Multi-Class Object Detection in Satellite Imagery

Under guidance of

Mr. Arulraj.M
Group Head, BG&WSG
(Scientist/Engineer-SG)
NRSC, ISRO

By

P.V.V.S.ADITYA (120EC0012)
K.MOURYA (120EC0023)

Department of Electronics and Communication Engineering
Indian Institute of Information Technology,
Design and Manufacturing,
Kurnool
(MAY 2023-JULY 2023)

Abstract

Multi-class object detection in satellite imagery is a challenging task due to the large size and complexity of the images, as well as the diversity of objects that can be present. However, deep learning techniques have shown great promise for multi-class object detection in satellite imagery. One of the most popular deep learning object detection algorithms is YOLO (You Only Look Once). YOLO divides the image into a grid of cells, and each cell predicts bounding boxes and class probabilities for the objects that fall within its boundaries. This approach allows YOLO to detect objects in real time; making it a good choice for applications such as monitoring deforestation, tracking illegal mining, and detecting ships at sea. This approach allows Faster R-CNN to detect multiple objects in an image, each with a different class label. Other deep learning object detection algorithms that have been used for multi-class object detection in satellite imagery include Mask R-CNN, YOLOv3, and SSD. These algorithms have achieved high accuracy on benchmark datasets, and they are being used for a variety of applications.

As deep learning techniques continue to improve, multi-class object detection in satellite imagery is becoming increasingly accurate and reliable. This is leading to new applications for satellite imagery, such as disaster response, environmental monitoring, and urban planning.

Human visual system is very fast that can easily detect and identify the objects using this visual system it can easily identify multiple objects and detect the obstacle. This time there are very large amount of data, faster Gpu and good algorithm that we can easily train computers to detect and classify multiple objects with the high accuracy. In this we use an algorithm to detect and identify the objects that is **"YOU ONLY LOOK ONCE" (YOLO)** algorithm. This project is based on deep learning approach to solve the problem of object detection and recognition.

Acknowledgment

We would like to express our sincere gratitude to all the members of National Remote Sensing centre (ISRO), particularly **Mr. Arulraj. M**, Group Head, BG&WSG (Scientist/Engineer-SG), NRSC, ISRO, our **Mentor and Project guide** and the entire team for their invaluable guidance and unwavering support throughout the completion of this project. Their expertise and assistance have been instrumental in our success.

We would also like to extend our sincere gratitude to **Dr. K. Krishna Naik,** Associate Professor and HOD, Department of Electronics and Communication Engineering, and **Dr. K .V. Eswaramoorthy**, Assistant Professor, Placement Cell and Training Officer, for granting us the permission to undertake this project. Their valuable suggestions and inputs have significantly contributed to the efficiency and organization of our work.

Additionally, we would like to acknowledge the data sources that have greatly enriched our research. We are grateful for providing the necessary datasets and resources, which have been vital in the development and evaluation of our object detection model.

The contributions of all these individuals and organizations have played a pivotal role in the successful completion of this project, and we extend our heartfelt appreciation for their invaluable assistance and support.

Declaration

We declare that this project report titled **"Multi-Class Object Detection in Satellite Imagery"**, submitted in partial fulfilment of the degree of B. Tech in Electronics and Communication Engineering is a record of original work which is carried out by us under the guidance **Mr. Arulraj. M** Group Head, BG&WSG (Scientist/Engineer-SG) **NRSC, ISRO.** In keeping with the ethical practice in reporting scientific information, due acknowledgements have been made wherever the findings of others have been cited.

All Confidential Information means any data or information like scientific or technical information, invention, design, process, procedure, formula, improvement, technology or method is a proprietary of NRSC /ISRO which were used to by the undersign for the partial fulfilment of the degree of B. Tech in Electronics and Communication Engineering, Indian Institute of Information Technology, Design and Manufacturing, Kurnool in which will not be disclosed to any industry/public without any confirmation from NRSC/ISRO.

P.V.V.S.ADITYA (120EC0012)
K.MOURYA (120EC0023)
B.Tech, Final year, ECE department
IIIT DM Kurnool

Table of Contents

1. Literature Survey

This survey will draw on a variety of sources, including journal articles, conference papers, and books. The survey will be comprehensive and up-to-date, and it will provide a valuable resource for researchers who are interested in multi-class object detection in satellite imagery.

- "**YOLO: Real-Time Object Detection**" by Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi (2016):
    - Introduces the YOLO (You Only Look Once) framework for real-time object detection.
    - Proposes a single neural network that simultaneously predicts bounding boxes and class probabilities for multiple objects in an image.
    - Achieves real-time object detection with competitive accuracy, making it suitable for applications that require fast and efficient detection on various platforms.

- "**YOLOv2: YOLO9000: Better, Faster, Stronger**" by Joseph Redmon and Ali Farhadi (2017):
    - Improved version of YOLO that introduces anchor boxes and multi-scale training for better localization accuracy.
    - Utilizes Darknet-19, a 19-layer architecture, for feature extraction.
    - Achieves state-of-the-art performance on various object detection datasets.

- "**YOLOv3: An Incremental Improvement**" by Joseph Redmon, Wei Liu, and Christian Szegedy (2018).
    - This paper introduces YOLOv3, an improved version of the YOLO (You Only Look Once) object detection model.
    - It presents architectural changes and optimizations that enhance the accuracy and speed of object detection.
    - The authors demonstrate the effectiveness of YOLOv3 on various benchmark datasets, including COCO and PASCAL VOC.

- "**YOLOv4: Optimal Speed and Accuracy for Object Detection**" by Alexey Bochkovskiy, Chien-Yao Wang, and Liang-Chieh Chen (2020).
    - This paper introduces YOLOv4, which is an improved version of YOLOv3 that is able to achieve optimal speed and accuracy for object detection
    - It presents novel architectural improvements, including CSPDarknet53 backbone and PANet feature fusion. The authors evaluate YOLOv4 on multiple datasets and demonstrate its superior performance compared to previous models.

- "**Scaled-YOLOv4: Scaling Cross Stage Partial Network**" by Chien-Yao Wang et al. (2020):
    - Presents Scaled-YOLOv4, an improved version of YOLOv4 with a focus on scaling the network architecture.
    - Proposes a Cross-Stage Partial Network (CSPNet) that optimizes feature reuse and improves computational efficiency.
    - Demonstrates superior performance in terms of accuracy and speed, especially when applied to larger models.

- "**You Only Look Twice: Rapid Multi-Scale Object Detection In Satellite Imagery**" by Adam Van Etten
    - The proposed method, called YOLT, utilizes a two-step approach to detect objects at multiple scales in a computationally efficient manner.
    - In the first step, the method uses a coarse-grained object detector to identify candidate object regions in the image.
    - In the second step, a fine-grained object detector is applied to refine the candidate regions and classify the objects with higher accuracy.

- "**Swin-Transformer-Enabled YOLOv5 with Attention Mechanism for Small Object Detection on Satellite Images**" by Hong gong (2022):
    - SPH-YOLOv5 is an improved YOLOv5 model for satellite image object detection.
    - SPH-YOLOv5 incorporates new features, Swin Transformer Prediction Heads (SPHs), and Normalization-based Attention Modules (NAMs) for better performance.

- "**Experimental evaluation shows improved mean Average Precision (mAP) on satellite image datasets compared to baseline YOLOv5**" by Muhammed Enes , Zaide DURAN (2022):
    - Deep learning algorithms, such as YOLOv2 and YOLOv3, are widely used for object detection in aerial or terrestrial images.
    - This study utilized the DOTA dataset and trained YOLOv2 and YOLOv3 algorithms in the Google Colaboratory cloud service using Python.
    - YOLOv2 outperformed YOLOv3 in 5 out of 9 classes, while YOLOv3 showed better performance in detecting small objects. YOLOv3 also achieved faster detection time, averaging 2.5 seconds per image compared to YOLOv2's 43 seconds.

- "**Small Ship Detection on Optical Satellite Imagery with YOLO and YOLT**" by Wilder Nina,William Condori, Vicente Machaca, Juan Villegas & Eveling Castro (2020):
    - YOLO (You Only Look Once) and YOLT (You Only Look Twice) are deep learning algorithms used for object detection in satellite imagery.
    - The study compared the performance of YOLO and YOLT in detecting small objects (ships) in optical satellite imagery using the HRSC and MSDS datasets.
    - YOLT achieved superior results in detecting small objects, with an Average Precision (AP) of 76.06% in the MSDS dataset, while YOLO achieved 75% AP in the HRSC dataset, which consisted of objects of various sizes.

- **"YOLO-S: A Lightweight and Accurate YOLO-like Network for Small Target Detection in Aerial Imagery"** by Alessandro Betti and Mauro Tucci (2023):
    - YOLO-S is a simple, fast, and efficient network designed for small target detection in mobile or edge applications.
    - Evaluation on AIRES and VEDAI datasets demonstrates that YOLO-S outperforms baselines, achieving a 15% higher accuracy (mAP) than YOLOv3 on the VEDAI dataset while being 25-50% faster than YOLOv3 and only 15-25% slower than Tiny-YOLOv3.

2. Introduction

Object detection is a computer vision task that involves identifying and locating objects in an image or video. This is a challenging task, as objects can be of different shapes, sizes, and appearances and they can be located anywhere in the image.

Object detection in satellite imagery is a rapidly evolving field with the potential to revolutionize the way we use satellite imagery. Object detection in satellite imagery plays a crucial role in a wide range of applications, including urban planning, environmental monitoring, disaster management, and military surveillance. The ability to automatically identify and localize various objects in satellite images, such as buildings, roads, vehicles, and vegetation, can provide valuable insights and support decision-making processes.

Object detection in satellite imagery is a challenging task due to the large size and complexity of the images, as well as the diversity of objects that can be present. However, deep learning techniques have shown great promise for object detection in satellite imagery. Deep learning algorithms are able to learn the features of objects from large datasets of labelled images. This allows them to identify objects in new images with high accuracy.

There are a number of different deep learning algorithms that can be used for object detection in satellite imagery. Object detection in satellite imagery has seen significant advancements with state-of-the-art (SOTA) deep learning models. The journey started with R-CNN, followed by Fast R-CNN and Faster R-CNN, which improved speed and accuracy. SSD and YOLO introduced single-shot detection approaches, with YOLO achieving real-time performance by dividing the image into a grid. YOLOv2, YOLOv3, and YOLOv4 further improved accuracy and handling of multi-class detection. The latest model, YOLOv8, integrates advanced features such as anchor-free detection and focal loss.

2.1. Aim

- To develop an advanced multi-class object detection model specifically designed for satellite imagery using a state-of-the-art (SOTA) deep learning model.
- Enable accurate and real-time detection and localization of diverse objects, including Airplane, Car, Ship, Cross Road, Stadium, Railway Station, and Storage Tank in satellite images.
- Leverage the strengths SOTA deep learning model to attain real-time performance and multi-class detection capabilities, to enhance satellite image analysis.

2.2. Objectives

- Review the most popular deep learning algorithms that can be used for this task and select the best model.
- Collect and preprocess a comprehensive dataset of labeled satellite images with annotations indicating the presence and location of different objects.
- Implement and train the best chosen model for object detection in satellite imagery using the prepared dataset.
- Evaluate the performance of the developed system using various metrics, including precision, recall, and mean Average Precision (mAP), for each object class individually and across all classes.
- Provide a scalable and efficient solution for object detection in satellite imagery, capable of handling large-scale datasets and real-time inference.

2.3. Motivation

- Overcome the limitations of traditional methods, such as manual inspection and rule-based approaches, by automating the object detection process in satellite imagery.
- Exploit the potential of deep learning and the advancements in object detection algorithms to enhance the analysis of satellite imagery, improving decision-making processes.
- Address the requirement for accurate and efficient object detection in satellite imagery across diverse applications.

3. Theoretical Background

3.1. Object Detection

3.1.1. Introduction

Object detection is a computer vision task that involves identifying and localizing multiple objects within an image. It goes beyond object recognition, which focuses solely on recognizing objects without providing their precise location. Object detection combines object recognition and object localization to provide a comprehensive understanding of the objects present in an image.

3.1.2. Object Recognition

Object recognition refers to the process of identifying objects within an image and assigning them to predefined categories or classes. It involves training a machine learning model, typically a deep neural network, on a labeled dataset to learn the visual features and patterns associated with different objects. The model learns to classify regions of an image into various object classes, enabling it to recognize objects based on their visual characteristics.

3.1.3. Object Localization

Object localization is the process of precisely determining the location of detected objects within an image. It involves identifying the object's spatial extent by predicting the coordinates of a bounding box that tightly encloses the object. The bounding box typically consists of the coordinates for the top-left and bottom-right corners or the center coordinates along with width and height values. Object localization allows for precise positioning and spatial understanding of the detected objects.

3.1.4. Object Detection Process

The object detection process can be summarized into the following steps:

Input Image Acquisition: Obtain an image or a series of images as input data.

Pre-processing: Resize the input image to a fixed size suitable for the object detection algorithm. Normalize the pixel values to ensure consistent processing.

Feature Extraction: Extract meaningful features using deep learning architectures like CNNs which captures different levels of abstraction from low-level edges to high-level object features.

Object Recognition: Classify extracted features into predefined object classes using models trained on labeled data.

Object Localization: Predict bounding box coordinates enclosing detected objects using regression models or anchor-based methods.

Post-processing: Apply post-processing techniques, such as non-maximum suppression (NMS), to eliminate overlapping bounding boxes and filter out low-confidence detections.

Visualization and Output: Draw bounding boxes around detected objects, providing final output with bounding box coordinates, class labels, and confidence scores.

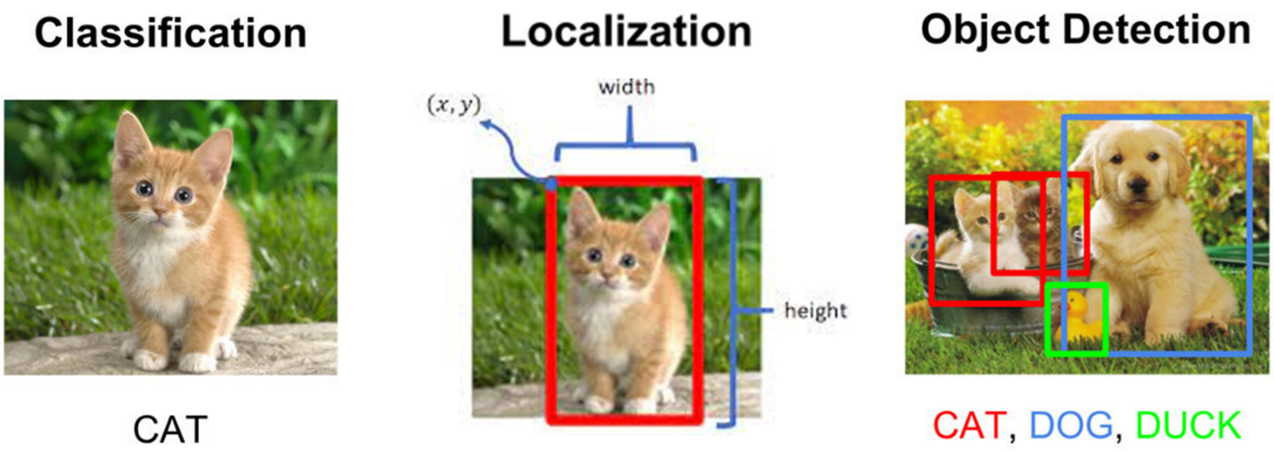


Fig.1 Object Detection Process

By combining object recognition and object localization, object detection provides a more comprehensive understanding of the objects present in an image. It enables a wide range of applications, including autonomous driving, surveillance, object tracking, and scene understanding

3.2. Deep Learning

Deep learning is a type of machine learning that uses artificial neural networks to learn from data. Artificial neural networks are inspired by the human brain, and they are able to learn to perform tasks by analysing large amounts of data.

Deep learning has been shown to be very effective for a variety of tasks, including:

- **Image recognition:** Deep learning can be used to recognize objects in images. This is a challenging task, as objects can be of different shapes, sizes, and appearances. However, deep learning algorithms have been shown to be very effective at this task.

- **Natural language processing:** Deep learning can be used to understand natural language. This is a challenging task, as natural language is often ambiguous and can be interpreted in many different ways. However, deep learning algorithms have been shown to be very effective at this task.

- **Speech recognition:** Deep learning can be used to recognize speech. This is a challenging task, as speech can be noisy and difficult to understand. However, deep learning algorithms have been shown to be very effective at this task.

Deep learning algorithms are typically trained on large datasets of labelled data. Labelled data is data that has been annotated with the correct answers. For example, if you want to train a deep learning algorithm to recognize objects in images, you would need to provide the algorithm with a dataset of images that have been labelled with the objects that they contain. Once a deep learning algorithm has been trained, it can be used to make predictions on new data. For example, if you have trained a deep learning algorithm to recognize objects in images, you could use the algorithm to predict the objects that are contained in a new image.

Here are some of the advantages of deep learning:

Accuracy: Deep learning algorithms can be very accurate, especially when they are trained on large datasets of labelled data.

Scalability: Deep learning algorithms can be scaled to large datasets, which allows them to be used to solve problems that involve a large amount of data.

Robustness: Deep learning algorithms can be robust to noise and other disturbances, which makes them suitable for use in real-world applications.

Overall, deep learning is a powerful tool that can be used to solve a wide variety of problems. However, it is important to be aware of the challenges associated with deep learning before using it.

3.3. Neural Networks

In the most basic definition, a neural network is a simulation of an animal (human) nervous system based on artificial neurons. Artificial neurons are supposed to imitate the function of a biological neuron; one or more inputs are being summed up and compared to a "firing" threshold that generates an output. The term "to fire" is used in biology and refers to a biological model of neural activity. Neural networks excel at complex tasks such as language recognition, computer translation or image recognition and what makes their use even more interesting, they are considered to be simply understood and constructed, as the fundamental mathematics behind it is relatively simple.

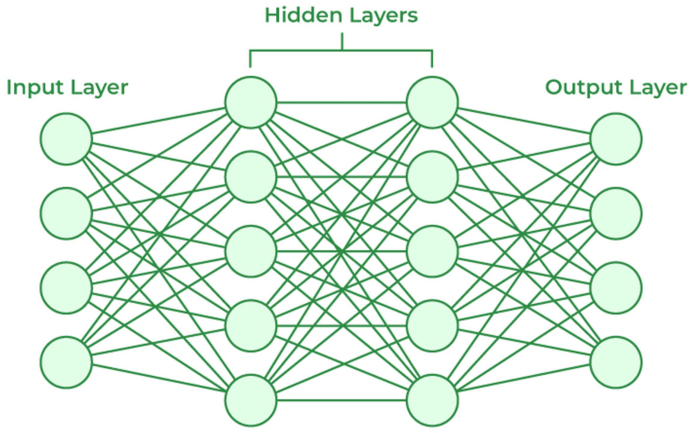


Fig.2 Multi-Layer Perceptron

An artificial neuron is, in a simplified sense, a series of inputs. These inputs are weighted and then the products of inputs and weights are summed. Then a threshold comparison decides the output. A single neuron with several inputs and one output is called a "perceptron".

Neural networks can be used for regression, although, depending on the complexity of the output, it might be excessive. Especially, if the mathematical output function is simple, linear or multiple linear regression might be a better choice, although these tend to easily overshoot functions.

3.4. Convolutional Neural Network (CNN)

A convolutional neural network (CNN/ConvNet) is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics convolution is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other. ConvNet is to reduce the images into a form that is easier to process, without losing features that are critical for getting a good prediction.

CNNs are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and output an activation value. When you input an image in a ConvNet, each layer generates several activation functions that are passed on to the next layer.

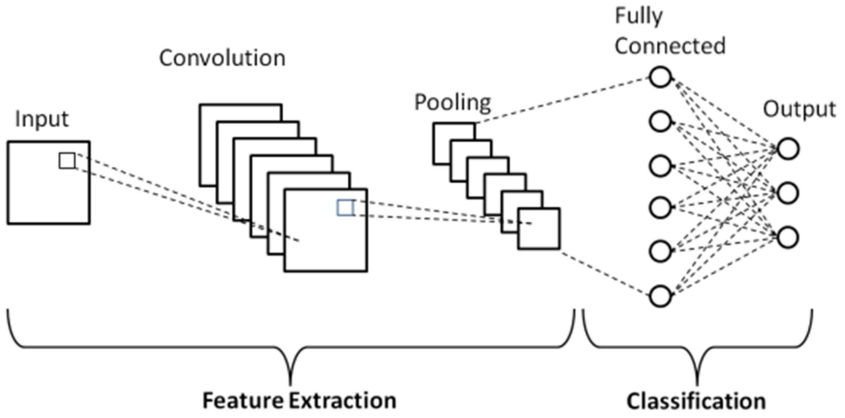


Fig.3.Convolution Neural Network

The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.

Based on the activation map of the final convolution layer, the classification layer outputs a set of confidence scores (values between 0 and 1) that specify how likely the image is to belong to a "class." For instance, if you have a ConvNet that detects cats, dogs, and horses, the output of the final layer is the possibility that the input image contains any of those animals. Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data by reducing the dimensions. There are two types of pooling average pooling and max pooling. I've only had experience with Max Pooling so far I haven't faced any difficulties.

3.5. Object Detection Models

Before deep learning took off in 2013, almost all object detection was done through classical machine learning techniques. Common ones included viola-jones object detection technique, scale-invariant feature transforms (SIFT), and histogram of oriented gradients.

These would detect a number of common features across the image, and classify their clusters using logistic regression, color histograms, or random forests. Today's deep learning-based techniques vastly outperform these.

Deep learning-based approaches use neural network architectures like RetinaNet, YOLO (You Only Look Once), CenterNet, SSD (Single Shot Multibox detector), Region proposals (R-CNN, Fast-RCNN, Faster RCNN, Cascade R-CNN) for feature detection of the object, and then identification into labels.

3.5.1. R-CNN Model Family:

- **R-CNN**: This utilizes a selective search method to locate RoIs in the input images and uses a DCN (Deep Convolutional Neural Network)-based region wise classifier to classify the RoIs independently.
- **SPPNet and Fast R-CNN:** This is an improved version of R-CNN that deals with the extraction of the RoIs from the feature maps. This was found to be much faster than the conventional R-CNN architecture.
- **Faster R-CNN:** This is an improved version of Fast R-CNN that was trained end to end by introducing RPN (region proposal network). An RPN is a network utilized in generating RoIs by regressing the anchor boxes. Hence, the anchor boxes are then used in the object detection task.
- **Mask R-CNN**: adds a mask prediction branch on the Faster R-CNN, which can detect objects and predict their masks at the same time.
- **R-FCN** : replaces the fully connected layers with the position-sensitive score maps for better detecting objects.

3.5.2. CenterNet Family:

- **SSD** places anchor boxes densely over an input image and uses features from different convolutional layers to regress and classify the anchor boxes.
- **DSSD** introduces a deconvolution module into SSD to combine low level and high-level features. While R-SSD uses pooling and deconvolution operations in different feature layers to combine low-level and high-level features.
- **RON** proposes a reverse connection and an objectness prior to extracting multiscale features effectively.
- **RefineDet** refines the locations and sizes of the anchor boxes for two times, which inherits the merits of both one-stage and two-stage approaches.
- **CornerNet** is another keypoint-based approach, which directly detects an object using a pair of corners. Although CornerNet achieves high performance, it still has more room to improve.

- **CenterNet** explores the visual patterns within each bounding box. For detecting an object, this uses a triplet, rather than a pair, of keypoints. CenterNet evaluates objects as single points by predicting the x and y coordinate of the object's center and it's area of coverage (width and height). It is a unique technique that has proven to out-perform variants like the SSD and R-CNN family.

3.5.3. YOLO (You Only Look Once) Family:

- **YOLO** uses fewer anchor boxes (divide the input image into an S × S grid) to do regression and classification. This was built using darknet neural networks.
- **YOLOv2 (YOLO 9000)** improves the performance by using more anchor boxes and a new bounding box regression method.
- **YOLOv3** is an enhanced version of the v2 variant with a deeper feature detector network and minor representational changes. YOLOv3 has relatively speedy inference times with it taking roughly 30ms per inference.
- **YOLOv4** works by breaking the object detection task into two pieces, regression to identify object positioning via bounding boxes and classification to determine the object's class.
- **YOLOv5** is an improved version of YOLOv4 with a mosaic augmentation technique for increasing the general performance of YOLOv4.
- **YOLOv6** uses a new backbone network called CSPDarknet53, which is more efficient and accurate than the previous Darknet53 network. It introduces a new method for generating the anchor boxes, called "dense anchor boxes." This method allows YOLOv6 to better detect objects of different sizes.
- **YOLOv7** incorporates new techniques, such as PANet and FPN, which improve the accuracy and speed of object detection. PANet is a technique that allows it to better learn the relationships between different objects in an image. FPN is a technique that allows it to better use the features from different layers of the backbone network.

- **YOLOv8** is the latest version of YOLO which uses a new backbone network called CSPDarknet63, which is even more efficient and accurate than the previous CSPDarknet53 network. It incorporates new techniques, such as CSAM and PANet, which further improve the accuracy and speed of object detection. CSAM is a technique that allows it to better learn the spatial relationships between different objects in an image.

After conducting a thorough background analysis and literature survey, we have **selected** the **YOLOv8** model as the most suitable model for the detection of objects in satellite images. This model was chosen because it is highly accurate, efficient, and scalable. It is also able to handle objects that are at different scales and that are located in cluttered scenes.

3.6. YOLOV8

3.6.1. Introduction

YOLOv8 is the latest version of the YOLO object detection and image segmentation model developed by Ultralytics. YOLOv8 is a cutting-edge, state-of-the-art (SOTA) model that builds upon the success of previous YOLO versions and introduces new features and improvements to further boost performance and flexibility. These include a new backbone network, a new anchor-free detection head, and a new loss function. YOLOv8 is also highly efficient and can be run on a variety of hardware platforms, from CPUs to GPUs.

The YOLOv8 architecture is able to achieve state-of-the-art accuracy and speed. It is able to detect objects in real time and it is able to detect objects of arbitrary sizes. YOLOv8, such as improved accuracy, enhanced speed, multiple backbones, adaptive training, advanced data augmentation, customizable architecture, and pre-trained models.

3.6.2. Architecture

The YOLOv8 architecture consists of three main parts:

- **Backbone:** The backbone is the main body of the network. It is responsible for extracting features from the input image. The CSPDarknet53 backbone is a deep convolutional neural network that is able to extract features at different scales.
- **Neck:** The neck is a connecting layer between the backbone and the head. It is responsible for fusing the features extracted from the backbone and for resizing the features to the desired resolution.
- **Head:** The head is responsible for generating the final output of the network. It predicts the bounding boxes, confidence scores, and class labels for the objects in the image.

Here are the key features of the YOLOv8 architecture:

- **CSPDarknet53 backbone:** It uses a CNN to extract features from an image and then predicts the bounding boxes and class labels of objects in the image. CSPDarknet53 is a hybrid network that combines the advantages of the CSPNet and Darknet53 networks.
- **FPN (Feature Pyramid Network):** The FPN (Feature Pyramid Network) is used to improve the accuracy of object detection at different scales.
- **GIoU (Generalized Intersection over Union) loss function:** The GIoU (Generalized Intersection over Union) loss function is used to improve the accuracy of object detection.
- **Anchor-free detection head:** It allows detecting objects of arbitrary sizes. It directly predicts the center of an object instead of the offset from a known anchor box. It reduces the number of box predictions, which speeds up Non-Maximum Suppression (NMS), a complicated post processing step that sifts through candidate detections after inference.

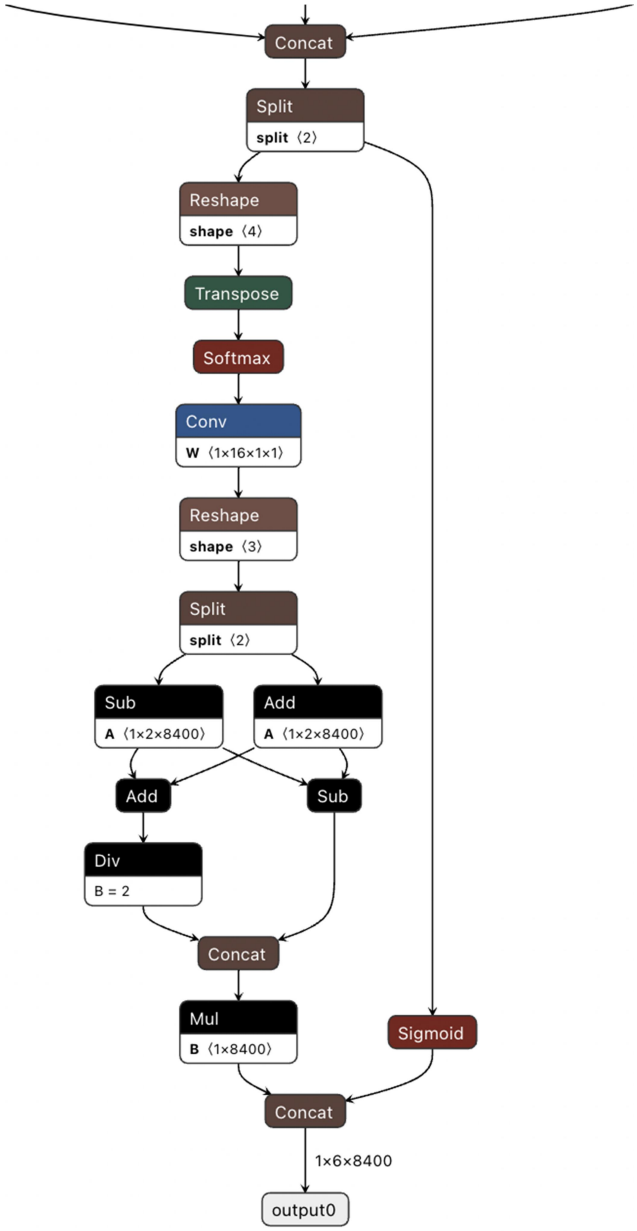
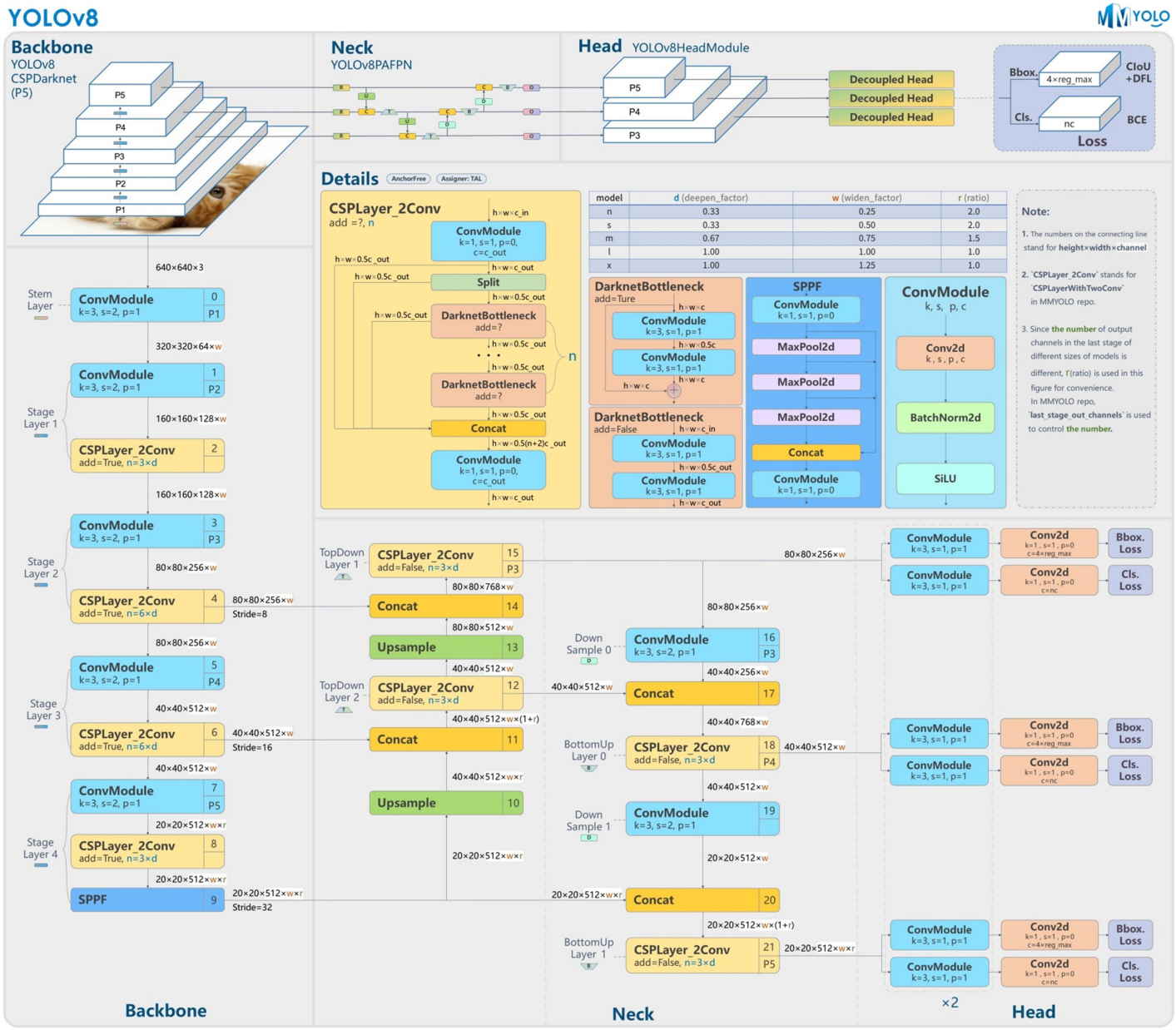


Fig.4 Anchor Free Detection

- **New Convolutions :** The stem's first 6x6 conv is replaced by a 3x3, the main building block was changed, and C2f replaced C3 and CBS is a block composed of a Conv, a BatchNorm and a SiLU later.In C2f, all the outputs from the Bottleneck (fancy name for two 3x3 convs with residual connections) are concatenated. While in C3 only the output of the last Bottleneck was used.



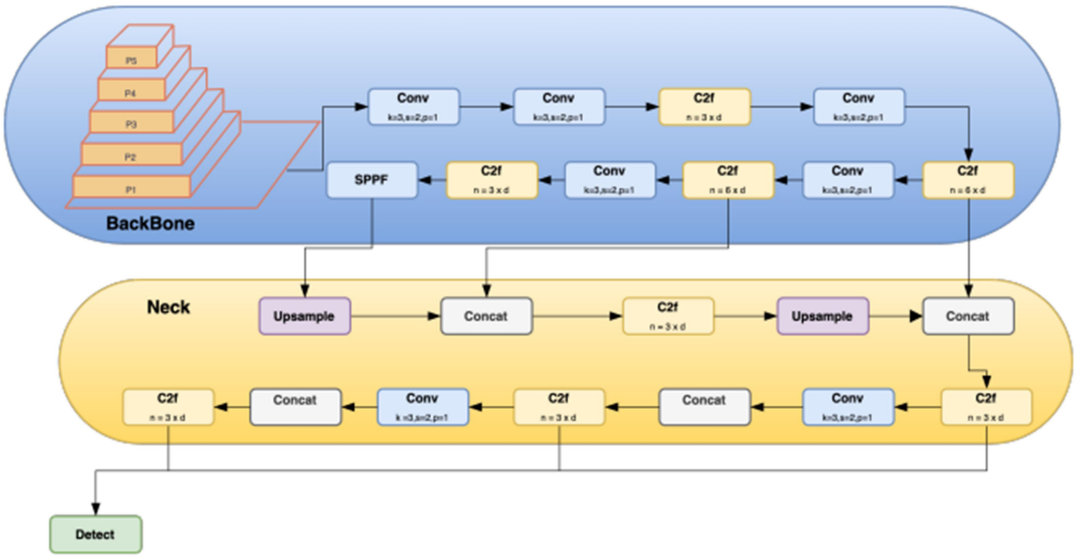


Fig.5 YOLOv8 Architecture

4. Methodology

The YOLOv8 object detection model for satellite images is implemented using the following steps:

- **Data Pre-processing:** The satellite images are pre-processed to enhance their quality, remove noise, and normalize the pixel values.
- **Dataset Creation:** An annotated dataset is created by manually labelling the objects of interest in the satellite images.
- **Model Architecture:** The YOLOv8 architecture is designed, consisting of a backbone network, feature pyramid, and detection layers.
- **Training:** The model is trained using the annotated dataset and optimized using techniques like stochastic gradient descent.
- **Evaluation:** The trained model is evaluated on a separate test dataset using evaluation metrics such as mean average precision (mAP).
- **Fine-tuning:** The model is fine-tuned using transfer learning techniques to improve its performance on specific satellite image domains.
- **Inference:** The trained model is used for object detection on new satellite images, providing bounding box coordinates and class labels.

4.1. Dataset

We collected this data from a variety of sources in order to have a comprehensive dataset for our project. The data from Various **Kaggle datasets** and **Sentinel** provides us with a wide variety of satellite imagery, while the data from **QGIS** and **NRSC** provides us with additional information about the ground features that are present in the images. The data from **Google Data Search** and the **AID** and **DOTA** datasets provide us with additional data that can be used to train machine learning models for object detection and tracking.

- **Kaggle:** A public data repository that contains a wide variety of datasets, including satellite imagery.

- **Sentinel data**: A collection of satellite imagery from the European Space Agency.
- **QGIS:** A free and open-source geographic information system (GIS) software.
- **Google Data Search:** A tool that allows users to search for and access datasets from a variety of sources.
- **DOTA Dataset:** DOTA is a large-scale dataset for object detection in aerial images. It can be used to develop and evaluate object detectors in aerial images.
- **AID Dataset:** AID is a new large-scale aerial image dataset, by collecting sample images from Google Earth imagery.

We believe that this dataset will be a valuable resource for our project, and we are grateful to the organizations that have made this data available to us.

After collecting the data, we annotated the images using the **LabelImg** and **Makesense.ai** tools.

We created labels for seven classes of objects:

- Car
- Ship
- Aeroplane
- Stadium
- Crossroad
- Railway station
- Storage tank

We choose these classes because they are common objects that are found in satellite imagery. We believe that these classes will be useful for a variety of applications, such as traffic monitoring, disaster response, urban planning and much more.

LabelImg and **Makesense.ai** are two annotation tools that can be used to annotate images for object detection.

LabelImg is a free and open-source tool that is available for Windows, macOS, and Linux. It is a simple tool that allows users to draw bounding boxes around objects in images.

Makesense.ai is a cloud-based tool that can be used to annotate images for object detection. Makesense.ai offers a variety of features.

The annotation process was time-consuming, but it was necessary to ensure that the labels were accurate. We used a combination of manual and automated methods to annotate the images. The manual method involved using the LabelImg tool to draw bounding boxes around the objects in the images. The automated method involved using the Makesense.ai tool to identify objects in the images using a deep learning model.

We believe that the annotated dataset will be a valuable resource for the research community. We will make the dataset available to the public, and we encourage other researchers to use it to develop new object detection algorithms.

4.2. Training

After annotating the images, we have made the test train split of the dataset: **80%** for training and **20%** for testing. This was done to ensure that the model was not over fitting to the training data.

The dataset used consisted of **1032** satellite images, with a **training dataset** containing **826** images comprising various classes of objects. The YOLOv8 model was trained using a batch size of 8 over **175 epochs**, followed by evaluation on a separate **testing dataset** consisting of **206** images. The reason for dividing the dataset into 80% training and 20% testing is to prevent over fitting. Over fitting occurs when a model learns the training data too well and is not able to generalize to new data. The reason for using a batch size of 8 is to improve the training efficiency. A batch size is the number of images that are processed at the same time during training. A larger batch size can improve the training efficiency, but it can also make the training process more unstable.

The reason for training the model for 175 epochs is to ensure that the model has converged. Convergence occurs when the model's

performance on the training dataset no longer improves.

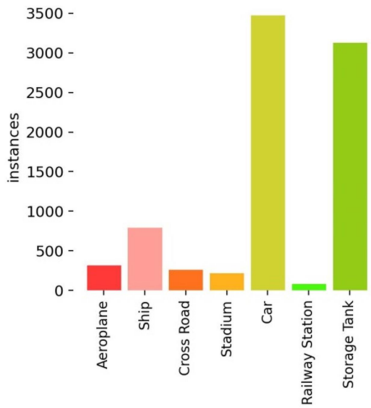


Fig.6 Labels

4.3. Implementations Details

The YOLOv8 object detection model for satellite images is implemented using the following tools and frameworks:

Programming Language: Python

Object Detection Library: Ultralytics

Frameworks & Libraries: PyTorch, PyYAML, Pillow, OpenCV, Pandas, Seaborn, Matplotlib

Annotation Tools: LabelImg, Makesense

Hardware: NVIDIA T4 Tensor Core GPU

Cloud Computing Platform: Google Colab

Front-end Framework: Streamlit

Google Colab is a cloud-based platform that allows users to run Python code in a web browser. It offers a user-friendly platform for computationally intensive tasks and is equipped with GPU resources for accelerating training and inference processes.

Ultralytics is a powerful and user-friendly open-source library that offers state-of-the-art computer vision algorithms and tools. It simplifies the process of implementing and deploying cutting-edge models. Its comprehensive documentation and active community support make it an excellent choice for those seeking to leverage the latest advancements in computer vision.

5. Evaluation Metrics

The performance of the YOLOv8 object detection model is evaluated using the following metrics:

- **Precision:** Precision is the fraction of the objects that were detected correctly. It is calculated as the number of true positives divided by the sum of the true positives and false positives.

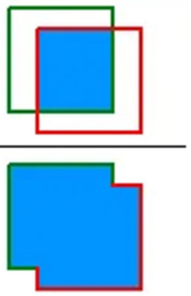
- **Recall:** Recall is the fraction of the objects that were actually present that were detected correctly. It is calculated as the number of true positives divided by the sum of the true positives and false negatives.

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{TP}{\text{all detections}}$$
$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{TP}{\text{all ground-truths}}$$

- **Precision-Recall (PR) Curve**: The precision-recall (PR) curve is a plot of precision and recall at varying confidence values. For a good model, precision and recall stay high even when the confidence score varies.

- **Intersection over Union (IoU):** IoU is a measure of the overlap between the predicted bounding box and the ground truth bounding box. It is calculated as the area of the intersection divided by the area of the union.

$$IOU = \frac{\text{area of overlap}}{\text{area of union}} =$$



- **Average Precision:** AP@α is Area Under the Precision-Recall Curve(AUC-PR) evaluated at α IoU threshold. Formally, it is defined as follows.

$$AP@\alpha = \int_0^1 p(r)\, dr$$

- **Mean Average Precision (mAP):** mAP is a measure of the overall accuracy of the object detection model. It is calculated by averaging the precision scores at different recall levels.

$$\text{mAP}@\alpha = \frac{1}{n} \sum_{i=1}^{n} \text{AP}_i \quad \text{for n classes}$$

mAP@[IoU threshold]: This type of mAP calculates the average precision at a specific Intersection over Union (IoU) threshold.
The IoU threshold determines the minimum overlap required between the predicted and ground truth bounding boxes to consider a detection as a true positive.

mAP@50 calculates the average precision at an IoU threshold of 0.5, measuring object detection accuracy with a focus on localization. It considers detections with an IoU of 0.5 or higher with the ground truth.

mAP@50-95 computes the average precision across IoU thresholds from 0.5 to 0.95, providing a comprehensive evaluation of detection performance at various levels of overlap. This metric captures the model's ability to detect and localize objects accurately across a wide range of IoU thresholds, offering a more comprehensive assessment of object detection accuracy.

These are just a few of the evaluation metrics that could be used to evaluate the performance of the object detection model for satellite images. The specific metrics that are used will depend on the specific goals of the project.

6. Results

The model demonstrated strong overall performance with an average precision of 88.9%, recall of 84.6%, mAP@50 of 88.5%, and mAP@50-95 of 50.9%. Among the individual classes, Stadium achieved the highest precision (86.1%) and recall (84.8%), resulting in an impressive mAP@50 of 90.0% and mAP@50-95 of 63.6%. Storage Tank exhibited the highest precision (96.0%) and mAP@50-95 (59.1%), indicating accurate detection and localization. However, Car had a relatively lower recall (70.1%) compared to other classes, leading to a lower mAP@50 (82.0%) and mAP@50-95 (43.2%).

Overall, the model's performance varied across different classes, indicating variations in the detection capabilities for different object categories. Further analysis can be performed to understand the reasons behind the variations and identify potential areas of improvement.

Class	Precision	Recall	mAP50	mAP50-95
All	88.9	84.6	88.5	50.9
Aeroplane	89.4	80.5	84.1	37.8
Ship	90.6	82.7	89.5	53.2
Cross Road	87.0	87.1	87.5	45.1
Stadium	86.1	84.8	90.0	**63.6**
Car	87.6	70.1	82.0	43.2
Railway Station	85.2	93.3	90.9	54.2
Storage Tank	**96.0**	**93.5**	**95.5**	59.1

Table 1: Evaluation Metrics of the Model for Object Detection

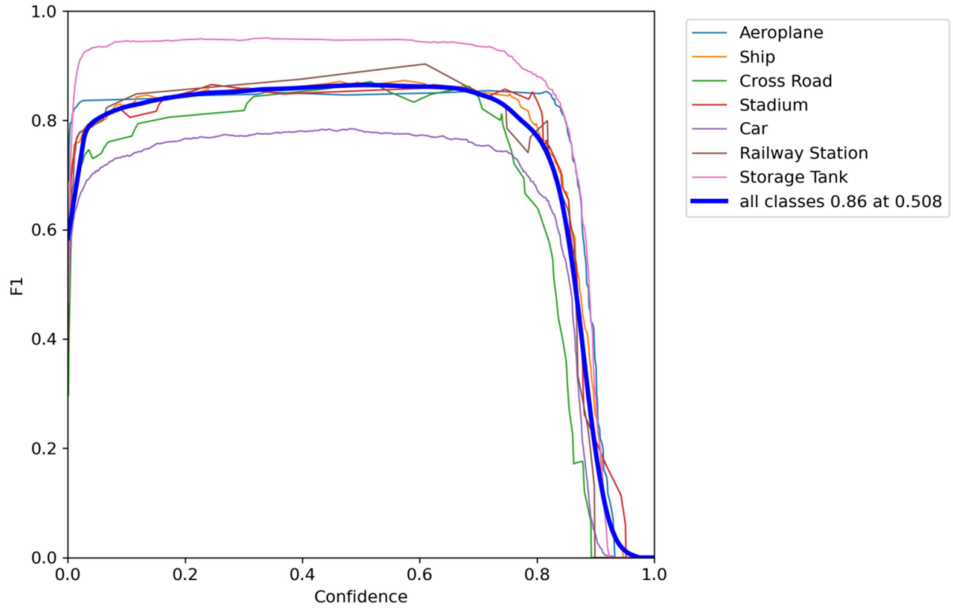


Fig.7 F1-Confidence curve

The x-axis of the graph shows the confidence of the model's predictions, while the y-axis shows the F1 score of the model's predictions. The F1 score is a measure of the overall performance of the model. The text and legend of the image tell us that the model has a precision of **86.0%** and a recall of **50.8%** when the confidence threshold is set to **0.508**.

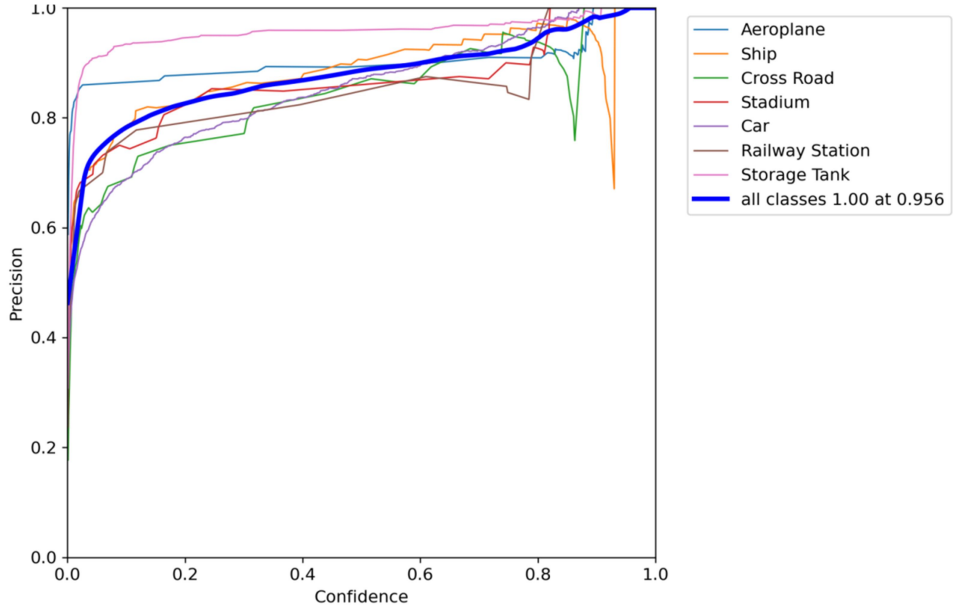


Fig.8 Precision-Confidence curve

The x-axis of the graph shows the confidence of the model's predictions, while the y-axis shows the precision of the model's predictions.. The blue line shows the precision-Confidence curve for the Aeroplane class. The red line shows the precision-Confidencel curve for the Storage Tank class.

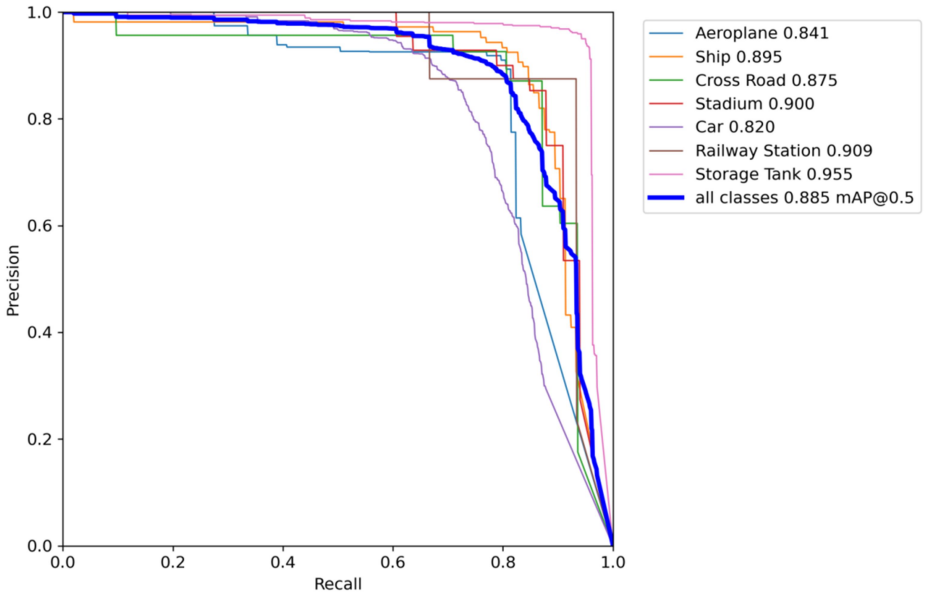


Fig.9 Precision Recall curve

The model has a higher precision for the Aeroplane class than for the Storage Tank class. This means that the model is more accurate when it predicts that an image contains an aeroplane than when it predicts that an image contains a storage tank.

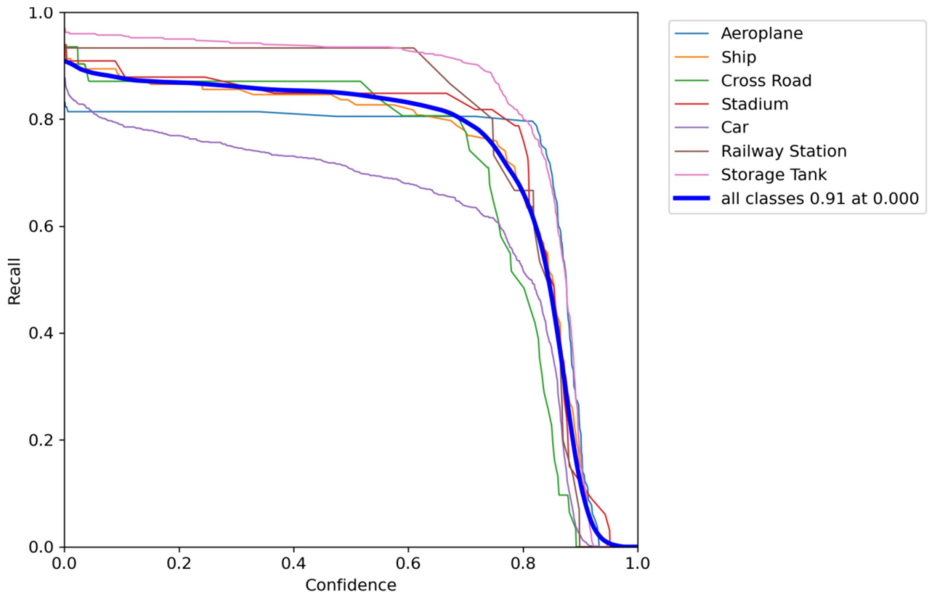


Fig.10 Recall Confidence curve

The model has a lower confidence for the Storage Tank class than for the Aeroplane class. This means that the model is less sure of its predictions when it detects a storage tank than when it detects an aeroplane.

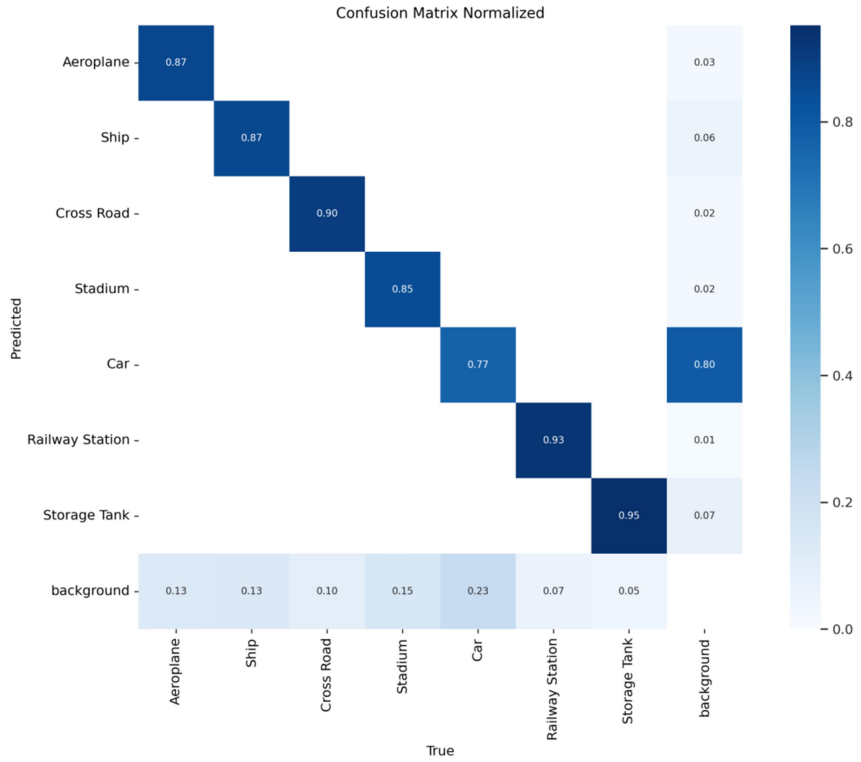


Fig.11 Confusion Matrix

The confusion matrix shows that the model has a good performance at classifying cross roads and stadiums. However, it does have some difficulty classifying cars, railway stations, and storage tanks. The background class is a special class that is used to represent samples that do not belong to any of the other classes. In this case, the background class represents samples that are not stadiums, cars, railway stations, or storage tanks.

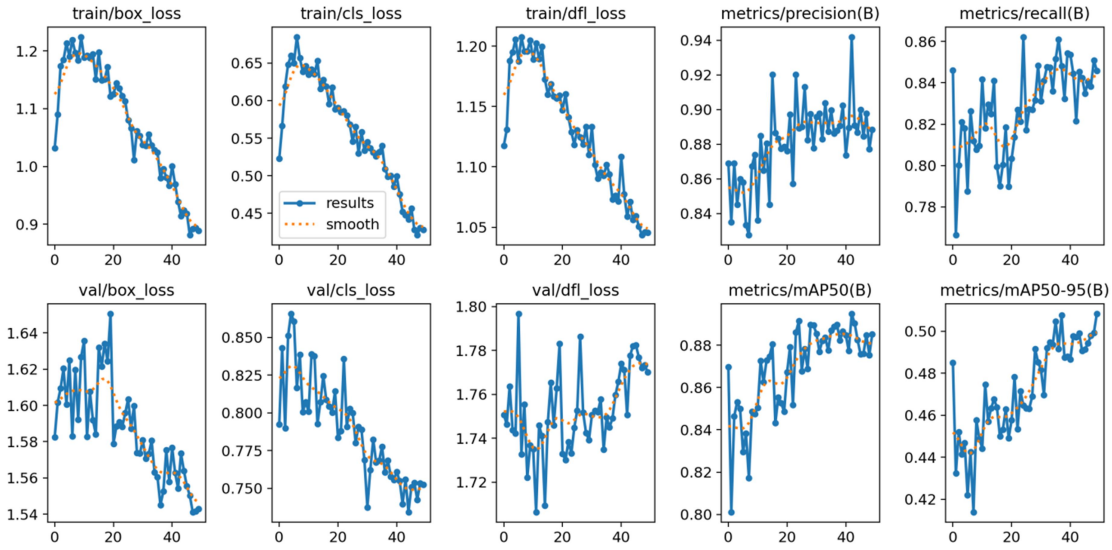


Fig.12 Performance Metrics

The graphs show that the model's performance improves over time, as the loss decreases and the precision and recall increase. The model's performance is also better on the validation set than on the training set, which indicates that the model is not overfitting to the training data.

The text in the image shows the values of the loss, precision, recall, and mAP metrics at different epochs. The values of these metrics can be used to evaluate the performance of the model and to compare different models.

Overall, the image provides a good overview of the performance of the model on different types of waves. The graphs show that the model's performance improves over time, and that the model is able to detect objects in an image with high accuracy.

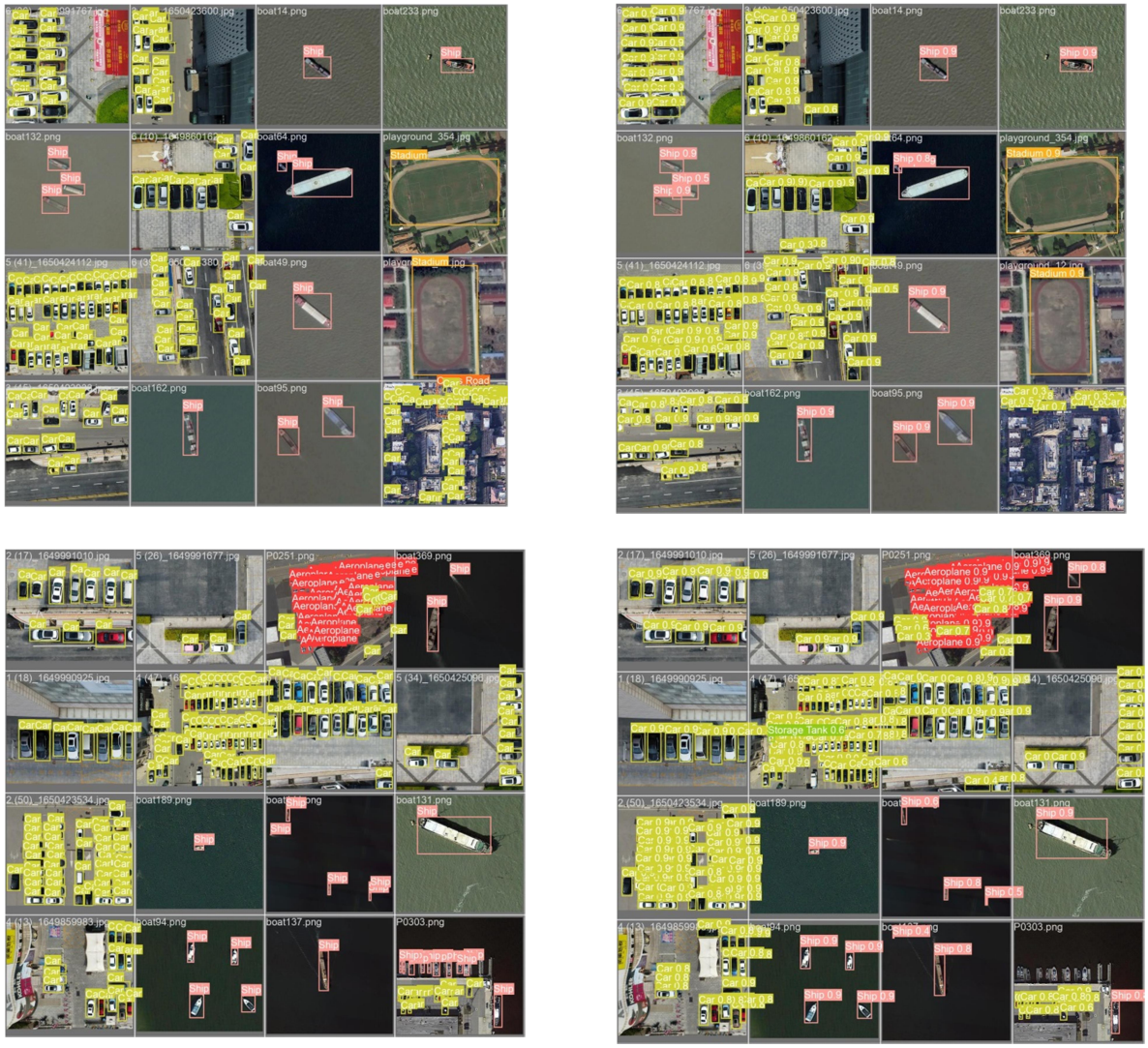


Fig.13 Labled Images (Left) & Predicted Images (Right)

7. Deployment

The Streamlit web app provides an intuitive user interface for multi-class object detection using YOLOv8. Users can upload satellite imagery in various formats, including JPEG, PNG, BMP, TIFF, and more. The app displays detected objects with class labels and bounding boxes. Customization options for adjusting confidence thresholds allow users to fine-tune the detection process.

Streamlit Web App Deployment on NRSC Server

To deploy a Streamlit web app on the NRSC server, follow these steps:

Prepare the Web App: Develop and test the Streamlit web app on your local machine to ensure it functions correctly.

Server Access: Obtain the necessary credentials and permissions to deploy the app on the NRSC server.

Set Up Dependencies: Install the required libraries and dependencies on the server to support the web app.

Copy the Web App Files: Transfer the app files, including the Python script and any additional modules or static files, to the server.

Environment Setup: Create a virtual environment on the server and install the necessary Python packages.

Configure the Server: Adjust server configurations, such as firewall rules and port settings, to enable access to the app.

Run the Web App: Execute the Streamlit web app on the server, ensuring it listens on the designated port.

Testing and Maintenance: Validate the app's functionality, perform thorough testing, and monitor and maintain the app regularly.

Deploying the Streamlit web app on the NRSC server provides a secure platform for showcasing and utilizing the object detection model.

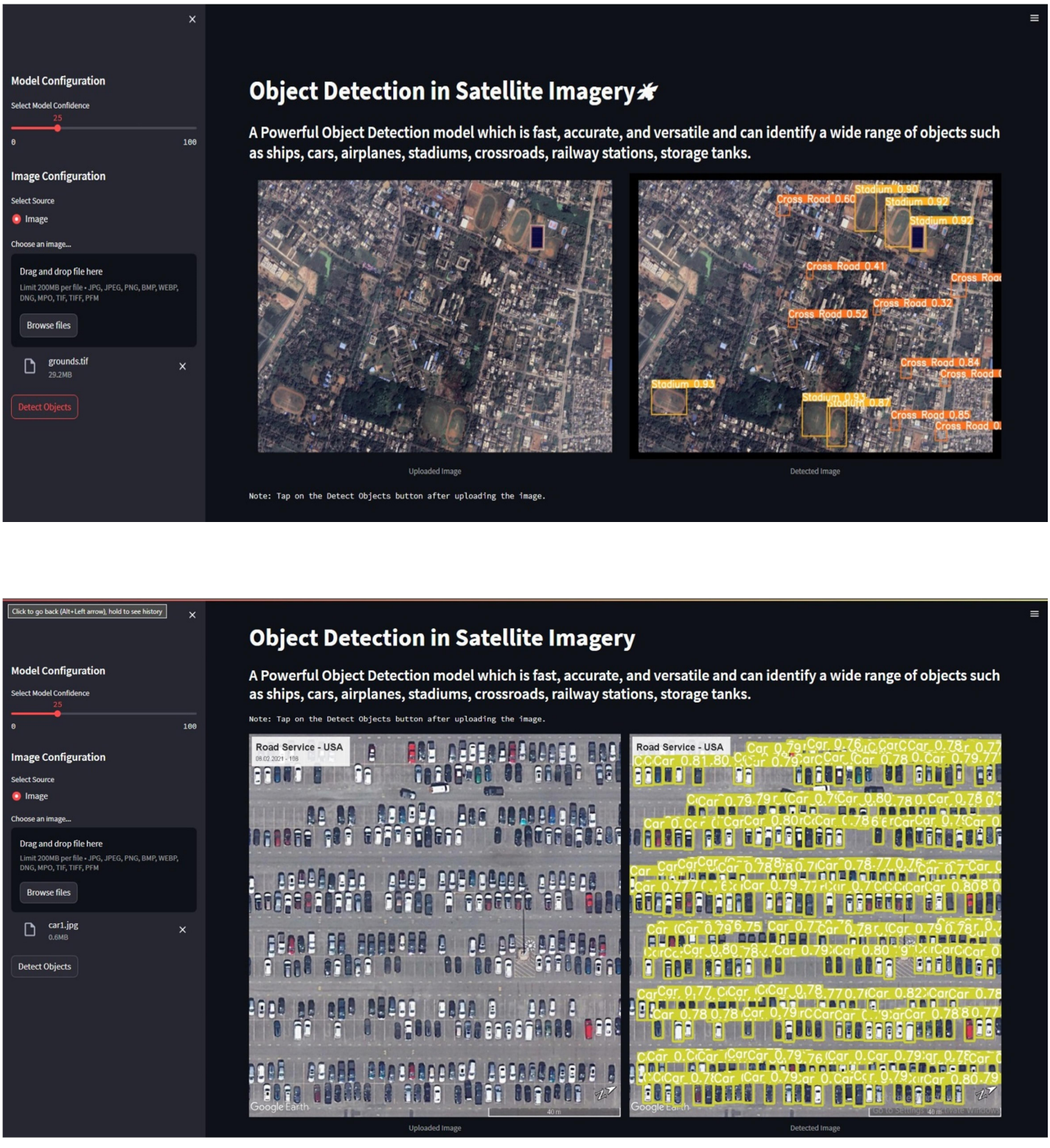


Fig.14. Streamlit Web App

Fig. 15.1. Input Image

Fig. 15.2. Output Image

Fig. 16.1. Input Image

Fig. 16.2. Output Image

8. Conclusion

We have performed multi-class object detection in satellite imagery using the YOLOv8 model. The model achieved impressive results across various evaluation metrics, including precision, recall, mAP@50, and mAP@50-95.

The overall performance of the model was strong, with an average **precision** of **88.9%** and a **recall** of **84.6%.** The **mAP@50** score was **88.5%,** indicating accurate object detection at an IoU threshold of 0.5. The **mAP@50-95** value of **50.9%** demonstrated the model's ability to maintain detection accuracy across a range of IoU thresholds from 0.5 to 0.95.

Analyzing the individual classes, we observed consistent high precision and recall for most object categories. Stadium and Storage Tank stood out with exceptional performance, achieving precision and recall scores of 86.1% and 84.8%, and 96.0% and 93.5%, respectively. These results highlight the model's capability to accurately detect and classify objects within satellite imagery, particularly in the context of stadiums and storage tanks.

However, certain classes, such as Car, showed relatively lower recall, impacting the overall performance of the model. This suggests the need for further investigation and potential enhancements to improve the detection of certain object categories.

In conclusion, our project demonstrates the effectiveness of YOLOv8 for multi-class object detection in satellite imagery. The achieved results indicate its potential for various applications, including urban planning, environmental monitoring, and disaster management. Further research can focus on refining the model's performance for challenging classes and exploring optimizations to improve overall detection accuracy. By advancing object detection in satellite imagery, we can unlock valuable insights and facilitate decision-making processes in various domains.

9. Future Work

Enhanced Model Architecture: Explore and experiment with advanced architectures or other state-of-the-art models, to further improve the object detection performance in satellite imagery. Investigate techniques like attention mechanisms, feature pyramid networks, or spatial-temporal modelling to enhance the model's ability to handle complex scenes and occlusions.

Dataset Expansion and Diversity: Acquire or generate larger and more diverse datasets to improve the model's generalization capabilities. Consider including more challenging scenarios, different weather conditions, and varying object scales to ensure robustness and adaptability of the model.

Fine-Grained Object Classification: Extend the object detection framework to incorporate fine-grained classification for specific object classes. Implement techniques like transfer learning or feature refinement to enable the model to distinguish sub-categories within each class, such as different types of cars, ships, or storage tanks.

Multi-Modal Fusion: Explore the integration of multi-modal data sources, such as satellite imagery combined with radar or LiDAR data, to enhance the accuracy and robustness of object detection.

Real-Time Object Detection: Investigate techniques to optimize the model and inference pipeline for real-time or near real-time object detection in satellite imagery. Explore efficient model architectures, pruning techniques, and hardware optimizations to achieve faster inference speeds without compromising accuracy.

Object Tracking and Temporal Analysis: Extend the research to incorporate object tracking and temporal analysis in satellite imagery. Develop methods to track objects across frames and analyze their movements and interactions over time, enabling applications such as activity recognition and anomaly detection.

By pursuing these future research directions, advancements in multi-class object detection in satellite imagery can be achieved, leading to improved accuracy, efficiency, and practical applications.

10. References

[1] Joseph Redmon, Ali Farhadi "**YOLO9000: Better, Faster, Stronger** (https://arxiv.org/abs/1612.08242)

[2] Joseph Redmon, Ali Farhadi "**YOLOv3: An Incremental Improvement**" (https://arxiv.org/abs/1804.02767)

[3] Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao "**YOLOv4: Optimal Speed and Accuracy of Object Detection**" (https://arxiv.org/abs/2004.10934)

[4] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi "**You Only Look Once: Unified, Real-Time Object Detection**" (https://arxiv.org/abs/1506.02640)

[5] Chien-Yao Wang, Alexey Bochkovskiy, Hong-Yuan Mark Liao" **Scaled-YOLOv4: Scaling Cross Stage Partial Network**" (https://arxiv.org/abs/2011.08036)

[6] Adam Van Etten "**You Only Look Twice: Rapid Multi-Scale Object Detection In Satellite Imagery**" (https://arxiv.org/abs/1805.09512)

[7] Hong gong, "**Swin-Transformer-Enabled YOLOv5 with Attention Mechanism for Small Object Detection on Satellite Images**" (https://www.mdpi.com/2072-4292/14/12/2861)

[8] Muhammed Enes ATİK Zaide DURAN Roni ÖZGÜNLÜK **"Comparison of YOLO Versions for Object Detection from Aerial Images"** (https://dergipark.org.tr/en/pub/ijegeo/issue/66005/1010741)

[9] **"Small Ship Detection on Optical Satellite Imagery with YOLO and YOLT" (**https://link.springer.com/chapter/10.1007/978-3-030-39442-4_49**)**

[10] **YOLO Object Detection Explained** (https://www.datacamp.com/blog/yolo-object-detection-explained)

[11] **A Basic Introduction to Object Detection** (https://www.analyticsvidhya.com/blog/2022/03/a-basic-introduction-to-object-detection/)

[12] **Introduction to Object Detection** (https://www.hackerearth.com/blog/developers/introduction-to-object-detection/)

[13] **Dive into YOLOv8: How does this state-of-the-art model work?** (https://openmmlab.medium.com/dive-into-yolov8-how-does-this-state-of-the-art-model-work-10f18f74bab1)

[14] **What is YOLOv8? The Ultimate Guide.** ((https://blog.roboflow.com/whats-new-in-yolov8/#:~:text=YOLOv8%20was%20launched%20on%20January%2010th%2C%202023.)

[15] **Streamlit documentation** (https://docs.streamlit.io/)

[16] **Ultralytics YOLOv8 Docs (**https://docs.ultralytics.com**)**

*****