the next layer.

Rectified Linear Unit

Rectified linear Unit or ReLU does give an imprint of linear function but is differentiable and supports backpropagation. The function activates when the linear transformation gives an output greater than 0. It can be represented mathematically as:

$$f(x) = max(0, x)$$

Like every model, even though ReLU is very efficient owing to the fact that it does not activate in every case saving computational burden, it has a limitation as it tends to miss weights and biases of some neurons in the backpropagation process leading to dead neurons. This limitation is commonly known as the dying ReLU problem.

Parametric ReLU

With the aim of solving the dying ReLU problem and preventing the gradient to become zero for sub-zero input, the function's slope of the negative part is transformed into an argument, say a. The value of 'a' is learnt in the process of backpropagation.

$$f(x) = max(ax, x)$$

While performing better than ReLU in tackling the dying ReLU problem, it still fails to solve it. In certain cases, the slope parameter 'a' is not necessarily passed accurately.

Exponential Linear Units

ELU or Exponential Linear Unit work similar to ReLU but has different functioning in the negative part as it modifies the slope. ELU utilizes an exponential function on the negative side of the input function. It can be represented mathematically as:

$$f(x) = \begin{cases} x \ for \ x \ \geq 0 \\ \alpha(e^x - 1) \ for \ x < 0 \end{cases} \tag{2.15}$$

Even though it should perform better than ReLU in remembering the weights, it is computationally heavy as a new operation of the exponential function is exploited at every layer and when compared to Parametric ReLU, it does not learn the value of 'a' and has a fixed function.

Conventionally, Tanh and sigmoid were more prominently used in NN until recently when ReLU was found to converge faster than the two. This study will also work with ReLU as the activation function. A comparison was done with the 'sigmoid' activation function and ReLU not only converges faster but also gave better results [18].

2.7 Loss Function

With every iterative epoch in a deep learning run, a model aims to minimize its loss function. The value of this loss function is always scalar irrespective of the complexity of the data. There is an option to provide multiple loss functions and the model averages the scoring to optimize the model unless specified otherwise.

An essential consideration while defining the loss metric is to take into account the characteristic to be minimised and to consider if the neural network could perform accidental optimisations, not explicitly directed to be performed. For example, if we consider two important metrics to measure the efficacy of time series weather predictions, RMSE and ACC (Anomaly Correlation Coefficient). If we put a condition that the model is only allowed to increase ACC (the model by default reduces scalar computed by a loss function, we input I'=1 ACC), it will lower pattern correlation between predicted and actual values but it will ignore large disparities of the magnitude of difference between these values. Thus, MSE or RMSE are preferred as loss functions in model training which will teach the model superior transformation than only correlation metrics. Hence, deciding the loss function to be used is essential to enable the model to focus on the characteristics to be minimised, i.e., pushes the model to learn a specific skill

2.8 Performance Metric

Apart from loss, we can look for certain other evaluation metrics to be calculated to check if the model is improving in an intended way or not improving in an unintended way at the end of each epoch such as accuracy, mean absolute error and RMSE/MSE, Pearson Correlation Coefficient, ACC etc.

For the crux of this study, Pearson Correlation Coefficient is used. It represents the relationship between the correlation coefficient matrix, R, and the covariance matrix, C as:

$$R_{ij} = \frac{C_{ij}}{\sqrt{C_{ii}C_{jj}}} \tag{2.16}$$

The Covariance indicates the level to which two variables have a linear relationship. The correlation value of 1 indicates that that two datasets are moving together and the value of 0 suggests that they move independently. There is no hard and fast rule that says a certain level of

2.9 Types of Climate forecasting Models

2.9.1 Dynamical Models

These are also referred to as a dynamical model which relies on General Circulation Models (GCMs). These models are basically complex groups of formulas simulating the physics and chemistry of the atmospheric state. In most of the cases, it contains partial differential equations solved mathematically depending on the thermodynamic properties of the simulated atmospheric processes. (Bauer et al. (2015)) shows that over time, the predictions by these NWP have improved and all the weather forecasts which are in operation come from these models. Even though these dynamical models are quite successful there is a huge scope for improvement as some of the limitations exist such as complexities at the poles. In the forecasting of phenomena like precipitation and lightning, these models are extensively used but deep learning has proved to be beneficial in further improvement.

2.9.2 AI Models

The weather forecasting problem can be seen as a classical machine learning problem where we can get test and training data from the existing atmospheric state, however, the traditional machine learning algorithms related to regression could not prove many benefits in this case. The recent progress in deep learning algorithms especially computer vision has made it possible to learn from the atmospheric data using CNNs and developments in RNNS and LSTMs have improved the lead time predictions as in Natural Language Processing.

Only lately have there been studies towards emulating GCMs or developing complete trainable models for solely data-driven predictions utilizing machine learning, notably NNs, for such issues. Models are classified into three major categories:

1. **Direct:** A unique model is trained for each subsequent prediction time step (eg.: monthly in our study). The model does not differ in the NN architecture but the data fed in is different and as a result for every time step forecasting step, Y is altered in any supervised training model f(X,Y) accordingly.

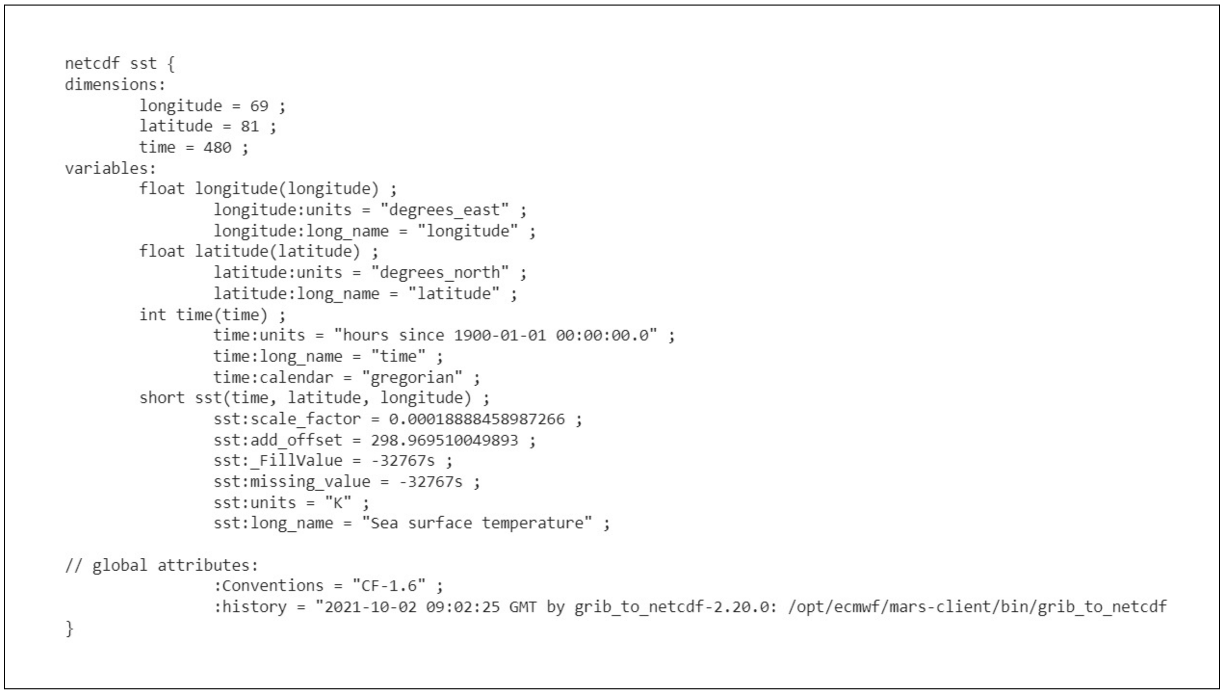
2. **Iterative:** These models are more complex than direct models and there is an option to keep variable 't' time steps. They can predict future time steps with n or multiples of n lead times depending on the problem. These models are tougher to train and demonstrate less proficiency than direct models.
3. **Continuous:** Time is employed as an extra input in this sort of model, and a fixed model is trained to create all prediction lead times[19].

Chapter 3

Data Insights

3.1 Climate data format

NetCDF or Network Common Data Form is the most commonly used format for climate models. It is saved with extension '.nc' and stores array-oriented scientific data. The depth of the detail stored in the data can be seen in the figure 3.1



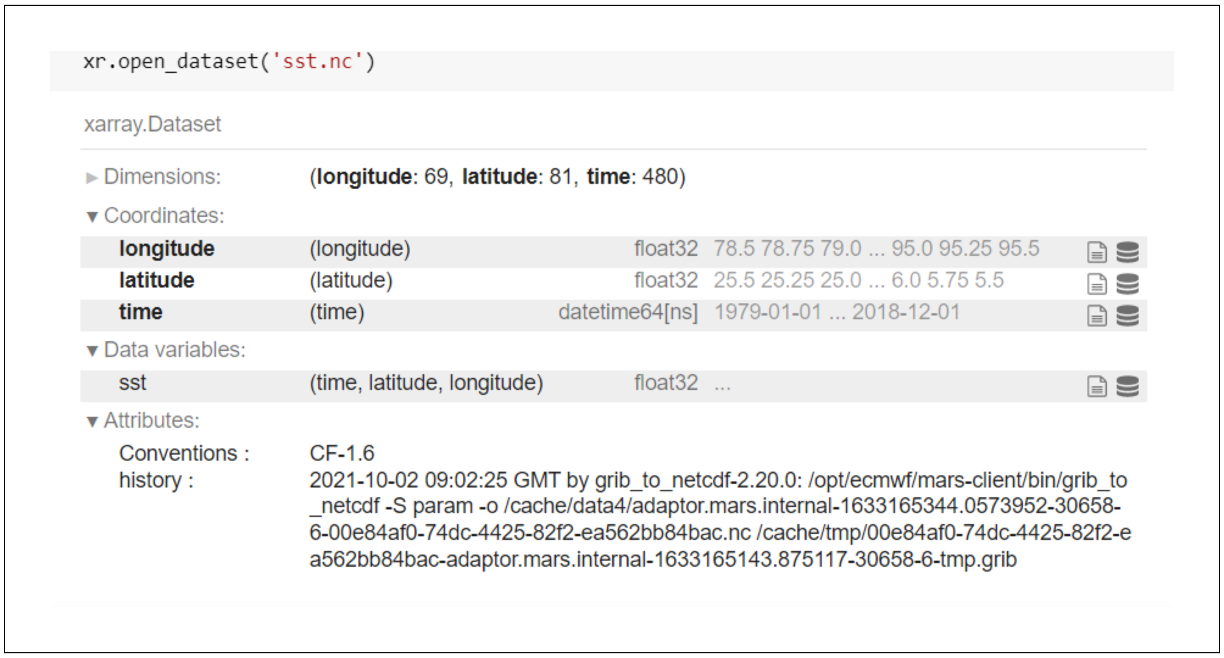
```
netcdf sst {
dimensions:
        longitude = 69 ;
        latitude = 81 ;
        time = 480 ;
variables:
        float longitude(longitude) ;
                longitude:units = "degrees_east" ;
                longitude:long_name = "longitude" ;
        float latitude(latitude) ;
                latitude:units = "degrees_north" ;
                latitude:long_name = "latitude" ;
        int time(time) ;
                time:units = "hours since 1900-01-01 00:00:00.0" ;
                time:long_name = "time" ;
                time:calendar = "gregorian" ;
        short sst(time, latitude, longitude) ;
                sst:scale_factor = 0.00018888458987266 ;
                sst:add_offset = 298.969510049893 ;
                sst:_FillValue = -32767s ;
                sst:missing_value = -32767s ;
                sst:units = "K" ;
                sst:long_name = "Sea surface temperature" ;

// global attributes:
                :Conventions = "CF-1.6" ;
                :history = "2021-10-02 09:02:25 GMT by grib_to_netcdf-2.20.0: /opt/ecmwf/mars-client/bin/grib_to_netcdf
}
```

Figure 3.1: The figure shows the details NetCDF datafile holds

The documentation of the NetCDF data format can be referred to view all the variables stored in it with the most important being the dimension name, dimension sizes, variable name, fill values, missing values brief and many more. The coordinate variables are mostly latitude, longitude and time, however, we can rename them as per requirement. The main variable, say, MLD in our case is saved as MLD (latitude, longitude, time). Some of the benefits of using the NetCDF data format are that it is self-describing in nature as it contains all metadata information. The dataset is scalable as any part of the data can be easily accessed and modified as per requirement. The dataset is easily appendable as new data can be replaced or added to the existing.

Even though the NetCDF package in Python has a modification option, 'Xarray', an open source package in Python, is commonly used due to its extensive customization option. This package is entirely based on NumPy and even has similar commands. Xarray makes processing and manipulation of labelled multi-dimensional data efficient and straightforward. Apart from Numpy, Xarray borrows its features from Pandas, especially for Tabular data. When it comes to parallel computing for large-sized data, it incorporates Dask very well making it one of the most popular packages in Python for climate data.



```
xr.open_dataset('sst.nc')
```

```
xarray.Dataset
▸ Dimensions:       (longitude: 69, latitude: 81, time: 480)
▾ Coordinates:
    longitude       (longitude)                 float32   78.5 78.75 79.0 ... 95.0 95.25 95.5
    latitude        (latitude)                  float32   25.5 25.25 25.0 ... 6.0 5.75 5.5
    time            (time)               datetime64[ns]   1979-01-01 ... 2018-12-01
▾ Data variables:
    sst             (time, latitude, longitude) float32   ...
▾ Attributes:
    Conventions :   CF-1.6
    history :       2021-10-02 09:02:25 GMT by grib_to_netcdf-2.20.0: /opt/ecmwf/mars-client/bin/grib_to
                    _netcdf -S param -o /cache/data4/adaptor.mars.internal-1633165344.0573952-30658-
                    6-00e84af0-74dc-4425-82f2-ea562bb84bac.nc /cache/tmp/00e84af0-74dc-4425-82f2-e
                    a562bb84bac-adaptor.mars.internal-1633165143.875117-30658-6-tmp.grib
```

Figure 3.2: The figure shows data description using xarray

Figure 3.2 can be referred to understand the data description displayed by

Xarray. The data variable and coordinate variables with their sample data points can be seen. *xr.open_dataset* function is designated for reading single file and to display the data description with variables, coordinate variables and attributes, the 'ds.attrs' and ds.coords functions are used.

3.2 Climate data types

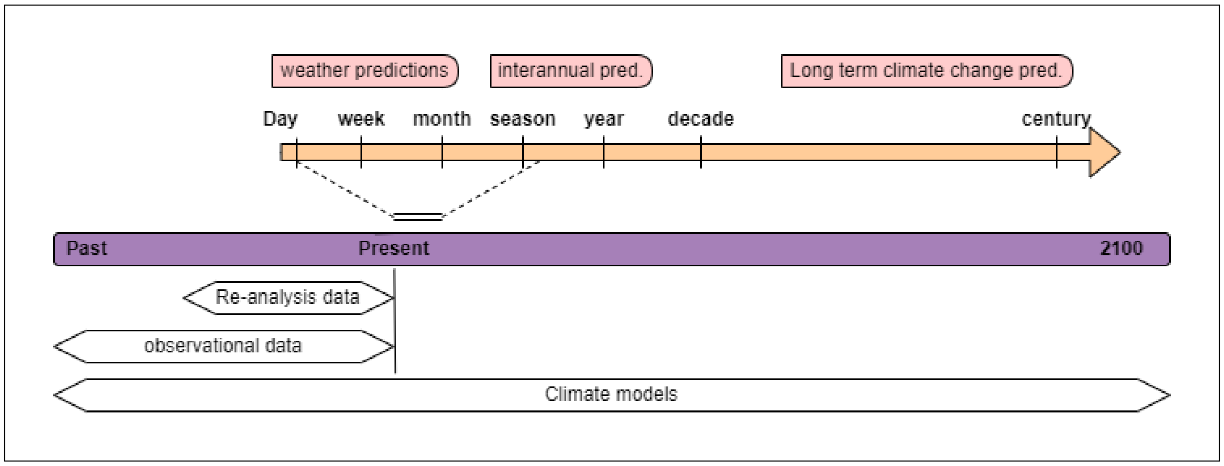


Figure 3.3: The figure shows the range for which different data types exist

Climate data are can be classified into 3 types based on their sources. All the meteorological data comes under Observational, Reanalysis or data by climate models. Figure 3.3 shows the range at which these sources return the meteorological data.

3.2.1 Observational Data

In this category, in-situ observations refer to the data which are directly generated by meteorological stations at/near the earth's surface or inputs from ships, mooring or stationary platforms kept across seas. Indirect observational data refers to the estimates from satellites, RADAR and investigations of ice cores, boreholes temperature profiles and coral reefs which are not accurate but they give insights into historical climate variables. However, inputs from satellites are equally crucial as direct observations. Satellites contain imaging from infrared to the visible spectrum and have sound sensors using which meteorological variables can be determined. AI is also used to understand satellite inputs after correcting interferences from the atmosphere.

3.2.2 Data from Climate models:

The climate model is referred to as the numerical depiction of the ocean and climate system derived from the physics, chemistry and biology of the environment enclosed by it, their exchanges and feedback. These models are, in layman's terms, systems of mathematical(differential) equations that follow the laws of chemistry and physics. The climate model further bifurcates into the weather model which deals with predictions ranging from minutes to a month and a general climate model which can give out data in the range of decades. So if we are looking into the weather of Pune for tomorrow, the weather model would make more sense as the climate model deals with gross data of a higher range.

3.2.3 Re-analysis data

A meteorological re-analysis data is generated by the combination of climate models and observations and it gives the most accurate description of climate variables. It contains the estimation from all varieties of atmospheric variables like pressure, temperature, wind and surface parameters like precipitation, evaporation and further in-sea profiles like MLD. A global re-analysis data contains the outputs when global model data and observations are fed to get the data and can range back by decades.

3.3 Data Source

3.3.1 ERA5

ERA5 is the most popular repository for referring to meteorological datasets. These data are derived from the Copernicus satellite and they provide hourly to monthly estimates of a huge collection of land, oceanic and atmospheric meteorological variables. This repository contains monthly dataset since 1959 and is updated every quarter in real-time. The initial daily updates are uploaded 6 times a month. As described in the re-analysis section, ERA5 uses a huge amount of historical observational data along with climate modelling to give out the re-analysis data. The parameters namely monthly averaged reanalysis Evaporation, Mean surface latent heat flux, Mean surface net long-wave radiation flux, Mean surface sensible heat flux, Sea surface temperature and Total precipitation were downloaded from ERA5. The dataset was already in 0.25° resolution and was used as it is.

3.3.2 ORAS5

ORAS5 dataset is the sea-ice and global ocean data ensemble reanalysis released by the Ocean Reanalysis System 5 or ORAS5 and this dataset is essentially the monthly mean. It provides historical data since 1979 [20]. The parameters Potential temperature, Salinity, Mixed Layer Depth and Net Upward Water Flux were downloaded from ORAS5 Reanalysis data. Potential temperature and Salinity are 3d variables varying with depth (Leviation) and they denote the stratification of the ocean surface. The combined data size ended up being around 200MB. The TropFlux provides surface heat and momentum flux data of tropical oceans (30°N-30°S) between January 1979 and September 2011.

3.3.3 Tropflux

The TropFlux provides Surface Heat and Momentum Flux data of tropical oceans (30°N - 30°S) since January 1979 [21]. The parameter wind stress magnitude was downloaded from the Tropflux dataset. This dataset is considered to be the most reliable for flux-related modelling.

3.4 Data Overview

All the datasets were downloaded for the Bay of Bengal region. The grid for the dataset was 5 °N to 25 °N latitudes and 78 °E to 96 °E longitudes. The Surface Heat and Momentum Flux from Tropflux could only be downloaded for the complete ocean and so it was cropped using the NCO package. One dataset from each of the repositories is discussed with figures plotted using Panoply [22].

3.4.1 Training Data: Net non-solar heat flux

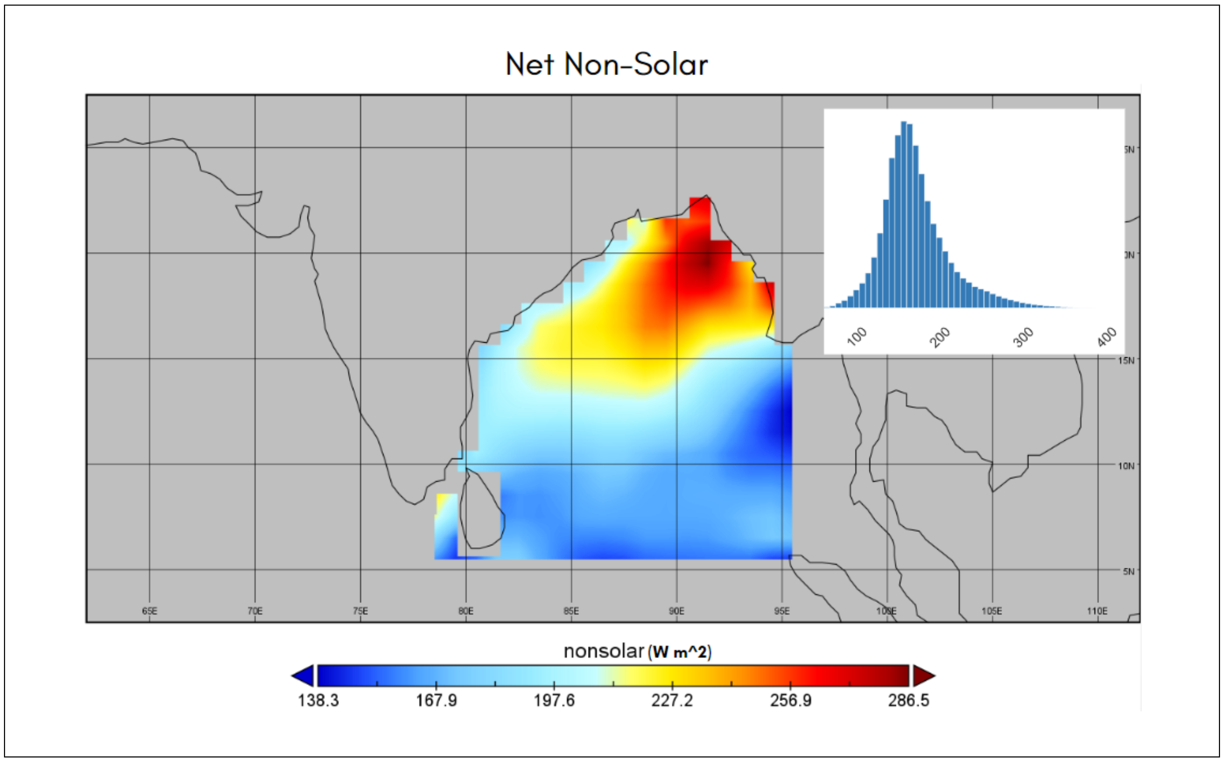


Figure 3.4: net non-solar in Jan 1979 in the BOB region and the histogram (right-top) shows the data distribution for the whole dataset

The Net Non-Solar data had no missing values and showed a very uneven distribution of values. These refer to the heat released by the ocean's surface and was calculated by summing the Mean surface latent heat flux, Mean surface net long-wave radiation flux and Mean surface sensible heat flux using an in-house script. The initial impression of the data for the month Jan 1979 has been shown in the figure 3.4. The dataset has 40.2% missing values corresponding to the land area enclosed in the rectangular grid cropped for the BOB region. The right-top corner of the figure 3.4 shows the distribution of the data sample. This distribution is shown in every variable discussed further. The maximum and minimum values of the dataset were 394.2 and 38.1. These datasets were also downloaded from the ERA5 repository for the selected BOB region and had a resolution of 0.25°.

3.4.2 Training Data: Total Precipitation

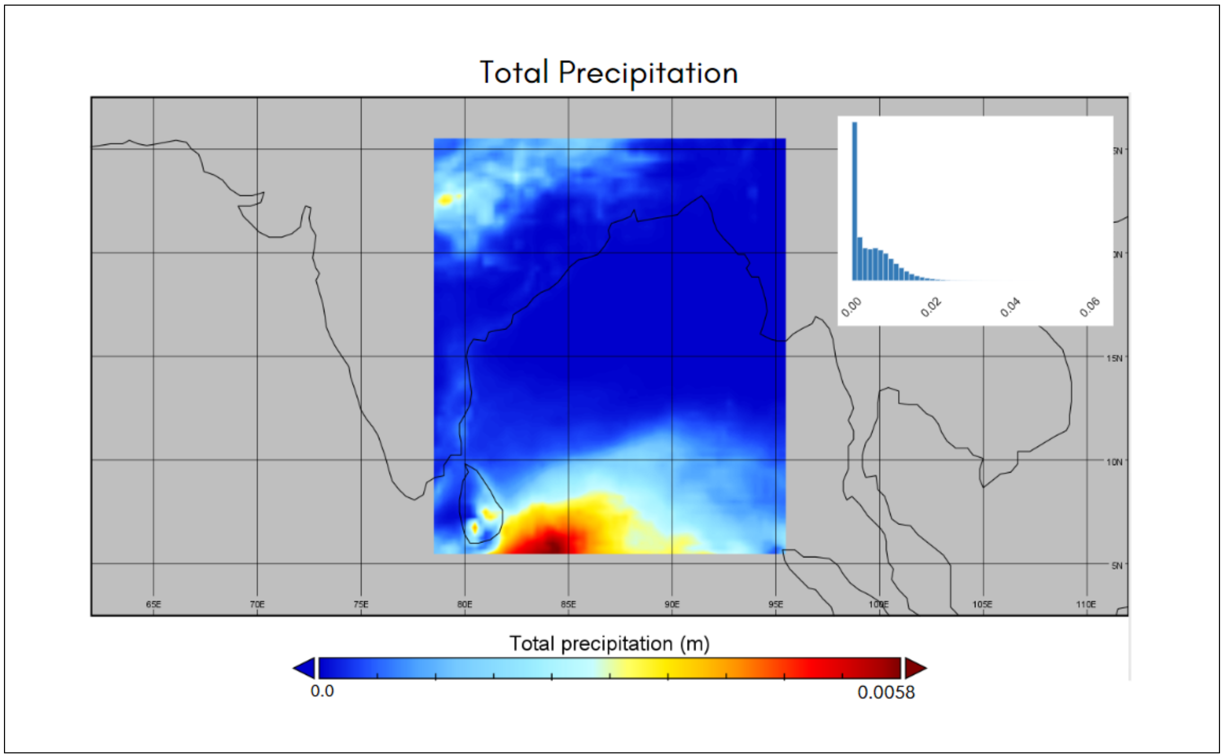


Figure 3.5: Total precipitation in Jan 1979 in the BOB region and the histogram (right-top) shows the distribution of whole the dataset

Precipitation data was downloaded from the ERA5 repository and had a resolution of 0.25°. Figure 3.5 shows how the data looks for Jan 1979. [23] shows how precipitation can influence the Mixed Layer Depth and hence has been included in this study. The dataset has no missing value as the data for precipitation is available for both land and sea.

3.4.3 Training Data: Wind stress Magnitude

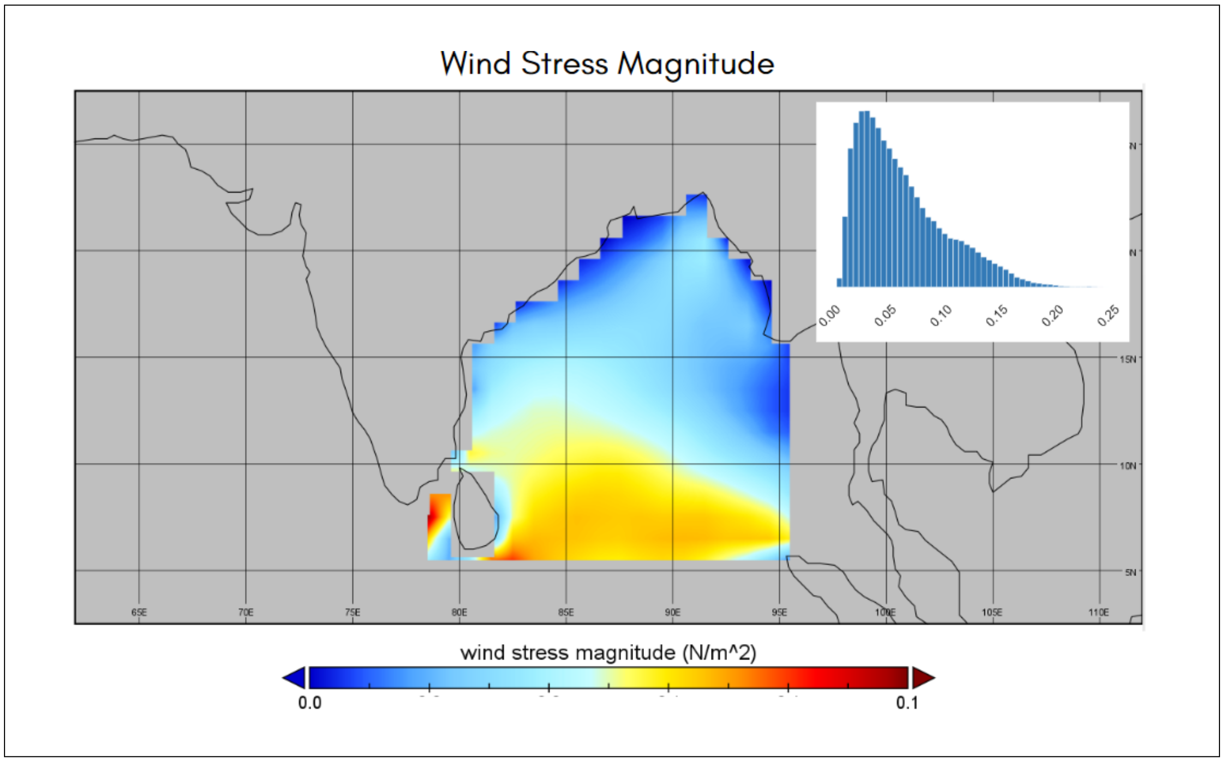


Figure 3.6: Wind stress magnitude in Jan 1979 in the BOB region and the histogram (right-top) shows the distribution of the whole dataset

Wind Stress magnitude denotes the impact of wind mixing on the Mixed Layer Depth. This data was downloaded from the Tropflux repository and was in the resolution of 1°. Figure 3.6 shows how the data looks for Jan 1979. The dataset was then interpolated with a linear interpolation method using in-house python script. The dataset had minimum and maximum values of 0.0058 and 0.2544. and contains 40.2% missing values corresponding to the land enclosed.

3.4.4 Training Data: Evaporation

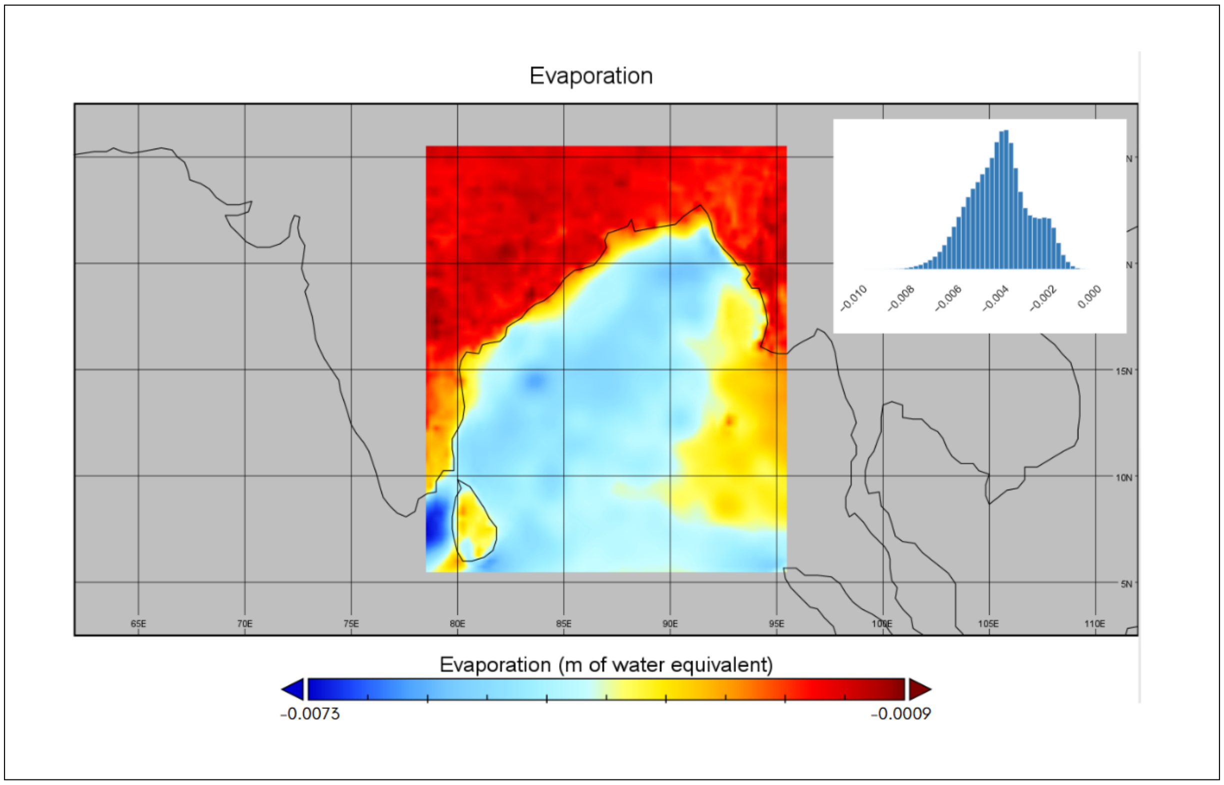


Figure 3.7: evaporation in Jan 1979 in the BOB region and the histogram (right-top) shows the distribution of the whole dataset

The data from ERA5 had 0.25° resolution by default and was used as it is.Figure 3.7 shows how the data looks for Jan 1979. The data were negative denoting the amount of water lost due to evaporation. The data would be normalised for training purpose. The data had no missing value as land surface also contributes to evaporation.

3.4.5 Training data: Sea surface Temperature

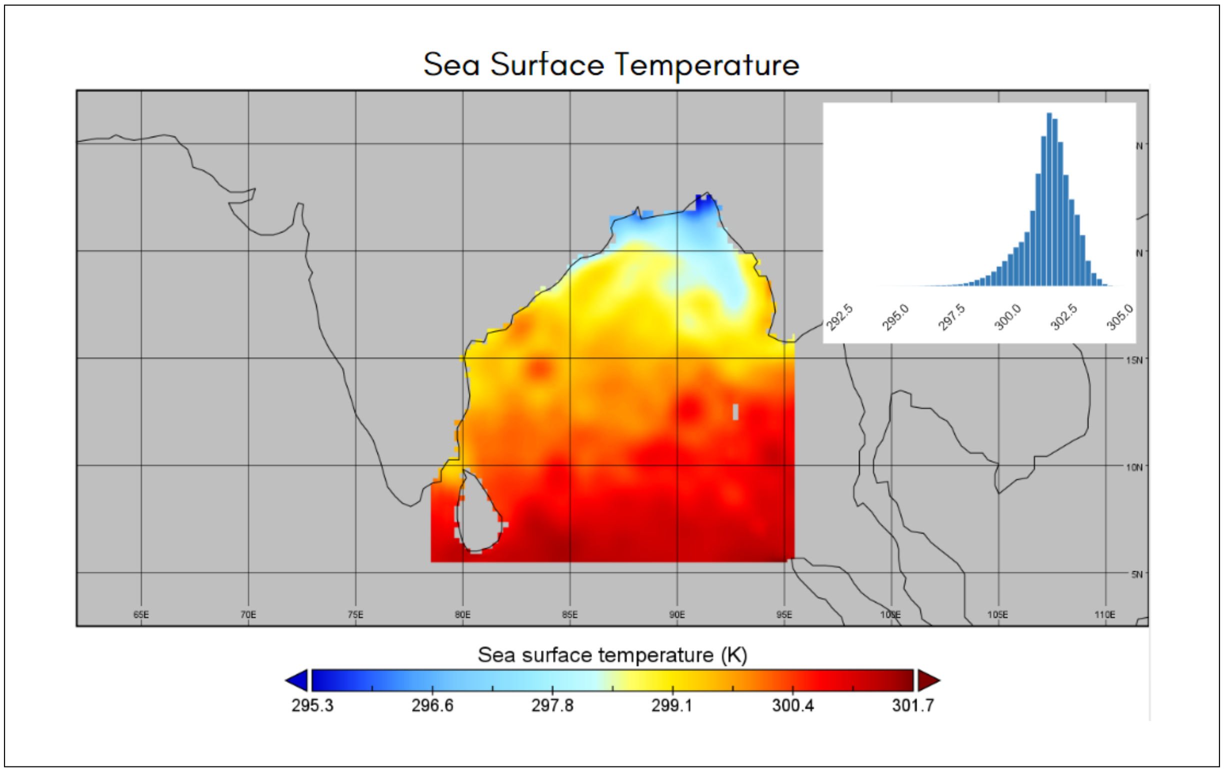


Figure 3.8: Sea surface temperature in Jan 1979 in the BOB region and the histogram (right-top) shows the distribution of the whole dataset

The surface temperature is a direct measure of heat content in the Mixed Layer of any ocean. Figure 3.8 shows how the data looks for Jan 1979. This data was downloaded from the ERA5 dataset and was present in 0.25° resolution. The dataset had 37.2% missing values corresponding to the land surface enclosed in the BOB rectangular grid. The data ranged from 292.78 to 305.16.

3.4.6 Testing Data: Mixed Layer Depth

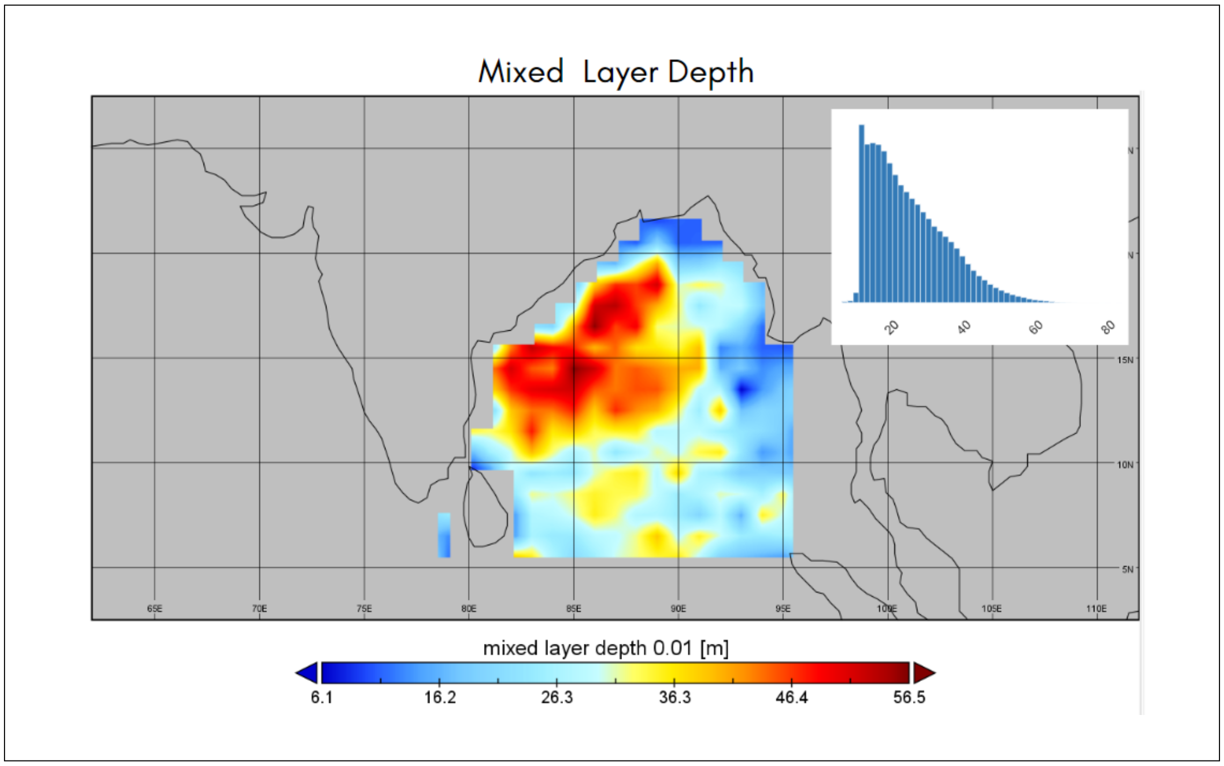


Figure 3.9: Mixed Layer Depth in Jan 1979 in the BOB region and the histogram (right-top) shows the distribution of the whole dataset

Mixed Layer Depth is the main phenomenon this study revolves around. As mentioned earlier, the study will aim to show how the parameters influence it in a data-driven approach. The monthly averaged reanalysis data of MLD was downloaded from the ORAS5 repository. The data was in 0.25° resolution and was used directly after cropping it into our required grid. The MLD dataset had 44.9% missing values of which the major component should be from the land surface. It is also possible that, as this data is taken from the ORAS5 repository the data stored treats more part of the map as land surface compared to ERA5. They ranged from 6.08m to 83.55m metres. The higher value of MLD corresponds to the central part of the BOB during the winter season. Figure 3.9 shows the MLD in the geographical plot for Jan 1979.

3.5 Data Pre-processing

For any AI model, it becomes important to input the data in a machine-readable format and compensate for missing values or outliers. The below figure 3.10 shows the flow chart for the data processing pipeline described in the next subsection.

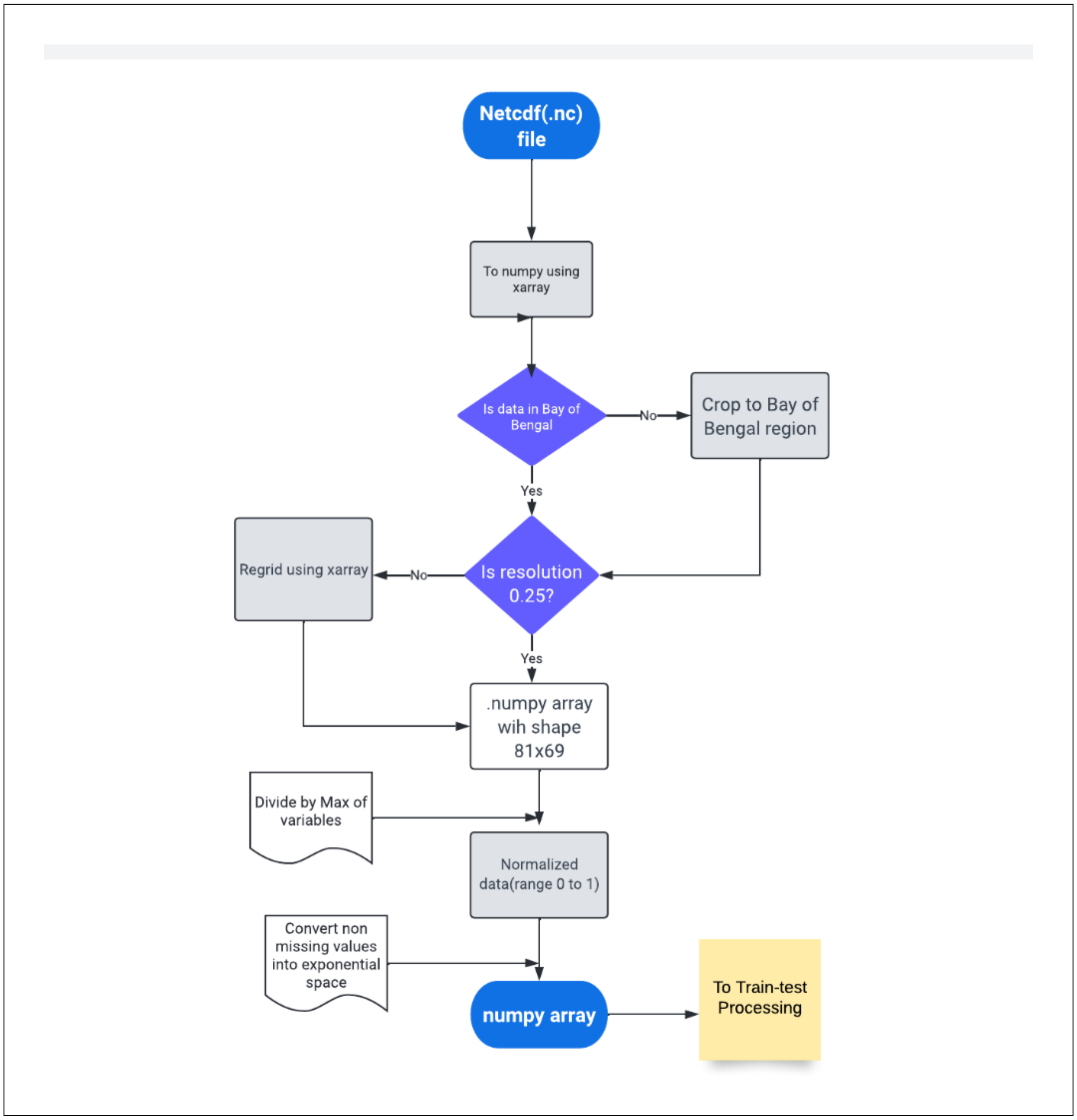

Netcdf(.nc) file
To numpy using xarray
Is data in Bay of Bengal
No
Crop to Bay of Bengal region
Yes
Regrid using xarray
No
Is resolution 0.25?
Yes
.numpy array wih shape 81x69
Divide by Max of variables
Normalized data(range 0 to 1)
Convert non missing values into exponential space
numpy array
To Train-test Processing

Figure 3.10: Flow chart showing pre-processing steps

3.5.1 Pre-processing pipeline

1. All the datasets were cropped using 'Xarray' to the bay of Bengal region(5°N to 25.5° & 78.5°E to 95.5°E) which is the focused region in this study.

2. The dataset was then checked for resolution and regrid to 0.25° using Xarray. In our case, MLD was at 1° resolution and the rest at 0.25° resolution. Now every dataset had a shape of 81x69.

3. The dataset was then viewed in python and evaporation data which was fully negative was converted to non-negative.

4. All non-missing values were stored separately from missing and treated further.

5. In our study, all the variables were non-negative and were then normalized using min-max scaling.

6. Using the formula:
$$z = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.1}$$
where z represents the normalized value, the dataset will always be in the range of -1 to 1 (0 to 1 in our study)

7. The real dataset was then converted into e^7 exponential space and the missing values were changed to 0. This is to nullify the biasing of the missing values on model prediction. Especially, in our study, most of the variables are limited to the ocean and contain huge missing values corresponding to land. This dataset was then sent to the train-test pipeline for further editing as discussed in Chapter 2

Chapter 4

Methodology

4.1 Insights into ConvLSTM model

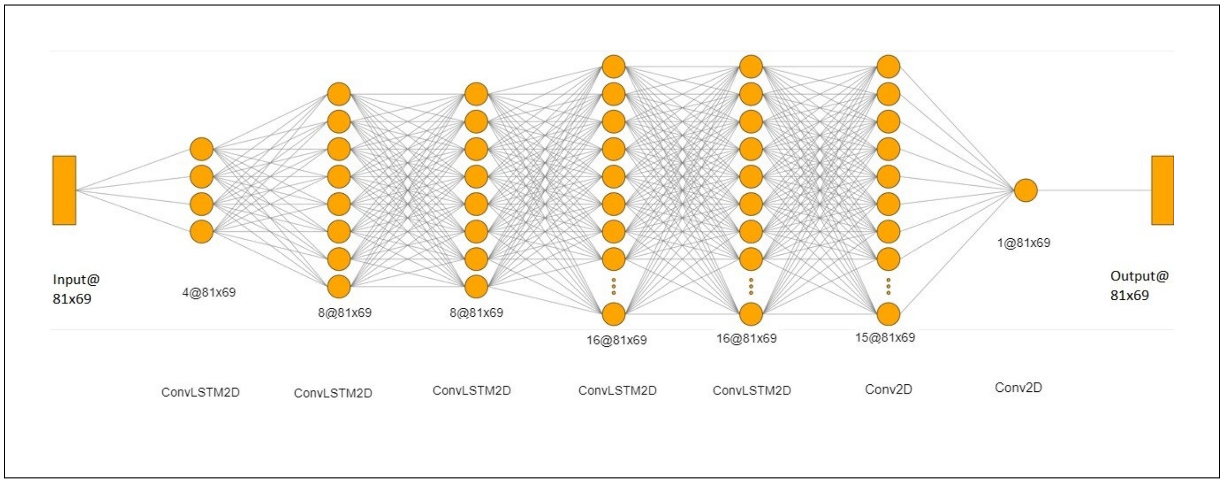


Figure 4.1: Model architecture

Current advances in deep mastering, in particular, recurrent neural network (RNN) and Long short-term memory (LSTM) model, as mentioned in the papers by Alex Graves[24] and Hochreiter et al.[25], shows that these models are highly effective. According to the philosophy underlying the deep studying technique, if we've got sufficient data to train it, we are close to resolving the problem. Our study satisfies the data problem as it was easy to acquire a large amount of radar echo information continuously and combined with the climate model and Argo observations, a re-analysis data was formed. Now the requirement is an appropriate version of end-to-end learning. The original LSTM encoder-decoder framework proposed in the article

Sutskever et al.[26] gives a well-defined framework for sequence-to-sequence learning problem by training temporally joined LSTMs, one each for input and output sequence. The article Ranzato et al.[27] shows that RNN based model on visual data of words obtained by studying the image patches can do the interpolation of intermediate frames and prediction of future video frames. LSTMs, one for the input sequence and the other for the output series. This model could predict only one frame ahead as the size of the convolutional kernel for transitioning from state to state is restricted to 1. In the article Srivastava et al.[28], the latter work is followed up by focusing on the importance of multi-step prediction. Their model which reconstructs the input sequence and predicts the future sequence simultaneously can be used for our purpose but it lacks the use of spatiotemporal correlation. This issue is solved in Shi et al.[17] where a novel convLSTM network is proposed for precipitation nowcasting. The idea of C-LSTM is extended to ConvLSTM by adding convolutional layers in both input-to-state and state-to-to transitions. Similarly, By stacking multiple Convolutional LSTM layers and forming an encoding-forecasting structure, an end-to-end trainable model can be built for Mixed Layer prediction forecasting and understanding of the dynamics. Figure 4.1 shows our model architecture, the details of which are shown in the summary of the model in section 4.5. Further, Figure 4.2 shows the flowchart for the progress of the study.

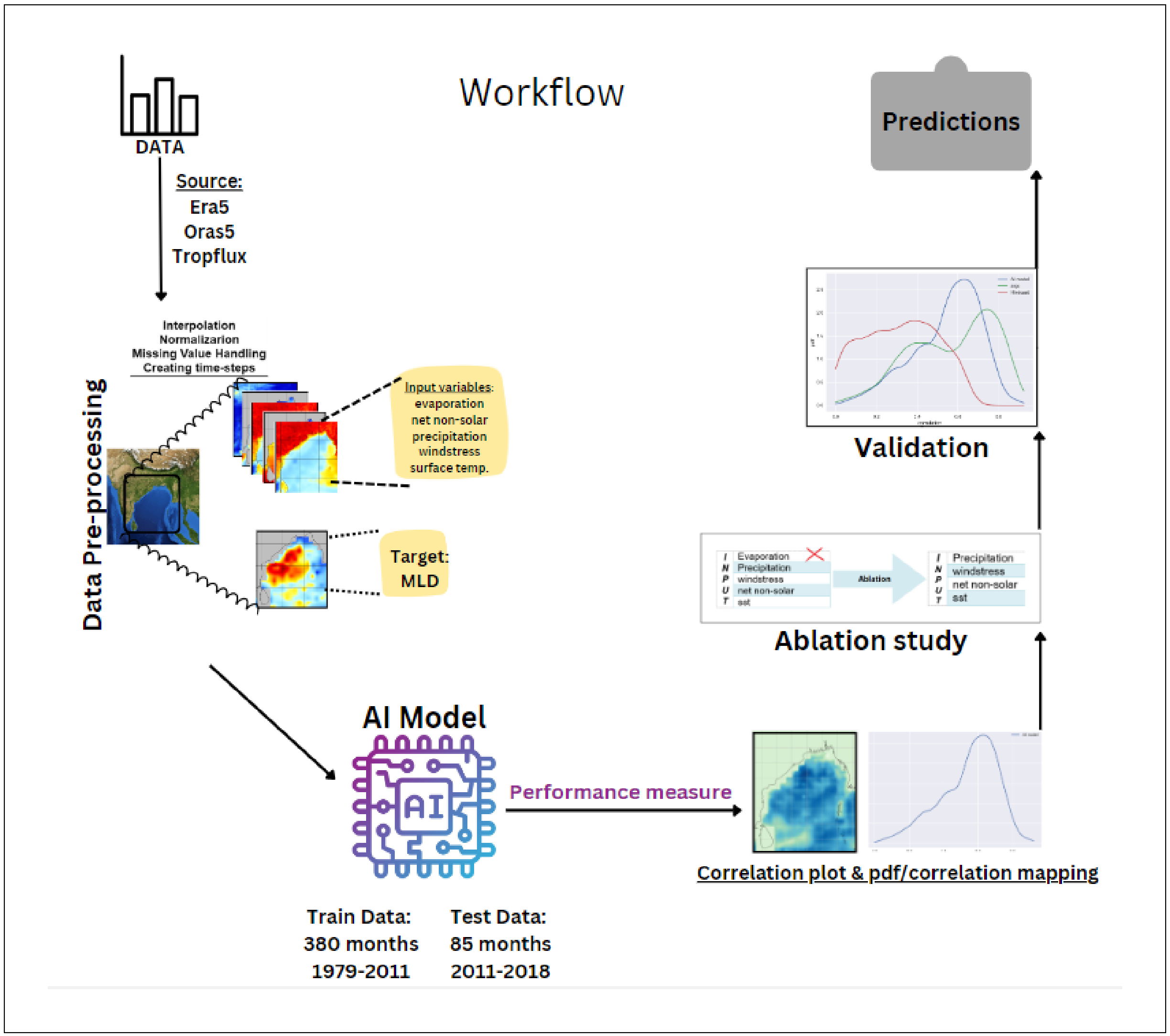


Figure 4.2: Flow Chart for the MLD forecasting using the AI model

4.2 Model Training

The dataset used for the model training consisted of 380 months from each of the input and output variables. The input variables were further converted in the sliding window of 15 months. In the model, we feed 15 months of data to the ConvLSTM model and predict the 1 lead month and the sliding window, as explained below.

4.2.1 The Sliding Window

The concept of a sliding window to initialize the training data as shown in figure 4.3 is described in the following points:

- For the training, a dataset containing 395 months starting Jan 1979 to Nov 2011 was used
- A deep learning model uses the target variable throughout the training for validation.
- As seen in the figure 4.3, the 1st window consists of 15 months of training variables, i.e., Evaporation, precipitation, sea surface temperature, wind stress magnitude and net non-solar heat flux and the 16th month MLD value as predicted and validation data.
- the 1st data point, say, Input1 has months 1-15 as input, Input2 has Months 2-16 as input and Input 380 has months 380-394 as input and accordingly month 16, month 17 and month 394 as predictions and validation data.
- Hence the input data has a dimension of 380 x 15 x 81 x 69 x 5 [no. of data points, x depth of sliding window x latitude x longitude x no. of channels] and the prediction has a dimension of 380 x 1 x 81 x 69 in the same schema.

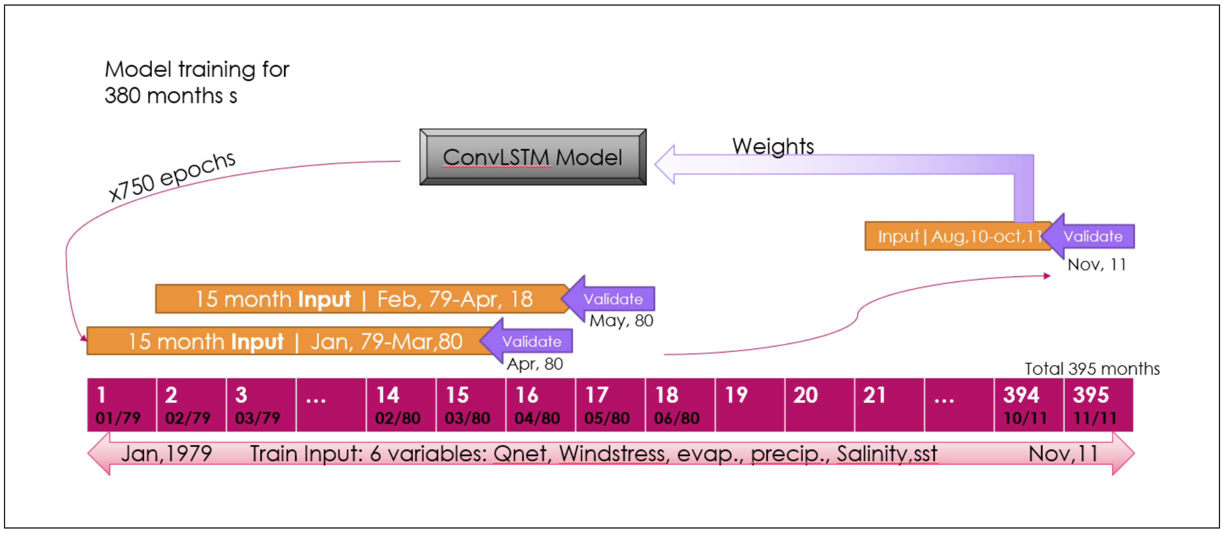


Figure 4.3: Iterative sliding window taking 15 months as input data and 16^{th} month as validation data

4.2.2 Schematics

The data downloaded from the mentioned sources (Section 3) were passed through preprocessing stages. As we have our target variables, it ends up to a supervised learning problem for which we split our data into train and test data in the ratio of 380:85 with the numbers representing the number of months in each set. The deep learning model by default will split the training data further in training and validation sets throughout the training loops. The deep learning model iterates over the data to create multiple splits such that all parts of the data are used for training and validation. Each of these data points has values of 15 months corresponding to each variable or channel and MLD as test data. The data is then sent into the described (Section 4.1) AI model for training. The ConvLSTM model is designed to incorporate multiple channels or variables in the input. This regression model is able to identify the non-linear trend and the relations between the input and target variables, MLD in this case. The training data contains 380 datasets and the test data contains 85 datasets as mentioned. Each dataset contains 15 months of data with 14 months overlapping with the next set. This makes the total months used in the study as 480(380+15+85). Hence, data from the period Jan 1979 to Nov 2011 is used for training and data from the period Dec 2011 to Dec 2018 is used for the testing. Further, in the ablation study, multiple models with varying input variables will be trained and due to a large number of training sets, the models were trained on 3 GPU's namely, Nvidia v100 and Intel Silverlake from Gadi supercomputer under NCI Australia, Nvidia A100 in Pratyush HPC in IITM Pune and XC50 in the Pratyush HPC in IITM Pune. Each of the models was run for 1000 epochs. The training time for each epoch varied from 20 seconds to 70 seconds depending on the number of variables and GPU used. The models were trained parallelly on multiple GPUs to save time and the best model with the least validation loss was saved to avoid overfitting.

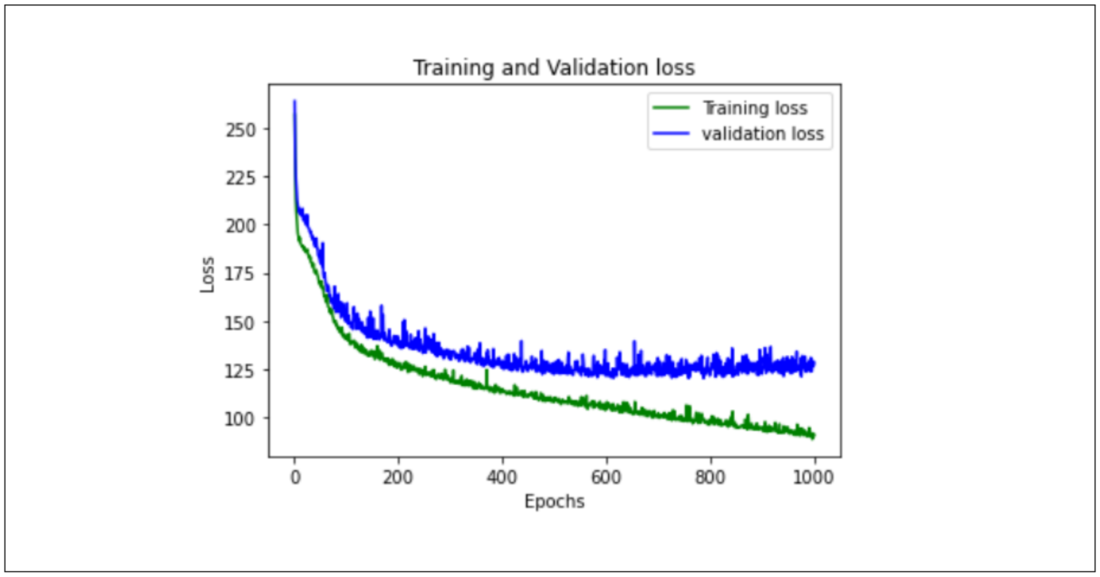


Figure 4.4: Learning curve

The learning curve in figure 4.3 suggests that the model converged before the completion of 1000 epochs and the model saved the system saved the model corresponding to the least 'validation loss'. This trend of converging at around 500 epoch was seen in nearly all subsequent models in the ablation study. The model corresponding to the least validation loss in every combination of input variables was then used to predict the MLD in the test data range and was then compared with ground truth, MLD from re-analysis data.

4.3 Model Testing

The trained model saved in the hierarchical Data format(H5) was loaded and used on the test data to get the prediction and was scored to get the performance measure. The testing was also done with data in the sliding window format as shown in figure 4.4. The test data contained 85 data points and the input variables 100 months of data starting Sep 2010 in the set of 15 months. The target and predicted values were from Dec 2011 to Dec 2018. The predicted values and the ground truth were used for scoring the performance of the model.

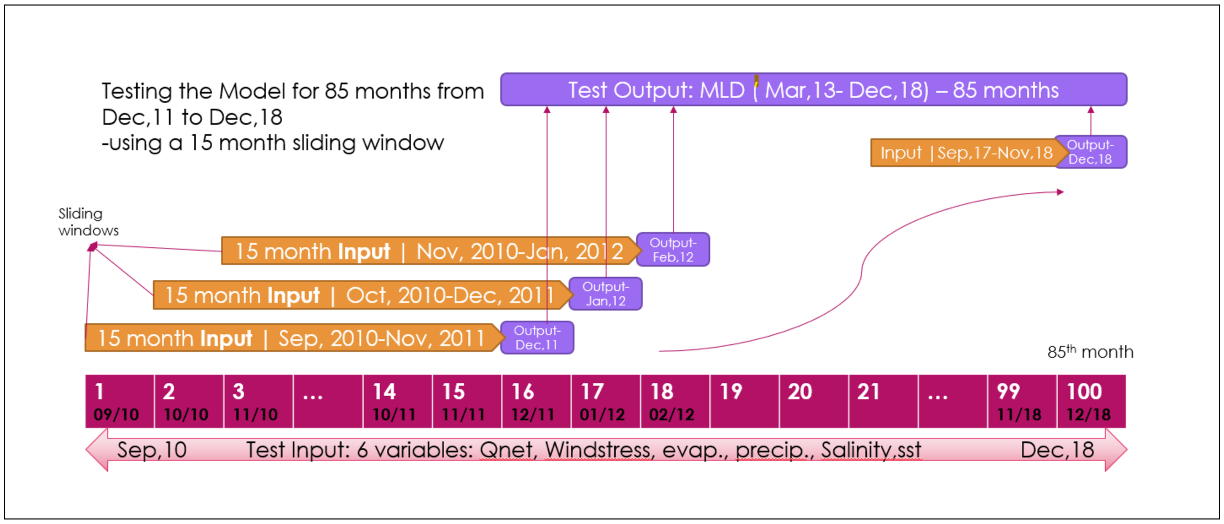

Testing the Model for 85 months from
Dec,11 to Dec,18
-using a 15 month sliding window
Test Output: MLD (Mar,13- Dec,18) – 85 months
Input | Sep,17-Nov,18
Output-Dec,18
Sliding windows
15 month Input | Nov, 2010-Jan, 2012
Output-Feb,12
15 month Input | Oct, 2010-Dec, 2011
Output-Jan,12
15 month Input | Sep, 2010-Nov, 2011
Output-Dec,11
85th month
1 09/10
2 10/10
3 11/10
...
14 10/11
15 11/11
16 12/11
17 01/12
18 02/12
19
20
21
...
99 11/18
100 12/18
Sep,10
Test Input: 6 variables: Qnet, Windstress, evap., precip., Salinity,sst
Dec,18

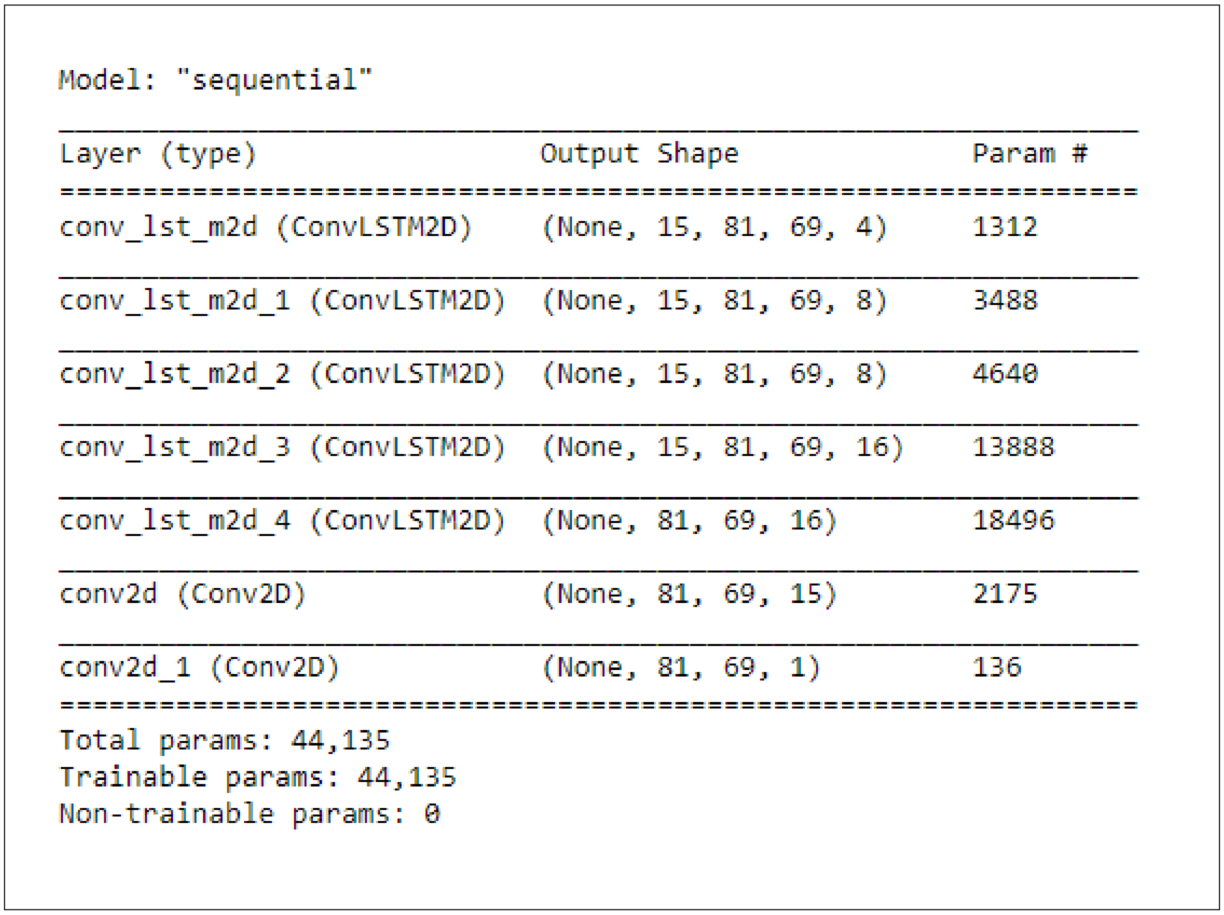
Figure 4.5: Iterative sliding window taking 15 months as input data and returning16th month as prediction

4.4 Model Parameters

- Data Variables: Evaporation, precipitation, sea surface temperature, wind stress magnitude and net non-solar heat flux (Multi-variable input).
- Data Frequency: Monthly
- Data Resolution: 0.25 $\circ$ spatial resolution data enclosed by 78.5E to 95.5 E and 5.5N to 25.5N
- Data Pre-processing: As described in Chapter 3
- I/O Sequence: 15 input time steps corresponding to 15 input days for 5 variables, and 1 successive day of MLD as lead day for output.
- Train/Validation Split: Jan 1979 to Sep 2011 for model training and validation in the ratio 70:30 iteratively over whole data
- Test Set: The last 85 data samples were used as the test set and used in the final model performance evaluation.
- Models: Direct forecasting using convolutional LSTM-based layers, followed by Conv2D layer for 2D output.
- Architecture: As shown in figure 4.1
- Loss: Mean Squared Error

- Optimizer: Adam with 0.0001 learning rate.
- Activation: ReLU
- Epochs: Fixed 1000 epochs (from the learning curve it was seen that the validation loss was minimum near 500 epoch)
- Evaluation Metrics: Pearson Correlation Coefficient
- Results: Chapter 5

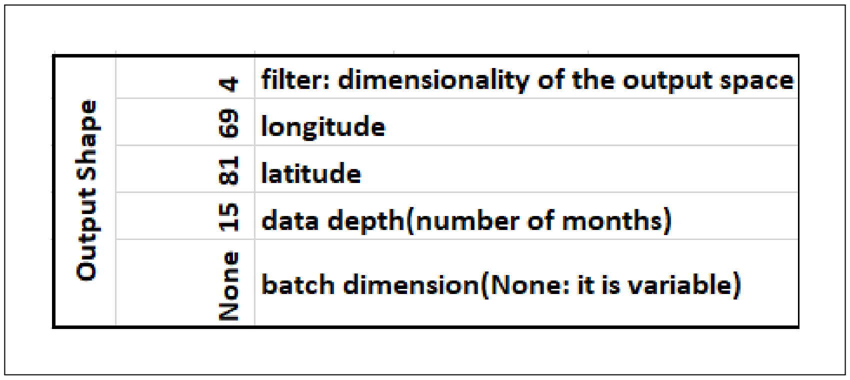
4.5 Model Summary



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv_lst_m2d (ConvLSTM2D)    (None, 15, 81, 69, 4)     1312
_________________________________________________________________
conv_lst_m2d_1 (ConvLSTM2D)  (None, 15, 81, 69, 8)     3488
_________________________________________________________________
conv_lst_m2d_2 (ConvLSTM2D)  (None, 15, 81, 69, 8)     4640
_________________________________________________________________
conv_lst_m2d_3 (ConvLSTM2D)  (None, 15, 81, 69, 16)    13888
_________________________________________________________________
conv_lst_m2d_4 (ConvLSTM2D)  (None, 81, 69, 16)        18496
_________________________________________________________________
conv2d (Conv2D)              (None, 81, 69, 15)        2175
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 81, 69, 1)         136
=================================================================
Total params: 44,135
Trainable params: 44,135
Non-trainable params: 0
```

Figure 4.6: Summary of the ConvLSTM model

Figure 4.6 shows the summary of the model used for training our initial set. The Output shape shows how the data looks at every layer output. It can be noticed that in 1st 5 layers, upscaling is taking place and the number of output filters is increased up to 16.



Output Shape		
	4	filter: dimensionality of the output space
	69	longitude
	81	latitude
	15	data depth(number of months)
	None	batch dimension(None: it is variable)

Figure 4.7: Description of the input/output shape

These upscaling layers have 5 dimensions, as seen in Figure 4.6 and Figure 4.7, of which 2nd dimension denotes the sliding window length or 'data depth', 3rd dimension denoted the latitude, 4th denotes the longitude and the last dimension denotes the number of output filters in every layer. As evident in Figure 4.6, the final output has a structure of a single image with a single output filter and can be directly plotted into a geographical map.

4.6 Evaluation Metric

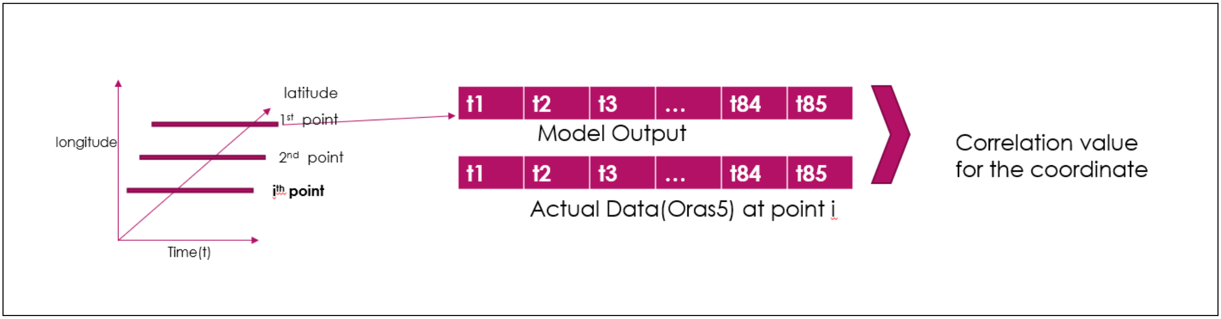


Figure 4.8: The data structure used to feed the correlation matrix

The evaluation metric used in this study is the Pearson correlation coefficient. The correlation is computed for the time series predictions of the input and output samples at all locations as shown in figure 4.8. As the predictions are themselves 2-D spatial images, the correlation coefficient is computed for the corresponding time series for each spatial point, i.e., corresponding to each coordinate in the geographical map, across the samples and is represented as a correlation matrix of the same shape as the input grid shape. In order to understand improvements in the correlation over various model testing, we compare the respective correlation density distributions of the matrix produced.

Chapter 5

Results

5.1 Preliminary Result

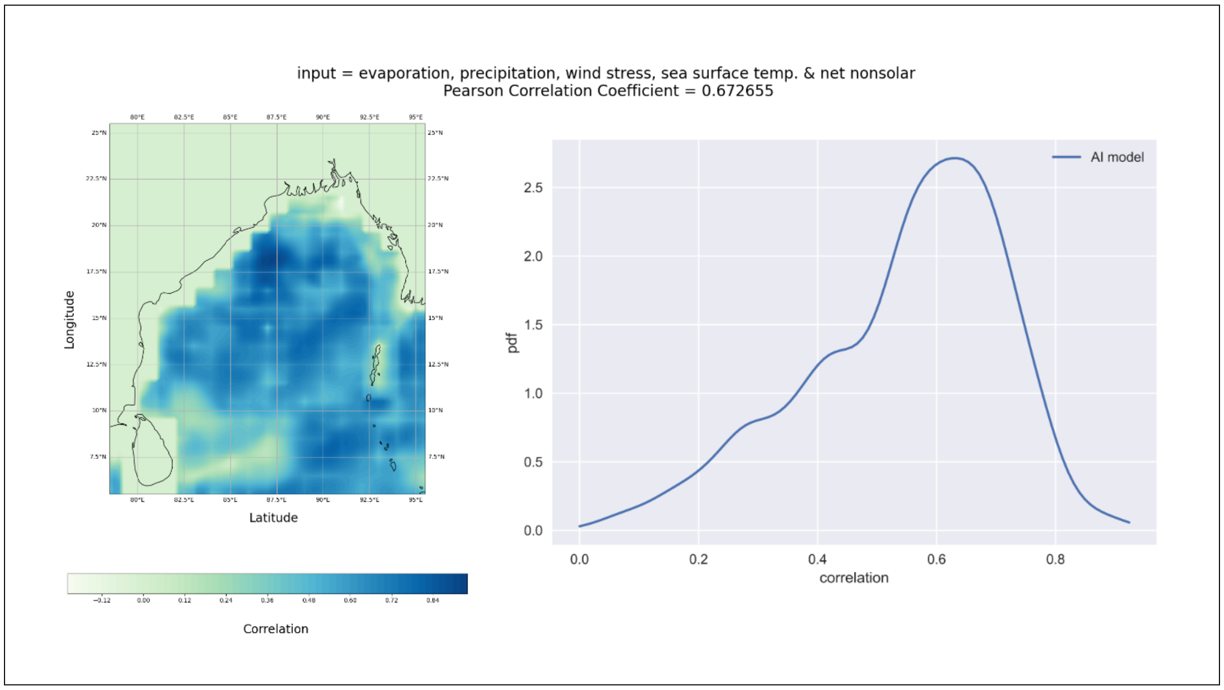


Figure 5.1: geographical plot showing correlation between predictions from the AI and the ground truth (left), and density vs correlation plot for the same (right)

As backed by the ocean dynamics theory, MLD is influenced by several forcings including evaporation, precipitation, sea surface temperature, wind stress on the surface and heat transfers across the surface. This made the base for using these forcings as variables for training the AI model. The parameters used for the compilation and training of the model is described in Chapter 4. Figure 5.1 shows the correlation value when plotted in the geographical map of the Bay of Bengal, and the density vs correlation plot. It is to be noted that correlation value is not a direct performance measure as there is no cut-off to judge the performance of the model. However, further, the model will be subject to ablation study and validation studies using the correlation to be a comparison metric to grade the performance. In each of these plots, as described in section 4.6, the correlation of the predicted data, be it from AI or dynamical model, for every point with a specific latitude-longitude, is calculated and plotted in the geographical map of the Bay of Bengal.

5.2 Ablation Study

The model with all 5 variables as input was then used for the ablation study. The term 'Ablation' originated in the late 19^{th} century in the field of practical neuropsychology where the fragments of the animal brain were surgically dissected to study behavioural change. In the world of machine learning and more specifically complex AI, this term is being reused to demonstrate a practice of altering certain parts of the neural network or modifying the hyperparameters with the aim to get a better understanding or in our case better result from the model. In our study, three sets of experiments have been conducted in the purview of the ablation study. The logic used to work on the layer ablation shares similarity with the Leave one Component out (LOCO) rule that suggests specifying a series of components to be removed or added one by one from the initial system. It is an unusual practice in the academic community to pursue ablation study which turns out to be crucial in validating results before publishing in a repository [29]. Likewise, reporting a good-performing deep learning model could be a blessed coincidence but an ablation study like this study give an opportunity to justify the choice of the model and parallelly suggests the crucial contributing component in the success of the model. It is unfortunate that scientists tend to avoid ablation study as it is time and energy-consuming.

5.2.1 Feature Ablation

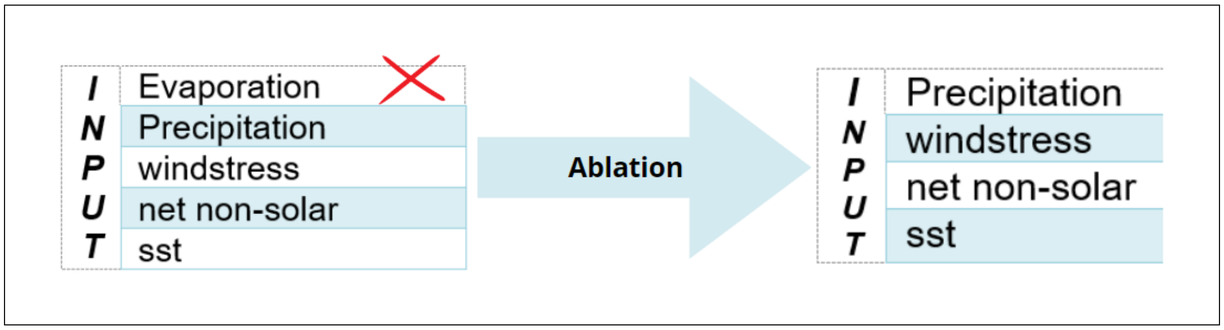


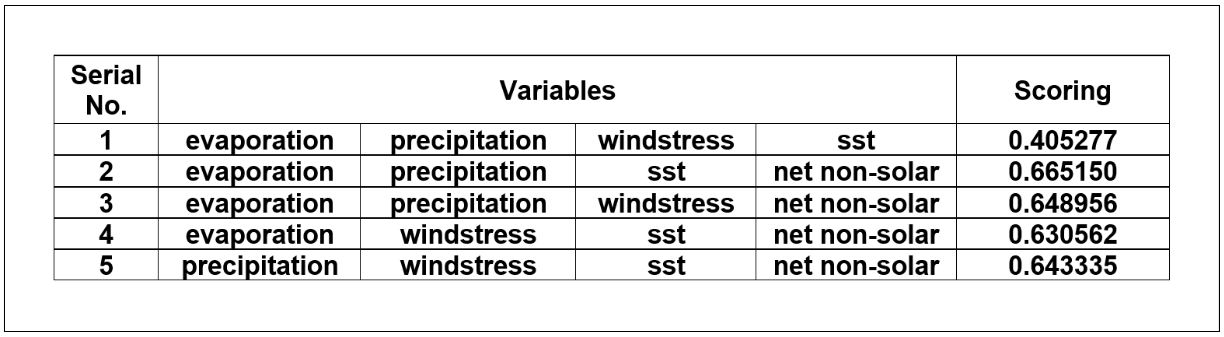
Figure 5.2: Feature ablation: Ablation of 1 channel from input data

The idea of Feature ablation is inspired by the study by Molinari, A.[30] where MAGGY ablator was used in Pytorch to process. Feature ablation refers to the ablation or addition of data type as shown in figure 5.2. In this study, with the aim of understanding the forcing on MLD, different combinations of the forcings were taken as input for the AI model and predictions were noted. Since the total number of input data was 5 and so we ended up with 31 combinations given by simple combination-

$$\sum_{i=1}^{5} = C_i^5 = 31 \tag{5.1}$$

The AI model was run with the same hyperparameters as the initial model for 1000 epochs. The predictions were plotted on the Bay of Bengal geographical plot as done in section 5.1. The average Pearson correlation value was considered to choose the best-performing model. The ablation was done for 4 variations removing 1 feature every time.

Ablation of 1 variable



Serial No.	Variables				Scoring
1	evaporation	precipitation	windstress	sst	0.405277
2	evaporation	precipitation	sst	net non-solar	0.665150
3	evaporation	precipitation	windstress	net non-solar	0.648956
4	evaporation	windstress	sst	net non-solar	0.630562
5	precipitation	windstress	sst	net non-solar	0.643335

Figure 5.3: Correlation values of the predictions from the model with 3 input variables

In this test, single variable was removed and the model was trained with the remaining 4 variables as input. As given by $C^5_4 = 5$, we get 5 different combinations of model inputs. Table 5.3 shows the Pearson Correlation Coefficient value as a performance metric for the AI model predictions with 4 variables as input. The corresponding figure 5.4 shows the correlation values for each of the combinations of 4 variables in the Bay of Bengal map.

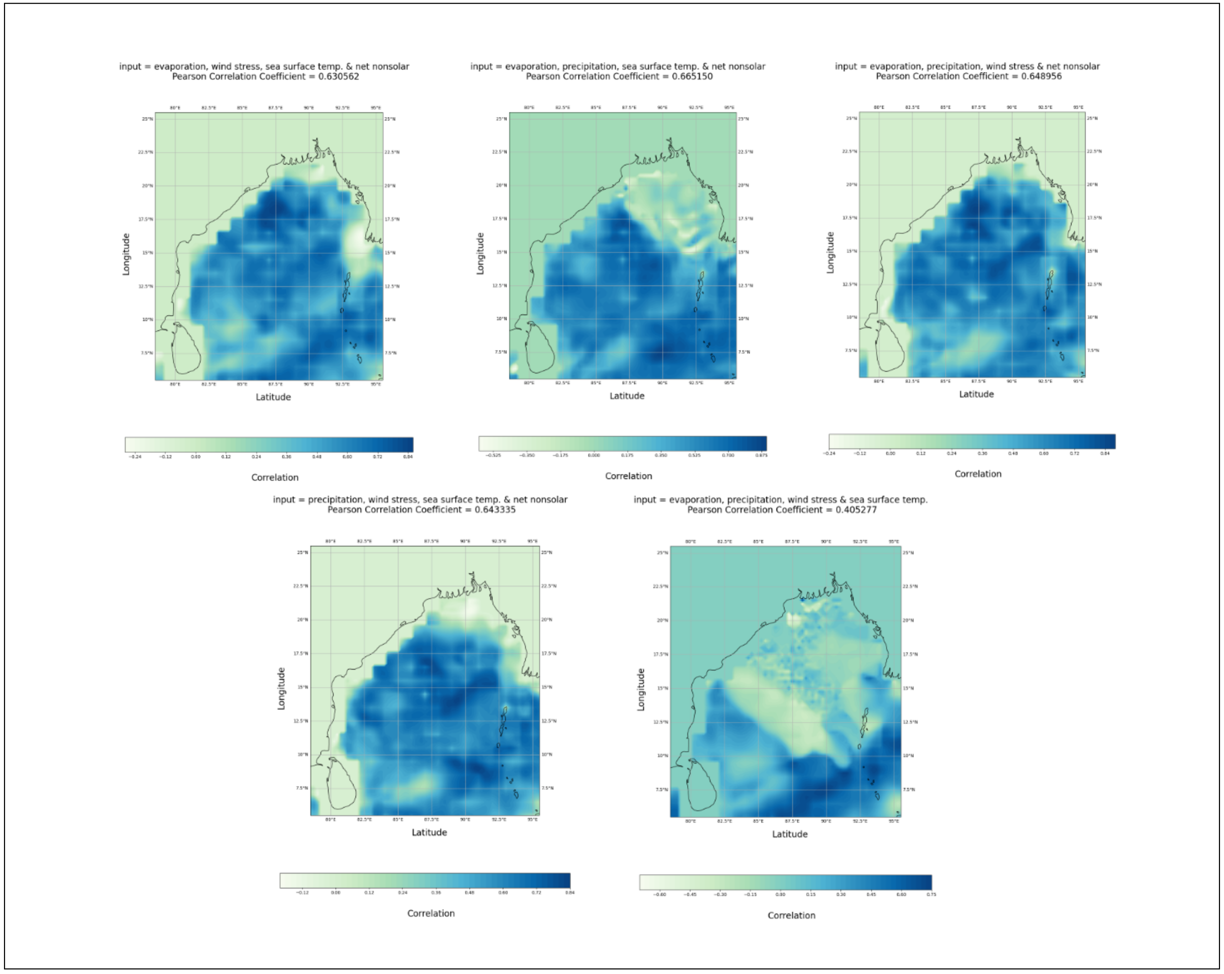
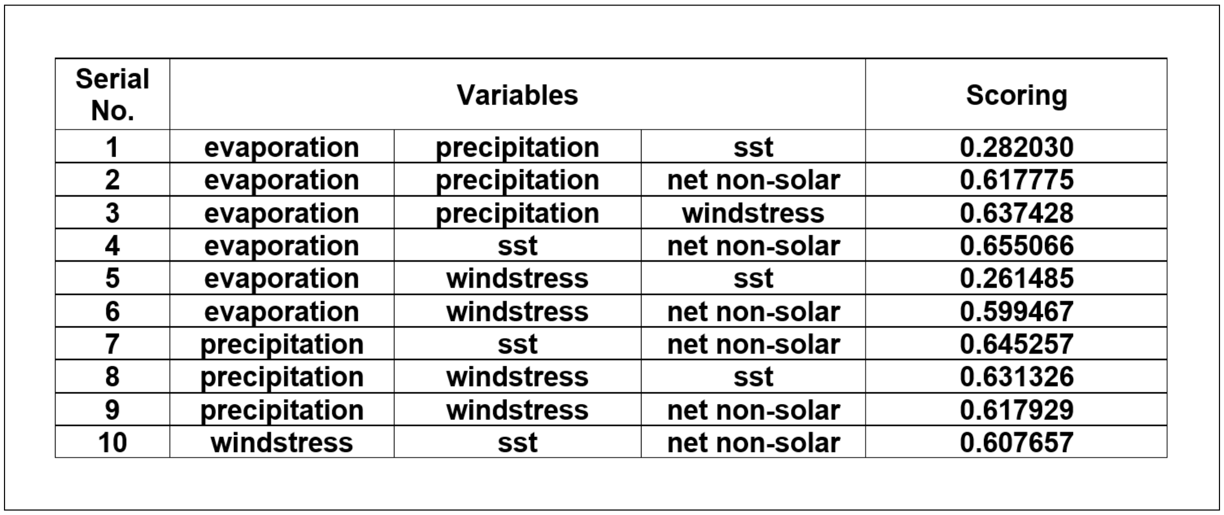

input = evaporation, wind stress, sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.630562
input = evaporation, precipitation, sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.665150
input = evaporation, precipitation, wind stress & net nonsolar
Pearson Correlation Coefficient = 0.648956
input = precipitation, wind stress, sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.643335
input = evaporation, precipitation, wind stress & sea surface temp.
Pearson Correlation Coefficient = 0.405277
Longitude
Latitude
Correlation

Figure 5.4: geographical plots showing correlation between predictions from the AI model and the ground truth for 4 input variables

The best performance we can get from this experiment was Pearson correlation coefficient of 0.665 which is lower than our actual model. Other hyperparameters and model training conditions were kept the same as the initial model. This shows that the model seems to learn better when 5 variables were used as the input of the model when compared to 4 variables being used.

Ablation of 2 variables



Serial No.	Variables			Scoring
1	evaporation	precipitation	sst	0.282030
2	evaporation	precipitation	net non-solar	0.617775
3	evaporation	precipitation	windstress	0.637428
4	evaporation	sst	net non-solar	0.655066
5	evaporation	windstress	sst	0.261485
6	evaporation	windstress	net non-solar	0.599467
7	precipitation	sst	net non-solar	0.645257
8	precipitation	windstress	sst	0.631326
9	precipitation	windstress	net non-solar	0.617929
10	windstress	sst	net non-solar	0.607657

Figure 5.5: Correlation values of the predictions from model with 3 input variables

In this test, 2 variables were removed and the model was trained with the remaining 3 variables as input. As given by $C^5_3 = 10$, we get 10 different combinations of model inputs. Table 5.5 shows the Pearson Correlation Coefficient value as a performance metric for the AI model predictions with 3 variables as input. The corresponding figure 5.6 shows the correlation values for each of the combinations of 3 variables in the Bay of Bengal map.

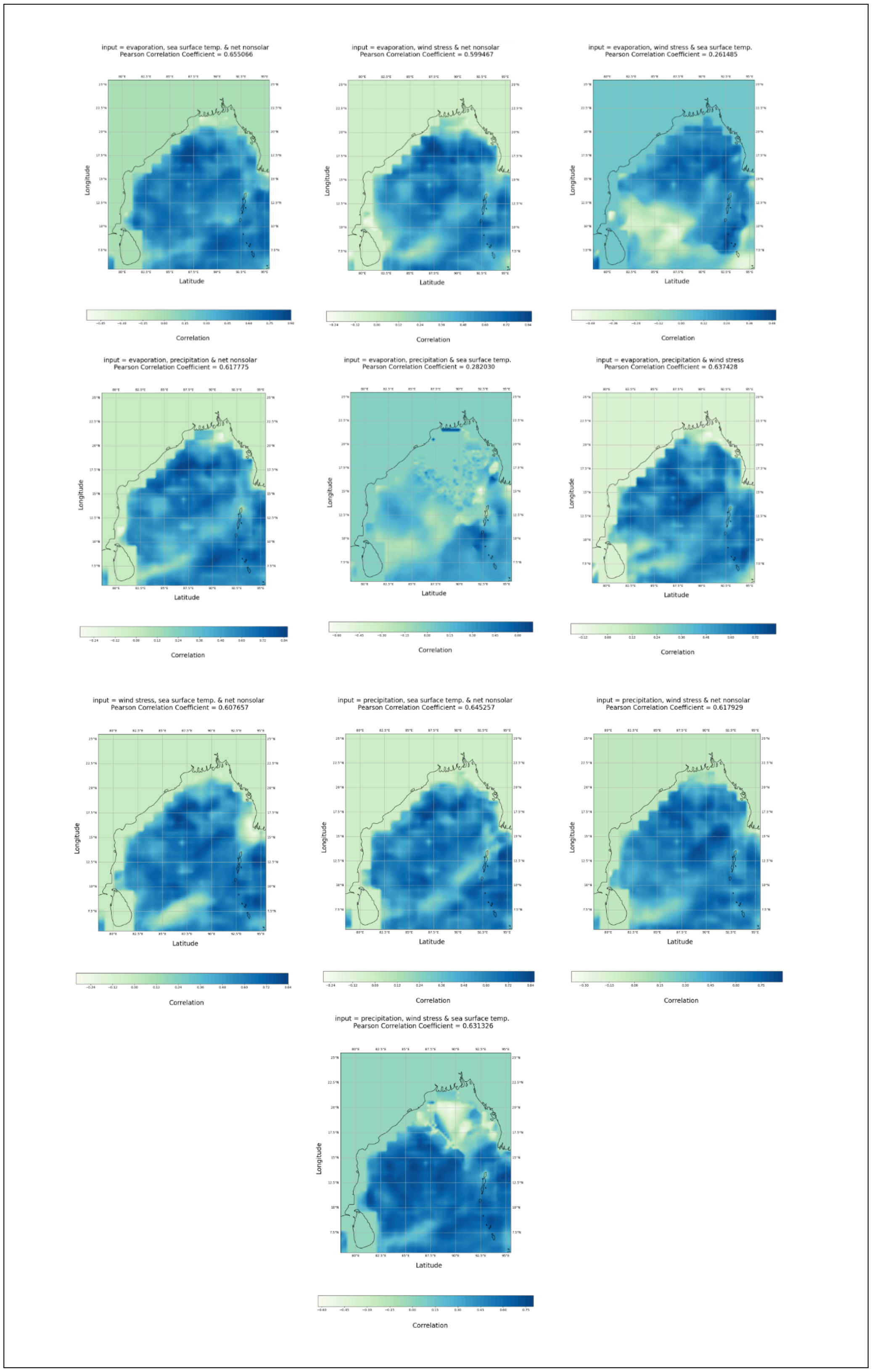
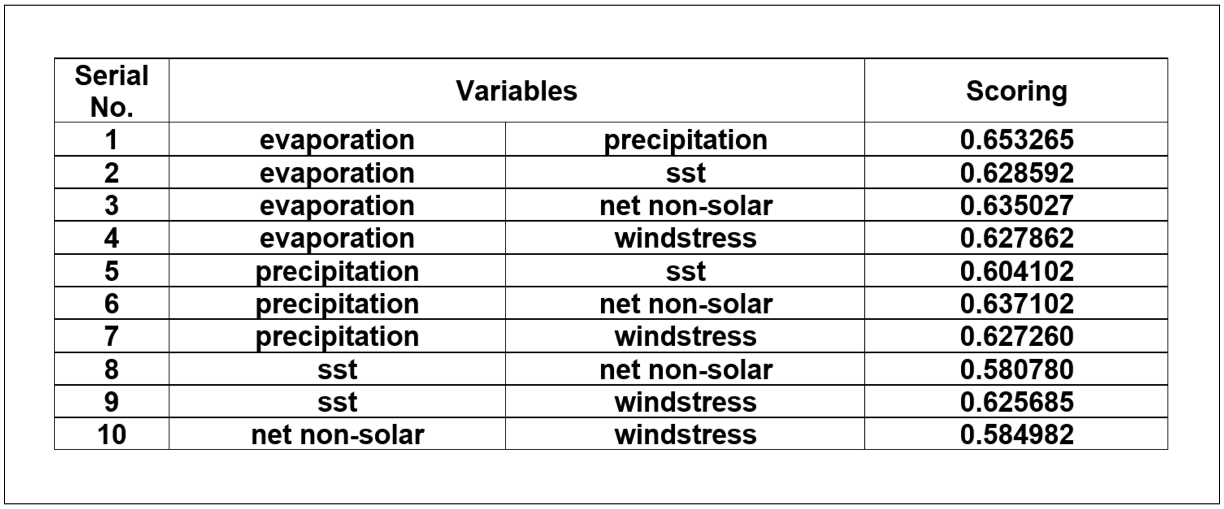

input = evaporation, sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.655066
input = evaporation, wind stress & net nonsolar
Pearson Correlation Coefficient = 0.599467
input = evaporation, wind stress & sea surface temp.
Pearson Correlation Coefficient = 0.261485
input = evaporation, precipitation & net nonsolar
Pearson Correlation Coefficient = 0.617775
input = evaporation, precipitation & sea surface temp.
Pearson Correlation Coefficient = 0.282030
input = evaporation, precipitation & wind stress
Pearson Correlation Coefficient = 0.637428
input = wind stress, sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.607657
input = precipitation, sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.645257
input = precipitation, wind stress & net nonsolar
Pearson Correlation Coefficient = 0.617929
input = precipitation, wind stress & sea surface temp.
Pearson Correlation Coefficient = 0.631326
Longitude
Latitude
Correlation

Figure 5.6: geographical plots showing correlation between predictions from the AI model and the ground truth for 3 input variables

As evident in the above figures, the model with input variables as evaporation, sea surface temperature and net non-solar heat flux performed the best among the models with 3 variables with a correlation value of 0.655. This suggests that this combination of forcings is more prevalent than the others. However, it can be noticed that the Pearson correlation coefficient value for this combination is still less than the original model with all 5 variables, making it irrelevant for our use in forecasting.

Ablation of 3 variables



Serial No.	Variables		Scoring
1	evaporation	precipitation	0.653265
2	evaporation	sst	0.628592
3	evaporation	net non-solar	0.635027
4	evaporation	windstress	0.627862
5	precipitation	sst	0.604102
6	precipitation	net non-solar	0.637102
7	precipitation	windstress	0.627260
8	sst	net non-solar	0.580780
9	sst	windstress	0.625685
10	net non-solar	windstress	0.584982

Figure 5.7: Correlation values of the predictions from model with 2 input variables

In this test, 3 variables were removed and the model was trained with the remaining 2 variables as input. As given by $C^5_2 = 10$, we get 10 different combinations of model inputs. Table 5.7 shows the Pearson Correlation Coefficient value as a performance metric for the AI model predictions with 2 variables as input. The corresponding 5.8 shows the correlation values for each of the combinations of 2 variables in the Bay of Bengal map.

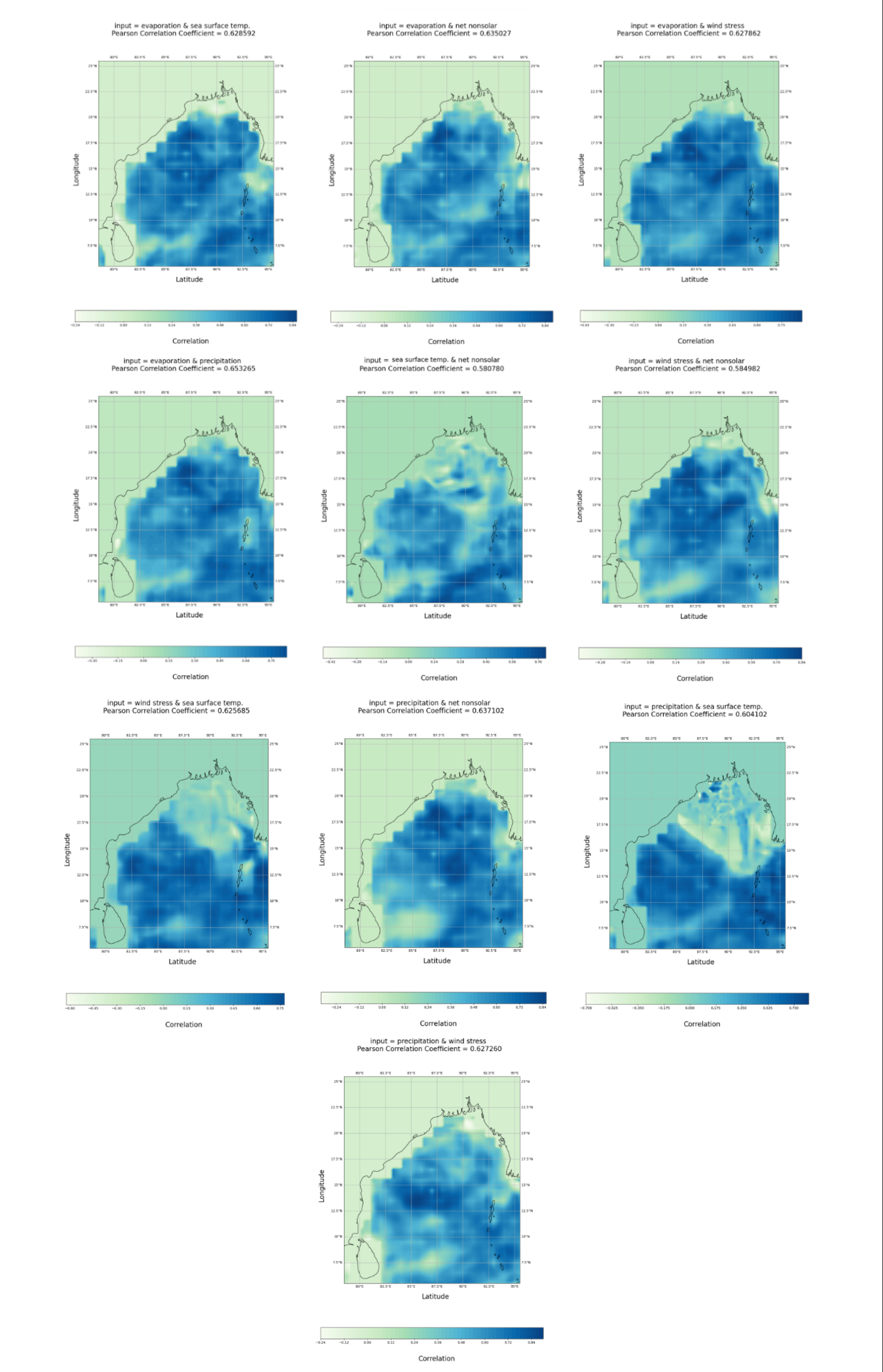
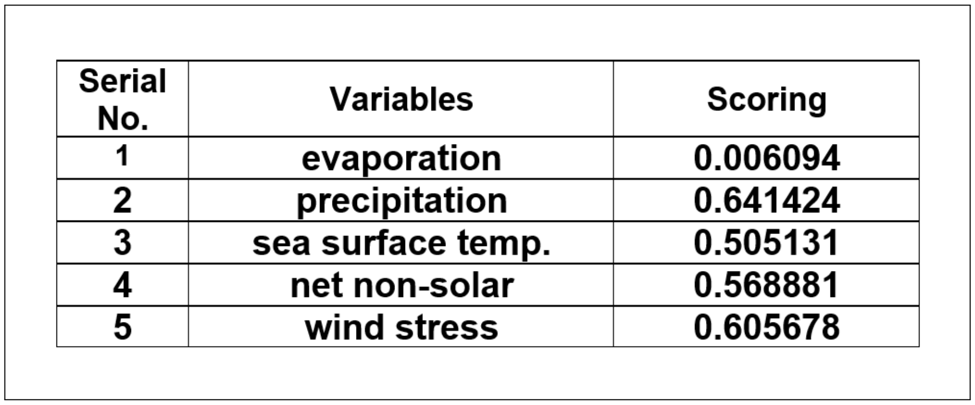

input = evaporation & sea surface temp.
Pearson Correlation Coefficient = 0.628592
input = evaporation & net nonsolar
Pearson Correlation Coefficient = 0.635027
input = evaporation & wind stress
Pearson Correlation Coefficient = 0.627862
input = evaporation & precipitation
Pearson Correlation Coefficient = 0.653265
input = sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.580780
input = wind stress & net nonsolar
Pearson Correlation Coefficient = 0.584982
input = wind stress & sea surface temp.
Pearson Correlation Coefficient = 0.625685
input = precipitation & net nonsolar
Pearson Correlation Coefficient = 0.637102
input = precipitation & sea surface temp.
Pearson Correlation Coefficient = 0.604102
input = precipitation & wind stress
Pearson Correlation Coefficient = 0.627260
Longitude
Latitude
Correlation

Figure 5.8: geographical plots showing correlation between predictions from the AI model and the ground truth for 2 input variables

As evident in the above-mentioned figures, the model with input variables as Evaporation and Precipitation performed the best among the models with 2 variables with a correlation value of 0.653. This suggests that this combination of forcings is more prevalent than the others. However, it can be noticed that the Pearson correlation coefficient value for this combination is still less than the original model with all 5 variables, making it irrelevant for our use in forecasting.

Ablation of 4 variables



Serial No.	Variables	Scoring
1	evaporation	0.006094
2	precipitation	0.641424
3	sea surface temp.	0.505131
4	net non-solar	0.568881
5	wind stress	0.605678

Figure 5.9: Correlation values of the predictions from the model with single input variables

In this test, 4 variables were removed and the model was trained with the remaining 1 variable as input. As given by $C_4^5 = 5$, we get 5 different combinations of model inputs. Table 5.9 shows the Pearson Correlation Coefficient value as a performance metric for the AI model predictions with a single variable as input. The corresponding 5.4 shows the correlation values for a single input variable in the Bay of Bengal map.

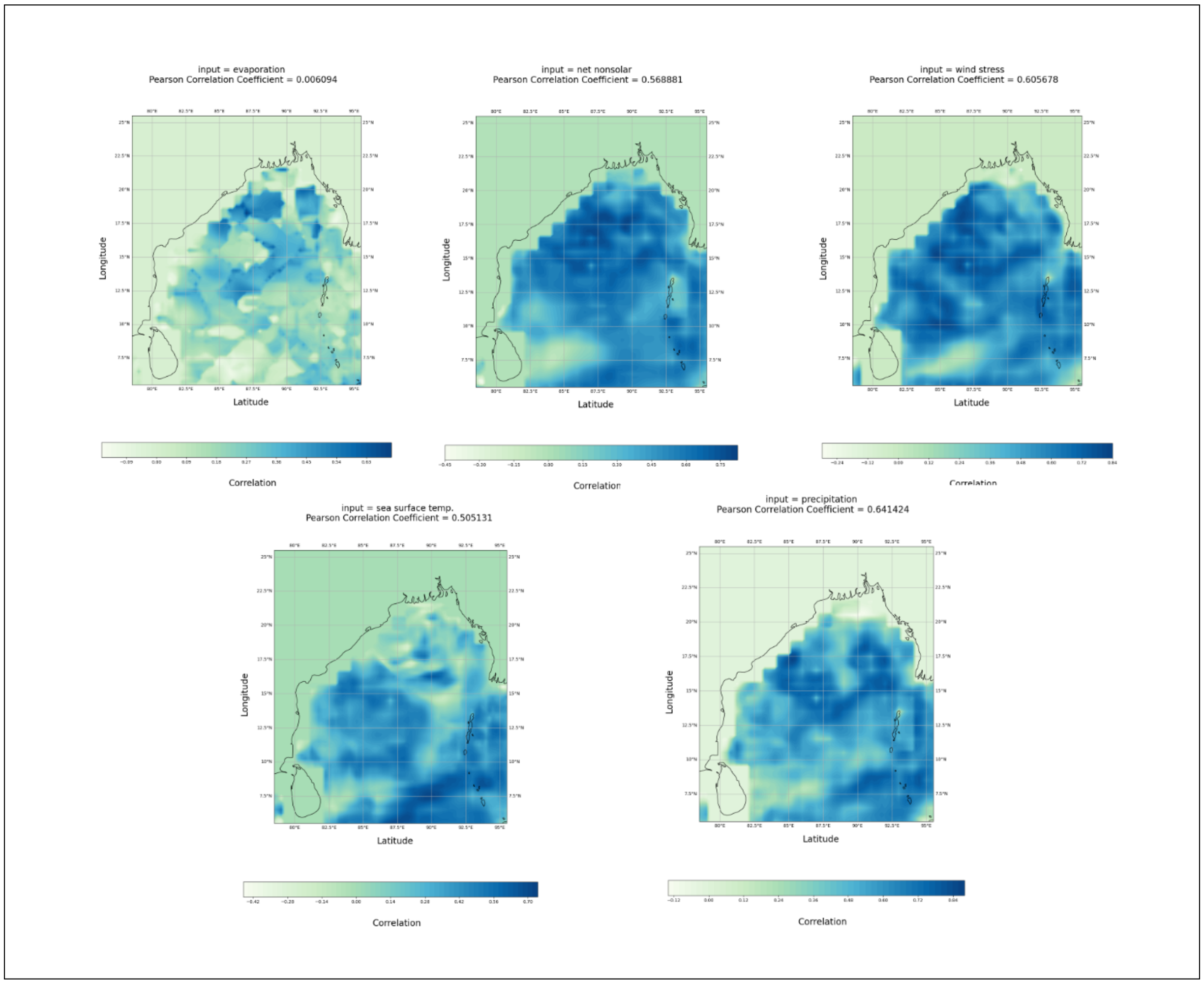

input = evaporation
Pearson Correlation Coefficient = 0.006094
input = net nonsolar
Pearson Correlation Coefficient = 0.568881
input = wind stress
Pearson Correlation Coefficient = 0.605678
input = sea surface temp.
Pearson Correlation Coefficient = 0.505131
input = precipitation
Pearson Correlation Coefficient = 0.641424
Longitude
Latitude
Correlation

Figure 5.10: geographical plots showing correlation between predictions from the AI model and the ground truth for single input variables

As evident in the figures 5.9 and 5.10, the model with input variable Precipitation performed the best among the models with single input with a correlation value of 0.641. This suggests that the forcing Precipitation when acted separately is more prevalent in predicting MLD than the others. However, it can be noticed that the Pearson correlation coefficient value for this is still less than the original model with all 5 variables, making it irrelevant for our use in forecasting.

5.3 Layer Ablation

Layer ablation refers to the removal or addition of neural architecture with the activation function associated with it. The original model contains four

Up-sampling layers and in two separate perturbations we compare the result.

5.3.1 Trial 1: Removal of 1 layer

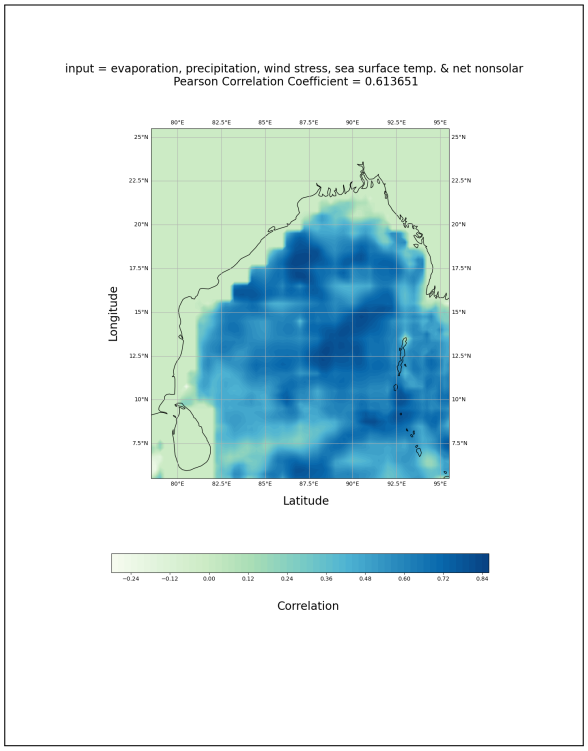

input = evaporation, precipitation, wind stress, sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.613651
Longitude
Latitude
Correlation

Figure 5.11: geographical plot showing correlation between predictions from the model with an ablated layer and the ground truth

One of the layers of the neural network Architecture is removed and the AI model is trained with 5 variables as input for 1000 epochs. Other hyperparameters remain same with the maximum number of Output filters (filters = 8) in the convolution layer being 8. The performance of the model is compared using the Pearson's correlation coefficient which gave the correlation value of 0.614. This suggests the AI model does not gain in learning when the layer was removed. Figure 5.12 shows the Pearson correlation coefficient matrix on geographical plot.

5.3.2 Trial 2: Addition of 1 layer

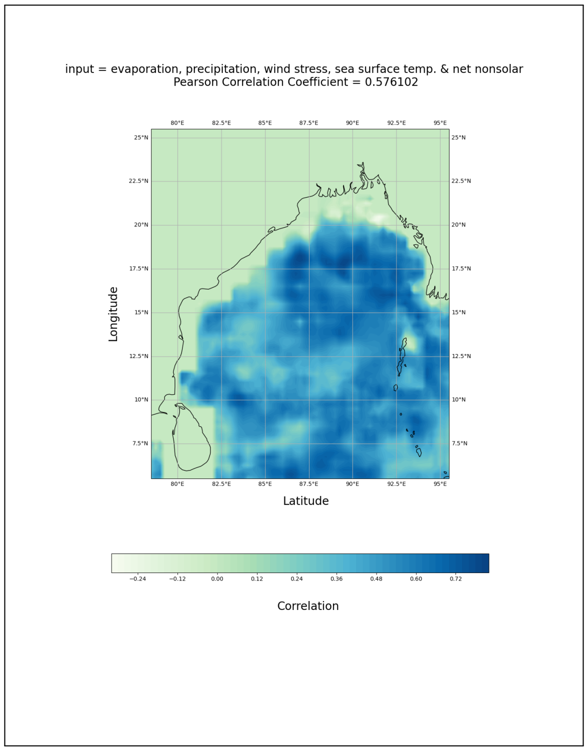

input = evaporation, precipitation, wind stress, sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.576102
Longitude
Latitude
Correlation

Figure 5.12: geographical plot showing correlation between predictions from the model with an added layer and the ground truth

In this perturbation, a ConvLSTM layer is added and similarly, all other hyperparameters were kept the same. The maximum number of the output filters in the interior layer defined by ‘filters=32’ was 32. A downsampling layer was also added corresponding to the upsampling layer. The performance of the model measured y Pearson Correlation Coefficient was 0.576 and this again suggests that our initial model performed better and should be used for further studies.

5.4 Trial with Skip Connections

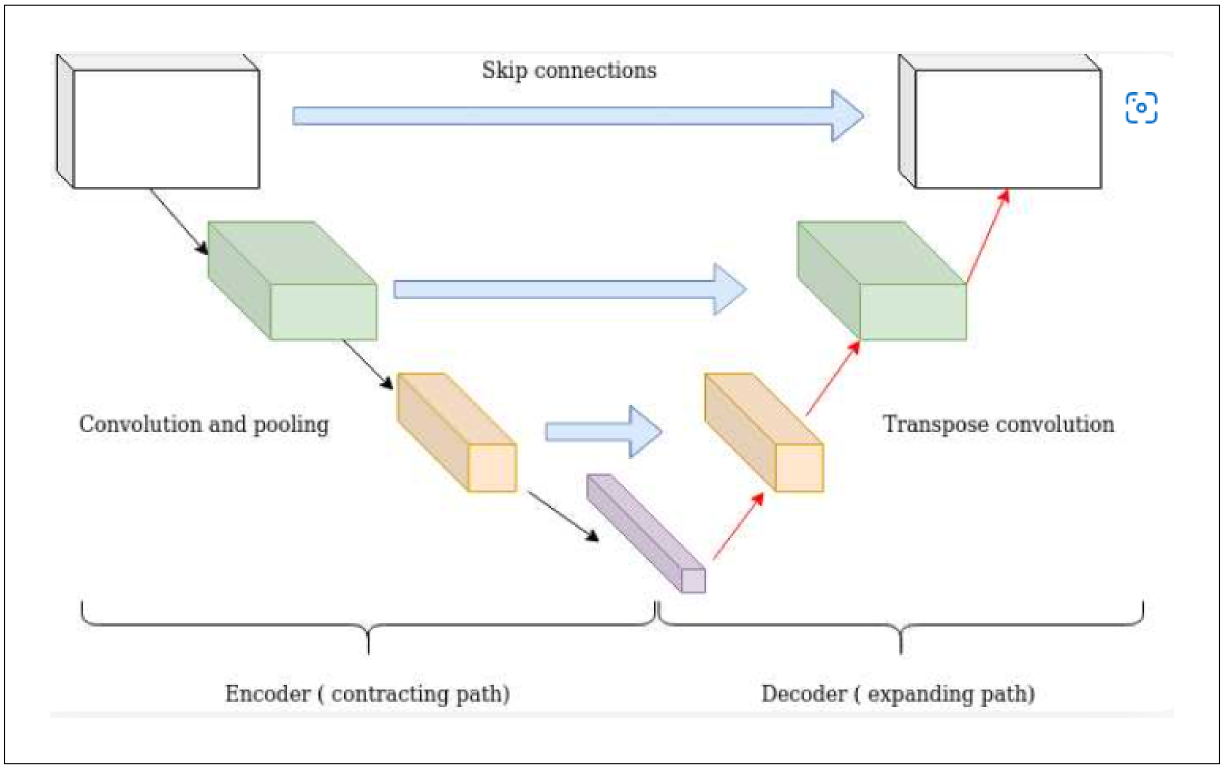


Figure 5.13: mechanism of skip connection in an AI model (Image taken from Michal et al.[3])

In current times, skip connection is a commonly used module in CNN architectures. The skip connection provides a substitute path for the layer gradients. It is often seen that adding additional paths in the architecture benefits the convergence of the model. Trivially skip connections skip some layers in the neural network architecture and feed the input of a layer with the output of any distant previous layers instead of just the adjacent previous layer. The central idea in this process is to backpropagate through the identity function, which is done by just adding a simple vector. This process can be seen in Residual networks or ResNet, which stack up skip connections together, and the gradient gets multiplied by 1 for each such skip connection maintaining the values across layers and countering the vanishing gradient problem.

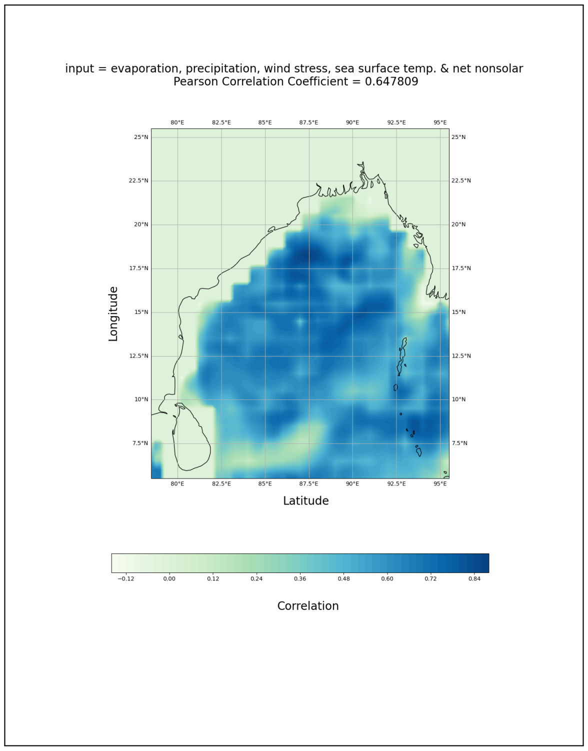

input = evaporation, precipitation, wind stress, sea surface temp. & net nonsolar
Pearson Correlation Coefficient = 0.647809
Longitude
Latitude
Correlation

Figure 5.14: geographical plot showing correlation between predictions from the model with skip connect and the ground truth

The model with the 2 skip connections as shown in the figure 5.14 returned a Pearson Correlation Coefficient of 0.648 and hence in this study, it could not provide any benefit to the model. The Geographical plot shows its performance at the regional level.

The ablation study proved that interestingly, our initial model turned out to be the best one and will hence be used for further validation studies in this chapter.

5.5 Validation with observational Data

This is the first study which has tried to forecast MLD in the Bay of Bengal region and has used so many forcings for predictions. The idea of using forcing is to confirm that all these forcing collectively impacts the formation of MLD and even removing a single forcing impact the predictions.

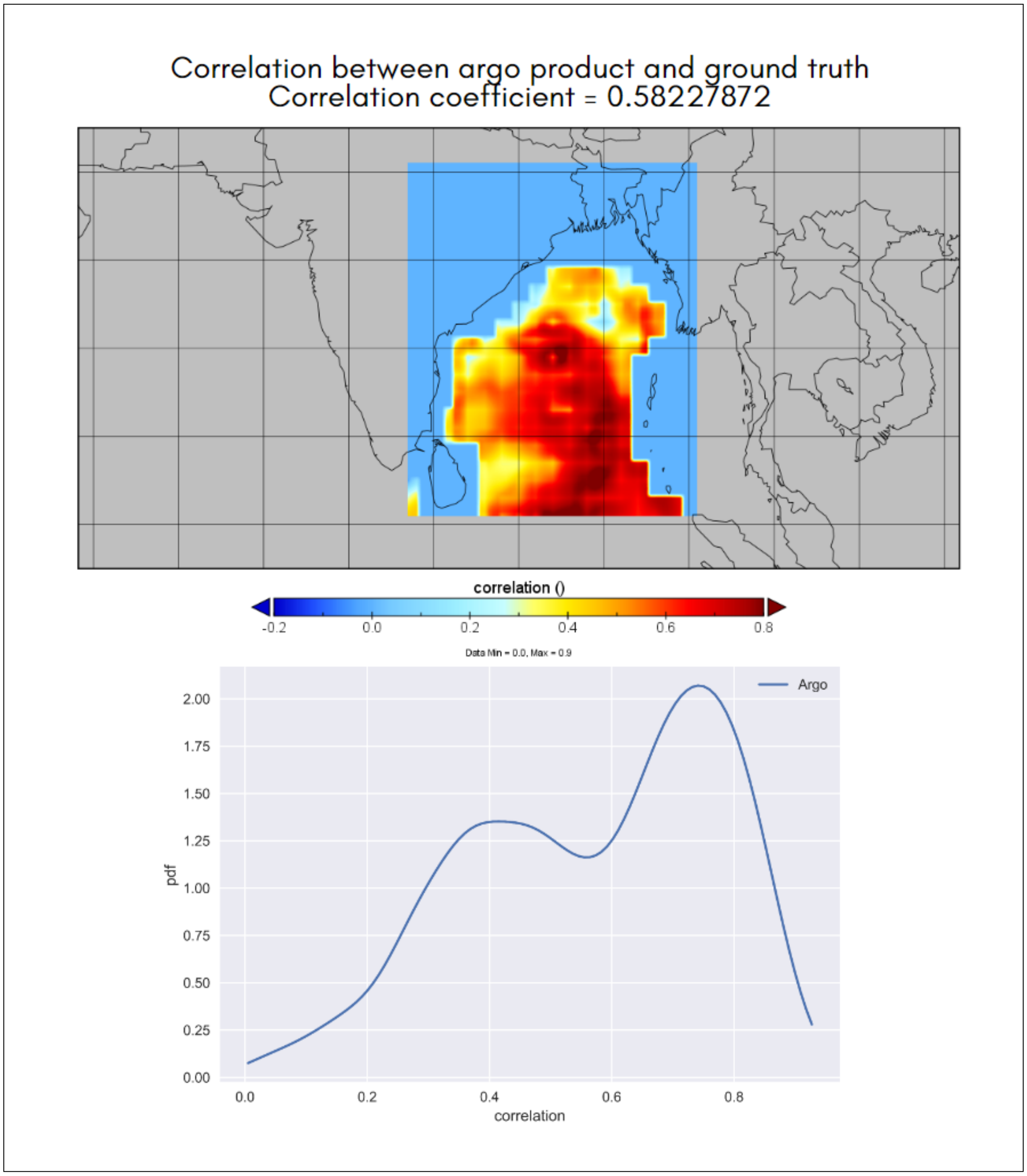


Figure 5.15: geographical plot showing correlation between predictions from the argo product data and the ground truth (top), and density vs correlation plot for the same (bottom)

The predictions from our AI model were compared to observational data obtained from the observation station, referred as gridded Argo data and is downloaded from the ERDDAP sever. The Pearson correlation coefficient of the Argodata vs Oras5 Reanalysis data was calculated and plotted in the geographical map in the figure 5.15. The correlation value obtained was 0.582 which is quite less when compared to the correlation between AI model predictions and reanalysis data. Even though it is not very healthy to compare a prediction to direct observation but still for the crux of the comparison, this suggests that our model is performing better than the direct observation data. This trend could be because Reanalysis data is made by taking into account the observational data and input from dynamical models, and our model learns directly from the reanalysis data.

5.6 Validation with Hindcast model forecast

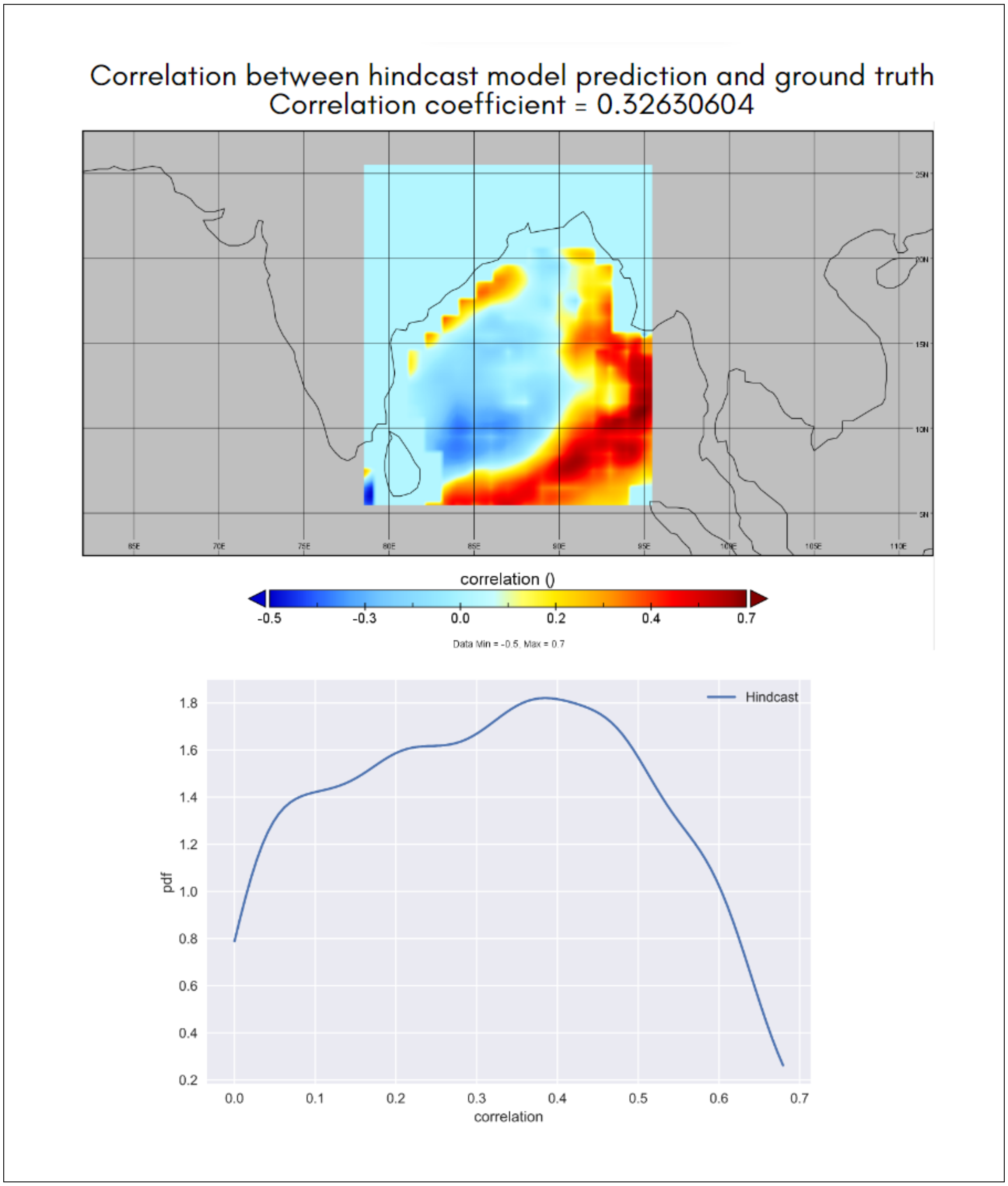


Figure 5.16: geographical plot showing correlation between predictions from the hindcast model prediction and the ground truth (top), and density vs correlation plot for the same (bottom)

Hindcast is a numerical weather prediction model (physics-based model running on partial differential equations) which is started from the exact observational date. The outputs from these models are presently used as forecasts, which go to the public. Hindcast has been running for the past many years. Also, the data is daily CMIP6 (the Sixth Phase of the Coupled Model Intercomparison Project) data averages to get the monthly data. Currently, all weather predictions are being done using some sort of dynamical model, especially in India. For basic phenomena prediction like rainfall, dynamical model seems to be performing well with some scope for improvement but when it came to prediction using a complex system like in our case, it seems to be not doing well. MLD is not a very popular phenomenon and research needs to be done in dynamical models and AI-based model to get a better result. In this comparison, our AI model seems to be doing much better than the Hindcast model as the latter gave a correlation score of 0.326 vs the 0.673 of the AI model.

5.7 Final Result

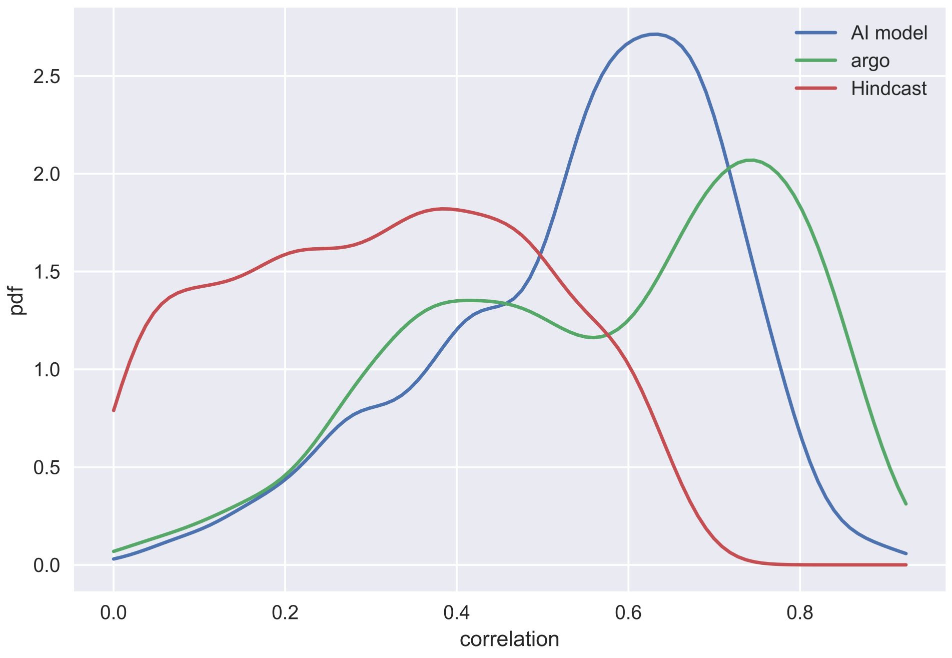


Figure 5.17: density vs correlation plot for the correlation values of AI model, Hindcast forecasts and the argo product data

Figure 5.17 shows that the AI model with all 5 input variables performed better than the most commonly used physics-based dynamical model. The AI model was put into an ablation study and after an extensive comparison with the perturbations, the same AI model came out to be the best performer and is to be used in future problems.

Chapter 6

Conclusion

This study used the ConvLSTM to forecasting MLD over the Bay of Bengal region. The crux of forecasting was to find and validate the influence of various forcings on MLD and multiple models were trained to get that return. The qualitative studies suggest that MLD formation is dependent on Sea Surface Temperature, Wind mixing, Heat flux, evaporation and precipitation but there is no study which tries to validate this using data. In our novel approach, we showed that all these variables combined gave us the best result when forecasting and when forcing were ablated, the performance differed with none being better than the initial. Apart from realizing how and which variable impacts MLD the most, this study also gives us a novel approach to look into forecasting Cyclones, Heat waves and other ocean surface-induced phenomena.

Currently, prominent weather forecasting agencies like Indian Meteorological Department (IMD) essentially use dynamical models like the Hindcast. The dynamical models often have discrepancies in predictions, for example, it does not rain even after IMD predicts, on a certain day. It makes the basis for us to focus on the AI model for improving real-time learning and predictions, and it needs to be incorporated into the government system. The idea is to use AI to assist the current model and eventually transform the current way of weather predictions. This study showed that predictions from the Hindcast model gave a correlation coefficient of 0.32 whereas our AI model gave 0.673 and hence suggesting the importance of extensive research in this area.

Our future work includes studying MLD in the whole of the Indian Ocean and parts of the Pacific Ocean which are known to impact weather in India.

Further, the forecast could be mapped with other ocean variables such as cyclone incidents and the model could be used in predicting its prediction. This study deals with monthly data and so would be beneficial in predicting such events with sufficient time for evacuation and safety measures to prevent damages from such cyclonic events.

An important area to look into will be training the model using daily or weekly data to get more accurate predictions when we are dealing with a short-time phenomenon such as lightning which develops fast and probably could not be caught with monthly data. Another crucial aspect is to do a qualitative analysis of the correlation plot. It is evident that in some cases even though the model did not perform well overall looking into Pearson Correlation Coefficient, it did well with a correlation value of greater than 0.8 in particular regions in the Bay of Bengal. An attempt to use this regional result could use a geographical ensemble of multiple regional models to forecast MLD across the whole BOB region.

Chapter 7

Code Availability and Data Source

The source codes along with all the helper modules used in this study can be found on `https://github.com/githubwasmyidea/msthesis` as jupyter notebook source format.

The dataset used in the study were downloaded from multiple sources as follows:

- Evaporation, Sea surface temperature, Precipitation, Mean surface latent heat flux, Mean surface net long-wave radiation flux and Mean surface sensible heat flux were downloaded from the ERA5 repository- `https://shorturl.at/pAW16`
- Wind stress magnitude was downloaded from the Tropflux repository- `https://incois.gov.in/tropflux/DataHome.jsp`
- Mixed Layer Depth data was downloaded from ORAS5(ERDDAP) repository- `http://apdrc.soest.hawaii.edu/erddap/search/index.html?page=1&itemsPerPage=1000&searchFor=oras5`
- The argo gridded data was also downloaded from ERDDAP repository- `http://apdrc.soest.hawaii.edu/erddap/griddap/hawaii_soest_4daf_fed7_948a.html`

Bibliography

[1] Chao Jia. Changes of dance sports in air equipment dance performance using convolutional neural network. *Mobile Information Systems*, 2022:1–11, 06 2022.

[2] Zhuoning Yuan, Xun Zhou, and Tianbao Yang. Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. pages 984–992, 07 2018.

[3] Michal Drozdzal, Eugene Vorontsov, Gabriel Chartrand, Samuel Kadoury, and Chris Pal. The importance of skip connections in biomedical image segmentation. *CoRR*, abs/1608.04117, 2016.

[4] Leonard Smith, Jan Barkmeijer, and Tim Palmer. Model error in weather forecasting. *Nonlinear Processes in Geophysics*, 8, 09 2001.

[5] Ingrid Daubechies, Ronald A. DeVore, Simon Foucart, Boris Hanin, and Guergana Petrova. Nonlinear approximation and (deep) relu networks. *CoRR*, abs/1905.02199, 2019.

[6] David Rolnick, Priya L. Donti, Lynn H. Kaack, Kelly Kochanski, Alexandre Lacoste, Kris Sankaran, Andrew Slavin Ross, Nikola Milojevic-Dupont, Natasha Jaques, Anna Waldman-Brown, Alexandra Sasha Luccioni, Tegan Maharaj, Evan D. Sherwin, S. Karthik Mukkavilli, Konrad P. Kording, Carla P. Gomes, Andrew Y. Ng, Demis Hassabis, John C. Platt, Felix Creutzig, Jennifer Chayes, and Yoshua Bengio. Tackling climate change with machine learning. *ACM Comput. Surv.*, 55(2), feb 2022.

[7] Frances V. Davenport and Noah S. Diffenbaugh. Using machine learning to analyze physical causes of climate change: A case study

of u.s. midwest extreme precipitation. *Geophysical Research Letters*, 48(15):e2021GL093787, 2021. e2021GL093787 2021GL093787.

[8] Ariane Middel, Negin Nazarian, Matthias Demuzere, and Benjamin Bechtel. Urban climate informatics: An emerging research field. *Frontiers in Environmental Science*, 10, 05 2022.

[9] Ali Nassif, Manar Abu Talib, Qassim Nasir, and Fatima Dakalbab. Machine learning for anomaly detection: A systematic review. *IEEE Access*, PP:1–1, 05 2021.

[10] Michael Alexander, James Scott, and Clara Deser. Processes that influence sea surface temperature and ocean mixed layer depth variability in a coupled model. *Journal of Geophysical Research: Oceans*, 105, 05 2000.

[11] Qing Li and Baylor Fox-Kemper. Anisotropy of langmuir turbulence and the langmuir-enhanced mixed layer entrainment. *Phys. Rev. Fluids*, 5:013803, Jan 2020.

[12] Yusuke Ushijima and Yutaka Yoshikawa. Mixed layer deepening due to wind-induced shear-driven turbulence and scaling of the deepening rate in the stratified ocean. *Ocean Dynamics*, 70, 01 2020.

[13] Tvs Udaya Bhaskar, Debadatta Swain, and Muthalagu Ravichandran. Inferring mixed-layer depth variability from argo observations in the western indian ocean. *Journal of Marine Research*, 64:393–406, 05 2006.

[14] Levitus S. Monterey G. Seasonal variability of mixed layer depth for the world ocean. *U.S. Gov. Printing Office, Wash. DC.*, 1997.

[15] Conghui Fan, Juanjuan Wang, and Jinbao Song. Factors influencing the climatological mixed layer depth in the south china sea: Numerical simulations. *Chinese Journal of Oceanology and Limnology*, 28:1112–1118, 09 2010.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[17] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai Kin Wong, and Wang-chun WOO. Convolutional lstm network: A machine learning approach for precipitation nowcasting. 06 2015.

[18] Akhilesh Waoo and Brijesh Soni. Performance analysis of sigmoid and relu activation functions in deep neural network. pages 39–52, 07 2021.

[19] Nils Thuerey, Philipp Holl, Maximilian Müller, Patrick Schnell, Felix Trost, and Kiwon Um. Physics-based deep learning. *CoRR*, abs/2109.05237, 2021.

[20] Hao Zuo, Magdalena Balmaseda, Steffen Tietsche, Kristian Mogensen, and Michael Mayer. The ecmwf operational ensemble reanalysis-analysis system for ocean and sea-ice: a description of the system and assessment. *Ocean Science Discussions*, pages 1–44, 01 2019.

[21] Praveen Kumar, Jerome Vialard, Matthieu Lengaigne, V.s.N. Murty, and M. McPhaden. Tropflux: Air-sea fluxes for the global tropical oceans-description and evaluation. *Climate Dynamics*, 38:1521–1543, 04 2012.

[22] Panoply data viewer [computer software].(version 4.12.11, released 2021-08-28) retrieved from http://giss.nasa.gov.

[23] Xiaofan Li, Chung-Hsiung Sui, and William Lau. Effects of precipitation on ocean mixed-layer temperature and salinity as simulated in a 2-d coupled ocean-cloud resolving atmosphere model. *Journal of the Meteorological Society of Japan*, 78, 10 2000.

[24] Alex Graves. Generating sequences with recurrent neural networks. 08 2013.

[25] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[26] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

[27] MarcAurelio Ranzato, Arthur Szlam, Joan Bruna, Michael Mathieu, Ronan Collobert, and Sumit Chopra. Video (language) modeling: a baseline for generative models of natural videos. 12 2014.

[28] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. *CoRR*, abs/1502.04681, 2015.

[29] Richard Meyes, Melanie Lu, Constantin Waubert de Puiseau, and Tobias Meisen. Ablation studies in artificial neural networks. 01 2019.

[30] Alessio Molinari. Designing a performant ablation study framework for pytorch. (2020:790):58, 2020.