Spring 5-1-2021

A Deep Learning-Based Automatic Object Detection Method for Autonomous Driving Ships

Ojonoka Erika Atawodi
University of Southern Mississippi

A DEEP LEARNING-BASED AUTOMATIC OBJECT DETECTION METHOD FOR AUTONOMOUS DRIVING SHIPS

by

Ojonoka Erika Atawodi

A Thesis
Submitted to the Graduate School,
the College of Arts and Sciences
and the School of Computing Sciences and Computer Engineering
at The University of Southern Mississippi
in Partial Fulfillment of the Requirements
for the Degree of Master of Science

Approved by:

Dr. Bo Li, Committee Chair
Dr. Lina Pu
Dr. Chaoyang (Joe) Zhang

May 2021

ABSTRACT

An important feature of an Autonomous Surface Vehicles (ASV) is its capability of automatic object detection to avoid collisions, obstacles and navigate on their own.

Deep learning has made some significant headway in solving fundamental challenges associated with object detection and computer vision. With tremendous demand and advancement in the technologies associated with ASVs, a growing interest in applying deep learning techniques in handling challenges pertaining to autonomous ship driving has substantially increased over the years.

In this thesis, we study, design, and implement an object recognition framework that detects and recognizes objects found in the sea. We first curated a Sea-object Image Dataset (SID) specifically for this project. Then, by utilizing a pre-trained RetinaNet model on a large-scale object detection dataset named Microsoft COCO, we further fine-tune it on our SID dataset. We focused on sea objects that may potentially cause collisions or other types of maritime accidents. Our final model can effectively detect various types of floating or surrounding objects and classify them into one of the ten predefined significant classes, which are buoy, ship, island, pier, person, waves, rocks, buildings, lighthouse, and fish. Experimental results have demonstrated its good performance.

ACKNOWLEDGMENTS

I would like to express my sincere appreciation to my supervisors, Dr Bo Li and Dr Lina Pu for their assistance, mentorship and supervision during the course of my master's degree, especially in the preparation of this thesis and the research knowledge they have taught me over time.

I would also like to thank my committee member, Dr Chaoyang (Joe) Zhang whose classes prepared me for this thesis and introduced me to some of the fundamentals of machine learning.

DEDICATION

This research is dedicated to my parents Prof. and Dr Atawodi, my family Mr Ilemona Atawodi, Dr Oguche and Dr Ojochogwu Alhassan, Ms Aisha Alidu, Mr and Mrs Richard Alidu, Dr and Mrs Sunday Idakwo, Yahaya Atawodi, Abdulsalam Atawodi, Yusuf Atawodi, Edoeje David, Mrs Sadia, The Atawodis, The Alidus. My friends Aya Gberikon, Ene Abba, Shalom Frank, Tomi Adedeji, Ewoma Eguwe, Matilda Ajisafe, Blessing Ebegbuna, Racahel Ifietekhai, Tomi Amenze, Bisola Animashaun and my precious nephew Aaron Ajogwu Alhassan. Thank you all for your encouragement and prayers.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

LIST OF ABBREVIATIONS

BM	Base Model
IR	Infrared
EO	Electro-optical
NLP	Natural Language Processing
R-CNN	Region Based Convolutional Networks
YOLO	You Only Look Once
SSD	Single-shot Detector
ASV	Autonomous Surface Vehicle
USV	Unmanned Surface Vehicle
IoU	Intersection of Union
SID	Sea-object Image Dataset
mAP	Mean Average Precision
FPN	Feature Pyramid Network
COCO	Common Objects in Context
AIS	Automatic Identification System
AFP	Artificial Potential Field
DRL	Deep Reinforcement Learning
CAFAR	Constant False Alarm Rate
CA-CAFAR	Cell Average Constant False Alarm Rate
DDPG	Deep Deterministic Policy Gradient

CHAPTER I - INTRODUCTION

With the rapid growth in technology and computing advancement computers are now performing complex Artificial Intelligence tasks in the areas of natural language processing (NLP), fuzzy logic, robotics, neural networks and expert systems, and these areas are now being applied in our day to day activities. Virtual assistants, autonomous self-driving vehicles, image recognition, speech recognition and so many more are real-life examples of them.

Object recognition is a computer vision technique that involves identifying, detecting, and classifying objects in videos or images. Object detection is a subset of object recognition that goes further to localize and classify objects within an image, and this technique helps to address the task of predicting location and class of an object in an image as it poses to be an important problem in computer vision [1]. Recently, many object detection algorithms have produced significant advancement in performance as a result of leveraging machine learning, especially deep learning, techniques.

There are two main types of object detectors. The first type is the two-stage detector, which generates areas of interest using a Region Proposal Network in the first stage, and then proceed to transmit the proposals through the pipeline for classification and bounding box regression [1]. Examples of two-stage detectors are Faster R-CNN (Region-based Convolutional Neural Networks) or Mask R-CNN [2].

On the other hand, we also have one-stage object detectors which are often more efficient than the two-stage ones due to their simple architecture [3]. Modern one-stage object detectors [4] adopt a simple fully convolutional architecture [5], which produces classification probabilities and box offsets at respective spatial position, in terms of pre-

defined anchor boxes [3]. Examples of these single-stage detectors are YOLO (You Only Look Once) [5], SSD (Singe Shot MultiBox Detector) [6] and RetinaNet [7].

In this thesis, the problems associated with object detection in the context of autonomous maritime vehicles are investigated, more categorically object detection in the scenario of various objects found in the sea, where the objects of interest were represented by 10 different classes.

This aim of this research is to build an object detection model, using RetinaNet to identify, classify and localize classes of maritime images and Coco dataset to leverage features and evaluate what combination of features would provide the best accuracy and results for autonomous vehicles.

1.1 Background

Maritime accidents, although unpredictable, have been grouped into four categories based on the incidents that have occurred and been reported in the past. The categories are as follows: equipment failure, lack of proper training and human error, collision at sea and poor maintenance [8].

Human errors have been tagged as one of the main reasons for maritime accidents in the world. Hence, this is also why currently maritime vehicles are transitioning to reduce the extent of human errors' impact, resulting in the creation and rise of autonomous maritime vehicles. Besides human errors, collision is a big issue that maritime faces; maritime transportation involves a wide range of objects that may collide each other while in the sea. These objects include other maritime vehicles, rocks, sea animals, anchors, mooring buoys and iceberg [9].

Autonomous surface vehicles (ASVs) can be defined as an unmanned vessel or vehicle that performs a series of tasks in an unorganized environment, without intervention or control from human operators [10] to exhibit unpredictable dynamics.

A key technology that has been recently applied to unmanned vehicles is visual recognition irrespective of the field; land, aerial and maritime or industry; research, military, and civil engineering. It is important for these vehicles to easily identify objects in their surroundings to be able to make informed decisions ranging from avoidance, following path, reading signs or interaction with the objects [11].

Table 1.1 *Applications of Autonomous Surface Vehicles (ASVs)*

Areas of ASV Application	**Specific Application**
Research (Scientific)	Robotic research, Sail drone
Oceanography exploration	Hydrographic survey, Seafloor mapping, Oceanographic data collection
Environmentally sustainable missions	Climate monitoring, Environmental surveys, Oil and gas
Military & naval adaptation	Seaborn targets, Anti-submarine drones & warfare
Commercial transportation	Passenger ferries, Commercial shipping
Others	Surveillance, Inspection of infrastructure, Seaweed farming

Autonomous surface vehicles are applicable to a wide range of areas (as listed in Table 1) which include but are not limited to research in the sciences, oceanography

exploration, environmentally sustainable missions [12], military and naval adaptation, commercial transportation, and other applications.

Object detection remains as one of the important research areas in computer vision, which has continuously been developing over the years. This importance also extends to ASVs as obstacle detection is of central importance to autonomous vehicles patrolling waterbodies [13]. Before further development, ASVs were constantly being worked on and minimize the need for human control to reduce and eliminate human errors, which will in turn achieve effective, reliable, and safe operations [14].

There are numerous factors that affect the results of object detection systems. The most significant factor that stands out is the feature description and learning algorithm [15].In the last few years, great progress has been made to handle these factors [16]. Older existing algorithms are now being outperformed by convolutional neural network (CNN) based deep learning technologies [17]. R-CNN was introduced to the object detection field, and it has achieved more than 30% relative progress compared to existing algorithms in terms of precision. R-CNN was designed to carry out region proposal generation and classification independently, unlike single-stage detection algorithms which supersede them by performing all the processes of object detection at the same time. YOLO and RetinaNet are two representative one-stage object detection algorithms, and both exhibit high speed. However, RetinaNet is even more accurate and efficient, because it has the speed of single-stage detectors as well as superior accuracy than previous either one-stage or two-stage detectors. For this reason, the RetinaNet algorithm and its network model are chosen for the object detection module in this project.

In this work, we propose and implement the use of one of the fastest object detectors in current related literature. This thesis focuses on object recognition for an autonomous driving ship. The focus was on neural networks-based approaches with an emphasis in convolutional neural networks-based deep learning techniques. A combination of COCO [18] dataset and our created Sea-object Image Dataset (SID) which consists of images of objects found in a maritime environment was used for training and experimentation. After training, an analysis was carried out to discuss the results from the multiples trainings, evaluate the limitations of the study and present future work ideas.

CHAPTER II – LITERATURE REVIEW

In this chapter, background of definitions and explanations that are necessary to understand machine learning algorithms are provided, especially deep neural networks, and its subclass CNN, which are frequently used when it comes to image recognition.

A description of a specific CNN-based object detection model called RetinaNet, and related works to the topic are also discussed later in this chapter.

2.1 Machine Learning

Machine learning is used to train machines to interpret pattern or extract specific information from data. With the numerous datasets available for different industries and challenges, the demand for machine learning remains on the rise.

After a machine learning algorithm has recognized a pattern, classification takes place at a much faster rate than any human-controlled system can operate at. Although many studies have been carried out on how to train machines to learn by themselves [16] [19], the main purpose of machine learning is for the machines to learn from the data provided as input.

2.1.1 Machine Learning Algorithms

Although, there are numerous classes of machine learning algorithms as shown in Figure 2.1, we discuss three major classes of machine learning algorithms categorized by how the algorithm is trained.

- **Supervised Learning:** This algorithm requires external assistance, a ground-truth dataset and its focus is prediction. The ground-truth dataset is divided into training and testing datasets, while the training dataset has been labelled and the testing dataset needs to be classified or predicted.

The supervised learning algorithm learns patterns from the training dataset and further apply the patterns to the testing dataset for classification [20]. Three popular examples of supervised algorithm are decision trees, naive bayes, and support vector machines (SVM).

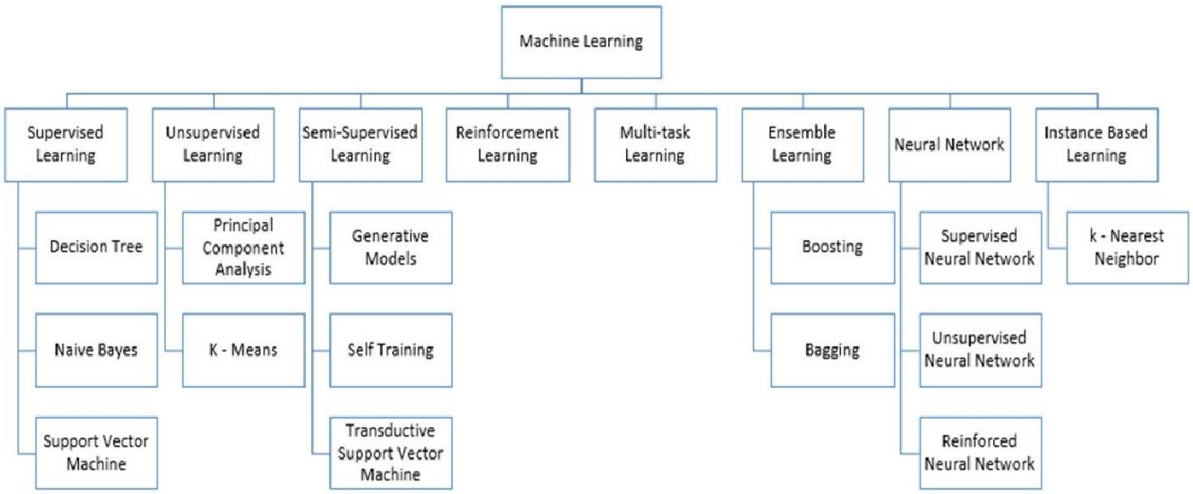


Figure 2.1 *Types of machine learning algorithms [21]*

- **Unsupervised Learning:** Unsupervised learning is a classification algorithm that learns new features using unlabeled data. When data input is introduced, features that have been previously learned are used to recognize the current class of the data. Examples of this algorithm are Principal Component Analysis (PCA) and K-means.
- **Reinforcement Learning:** Reinforcement learning algorithm learns by making decisions which are generally based on what next action to take, in such a way that the result is positive. For this type of algorithm, the machine has no idea on which actions to take until it's in a situation to do so. This learning algorithm depends on two features which are trial and error, as well as search, and delayed outcome.

2.1.2 Machine Learning Process

This outlines the 7 steps of machine learning [22].

1. **Data Collection:** This process is research specific since relevant data depends on the project that is at hand [23]. Therefore, once you know what you want and the equipment in your hand, the first step would be to gather quality training data. Quality data is significant as it determines how well the model will perform.
2. **Data pre-processing:** After gathering training data, transformation into the right format and loaded into the machine learning algorithm for training and then pre-processed data is further split into two different datasets, that is the training and testing datasets. Feature extraction happens at this level and helps to reduce the possibility of overfitting to improve generalization.
3. **Choosing a model:** Different algorithms work for different tasks. Therefore, depending on the task to be carried out a right model should be selected.
4. **Training:** At this stage the model is built, and the process may be carried out once or repeatedly, depending on the algorithm.
5. **Evaluation:** Once training is completed and the model has been built, the testing dataset is fed into the model for evaluation, just in the same way that the training dataset was trained. The classification accuracy is taken at this point. Results closer to a hundred percent indicates the possible perfect performance of the model.
6. **Parameter Tuning:** Hyperparameter tuning takes place in this step. One or more parameters may be tuned to improve performance. Hyperparameters include learning rate, training steps, batch size etc.

7. **Prediction:** Using results from the testing data to have an idea of how the model will perform in the real world or a real-life scenario.

2.2 Machine Learning Algorithms Used in Autonomous Vehicles

The following are popular machine learning algorithms used in autonomous vehicles such as self-driving cars, ships etc.

1. **Decision Matrix Algorithm**

Decision matrix algorithm identifies and systematically analyzes the rate of performance between information and values set. This algorithm is mainly used to aid decision making, as the models independently train themselves to make predictions while reducing the probability of errors in decision making. The most popularly used decision matrix is AdaBoost [24].

2. **Clustering Algorithm**

The clustering algorithm works by locating and detecting objects in images that are usually not visible by the system [25]. This algorithm partitions an image data set into several disjoint groups or clusters, represented as a set of datapoints which are further used to classify the individual points into a specific group [26].

3. **Pattern Recognition**

The images obtained by the system using the Advanced Driver Assistance Systems sensors usually consist of data from all kinds of environment. This algorithm assists in filtering images to determine the instance of an object by first ruling out the data points that are not important [24]. The pattern recognition plays a major role in dataset since before objects can be classified, they must be recognized. In a lot of ways, the features

of the image (line segments, circular arcs) are combined to form the features that are utilized for recognizing an object.

4. Regression Algorithm

This algorithm is great for predicting events. The relation between two or more variables are evaluated by the algorithm and the effects are collated on distinct scales and are driven by the following metrics:

- Shape of regression line
- Type of dependent variable
- Number of independent variables

Images play an important role in the Advanced Driver Assistance Systems for actuation. The biggest challenge of any algorithm is to create an image-based machine learning model for prediction and feature selection [27].

Regression algorithms are generally used for long learning and short prediction. Examples of regression algorithms used in autonomous cars are neural network regression, decision forest regression and Bayesian regression.

2.3 Convolutional Neural Network

Convolutional neural network (CNN) is a class of artificial neural network that has become dominant in the field of computer vision and its applications. This network model works best with two-dimensional image data, although it can still be used with one- and three-dimensional image data. The only significant difference between CNNs and ANNs is that the former is used mainly within images around pattern recognition.

The central layer of the model where CNN got its name from performs an operation called "*convolution*". This operation extracts features by applying a convolutional filter on a raw/ feature image.

CNN is designed for processing of data which comes in shape of high dimensional array, for instance pixels of an image. The key components that outline the performance of convolutional neural network are:

- Local connection
- Shared weights
- Pooling
- Use of multiple layers

CNN automatically adapts to learn spatial features hierarchy through backpropagation by utilizing building block such as the convolutional layer, pooling layer, and the fully connected layers.

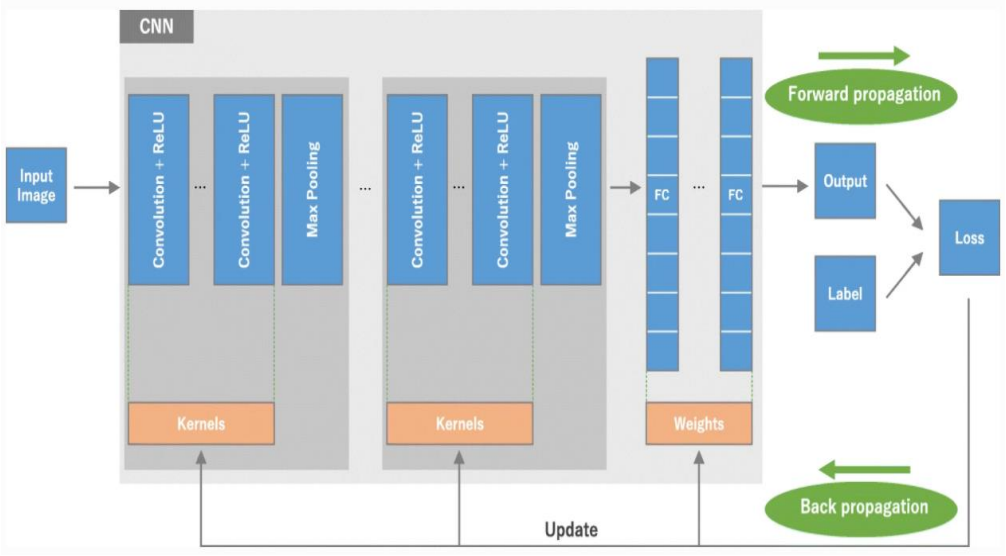


Figure 2.2 *An overview of CNN training process and architecture [28].*

The performance of the model in terms of weights and kernels can be calculated using loss function through forward propagation on the training dataset, while adjusting learning parameters in accordance to loss function through backpropagation

2.4 Related Work

Over the last several decades, maritime research in object detection has been developed particularly in unmanned surface vehicles operations. Information on vessels in the sea are usually obtained from systems like automatic identification system (AIS). However, this system is not available on all ships, which is why additional sensing methods such as navigation sensors, radars and so many more are required.

Han et al. [29] proposed algorithms for target detection and tracking by sensor fusion in their framework while benefitting from its applications in navigation and collision avoidance systems for autonomous vehicles. The proposed target tracking and autonomous avoidance algorithm was based on the fusion of detection sensors such as lidar, radar, infrared (IR) and electro-optical (EO) cameras to achieve a reliable detection and continual target tracking. The algorithm was applied in a USV system called “Aragon”, and the data from both the detection and the tracking combined was useful for the obstacle information systems used in collision avoidance. These sets of information also play a significant role in the collision avoidance maneuvering design.

The most successful detection algorithm in their study was the constant false alarm rate (CFAR), due to the reason that it could detect objects from the measurement of a sensor by minimizing any noise caused by factors such as windy/ rough sea, wave, sunlight, rain or fog on the water surface. Therefore, cell averaging constant false alarm rate (CA-CFAR) was utilized by the radar to fix any issues that may be related to noises. The experimental

results showed that the two active sensors lidar and radar provided more reliable performance when it came to detection in the marine environment in comparison to the cameras used as passive sensors. Their more reliable detection can be associated with the tuning of various hardware settings and navigation algorithms depending on the conditions for detecting targeted small-size vessels.

With the advancement of machine learning, some studies have suggested the use of deep reinforcement learning (DRL) for autonomous path planning of unmanned ships in new or unknown environments. Guo et al. [30] proposed a deep deterministic policy gradient (DDPG) algorithm-based model which utilizes the interaction between the historical experience data and the environment, to train the agent on the optimal action strategy. Their research was motivated by the work of Zhu et al. [31] who proposed a car-following framework for autonomous cars using DDPG. The autonomous cars in their framework learned the model from the environment based on a trial-and-error approach. Their experiments obtained good results and revealed that DDPG can understand the driver's behavior and in turn assist in developing human-like autopilot algorithms and traffic flow models.

Guo et al. [30] utilized certain knowledge of the DDPG framework proposed by Zhu et al. [31] to suggest a path planning model for unmanned ships based on the DRL method that helps agents find the most efficient path independently. They combined the artificial potential field (APF) and DDPG to obtain an APF-DDPG autonomous path planning model with faster convergence and higher decision-making power. The traditional path planning algorithm resulted in low accuracy and redundancy because the historical experience data did not play a role in the training and improving the learning

efficiency of the algorithm, in contrast to the APF-DDPG based algorithm. The experimental results show that unmanned ships make reasonable actions in an unfamiliar environment, and it also highlights an improved DRL method over the classical one, which can be attributed to the combination of APF and DDPG.

2.4.1 The Role of CNN in Autonomous Collision Avoidance

In recent times, A deep neural network has proven to be very successful in many classification tasks is the Convolutional Neural Network [32]. The CNN have become very common to use in computer vision problems and applications [33], and they also give good results in various benchmarks for classification tasks [34]. In the paper by Sharif Razavian et al. [35] they experimented with a convolutional neural network for different recognition tasks, and came to the conclusion that, deep learning alongside CNN are essential as primary candidates in any visual or object recognition task.

A convolution neural is a special kind of linear operation. The architecture of a convolutional neural network is structured as a series of stages, whereas each stage consists of convolution and pooling layers as well as non-linearities [36], another motivation is that the convolutional neural network is a type of a deep fully connected network, which is easier to train and generalize better than a network with only fully connected layers [33].

The three-mechanisms associated with convolutional neural network are local receptive fields, weight sharing and subsampling. Each layer of a convolutional neuron is connected only to a local region of the input.

In contrast to a fully connected layer where each neuron is connected to each neuron in the previous layer. It is this region that is called the local receptive field.

The receptive field enables the neurons to extract features such as edges and corners in an image. In a convolutional layer, the neurons are organized in planes, which are called feature maps. The neurons that belongs to the same feature map are sharing weights. If we think of the neurons as feature extractors, all neurons in the same feature map will detect.

2.4.2 Relationship Between Human Cognitive Abilities and Ship Collision Avoidance Algorithms

Collision avoidance is a one of the major issues that mariners face. Therefore, the success of an autonomous ship navigation depends on the development of an efficient and effective real-time intelligent algorithms for collision avoidance. Most algorithms attempt to imitate the mariner's cognitive abilities [37]. To better understand the relationship and differences between the captain in the ship and the intelligent navigation system, the following human operations that will be performed for the collision avoidance process should be put into consideration to make it easier to understand the factors that influence collision avoidance in ships:

Ship Type: Different types of ships vary in their maneuvers for collision avoidance, therefore for every ship type there's a special training the captain and crew go through. The CA autonomous navigation algorithms ''understand'' ship type as ''ship mathematical model''. The ship mathematical models provide predictions of the ships behavior based on inertial but are often very simple and result to erroneous predictions, or predictions that's are complex to run in real time.

Traffic Categories: With the different forms of traffic that happens on the sea, a vessel can collide with an obstacle or another vessel. Therefore, traffic is divided into two types:

1) Traffic confined within an environment like canals, port [38].
2) Traffic in open waterways

Both types of traffic leaves room for additional complexity that may be caused by under-surface environment like wrecks, seabed level, animals, buoys as well as static and dynamic sea obstacles.

Weather: The effect of weather is different on each type of ship. Which means that different weather conditions require different maneuvers techniques from the pilot crew. The most important aspect is combining safety and collision avoidance concurrently. In most autonomous navigation algorithms, the sea weather conditions are rarely considered.

Navigation Technology: Algorithms for ship collision are generally considered to be equipped with GPS and Automatic Radar Plotting Aid (ARPA). Although, none of the algorithms put into consideration the navigational data, instrumentation, and communication for collision avoidance that modern ships now have available.

All the above factors discussed above require human intervention and operation for a collision free navigation. The operations are subjective to every personnel captain & crew based on their own cognitive abilities and training. Consequently, the investigation of human cognitive demands for collision avoidance is useful.

2.4.3 Soft Computing technique for Ship Collision Avoidance

A hybrid system for collision avoidance and track-keeping was proposed by Hwang et al., 2001 [39]. The systems combine expert systems, fuzzy logic, state, and collision avoidance was carried out by the fuzzy expert system. It was designed to utilize the knowledge base of facts and rules with the aid of an inference engine. The inference engine was responsible for the simulation of the expert system decisions for ship collision avoidance.

2.5 RetinaNet

RetinaNet was formed using a combination of ResNet + Feature Pyramid Network (FPN), bounding box regression and subnetworks. Resnet and FPN provide the fundamental structure for feature extraction, while the subnetworks conduct two specific tasks which are bounding box regression and classifications.

2.5.1 Design of RetinaNet

The design of RetinaNet shares similarities with other one-stage detectors by the introduction of 'anchor' concept introduced by Regional Proposal Network (RPN) [40] and use of feature pyramids introduced by Single Shot Detector (SSD) [6]and Feature Pyramid Network (FPN) [41].

RetinaNet outperforms many typical two-stage detectors such as Faster R-CNN due to its state-of-the-art performance. Even as a one-stage detector, it has achieved comparable performance as two-stage detectors. Two stage detectors are usually applied to a smaller set of object location data whereas with one-stage detector produce simpler and faster output by utilizing a larger set of object location data. We need to mention that two-stage

frameworks constantly achieve the highest accuracy on the COCO benchmark which is a challenging feat [18].

Results have shown that when RetinaNet is trained with focal loss it exceeds the accuracies of two-stage detectors and has a similar speed as previous one-stage detectors [3].

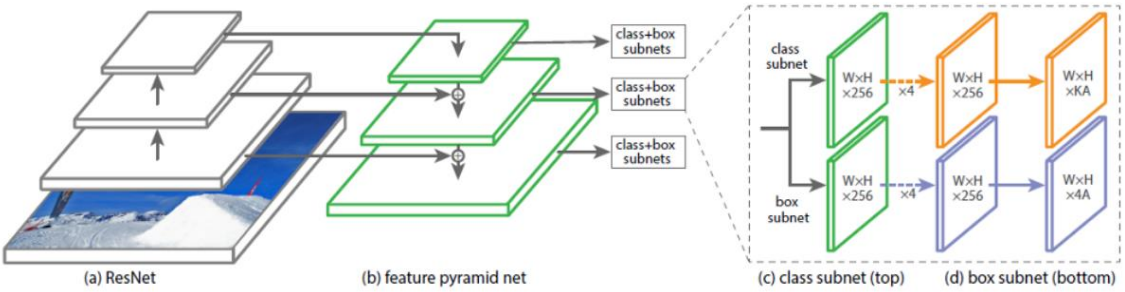


Figure 2.3 *RetinaNet detector network Architecture showing ResNet, FPN and both subNets [7]*

2.5.2 ResNet

Residual Network also known as ResNet, as demonstrated in Figure 2.6, was proposed by He et al. in 2015 [42]. ResNet was extended the year after and began to follow an ethical technique to include shortcut connections for every two layers of the VGG-style network [43]. In fact, ResNet was not the first network to utilize and implement shortcut connections: for instance, Highway Network established gated shortcut connection first [25]. ResNet outstandingly reduces the complexity of training using shortcut connections [44] and performs well with regards to generalization error.

There are three types of shortcut connections. Shortcut Type 1 handles numbers that go as high as infinity when the number of layers increases or grows, and this type of shortcut is similar to having no shortcuts. When stacked with additional layers of residual

blocks, shortcut Type 2 results in smaller training error, unlike shortcut Type 1 or 3 which does not result in a smaller training error [42].

2.5.3 Feature Pyramid Network (FPN)

Feature pyramid network builds on image pyramid, which constitutes of top-down and bottom-up pathways. The top-down pathway creates higher resolution layers from the semantic layer [26], where the higher resolution is unsampled spatially but semantically stronger [45]. The bottom-up pathway utilizes ResNet to build the bottom-up pathway. It consists of convolutional modules (also known as *convi*, where i equals to 1 to 5) with each module having many layers. Moving up from the bottom- up pathway, spatial dimension decreases by half which results in doubling of the strides [26].

Feature pyramids are scale-invariant which means that a change in scale of an object counteracts by adjusting and changing its level in the pyramid. Additionally, the pyramid model enables the model to detect object irrespective of their range of scale.

2.6 Class Imbalance

Class Imbalance causes two major problems when it comes to training. The first problem talks about inefficient training that contribute to no useful learning signal due to easy negatives from most locations. The second problem can be attributed to easy negatives which overwhelm training and can lead to degeneration of models. The conventional solution to this problem is to introduce a negative mining processes [25] that samples more complex reweighing [46] .

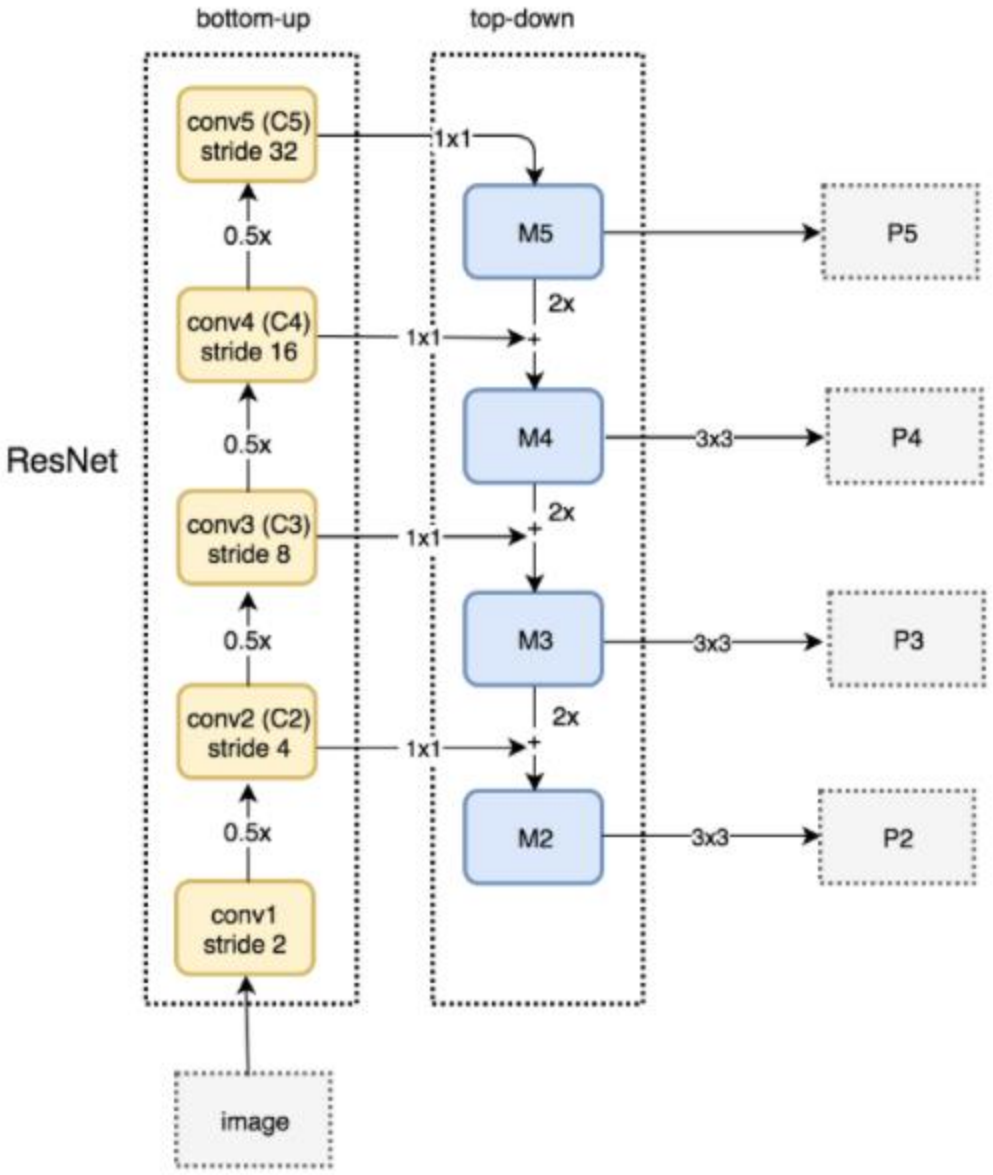


Figure 2.4 *Top-down and bottom-up pathways architecture in ResNet [26]*

Although in their study, [7] proposed that focal loss conventionally handles any case of imbalance that may come up using a one-stage detector. This allows for efficient training without sampling and cancels out all easy negatives from overwhelming loss and computed gradients

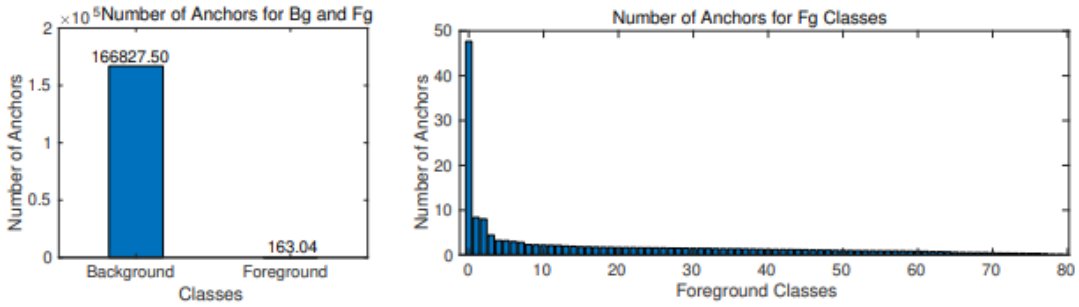


Figure 2.5 *RetinaNet anchors plotted against MS-COCO highlighting class imbalance in some classes. (Left) Imbalance between Background and Foreground, (Right) Imbalance between foreground classes [2].*

2.7 Focal Loss

The major aim and importance of focal loss is to address the high rate of imbalance between foreground and background classes of one-stage object detection scenarios while training. Cross entropy was introduced for binary classification, focal loss step-by-step equation is devised below [47].

$$CE\ (p, y) = \begin{cases} -\log(p)\,, & y = 1 \\ -\log(1-p)\,, & otherwise \end{cases} \tag{1}$$

$$p_t = \begin{cases} p, & y = 1 \\ 1 - p\,, & otherwise \end{cases} \tag{2}$$

$$\text{CE (p, y)} = \text{CE } (p_t) = -log\ (p_t) \tag{3}$$

$$\alpha_t = \begin{cases} \alpha, & y = 1 \\ 1 - \alpha\,, & otherwise \end{cases} \tag{4}$$

α_t was added to the Equation (3) to fix the class imbalance problem.

$$\text{CE } (p_t) = -\alpha_t * -log(p_t) \tag{5}$$

Equation (5) handles only the weight of positive and negative samples and does not consider hard examples.

$$FL\ (p_t) = -\ (1 - p_t)^{\gamma}\ log(p_t) \quad (6)$$

Equation (6) represents the non-alpha form of focal loss equation.

$$FL\ (p_t) = -\ \alpha t\ \ (1 - p_t)^{\gamma}\ log(p_t) \quad (7)$$

Equation (7) represents the alpha form of focal loss equation.

CHAPTER III - METHODOLOGY

In this brief chapter an introduction to the process employed in this work is highlighted. Details are provided from data curation dataset that was used for this thesis and discussion of results in the following chapters.

One of the main aims of this thesis involves selecting and applying combination of augmentation techniques that will help increase the mAP scores and object classification accuracy of the RetinaNet algorithm on our SID.

Several combinations of augmentations were iterated until a combination with the highest mAP score and object classification accuracy was obtained.

3.1 Design of the Sea-object Image Dataset (SID)

For a seamless training, the dataset has to meet some specifications like including an annotation of the images, with data files listing the labelled objects and the positions of their bounding box (x_{min}, y_{min}, x_{max}, y_{max}) in the images. The images in our dataset were resized to have the same aspect ratio for easy differentiation of fixated locations.

In addition to our SID, Microsoft COCO was used because the large number classes available in the images are helpful for the pre-training of our object detection model. The categories of our interest that are already covered in the COCO dataset include: boats, people, and birds.

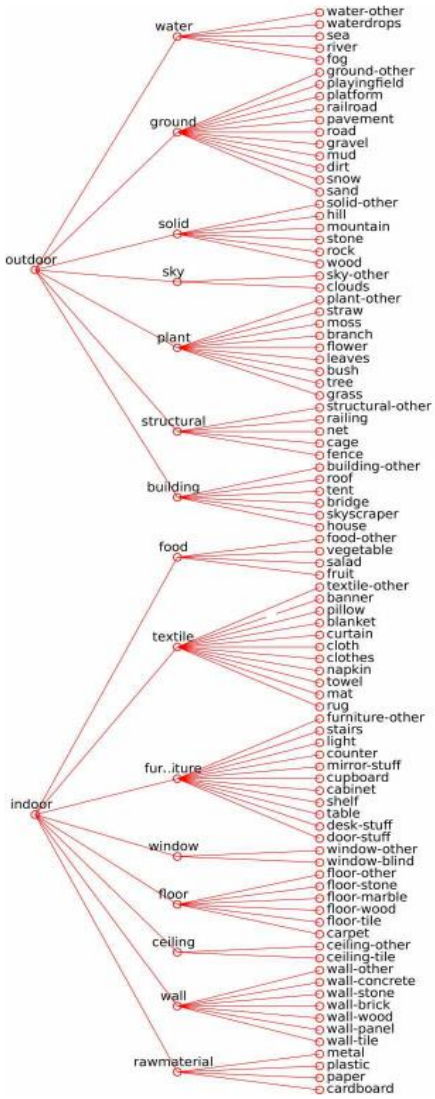


Figure 3.1 *Label hierarchy showing classes from COCO-Stuff, divided into outdoor and indoor classes. [48]*

3.1.2 Class Selections of the Sea-object Image Dataset (SID)

A brief study on the non-trivial objects around/in the sea was conducted first. The names of this type of objects consist the following: trees, rocks, clouds, apartment buildings, other buildings, boats, buoys, waves, persons, ships, birds, sea lions, seals, islands, piers, sharks, whales, ocean sunfish, oarfish, sun, and lighthouses.

After collecting the list of classes, an evaluation was performed next to see which classes were insignificant w.r.t ship-related collisions and which classes could be merged into one. Trees, cloud, birds, sea lions and seals were removed from the list as they have little to no impact on collision

All the various types of fish were merged into a class called fish, while apartment buildings and other buildings were also combined together to form a class named building. It repeats the figure's content. Keeping only one is enough.

A final list was ultimately formed by only selecting those classes that are susceptible to cause a significant damage if a collision occurred between the corresponding objects and the ASV. Figure 3.2 highlights the ten classes that were selected for SID dataset.

Figure 3.2 *Ten classes of the Sea-object Image dataset (SID)*

3.1.3 Data Scrapping

A Web crawler was used to find and import images from the Internet to create SID. Since data scrapping remains one of the major sources of data for machine learning, it was exploited to get relevant images for the dataset.

The Web crawler used was written in Python and had the capability to search across various websites on the Internet and download images once a keyword has been inputted into the code.

3.1.4 Data Cleaning

After the data scraping was carried out to collect the images, a data cleaning procedure was implemented to ensure that they are ready for use, consistent, and correct. This stage is the most time consuming and complicated as there is no specific technique that fits all types of the data. Therefore, depending on the class of the images obtained and their source, the cleaning process was different.

A huge part of this process involved identifying and deleting corrupted images or file types, duplicates, and low-resolution images. Other processes in this step include: removing extra spaces in the names and spelling check.

3.1.5 Labelling

Microsoft Visual Object Tagging Tool (VoTT) was used to label the images for the Sea-object Dataset.VoTT is an open-source image annotation and labeling app written in TypeScript and used for labelling images and video assets. It allows importing of data from your local or cloud storage and exporting of labelled images back into your local or cloud storagedepending on your choice. It facilitates end-to-end machine learning pipeline

and building of object detection models from labelled images or videos.

This application can export labelled dataset in any format of your choice. The Sea-object Dataset was exported in CSV format, although one can export to Azure custom vision service, Microsoft cognitive Toolkit (CNTK), Pascal VOC and TensorFlow Records.

Figure 3.3 *Labelling process of a fish in VoTT for the Sea-object Dataset*

Figure 3.4 *Labelling process of a ship in VoTT for the Sea-object Dataset*

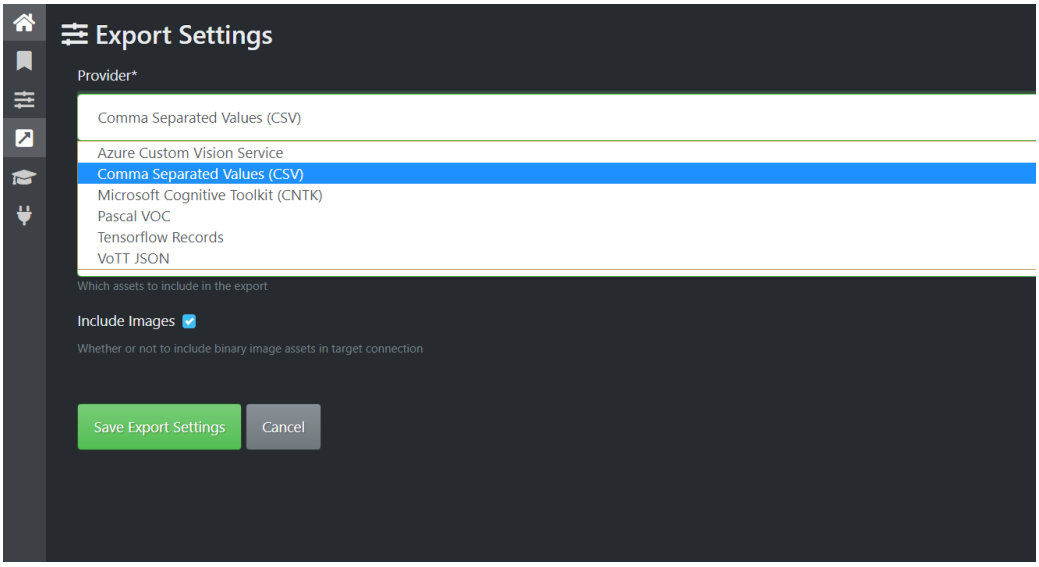


Figure 3.5 *Export settings available in Microsoft's VoTT.*

After the labelling was completed and the annotation CSV file exported, the annotation file also went through data scrubbing, cleaning and manipulation using a Python data analysis library called pandas. Some of the involved processes include normalizing data points that didn't look right, validating data accuracy, scrubbing for duplicates, identifying, and deleting columns or rows with single values or null cells.

3.1.6 Training

For training, the model uses the four points of the bounding box (x_{min}, y_{min}, x_{max},y_{max}) from the annotation csv file to train. For the pre-training on the COCO dataset, the input images were of different sizes and the default RetinaNet aspect ratio of 800 x 1333 was not altered for this round of training.

For the fine-tuning of the pre-trained model based on our SID, the input images were resized to the 512 x 512 aspect ratio to ensure uniformity of the dataset and most

importantly uniformity of the of the anchor boxes. To improve the performance of our model, we also performed data augmentation before the fine-tuning.

3.1.7 Variation of IoU Threshold

Intersection of union (IoU) is a metric used in machine learning to measure how accurate the predicted bounding box made by the model with respect to the ground-truth bounding box [52]. The IoU threshold refers to the value that is computed with the presumption that scores lower than this value cannot be detected by the model.

After every training, the IoU threshold was changed from the default value of 0.5 to 1. This variation was used to determine the best prediction performance after training. It was observed that different thresholds had a non-trivial effect on the accuracy of the prediction.

3.2 Evaluation Methodology

To evaluate the object detector, two different test datasets were used. The first test dataset evaluates its performance without data augmentation, while the second test dataset specifically evaluates the impact of various data augmentation methods.

Although there are many machine learning algorithm evaluation metrics, we focused mainly on the accuracy evaluation metric.

3.3 Tools and Environment Used

3.3.1 Google Colaboratory

Google Colaboratory also known as “Colab” is product that allows users write and run python codes through their browsers just like Jupyter notebook. The major difference between Colab and Jupyter notebook is that the latter is open source while the other is not.

Colab is a cloud service that provides GPU to its users. It also enables users to develop deep learning, machine learning and data analysis projects and applications implementing libraries such as OpenCV, PyTorch, TensorFlow, Keras with access to HIGH-RAM runtime.

3.3.2 Anaconda

Anaconda collection of packages distributed for data science and machine learning projects. It is free, open source and comes with a package, an environment manager and conda. Each of these play a vital role in creating a project.

The package consists of python, conda and over 150 packages libraries and their corresponding dependencies, which makes it easier when creating an environment for new projects. Conda assist in creating environments with different libraries, while the environment manager helps in modifying or handling various environments created a user.

3.3.3 Pandas

Anaconda collection of packages distributed for data science and machine learning projects. It is free, open source and comes with a package, an environment manager and conda. Each of these play a vital role in creating a project.

The package consists of python, conda and over 150 packages/ libraries and their corresponding dependencies, which makes it easier when creating an environment for new projects. Conda assist in creating environments with different libraries, while the environment manager helps in modifying or handling various environments created a user.

3.3.4 Keras

Keras is another one the powerful python libraries that handle deep learning models. It develops and evaluates deep learning models and allows for definition and training of neural network models.

3.3.5 NumPy

NumPy is one of the popular and powerful python libraries in the package, this is because it handles the multi-dimensional arrays and data structures. It goes beyond that as it is also used to perform some mathematical operations.

3.3.6 TensorFlow

TensorFlow serves as foundation to some libraries by wrapping those libraries on top of itself to simplify the process. It supports both GPU and CPU and offers API that makes some machine learning processes easier.

3.3.7 Libraries

The following are the libraries installed and used in the environments and their respective versions

- Cython
- Keras-resnet ==0.1.0
- H5py
- Keras
- Matplotlip
- Numpy>=1.14
- Opencv-python>=3.3.0

- Pillow
- Progressbar2
- Tensorflow

3.4 System Configuration

The system used to carry out this research had the following configuration:

- Tesla P100-PCIE-16GB GPU
- Intel(R)Xeon(R) CPU @ 2.30 GHz
- 26 GB RAM
- Python 3.7
- 200 GB Disk Space

3.5 Implementation Details

Our research was carried out with Keras library and was then built on top TensorFlow with a 16GB Tesla P100-PCIE GPU. Several settings were adjusted to get the best version of RetinaNet for our model.

3.6 Definition of Terms

To understand and interpret the results shown and discussed in chapter IV, some important terms would be defined to give a clearer picture of their meaning and why there are important.

- **Ground -truth**

  Ground-truth refers to the bounding boxes of training and testing dataset respectively which are hand labeled.

- **Prediction**

  The output of a model when an input has been provided.

- **True Positive (TP)**

  A true positive result occurs when the prediction which is said to be positive is true.

- **False Positive (FP)**

  A false positive result occurs when the prediction which is said to be positive is false, meaning the model did not predict an object that was in the image.

- **True Negative**

  A true negative occurs when the prediction which is said to be negative is true.

- **False Negative**

  A true negative occurs when the prediction which is said to be negative is true.

- **Training Set**

  A part of the dataset used for training the object detector.

- **Accuracy**

  Accuracy is the proportion of correct predictions a model makes to the number of training examples. Accuracy is defined differently in multi-class classification from how it is defined in binary classification. In multi-class classification it is defined as**,**

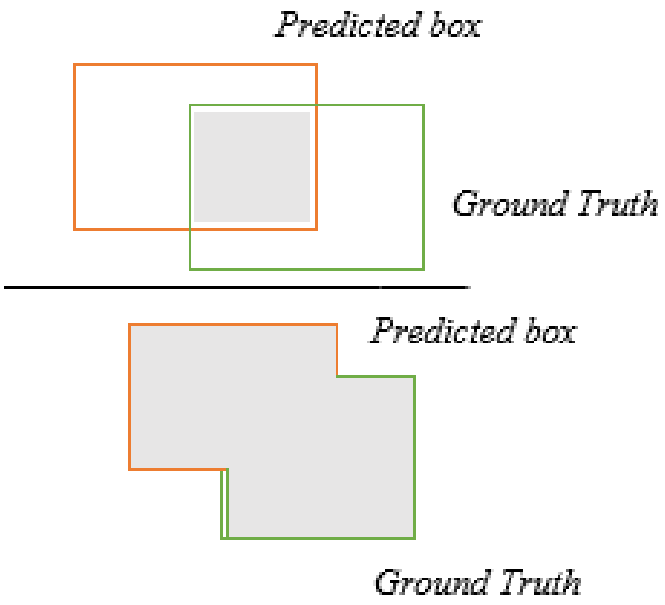
$$\textbf{Accuracy} = \frac{Correct\ Predictions}{Total\ Number\ of\ Examples}$$

- **Intersection over Union**

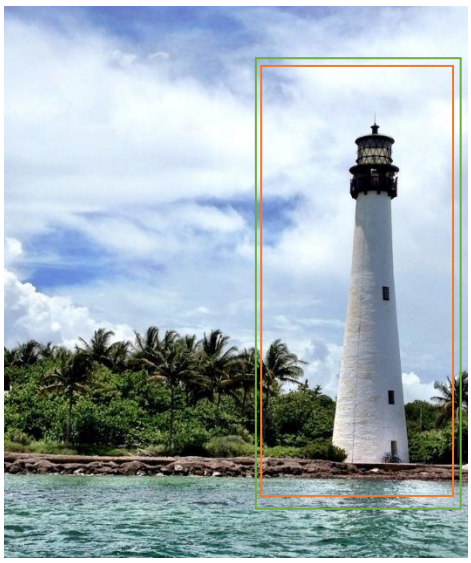
  In machine learning, object detection tasks utilize IoU as a metric to evaluate models. It measures the accuracy of models to predict bounding boxes with respect

to ground-truth of the bounding box. Any algorithm that predicts bounding boxes as output can easily be evaluated using IoU [49]. The IoU for the predicted bounding box and the ground-truth bounding box is the ratio between the total area and the overlapping area. The value of IoU ranges from 0 to 1, with 0 indicating no overlap and 1 indicating that the predicted bounding box and ground- truth precisely have the same coordinates.

$$\textbf{Intersection over Union} = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

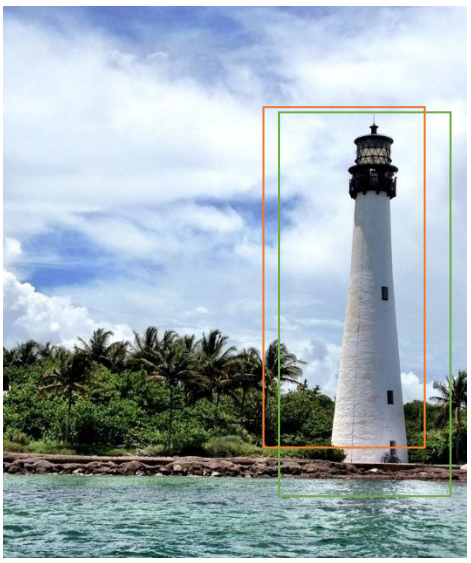


Below are examples of sample IoU scores and their respective bounding boxes. The green bounding box represents the ground truth, while the red bounding box represents the predicted bounding box.



IoU: 0.9

Condition: Excellent

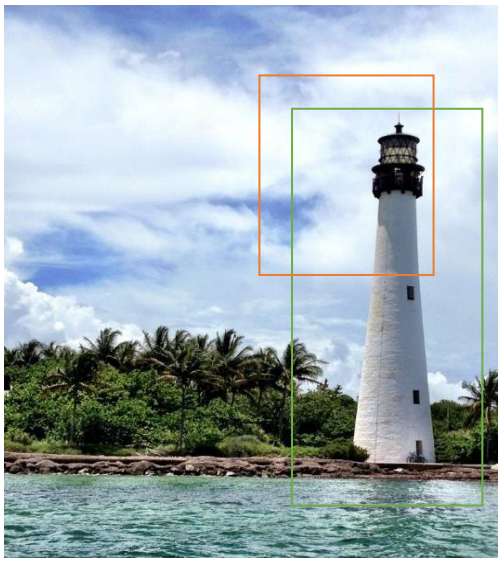


IoU: 0.7

Condition: Good

IoU: 0.4

Condition: Poor



IoU: 0.0

Condition: Very Poor

Figure 3.6 *Sample IoU scores and corresponding bounding box for each score*

- **Recall**

  Recall calculates how the model correctly identifies the positives.

$$\text{Recall} = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative\ (FN)}$$

- **Precision**

  Precision is used to measure classification models; it calculates this by identifying the frequency in which the model was able to predict the positive class correctly.

$$\text{Precision} = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Positive\ (FP)}$$

- **Average Precision (AP)**

  Average precision is a metric used in calculating the accuracy of object detectors by taking the average of the precision value for each result. The results are ranked where there's an increase in recall compared to the previous result [52].

$$\text{AP} = \sum_{k=0}^{k=n-1} [\ \text{Recalls(k)} - Recalls\ (k+1)] * precisions(k)$$

  Where Recalls (n) = 0

  Precisions (n) = 1

  Number of

  thresholds = n

- **Mean Average Precision (mAP)**

  Mean average precision is the average of average precision (AP) which is measured by dividing the mean AP with the IOU threshold.

$$\text{mAP} = \sum_{k=0}^{k=n-1} \text{AP}_k$$

Where AP_K = the AP of class k

n = number of classes

- **Inference**

  Inference is the operation that occurs when the trained model is applied to unlabeled examples to make predictions

- **Batch size**

  Batch size refers to the number of training examples which is fixed during training and utilized in one iteration.

- **Epoch**

  An epoch refers to the number of passes over the entire dataset. This simply means that all training examples in the dataset has been seen once.

CHAPTER IV – RESULTS AND ANALYSIS

In this chapter, presentation of results from the implementation of the methodology shown in Chapter III, to analysis of the results is also discussed below.

4.1 Distribution of Training Data

Table 4.1 *Training data distribution*

Class	**Number of labels**
Person	808
Building	674
Lighthouse	640
Buoy	554
Rock	386
Pier	340
Ship	322
Wave	300
Fish	296
Island	137

The table above shows the distribution of classes of our dataset, across all our classes. Smaller numbers in certain classes can be associated with lack of high-resolution images of those classes.

4.2 Model Performance

In this section, the experiment workflow will be discussed followed by the performance of each model.

4.2.1 Model Training and Experiment Workflow

Six variational models of RetinaNet were trained and evaluated for this study, which is explained below by the procedure employed.

1. **Base Model** - Base model is trained without any modification to the raw training data.
2. **Base Model + Unlabeled Images** - Includes the initial images and 20% more unlabeled images. This was done to give the model clear examples of classes that it should not recognize, as it has already been trained with labels of classes and objects that should be detected. The goal is to show the model objects that should be ignored. This can potentially limit the number of false positives reported.
3. **Base Model + Augmented Images** - Focuses on trying out various augmentation methods on the raw training set. Methods include Sharpen, Blur, Flip and Rain.
4. **Base Model + Resized -** All input images were resized to 512 x 512.
5. **Base Model + Resized + Unlabeled Images + Augmentation –** Model trained of resized original images, unlabeled images followed by augmentation.
6. **Base Model + Resized + Unlabeled Images** - Same as the fifth case but without augmentation.

The diagram below shows the experiment workflow for the training of the six variational models.

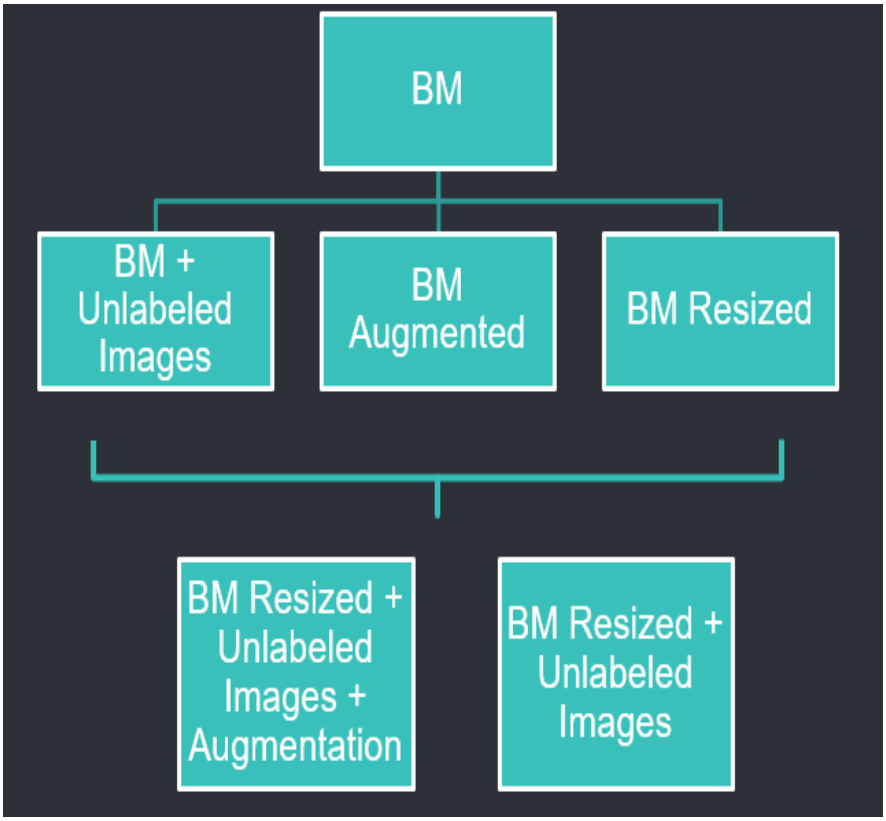


Figure 4.1 *Workflow of model training and experiments*

4.2.2 Class-level Performance of the Six-model Variation

In this section we evaluate the class-level performance, in terms of AP, for each of the six-model variation.

From the base model, which is the model variation without any modification, using the dataset just as it is. It can be noticed that classes such as rock, wave, and buoy are doing poorly i.e. below average. Although more classes were above average compared to those that did poorly after the base training.

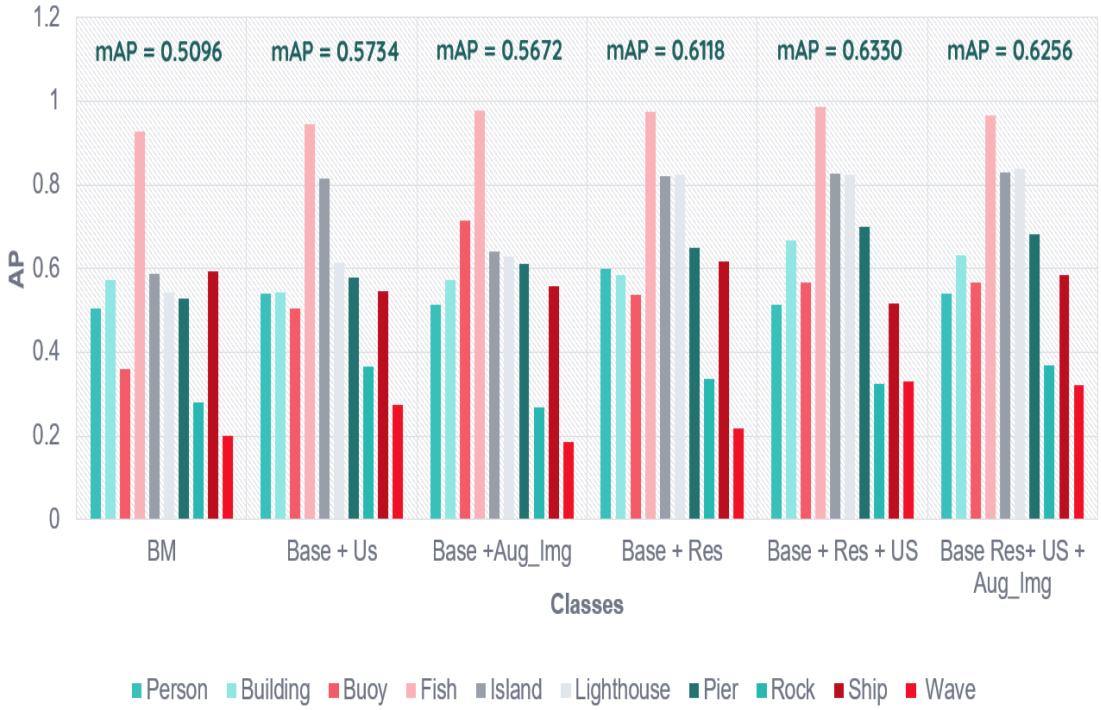


Figure 4.2 *Results of model training showing AP of the ten classes and overall mAP from the six-variational models*

From the Base Model + Unlabeled sample training, the AP of Buoy, Wave, Rocks Island, Lighthouse and Pier noticeably improved, while person and building reduced by a few points. The overall performance mAP of this training went from 0.5069 of the previous model variation (Base Model) to 0.5735 for this Base Model + Unlabeled sample trained model.

From the Base Model + Augmentation Sample training, the images were augmented using four main augmentation method which are sharpen, blur, flip and rain. With the four methods of augmentation applied to this training model, buoy and building were the classes with noticeable change in AP, and this shows in the overall performance Map which resulted to 0.5672, compared to the previous training model (Base Model + Unlabeled sample training) which resulted in an overall performance of 0.5735.

From the Base Model + Resize Sample training, Building, Island, Lighthouse, Pier, Rock, Ship and wave made significant improvement in their respective APs which significantly brought about an increase in overall Map of 0.5672 in the last model training (Base Model + Augmentation Sample) to 0.6118 in this current trained model.

Results from the Base Model + Resize + Unlabeled Sample training brought about an increase in the AP of most of the classes and this in return increased the overall performance to 0.6256.

Results from the Base Model + Resize + Unlabeled + Augmentation Sample training improved the overall performance from 0.6256 of the previous variation of the model (Base Model + Resize + Unlabeled Sample) to 0.6330, while increase of class AP is not significantly noticeable.

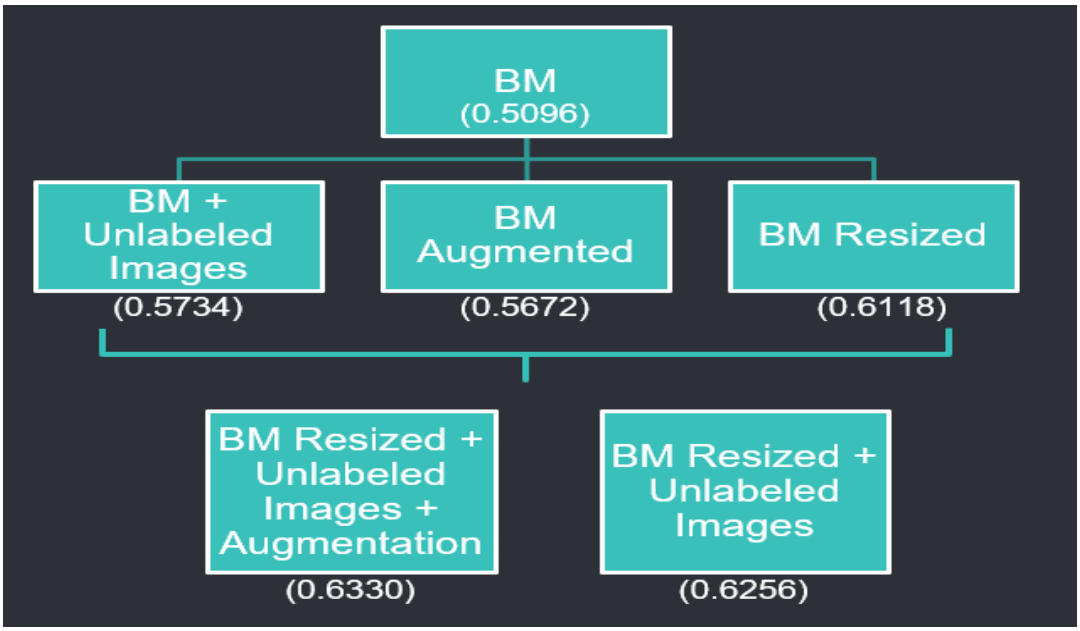


Figure 4.3 *Workflow of model training and experiment highlighting overall mAP of each Model.*

4.3 Learning Curve

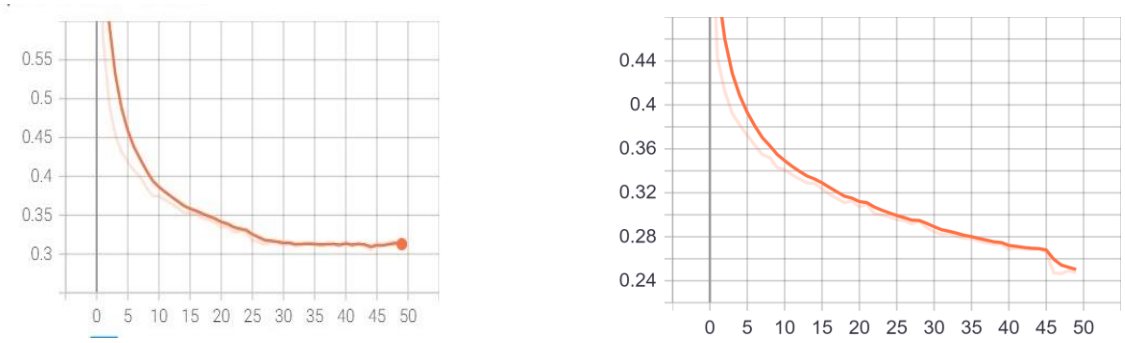


Figure 4.4 *Learning curve of the Base model versus the best performing model*

Loss function is one of the ways that a machine learning algorithm optimizes. It evaluates how an algorithm models the data that it was trained with [50]. If the prediction and the results after training differ from each other, then the loss function would end up with a relatively larger number.

Figure 4.9 presents the classification loss of the Base Model (BM) and that of BM Resized + Unlabeled Images + Augmentation model variation. The learning curve compares the learning curve of the Base model, which resulted in a classification loss of about 0.3 compared to that of the best model that further reduced the loss of 0.26.

Although 0.26 is not an ideal classification loss score, it is closer to 0 which indicates why the BM Resized + Unlabeled Images + Augmentation model variation, performs better in detection and in overall mAP. It shows that the loss function is proficient in decreasing error in predictions with the help of optimization function.

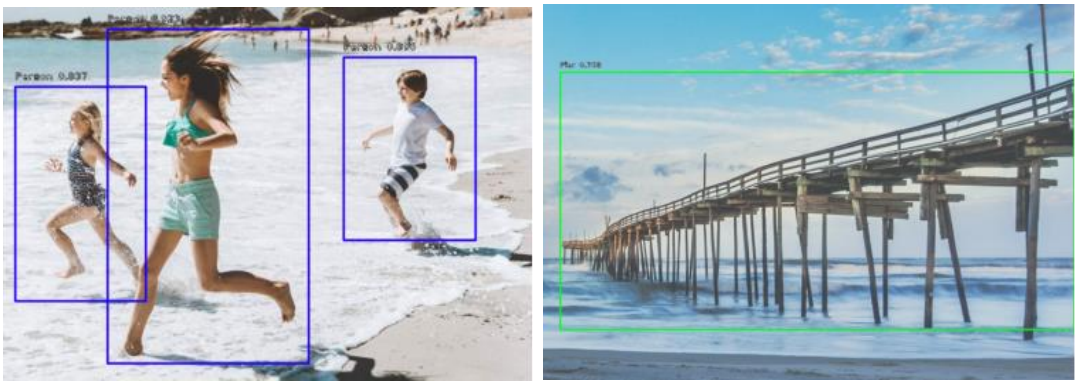
4.4 Detection from the Automatic Detector for Autonomous Ships

Figure 4.5 *An example rock object detected by the best model variation.*

Figure 4.6 *An example ship object detected by the best model variation.*

(a) (b)

Figure 4.7 *An example (a) fish and (b) island object detected by the best model variation.*



(a) (b)

Figure 4.8 *An example (a) person and (b) pier object detected by the best model variation*

(a) (b)

Figure 4.9 *An example (a) buoy and (b) lighthouse and building object detected by the best model variation.*

The results above from Figure 4.5 to 4.9 show various cases where the model trained on the Sea-object Image Dataset detects some of the classes. In Figure 4.10, the object ‘Rock’ is detected at an IoU score of 0.709, which translates to a 70% overlap between the predicted bounding box of the model and the ground-truth box. It is a really good detection and means that the two boxes overlap.

In Figure 4.6, a Ship is detected at sea with an IoU score of 0.721, which translates to a 72% overlap between the predicted bounding box of the model and the ground-truth box. It is also a really good detection, with a significant overlap of both boxes.

In Figure 4.7, a Fish was detected diving out of the sea at an IoU score of 0.811 in image (a), which translates to an 81% overlap between the predicted bounding box of the model and the ground-truth box. It signifies an extremely good detection from the model as both boxes are more aligned within this IoU score range. An Island is detected in image (b) with an IoU score of 0.517, which translates to a 51% overlap between the predicted bounding box of the model and the ground-truth box. This detection can be categorized as 'less good' than the previous detections, this is because generally IoU scores above 0.5 are considered an average/good prediction.

In Figure 4.8, image (a) detects three persons with the respective IoU scores- 0.837,0.923, 0.895, which translates to an 83%, 92% and 89% overlap between the predicted bounding box of the model and the ground-truth box. These are all excellent detections, with an equally as good overlap of both boxes. In image (b), A Pier is detected with an IoU score of 0.758, which translates to a 75% overlap between the predicted bounding box of the model and the ground-truth box, also signifying a good detection like the examples above.

In the last example, Figure 4.9 detects a Buoy in image (a) at an IoU score of 0.841, which translates to an 84% overlap between the predicted bounding box of the model and the ground-truth box. Image (b) detects both a Lighthouse at an IoU score of 0.841 and 0.538 for Building, which translates to an 84% and 53% overlap between the predicted bounding box of the model and the ground-truth box, which reflects a really good detection for lighthouse and an average good detection for building.

4.5 Discussion and Conclusion

Augmentation provides improvement in model performance **(11.31%)** but at a great cost, often an increase directly proportional to the magnitude of augmented data. Although, this cost is not a factor when scaling. This is because compute resources will be available to handle the task that comes with this method.

The introduction of unlabeled images presented a more noticeable increase in performance **(12.53%)** without an increase in cost in comparison to augmentation.

Resizing all input images to the same size appears to have a positive effect on the performance of the model. This can be tied to the importance of anchor boxes in object detection models.

A combination of all stated techniques increased performance by **24.24%** compared to 22.78% without augmentation. A **1.46%** increase at x4 resource requirement.

Waves and rocks were particularly hard to detect as they can be easily mistaken for background. Object segmentation models may be best for detecting wave and rocks, as that method works best for irregular objects.

CHAPTER V – CONCLUSION & FUTURE WORK

5.1 Summary

An introduction to the background information of autonomous surface vehicles (ASVs), object detectors and CNN based object detectors, especially RetinaNet was introduced with background information, and then a proposal of the research project designing an object detector that would detect and recognize different classes of objects in the sea was discussed, with further explanation on the overall system design and methodology adapted. The experimental results have demonstrated that adding unlabeled images to the training set provided a 12.53% increase in mAP while the augmentation contributed to an increase of 11.31% in the performance.

Overall, the combination of augmentation, resizing all inputs to the same size and addition of unlabeled samples provided the most improvement with a 24.24% increase compared to a combination that excludes augmentation at 22.78%. Classes such as island and buoy had the most improvement following the application of the techniques described.

5.2 Challenges and Limitations

- **Lack of high-resolution images:**

  While creating our SID, the first step involved scaling down the dataset due to a limited number of high-resolution images for some classes. To avoid the issue of class imbalance, a smaller dataset with classes having around the same number of images was created. Example of classes that had fewer high-resolution images are Fish, waves, islands among others.

5.3 Future Work

- **Use a newer dataset or larger dataset with high resolution images**

  Training on a larger maritime dataset that include the classes trained on, with additional maritime classes will certainly improve the performance of our model.

- **Addition of features to help with collision avoidance and decision making**

  It will be beneficial to design and implement a real-time RetinaNet-based object detector that both accurately and efficiently detect objects and even move further to make decisions based on the objects that were detected.

5.4 Contribution

In this work, the impact of input image size on model performance was explored. Literatures have explained that the higher the resolution of an image, the better the details learned. However, with varying image sizes, the anchor boxes which are pivotal to the detection of objects do not operate on the principle of one size fits all. As a result, we tested the impact of the sizes of input images on the selection of anchor boxes which resulted to better IoU and overall prediction performance.

The effect of varying augmentation techniques was also investigated. Several augmentation techniques have been reported in literature to increase the size of training data. With object detection as with most machine learning tasks, usually the more training data available the better. However, a lot of these augmentation techniques have been reported to either increase false positive identifications or do not yield a performance improvement that is comparable to the effort employed. We had explored a set of techniques to find out which combinations best improve performance without expensive

tradeoffs.

REFERENCES

[1] P. Soviany and R. T. Ionescu, "Optimizing the Trade-Off between Single-Stage and Two-Stage Deep Object Detectors using Image Difficulty Prediction," in *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC) (pp. 209-214).IEEE*, 2018.

[2] K. He, G. Gkioxari, P. Dollar and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE international conference on computer vision (pp. 2961-2969)*, 2017.

[3] X. Lu, Q. Li, B. Li and J. Yan, "MimicDet: Bridging the Gap Between One-stage and Two-stage Object Detection," *arXiv preprint arXiv*, no. 2009.11528, 2020.

[4] S. Zhang, W. Longyin, X. Bian, L. Zhen and L. Z. Stan, "Single-shot refinement neural network for object detection.," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

[5] J. Redmond, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition. 2016*, 2016.

[6] W. Liu, A. Dragomir , E. Dumitru , S. Christain, R. Scott, F. Cheng-Yang and C. B. Alexander , "Ssd: Single shot multibox detector.," in *In European conference on computer vision*, 2016.

[7] T.-Y. Lin, G. Priya, G. Ross and H. Kaiming, "Focal loss for dense object detection.," in *In Proceedings of the IEEE international conference on computer vision*, 2017.

[8] M. I. G. " Top Reasons for Maritime Accidents," 18 July 2014. [Online]. Available: https://www.maritimeinjuryguide.org/blog/top-reasons-maritime-accidents/. [Accessed 20 February 2021].

[9] Z. Rao and A. Fernando, "Use of an artificial neural network to capture the domain knowledge of a conventional hydraulic simulation model," *Journal of Hydroinformatics*, vol. 9, no. 1, pp. 15-24, 2007.

[10] M. Breivik, *Topics in Guided Motion Control of Marine Vehicles*, ResearchGate, 2010.

[11] S.-J. Lee, M.-I. Roh, H. Lee and J.-S. Ha, "Image-based Ship Detection and Classification for Unmanned Surface Vehicle Using Real-Time Object Detection Neural Networks," in *Conference: The 28th International Ocean and Polar Engineering Conference*, 2018.

[12] Z. Liu, Z. Youmin, Y. Xiang and Y. Chi, "Unmanned surface vehicles: An overview of developments and challenges.," *Annual Reviews in Control*, vol. 41, pp. 71-93, 2016.

[13] Y. Xu, L. Jun, J. Chen, S. Guangtian and G. Yangjian, "A novel approach for visual Saliency detection and segmentation based on objectness and top-down attention," in *In 2017 2nd international conference on image, vision and computing (ICIVC) pp.361-365 IEEE*, 2017.

[14] S. Campbell, N. Wasif and I. W. George, "A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres," *Annual Reviews in Control* , vol. 36, no. 2, pp. 267-283, 2012.

[15] R. Benson, O. Mohammed, H. Jan and S. Bernt, "Ten years of pedestrian detection, what have we learned?," in *In European Conference on Computer Vision pp. 613-627. Springer, Cham,* , 2014.

[16] M. Welling, A First Encounter with Machine Learning, Irvine, CA.: University of California, 12, 2011.

[17] J. Huang, R. Vivek, S. Chen, Z. Menglong, K. Anoop, F. Alireza, F. Ian , Z. Wojna, Y. Song, S. Guadarrama and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," in *In Proceedings of the IEEE conference on computer vision and pattern recognition pp. 7310-7311.*, 2017.

[18] L. Tsung-Yi, M. Michael , B. Serge, H. James , P. Pietro, R. Deva, D. Piotr and Z. C. Lawrence, "Microsoft COCO: Common Objects in Context," in *European Conference on Computer Vision*, 2014.

[19] M. Bowles, Machine learning in Python: essential techniques for predictive analysis, John Wiley & Sons, 2015.

[20] S. B. Kotsiantis, Z. I and P. P, "Supervised machine learning: A review of classification techniques.," *Emerging artificial intelligence applications in computer engineering* , vol. 160, no. 1, pp. 3-24, 2007.

[21] D. Alon, "Machine learning algorithms: a review," *International Journal of Computer Science and Information Technologies* , pp. 1174-1179, 2016.

[22] Y. G, "The 7 Steps of Machine Learning," 31 August 2017. [Online]. Available: https://towardsdatascience.com/the-7-steps-of-machine-learning-2877d7e5548e. [Accessed 3 March 2021].

[23] K. Chumachenko, "Machine learning methods for malware detection and classification," in *ACSAC '20: Annual Computer Security Applications Conference. Pages 54–68*, 2020.

[24] S. Ravindra, "The Machine Learning Algorithms Used in Self-Driving Cars," June 2017. [Online]. Available: https://www.kdnuggets.com/2017/06/machine-learning-algorithms-used-selfdriving-cars.html. [Accessed 5 March 2021].

[25] R. K. Srivastava, K. Greff and J. Schmidhuber, "Training Very Deep Networks," *arXiv preprint*, vol. arXiv:1507, no. 06228 , 2015.

[26] J. Hui, "Understanding Feature Pyramid Networks for object detection (FPN)," 26 March 2018. [Online]. Available: https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c. [Accessed 30 January 2021].

[27] IIoT-World.com., "Machine Learning Algorithms in Autonomous Driving," 15 March 2018. [Online]. Available: https://iiot-world.com/artificial-intelligence-ml/machine-learning/machine-learning-algorithms-in-autonomous-driving/. [Accessed 15 March 2021].

[28] R. Yamashita, N. Mizuho, K. Richard, D. Gian and T. Kaori, "Convolutional neural networks: an overview and application in radiology.," *Insights into imaging*, vol. 9, no. 4, pp. 611-629, 2018.

[29] J. Han, C. Yonghoon, K. Jonghwi, K. Jinwhan, S. Namsun and Y. K. Sun, "Autonomous collision detection and avoidance for ARAGON USV: Development and field tests," *Journal of Field Robotics*, vol. 37, no. 6, pp. 987-1002, 2020.

[30] S. Guo, Z. Xiuguo, Z. Yisong and D. Yiquan, "An autonomous path planning model for unmanned ships based on deep reinforcement learning.," *Sensors,* vol. 20, no. 2, p. 426, 2020.

[31] M. Zhu, W. Xuesong and W. Yinhai, "Human-like autonomous car-following model with deep reinforcement learning.," *Transportation research part C: emerging technologies,* no. 97, pp. 348-368, 2018.

[32] A. Krizhevsky, S. IIya and H. E. Geoffrey, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems,* 2012.

[33] Y. LeCun, B. Yoshua and H. Geoffrey, "LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning," *Nature,* vol. 521, no. 7553, pp. 436-444, 2015.

[34] O. Russakovsky, D. Jia, S. Hao, K. Jonathan, S. Sanjeev, M. Sean, H. Zhiheng, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International journal of computer vision,* vol. 115, no. 2, pp. 211-252, 2015.

[35] R. Sharif, A. S. Razavian, H. Azizpour, J. Sullivan and S. Carlsson, "CNN Features off-the-shelf: an Astounding Baseline for Recognition," in *In Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp. 806-813.*, 2014.

[36] Y. LeCun and Y. Bengio, Convolutional networks for images, speech, and time series, The handbook of brain theory and neural networks, 1995.

[37] S. Karen and A. Zissernman, "Very deep convolutional networks for largescale image recognition," *arXiv preprint arXiv*, pp. 1409-1556, 2014.

[38] F. A. El-kadar, M. S. A. El-soud, K. El-Serafy and E. A. Hassan, "An Integrated Navigation System For Suez Canal (SCINS); INTEGRATED NAVIGATION SYSTEM," *The Journal of Navigation*, vol. 56, no. 2, p. 241, 2003.

[39] C.-N. Hwang, Y. Joe-Ming and C. Chung-Yen, "The design of fuzzy collision-avoidance expert system implemented by H∞-autopilot.," *Journal of Marine Science and technology*, vol. 9, no. 1, pp. 25-37, 2001.

[40] R. Shaoqing, H. Kaiming , G. Ross and S. Jian , "Faster R-CNN: towards real-time object detection with region proposal networks.," *IEEE transactions on pattern analysis and machine intelligence* , vol. 39, no. 6, pp. 1137-1149, 2016.

[41] T.-Y. Lin, D. Piotr, G. Ross , H. Kaiming, H. Bharath and B. Serge , "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[42] K. He, Z. Xiangyu, R. Shaoqing and S. Jian , "Deep residual learning for image recognition," in *In Proceedings of the IEEE conference on computer vision and pattern*, 2016.

[43] S. Karen and Z. Andrew, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint* , vol. 1409, no. 1556, 2014.

[44] S. Li, J. Jiantao, H. Yanjun and W. Tsachy , "Demystifying resnet," *arXiv preprint*, no. arXiv:1611.01186., 2016.

[45] S.-H. Tsang, "Review: FPN — Feature Pyramid Network (Object Detection)," 17 January 2019. [Online]. Available: https://towardsdatascience.com/review-fpn-feature-pyramid-network-object-detection-262fc7482610. [Accessed 17 February 2021].

[46] S. R. Bulo, G. Neuhold and P. Kontschieder, "Loss max-pooling for semantic image segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[47] S. Trivedi, "Understanding Focal Loss—A Quick Read," 2 May 2020. [Online]. Available: https://medium.com/visionwizard/understanding-focal-loss-a-quick-read-b914422913e7. [Accessed 10 March 2021].

[48] H. Caesar, U. Jasper and F. Vittorio, "Coco-stuff: Thing and stuff classes in context," in *In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1209-1218).*, 2018.

[49] A. Rosebrock, "Intersection over Union (IoU) for object detection," 2016. [Online]. Available: https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/. [Accessed 3rd Maech 2021].

[50] R. Parmar, "Common Loss functions in machine learning," 02 September 2018. [Online]. Available: https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23. [Accessed 3rd March 2021].

[51] Felzenszwalb, F. Pedro, B. Ross, B. R. Girshick and M. David , "Cascade object detection with deformable part models," in *IEEE Computer society conference on computer vision and pattern recognition*, 2010.

[52] A. Shrivastava, G. Abhinav and G. Ross, "Training region-based object detectors with online hard example mining," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[53] Z. Tian, C. Shen, H. Chen and T. He, "Fcos: Fully convolutional one-stage object detection.," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[54] I. S. Atawodi, *A Machine Learning Approach to Network Intrusion Detection System,* Masters Thesis , 2019.

[55] Y. Wang, C. Wang, H. Zhang, Y. Dong and S. Wei, "Automatic Ship Detection Based on RetinaNet Using Multi-Resolution Gaofen-3 Imagery," *Remote Sensing,* p. 531, 2019.