

Boosting Tropical Cyclone Prediction Efficiency Through Time Series Analysis: An Ensemble Learning Approach

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering

by

CHILAMKURTHI DINESH SURYA PAVAN (Reg. No. 37110143)

DEVABHAKTHUNI SAI VENKATA ANUDEEP (Reg. No 37110180)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC | 12B Status by UGC | Approved by
AICTE**

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600 119

MARCH – 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with “A” grade by NAAC I 12B Status by
UGC I Approved by AICTE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **CHILAMKURTHI DINESH SURYA PAVAN (Reg. No. 37110143)** and **DEVABHAKTHUNI SAI VENKATA ANUDEEP (Reg. No. 37110180)** who carried out the project entitled **“Boosting Tropical Cyclone Prediction Efficiency Through Time Series Analysis: An Ensemble Learning Approach”** under my supervision from November 2020 to March 2021.

Internal Guide

Mr. S. Dhamodaran, M.Tech., (Ph.D)

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **DEVABHAKTHUNI SAI VENKATA ANUDEEP** hereby declare that the Project Report entitled “**Boosting Tropical Cyclone Prediction Efficiency Through Time Series Analysis: An Ensemble Learning Approach**” is done by me under the guidance of **Mr. S. Dhamodaran, M.Tech., (Ph.D)** Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D.**, Dean, School of Computing and **Dr. L. Lakshmanan M.E., Ph.D.**, and **Dr. S. Vigneshwari M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mr. S. Dhamodaran, M.Tech., (Ph.D)** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Hurricanes are powerful and deep tropical cyclones that come along could cause serious human and property losses. The causes of hurricane are complex and factors affecting hurricane intensity vary. The current intensity predictions are based on hurricane movement patterns and statistical methods are greatly influenced by expert experience, so prediction results are not satisfying. Although machine learning and statistical methods have been extensively used to study cyclones.

The statistical analysis of tropical cyclone is very crucial. The current dynamic statistical analysis of tropical cyclone could computationally blow up and have more enhancement on the prediction. This research is based on ensemble learning for predicting tropical cyclone. A multi-dimensional element choice layer is proposed in our technique to pick the most associated meteorological factors and time reach to typhoon tracks from the point of view of property relationship examination and transient connection investigation. This layer is fundamental for assessing the effects of meteorological factors on typhoon track determining.

TABLE OF CONTENTS

ABSTRACT

LIST OF FIGURES

CHAPTER No.	TITLE	PAGE NO.
1	INTRODUCTION	1
	1.1 ABOUT THE PROJECT	1
	1.2 NEURAL NETWORK	1
	1.3 ENSEMBLE LEARNING	2
	1.4 MACHINE LEARNING	3
2	LITERATURE SURVEY	4
	2.1 LITERATURE SURVEY 1	4
	2.2 LITERATURE SURVEY 2	5
	2.3 LITERATURE SURVEY 3	5
3	AIM AND SCOPE OF THE INVESTIGATION	7
	3.1 AIM OF PROJECT	7
	3.2 SCOPE	7
	3.2.1 ADVANTAGES	7
4	SYSTEM DESIGN & METHODOLOGY	8
	4.1 EXISTING SYSTEM	8
	4.1.1 DRAWBACKS OF EXISTING SYSTEM	8
	4.2 PROPOSED SYSTEM	8
	4.2.1 ADVANTAGES OF PROPOSED SYSTEM	9
	4.3 SYSTEM REQUIREMENTS	9
	4.3.1 HARDWARE SPECIFICATION	9
	4.3.2 SOFTWARE SPECIFICATION	9
	4.4 ENSEMBLE APPROACH	10
	4.4.1 BOOSTING	10
	4.4.2 GRADIENT TREE BOOSTING	11
	4.5 ARCHITECTUREL DESIGN	12

	4.6 MODULES	12
	4.6.1 DATA PREPROCESSING	12
	4.6.2 ENSEMBLE ANALYSIS	13
	4.6.3 ENSEMBLE PREDICTION	14
5	EXECUTION AND RESULTS	15
	5.1 EXECUTING THE PROJECT	15
	5.2 RESULTS	15
6	CONCLUSION AND FUTURE ENHANCEMENT	21
	6.1 CONCLUSION	21
	6.2 FUTURE ENHANCEMENT	27
	REFERENCES	22
	APPENDIX	
	A. SOURCE CODE	24

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
4.4	architecture diagram	12
5.1	simple deep representation by classification model	15
5.2	collecting inputs from datasets	15
5.3	summary of data preprocessing	16
5.4	predicted output	17
5.5	layer visualizations	18
5.6	image of layer 0	19
5.7	image of layer 1	19
5.8	image of layer 5	20

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Cyclones, particularly on the eastern seaboard of India, have consistently been a significant reason for misfortune to lives and property. The precise expectation of the way and landfall of a cyclone can carry a colossal alleviation to beach front populaces that are in its way. The figure/cautioning of hefty precipitation incorporates season of initiation, term, territory of event and power of substantial precipitation. The conjecture of hurricane winds includes anticipating season of beginning, length, territory of event, and extent. Aside from the four strategies referenced above, expectation of storm winds incorporates dynamical factual methods. In this paper, we adapt the method with multi-task modular backpropagation that features Ensemble learning. The method is tested on benchmark chaotic ensemble problems and compared with its counterparts. The results show that the method can alleviate the problems associated with timely convergence of the neuro evolution approach and provides better performance.

1.2 NEURAL NETWORK

Generally, a Neural Network (NN) contains a compilation of artificial neurons which are unified to execute some computation on input data and produces output forms. In deep learning, these layered portrayals are learned through models called neural network (NN), organized in exacting layers stacked on top of one another. The term neural organization is a reference to neurobiology, yet albeit a portion of the focal ideas in deep learning were created to some degree by drawing motivation from our comprehension of the mind, deep-learning models are not modelling of the cerebrum. There's no proof that the mind executes anything like the learning instruments utilized in current deep-learning models. You may run over pop-science articles broadcasting that deep learning works like the cerebrum or was designed according to the mind, yet that isn't the situation. It would be mistaking and counterproductive for newcomers to the field to consider deep learning being in any capacity identified with neurobiology; you needn't bother with that cover of "much the same as our personalities" persona and secret, and you should fail to remember

anything you may have found out about speculative connections between deep learning and science. For our motivations, deep learning is a numerical system for learning portrayals from information.

Deep learning has arrived at a degree of public consideration and industry speculation at no other time found throughout the entire existence of Artificial Intelligence, yet it isn't the primary effective type of machine learning. It's protected to say that a large portion of the machine-learning calculations utilized in the business today aren't deep learning calculations. Deep learning isn't generally the correct apparatus for the work—at times there isn't sufficient information for deep learning to be pertinent, and in some cases the issue is better tackled by an alternate calculation. In the event that deep learning is your first contact with machine learning, at that point you may end up in a circumstance where the sum total of what you have is the deep-learning hammer, and each machine-learning issue begins to resemble a nail.

1.3 ENSEMBLE LEARNING

In statistics and machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone. Unlike a statistical ensemble in statistical mechanics, which is usually infinite, a machine learning ensemble consists of only a concrete finite set of alternative models, but typically allows for much more flexible structure to exist among those alternatives.

An ensemble is itself a supervised learning algorithm, because it can be trained and then used to make predictions. The trained ensemble, therefore, represents a single hypothesis. This hypothesis, however, is not necessarily contained within the hypothesis space of the models from which it is built. Thus, ensembles can be shown to have more flexibility in the functions they can represent. This flexibility can, in theory, enable them to over-fit the training data more than a single model would, but in practice, some ensemble techniques (especially bagging) tend to reduce problems related to over-fitting of the training data. Empirically, ensembles tend to yield better results when there is a significant diversity among the models. Many ensemble methods, therefore, seek to promote diversity among the models they combine. Although perhaps non-intuitive, more random algorithms

(like random decision trees) can be used to produce a stronger ensemble than very deliberate algorithms (like entropy-reducing decision trees). Using a variety of strong learning algorithms.

1.4 MACHINE LEARNING

Machine Learning combines computer science, mathematics, and statistics. Statistics is essential for drawing inferences from the data. Mathematics is useful for developing machine learning models and finally, computer science is used for implementing algorithms. However, simply building models is not enough. You must also optimize and tune the model appropriately so that it provides you with accurate results. Optimization techniques involve tuning the hyperparameters to reach an optimum result. The world today is evolving and so are the needs and requirements of people. Furthermore, we are witnessing a fourth industrial revolution of data. In order to derive meaningful insights from this data and learn from the way in which people and the system interface with the data, we need computational algorithms that can churn the data and provide us with results that would benefit us in various ways. Machine Learning has revolutionized industries like medicine, healthcare, banking, and several other industries. Therefore, Machine Learning has become an essential part of modern industry. Data is expanding exponentially and in order to harness the power of this data, added by the massive increase in computation power, Machine Learning has added another dimension to the way we perceive information. Machine Learning is being utilized everywhere. The electronic devices you use, the applications that are part of your everyday life are powered by powerful machine learning algorithms. With an exponential increase in data, there is a need for having a system that can handle this massive load of data. Machine Learning models like Deep Learning allow the vast majority of data to be handled with an accurate generation of predictions. Machine Learning has revolutionized the way we perceive information and the various insights we can gain out of it. These machine learning algorithms use the patterns contained in the training data to perform classification and future predictions. Whenever any new input is introduced to the ML model, it applies its learned patterns over the new data to make future predictions.

CHAPTER 2

LITERATURE SURVEY

2.1 LITERATURE SURVEY 1

Jeberson Retna Raj, Senduru Srinivasulu “Change Detection of Images Based on Multivariate Alteration Detection Method” Year 2020.

Efficiency are Rapid response to the complex problems, Rapid response to the decision making, Removing the speckle noises, Improve the accuracy of segmentation and Drawbacks are Noises are not completely removing, Not an Automatic Feature Engineering, Narrowly specialized knowledge

Change detection in heterogeneous satellite image is paramount important for analysing and interpreting the changes in Region of Interest (ROI) of earth surface. The illumination of light effect at night before and after the cyclone is released by NASA open-source images. The image is co-registered for loading the image for reading and calibrated for setting the world coordinates for further processing. Generally, the satellite image by default contains speckle noise due to the position of sensor of satellite for capturing the land surface and the atmospheric effect. The speckle noise is to be removed by the pre-processing algorithm. The noise free image is used as the input for segmentation and change detection. The images are co-registered and calibrated for fit in the tool for process the image. The speckle noise is removed from the image and the resultant image is given as an input for predict regression. To improve the accuracy of segmentation and multivariate alteration detection algorithm, the output of the predict regression is used. The Large-Scale Mean Shift Segmentation (LSMS) is used to segment the ROI image detail with other part of the image. The multivariate alteration detection algorithm used to detect the changes in the two satellite images. Finally, the severity of changes can be estimated with the prediction algorithm.

2.2 LITERATURE SURVEY 2

Yanfei Zhang Rohitash Chandra Junbin Gao “Cyclone Track Prediction with Matrix Neural Networks” Year 2018.

Efficiency are Supports universal function approximators, supports spatial correlation, such as those that involve coordinates in cyclone tracks, Closest to the true cyclone trajectory. Drawbacks are Does not support vector-based input in conventional FNN, offset with a high value when compared to the initial point of the true trajectory, High Processing Cost

We first describe conventional feedforward neural networks (FNNs) that are known as universal function approximators which makes them appropriate for regression and time series prediction. FNNs typically have one or more hidden layer(s) and an output layer. Each layer features neurons (represented as a vector) that are interconnected with the weights that are represented as a matrix. Each cyclone features the latitude, longitude, wind intensity recorded at 6-hour regular intervals for the duration of the cyclone. We only consider the track information given by the latitude and longitude and discard the wind-intensity, which can be considered for analysis in future work. Each cyclone generally occurs at a unique initial location and duration given by the total time (time steps T_j). The unique initial location accounts special bias in the dataset. This is due to the way the latitude and longitude given as coordinates describe the cyclone track. For instance, if we consider two cyclones that have the same track pattern at different initial locations, due to the spatial bias, these cyclones would be interpreted as different cyclones by the model.

2.3 LITERATURE SURVEY 3

Rohitash Chandra “Multi-task Modular Backpropagation for Dynamic” Time Year 2018.

Efficiency are Support multi-task modular back-propagation, increasing order or size of the cascade, supports transfer and restore knowledge. Drawbacks are Restore mechanism is not implemented, Transfer optimisation exploits solutions from related problems, Difficulties to obtain better performance

The optimisation problem is learning the weights of a cascaded neural network architecture. The base problem is the network module that is defined by lowest number of input features and hidden neurons. The weights in the base network are part of larger cascaded network modules that consist of more hidden neurons and input features. This can be viewed as modules of knowledge that are

combined for larger subtasks that use knowledge from smaller subtasks as building blocks. The cascaded network architecture can also be viewed as an ensemble of neural networks that feature distinct topologies in terms of number input and hidden neurons. Since we consider one step ahead time series problem, one neuron in output layer is used for all the respective modules. We note that a task of prediction given different sets of input features is not similar to conventional multi-task learning where the output of the network defines the task. In our case, the task is defined by the input of the network, whereas the output of the network is fixed. Hence, we refer to the notion of tasks as subtasks since it is defined by the input of the network. A subtask refers to the problem of learning the given modules that are constructed given timespan S_n in a cascaded manner.

CHAPTER 3

AIM AND SCOPE OF THE INVESTIGATION

3.1 AIM OF PROJECT

The primary aim of this proposed system Boosting Tropical Cyclone Prediction Efficiency Through Time Series Analysis: An Ensemble Learning Approach is to predict the cyclone from the different past occurred cyclone in the world which will help the peoples in near the sea areas and storms that came along could cause serious human and property losses. The causes of hurricane are complex and factors affecting hurricane intensity vary. The current intensity predictions are based on hurricane movement patterns and statistical methods are greatly influenced by expert experience

3.2 SCOPE

We adapt the method with multi-task modular backpropagation that features Ensemble learning. The method is tested on benchmark chaotic ensemble problems and compared with its counterparts. The results show that the method can alleviate the problems associated with timely convergence of the neuro evolution approach and provides better performance

3.2.1 ADVANTAGES

- Autocorrelation analysis
- Efficiency in the ensemble analysis
- Better Prereduction
- Analyze both the linear and non-linear correlation
- Resources are optimized

CHAPTER 4

SYSTEM DESIGN & METHODOLOGY

4.1 EXISTING SYSTEM

The existing system is a data-driven deep learning model to predict tropical cyclone tracks using the spatial locations and multiple meteorological factors. This model comprises a multidimensional feature selection layer, a CNN layer and a GRU layer. The proposed model was trained using a dataset of real-world tropical cyclones from the years 1945 to 2017. Through the comparison experiments, the results verify that the proposed model outperforms the traditional forecasting methods, including a climatologically aware forecasting technique, the Sanders Barotropic technique and a numerical weather prediction (NWP) model. A multi-dimensional feature selection layer is proposed in our method to choose the most correlated meteorological variables and time range to tropical cyclone tracks from the perspective of attribute correlation analysis and temporal correlation analysis. This layer is essential for evaluating the impacts of meteorological variables on tropical cyclone track forecasting

4.1.1 DRAWBACKS OF EXISTING SYSTEM

- Cannot analyse the implicit data features
- Complex correlation of meteorological variables is not considered
- Time series variable is not stable
- Test are significantly less than the critical statistics
- Will not culling abnormal data

4.2 PROPOSED SYSTEM

we apply multi-task modular backpropagation for the application of dynamic ensemble prediction. This provides an alternate approach for training cascaded network architectures for dynamic ensemble that have typically been done through. Multi-task learning has also inspired optimisation methods such as evolutionary algorithms were transfer optimisation exploits solutions from related problems. The different subtasks were implemented as different topologies given by the number of hidden neurons with the goal to store knowledge in modules so that perturbation of

certain modules does not affect the decision-making process as a whole. The set of input features defines the module of knowledge in a cascaded network architecture that addresses the dynamic nature of the problem. The set of input features is defined as the timespan S_n which is essentially a subset of the embedding dimension (D) from space reconstruction via Taken's theorem. The timespan is composed of overlapping data-points. Data reduction: In this step we reduced the representation of dataset in smaller volumes, we removed irrelevant attributes. The result of this step is two data sets on the form of ensemble. Hydrological time data are affected by seasonal, topographic, weather and other natural conditions. It has both temporal correlation and nonlinear relationship. The deterministic components, periodic components and uncertain factors of hydrological ensemble make the single model have limited space to improve the forecast accuracy.

4.2.1 ADVANTAGES OF PROPOSED SYSTEM

- Autocorrelation analysis
- Reduce the dimension of input variables and remove irrelevant noise data
- Efficiency in the ensemble analysis
- Analyze both the linear correlation or non-linear correlation between variables
- Resources are optimized

4.3 SYSTEM REQUIREMENTS

4.3.1 HARDWARE SPECIFICATION

- Hardware: Minimum Requirement
- Disk Space: 32 GB or more, 10 GB or more for Foundation Edition
- Processor: 1.4 GHz 64 bit
- Memory: 512 MB
- Display: (800 × 600) Capable video adapter and monitor

4.3.2 SOFTWARE SPECIFICATION

- Python
- Jupyter Notebook
- NumPy, Keras

4.4 ENSEMBLE APPROACH

The ensemble learning algorithm to make tropical cyclone prediction. We combined various learning algorithm into ensemble. When utilizing the sacking strategy, we consolidate the aftereffect of numerous choice trees and produce a solitary yield/forecast by taking a lion's share check. Under an alternate examining instrument, the outcomes had been joined to deliver a solitary forecast for the arbitrary woods. Under a successive mistake decrease technique for choice trees, the boosting strategy likewise gives improved answers. Despite the fact that we are managing dubious information, which includes probabilities, we don't plan to have strategies that give results out of a black box and carry on without reliable arrangements. A hypothesis ought to clarify the working and we need a confirmation that the outcomes will be steady and there is no dark enchantment about it

In order for ensemble methods to be more accurate than any of its individual members, the base learners have to be as accurate as possible and as diverse as possible, to convert the weak learns into strong learners we boosting in it.

4.4.1 BOOSTING

Boosting refers to a family of algorithms that are able to convert weak learners to strong learners. The main principle of boosting is to fit a sequence of weak learners– models that are only slightly better than random guessing, such as small decision trees– to weighted versions of the data. More weight is given to examples that were misclassified by earlier rounds.

The predictions are then combined through a weighted majority vote (classification) or a weighted sum (regression) to produce the final prediction. The principal difference between boosting and the committee methods, such as bagging, is that base learners are trained in sequence on a weighted version of the data.

A simple imputation model of neurons is perception. Perception is one of the earliest deep learning algorithms for binary classification. Typically, a single neuron will contain one or more inputs, a cell body and only one output. The incredible property of the perceptron calculation is that, if there is a bunch of boundary esteems that empowers the perceptron to order all the training set accurately, the calculation is ensured to it. Shockingly, for most certifiable models there is no such set. Then

again, and, after its all said and done perceptron's frequently function admirably as in there are boundary settings that mark an exceptionally high level of the models effectively. The calculation works by emphasizing over the training set a few times, changing the boundaries to expand the quantity of effectively identified models. In the event that we get however the training set with no boundaries expecting to transform, we realize we have a right set, and we can stop. For example, a single $256 * 256$ image would have 65,536 inputs, one for each pixel as a input to neuron and that neuron will have only one output. Any perception contains a vector of weights

$$W = [w_1, w_2, w_3, \dots, w_n]$$

(equation 4.1)

Each weight corresponds to one input. It will also contain one notable weight called bias say b . The variable w and b are considered as parameters for the perception. With these inputs the perception can be derived as below function

$$f_{\Phi}(x) = \begin{cases} 1 & \text{if } b + \sum_{i=1}^l x_i w_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

(equation 4.2)

he above equation can be simplified as below

$$f_{\Phi}(x) = \begin{cases} 1 & \text{if } b + w \cdot x > 0 \\ 0 & \text{otherwise} \end{cases}$$

(equation 4.3)

4.4.2 GRADIENT TREE BOOSTING

It is a generalization of boosting to arbitrary differentiable loss functions. It can be used for both regression and classification problems. Gradient Boosting builds the model in a sequential way.

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

(equation 4.4)

At each stage the decision tree $h_m(x)$ is chosen to minimize a loss function L given the current model $F_{m-1}(x)$:

$$F_m(x) = F_{m-1}(x) + \underset{h}{\operatorname{argmin}} \sum_{i=1}^n L(y_i, F_{m-1}(x_i) + h(x_i))$$

(equation 4.5)

The algorithms for regression and classification differ in the type of loss function used.

4.4 ARCHITECTURAL DESIGN

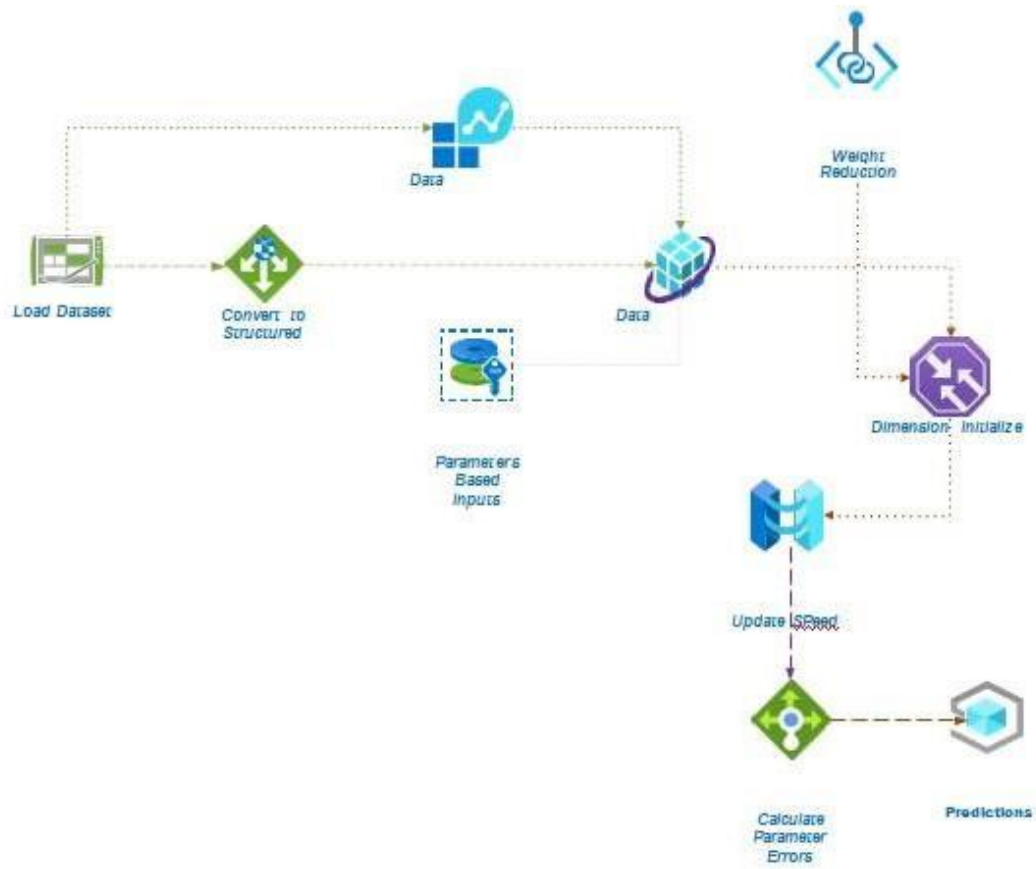


Fig: 4.1 architecture diagram

4.5 MODULES

4.5.1 DATA PREPROCESSING

Ensemble models are very useful models when you have serially correlated data. Most of business houses work on ensemble data to analyse sales number for the next year, website traffic, competition position and much more. However, it is also one of the areas, which Chapter 6 System Implementation many analysts do not understand. After the analytical model removed data errors and redundancies, the next step in changing the unstructured data to structured data is data modelling. Analysts' study data relationships and mark co-relations in what could be a lengthy process, but a very important one, as these data relationships give an insight into what intelligence a business can drive from them. Data modelling differs from case to case and client to client. Organizations have to figure out for themselves the accuracy they need. The final step is data visualization, driving insights from the now structured data through visual aids. There are numerous graphs and charts organizations may deploy to visualize data, so an intelligent evaluation is important with the choices of data visualizations tools at hand.

4.5.2 ENSEMBLE ANALYSIS

Ensemble data often arise when monitoring industrial processes or tracking corporate business metrics. The essential difference between modelling data via ensemble methods or using the process monitoring methods. Ensemble analysis accounts for the fact that data points taken over time may have an internal structure (such as autocorrelation, trend or seasonal variation) that should be accounted for. Ensemble data is opposed to cross sectional data which observes individuals, companies, etc. at a single point in time. Because data points in ensemble are collected at adjacent time periods there is potential for correlation between observations. This is one of the features that distinguishes ensemble data from cross-sectional data.

Inherent in the collection of data taken over time is some form of random variation. There exist methods for reducing or cancelling the effect due to random variation. An often-used technique in industry is "smoothing". This technique, when properly applied, reveals more clearly the underlying trend, seasonal and cyclic components. A ensemble graph plots observed values on the y-axis against an

increment of time on the x-axis. These graphs visually highlight the behaviour and patterns of the data and can lay the foundation for building a reliable model. Mean reverting data returns, over time, to a time-invariant mean. It is important to know whether a model includes a non-zero mean because it is a prerequisite for determining appropriate testing and modelling methods.

4.5.3 ENSEMBLE PREDICTION

Boosting is an iterative technique which adjusts the weight of an observation based on the last classification. If an observation was classified incorrectly, it tries to increase the weight of this observation and vice versa.

Boosting in general decreases the bias error and builds strong predictive models. Boosting has shown better predictive accuracy than bagging, but it also tends to over-fit the training data as well. Thus, parameter tuning becomes a crucial part of boosting algorithms to make them avoid overfitting.

Boosting is a sequential technique in which, the first algorithm is trained on the entire data set and the subsequent algorithms are built by fitting the residuals of the first algorithm, thus giving higher weight to those observations that were poorly predicted by the previous model.

CHAPTER 5

EXECUTION AND RESULTS

5.1 EXECUTING THE PROJECT

The end project is run in the Jupyter notebook, For the inputs the program will collect the three types of the data sets they are test, train, validation from the datasets to predict the output from it. It will validate in the layer wise format to get the higher prediction. As shown in below figure

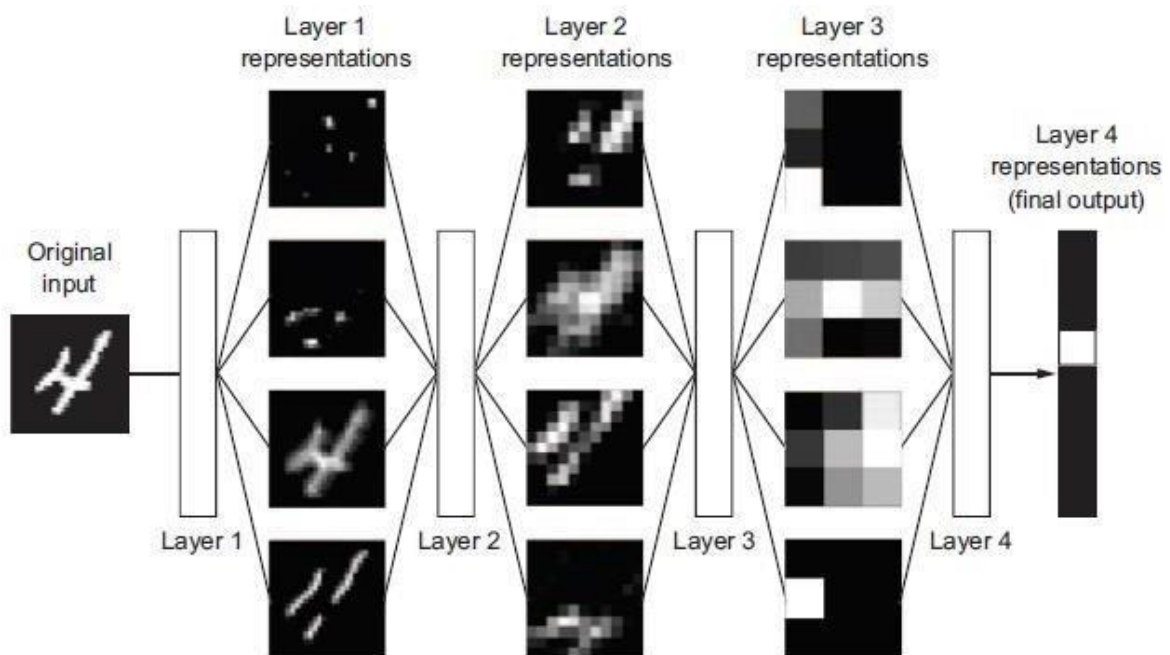


Fig: 5.1 Simple deep representation by classification model

5.2 RESULTS

```
In [30]: print('Check if samples are of same size:\n\nTrain:')
print(len(os.listdir(train_dir_c)), len(os.listdir(train_dir_w)))

Check if samples are of same size:

Train:
400 400

In [31]: print('Validation:')
print(len(os.listdir(validation_dir_c)), len(os.listdir(validation_dir_w)))

Validation:
100 100

In [32]: print('Test:')
print(len(os.listdir(test_dir_c)), len(os.listdir(test_dir_w)))

Test:
100 100
```

Fig: 5.2 collecting inputs from datasets

The Fig 5.2 is the collecting inputs from datasets. Here we have total three datasets they are train, validation, and test. In train dataset total 400 images are collected in validation dataset total 100 images are collected and in test dataset total 100 images are collected. The program will pick the random images to test, train and validating the images

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 178, 178, 32)	896
max_pooling2d (MaxPooling2D)	(None, 89, 89, 32)	0
conv2d_1 (Conv2D)	(None, 87, 87, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 43, 43, 64)	0
conv2d_2 (Conv2D)	(None, 41, 41, 128)	73856
max_pooling2d_2 (MaxPooling2D)	(None, 20, 20, 128)	0
conv2d_3 (Conv2D)	(None, 18, 18, 128)	147584
max_pooling2d_3 (MaxPooling2D)	(None, 9, 9, 128)	0
conv2d_4 (Conv2D)	(None, 7, 7, 256)	295168
max_pooling2d_4 (MaxPooling2D)	(None, 3, 3, 256)	0
flatten (Flatten)	(None, 2304)	0
dropout (Dropout)	(None, 2304)	0
dense (Dense)	(None, 512)	1180160
dense_1 (Dense)	(None, 2)	1026
Total params: 1,717,186		
Trainable params: 1,717,186		
Non-trainable params: 0		

Fig: 5.2 summary of data preprocessing

The Fig 5.2 is the summary of data preprocessing. This will show the summary of the image preprocessing, in this it will convert the weak learners into strong learners by eliminating the noise, converting into 2d image, dense layer. Here to do all this we are using the methods which was provided by the keras library con2d, max-pooling, flatten, dense layers as we can see from the above figure these are used to convert the weal image into strong image which is used to get better prediction.

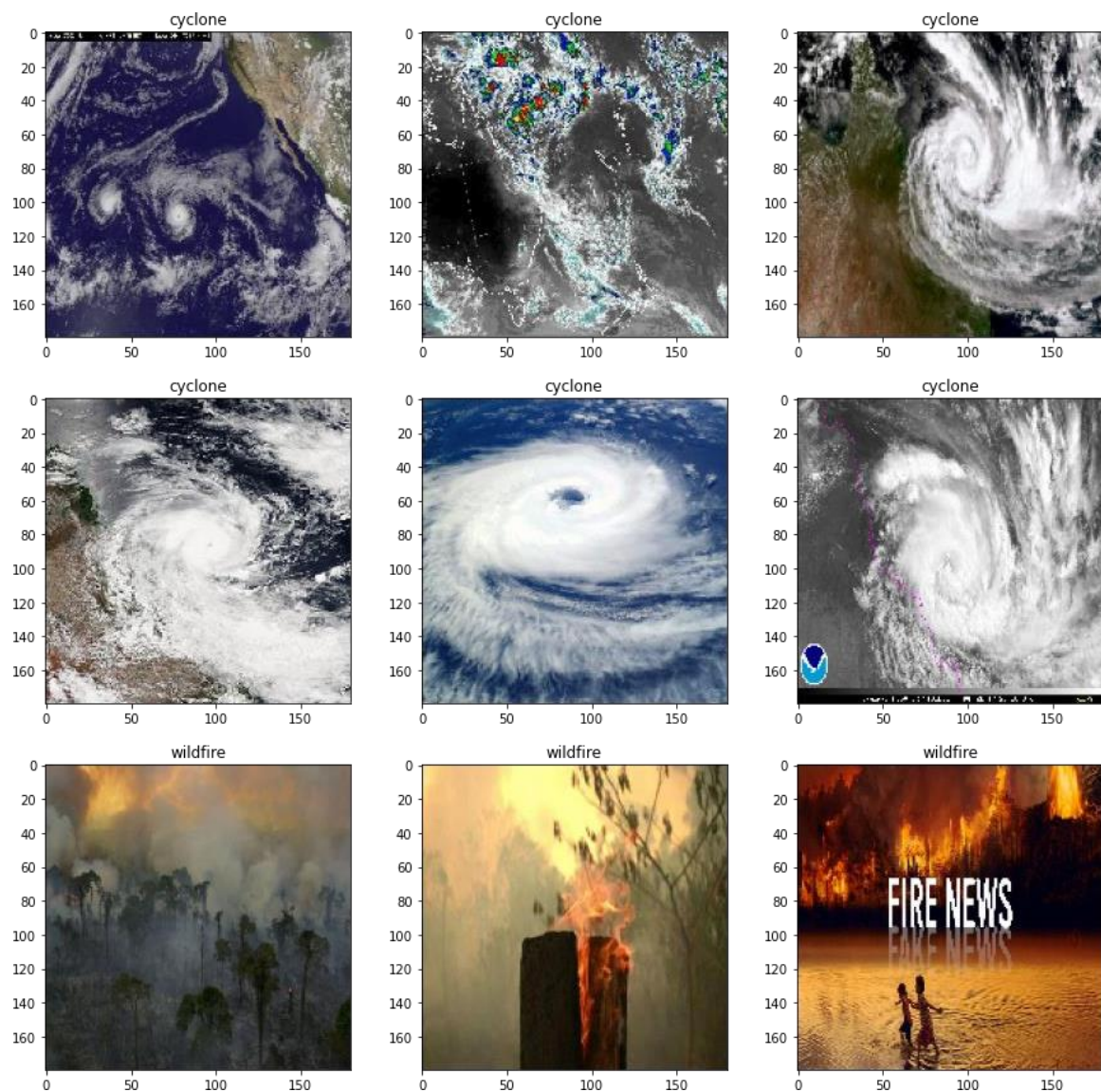


Fig: 5.3 predicted output

The Fig 5.3 is the predicted output, here by using algorithm and collected inputs from the datasets it will predict the image and display it here, to check the output here included some wildfire inputs also to get the best predicted output. These are

the randomized image outputs to show the predicted images as we can see in figure it shows the output in the above of each predicted image.

The below image should present the cyclone

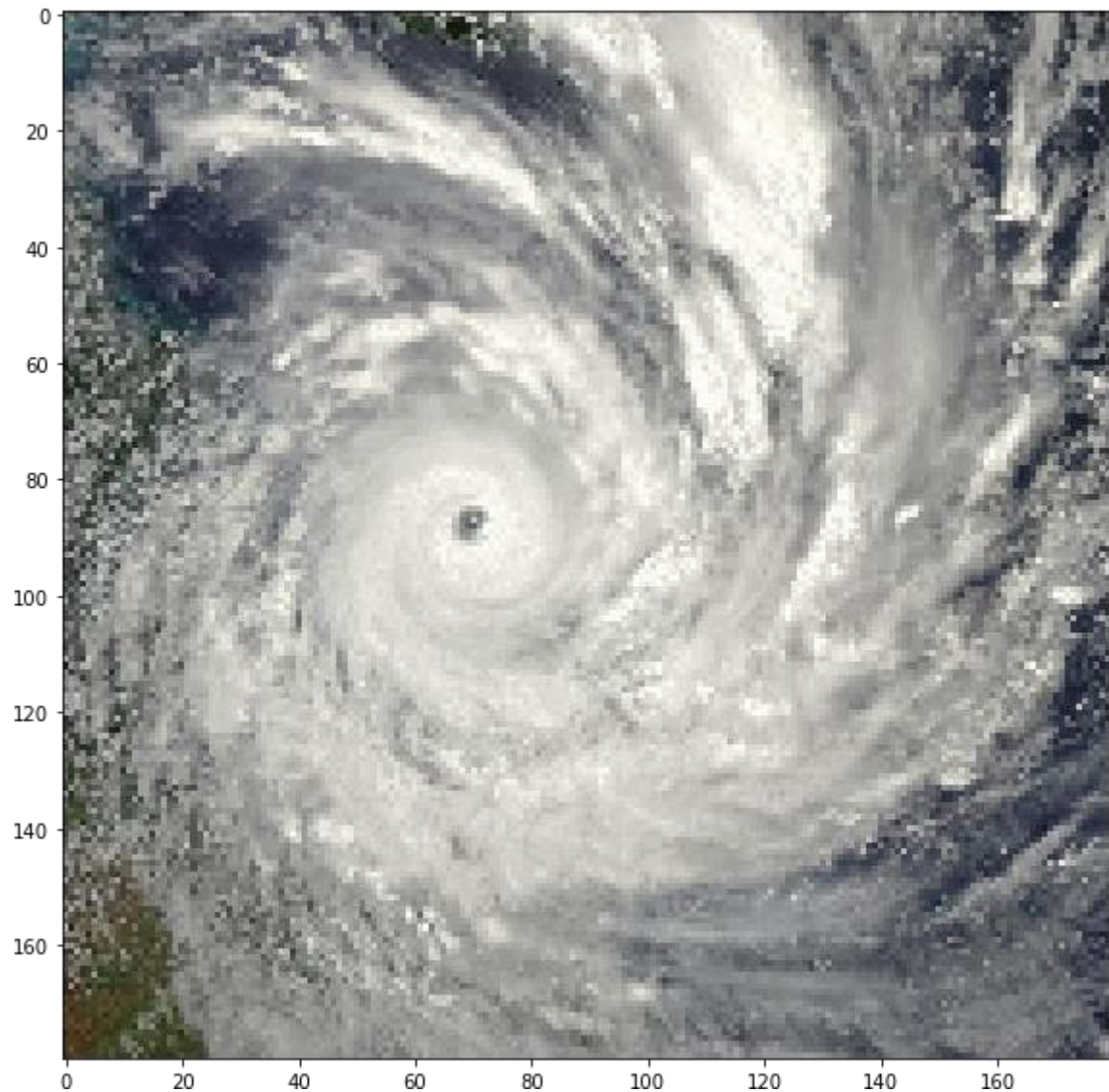


Fig: 5.4 layer wise visualizations

The Fig 5.4 is the layer wise visualizations. Here to show how the image are predilected by the layer wise prediction of the images in visualization it was show the five layers of the visualization the following images are the images from the layers

Images from the layer 0

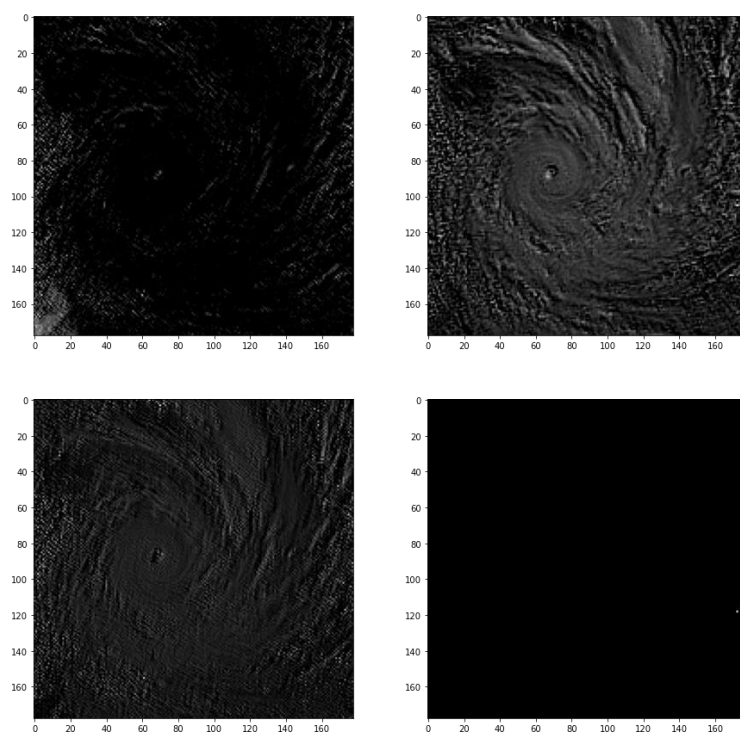


Fig: 5.5 image of layer 0

Images from the layer 1

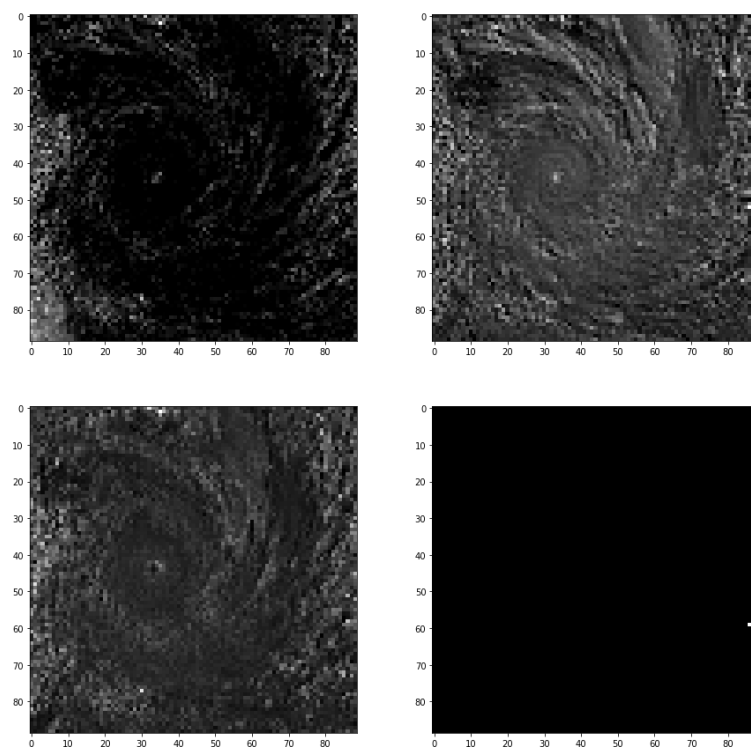


Fig: 5.6 images of layer 1

Images from the layer 5

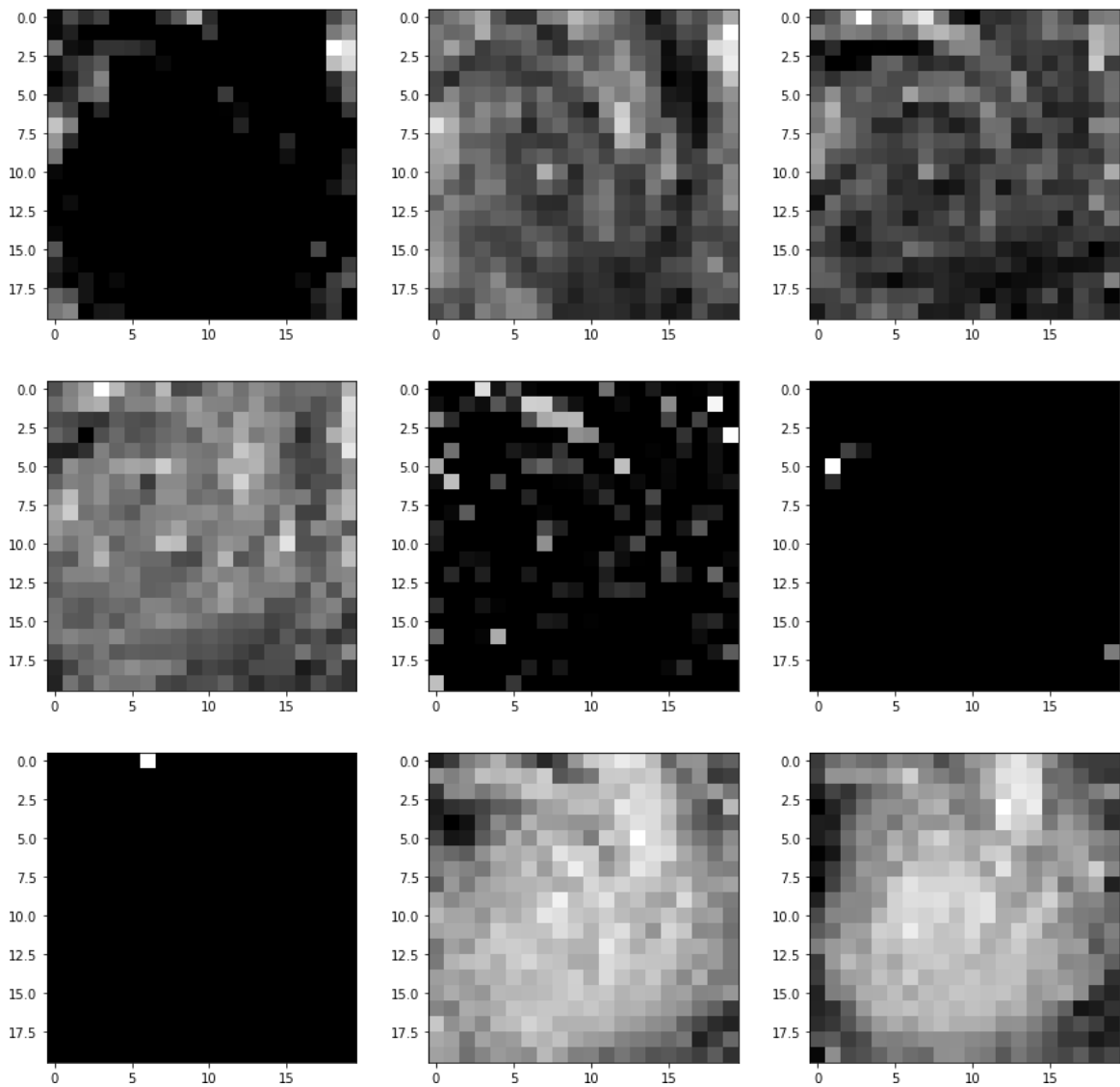


Fig: 5.7 images of layer 5

The Fig 5.5, 5.6. 5.7 are the images of layers, it shows the deep visualization of the input images to get better predicted output from the input which is taken from the input dataset randomly.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The regression prediction algorithm used to fill the missing values and the output of this step used to increase the accuracy. A scope of mechanical advances has now made cyclone gauging significantly more precise, along these lines decreasing dangers. Alongside cyclones, issues related with cyclones, for example, precipitation, storm flood and seaside immersion can likewise be precisely figure regarding greatness, time and spot of event. This has brought about decrease in loss of lives, and cut down the costs engaged with departure of populace.

The exhibition of these calculations with the given engineering is looked at and the outcomes are examined. We additionally eliminated the spatial inclinations in the first dataset by normalizing all the cyclones with their beginning areas. The outcomes show that the proposed strategy accomplished better execution when contrasted with three set up Time arrangement strategies from the writing. The results show that the proposed method achieved better performance when compared to three established Time series methods from the literature.

6.2 FUTURE ENHANCEMENT

In future, the severity of damages, loss of crop, vegetation, agriculture can be estimated with the help of other deep learning algorithms. The same algorithm can be applied in medical images to detect the changes in cells and tissues which help the user to take decisions.

REFERENCE

- [1] A sparse recurrent neural network for trajectory prediction of Atlantic hurricanes, in Proc. Genetic Evol. Comput. Conf. (GECCO), 2016, pp. 957 964.
- [2] J. Chen, L. Yang, Y. Zhang, M. Alber, and D. Z. Chen, ``Combining fully convolutional and recurrent neural networks for 3D biomedical image segmentation," in Proc. Adv. Neural Inf. Process. Syst., 2016, pp. 3036 3044.
- [3] K. He, X. Zhang, S. Ren, and J. Sun, ``Deep residual learning for image recognition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770 778.
- [4] L. Chen, X. Pan, Y.-H. Zhang, M. Liu, T. Huang, and Y.-D. Cai, ``Classification of widely and rarely expressed genes with recurrent neural network," Comput. Struct. Biotechnol. J., vol. 17, pp. 4960, Dec. 2018.
- [5] Maoguo Gong, Jiaojiao Zhao, Jia Liu, Qiguang Miao, Licheng Jiao, "Change Detection in Synthetic Aperture Radar Images Based on Deep Neural Networks", IEEE Transactions on Neural Networks and Learning Systems, VOL. 27, NO. 1, January 2016, PP:125-138
- [6] Ning Lv, Chen Chen, Tie Qiu, Arun Kumar Sangaiah, Deep Learning and Superpixel FeatureExtraction Based on Contractive Auto encoder for Change Detection in SAR Images, IEEE Transactions on Industrial Informatics, VOL. 14, NO. 12, DECEMBER 2018, PP:553- 5538
- [7] R. Chandra and K. Dayal, "Cooperative neuro-evolution of elman recurrent networks for tropical cyclone wind-intensity prediction in the south pacific region," in Evolutionary Computation (CEC), 2015 IEEE Congress on. IEEE, 2015, pp. 1784–1791.
- [8] S. Hong, S. Kim, M. Joh, and S.-k. Song, "Globenet: Convolutional neural networks for typhoon eye tracking from remote sensing imagery," arXiv preprint arXiv:1708.03417, 2017.
- [9] S. J. Lim, S. J. Jang, J. Y. Lim, and J. H. Ko, ``Classification of snoring sound based on a recurrent neural network," Expert Syst. Appl., vol. 123, pp. 237 245, Jun. 2019.

[10] Yangyang Li, Cheng Peng, Yanqiao Chen, Licheng Jiao, Linhao Zhou, Ronghua Shang, "A Deep Learning Method for Change Detection in Synthetic Aperture Radar Images", IEEE Transactions on Geoscience and Remote Sensing, Vol. 57, No. 8, August 2019

APPENDIX

A. SOURCE CODE

```
import os
import importlib
import numpy as np
from datetime import datetime
import json
import keras
from keras import layers
from keras import models
from keras import Model
from keras import optimizers
from keras.models import load_model
from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
from keras import callbacks
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint
import matplotlib.pyplot as plt
import random
import cv2 as cv
model_dir = 'DataSet/DisasterModel'
train_dir = os.path.join(model_dir, 'train')
validation_dir = os.path.join(model_dir, 'validation')
test_dir = os.path.join(model_dir, 'test')
train_dir_c = os.path.join(train_dir, 'cyclone')
train_dir_w = os.path.join(train_dir, 'wildfire')
validation_dir_c = os.path.join(validation_dir, 'cyclone')
validation_dir_w = os.path.join(validation_dir, 'wildfire')
test_dir_c = os.path.join(test_dir, 'cyclone')
```

```

test_dir_w = os.path.join(test_dir, 'wildfire')
# print('Check if samples are of same size:\n\nTrain:')
print(len(os.listdir(train_dir_c)), len(os.listdir(train_dir_w)))
print('Validation:')
print(len(os.listdir(validation_dir_c)), len(os.listdir(validation_dir_w)))
print('Test:')
print(len(os.listdir(test_dir_c)), len(os.listdir(test_dir_w)))
model = None
model_path = 'best_model.h5'
model_history_path = 'history'
model_exists = os.path.exists(model_path)
history_exists = os.path.exists(model_history_path)
if model_exists:
    model = load_model(model_path)
else:
    model = models.Sequential()
    model.add(layers.Conv2D(32, (3, 3), activation = 'relu', input_shape= (180, 180,
3)))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(64, (3, 3), activation = 'relu'))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(128, (3, 3), activation= 'relu'))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(128, (3, 3), activation= 'relu'))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(256, (3, 3), activation = 'relu'))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Flatten())
    model.add(layers.Dropout(0.3))
    model.add(layers.Dense(512, activation = 'relu'))
    model.add(layers.Dense(2, activation = 'softmax'))

```

```

model.summary()

if not model_exists:
    model.compile(optimizer=optimizers.RMSprop(lr=1e-4),
                  loss='categorical_crossentropy', metrics=['acc'])
train_datagen = ImageDataGenerator(rescale= 1./255,
                                    rotation_range= 40,
                                    width_shift_range= .2,
                                    height_shift_range= .2,
                                    shear_range= .2,
                                    zoom_range= .2,
                                    horizontal_flip= True)

test_datagen = ImageDataGenerator(rescale= 1./255)
train_generator = train_datagen.flow_from_directory(train_dir,
                                                    target_size= (180, 180),
                                                    batch_size=32,
                                                    class_mode='categorical')

validation_generator = test_datagen.flow_from_directory(validation_dir,
                                                        target_size=(180, 180),
                                                        batch_size=32,
                                                        class_mode='categorical')

print("\n\nCheck a size of data/label batches:")
for data_batch, label_batch in train_generator:
    print('data batch size:', data_batch.shape)
    print('label batch size:', label_batch.shape)
    break

es = EarlyStopping(monitor='val_acc', mode='max', verbose=1, patience=6)

mc = ModelCheckpoint('best_model.h5', monitor='val_acc', mode='max',
                    verbose=1, save_best_only=True)

history = None

if history_exists:
    with open(model_history_path, 'r') as json_file:

```

```

        history = json.load(json_file)
else:
    history      =      model.fit_generator(train_generator,steps_per_epoch=150,
epochs=10,      validation_data=validation_generator,      validation_steps=50,
callbacks=[es, mc]) ##in callback one can put tensorboard_callback once lines 114
and 117 are run

model_dir = 'Dataset/DisasterModel'
test_dir = os.path.join(model_dir, 'test')
best_model = 'best_model.h5'
label_names = {0: 'cyclone', 1: 'wildfire'}
model = load_model(best_model)
test_datagen = ImageDataGenerator(rescale=1./255)
test_generator = test_datagen.flow_from_directory(test_dir, target_size=(180, 180),
batch_size=1)
print(len(test_generator))
loss_test, acc_test = model.evaluate(test_generator)
print("\nThe accuracy of the model is:", str(np.round(acc_test, 2)), '% for loss
value',str(np.round(loss_test, 2)), '%.')
def run_prediction():
    n = 331
    plt.figure(figsize=(15, 15))
    for i in range(9):
        plt.subplot(n)
        idx = random.randint(0, len(test_generator)-1)
        predict = model.predict(test_generator[idx][0])
        plt.imshow(test_generator[idx][0][0])
        plt.title(label_names[np.argmax(predict)])
        n += 1
run_prediction()
best_model = 'best_model.h5'
model = load_model(best_model)
sample_datagen = ImageDataGenerator(rescale = 1./255)

```

```

sample_dir = r'Dataset/DisasterModel'

sample_generator = sample_datagen.flow_from_directory(sample_dir,
target_size=(180, 180), batch_size=1)

def run_visualisation():

    i = random.randint(0, len(sample_generator) - 1)

    plt.figure(figsize=(10, 10))

    predict = model.predict(sample_generator[i][0])

    title = 'The below image should present the ' +
str(label_names[np.argmax(predict)]) + '\n'

    plt.title(title, fontsize=20)

    plt.imshow(sample_generator[i][0][0])

    plt.show()

    layer_outputs = [layer.output for layer in model.layers]

    activation_model = Model(inputs=model.input, outputs=layer_outputs)

    activations = activation_model.predict(sample_generator[i][0])


def display_activation(activations, col_size, row_size, act_index):

    activation = activations[act_index]

    activation_index=0

    fig, ax = plt.subplots(row_size, col_size, figsize=(15, 15))

    title1 = 'Images from the layer ' + str(act_index)

    fig.suptitle(title1, fontsize=16)

    for row in range(0, row_size):

        for col in range(0, col_size):

            ax[row][col].imshow(activation[0, :, :, activation_index], cmap='gray')

            activation_index += 1


display_activation(activations, 2, 2, 0) ## 4 images from the 1st layer

plt.show()

display_activation(activations, 2, 2, 1) ## 4 images from the 2nd layer

plt.show()

```

```
display_activation(activations, 2, 2, 3) ## 4 images from the 4th layer  
plt.show()  
display_activation(activations, 3, 3, 5) ## 9 images from the 6th layer  
plt.show()  
run_visualisation()
```