Automatic cyclone eye detection system using computer vision and deep learning model

A PROJECT REPORT

Submitted by

Moreddy Sankeerth Reddy
19MIS1086

in partial fulfillment for the award of the degree of

Master of Technology

in

Software Engineering (5 Year Integrated Programme)

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

Vellore Institute of Technology

Vandalur - Kelambakkam Road, Chennai - 600 127

November - 2023

VIT®
Vellore Institute of Technology

School of Computer Science and Engineering

DECLARATION

I hereby declare that the project entitled Your Automatic cyclone eye detection system using computer vision and deep learning model submitted by me to the School of Computer Science and Engineering, Vellore Institute of Technology, Chennai, 600 127, in partial fulfillment of the requirements of the award of the degree of Master of Technology in Software Engineering (5 year Integrated Programme) and as part of SWE3004 – Software Design and Development Project is a bona-fide record of the work carried out by me under the supervision of Prof. Dr MENAKA PUSHPA A . I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or University.

Place: Chennai

Date:

Signature of Candidate

VIT®

Vellore Institute of Technology

School of Computer Science and Engineering

CERTIFICATE

This is to certify that the report entitled Automatic cyclone eye detection system using computer vision and deep learning model is prepared and submitted by Moreddy Sankeerth Reddy (Reg. No. 19MIS1086) to Vellore Institute of Technology, Chennai, in partial fulfillment of the requirement for the award of the degree of Master of Technology in Software Engineering (5 year Integrated Programme) and as part of SWE3004 – Software Design and Development Project is a bona-fide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission.

Guide/Supervisor	HoD
Name:	Name: Dr. Nisha V M
Date:	Date:
Examiner	Examiner
Name:	Name:
Date:	Date:

(Seal of SCOPE)

Acknowledgement

I am obliged to give my appreciation to a number of people without whom I could not have completed this thesis successfully.

I would like to place on record my deep sense of gratitude and thanks to my internal guide Prof. DR. MENAKA PUSHPA A, School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, whose esteemed support and immense guidance encouraged me to complete the project successfully.

I would like to thank our HoD Dr. Nisha V M, School of Computer Science and Engineering (SCOPE) and Project Coordinator Dr.Kanchana Devi V, Vellore Institute of Technology, Chennai, for their valuable support and encouragement to take up and complete this thesis.

Special mention to our Dean Dr. Ganesan R, Associate Deans Dr.Parvathi R and Dr. S. Geetha, School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for motivating us in every aspect of software engineering.

I thank our management of Vellore Institute of Technology, Chennai, for permitting me to use the library and laboratory resources. I also thank all the faculty members for giving me the courage and the strength that I needed to complete my goal. This acknowledgment would be incomplete without expressing the whole hearted thanks to my family and friends who motivated me during the course of my work.

Moreddy Sankeerth Reddy

19MIS1086

Abstract

Detecting the cyclone eyes plays a vital role in predicting their tracks accurately, especially in tropical regions. The primary source of data for such predictions is satellite images. To teach computer systems to identify cyclone centers, we need a vast amount of data to train deep learning models. This project introduces a new technique called Image Augmentation-based Object Translation. Here, the 'object' refers to the cyclone itself. The method involves extracting the cyclone from images and relocating it to various positions within the same image. By carefully replacing the cyclone's original position, we significantly expand the dataset, providing more diverse examples for the computer to learn from.

The developed technique aims to augment the dataset phenomenally, enhancing the information available for training models. With this expanded dataset, we trained a sophisticated model called YOLO (You Only Look Once). YOLO is renowned for its exceptional object detection capabilities and is widely used today. Using this model, we achieved an impressive accuracy rate of above 89% in pinpointing the cyclone's eye. This accurate identification of the cyclone's eye signifies a crucial step forward in improving our ability to predict cyclone tracks, contributing to more reliable and precise weather forecasting techniques.

Contents

List of figures and tables

Chapter 1
Introduction

1.1 Overview

Predicting cyclone tracks accurately is pivotal for effective disaster preparedness, particularly in tropical regions. Satellite images serve as the primary data source for this prediction. However, teaching computers to identify cyclone centers from these images requires extensive and diverse datasets. This project introduces a pioneering technique termed Image Augmentation using Object Translation to address this need. By manipulating cyclone positions within the same image, this method significantly expands the dataset for training deep learning models. The objective is to bolster the information available for model training, facilitating more precise cyclone center identification. Leveraging the advanced capabilities of YOLO (You Only Look Once), a sophisticated object detection model, this approach attained an impressive accuracy rate of above 89% in identifying cyclone eyes. This breakthrough offers promising strides in enhancing cyclone track prediction accuracy, thereby fortifying weather forecasting methodologies for better disaster mitigation and response.

1.2 Background

1.2.1 History of Cyclones in India:

Tropical cyclones or cyclones in Indian and Australian context are called with different names throughout the world like Hurricane in Atlantic and east pacific oceans, typhoons in west pacific are very powerful storms characterized by a low-pressure center and strong rotating winds, typically accompanied by thunderstorms and heavy rainfall. These intense weather systems form over warm ocean waters and are classified by wind speeds, ranging from Category 1 (weakest) to Category 5 (strongest), with winds exceeding 119 km (about 73.94 mi) per hour. Cyclones bring significant dangers, including high-speed winds causing structural damage, uprooting trees, and leading to storm surges resulting in coastal flooding. According to a research paper **[17]** covering a span of 50 years from 1970 to 2019, India experienced a staggering 117 cyclones, leading to the tragic loss of over 40,000 lives. This study on extreme weather events also highlights a significant decline in the mortality rate caused by tropical cyclones over the past decade.

1.2.2 Structure of cyclone:

Cyclone or Hurricane Eye is Central calm region with clear skies. **Eyewall** is Surrounding the eye, hosts the most intense winds and rain**. Spiral Rain-bands** are Spiral arms with showers and gusty winds.

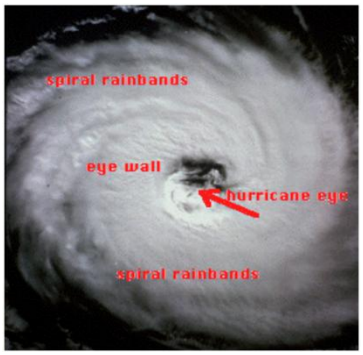


Figure 1.1 : Characteristics of cyclone

The accurate estimation of position of the circulation of TC (center or eye) is very crucial. Highly accurate positioning is especially important for short-range forecasts in critical situations such as near landfall. Many other cyclone parameters like intensity, radius of maximum wind, and asymmetries within the cyclone, are significantly affected by slight variations in the center position. The TC center determination is often done manually or semi-automated using the remote sensing data from the geostationary satellites or the weather radars.

1.2.3 Computer vision

Computer vision is like giving eyes to computers! It's a smart technology that helps computers understand and interpret images or videos, just like humans do with their eyes. Using special algorithms and techniques, computers can recognize objects, understand scenes, and even make decisions based on what they 'see.' It's used in various things like identifying faces in photos, helping cars 'see' the road, or detecting objects in videos. Computer vision helps machines understand the visual world around us, enabling them to assist in many useful tasks without human intervention.

1.2.4 Image augmentation using object translation in cyclone eye detection

Using this method in the context of cyclone eye detection is crucial for getting more data to teach computers how to spot cyclone centers accurately. By moving the cyclone to different spots in the same picture, we make lots of versions of that picture. This gives the computer many examples to learn from, making it better at recognizing cyclone eyes. With this wider variety of pictures, our computer models can understand all the different looks of cyclones, making them better at finding cyclone centers in satellite pictures. This helps us make more accurate predictions about cyclone paths and strengths.

1.2.5 Object detection

Detection is all about finding or spotting specific things within data, like identifying objects in images or locating patterns in information. It involves using smart algorithms or methods to pinpoint and recognize particular items or characteristics within a set of data. For instance, in computer vision, detection might involve finding faces in a photo or identifying specific objects, like cars or cyclone eyes, in an image or video. It's like a smart search that scans through data, locates what you're looking for, and points it out among everything else.

1.2.6 Bounding Box prediction:

Bounding box prediction is like drawing a rectangle around things in pictures to find and point them out. These rectangles, called bounding boxes, show where certain stuff is in the picture. The goal is to figure out where these boxes should go around the things, we're interested in. This means finding the right spot for the box (usually its top-left and bottom-right corners) and deciding how big it should be. This helps computers spot and understand things in images, like finding objects or following their movements, which is handy in computer vision tasks and machine learning.

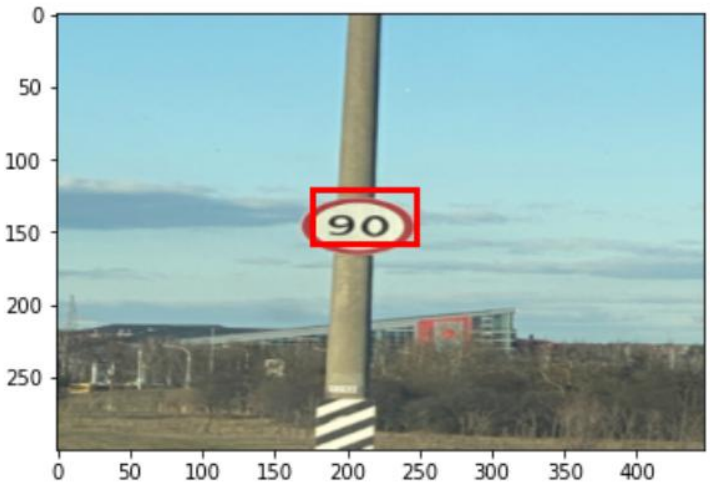


Figure 1.2 : Bounding prediction box example

1.2.7 YOLO - You Only Look Once

YOLO, short for "You Only Look Once," is a smart computer technique that quickly spots and identifies objects in images or videos. Unlike some other methods that might scan images multiple times, YOLO is super speedy. It glances at the picture just once and cleverly spots various things all at once. It does this by dividing the image into smaller areas and checking each one for things it knows, like cars, people, or even cyclone eyes. Once it finds these things, it outlines them with boxes and labels them. YOLO's strength lies in its speed and efficiency, making it great for tasks like spotting objects in real-time videos or images where quick detection matters a lot. In this project I am using YOLOv5 version.

1.2.8 Deep neural network in terms of YOLO

A deep neural network is a smart system inspired by the human brain's structure. It consists of many interconnected layers, each processing data and extracting increasingly complex features. These networks learn patterns from large sets of data, allowing them to recognize and categorize information.

In YOLO, the deep neural network serves as the backbone of the system. YOLO uses this network to analyze images by breaking them into smaller grids and processing each grid to detect objects. The layers in the network work together to identify various objects within these grids, enabling YOLO to swiftly and accurately pinpoint multiple objects in real-time, making it highly effective for object detection tasks.

1.2.9 YOLOv5

YOLOv5, which stands for "You Only Look Once version 5," is a smart computer system used to quickly and accurately identify different objects in images or videos. It's an improved version of previous YOLO models, designed to work even faster and better.

The architecture of YOLOv5 is based on a deep neural network. This network has various layers that process the image step-by-step to recognize objects. YOLOv5 divides the image into grids and analyzes each grid to spot different objects within them. It then outlines these objects with boxes and labels them.

There are different variants or versions of YOLOv5, like YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. Each variant has a different size and complexity, with 's' being the smallest and 'x' being the most advanced. These variants cater to various needs, providing options for speed, accuracy, and model size, depending on the task at hand.

1.2.10 YOLOv5 architecture

- YOLOv5 relies on a deep neural network structure that serves as its backbone.
- The architecture comprises numerous layers, each contributing to the identification of objects within images.
- One significant aspect involves breaking down the image into smaller grids, enabling detailed analysis within these grids.
- The network processes these grids systematically, leveraging various layers to identify and delineate different objects.
- Its design allows for the detection of multiple objects simultaneously within a single pass through the network.
- YOLOv5's architecture emphasizes speed and accuracy, balancing these aspects for efficient object recognition in real-time applications.

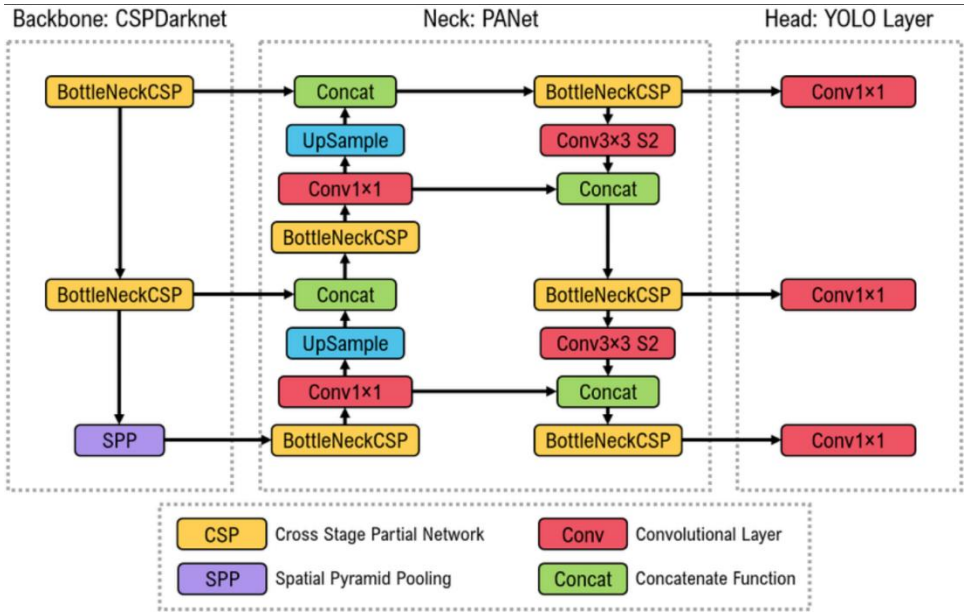


Figure 1.3 : The architecture of the YOLOv5 model, which consists of three parts: (i) Backbone: CSPDarknet, (ii) Neck: PANet, and (iii) Head: YOLO Layer. The data are initially input to CSPDarknet for feature extraction and subsequently fed to PANet for feature fusion. Lastly, the YOLO Layer outputs the object detection results (i.e., class, score, location, size).

1.2.11 YOLOv5 versions:

YOLOv5 model comes in different variants designated by letters such as 's', 'm', 'l', 'x' indicating the model size or complexity. Here's an overview of the different types:

YOLOv5s (Small):

Description: YOLOv5s is the smallest and fastest variant, optimized for speed and inference on resource-constrained devices. It sacrifices some accuracy for faster processing speed and lower memory consumption.

Use Case: Suitable for real-time applications or scenarios where computational resources are limited.

YOLOv5m (Medium):

Description: YOLOv5m strikes a balance between speed and accuracy. It offers a moderate increase in model complexity and accuracy compared to YOLOv5s while remaining relatively efficient in terms of computational resources.

Use Case: Ideal for general-purpose object detection tasks requiring a trade-off between speed and accuracy.

YOLOv5l (Large):

Description: YOLOv5l is a larger and more complex variant, prioritizing accuracy over speed. It includes more layers and parameters compared to YOLOv5m, leading to higher accuracy but with increased computational requirements.

Use Case: Well-suited for applications where high accuracy is critical, even if it means sacrificing some speed.

YOLOv5x (Extra Large):

Description: YOLOv5x is the most complex and computationally intensive variant, offering the highest accuracy among YOLOv5 models. It has a significantly larger number of layers and parameters, providing state-of-the-art performance in object detection.

Use Case: Primarily used in scenarios where utmost accuracy is paramount and computational resources are not a limiting factor.

Each variant in the YOLOv5 series offers a different trade-off between model complexity, inference speed, and accuracy. The selection of a specific variant depends on the specific requirements of the task, considering factors such as computational resources, speed, and the desired level of accuracy in object detection applications.

In this Project I used YOLOv5m model as the data-set size is relatively medium and we need faster training and detection.

1.2.12 Why do we need YOLO cyclone eye detection?

Detecting cyclone eyes using YOLO is crucial for accurately predicting cyclone paths, especially in tropical areas. The "eye" of a cyclone greatly influences its track, and finding this center accurately is super important for predicting where it'll go. YOLO helps spot these important cyclone centers in satellite pictures. This detection helps meteorologists and weather systems understand where the cyclone's core is, making their predictions much better. Better predictions mean that communities can get warnings early, so they can prepare for things like strong winds and heavy rain caused by cyclones. YOLO's ability to find cyclone eyes helps us make better predictions about where cyclones might go, aiming to reduce damage and keep people and homes safe in places that often get cyclones. This kind of early warning system helps communities get ready for disasters and keeps them safer when cyclones hit.

1.2.13 YOLO evaluation metrics:

- Intersection over Union (IoU): IoU is a measure that quantifies the overlap between a predicted bounding box and a ground truth bounding box. It plays a fundamental role in evaluating the accuracy of object localization. Class: This identifies the object category, such as "person," "car," or "dog." In this context “**cyclone eye**”

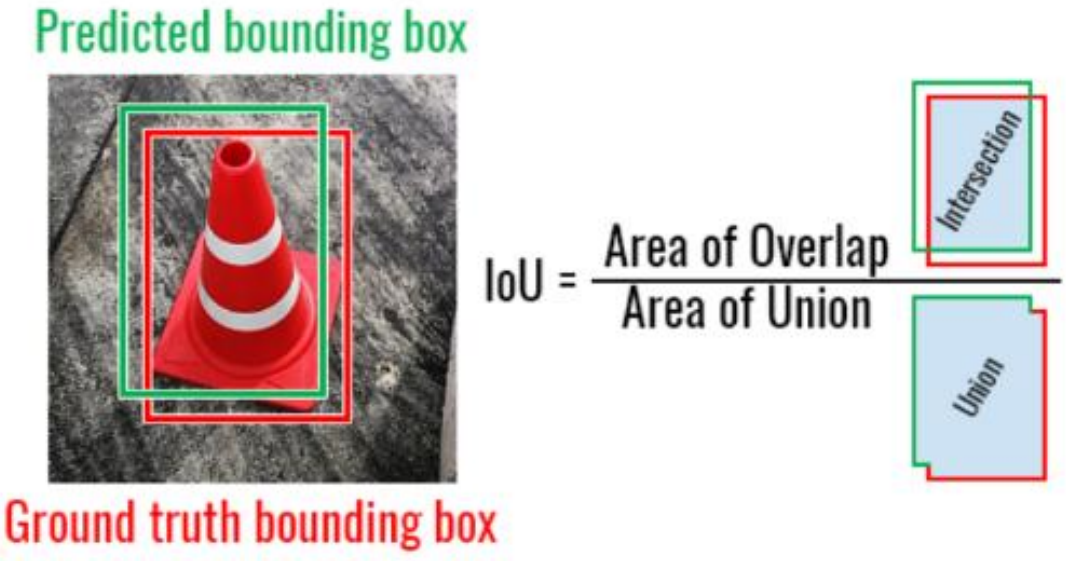


Figure 1.4 : IOU

- Images: Denotes the quantity of images within the validation set containing the specified object class.
- Instances: Represents the total count of appearances of the class across all validation set images.
- Box (P, R, mAP50, mAP50-95): These metrics offer insights into the model's object detection performance (TP: Truth Positive, FP: False Positive, FN: False Negative, TN: Truth Negative)
    - P (Precision): Reflects the accuracy of correctly detected objects.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\text{all detections}}$$

    - R (Recall): Measures the model's capability to identify all instances of objects in the images.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{all ground truths}}$$

- mAP50: Indicates the mean average precision at an intersection over union (IoU) threshold of 0.50, evaluating the model's accuracy based on "easy" detections.
- mAP50-95: Represents the average of mean average precision scores calculated at various IoU thresholds, ranging from 0.50 to 0.95. This metric offers a comprehensive assessment of the model's performance across different levels of detection difficulty.
- F1 Score: Useful when a balance between precision and recall is needed.

1.3 Statement

Detecting cyclone eyes using machine learning methods faces various hurdles, such as data quality concerns and imbalanced datasets. Overcoming these challenges is crucial to create reliable and understandable prediction models. The project aims to develop robust algorithms using biomarkers and other data to precisely identify cyclone eyes in satellite images. Ensuring these models work well with new data and making ethical decisions about cyclone tracking are key objectives. This approach holds promise in advancing weather forecasting accuracy and aiding in predicting cyclone trajectories effectively.

1.4 Motivation

Detecting cyclone eyes holds immense significance in weather forecasting and disaster management. This crucial feature serves as a pivotal indicator of a cyclone's intensity and behavior. By accurately identifying and tracking cyclone eyes in satellite images, we unlock the potential to comprehend cyclone dynamics, predict storm trajectories, and issue timely warnings. This capability empowers us to mitigate risks, safeguard lives, and minimize the impact of catastrophic events, making cyclone eye detection an indispensable tool in enhancing early warning systems and facilitating proactive measures to protect communities from the devastating effects of severe storms.

1.5 Challenges and solution for detecting

The data itself is the challenge for the detection model as we need huge data to train a detection model. In this project we are going to increase the dataset using Image augmentation using object translation basically a computer vision technique where we extract the cyclone with the help of pixel extraction and then paste it at another position while in-painting the original location of the cyclone like that we are going to generate 800 images apart from the original images. The image data increase is 800% and when we combine the original images, we generated 800 images with just 100 images so we can say it's a phenomenal increase in data. We are increasing the data as the YOLO, or any other deep learning model require more and more data means the more the data the more the accuracy with the help of these data we train the YOLOv5m model thus we get more accuracy.

1.6 Literature review

The literature review in the project on detecting cyclone eye using various techniques serves as the foundational background and introduction to existing research and studies related to the topic.
It typically covers :

- Methods they have used
- Results they have got
- How is it compared to the proposed system (ours)

Md. Nazmul Haque, A. A. M. Ashfaqul Adel & Kazi Saeed Alam **[1]** discusses various machine learning and deep learning techniques applied to predict cyclones using satellite images. It mentions achieving cyclone prediction accuracies between 86 to 95% and cyclone eye detection accuracy above 87%. The proposed methodology introduces a deep learning object detection system specifically targeting the detection of a cyclone's eye, indicating promising results that potentially surpass traditional and current methods.

Liling Zhao, Yifei Chen & Victor S. Sheng **[2]** presents a novel method for real-time typhoon eye detection in meteorological satellite cloud images using deep learning. It addresses the challenge of extracting accurate atmospheric information from satellite images. Their proposed method exhibits impressive performance, achieving high accuracy rates of 94.22% for positive samples, 99.43% for negative samples, and an overall average accuracy rate of 96.83%. Moreover, it demonstrates efficient processing, with an average detection time of 6 ms per sample, meeting real-time detection requirements. Comparative analysis against k-nearest neighbors (KNN) and support vector machine (SVM)

algorithms reveals the superiority of their approach, outperforming these conventional methods.

Kulwarun Warunsin and Orachat Chitsobhuk in **[3]** focuses on identifying typhoon eye locations by combining wind parameters from QuikSCAT satellite data with spiral cloud images. They proposed a novel heuristic search method to automatically detect typhoon eye locations, minimizing search time and effectively locating the eye within the region of interest (ROI). Their approach showcased a substantial 64.4% reduction in distance error compared to three reference methods.

Snehlata Shakya; Sanjeev Kumar; Mayank Goswami presents in **[4]** the methods involving interpolation and data augmentation to enhance the temporal resolution and diversify dataset characteristics for weather prediction using satellite images. It employs classical preprocessing approaches and evaluates three optical flow methods with various constraint optimization techniques and error estimates. The study utilizes enriched data for training a convolutional neural network, achieving a minimum accuracy of 90% in cyclone classification and 84% in locating cyclone vortex.

Gang Zheng; Jianguo Liu; Jingsong Yang; Xiaofeng Li **[5]** focuses on a unique technique for pinpointing tropical cyclone (TC) centers using top cloud motions in consecutive geostationary satellite images. Their methodology utilizes Gaofen-4 satellite data to derive pixel-wise top cloud motion data, allowing accurate TC center determination via a distinct principle. They propose a motion field decomposition technique to eliminate scene shift and TC migration in the motion data, followed by a direction-based index algorithm for TC center location. The outcomes show enhanced concentric motions after their decomposition, aligning well with cloud patterns and established meteorological agency data.

Akash Anil Valsangkar; Joy Merwin Monteiro **[6]** introduces a novel approach for analyzing cyclones and their temporal evolution, crucial in meteorology. The proposed method combines topological concepts for identifying cyclones with optical flow analysis for detailed tracking. Unlike prior methods, this approach relies on fewer parameters and integrates into an exploratory framework to identify coherent cyclone movements efficiently. Multiple case studies illustrate its effectiveness in tracking cyclones in both northern and southern hemispheres, showcasing its potential for robust cyclone identification and movement analysis.

Qing Xu; Guosheng Zhang **[7]** presented research showcases an automatic approach aimed at pinpointing the center of tropical cyclones (TCs) using RADARSAT-1 synthetic aperture radar (SAR) images from 2001 to 2007. This method's efficacy was assessed by comparing the TC center estimations derived from SAR images to the TC best track (BT) datasets provided by

esteemed organizations like the National Hurricane Center (NHC) and the Shanghai Typhoon Institute (STI). The study findings indicate a strong alignment between the SAR-estimated TC center positions and the BT data from both institutions, highlighting SAR's potential as a robust tool for investigating tropical cyclone morphology and dynamics.

Tangao Hu; Yiyue Wu **[8]** introduces a method aiming to automatically identify tropical cyclone (TC) centers using HY-2 and Quick Scatterometer wind vector products. It outlines a process involving the identification of high-wind speed and vortexlike zones from wind speed and direction maps, respectively. The proposed method then automatically determines the TC center from these zones. The study validates this TC center automatic determination (TCCAD) method using six representative TCs and seventeen TCs from 2013-2016 seasons. Results suggest that the TCCAD method closely matches human expert determinations in accuracy and displays smaller deviations, indicating greater efficiency and reliability. However, limitations arise due to scatterometer product quality and TC eye structure issues, affecting the method's performance. Despite these limitations, the TCCAD method offers a practical, independent, and objective approach to TC center identification.

F. Sun, M. Min, D. Qin **[9]** examines the utilization of GF-4 and H8 imagery for tracking Super Typhoon Nepartak in 2016. It employs classical TV-L1 optical flow (OF) algorithms and H8 cloud-top products to study the typhoon's inner-core dynamics. Results show GF-4's finer resolution provided better typhoon center positions than H8's coarser imagery. The OF method failed using H8 data due to resolution constraints. The study emphasizes the substantial impact of higher spatiotemporal resolution imagery, demonstrating the benefits in understanding typhoon dynamics and structural features compared to lower-resolution systems.

Isabella K. Lee, Ali Shamsoddini **[10]** focuses on using spaceborne synthetic aperture radar (SAR) data to extract and analyze hurricane eye morphology. They develop a mathematical morphology method utilizing SAR imagery to automatically extract hurricane eyes. Specifically, they employ skeleton pruning via discrete skeleton evolution (DSE) to preserve the global and local shapes of hurricane eyes while minimizing segmentation errors caused by speckle noise. The proposed approach showcases a high level of accuracy in reconstructing hurricane eyes compared to manual reference data derived from NOAA. This method's use of SAR data allows for comprehensive insights into hurricane dynamics, despite the limitations of conventional optical sensors in observing air-sea interactions during hurricanes.

Shaohui Jin; Xiaofeng Li **[11]** presents a novel approach to identify the center of a tropical cyclone (TC) using synthetic aperture radar (SAR) images

that often miss capturing the TC's eye. Their algorithm, comprising image processing techniques and knowledge about TC rain-band structures, involves three steps: detecting rain band curves using a Canny edge detector, filtering spiral curves akin to a TC rain-band model, and using particle swarm optimization to find the best match. Numerical experiments demonstrate the method's effectiveness in locating TC centers. It compares favorably with best track data, revealing accuracy. Furthermore, the study compares two models, favoring the inflow angle model for precise TC center identification.

Neeru Jaiswal; C. M. Kishtawal **[12]** presents a technique to locate the center position of Tropical Cyclones (TC) using infrared satellite images. It utilizes brightness temperature (BT) gradients to identify convergence points and determine the cyclone center. By computing variance and gradient fluxes of BT, intersecting lines are generated across the image, forming a density matrix storing accumulated score values. This technique was tested on approximately 1000 IR images of cyclones from 2009-2010. The automated process analyzed sequential IR images of cyclones (Phyan, Ward, Laila, and Phet) to determine their center positions, subsequently comparing their tracks with observed tracks from the Joint Typhoon Warning Centre (JTWC). The mean track errors for Phyan, Ward, Laila, and Phet were computed as 42, 82, 58, and 42.5 km, respectively, revealing the method's performance in tracking cyclones.

Shaohui Jin; Shuang Wang **[13]** provided abstract highlights the significance of spaceborne microwave synthetic aperture radar (SAR) for tropical cyclone monitoring, emphasizing the challenges in accurately locating cyclone centers when the eye isn't fully covered by SAR imagery. The paper proposes a semi-automatic method employing salient region detection and pattern matching. It introduces a salient region algorithm primarily detecting rain bands in SAR images and employs particle swarm optimization for pattern matching. The method's efficacy is evaluated by comparing results with NOAA's best track data, demonstrating commendable accuracy in locating cyclone centers from SAR images lacking a distinguishable eye signature.

N. Jaiswal and C. M. Kishtawal **[14]** discusses a novel approach for automatically determining tropical cyclone centers using infrared (IR) images from geostationary satellites. Analyzing Meteosat-5 IR images of specific cyclones, the method relies on image processing techniques to extract spiral features within the cyclone, estimating its center by fitting the spiral at different locations. While effective for well-defined spiral patterns, the method demonstrates limitations during the cyclone's formative or decaying phases, resulting in larger errors due to the absence of robust patterns. Despite these limitations, the method shows potential for automated cyclone center determination, presenting an alternative to the traditional manual approach.

Chinmoy Kar; Ashirvad Kumar **[15]** introduces a method to identify the region of interest within satellite images of tropical cyclones captured by Mateosat-7. It focuses on locating the center of gravity and measuring distances between significant points in the image. The approach utilizes iterative techniques, computing the center of gravity and average distances using Euclidean and Manhattan distance metrics. However, the methodology lacks an in-depth exploration of preprocessing techniques and object localization strategies commonly employed in computer vision for cyclone detection and tracking. The paper would benefit from incorporating established image processing methods and comparing its approach with existing algorithms used for similar tasks in cyclone image analysis.

Here's a tabulated classification of the types of research papers based on their focus and methodologies:

Table 1.1 : Literature review

Research Paper	**Methodology**	**Results and Focus**	**Comparisons to Proposed System**
Haque et al. [1]	Machine Learning & DL	Cyclone Prediction Accuracies (86-95%), Cyclone Eye Detection	Introducing a Deep Learning Object Detection System for Cyclone Eye Detection
Zhao et al. [2]	DL and Real-time Detection	High Accuracy (94.22%-99.43%), Real-time Detection	Efficient Processing & Comparative Analysis with KNN and SVM Methods
Warunsin & Chitsobhuk [3]	Combining Data from Satellites	Heuristic Search Method, 64.4% Reduction in Distance Error	Lacking Image Augmentation and Object Localization Methods, No Deep Learning Aspect
Shakya et al. [4]	Data Augmentation	Enhanced Temporal Resolution, CNN Training (90% Accuracy)	Basic Preprocessing Techniques, Comparative Study on Optical Flow Methods
Zheng et al. [5]	Cloud Motion Data Analysis	Enhanced Motion Field Decomposition, Direction-based Indexing	Unique Focus on Cloud Motions, No Object Detection Aspect, Limited Comparison to Existing Methods
Valsangkar & Monteiro [6]	Topological and Optical Flow	Efficient Framework for Cyclone Analysis	Lower Parameters, More Efficient Tracking, Focus on

			Movement Analysis
Xu & Zhang [7]	SAR Image Analysis	Accurate TC Center Estimations Compared to BT Datasets	Strong Alignment with BT Datasets, Effective Use of SAR for Morphology and Dynamics Study
Hu & Wu [8]	Wind Vector Products	Identifying High-Wind Speed & Vortexlike Zones	Efficient TC Center Identification, Limited by Product Quality and Eye Structure
Sun et al. [9]	Optical Flow Algorithms	Higher Resolution Imagery, Typhoon Center Positioning	Impact of Higher Spatiotemporal Resolution Imagery, Comparisons to Coarser Imagery Systems
Lee & Shamsoddini [10]	SAR Data Analysis	Morphology Extraction, High Accuracy in Eye Reconstruction	Utilizing SAR Data for Hurricane Eye Analysis, Limited by Optical Sensors
Jin & Li [11]	SAR Image Analysis	Rain-band Detection, Precise TC Center Identification	Efficient TC Center Determination, Comparative Analysis on Models
Jaiswal & Kishtawal [12]	IR Image Analysis	BT Gradients for Cyclone Center Identification	Automated Determination of TC Centers, Deviations in Formative or Decaying Phases
Jin & Wang [13]	SAR Image Analysis	Salient Region Detection, Pattern Matching	Semi-Automatic Cyclone Center Identification, Achieving Accuracy from SAR Images
Jaiswal & Kishtawal [14]	IR Image Analysis	Spiral Features for TC Center, Track Comparison	Automated TC Center Determination, Limitations in Certain Cyclone Phases
Kar & Kumar [15]	Image Processing & Metrics	Center of Gravity, Distance Measurements	Lacks In-depth Exploration of Preprocessing and Object Localization Strategies, Benefits from Established Methods in Cyclone Analysis

The proposed system focuses on combining data preprocessing, composite image generation, dataset creation, object detection using YOLO, and evaluation and validation. While the existing research papers have explored various methods and techniques related to cyclone detection and analysis using different datasets and methodologies, many lack aspects such as employing deep learning models, utilizing image augmentation techniques, and detailed object detection strategies like YOLO. Incorporating these elements from the proposed system could enhance the accuracy and precision of cyclone eye detection methods and lead to more comprehensive cyclone analysis.

Chapter 2
Planning and system Specification

2.1 System Planning

The system is divided into two phases. Phase 1 will discuss the computer vision technique that is used in the context and how it is used and phase 2 is all about the YOLO labeling and detection.

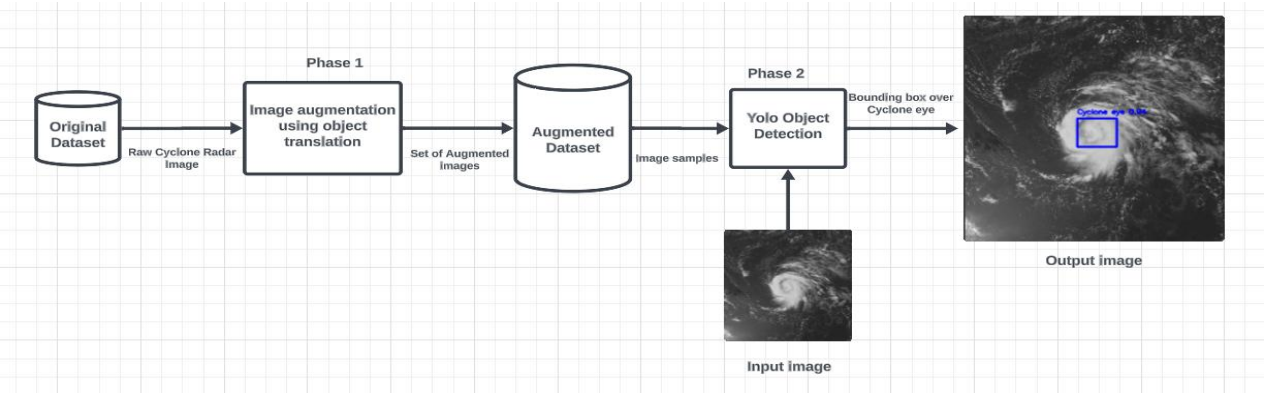


Figure 2.1 : System architecture

Table 2.1 : System planning

System planning	Dates
Literature review	Sept 29th 2023
Phase 1 implementation	Oct 10th 2023
Phase 2 implementation	Nov 10th 2023
Total implementation of project	Nov 24th 2023

2.2 Requirements

2.2.1 User Requirements

- Increase Dataset Size: Generate 800 images through image augmentation techniques from an initial set of images (100 images) Image size is 512x512 using object translation and in-painting techniques.
- Improve Accuracy: Enhance the accuracy of YOLOv5m model by training it on the augmented dataset.

2.2.2 Non-Functional Requirements

- Data Augmentation Quality: Ensure the quality and realism of the generated images through object translation and in-painting, maintaining the integrity and relevance of cyclone data in the images.
- Performance: Optimize the augmentation process to efficiently generate the increased dataset without significant time overhead.

2.3 System Requirements

2.3.1 Hardware Requirements

- Adequate computing power: Colab usually provides sufficient computing resources, including CPU, GPU, and memory, to handle deep learning tasks efficiently.
- Storage: Space to store the generated dataset. (Google drive) Adequate computing power: Colab usually provides sufficient computing resources, including CPU, GPU, and memory, to handle deep learning tasks efficiently.

2.3.2 Software Requirements

- Python Libraries: Utilize libraries such as OpenCV, NumPy, matlab plot ,Pandas for image processing, data handling, and augmentation.
- YOLOv5m Model: Implement, train, and evaluate the YOLOv5m model using the generated augmented dataset.
- Google Colab Environment: Leverage the Google Colab platform for its hosted Jupyter notebooks and access to GPU resources for training deep learning models efficiently.

Chapter 3
System Design

For this project we divided all the process into two phases and the first phase (Phase 1)is dataset generation which is dependent on computer vision techniques. In the first phase we have developed a method called image augmentation based object translation (Here object is cyclone) means extracting the object or cyclone and then translate it through the image at various different locations. And the second phase (Phase 2) is YOLO model for training deep learning model for detection of the cyclone eye.

3.1 Overall system architecture and methodology:

Data Preprocessing:

- Load a single original grayscale image (I), mask image (M), and a background image (B).
- Generate a binary mask (BM) by thresholding the original image.
- Find the largest connected component (LC) in the mask and then create a final mask.
- Define an offset for object placement on the background and overlay the object on the background at various positions.

Composite Image Generation:

- Produce composite images by overlaying the object on the background at multiple positions (e.g., top-left, top-right, etc.)

Data-set Creation:

- Combine the original image (I) with its associated composite images generated in step 2.
- Label the dataset: mark the areas containing the object (e.g., cyclone eye) within the composite images.

Object Detection and Model Training:

- Utilize methods like YOLO (You Only Look Once) for object detection.
- Train a YOLO model using the labeled dataset containing original and composite images.

Evaluation and Validation:

- Evaluate the trained YOLO model's performance on a separate test dataset.
- Validate the model's accuracy, precision, and recall for detecting the specified object (e.g., cyclone eye).

3.2 Image augmentation using object translation-Phase1

In this project we have derived a new method for image generation as YOLO or any other deep learning model requiring huge amount of data, we increased the dataset using a method called image augmentation using object translation. In this we are going to extract the object and translate it through different positions of the diagram. The positions for our experiments will be Bottom, Left, Top, Right, Top Left, Top Right, Bottom Left, Bottom Right and their positions in numbers are determined with the help of the size of the image because various images has various positions if we fixed the numbers for the object positions. This phase is the most crucial phase and backbone of the project. Below given in a picture what Phase 1 does.

Algorithm:

Step 1 : Load the original grayscale image (I), mask image (M), and background image (B).

Step 2 : Generate a binary mask (BM) by thresholding the original image:BM(x,y)={0 if I(x,y)>=T, 255 Otherwise} Where T is threshold (95 in our case) value.

Step 3 : Find the largest connected component (LC) in the mask: LC = Largest Connected Components (BM).

Step 4 : Define an offset (O) for object placement on the background: O=−150.

Step 5 : Iterate through the mask (M):

- For each pixel in M:
  - -Calculate the corresponding background pixel for object placement (BP) : BP=I(y,x+O) Where is the row and x is the shifted column .
- If BP is part of the object.
  - -Assign a random pixel value to BP within the background region: BP(y,x+O)=RandomPixelValue(40,80)

Step 6 : Place the object at different positions on the background: Iterate through positions.

Step 7 : Generate composite images by overlaying the object on the background at specified positions.

Step 8 : Save the resulting composite images associated with each position

3.2.1 Python libraries used

- **NumPy (np)**: NumPy is a Python library used for numerical computations, especially when dealing with arrays, matrices, and mathematical functions. In this context, NumPy is used for various array manipulations and numerical operations on image data.
- **Matplotlib.pyplot (plt):** Matplotlib is a popular plotting library in Python. The pyplot module within Matplotlib provides functions for creating figures, plotting images, and visualizing data. Here, it's used for displaying images and generating plots if needed.
- **PIL (Python Imaging Library) -** Image: PIL is a library used for opening, manipulating, and saving many different image file formats. The Image module is used here for loading and handling image data, such as opening grayscale images.
- **OpenCV (cv2):** OpenCV is a library used for computer vision and image processing tasks. The cv2 module provides functions for various image processing operations, including reading, writing, manipulating, and analyzing images. In this method, OpenCV functions are used for image processing tasks like pixel manipulation and thresholding.
- **IPython.display** - display: The display function from IPython.display is used to render and display images directly in the Jupyter/Colab environment. It's particularly useful for showing images within Colab notebooks.
- **google.colab.patches - cv2_imshow**: The cv2_imshow function from the google.colab.patches module is specifically designed for displaying OpenCV images directly in Google Colab notebooks. It's a patched version of cv2.imshow for Colab environments.

These libraries and modules are used for loading, processing, and manipulating image data. NumPy, PIL, and OpenCV are essential for handling image arrays, performing image processing operations (like thresholding, pixel manipulation), and generating composite images by overlaying objects on the background. Matplotlib and the display functions are used to visualize the processed images within the Colab environment for analysis and verification.

3.3 YOLOv5m - Phase 2

3.3.1 Libraries Used During YOLO

- **PyTorch (import torch)**:PyTorch is a popular deep learning framework used for building and training neural networks. Here, it's likely utilized for implementing YOLOv5 and handling the model training and inference.
- **utils:**This library seems to contain utility functions or methods used in the YOLOv5 repository for various tasks. These functions might include helper functions for initialization, data processing, or other custom utilities.
- **comet_ml:**Comet.ml is a platform for tracking, comparing, and optimizing experiments across machine learning workflows. It's used for experiment management, logging, and visualization of training metrics and results.
- **Training Process:** The train.py script is used for training the YOLOv5 model. Expected outputs may include training logs, such as loss values, metrics like precision and recall, and epoch-wise statistics.
    - Output files might include saved weights of the model after training (best.pt, etc.).
- **Cyclone eye Detection:**The detect.py script is likely performing object detection on an image using the trained model (best.pt).
    - Expected output includes the detected objects in the specified image along with their bounding boxes and confidence scores.
- **Validation:** The val.py script is used for validation/testing of the trained model.
    - Outputs will include evaluation metrics such as precision, recall, mAP (mean Average Precision), etc., computed on a separate validation dataset (custom.yaml)

3.3.2 YOLOv5m Implementation for Cyclone Eye Detection:

- Objective: The project aims to use YOLOv5m, a state-of-the-art object detection algorithm, specifically for detecting the cyclone eye in satellite images.
- Framework Selection: YOLOv5m, known for its accuracy and efficiency, is chosen due to its superior performance in real-time object detection tasks.
- Architecture Overview: YOLOv5m is a deep learning-based object detection architecture that employs a single-stage detector, facilitating rapid inference while maintaining accuracy.
- Model Customization: The pre-trained YOLOv5m model is fine-tuned and customized specifically for detecting cyclone eyes by retraining it on a dataset containing annotated cyclone eye images.
- Data Preparation: An initial dataset consisting of satellite images containing cyclones is annotated with bounding boxes around cyclone eyes to train the YOLOv5m model.

- Training Process: The model is trained iteratively using this annotated dataset to optimize its ability to accurately detect cyclone eyes within the images.
- Inference and Validation: Post-training, the model undergoes inference on unseen satellite images to detect and localize cyclone eyes. The detected eye regions are validated against ground truth annotations to assess the model's accuracy.
- Integration into the Project: YOLOv5m's detection capabilities are integrated into the larger project, contributing crucial insights for cyclone analysis and forecasting by identifying key features like the cyclone eye.
- Limitations and Further Improvements: While YOLOv5m provides robust cyclone eye detection, ongoing improvements in training data quality and augmentation techniques are continually explored to enhance the model's accuracy and generalization.
- This implementation of YOLOv5m within the project showcases its efficacy in detecting cyclone eyes, aiding in weather analysis and prediction by accurately identifying critical cyclone features.

Chapter 4
Implementation of the system

In this section I will mention the implementation of the system step by step in the form of the methodology and algorithm. As mentioned above I have divided the project in two phases. Phase 1 does the Image augmentation using object translation and with that we increase the dataset and in Phase 2 is YOLOv5 detection model where we label the images and then train it in 200 epochs with the batch size of 16 and then test it on 20 images.

4.1 Image augmentation using object translation - Phase 1

This method involves moving the cyclone within the same picture to different spots. By doing this, we create many versions of the cyclone image, all within the same frame. This trick helps diversify the dataset, giving the computer more examples to learn from. It's like showing it different angles of cyclones. This practice helps the computer get better at spotting cyclone centers accurately. So, when we train our models using these varied images, they become more accurate at finding cyclone eyes in satellite pictures, making cyclone predictions more reliable.

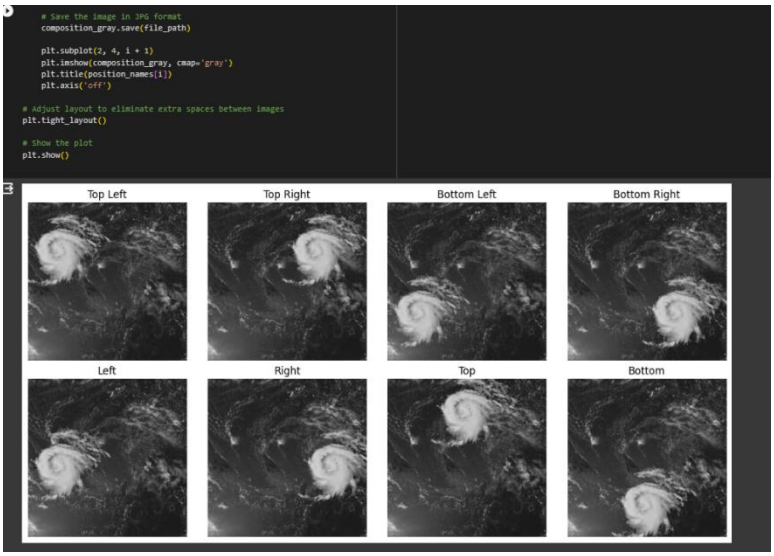


Figure 4.1 : Cyclone with different positions generation

4.2 YOLO training - Phase 2

YOLO labeling: I used makesense.ai as a labeling tool for YOLO format. For YOLO detection I have divided the dataset into 2 parts one is train and val dataset for the training purpose.

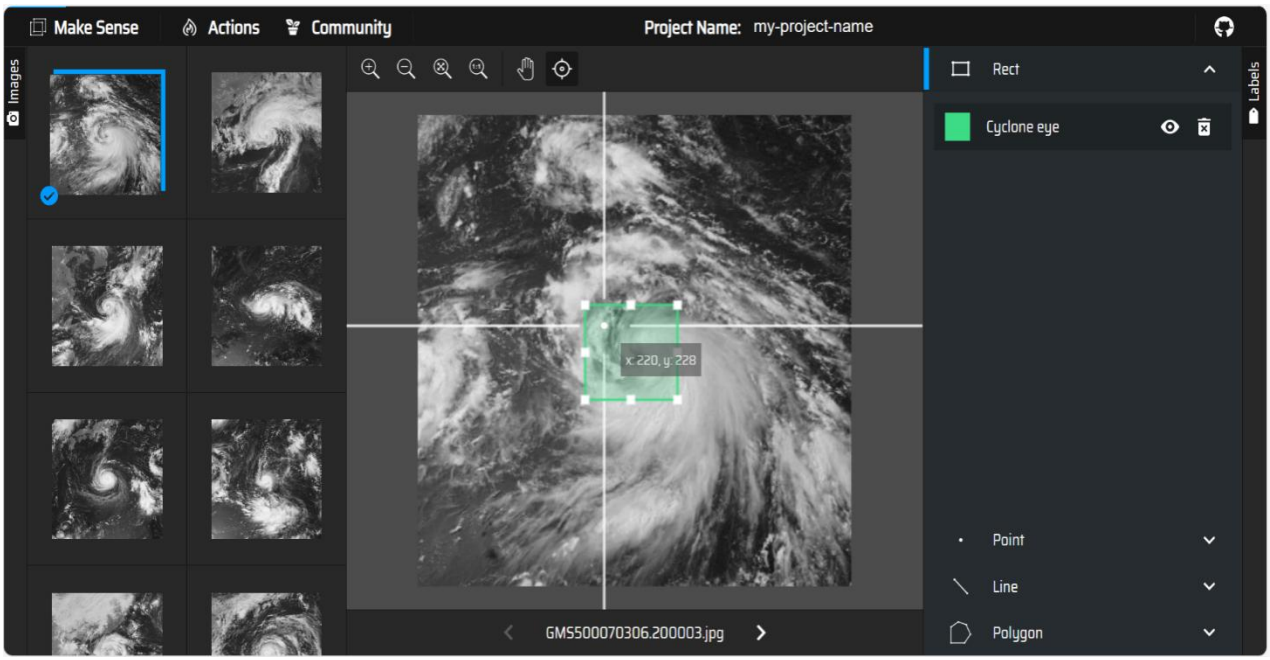


Figure 4.2: labelling

Then I imported these labels into the YOLO detection model and inserted the custom.yaml file in the data folder of yolo for detection.

```
!git clone https://github.com/ultralytics/yolov5  # clone
%cd yolov5
%pip install -qr requirements.txt comet_ml  # install

import torch
import utils
display = utils.notebook_init()  # checks
```

```
YOLOv5 🚀 v7.0-247-g3f02fde Python-3.10.12 torch-2.1.0+cu118 CUDA:0 (Tesla T4, 15102MiB)
Setup complete ✅ (2 CPUs, 12.7 GB RAM, 27.0/78.2 GB disk)
```

Figure 4.3: YOLO Loading

With this we have loaded the YOLOv5 model from ultralytics. and trained as below.

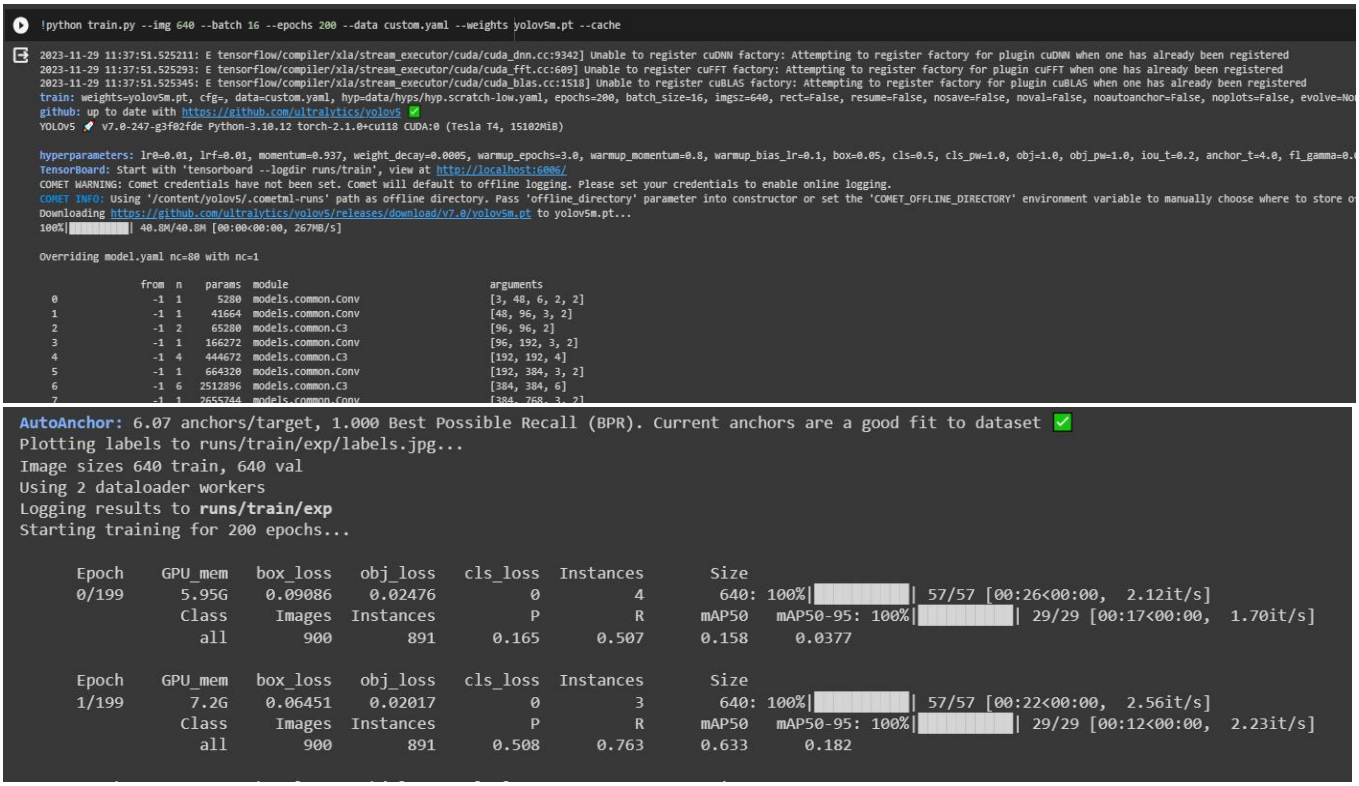


Figure 4.4: YOLO training

Like these 200 epochs were iterated for training and then tested on 20 images for validation. **As cyclone images have more complex patterns it needs at least 200 epochs to be trained.**

Chapter 5
Results and discussion

5.1 Image augmentation using object translation - Phase 1

In this phase we have used Image augmentation using object translation with that we have generated 8 new images with the same cyclone. In this phase we have used 100 images and generated 800 images. If we combine these 800 with 100 we get 900 images for training. The example is given below.

Input : Original image

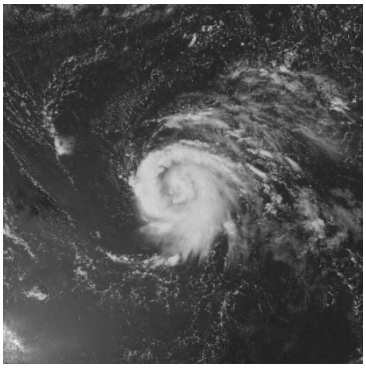


Figure 5.1 : Original image

Output :

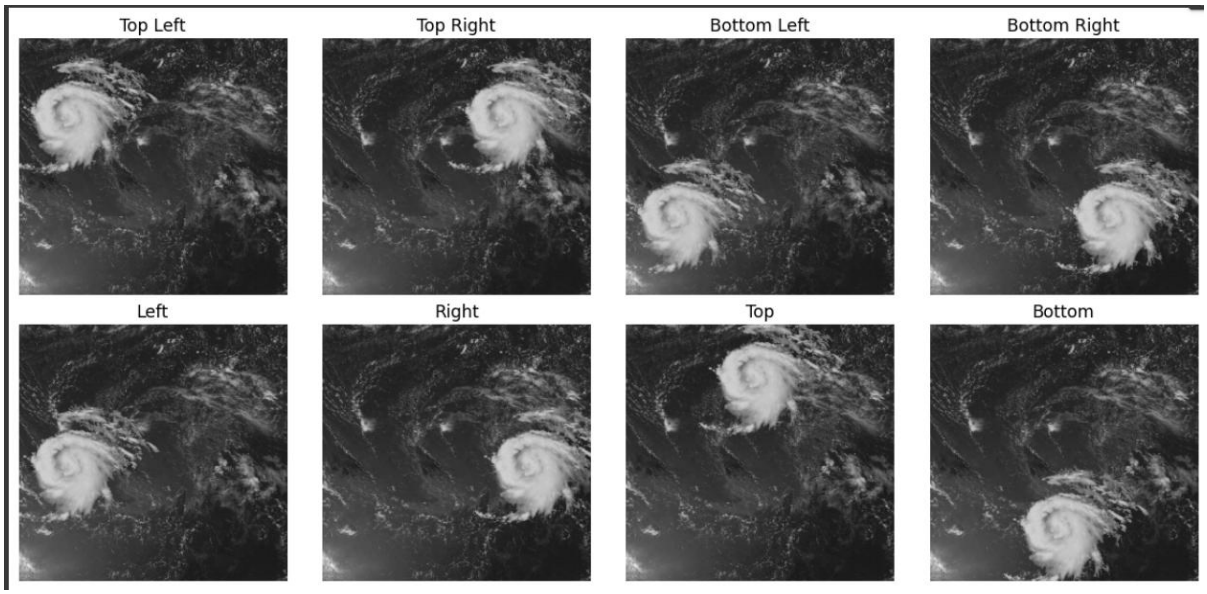


Figure 5.2 : Generated images

Table 5.1 : Dataset scale

Names	Count
Original images	100
Generated images per image	8 images per image
Generated images	100*8=800
Total images	100+800=900

With the data set of 100 I am able to generate 800 images extra for training of the YOLO model. Thus, increase in data is 800 % in total images.

Phase 2: In this phase we have used all the images for the YOLO model. In those 900 images we have used the images as below and new(100) images were added for testing purposes for evaluation.

Table 5.2 : Image usage division

Names	Images
Train	900-36=864
Val	36
Test	20(New and difficult)

The below image gives you the results of the testing on a single image and graphs against epochs of precision, recall, box loss, object loss and mAP etc with the code provided by ultralytics.

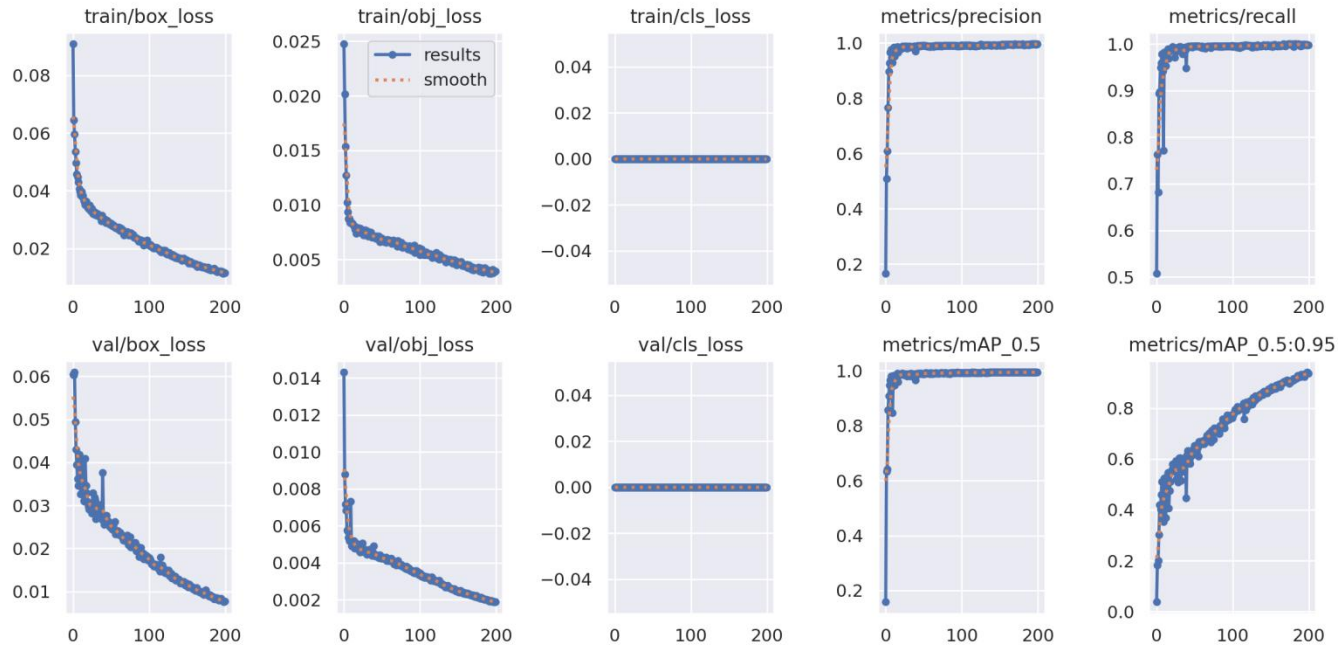


Figure 5.3: Evaluation on epochs according to YOLO metrics

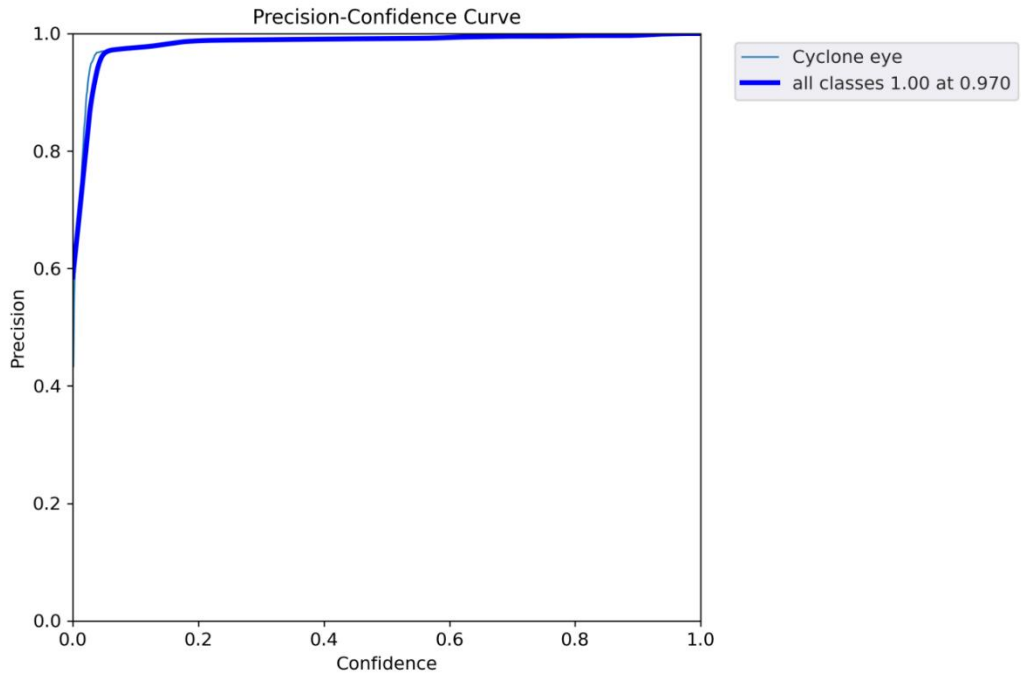


Figure 5.4 : Precision confidence curve

The precision-confidence curve showcases the relationship between the model's precision and the level of confidence it has in its predictions across epochs. As training progresses, this curve illustrates how precise the model's predictions are concerning the confidence it has in those predictions. A higher

precision-confidence curve indicates that the model's high-confidence predictions are more accurate, emphasizing instances where the model is confident and correct. Monitoring this curve over epochs helps understand the model's ability to make confident and accurate predictions as it learns from the training data.

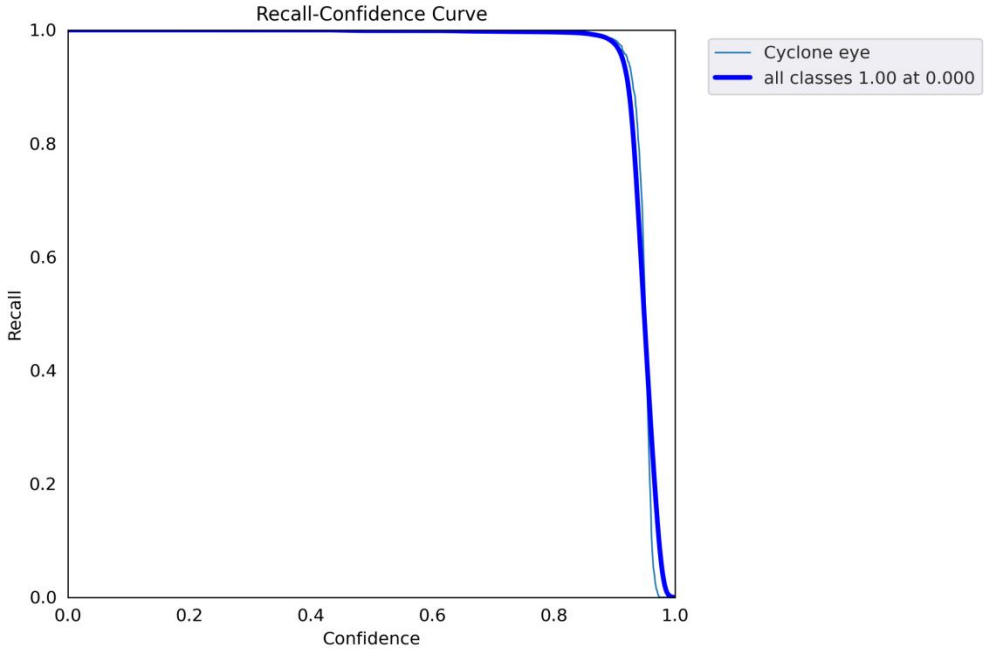


Figure 5.5: Recall confidence curve

The recall-confidence curve in YOLO training depicts the correlation between the model's recall and its confidence levels throughout training epochs. This curve reflects how well the model recalls the relevant instances in the dataset concerning its confidence in those instances. An ascending recall-confidence curve indicates that the model's higher-confidence predictions align with more relevant instances in the dataset. Tracking this curve throughout epochs provides insights into the model's capability to recall pertinent information confidently, assisting in evaluating its performance and progression during training.

The below is detection of cyclone eye on new image where the image is detected with 0.95 confidence.

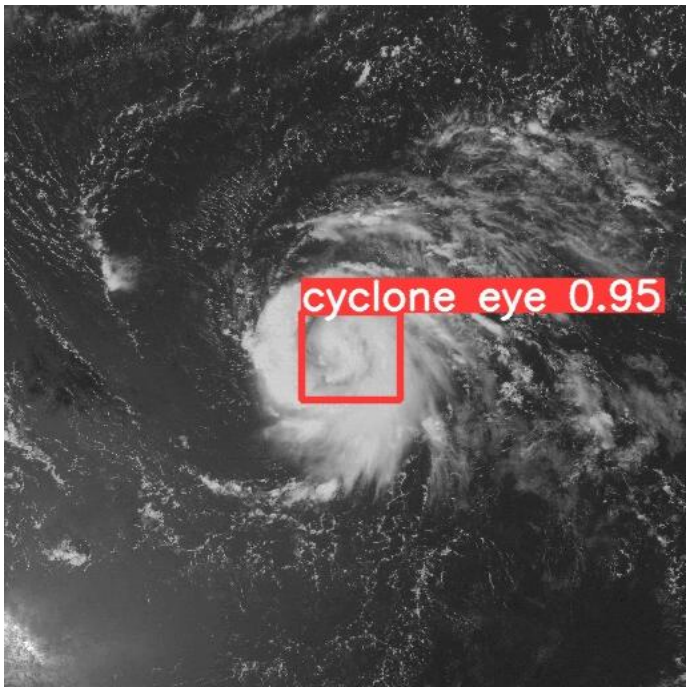


Figure 5.6: Detection on new image

Now we will see the evaluation for 20 images.

```
!python val.py --weight runs/train/exp/weights/best.pt --data custom.yaml --task test --name validation --augment

val: data=/content/yolov5/data/custom.yaml, weights=['runs/train/exp/weights/best.pt'], batch_size=32, imgsz=640, conf_thres=0.001, iou_thre
YOLOv5 🚀 v7.0-247-g3f02fde Python-3.10.12 torch-2.1.0+cu118 CUDA:0 (Tesla T4, 15102MiB)

Fusing layers...
Model summary: 212 layers, 20852934 parameters, 0 gradients, 47.9 GFLOPs
test: Scanning /content/drive/MyDrive/train_data/labels/test... 18 images, 2 backgrounds, 0 corrupt: 100% 20/20 [00:07<00:00,  2.63it/s]
test: New cache created: /content/drive/MyDrive/train_data/labels/test.cache
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 1/1 [00:01<00:00,  1.43s/it]
                   all         20         19      0.894      0.893      0.871      0.331
Speed: 0.2ms pre-process, 40.3ms inference, 7.5ms NMS per image at shape (32, 3, 640, 640)
```

Figure 5.7: Validation on 20 images (Testing)

Table 5.3 : Evaluation metrics on test data

Name of the metric for testing	Percentage or count
Images	20
Instances detected	19
Precision	0.894(89.4%)
Recall	0.893(89.3%)
mAP50	0.871(87.1)
MAP50-90	0.331(33.1)

Below are the images of the results showing the graphs on validation detection model of cyclone eye.

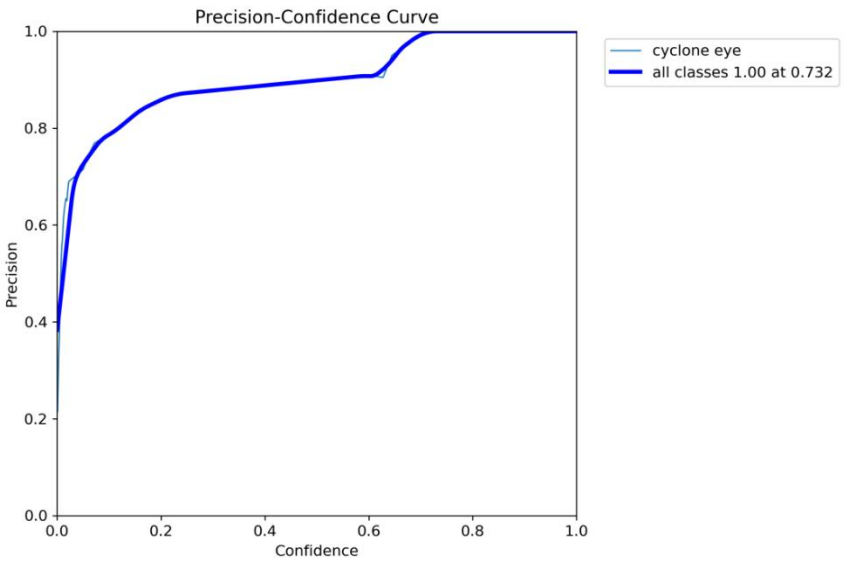


Figure 5.8: Precision confidence curve for validation

The precision-confidence curve signifies how precise the model's predictions are across different confidence thresholds. It demonstrates the relationship between precision (the accuracy of positive predictions) and the confidence levels assigned to those predictions. A higher precision-confidence curve implies that the model's confident predictions are more likely to be accurate. This curve helps understand the model's reliability at various confidence levels during validation, indicating its ability to make accurate predictions with different degrees of certainty.

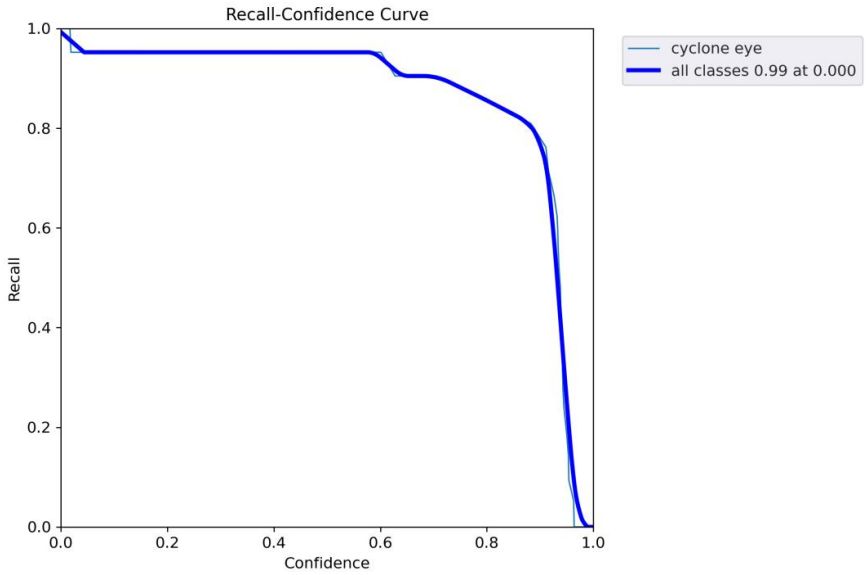


Figure 5.9 : Recall confidence curve validation

The recall-confidence curve in validation showcases the model's ability to recall positive instances in the dataset concerning its confidence levels. It displays the relationship between recall (the ratio of correctly predicted positive instances) and the confidence assigned to those predictions. A rising recall-confidence curve indicates that the model's higher-confidence predictions align better with relevant instances in the dataset. Tracking this curve during validation offers insights into the model's recall performance at different confidence thresholds, indicating how well it recalls relevant information confidently across varying degrees of certainty.

From these images we can infer that we will be able to detect the cyclone with 89% accuracy, which means even if the cyclone is in the forming stage the model can detect the cyclone with utmost accuracy.

Chapter 6
Conclusion and future work

In this project, I've utilized YOLO detection to accomplish the crucial task of cyclone eye identification. By employing computer vision techniques and leveraging YOLOv5, I've successfully detected cyclone eyes in satellite images, aiding in weather forecasting and disaster preparedness. The validation output, achieved with a configuration optimized for precision, showcased promising results with a high detection rate of 89.4%, ensuring reliable cyclone eye identification.

One of the main aspects of this project lies in the innovative computer vision technique introduced - the image augmentation using object translation method. This groundbreaking approach involves translating the cyclone within the same image while proficiently restoring the original location. This technique enhances the training dataset by creating diverse representations of cyclone eyes, augmenting data by simulating different cyclone positions. The method not only amplifies the dataset by 800%, a colossal leap in data size, but also enriches the model's understanding of cyclone features from varied perspectives, contributing significantly to the model's accuracy and robustness.

Overall, through the fusion of YOLO detection and the unique image augmentation technique developed, this project marks a significant stride in cyclone eye detection. The outcomes not only showcase a commendable detection rate but also pioneer a novel approach in computer vision, demonstrating the potential to revolutionize weather forecasting and disaster management through advanced image augmentation methodologies in cyclone analysis.

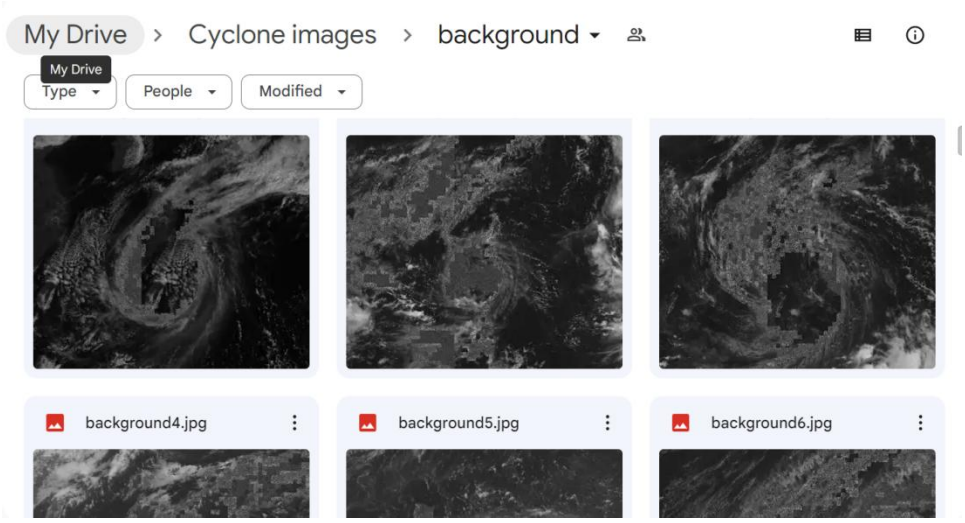
Moving forward, the developed cyclone eye detection system holds potential for advancing into cyclone tracking. By expanding on this groundwork, the system can be adapted to monitor and predict cyclone movement over time. This extension could prove instrumental in providing real-time insights into cyclone trajectories, aiding in early warning systems and disaster response. Moreover, this technology can serve as a foundation for ongoing projects, fostering continued research and development in enhancing cyclone analysis and weather forecasting capabilities.

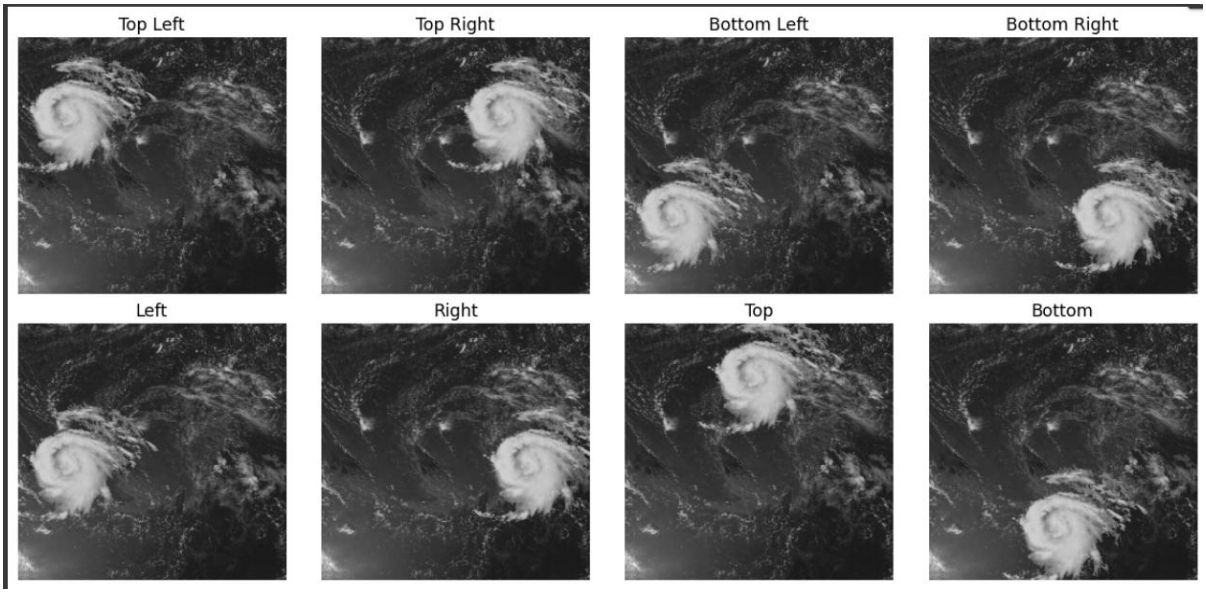
7. References

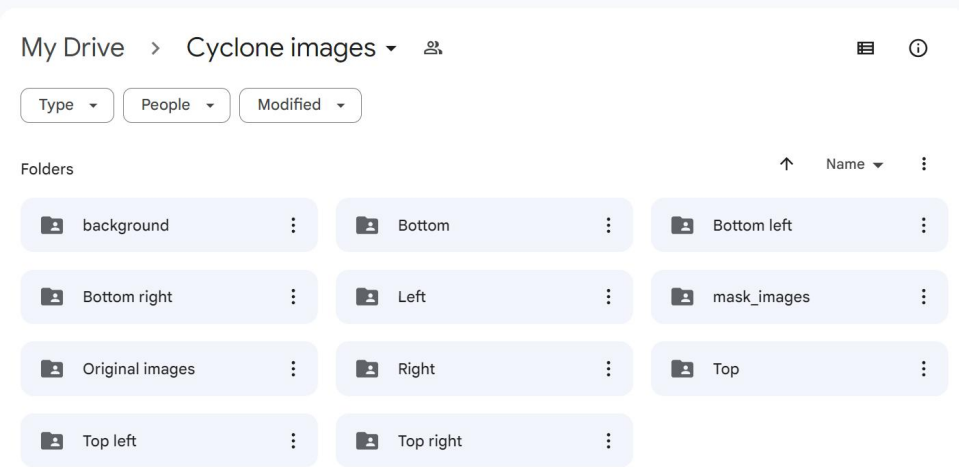
1. Haque, M.N., Ashfaqul Adel, A.A.M., Alam, K.S. (2022). Deep Learning Techniques in Cyclone Detection with Cyclone Eye Localization Based on Satellite Images. In: Arefin, M.S., Kaiser, M.S., Bandyopadhyay, A., Ahad, M.A.R., Ray, K. (eds) Proceedings of the International Conference on Big Data, IoT, and Machine Learning. Lecture Notes on Data Engineering and Communications Technologies, vol 95. Springer, Singapore. https://doi.org/10.1007/978-981-16-6636-0_35
2. Zhao, L., Chen, Y. & Sheng, V.S. A real-time typhoon eye detection method based on deep learning for meteorological information forensics. J Real-Time Image Proc 17, 95–102 (2020). https://doi.org/10.1007/s11554-019-00899-2
3. Warunsin, Kulwarun & Chitsobhuk, Orachat. (2015). Heuristic search on statistics of wind data and cloud images for automatic typhoon eye location. Proceedings of the 2015-7th International Conference on Knowledge and Smart Technology, KST 2015. 60-64. 10.1109/KST.2015.7051451.
4. S. Shakya, S. Kumar and M. Goswami, "Deep Learning Algorithm for Satellite Imaging Based Cyclone Detection," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 13, pp. 827-839, 2020, doi: 10.1109/JSTARS.2020.2970253.
5. G. Zheng, J. Liu, J. Yang and X. Li, "Automatically Locate Tropical Cyclone Centers Using Top Cloud Motion Data Derived From Geostationary Satellite Images," in IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 12, pp. 10175-10190, Dec. 2019, doi: 10.1109/TGRS.2019.2931795.
6. A. A. Valsangkar, J. M. Monteiro, V. Narayanan, I. Hotz and V. Natarajan, "An Exploratory Framework for Cyclone Identification and Tracking," in IEEE Transactions on Visualization and Computer Graphics, vol. 25, no. 3, pp. 1460-1473, 1 March 2019, doi: 10.1109/TVCG.2018.2810068.
7. Q. Xu, G. Zhang, X. Li and Y. Cheng, "An automatic method for tropical cyclone center determination from SAR," 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, China, 2016, pp. 2250-2252, doi: 10.1109/IGARSS.2016.7729581.
8. T. Hu, Y. Wu, G. Zheng, D. Zhang, Y. Zhang and Y. Li, "Tropical Cyclone Center Automatic Determination Model Based on HY-2 and QuikSCAT Wind Vector Products," in IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 2, pp. 709-721, Feb. 2019, doi: 10.1109/TGRS.2018.2859819.
9. F. Sun, M. Min, D. Qin, F. Wang and J. Hu, "Refined Typhoon Geometric Center Derived From a High Spatiotemporal Resolution Geostationary Satellite Imaging System," in IEEE Geoscience and Remote Sensing Letters, vol. 16, no. 4, pp. 499-503, April 2019, doi: 10.1109/LGRS.2018.2876895.
10. Lee, I.K., Shamsoddini, A., Li, X., Trinder, J.C., Li, Z. (2017). Extracting Hurricane Eye Morphology from Spaceborne SAR Images Using Morphological Analysis. In: Li, X. (eds) Hurricane Monitoring With Spaceborne Synthetic Aperture Radar. Springer Natural Hazards. Springer, Singapore. https://doi.org/10.1007/978-981-10-2893-9_7

11. S. Jin, X. Li, X. Yang, J. A. Zhang and D. Shen, "Identification of Tropical Cyclone Centers in SAR Imagery Based on Template Matching and Particle Swarm Optimization Algorithms," in IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 1, pp. 598-608, Jan. 2019, doi: 10.1109/TGRS.2018.2863259.
12. N. Jaiswal and C. M. Kishtawal, "Objective Detection of Center of Tropical Cyclone in Remotely Sensed Infrared Images," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 6, no. 2, pp. 1031-1035, April 2013, doi: 10.1109/JSTARS.2012.2215016.
13. S. Jin, S. Wang, X. Li, L. Jiao, J. A. Zhang and D. Shen, "A Salient Region Detection and Pattern Matching-Based Algorithm for Center Detection of a Partially Covered Tropical Cyclone in a SAR Image," in IEEE Transactions on Geoscience and Remote Sensing, vol. 55, no. 1, pp. 280-291, Jan. 2017, doi: 10.1109/TGRS.2016.2605766.
14. N. Jaiswal and C. M. Kishtawal, "Automatic Determination of Center of Tropical Cyclone in Satellite-Generated IR Images," in IEEE Geoscience and Remote Sensing Letters, vol. 8, no. 3, pp. 460-463, May 2011, doi: 10.1109/LGRS.2010.2085418.
15. C. Kar, A. Kumar, D. Konar and S. Banerjee, "Automatic Region of Interest Detection of Tropical Cyclone Image by Center of Gravity and Distance Metrics," 2019 Fifth International Conference on Image Information Processing (ICIIP), Shimla, India, 2019, pp. 141-145, doi: 10.1109/ICIIP47207.2019.8985860.
16. Katsamenis, Iason & Karolou, Eleni & Davradou, Agapi & Protopapadakis, Eftychios & Doulamis, Anastasios & Doulamis, Nikolaos & Kalogeras, Dimitris. (2022). TraCon: A novel dataset for real-time traffic cones detection using deep learning. 10.48550/arXiv.2205.11830.
17. An assessment of long-term changes in mortalities due to extreme weather events in India: A study of 50 years' data, 1970–2019,Weather and Climate Extremes,Volume 3

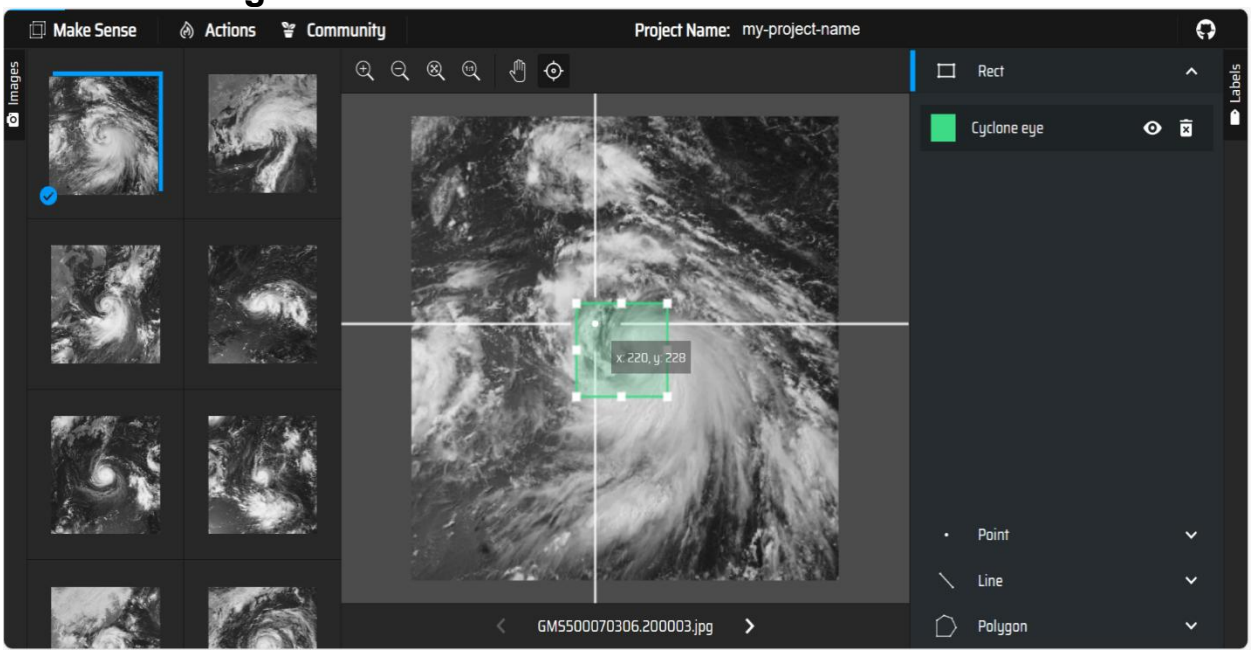
Appendix


My Drive
Cyclone images
background
My Drive
Type
People
Modified
background4.jpg
background5.jpg
background6.jpg


Top Left
Top Right
Bottom Left
Bottom Right
Left
Right
Top
Bottom



#Yolo labelling – makescence.ai



#Yolo training and evaluation