IIT Indore online internship report

On

Cyclone track using deep learning

From June 1st 2021 to July 31st 2021

Submitted by Abhinav Dhar

Bachelors of Technology Electronics & Communication Engineering

Under the mentorship of

Dr. Saurabh Das

Assistant Professor

Department of Astronomy, Astrophysics and Space Engineering

Indian Institute of Technology Indore

TABLE OF CONTENTS

Introduction

A tropical cyclone is a low-pressure system with a warm core that forms over tropical and subtropical waters. The tropical and subtropical oceans receive large amounts of energy from the sun. This energy is released into the atmosphere in the form of water vapor. The release of heat causes an upward motion of air, creating a low-pressure zone that is set into spin by the rotation of the earth. When the energy contained in this spiralling airflow is high enough, a cyclone is formed.

Hurricanes are circular cyclonic storms that draw their energy from the warm tropical sea. They are generally smaller than middle-latitude cyclones (500–1000 km in horizontal extent, compared with >1500 km), which depend on the tropics-to-pole horizontal temperature gradient for their energy. The hurricane vortex generally fills the depth of the troposphere. Its effects may reach from the upper 200 m of the sea into the lower stratosphere. Hurricanes are warm core in the sense that the air near the centre is warmer than the surrounding atmosphere. Because warm air is less dense than cold, this property causes their low hydrostatic central pressures, which can be as much as 10% below that in the normal tropical atmosphere, and their sustained winds which can reach 165 nautical miles.

In the north western Pacific, meteorologists call cyclones of hurricane intensity typhoons. In the Southern Hemisphere and throughout the Indian Ocean, they are simply called cyclones. Tropical Cyclone (TC) is the generic term for warm-core systems worldwide. In some contexts, TC refers to systems of tropical storm or hurricane intensity; in others, TC includes depressions as well.

The eye of cyclone

The eye, which occupies the geometric centre of the TC vortex, is characteristically tens of kilometers in radius. It generally appears as the TC crosses the threshold of hurricane intensity. The eye is often clear from a kilometer or two above the surface to the tropopause, particularly in the most intense hurricanes. Within the eye, there is invariably a stagnation point where the winds are calm, but the clear eye is not filled with calm winds, literary metaphors notwithstanding. Around the eye is a ring of convective clouds, called the wall cloud, where water vapor drawn from the sea under the strong-wind part of the vortex condenses, transforming heat of vaporization into sensible heat to fuel the storm. The inner edge of the eyewall is also the site of the hurricane’s strongest winds. In this context, it is important to recall that hurricanes are both more intense and more compact than their middle-latitude relatives. Damaging winds are often confined within 100 km of their centres.

Forecasting Cyclones

Forecasting of cyclones involves the prediction of several interrelated features, such as the cyclone’s track, intensity, induced storm surges and accompanying rainfall, and the coastal areas threatened. Among all these interrelated features it is most important to know in which direction a cyclone will move so that the inhabitants of potentially affected areas can be warned well ahead of time, in this way minimizing damage to life and property. For this reason, forecasting of cyclone track has been considered as one of the most important forecasting functions by many meteorological offices around the world. As cyclones formed in different basins exhibit large variation in behaviour, meteorological offices tend to use a combination of techniques to forecast cyclones in order to achieve highest possible accuracy and reliability. All forecasting techniques have in common that they take into account the recent-past behaviour of the current cyclone and/or the behaviour of previously encountered, similar cyclones. Previous cyclones can be similar to the present one in terms of their creation—created in the same ocean basin and/or during the same season or even the same month during a previous year. They can also be similar in terms of their behaviour pattern—they may have had a similar movement track as the present cyclone has exhibited up till present time. The basic assumption is that whatever forces are affecting the present cyclone also have affected these previous cyclones, also called predictors, and that studying these predictors will therefore help to forecast the track of the current cyclone.

Problem Statement

The aim was to predict the latitude and longitudinal coordinates of a cyclone using the data obtained from the International Best Track Archive for Climate Stewardship (IBTrACS). The dataset used in the project contained data of cyclones occurring in the North Indian ocean from 2002 to 2019.

Information on the data

The intent of the IBTrACS project is to overcome data availability issues. This was achieved by working directly with all the Regional Specialized Meteorological Centres and other international centres and individuals to create a global best track dataset, merging storm information from multiple centres into one product and archiving the data for public use.

Column Information

From the original Dataset the following columns were extracted:

- SUBBASIN- Showed in which basin the cyclone originated. In the North Indian Ocean there are two subbasins, i.e. Arabian Sea (AS) and Bay of Bengal (BB)
- ISO_TIME- Time provided in Universal Time Coordinates (UTC). Format is YYYY-MM-DD HH:mm:s
- LAT- Latitude
- LON- Longitude
- WMO_WIND- Maximum sustained wind speed from the WMO agency for the current location.
- WMO_PRESSURE- Maximum sustained pressure from the WMO agency for the current location.
- DIST2LAND- Distance to land from the current position.
- LANDFALL- Nearest location to land within next 6 hours.
- NEWDELHI_GRADE- Types of disturbances:

a. Low pressure area W<17 knots
b. D - Depression 17<=W<28
c. DD - Deep Depression 28<=W<34
d. CS - Cyclonic Storm 34<=W<48
e. SCS - Severe Cyclonic Storm 48<=W<64
f. VSCS - Very Severe Cyclonic Storm 64<=W<120
g. SCS - Super Cyclonic Storm W>=120 km

- STORM_SPEED- Translation speed of the system as calculated from the positions in LAT and LON
- STORM_DIR- Translation direction of the system as calculated from the positions in LAT and LON. Direction is moving toward the vector pointing in degrees east of north.

Technologies Used

Python

Python is an interpreted, high-level, general-purpose programming language. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Google Collaboratory

Colaboratory, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs.

Python Libraries Used

Pandas

Pandas is a software library in Python used for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series data.

Numpy

NumPy is a library used to add and support large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

Scikit learn

It consists of various classification, regression and clustering algorithms including support vector machines, random forests, k-means and is designed to be used side by side with the Python numerical and scientific libraries like NumPy.

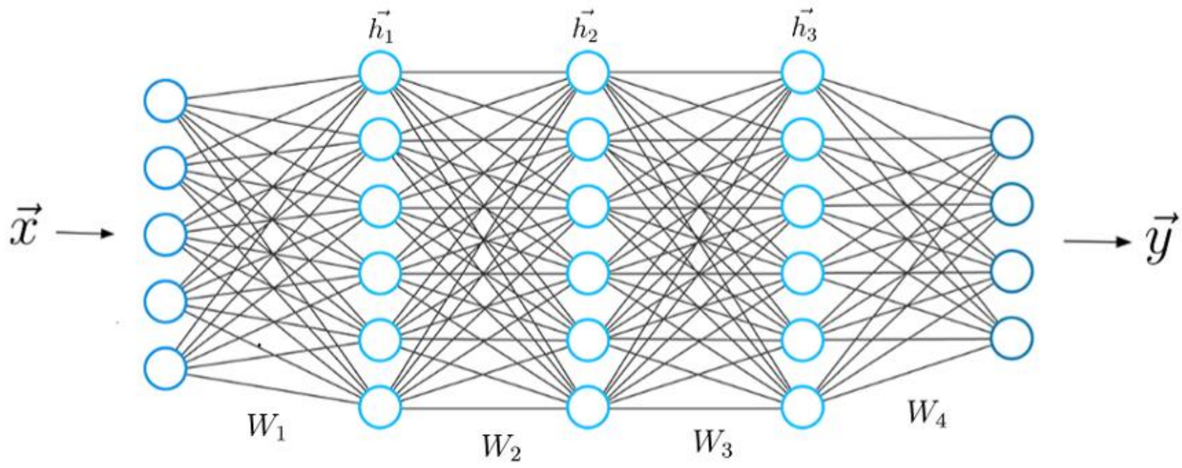
Keras

Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. It was developed with a focus on enabling fast experimentation. It is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

Concepts Applied

Deep Learning

Deep Learning is just a type of Machine Learning, inspired by the structure of a human brain. Deep learning algorithms attempt to draw similar conclusions as humans would by continually analyzing data with a given logical structure. To achieve this, deep learning uses a multi-layered structure of algorithms called neural networks.



RNN

Recurrent Neural Network is a generalization of feedforward neural network that has an internal memory. RNN is recurrent in nature as it performs the same function for every input of data while the output of the current input depends on the past one computation. After producing the output, it is copied and sent back into the recurrent network. For making a decision, it considers the current input and the output that it has learned from the previous input.

LSTM

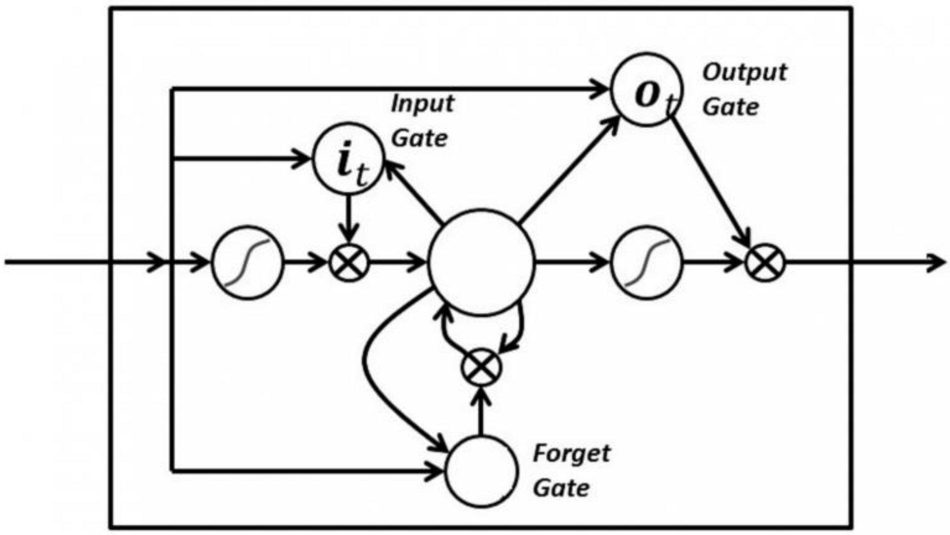
Long short-term memory networks are an extension for recurrent neural networks, which basically extends the memory. Therefore it is well suited to learn from important experiences that have very long time lags in between.

The units of an LSTM are used as building units for the layers of a RNN, often called an LSTM network.

LSTMs enable RNNs to remember inputs over a long period of time. This is because LSTMs contain information in a memory, much like the memory of a computer. The LSTM can read, write and delete information from its memory.

This memory can be seen as a gated cell, with gated meaning the cell decides whether or not to store or delete information (i.e., if it opens the gates or not), based on the importance it assigns to the information. The assigning of importance happens through weights, which are also learned by the algorithm. This simply means that it learns over time what information is important and what is not.

In an LSTM you have three gates: input, forget and output gate. These gates determine whether or not to let new input in (input gate), delete the information because it isn't important (forget gate), or let it impact the output at the current timestep (output gate). Below is an illustration of a RNN with its three gates:



ReLU LAYER

The ReLU (Rectified Linear Activation Function) is basically a linear function that activates the convoluted input, element wise based on the threshold (0). Neural networks with this activation function usually show better results and are easy to train, and are thus very popular.

Time Series Data

Time series data, also referred to as time-stamped data, is a sequence of data points indexed in time order. Time-stamped is data collected at different points in time.

These data points typically consist of successive measurements made from the same source over a time interval and are used to track change over time.

Our Data consists of storm data taken every 3 hours.

Pre-processing the data

The categorical data ("SUBBASIN") was transformed into readable data by using dummy variables.

```
[ ]  df=pd.get_dummies(df,prefix=['Basin'],columns=['SUBBASIN'])
     print(df)
```

```
                ISO_TIME        LAT      LON  ...  STORM_DIR  Basin_AS  Basin_BB
0    2002-05-06 03:00:00   9.30004  66.8800  ...        331         1         0
1    2002-05-06 06:00:00   9.96667  66.5000  ...        333         1         0
2    2002-05-06 09:00:00  10.61360  66.2109  ...        337         1         0
3    2002-05-06 12:00:00  11.15000  66.0000  ...        335         1         0
4    2002-05-06 15:00:00  11.44240  65.8147  ...        327         1         0
...                  ...       ...      ...  ...        ...       ...       ...
2835 2019-12-06 00:00:00   8.30000  53.4000  ...        239         1         0
2836 2019-12-06 03:00:00   8.14246  53.1113  ...        241         1         0
2837 2019-12-06 06:00:00   7.95000  52.7500  ...        243         1         0
2838 2019-12-06 09:00:00   7.85997  52.5475  ...        260         1         0
2839 2019-12-06 12:00:00   7.85000  52.1500  ...        266         1         0
```

The ("NEWDELHI_GRADE") column was mapped into a readable data by assigning each storm intensity grade as a numerical value.

```
temp_dict={'D': 1,
           'DD': 2,
           'CS': 3,
           'SCS': 4,
           'VSCS': 5,
           'SUCS': 6}
df['GRADE']=df.NEWDELHI_GRADE.map(temp_dict)
print(df)
```

```
                ISO_TIME       LAT      LON  ...  Basin_AS  Basin_BB  GRADE
0    2002-05-06 03:00:00   9.30004  66.8800  ...         1         0      1
1    2002-05-06 06:00:00   9.96667  66.5000  ...         1         0      1
2    2002-05-06 09:00:00  10.61360  66.2109  ...         1         0      1
3    2002-05-06 12:00:00  11.15000  66.0000  ...         1         0      1
4    2002-05-06 15:00:00  11.44240  65.8147  ...         1         0      1
...                  ...       ...      ...  ...       ...       ...    ...
2835 2019-12-06 00:00:00   8.30000  53.4000  ...         1         0      3
2836 2019-12-06 03:00:00   8.14246  53.1113  ...         1         0      3
2837 2019-12-06 06:00:00   7.95000  52.7500  ...         1         0      3
2838 2019-12-06 09:00:00   7.85997  52.5475  ...         1         0      3
2839 2019-12-06 12:00:00   7.85000  52.1500  ...         1         0      3
```

Finally, because cyclones in the north Indian Ocean are seasonal. A new column was created which consisted of the month in which the cyclone occurred.

```
df['month'] = df['ISO_TIME'].dt.month
print(df)
```

```
                ISO_TIME       LAT      LON  ...  Basin_BB  GRADE  month
0    2002-05-06 03:00:00   9.30004  66.8800  ...         0      1      5
1    2002-05-06 06:00:00   9.96667  66.5000  ...         0      1      5
2    2002-05-06 09:00:00  10.61360  66.2109  ...         0      1      5
3    2002-05-06 12:00:00  11.15000  66.0000  ...         0      1      5
4    2002-05-06 15:00:00  11.44240  65.8147  ...         0      1      5
...                  ...       ...      ...  ...       ...    ...    ...
2835 2019-12-06 00:00:00   8.30000  53.4000  ...         0      3     12
2836 2019-12-06 03:00:00   8.14246  53.1113  ...         0      3     12
2837 2019-12-06 06:00:00   7.95000  52.7500  ...         0      3     12
2838 2019-12-06 09:00:00   7.85997  52.5475  ...         0      3     12
2839 2019-12-06 12:00:00   7.85000  52.1500  ...         0      3     12
```

Standardization

StandardScaler standardizes a feature by subtracting the mean and then scaling to unit variance. Unit variance means dividing all the values by the standard deviation. It results in a distribution with a standard deviation equal to 1. The variance is equal to 1 also, because variance is equal to standard deviation squared.

The StandardScaler makes the mean of the distribution 0. The formula for standardization is given below

$$z = \frac{X - \mu}{\sigma} =$$

The symbols are:

X: the observation (a specific value that you are calculating the z-score for).

Mu(μ): the mean.

Sigma(σ): the standard deviation.

Transforming the data in order to input into LSTM

The input to every LSTM layer must be three-dimensional.

The three dimensions of this input are:

- **Samples**. One sequence is one sample. A batch is comprised of one or more samples.
- **Time Steps**. One time step is one point of observation in the sample.
- **Features**. One feature is one observation at a time step.

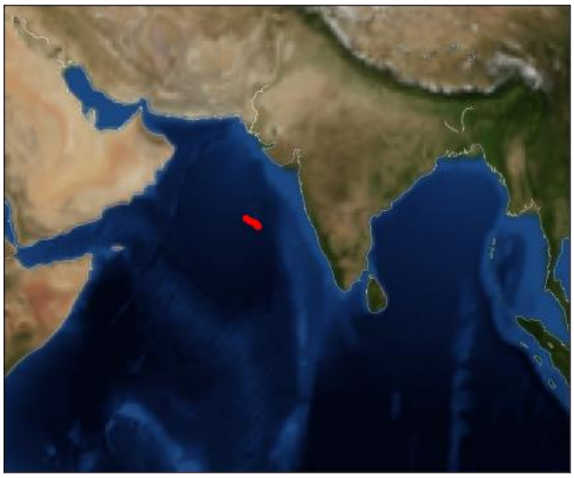
This means that the input layer expects a 3D array of data when fitting the model and when making predictions, even if specific dimensions of the array contain a single value, e.g. one sample or one feature.

When defining the input layer of your LSTM network, the network assumes you have 1 or more samples and requires that you specify the number of time steps and the number of features.

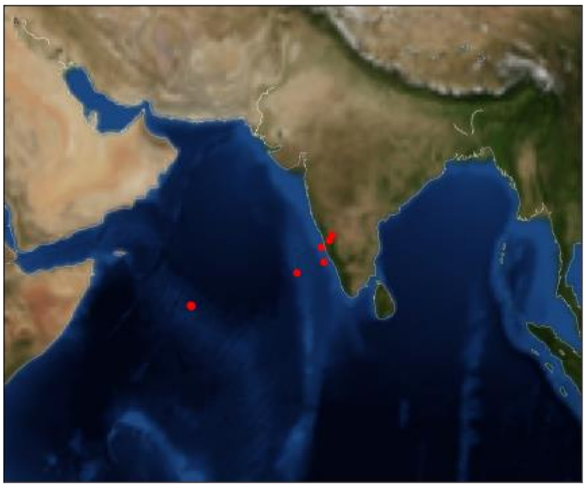
```
[ ]  trainX1 = []
     trainY1 = []

     n_future = 1
     n_past = 8

     for i in range(n_past, len(scaled_df) - n_future +1):
         trainX1.append(scaled_df[i - n_past:i, 0:scaled_df.shape[1]])
         trainY1.append(scaled_df[i + n_future - 1:i + n_future, 0])

     trainX1, trainY1 = np.array(trainX1), np.array(trainY1)
```

Results

The model predicts the track of the latest storm over the next seven days. The predicted track and the actual track has been plotted on the map as follows.



1: Predicted Track



2: Real Track

The above result clearly shows that the model needs improvement. This could be in the form of more data i.e more storms in the data and more detailed data as in data containing sea surface temperature, precipitation, different components of wind motion and speed etc.

In this model the latitude and longitude have been predicted separately. In order get a more wholesome prediction a model should be able to predict the latitude and the longitude collectively.

Lastly, The model can also be improved by using a different type of deep learning approach altogether such as Convolutional Neural Network, Deep Boltzmann Machine etc.

References

1. https://www.udemy.com/course/machinelearning/
2. https://www.udemy.com/course/python-the-complete-python-developer-course/
3. https://towardsdatascience.com/
4. https://machinelearningmastery.com/
5. https://github.com/dhar2020/Cyclone_Track_RNN_LSTM.git (For Code and Data)