# Introduction

I have constructed a CV pipeline that attempts to identify ships on the ocean. This is a problem that has numerous commercial, military, and academic applications. Some applications utilize advanced imagery but given my inexperience with radar imagery, I have limited this project to simple RGB satellite pictures. During this project, I ran in to repeated technical challenges and found no workable solution to developing a ship-detection algorithm using open-source object-detection tools. Part of my problem during this process is that the algorithm that I thought would be most useful, Mask R-CNN, seems to be unavailable in Google Colab. Therefore, the main thrust of my work focused on the “Yolo v5” object detection algorithm.

# Data Source

For this project, I used a subset of Microsoft® Bing™ Maps called the MAritime SATellite Imagery (MASATI-v2). This dataset contains “optical aerial images from the visible spectrum”, which is to say they are simple RBG pictures taken top-down (Gallego, 2018). All the images evaluated to date are squares with a length of 512 pixels, limiting the amount of preprocessing work needed to begin the modeling process.

The data is available at the following link: <https://www.iuii.ua.es/datasets/masati/index.html> and has been made publicly available for non-profit research or educational purposes. Accessing this dataset requires the completion of a short form which asks for basic user information and the intended use.

There is a wide range of heterogeneity in the images. There are built-up harbors and abandoned beaches, differing tidal ranges, ocean depths, and weather conditions.

## Potential Challenges

The challenge that I anticipated running into was the size of the ships compared to the entirety of the image. After reviewing a few images, it appeared that there were many instances where the ship represented only a few pixels. In other instances, the ship was significantly smaller than the dominant portions of an image, like the coastline or a series of breaking waves. Given this, I attempted a few different approaches to limit the influence of the other pieces of an image on the overall detection.

# Data Preparation

One of the main reasons that this data was selected was for its relative cleanliness. The main work in terms of cleaning was on reformatting the bounding boxes to the yolo formatting. Another decent part of my data cleaning time was focused on reorganizing the directories into their ‘train/validate/test’ splits and working within the “Google Drive” system. Uploading the images to google drive proved to be a very slow process resulting in lots of down time. However, once the data was properly organized the images were ready to be fed into the Yolo Model for Object Detection.

## Exploring the Hough Transform

One of the methods that I thought would be useful to help with this analysis was to apply the Hough Transform to the black and white images. The basic idea of the Hough Transform is to identify straight lines in an image. I thought that it would identify the wakes of the ships or maybe even the ships themselves. However, as Appendix Figure A1 shows, the images were detecting many straight lines. Thus, applying the filter appeared more likely to introduce noise rather than signal, and I decided to therefore proceed without this potential transform in mind.

# Object Detection Algorithms

## Mask R-CNN

The Mask R-CNN algorithm is an image segmentation system that identifies key areas in an image and then develops a classification for that area. I was particularly interested in this method because I thought that the image segmentation portion of the algorithm would help me to overcome one of my anticipated challenges: that the ship itself would get overwhelmed by the activity going on in the rest of the image.

I realized that there was a dependency issue with this algorithm. The requirements.txt file from Mask R-CNN’s github says that it requires tensorflow>-1.3.x. However, upon much review, I have come to believe that this also requires tensorflow<2.x. There was, at the very least, some reorganization that occurred when tensorflow was updated which means that the “Layer” class from keras.engine has been removed or relocated. I believe there is a work around for this on my personal machine, but I do not have the resources on my computer to conduct any sort of training locally. Google Colab, where the rest of the work is done, has suspended support for tensorflow version 1.x. Therefore, it is my understanding that Mask R-CNN cannot be deployed in a Google Colab environment.

## Yolo v5

The Yolo class of algorithms is another object-detection algorithm which looks at a small segment of an image and produces its estimate of what is in this segment. I was very interested in the method because I thought it would ensure that the small ships in the image would not be drowned out by the noise from the rest of the image. I did some reading online and found that this model trains very slowly when reading to and from the cloud, such as Google Colab. I originally tried training the moder over 10 epochs, but eventually tuned it down to 3 epochs in order to have results complete in a useful amount of time.

The results of the model training indicated that the model was having a hard time distinguishing the ships from the background noise. This is most acutely seen in the Recall Curve, which quickly collapses to zero before confidence reaches 0.2, shown in Appendix Figure A2. When I evaluated the model on the test set, this problem further revealed itself when the trained weights were unable to identify a single ship in the training set.

# Conclusion

Though I was unable to successfully develop a working CV pipeline for this project, I think there are several areas for future work on this project. First and foremost, running the Mask R-CNN on a friendly cloud environment, like Amazon Web Services (AWS). Working in AWS provides complete control over the environment, but that can come at a considerable cost.

Another approach to consider is upscaling the images such that the ships comprise more pixels. I did not try this initially because the Yolo model works by calculating the share of an image that an object takes up, so simply upsizing the images would have no effect. This approach would work better in conjunction with the Mask R-CNN model.

# Appendix Figures

## A1: Hough Transform Lines

A picture containing screenshot, map

Description automatically generated

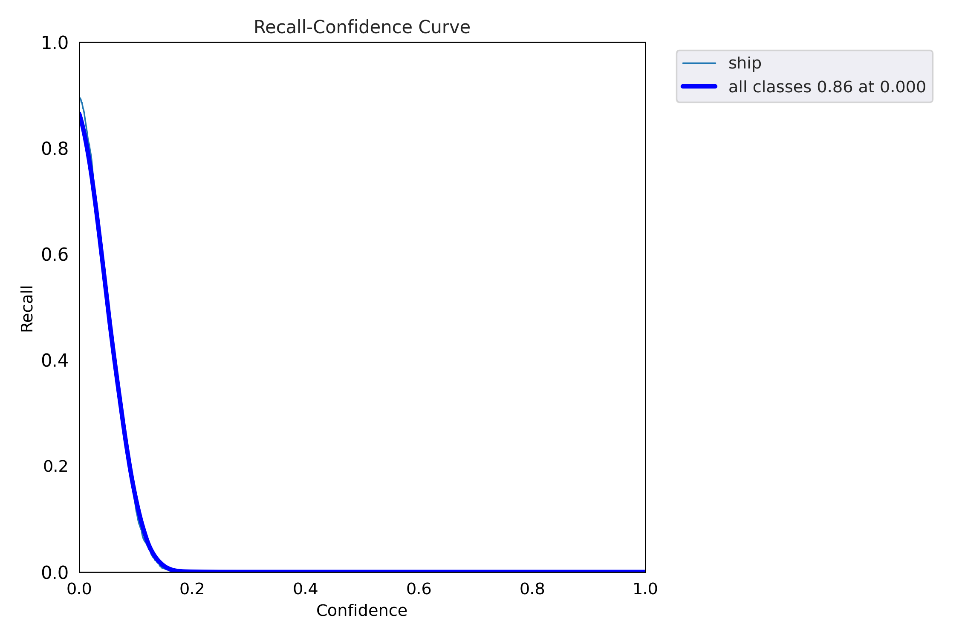
A picture containing screenshot, map, art

Description automatically generated

A picture containing screenshot, line, parallel, design

Description automatically generated

## A2: Yolov5 Training: Recall Curve



## A3: Summary Statistics of Objects

