University of Stavanger

Faculty of Science and Technology
Department of Electrical Engineering and Computer Science

Advanced Deep Learning for Whale detection in VHR Satellite Images

Master's Thesis in Computer Science
by

Rabbir Bin Rabbani

Internal Supervisors

Dr. Naeem Khademi

June 15, 2020

“The only true wisdom is in knowing you know nothing”

Socrates

Abstract

Whales are an integral part of our oceans. They keep the balance of the aquatic food chain and reproduction of most species [1]. Scientific studies of the cetacean species (whales, dolphins, etc) has led to many discoveries and advancements regarding echolocation, aquatic environments, marine life/biology and marine mammal intelligence and other important oceanic topics. Whale watching, in particular, have brought in 2 billion US dollars in just the year 2009, and has been a growing industry ever since, which has contributed greatly for both the economic gain of countries as well as funding many research involving the oceans and marine life [2].

With the use of very high resolution(VHR) satellite images we can locate whales. In this thesis, I have experimented with different advanced architectures in Deep Learning to detect whales in satellite images. I have also showed how effective the models are at finding whales in comparatively lower resolution images. Finally, I have used this model to draw bounding boxes around the whales.

Acknowledgements

I am very grateful to my supervisor Dr. Naeem Khademi for providing me this wonderful opportunity to pursue this thesis under him. His guidance, ideas, encouragement and insightful and valuable feedback amidst his busy schedule has been invaluable during the thesis.

Contents

Abbreviations

ESA	**T**he **E**uropean **S**pace **A**gency
ANN	**A**rtificial **N**eural **N**etwork
CNN	**C**onvolutional **N**eural **N**etwork
VHR	**V**ery **H**igh **R**esolution
NOAA	**N**ational **O**ceanic and **A**tmospheric **A**dministration
SWIR	**S**hort-**W**ave **I**nfra**R**ed
CAVIS	**C**louds, **A**erosols, **W**ater **V**apor, **I**ce and **S**now

Chapter 1

Introduction

Whales are fascinating creatures. They are the largest mammals and they are important to all species both living inside the ocean and in the land. They play several important roles in this world, for example, they play a large part in balancing the food chain in the ocean [1] and they have been known to absorb many tons of greenhouse gasses during their lifetime and when they die, they submerge and take all of it with them [3]. Just in 2009, whale watching brought in 2 billion US dollars and it has been a growing industry ever since [2].

1.1 Motivation

Locating whales is a very expensive and time consuming process and sometimes its just not feasible, especially in remote areas(e.g: [4, 5]). Illegal Whaling is an issue [6, 7] that whales face on a regular basis, but it isn't the only one. Occasionally, whales migrate through busy shipping lines and collision accidents leave whales severely injured, if not kill them outright [8]. They also become entangled quite often in fishing lines when they come close to shore [9].

1.2 Problem Definition

Satellite images have been around for decades now, but they lacked good enough resolution to detect objects in their images. In their paper, Hannah C. Cubaynes, et al. [10] discussed how whales can be manually detected in the images that the Worldview-3 sensors can produce. With Convolutional Neural Networks it maybe possible to turn the manual process to an automated one. Highlighting areas of interest maybe particularly useful to anyone navigating the oceans to avoid disturbing the natural habitats of whales. Satellite images, however, especially newer ones, like the Worldview-3 is very costly to operate and so the images are also expensive. Using the same model, we maybe able to detect whales using other in comparatively lower resolution satellite images.

1.3 Research Questions

This thesis aims to answer the following questions -

1. Can Deep Learning be used to identify whales in the Very High Resolution images of satellites?
2. Can the same model be used to identify whales in comparatively lower resolution images? If it is achievable, then up to what resolution can whales be detected?
3. Can the models be used to highlight areas of interest that contain whales in an image?

1.4 Use Cases

This thesis is primarily intended to help in the research of whales. The most obvious use case for detection is in the conservation of whales. But detecting whales also can be used by various industries operating in the oceans, especially the fishing and shipping industries to avoid or reroute away any areas where whales are currently active, for example the Right whale species occasionally gets injured or killed by boat accidents, since they migrate through some of the worlds most busiest shipping lanes [8] they also get tangled in fishing lines quite often when they come near the shore [9]. Knowing where the whales are in a given time period could be used to avoid such unfortunate accidents and also avoid any potential damage to the ships themselves at the same time.

1.5 Challenges

A lot of satellites that observe earth are privately owned, including the focus of this thesis, the WorldView-3 satellite, is owned and operated by Maxar, previously DigitalGlobe. Since images from satellites like this is used commercially, there are, understandably, no public datasets for the worldview-3 satellite, let alone public datasets that are centered around whales.

It should also be mentioned that the European Space Agency (ESA) allows limited access to images from privately owned satellites for research purposes. Unfortunately, the size limit for this is limited, approximately 500Km^2. This is a giant area which would be very useful in many different use cases, such as agriculture, but whales travel across far larger areas, especially while migrating. While we were able to find a few satellite images that have whales in them, their sizes exceeded the ESA limit. I could only ask for one image and only that one image would not be enough to train a Deep Learning Model and have good results. A list of all the satellite images I have managed to find from previous works is provided in Appendix section A.2.

1.6 Outline

Chapter 2 - Background: The first section in this chapter describes relevant theory behind the methods implemented in this thesis. The second section, I discuss some related works to provide some context to this thesis.

Chapter 3 - Solution Approach: This chapter discusses existing approaches and gives an overall idea of what is implemented in this thesis.

Chapter 4 - Implementation: This chapter shows what has been done in this thesis. It starts by discussing how the data was collected and preprocessed, then shows the architecture of the models and how they were trained and tested. Then it shows how the Object Detection algorithm was implemented.

Chapter 5 - Results: The results after the implementation in Chapter 4 are shown in this chapter.

Chapter 6 - Discussion: This chapter discusses the results shown in Chapter 5.

Chapter 7 - Conclusion and Future Directions: A conclusion for the thesis is given, including the answers to the questions in section 1.2.

Chapter 2

Background

As the name suggests, this chapter aims to provide a background to the work. It starts with short summaries of technical and theoretical information that is relevant to this thesis. First I provide some details as to the different kinds of satellites and what each is used for, followed by a brief elaboration to the kind of satellite that is the focus of this thesis and some relevant definitions.

Since I am using Deep Learning methods, I have written down some short descriptions of what I mean when I mention the terms such as Machine Learning, Deep Learning, Convolutional Neural Network, Classification, etc. I also needed to objectively measure the performance of my work, section 2.1.6 goes through the performance metrics that I have used. Then it moves to some more advanced terms such as Object Detection, which is the technical term for what I have done in this thesis and Transfer learning, which allowed me to have a very good accuracy despite having a relatively small dataset.

The second section discusses some related works. I wanted to give a very compressed context to some of the many wonderful work done before my thesis. This section starts by mentioning the very first satellite image and how it was the catalyst to understanding the value of satellite images to a lot of industries, from Agriculture to Oceanography. Then I focus a little more on how Satellite images are important to almost anyone studying the oceans past, present and the possible futures. Then I mention how recent research shows that the one of the most advanced satellites available to us, the WorldView-3 satellite, has enough resolution to help identify whales. And then I delve into pervious researches that have used satellite images to detect whales.

2.1 Technical and Theoretical Background

As mentioned above, this section aims to provide a short summary of the technical and/or theoretical knowledge that is required to understand the work that has been done in this thesis and also by other similar and related works.

2.1.1 Understanding Satellites and Satellite Imagery

There are many different types of satellites and all are specialized to one or more tasks.

When we think of satellites, our first thoughts are probably about the satellites observing other celestial objects in the distance, the most famous of these is the Hubble telescope. These are known as Astronomical satellites, they are also known as space telescopes. Perhaps some of the least known of the satellite types are the Biosatellites, which are used to study how different plants and animals behaves in the extreme conditions of space, such as radiation and zero gravity. The first of these satellites is the Sputnik 2, that carried the first animal in space, a dog named Laika [11]. There are, of course, the communication satellites, that we use most frequently, for telecommunications, from Telephone to Internet to Military Communications. Then there are the satellites that we use for Navigation systems, such as Global Positioning System (GPS) and Galileo Positional System (also GPS). There are also the satellites for military use whose full capabilities are, of course, kept classified by the government that operates them. They are generally used for reconnaissance, and there are even a small number of satellites that are equipped with automated weapons to defend themselves.

For this thesis, we are focusing on the Earth Observation Satellites, that are designed for researching the Earth from space. These satellites tend to be in low-earth orbit ($\leq$ 2000 Km above sea level) and most follow a Sun-synchronous orbit, where the satellite passes over any given point of the planet's surface at the same local mean solar time [12]. Simply put, the satellite will be over any point of Earth at the same point in the time of day (Ex: the Worldview-3 satellite passes over all points of Earth at exactly 10:30AM of the local time [13]). These satellites are used for many different purposes from Weather forecasting to Agriculture, and accommodate such a wide array of applications, they have to be equipped with many different sensors from the simple panchromatic(greyscale) sensors to the multi-band Multispectral images (3 - 8 bands) sensors, to the SWIR (Short-Wave Infrared) sensors. They also need instruments for CAVIS (Clouds, Aerosols, Water Vapor, Ice and Snow) to correct for the inconsistencies caused by specific weather conditions in order to capture quality images under all circumstances [14].

When we think of an image resolution, we generally think of the number of pixels along the height and width of the image. But to be useful for people using satellite images they generally have four clearly defined resolutions - spatial, spectral, temporal and radiometric resolutions. For the purposes of the thesis, we only needed to work with spacial resolution, which is the area covered by each pixel of the images (Ex: Worldview-2 satellite has a spacial resolution of 0.46m [15])

2.1.2 Machine Learning and Deep Learning

"Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed" [16]. Experience can be some kind of input data (text, excel tables, images, sound clips, videos, etc). With Machine Learning a computer can find patterns in the data to make some kind of predictions that would normally be very complex for a human being [17].

"Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called Artificial Neural Networks (ANN)" [18]. ANNs are discussed further in section 2.1.3.

2.1.3 Neural Networks

Neural Networks or Artificial Neural Networks (ANN) are designed to simulate biological neural networks. An artificial neuron in a Neural Network, is a function that serves the same function biological neurons have in the human brain. Each neuron receives several numbers as inputs, then the sum of the input is calculated and then processed through an activation function which decides if the information is to be sent to the next neuron.

The structure of an ANN is generally split up into three types of layers - input, hidden and output. The input layer contains information representing the features of the subject that is to be classified. The hidden layer contains one or more layers of artificial neurons. The output layer consists of neurons which calculates the final output of the network.

2.1.4 Convolutional Neural networks

"A Convolutional Neural Network is a Deep Learning algorithm which can take in an input image, assign importance to various aspects/objects in the image and be able to differentiate one from the other" [19].

Instead of learning a pattern in the input values, CNNs find a pattern in the images themselves, (such as, edges or corners) using convolution filters. This pattern then passed to the next layer as input to find even more patterns and so on until the output. In the case of images, CNNs show a far better performance than a regular ANN.

2.1.5 Classification

"In machine learning and statistics, classification is the problem of identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations whose category membership is known" [20]. Assuming we have a list of correctly identified observations, the individual observations are analyzed into a set of features which also have a defined class associated with them, this is usually called the training set. Using this set, the Classifier learns a pattern to correctly identify the class of another, possibly unknown, observation.

2.1.6 Performance metrics

In order to determine the effectiveness of any machine learning or deep learning method we must first define some metrics to objectively quantify it's performance.

Confusion Matrix

To understand how to calculate the performance of a Classification method, we have to first understand the confusion matrix. A confusion matrix shows the performance of a classification algorithm on a test dataset by showing the number of correctly identified observations (True Positives and True Negatives) and also the number of incorrectly identified observations (False Positives and False Negatives) in an NxN table, where N represents the number of classes the models is trained to classify. This can be further used to define several performance metrics. The ones I have used are Accuracy, Precision, Recall and F1-Score.

Accuracy

Accuracy is the overall performance of the classifier. Simply put, it tries to answer how much of the predictions were correct. It is calculated using the formula shown in Eqn.(2.1).

Given,
TP = True Positives
FP = False Positives
TN = True Negatives
FN = False Negatives

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.1}$$

Precision

Precision measures how accurate the predictions of a classifier is (Ex: How many of the Emails that are predicted to be Spam, are actually Spam?). It is calculated using the formula shown in Eqn.(2.2).

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Recall

Recall calculates how many of the actual positives the classifier has correctly predicted. It is a ratio of correct prediction against the ground truth (Ex: How many of the spam emails were correctly labelled by the model?). It is calculated using the formula shown in Eqn.(2.3).

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

F1-Score

"The F1 score is the harmonic mean of the precision and recall" [21]. It aims to give an overall picture of the performance of the model. It is calculated using the formula shown in Eqn.(2.4).

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{2.4}$$

2.1.7 Advanced Architectures in Deep Learning

Deep Learning Models have impressive performance on their own, but their architecture has to be customized and parameters have to be tuned to fit a specific use case. So it didn't take long for many different researches to be done on finding some more general purpose architectures that would do well in a specific problem domain instead of doing trial and error to find the right structure for every use case. There are many of such structures dedicated specifically for image based deep learning.

The advanced architectures I have used in this thesis are discussed below.

MobileNet

The creators of the MobileNet architectures intentions were to be able to run Deep Learning algorithms in hardwares with very low copmputational power [22]. one use case they wanted to fulfill was to be able to run image classification in a Raspberry Pi. MobileNets use a more effecient alternative to Standard Convolution operations known as Depthwise Separable Convolution [23], which reduce the computational intensity of MobileNets by a significant margin at the cost of a very small reduction in the accuracy.

Xception

To understand Xception, we have to loosely understand the Inception architectures. These architectures are made up of modules of several filters with differing sizes of convolution operations. The results of these filters are then concatenated to form a single output from the module. These modules are then stacked on top of each other several times so that the output from one module is the input to the next module.

These complex modules are replaced by simple depthwise separable convolution [23] operations to create the Xception architecture [24]. The authors of this architecture proved in their paper that Xception is both faster and more accurate, in most cases, than the Inception-V3 architecture.

ResNet

Same as the Inception Architecture discussed above, ResNet [25] architectures are made up of several specialized modules, called residual modules, put on top of each other to create the final model. ResNet models are often named with a number as a suffix to

ResNet that represent the number of modules stacked (Ex: ResNet50 is made up of 50 of these residual modules).

DenseNet

DenseNets [26] are a set of architectures, with a similar idea to the Inception and ResNet. They are made up of several layers of modules, called dense blocks, except these blocks are arranged so the input for each of these blocks is a concatenation of outputs from all previous blocks. This alleviates the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters. Like ResNet, DenseNet also usually has a number as a suffix to the name representing the number of these dense blocks that are present in the final architecture.

2.1.8 Transfer Learning

In Deep Learning, Transfer learning is the act of applying knowledge/experience gained from solving a problem to solve another similar problem. In technical terms, we use the weights from the previous solution to a problem to be the starting point for another similar problem. This has an added benefit of not needing a large dataset, provided that the weights are taken from a similar problem domain.

2.1.9 Object Detection

The purpose of object detection is to specify the areas of interest where specific objects are located in an image. The output of such a system is represented by coordinates of rectangular boxes containing objects and an associated object class given by highest probability and optionally, the class that has a probability higher than a specified threshold. Some popular object detection use cases include face recognition, pedestrian detection, Smile Detection, Automatic Image Annotation and many others.

2.1.10 Sliding Window Operation

A sliding window is a square or rectangular filter that “slides” across an image. The purpose of this, is to perform some kind of operation to these windows. In the context of this thesis, I wanted to perform a classification operation to identify objects present in these windows.

2.2 Related Works

This section aims to discuss works that are related to this thesis. I have taken a top-down approach, in order to build a background and help understand the work done before this thesis was even possible.

2.2.1 Taking images of the Earth from space

The first satellite photographs of Earth were taken by the Explorer-6 satellite on 1959 [27]. The image itself isn't very impressive, its not even high quality enough to be very useful, but it did help to make a lot of brilliant minds realize how useful satellite images could be in the future. Ever since then, all Space Agencies around the world has been sending increasingly advanced satellites to take pictures of the Earth for many different purposes and made these images part of the big data revolution. They are increasingly used to track both human and natural activity. They have made a difference in a wide variety of industries, some common uses are in Weather forecasting, Cartography, Fishing, Agriculture, Forestry, Geology, Regional Planning, Oceanography, and even Warfare [28].

2.2.2 Satellite images, Oceanography and the Whales

”Oceanography is the study of the physical, chemical, and biological features of the ocean, including the ocean's ancient history, its current condition, and its future” [29].

Researchers under the National Oceanic and Atmospheric Administration (NOAA) has been using different sensors in satellites to study the oceans for decades now (Ex: Sea surface temperature, ocean color, coral reefs, sea and lake ice, etc) [30], but detecting specific objects and living beings in the oceans require that the images have enough spacial resolution for the objects to be clearly visible, at least to the naked eye.

In their paper, [10], Hannah C. Cubaynes, et al. proved how modern satellites have enough resolution to detect whales. Their aim was to find out if different species of whales can be identified using Worldview-3 and their conclusion was that it's resolution was enough to distinguish whales from other objects in the ocean such as ships and underwater rocks, but not enough to differentiate between different species of whales. But knowing that whales can be detected from satellite images was enough proof that Deep Learning models may also be able to detect whales.

2.2.3 Whale detection in satellite images using Deep Learning

One of the first works I found about detecting whales using satellites was by Fretwell PT, et al. [31]. They used thresholding techniques to count whales in satellite images. Two softwares were used, "ENVI5" [32] and "ArcGIS" [33]. They had an 89% accuracy, but there was one key issue with their method, they had to manually calibrate many parameters in both of the softwares mentioned in order to get a good result. So even though using software to made it easier for them, the process was still mostly manual.

A more automated process was used by Emilio Guirado, et al. [34]. The approach used in this paper was using 2 CNNs. The first step was to detect the presence of whales, and the second step was to count the whales. They achieved an impressive accuracy of 94%. However, the model is specialized to counting whales, which can be very useful in some cases. But I wanted to work on a more general purpose method, where the model would pinpoint the areas of interest that have whales.

As mentioned before, there are no public datasets for Worldview-3. But Alex Borowicz et al. had a great idea in their paper [35]. Their dataset was aerial images that had a resolution of 2cm and they had downsampled them to a resolution of approximately 31cm. This could be thought of the images with similar resolution to the Worldview-3 satellite images. Unfortunately, while trying to run their code, we couldn't, at first due to syntax errors. When we managed to fix those syntax errors, we faced runtime errors.

Chapter 3

Solution Approach

This chapter discusses existing approaches and gives an overall idea of what is done in this thesis. It starts by discussing some existing methods that are used to find whales and then discussing some existing Object Detection methods. Finally, it discusses the approach that I have used for my thesis.

3.1 Existing Approaches to detecting Whales

3.1.1 Traditional methods of finding whales

The traditional methods of finding whales is from either land, ships or low flying aeroplanes. The method used depends completely on the location that is to be surveyed.

Whales usually stay inside peninsulas during their breeding seasons, they also sometimes stay close to shore while migrating. These are the only times when a land based survey can be used. This involves going to a nearby high place like a cliff, or even observation towers that are built close to known breeding and feeding grounds and using a high powered binocular to observe their behaviour.

When researchers are trying to simply find a pod of whales out in the ocean or take quick pictures or videos, they use aeroplanes that fly in low altitude, from there they either use a binocular or an on board camera and/or more specialized equipment to find whales.

By far, the most used method is to find whales using ships. That is because ships aren't simply used to observe whales like land and aircraft surveys, but also to closely monitor their health, take account of the population and even occasionally, to untangle them from fishing lines. There are also a sure way for researchers to find whales using a boat, they can use both SONAR and listen for whale songs to know if they are close to a pod of whales. Land is too far away to listen to whale songs and aeroplanes are just too fast.

3.1.2 Finding whales using GPS

A great method of finding and studying whales is to track them using GPS. One such program that is open for researchers is the "Whale and Dolphin Tracker" from the Pacific Whale Foundation which tracks the whales around the islands of Maui [36]. These trackers are not only for tracking their location however, as they can also record the depth, water temperature, water pressure and even underwater sounds.

There is also the WhaleTrack Program in the University of Tromso [37], which tracks hundreds of whales across the Atlantic and North sea. They developed their own GPS tracking system as well. Their system is open for everyone to see and maps the path each whale take over a given time period.

3.1.3 Other methods of finding whales using Deep Learning

Images are not the only way to find whales, sound is a very big element when it comes to the many whales and dolphin species. Whales are some of the largest animals there are and can easily be detected using SONAR. Another interesting characteristic about whales is their beautiful whale songs, which also happen to be unique to each species of whales. A study by Google in the year 2018 [38] used 15 years of data from NOAA about Whale Songs along with information about when and where they were recorded from. They used ResNet50 to learn the unique patterns in the spectogram from a Humpback whale songs and their model was able to distinguish and recognize Humpback whales based on the spectogram of these songs.

3.2 Existing Approaches in Object Detection

3.2.1 YOLOv3 Object Detection

YOLO (You Only Look Once) [39] is probably the most well known Object detection framework. It uses a single CNN that locates all objects and the class probability of those objects at the same time. It only needs to scan the image only once, hence the name. YOLO is a very popular method of Object Detection for several reasons, most notably it's speed. YOLO is fast enough to be run on a real time video feed. It is also very lightweight (just 53 layers), but has an accuracy almost the same as ResNet152 (152 layers).

3.2.2 RetinaNet

RetinaNet [40] is a combination of 2 Layers Deep Neural networks. The first layer is called the backbone network. It's purpose is to generate a defined number of feature maps using a method known as Feature Pyramid Nets [41]. The same objects may be in a wide range of sizes in images depending on the distance of the image from the camera. So the backbone network uses several filters of differing sizes to create multiple feature maps that pick up smaller or larger features proportional to the size of the filter.

Each of these feature maps are then sent to the second layer which is made of a pair of Classification and Regression Networks for each of the feature maps. The classification networks identify the class of the objects while the regression networks identifies the coordinates of the bounding box.

3.3 Evaluation of the Existing Methods

The traditional methods are quite expensive and require a lot of prior knowledge and expertise. In recent years, GPS tracking has been tremendous help in locating and studying whales but overtime these devices malfunction or breakdown and then have to be replaced by tracking .

Both of the object detection methods discussed in section 3.2 have their pros and cons. But there is a more important reason they were not feasible for me. The models expect the training images to be labelled to their own standards. After combining the MASATI and Bioconsult datasets, I had over 10,000 images and none of them were labelled. It was not a feasible option for me to label all these images one by one with the limited time I had.

3.4 Proposed Solution

My proposed object detection model is simple compared to the Object Detection frameworks mentioned in section 3.2. It runs a sliding window through the image. The model would detect if a particular window has a whale or not. If it detects a whale with a confidence of more than a certain threshold, then it will draw a bounding box around the whale.

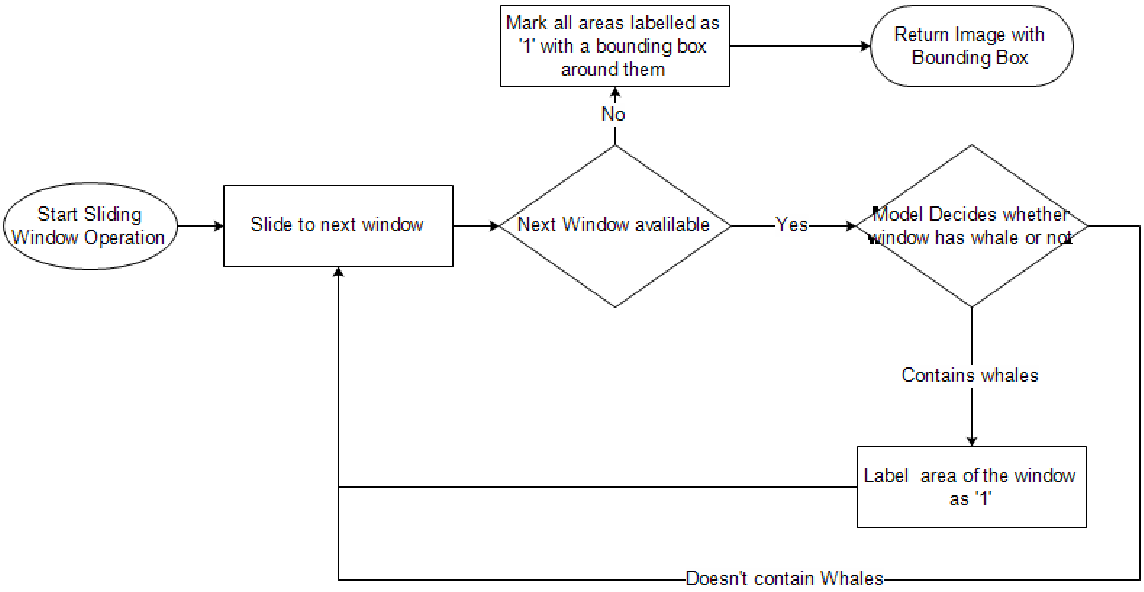
Mark all areas labelled as '1' with a bounding box around them
Return Image with Bounding Box
No
Start Sliding Window Operation
Slide to next window
Next Window avalilable
Yes
Model Decides whether window has whale or not
Contains whales
Label area of the window as '1'
Doesn't contain Whales

Figure 3.1: A Rough Idea for the Solution.

Fig. 3.1 shows a rough Idea of what happens when an image is provided to the object detection function. The detection part of the system starts and ends with the sliding window operation discussed in section 2.1.10. For every 'window' the model will check if the window contains whales or not. If the model predicts that the window contains whales then the area of the window is labelled as '1'. If not, nothing is done. This continues until the sliding window operation is done. Afterwards, all areas that was labelled '1' is considered to contain whales and a bounding box is drawn around them.

Chapter 4

Implementation

This chapter explains how each part of the system is implemented. I start BY explain how I collected and organized the dataset in section 4.1. I also explain how the downsampling was done, where and why I stopped downsampling the images. In section 4.2, I explain how the Training and Testing was setup and performed. Finally in section 4.3, I explain how the Object Detection was implemented.

4.1 Experimental Setup and Datasets

This section explains where the data was collected from, and how it was organized before the model is trained.

4.1.1 Organizing the datasets

Since the worldview-3 and many other satellites are privately owned there are no public datasets from these satellites, so we had to improvise. We took the data from 2 different datasets, one is the MASATI (Maritime Satellite Imagery) dataset [42] and in the dataset used by Bioconsult [35]. The MASATI dataset contained images of land, buildings, coast, water and ships and the Bioconsult dataset contained images of water and whales. We combined these datasets and split the images with resolutions of 32x32 pixels, then separated them into two folders, Whale and Not Whale. But since we had so many different objects in the Non Whale folder, and just whales in the other, we had a big imbalance in the dataset. We copied the whale images to match the same number of images as the non whale images for now. After they were the same size, we again separated them into three folders to represent the training, validation and Evaluation set.

4.1.2 Image Downsampling

As mentioned in the introduction, we wanted to know how well our models would perform with comparatively lower resolution images. For this, we used Image pyramids [43] in OpenCV to downsample the images. We took the 32x32 images and downsampled them upto 5 times, and upsampled them the same number of times so the images lose information during the downsampling but they retain their resolution in terms of pixels. By the 5th downsampling the images become a complete blur and nothing in the image can be determined, even with the naked eye, so we stopped our downsampling there. With this, we had the image approximate spacial resolution of 0.31m, 0.62m, 1.24m, 2.48m, 4.96m and 9.92m for us to experiment on.

Fig. 4.1 shows one example of the original image from the dataset. While really it is seen as very small, the whale is clearly distinguishable.

Figures 4.2 to 4.6 shows the same image as it can be seen after downsampling once and upto five times, each representing 0.31m, 0.62m, 1.24m, 2.48m, 4.96m and 9.92m spacial resolution respectively. As the images show, by the third and fourth downsampling everything is a blur and nothing can be distinguished or even guessed by looking at the images. By the fifth downsampling the image looks just like a single color image. This is where I stopped, as further downsampling would serve no purpose at all.

In the dataset folder, under the 'downsampled' folder these images can be found. The folders are numbered to represent how many times they have been downsampled, 0 being the dataset that was not downsampled, '1' being the dataset that was downsampled then upsampled once, and so on.

```python
# function to downsample and upsample a given image a specified number of times.
def downsample(image, times):
    for i in range(times):
        image = cv2.pyrDown(image)
    for i in range(times):
        image = cv2.pyrUp(image)

    return image
```

Listing 4.1: Downsampling using Image Pyramid

Code listing 4.1 shows the simple function that is used to downsample then upsample the images. The function 'cv2.pyrDown(image)' downsamples an image once, while the function 'cv2.pyrUp(image)' upsamples an image once. So in this function I am first

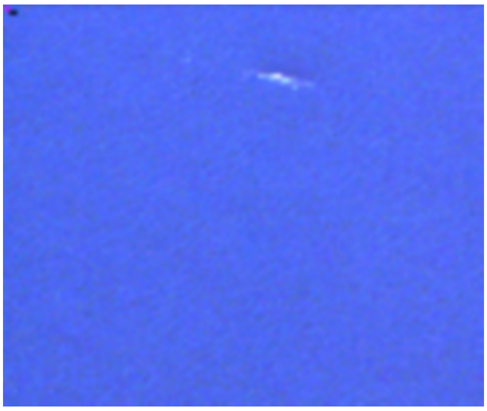

Figure 4.1: Original Image with no downsampling (~0.31m).

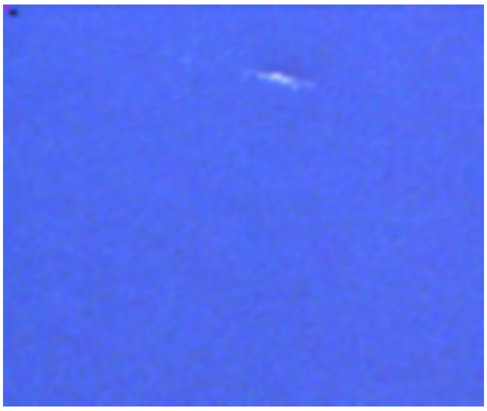

Figure 4.2: Image after downsampling once (~0.62m).

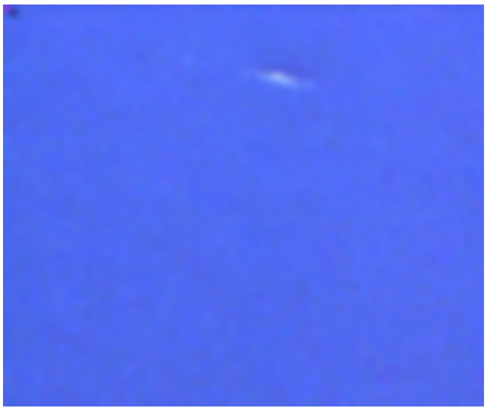

Figure 4.3: Image after downsampling twice (~1.24m).

Figure 4.4: Image after downsampling three times(~2.48m).

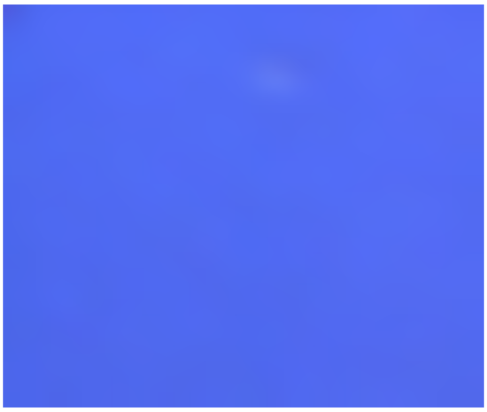

Figure 4.5: Image after downsampling four times (~4.96m).

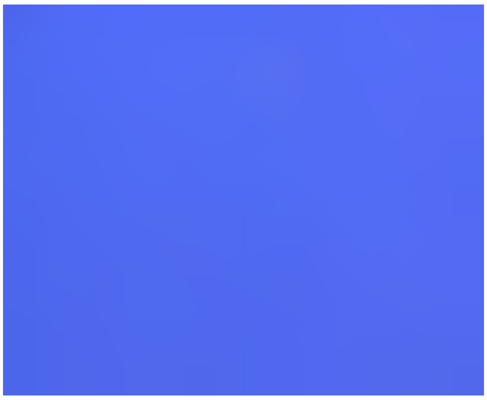

Figure 4.6: Image after downsampling five times (~9.92m).

downsampling an image a defined number of times and then upsampling them the same number of times.

4.2 Model Training and Testing

This section explains how the training and testing performed.

4.2.1 Loading the images

The images needed relatively little preprocessing, one thing I did have to handle is the mutiple copy pasted images in the whale folder. If the images were each morphed (by rotating them for example), then they would not technically be the same image. So I decided to randomly perform 3 kinds of morphing to the images, horizontal or vertical flipping or rotating to a random degree. Then I would not be training on copy pasted images.

The image pixel values must also be normalized from 0-255 to 0-1 since Neural networks only read values between 0 and 1. Because of the large number of images, it did not make sense to load all of them at once, so batches of images are incrementally loaded. I mentioned some random morphing on the images above as well, but they were not performed beforehand, instead they were applied when loading them.

These are all done using the ImageDataGenerator Class [44] in Tensorflow. The code listing 4.2 shows how the generator is configured.

```
datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    rescale=1./255,
    rotation_range=180,    # randomly rotate between 0-180 degrees,
    horizontal_flip=True, # allows to randomly flip the image Horizontally
    vertical_flip=True     # allows to randomly flip the image Vertically
)
```

Listing 4.2: ImageDataGenerator for training and validation set

This confuration first normalizes the pixel values by multiplying with 1/255, then randomly rotates and flips the images.

4.2.2 Training

All the models all follow a similar pattern. Code listing 4.3, shows the MobileNetV2 as an example of the structure for the models.

```
# Load the advanced architecture with pretrained weights of ImageNet.
base_model = tf.keras.applications.MobileNetV2(input_shape = (224, 224, 3),
                                               include_top=False,
                                              weights='imagenet')
base_model.trainable = True

model = tf.keras.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    Activation('relu'),
    layers.Dense(2, activation = 'softmax'),
])

model.compile(loss = tf.keras.losses.BinaryCrossentropy(from_logits=True),
              optimizer = tf.keras.optimizers.RMSprop(lr=learning_rate),
              metrics=['accuracy'])
```

Listing 4.3: Example code of the Model structure

All models follows this same structure. Except for the name of the base model, everything else remains the same. DenseNets work on 224x224 pixel images, so we decided that all input image, even though the actual image is 32x32 will be resized as input shape (224,224,3).

In the case of parameter tuning, only the learning rate is changed. The learning rates were 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}. The learning rates need to be such small numbers to account for the size of the models. The results from the training are saved in a ”<model>_results.csv” file the trained weights and training history are also saved for future use in the ”trained_models” folder.

All The training for all of the models is done in the same training notebook file.

4.2.3 Testing Setup

The testing is done in three stages for each of the models.

The first stage is to show the accuracy and loss of the training and validation set. This is done easily by loading the history from the ’trained_models’ folders. This helps to give us an overview of the performance of the models during training.

The second stage is to calculate the accuracy, precision, recall and f1-score and displaying the confusion matrix for each of the models and their different variations of learning rates to calculate their overall performance, so that we can decide which of the learning rates work best for each model.

The final stage is to show the accuracy of the model over different spacial resolutions. This is to find out how the models will perform in the case of comparatively lower resolution satellite images.

4.3 Object Detection Implementation

The final goal of the thesis was to detect whales in an image. As mentioned in section 3.4, our solution is a simple 32x32 pixel sliding window.

The Code Listing 4.4 shows the code for the object detection.

```python
def __predict_one(image):
    # Inputs:
    #   image -> Image for classification.
    # Returns:
    #   Prediction for the image.
    image = cv2.resize(image, (224,224), interpolation = cv2.INTER_AREA)
    image = image/255
    prediction = self.__model.predict([[image]], use_multiprocessing=True)

    if prediction[0][1] >= 0.85:
        return 1
    else:
        return 0

def detect(image_path, sliding_size = 32):
    # Draws a bounding box around whales inside the image.
    # Inputs:
    #   image_path -> Path to the image.
    # Returns:
    #   Image with the bounding box.

    # ---------------------------------------------------------------------------
    # Part-1: Load in the images
    # ---------------------------------------------------------------------------
    img_name = image_path.split("/")[-1]

    image = cv2.imread(image_path)

    img_height = image.shape[0]
    img_width = image.shape[1]
    channels = image.shape[2]
    # ---------------------------------------------------------------------------
    # Part-2: Sliding Window Operation
    # ---------------------------------------------------------------------------
    window_size_x = 32
    window_size_y = 32

    window_classes = np.zeros((img_height, img_width), np.uint8)

    for y in range(0, img_height+1, sliding_size):
        img_seg_y_min = y
        img_seg_y_max = y + window_size_y

        for x in range(0, img_width+1, sliding_size):
            img_seg_x_min = x
            img_seg_x_max = x + window_size_x

            cropped_image = image[img_seg_y_min:img_seg_y_max, img_seg_x_min:img_seg_x_max]

            if cropped_image.shape != self.__model_dims:
                pass
            prediction = self.__predict_one(cropped_image)
            window_classes[img_seg_y_min:img_seg_y_max, img_seg_x_min:img_seg_x_max] = prediction * 255

    # ------------------------------------------------------------------------
    # Part-3: Draw Bounding Box
    # ------------------------------------------------------------------------

    ret, thresh = cv2.threshold(window_classes, 127, 255, 0)

    contours, hierarchy = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

    for cnt in contours:
        x,y,w,h = cv2.boundingRect(cnt)
        cv2.rectangle(image, (x,y), (x+w,y+h), (0,0,255), 2)

    print("Result file:","object_detection_results/"+img_name)
    cv2.imwrite("object_detection_results/"+img_name, image)
    plt.imshow(image[:,:,::-1])
```

Listing 4.4: Object Detection Code

The first function listed '___predict_one' function, as the name would suggest is here to predict the class label for one image, or in this case, for one 'window'. It assumes that the image sent to it is a 32x32 pixel image. There is only one small change than the usual prediction. The function only returns '1' if the confidence of 0.85 that the image contains a whale.

But the main function here is the 'detect' function and it is divided into 3 parts marked by the comments. First it loads the image. Then starts the sliding window operation in the second part and predicts the class labels for each window, and saves it in the 'window_classes' variable. In the third and final part it draws contours using the 'window_classes' variable onto the original image.

Chapter 5

Results

In this chapter I present the results from the Implementation done in Chapter 4. This chapter starts by showing and explaining the training and testing results of all the models that I have tested from section 5.1 through to section 5.4. The next section 5.5, describes how well each model performs on a comparatively lower resolution image. Finally in section 5.6, I show some results from my Object Detection method.

5.1 Training and Testing results of MobileNetV2

Fig. 5.1, shows the performance of all of the MobileNetV2 models trained. The performance here is in terms of accuracy and loss. Each row of graphs shows the results of a model with the learning rate mentioned on top of the graph.

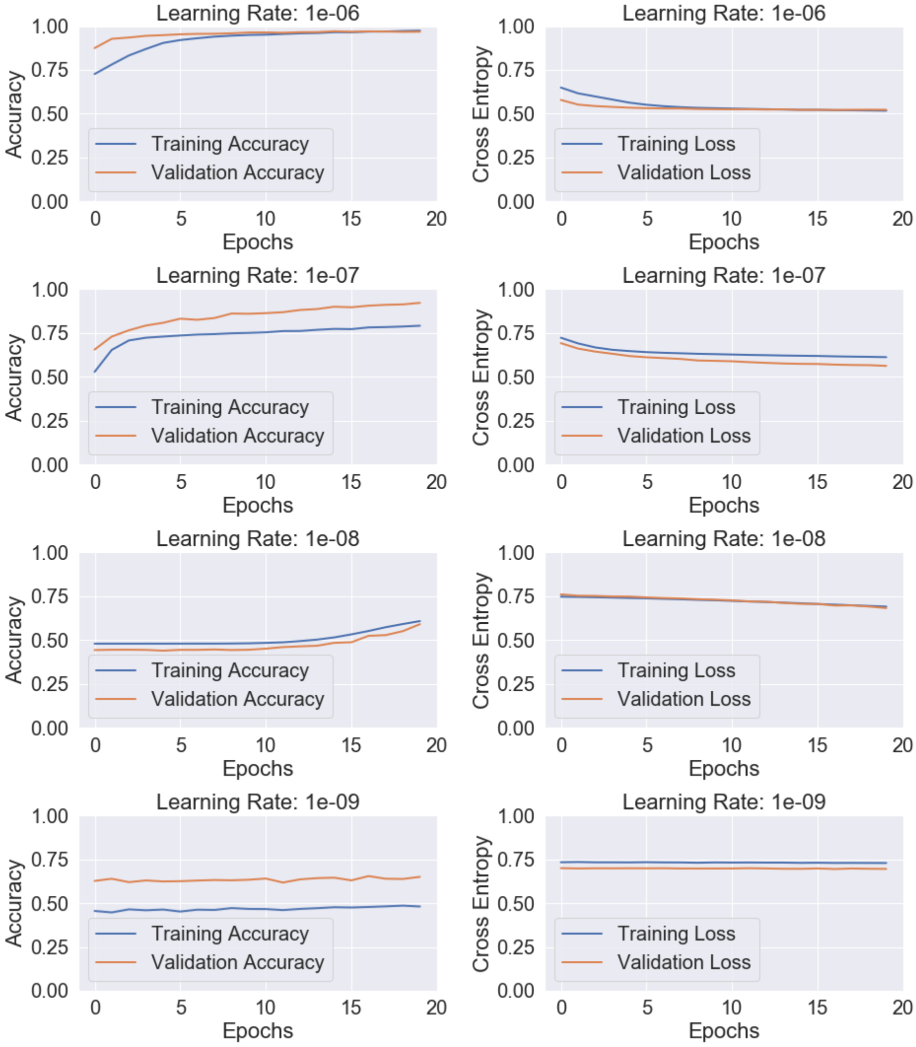

Figure 5.1: Accuracy and Loss during training MobileNetV2

Table 5.1 shows the accuracy, precision, recall and f1-score on the evaluation dataset. Fig. 5.2 shows the confusion matrix. The model with learning rate 1e^{-6} seems to be the clear winner since it has the highest F1-score with 0.94.

Table 5.1: Accuracy, Precision, Recall and F1-Score for MobileNetV2

Learning Rate	Accuracy	Precision	Recall	F1-score
1e^{-6}	0.941	0.933	0.948	0.940
1e^{-7}	0.852	0.854	0.860	0.857
1e^{-8}	0.663	0.712	0.627	0.667
1e^{-9}	0.756	0.851	0.723	0.782

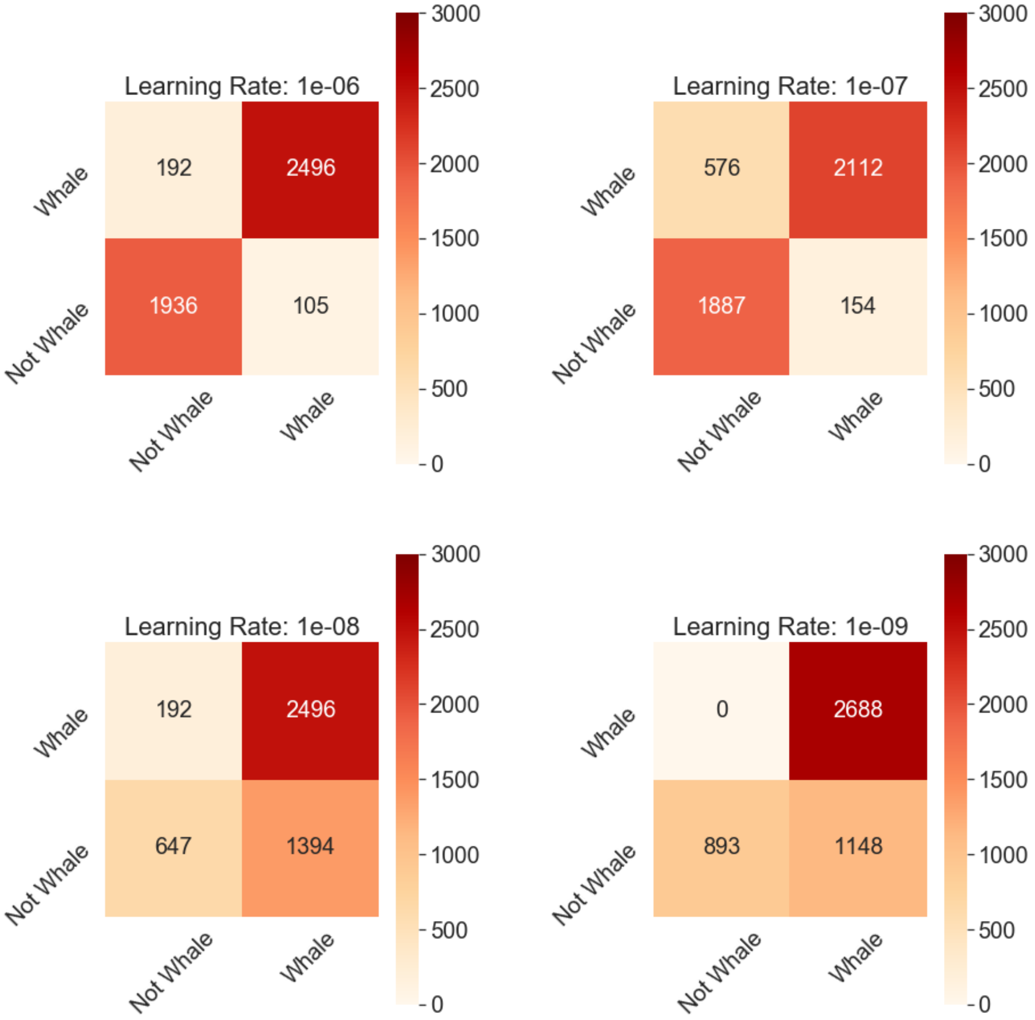

Figure 5.2: Confusion matrices for MobileNetV2.

5.2 Training and Testing results of XceptionV1

In the same way as MobileNetV2, Fig. 5.3, shows the performance of all of the XceptionV1 models trained. Each row of graphs shows the accuracy and loss over epochs of a model with the learning rate mentioned on top of the graph. There are just two rows of graphs on this model since my computer crashed everytime when I tried to run XceptionV1 with learning rates 1e^{-7} and 1e^{-9}.

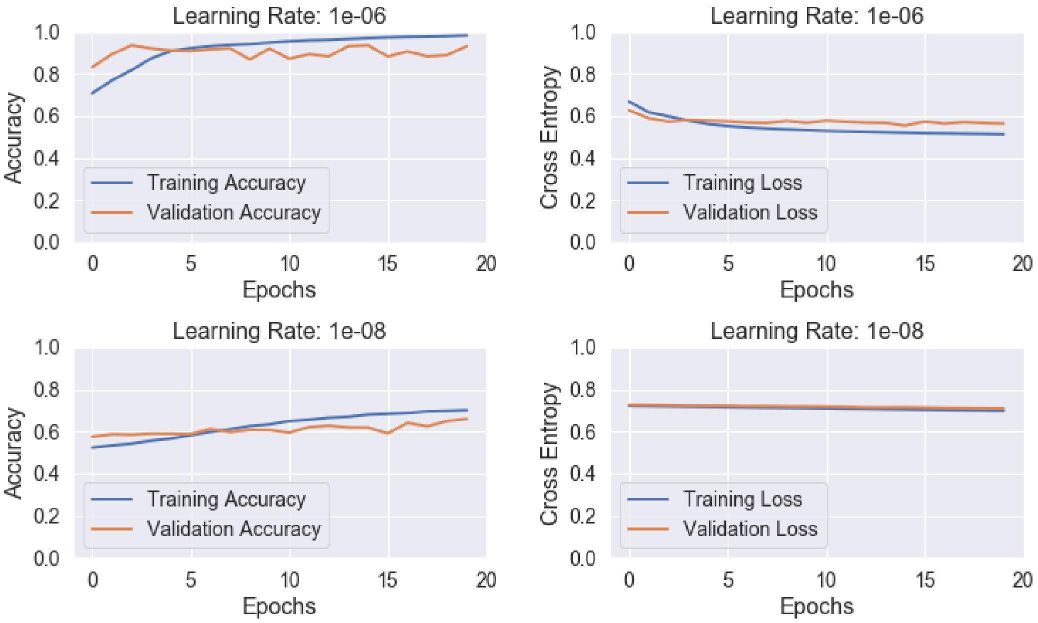

Figure 5.3: Accuracy and Loss during training XceptionV1.

The Table 5.2 and confusion matrix in Fig. 5.4, 1e^{-6} learning rate is the best of XceptionV1 with an accuracy of 0.96 and precision of 0.972.

Table 5.2: Accuracy, Precision, Recall and F1-Score for XceptionV1

Learning Rate	Accuracy	Precision	Recall	F1-score
1e^{-6}	0.960	0.972	0.955	0.963
1e^{-8}	0.554	0.737	0.604	0.664

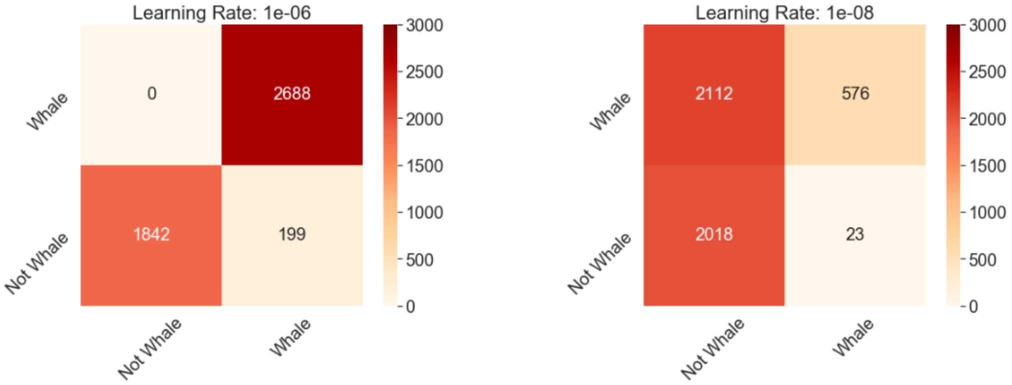

Figure 5.4: Confusion matrices for XceptionV1.

5.3 Training and Testing results of ResNet152V2

Fig. 5.5, shows the performance of the ResNet152V2 models trained. Same as some of the XceptionV1 models crashing my computer, ResNet152V2 crashes my computer when I try to train it with learning rate 1e^{-9}.

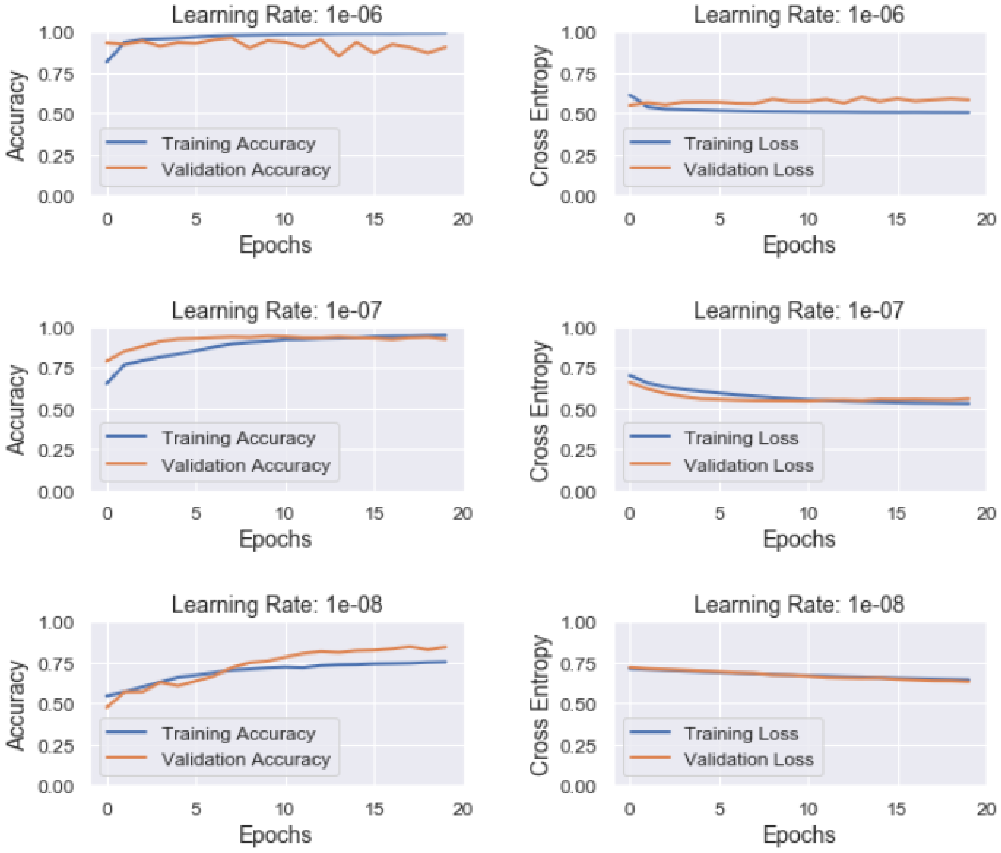

Figure 5.5: Accuracy and Loss during training ResNet152V2.

According to table 5.2 the model with learning rate 1e^{-7} was the highest performing model out of all the models in ResNet with 0.97 accuracy, precision and F1-Score. The Confusion Matrix in Fig. 5.6 further proves this point as this model has the lowest false positive rates.

Table 5.3: Accuracy, Precision, Recall and F1-Score for ResNet152V2

Learning Rate	Accuracy	Precision	Recall	F1-score
1e^{-6}	0.902	0.930	0.890	0.900
1e^{-7}	0.970	0.970	0.960	0.970
1e^{-8}	0.804	0.813	0.813	0.813

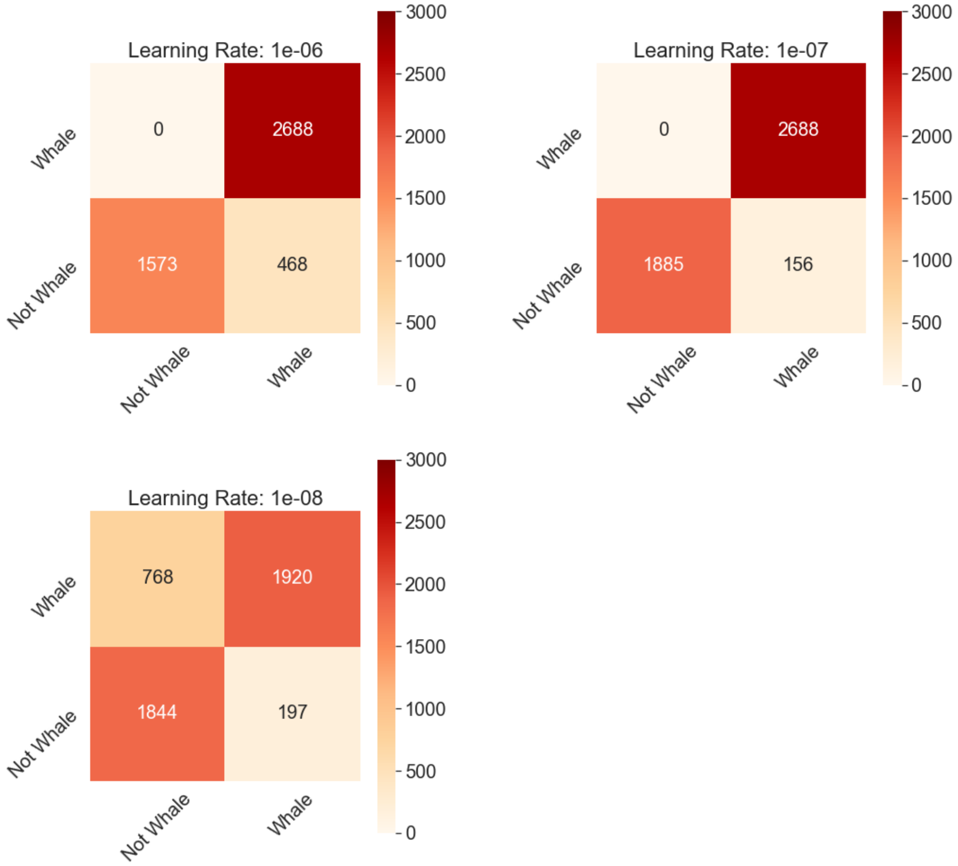

Figure 5.6: Confusion matrices for ResNet152V2.

5.4 Training and Testing results of DenseNet201

Fig. 5.7 shows the performance for DenseNet201. The Figures and table show the results for the models that did not crash my computer.

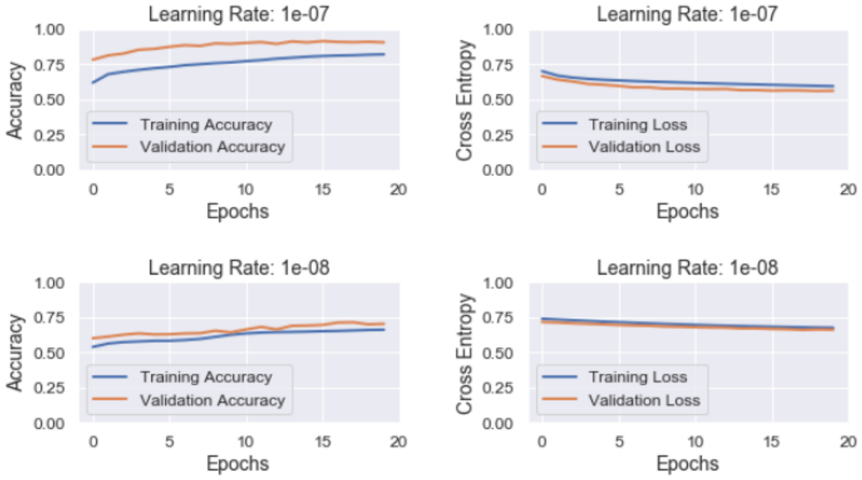

Figure 5.7: Accuracy and Loss during training DenseNet201.

From the table 5.4, the model with learning rate 1e^{-6} was the highest performing model out of all the models in DenseNet with 0.93 accuracy.

Table 5.4: Accuracy, Precision, Recall and F1-Score for DenseNet201

Learning Rate	Accuracy	Precision	Recall	F1-score
1e^{-6}	0.930	0.930	0.930	0.930
1e^{-8}	0.580	0.650	0.620	0.570

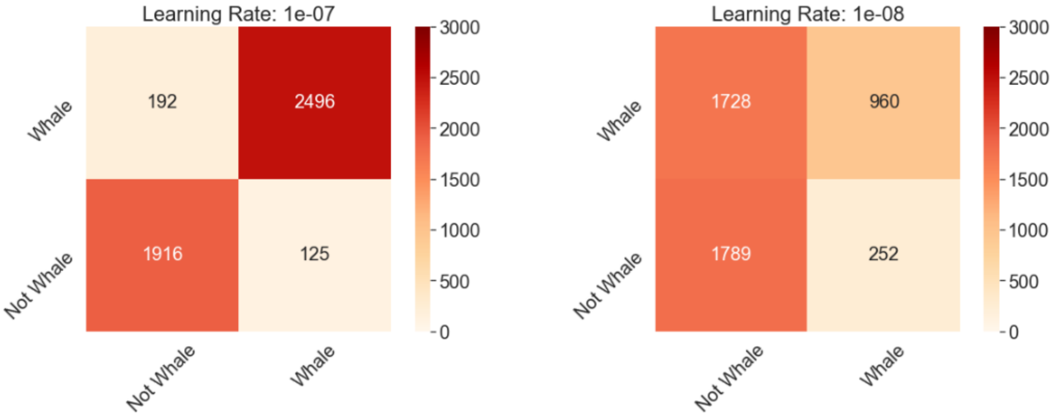

Figure 5.8: Confusion matrices for DenseNet201.

5.5 Model Performance over different Spacial Resolutions

The final part of testing for all of the models was to see how well the models perform on comparatively lower resolution images. In section 4.1.2, I explained how the downsampling was done and in section 4.2.3 I explained how we setup the tests for each of the models. For all the graphs in Figures 5.9 to 5.12 the X-axis represents the spacial resolution, which is the area covered by each pixel of the images and Y-axis represents their accuracy. As expected, they all decrease in accuracy as the spacial resolution increases, however the behaviour is different in each of the cases.

5.5.1 Performance of MobileNetV2 over increasing Spacial Resolutions

The highest performer in the MobileNetV2 models I trained was with learning rate 1e^{-6}. Which is in the top left of Fig. 5.9. It shows a steady decline in accuracy.

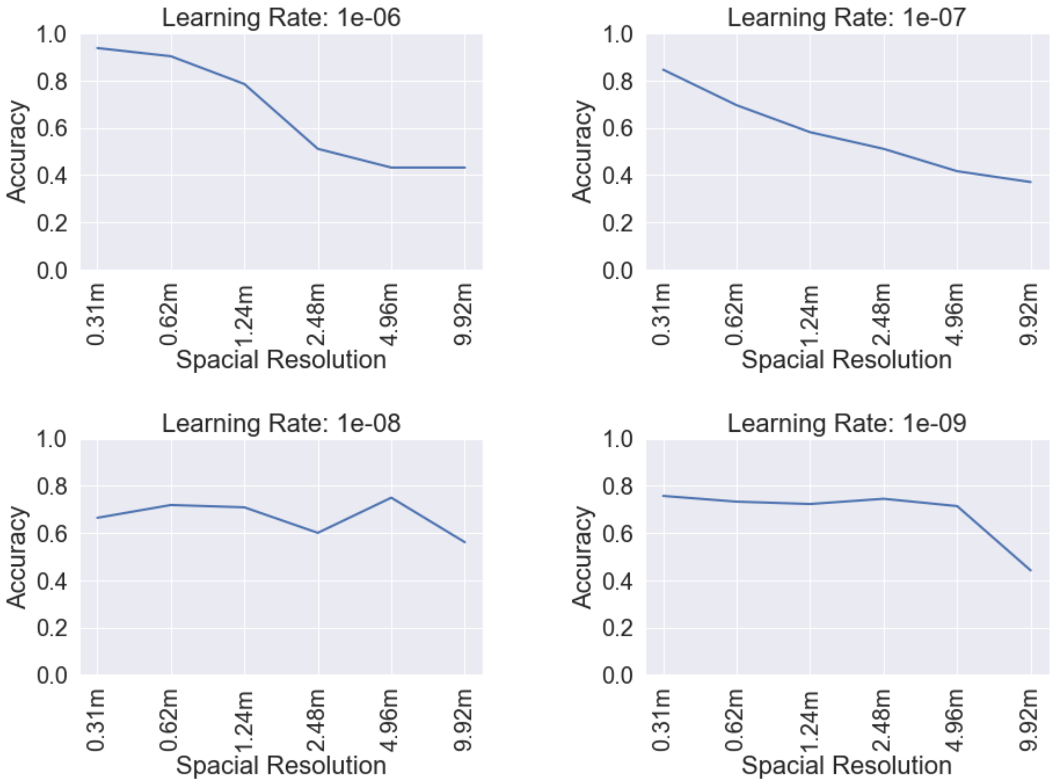

Figure 5.9: MobileNetV2 performance over different Spacial Resolutions.

5.5.2 Performance of XceptionV1 over increasing Spacial Resolutions

The highest performer in the XceptionV1 models I trained was again with learning rate 1e^{-6}, which is in the left of Fig. 5.10. This model had an accuracy of 97%. Its decline in accuracy starts after the first downsampling, so this model can be used to detect whales in 0.62m images. WorldView-2 and WorldView-1 has a spacial resolution of 0.46m. So that means that this model should be able to detect whales just fine in the WorldView-2 and WorldView-1 images just as well as it can detect in WorldView-3 images.

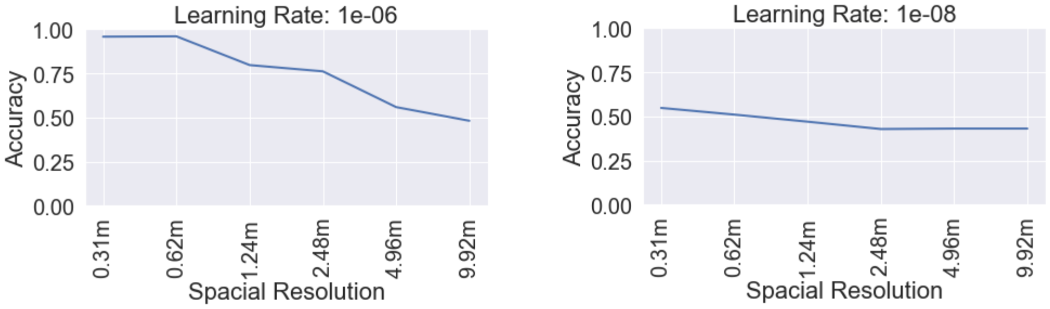

Figure 5.10: XceptionV1 performance over different Spacial Resolutions.

5.5.3 Performance of ResNet152V2 over increasing Spacial Resolutions

With ResNet152V2 the highest performing model was with learning rate 1e^{-7}, which is in the top-right of Fig. 5.11 below. This model shows the same accuracy in 1.24m spacial resolution as 0.31m. The IKONOS was a satellite with 0.82m spacial resolution. In terms of satellite usage lifetimes, it is old enough that it is no longer in service. So according to this data, my ResNet152V2 model can detect objects in images from much older satellites that have a spacial resolution of upto 1.24m.

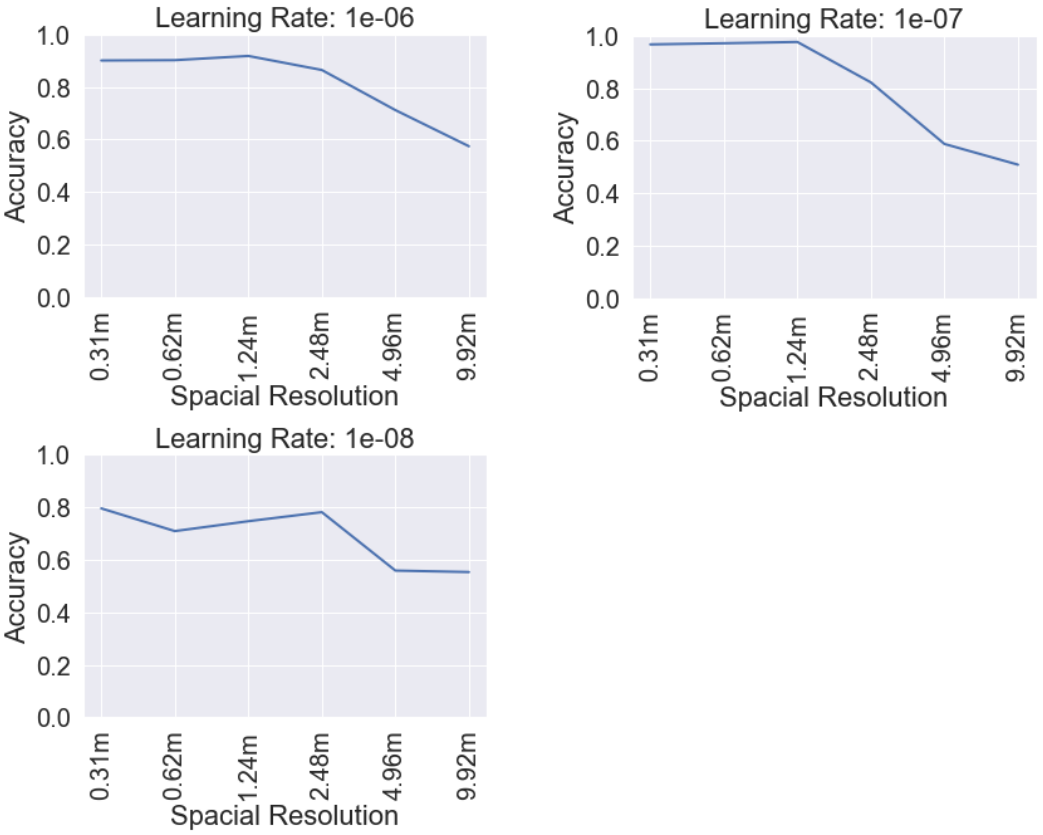

Figure 5.11: ResNet152V2 performance over different Spacial Resolutions.

5.5.4 Performance of DenseNet201 over increasing Spacial Resolutions

For the final model, DenseNet201, the highest performing model was with learning rate 1e^{-6}, which is in the left of Fig. 5.12. It is showing a steady decline in accuracy as spacial resolution increases.

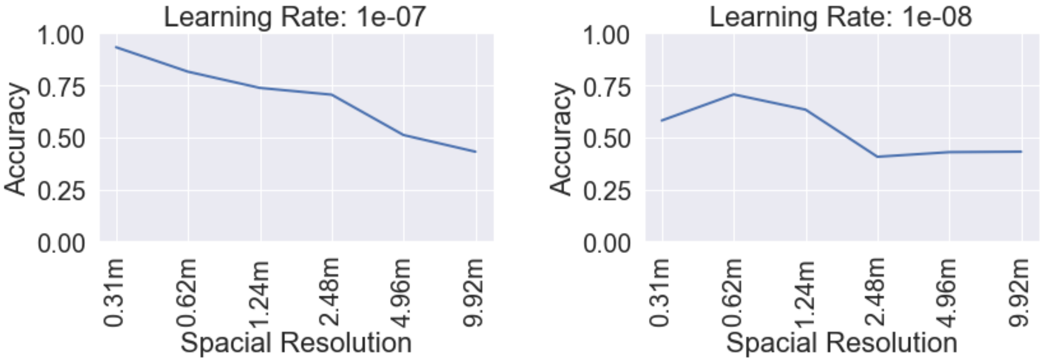

Figure 5.12: DenseNet201 performance over different Spacial Resolutions.

5.6 Object Detection Results

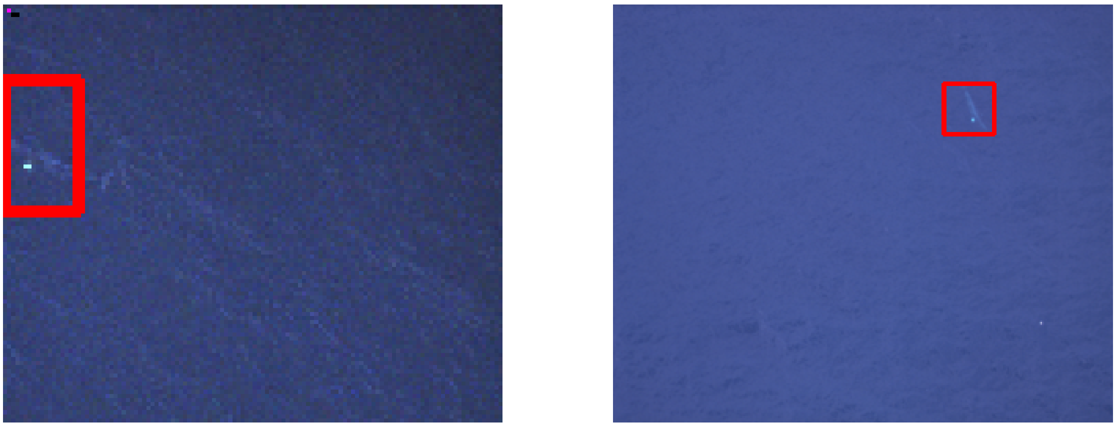

Figure 5.13: Whale Detection with bounding box.

Fig. 5.13 shows some examples of results of the object detection algorithm that was explained in section 4.3. Since Resnet152V2 was the best model, in terms of accuracy, I used it for the final Whale Detection algorithm. As evidenced by the images, the model was successfully able to detect and draw the bounding boxes around the whales.

Since I also wanted to detect in comparatively lower resolution satellite images I also ran the whale detection on downsampled images to see how well they actually perform. As mentioned before, I performed downsampling 5 times until nothing could be distinguished in the image. Fig. 5.14 shows one example of the whale detection algorithm on downsampled images. They are all downsampled from the second image (right) in Fig. 5.13. The object detect works fine for the first downsampled image and then draws a relatively larger bounding box for the second bounding box, suggesting that there will be a slight error margin for 1.24m spacial resolution. The object detection fails for the last three images, which isn't such a surprise since the whale in third and forth image is not very clear and the last image just seems like a flat blue image.

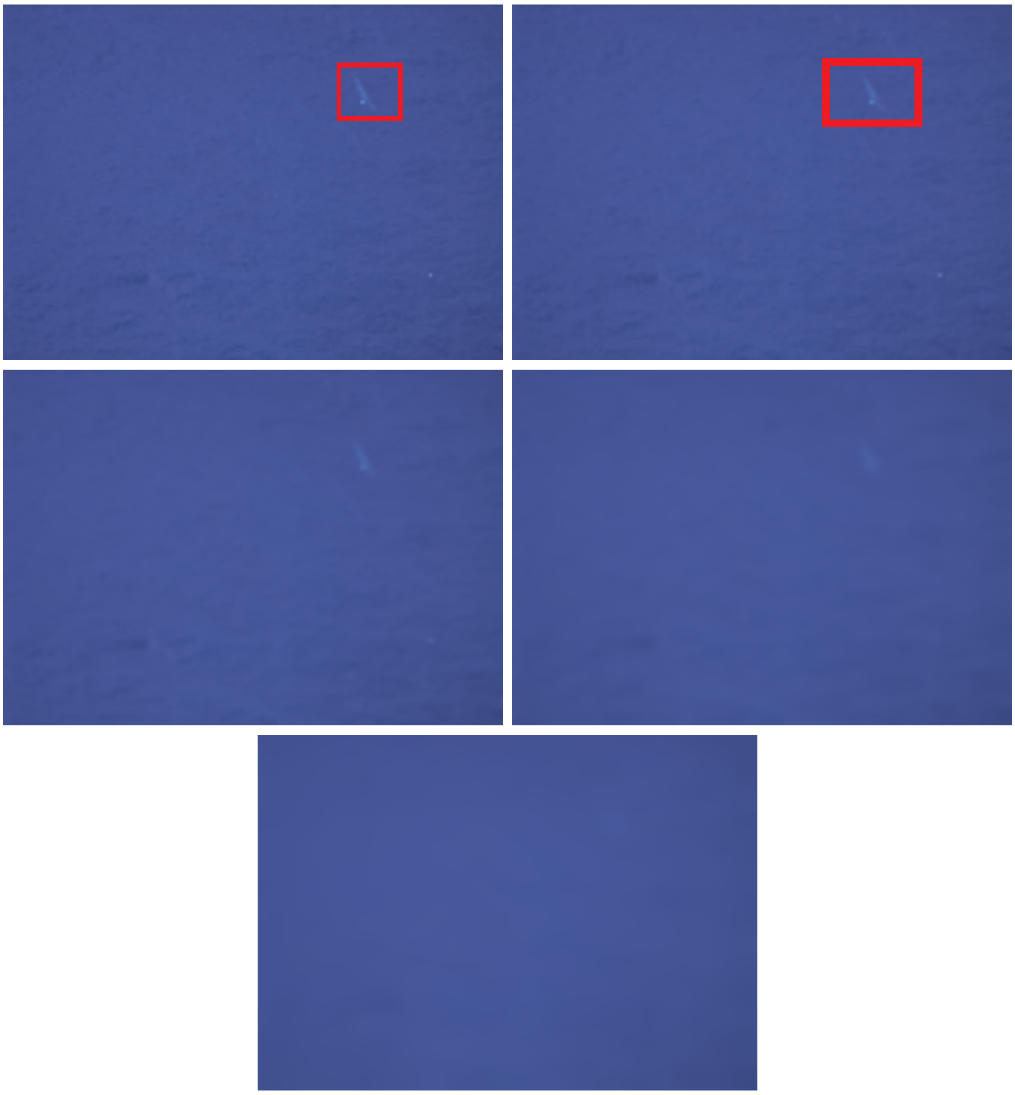

Figure 5.14: Whale Detection on downsampled images. The spacial resolution of the images are Top-left: ~0.62m, Top-right: ~1.24m, Middle-left: ~2.48m, Middle-right: ~4.96m and Bottom image: ~9.92m

Chapter 6

Discussion

This chapter first addresses some limitations within the dataset that I have used and then discusses the achieved results of this thesis.

6.1 Limitations in Dataset

As mentioned before, there are no publicly available satellite images for whales so I had to use aerial images. The aerial images are made up of 3-bands, the RGB bands, but satellite images come in 6-8 bands. In their paper, Emilio Guirado, et al. mentioned that one of these other 5 bands is already known to be more effective for detection in water [34]. This is known as the costal band and is a slightly lower in wavelength than the color blue, so it is invisible to us humans, but it is very useful when trying to identify objects in water. So while the aerial images were great as a simulation to real satellite images, having a dataset from the WorldView-3 would mean that I had 8-bands of information to work with, more than twice as much information as the aerial images.

6.2 Discussion on the Thesis Results

In chapter 5, I went through the results of this thesis. In terms of performance, ResNet had an accuracy of 97% and Xception had and accuracy of 96%, almost the same accuracy so they can be considered the best of the models. When I compared their performance on the downsampled images ResNet can detect with 97% accuracy on upto 1.24m resolution images while Xception starts to decline in accuracy after 0.62m image resolution. So for the purpose of finding whales in satellite images, ResNet152V2 performs the best even in comparatively lower resolution images. The object detection algorithm also works well for detecting whales.

In section 5.6, I showed some examples of detection in both the simulated VHR images and also some downsampled images. It was able to detect whales in the images with 1.24m resolution as expected from the ResNet models performance.

Chapter 7

Conclusion

The objective of this thesis was to experiment with some well-known Deep Learning Architectures in order to find whales in modern VHR satellite images, as well as some comparatively low resolution images and finally to pinpoint the areas with whales. The data proves that ResNet152V2 can detect whales in satellite images up to the spacial resolution of 1.24m.

At the very first chapter, in section 1.3, I wrote down some research questions that this thesis was going to answer. The answers are as follows:

1. Can Deep Learning be used to identify whales in the Very High Resolution images of satellites?
Ans: Yes, they can be used to identify whales. The highest accuracy I had was 97% with ResNet152V2 with a learning rate of 1e^{-7}.

2. Can the same model be used to identify whales in comparatively lower resolution images? If it is achievable, then up to what resolution can whales be detected?
Ans: Yes, they can. Of course they are not all going to show the same performance. But XceptionV1 with 1e^{-6} was able to detect whales in images with up to 0.62m resolution and ResNet152V2 with 1e^{-7} was able to detect whales in images with up to 1.24m resolution before they showed any decline in the accuracy.

3. Can the models be used to highlight areas of interest that contain whales in an image?
Ans: Yes, they can be used to pinpoint the areas containing whales. In Section 5.6, I showed how the algorithm performs.

List of Figures

List of Tables

Listings

Appendix A

Supplimentary Information

A.1 Codes and Dataset Download

The codes and dataset can be downloaded from this link.

A.2 Known Satellite Images with Whales

The table here is for Satellite images in DigitalGlobe that we had collected from previous papers we have read during the course of this thesis. By going to this link anyone can search for the images by ID by clicking on "Areas of Interest -> Search by Image ID". These are copyrighted images and owned by Maxar, and the website will only show a much smaller resolution image as a preview. Getting the original image requires signing up and buying the images from Maxar.

Table A.1: List Of Satellite images found

IDs	Location	Area(Km2)
104001001D325700	Pelagos, West of Italy	944
104001001D325700	Pelagos, West of Italy	1666
104001001E7B8900	Pelagos, West of Italy	1091
104001001E19F000	Pelagos, West of Italy	1047
1040010006C2B700	Maui Nui, Honolulu	1542
1040010029924200	Maui Nui, Honolulu	1182
10400100032A3700	Peninsula Valdes, Argentina	1767
1040010003121A00	Peninsula Valdes, Argentina	2066
104001002959ED00	Ignacio, Baja California Sur, Mexico	348

Bibliography

[1] Whale Facts. Why are whales important? URL https://www.whalefacts.org/why-are-whales-important.

[2] Seeker. Whale-watching, a booming business. URL https://www.seeker.com/whale-watching-a-booming-business-discovery-news-1766490722.html.

[3] UN Environment Programme. Protecting whales to protect the planet. URL https://www.unenvironment.org/news-and-stories/story/protecting-whales-protect-planet.

[4] K. M. Stafford D. K. Mellinger R. P. Dziak Nieukirk, S. L. and C. G. Fox. Low-frequency whale and seismic airgun sounds recorded in the mid-atlantic ocean. *Journal of the Acoustical Society of America*, 2004. 115:1832–1843.

[5] K. M. Stafford S. E. Moore R. P. Dziak Mellinger, D. K. and H. Matsumoto. An overview of fixed passive acoustic observation methods for cetaceans. 2007. 20(4):36–45.

[6] W. F. Perrin B. Würsig Phillip J.Clapham, C. Scott Baker and J. G. M. Thewissen. *Encyclopedia of marine mammals*. Academic Press, Elsevier Science Publishing Co Inc, San Diego, United States, third edition, 2009. ISBN 012373553X, 9780123735539.

[7] R. R. Reeves and T. D. Smith. A taxonomy of world whaling: Operations, eras, and data sources. 2003. 03-12. 28 pp.

[8] Vanderlaan Taggart. Vessel collisions with whales: the probability of lethal injury based on vessel speed. 2008. URL http://www.phys.ocean.dal.ca/~taggart/Publications/Vanderlaan_Taggart_MarMamSci-23_2007.pdf.

[9] Vanderlaan Taggart. Whales entangled in fishing lines: What can be done? 2007. URL https://www.sciencedaily.com/releases/2007/04/070426143431.htm.

[10] Connor Bamford Laura Gerrish Jennifer A. Jackson Hannah C. Cubaynes, Peter T. Fretwell. Whales from space: Four mysticete species described using new vhr

satellite imagery. doi: 10.1111/mms.12544. URL https://onlinelibrary.wiley.com/doi/full/10.1111/mms.12544.

[11] Wikipedia. Laika, . URL https://en.wikipedia.org/wiki/Laika.

[12] Wikipedia. Sun-synchronous orbit, . URL https://en.wikipedia.org/wiki/Sun-synchronous_orbit.

[13] DigitalGlobe. Worldview-3 datasheet. URL https://www.spaceimagingme.com/downloads/sensors/datasheets/DG_WorldView3_DS_2014.pdf.

[14] Maxar. Worldview-3: Three things you should know. URL https://blog.maxar.com/earth-intelligence/2014/worldview-3-three-things-you-should-know.

[15] Satellite Imaging Corporation. Worldview-2 satellite sensor. URL https://www.satimagingcorp.com/satellite-sensors/worldview-2/.

[16] Expert System. What is machine learning? a definition. URL https://expertsystem.com/machine-learning-definition.

[17] Tom Mitchel. *Machine Learning*. McGraw-Hill, illustrated edition, 1997. ISBN 0071154671, 9780071154673. URL http://www.cs.cmu.edu/~tom/mlbook.html.

[18] Jason Brownlee. What is deep learning? URL https://machinelearningmastery.com/what-is-deep-learning/.

[19] Sumit Saha. A comprehensive guide to convolutional neural networks. URL https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[20] Wikipedia. Statistical classification, . URL https://en.wikipedia.org/wiki/Statistical_classification.

[21] Wikipedia. F1-score, . URL https://en.wikipedia.org/wiki/F1_score.

[22] Menglong Zhu Andrey Zhmoginov Liang-Chieh Chen Mark Sandler, Andrew Howard. Mobilenetv2: Inverted residuals and linear bottlenecks. 2018. URL https://arxiv.org/abs/1801.04381.

[23] Chi-Feng Wang. A basic introduction to separable convolutions. URL https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728.

[24] François Chollet. Xception: Deep learning with depthwise separable convolutions. 2016. URL https://arxiv.org/abs/1610.02357.

[25] Shaoqing Ren Jian Sun Kaiming He, Xiangyu Zhang. Deep residual learning for image recognition. 2015. URL https://arxiv.org/abs/1512.03385.

[26] Laurens van der Maaten Kilian Q. Weinberger Gao Huang, Zhuang Liu. Densely connected convolutional networks. 2016. URL https://arxiv.org/abs/1608.06993.

[27] Great Images in NASA. First picture from explorer vi satellite. URL https://web.archive.org/web/20091130171224/http://grin.hq.nasa.gov/ABSTRACTS/GPN-2002-000200.html.

[28] Wikipedia. Satellite imagery - uses, . URL https://en.wikipedia.org/wiki/Satellite_imagery#Uses.

[29] National Geography. Oceanography. URL https://www.nationalgeographic.org/encyclopedia/oceanography/#:~:text=Oceanography%20is%20the%20study%20of,current%20condition%2C%20and%20its%20future.

[30] NOAA. How are satellites used to observe the ocean? URL https://oceanservice.noaa.gov/facts/satellites-ocean.html.

[31] Forcada J Fretwell PT, Staniland IJ. Whales from space: Counting southern right whales by satellite. 2014. URL https://doi.org/10.1371/journal.pone.0088655. PLoS ONE 9(2): e88655.

[32] ENVI. A software solution for processing and analyzing geospatial imagery. URL https://www.harrisgeospatial.com/Support/Self-Help-Tools/Help-Articles/Help-Articles-Detail/ArtMID/10220/ArticleID/18079/8406.

[33] ArcGIS. Mapping and analysis. URL https://www.esri.com/en-us/arcgis/products/arcgis-online.

[34] Marga L. Rivas Domingo Alcaraz-Segura Emilio Guirado, Siham Tabik and Francisco Herrera. Whale counting in satellite and aerial images with deep learning. 2019. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6776647/.

[35] Humphries G Nehls G Hoschle C Kosarev V et al Borowicz A, Le H. Aerial-trained deep learning networks for surveying cetaceans. 2019. URL https://doi.org/10.1371/journal.pone.0212532. PLoS ONE 14(10): e0212532.

[36] Pacific Whale Foundation. Whale and dolphin tracker. URL https://www.pacificwhale.org/research/citizen-science/whale-and-dolphin-tracker/.

[37] University of Tromso. Whaletrack program. URL https://uit.no/prosjekter/prosjekt?p_document_id=504905.

[38] Google AI blog. Acoustic detection of humpback whales using a convolutional neural network. URL https://ai.googleblog.com/2018/10/acoustic-detection-of-humpback-whales.html.

[39] Ross Girshick Ali Farhadi Joseph Redmon, Santosh Divvala. You only look once: Unified, real-time object detection. 2015. URL https://arxiv.org/abs/1506.02640.

[40] Fengbo Ren Yixing Li. Light-weight retinanet for object detection. 2017. URL https://arxiv.org/abs/1612.03144.

[41] Ross Girshick Kaiming He Bharath Hariharan Serge Belongie Tsung-Yi Lin, Piotr Dollár. Feature pyramid networks for object detection. 2019. URL https://arxiv.org/abs/1905.10011.

[42] Antonio Pertusa Antonio-Javier Gallego and Pablo Gil. Automatic ship classification from optical aerial images with convolutional neural networks. *Remote Sensing*, 10 (4), 2018. ISSN 2072-4292. doi: 10.3390/rs10040511.

[43] OpenCV. Image pyramids. URL https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_pyramids/py_pyramids.html.

[44] Tensorflow. Imagedatagenerator. URL https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator.