SHIP DETECTION IN SATELLITE IMAGES USING DEEP LEARNING

THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

MASTER OF TECHNOLOGY
IN
TELECOMMUNICATION ENGINEERING

Submitted By

Shruti Mary Mathew
20EC4104

under the supervision of

Dr. Gautam Kumar Mahanti

Department of Electronics and Communication Engineering
National Institute of Technology, Durgapur
West Bengal, India
May 2022

NATIONAL INSTITUTE OF TECHNOLOGY

Mahatma Gandhi Avenue

Durgapur - 713209, INDIA

CERTIFICATE OF RECOMMENDATION

This is to recommend that the work undertaken in this thesis entitled "*Ship Detection in Satellite Images using Deep Learning*" has been carried out by *Shruti Mary Mathew* under my supervision and may be accepted in partial fulfilment of the requirement for the degree of Master of Technology in Telecommunication Engineering.

Dr. Gautam Kumar Mahanti
Project Supervisor,
ECE Department,
NIT Durgapur

Dr. Durbadal Mondal
Head of the Department,
ECE Department,
NIT Durgapur

NATIONAL INSTITUTE OF TECHNOLOGY

Mahatma Gandhi Avenue

Durgapur - 713209, INDIA

CERTIFICATE OF APPROVAL

The foregoing thesis is hereby approved as a creditable study of engineering subject to warrant its acceptance as a pre-requisite to obtain the degree for which it has been submitted. It is understood that by this approval the undersigned don't necessarily endorse or approve any statement made, opinion or conclusion drawn therein but approve the thesis only for the purpose for which it is submitted.

Board of Examiners

1. ------------------------

2. ------------------------

Project Supervisor

NATIONAL INSTITUTE OF TECHNOLOGY

Mahatma Gandhi Avenue

Durgapur - 713209, INDIA

UNDERTAKING

I, Shruti Mary Mathew , Roll No: 20EC4104 of M.Tech (Telecommunication Engineering) hereby declare that my M. Tech. Project / Thesis titled "Ship Detection in Satellite Images using Deep Learning" is my own contribution. I declare that I have not indulged in any form of plagiarism to carry out the project and also while writing this report. The work or ideas of other authors which are utilized in this report has been properly acknowledged and mentioned in the Bibliography. I undertake total responsibility if any traces of plagiarism is found at any later stage. In such a case my degree will be cancelled automatically.

Shruti Mary Mathew
20EC4104
M.Tech (Telecommunication Engineering),
ECE Department,
NIT Durgapur

Dedicated to

My Parents

Aknowledgment

I would like to express my sincere gratitude to my respected supervisor Dr. Gautam Kumar Mahanti, Professor, Electronics and Communication Department, National Institute of Technology, Durgapur for his continuous support and motivation. Besides my supervisor, I would like to extend my special thanks to Dr. Minita Mathew, Manager and Mr. Hariom Gautam, Consultant, in AI and Cognitive Science Department, Datamatics Global Services Limited for their guidance in carrying out some part of this project. I sincerely thank all the officials and other staff members of Datamatics Global Services Limited, Bangalore, Karnataka, India who rendered their help.

Abstract

The recent increase in satellite imagery has attracted great interest focusing the research to this area. Automatic Ship detection is one of the research areas that has various applications such as maritime security and traffic control.

This project follows two approaches with two different types of dataset to come up with a solution to the detection of ship using deep learning. The first approach was implemented using an image classifier and selective search algorithm to find the region of interest with the primary aim of understanding the detection and classification problem in depth. The first approach used a classification dataset which consisted of images with and without ship. The second approach was implemented using YOLOv3, Faster RCNN and SSD algorithm. The comparative analysis study has been conducted based on speed of detection and Mean Average Precision of each models. The results obtained from both approaches are shown. The detection dataset was used for this approach which consisted of more than 1 ship in an image.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	:	**Description**
CNN	:	Convolutional Neural Network
FRCNN	:	Faster Region-based Convolutional Neural Network
SSD	:	Single Shot Detector
YOLO	:	You Only Look Once
AP	:	Average Precision
MAP	:	Mean Average Precision

Chapter 1

INTRODUCTION

This chapter starts with the motivation behind doing this thesis, followed by related work done by other authors and objectives which was achieved.

1.1 Motivation

Computer vision has been a part of artificial intelligence that trains PCs to decipher and comprehend the world. Machines would be able to distinguish and localize objects then, at that point, respond to what they "see" utilizing digital pictures from cameras, recordings, and deep learning models. Object detection is a type of computer vision which is used to locate and classify objects in an image or video. In this project, ship is the object to be detected from satellite images. Ship detection is used in various applications such as dynamic harbour surveillance, traffic monitoring, fishery management, sea pollution monitoring, the defence of territory and naval battles, etc.

As per the Index of objects launched into Outer Space, there were 7,389 individual satellites in Space at the end of April, 2021. There has been an increase of 27.97% compared to 2020 has been noticed. The Index of Objects Launched is maintained by the United Nations Office for Outer Space Affairs.

Remote sensing is the obtaining of data about an object or peculiarity without in touch with the object, rather than in-situ or on location perception. The term is applied particularly to obtain data about the Earth and different planets. Remote sensing is utilized in various fields, including topography and most Earth science disciplines; it additionally has military, intelligence, business, financial, planning, and humanitarian applications, among others. In current use, the expression "remote sensing" by and large depends on the utilization of satellite or airplane-based sensor advances to distinguish and classify objects on Earth. It could be parted into "active" remote sensing (when a signal is sent by the satellite and its reflection is detected by using sensor) and ”passive” remote detecting (when the reflection of light from sun is recognized by the sensor). Satellite images being publicly available has increased the researches in this field.

1.2 Related Works

According to Kanjir et al.[2], commonly used sensors in marine surveillance applications are optical, infrared, and radar sensors. Radar is a common device that has been used in ship monitoring and detection. The satellite-based sensor is also popular for ship detection, which necessitates remote sensing, continuous monitoring, and frequent data collection.

Deep learning-based algorithms for classification are becoming increasingly popular in research. Lin et al.[3] proposed a rotational-invariant detection method for detecting objects in remote sensing images. Their invariant characteristics have resulted in high accuracy in detecting tough objects in remote sensing images. Huang et al. presented their random forest method, which has a faster training speed than the conventional approach while maintaining the same accuracy performance.

Yokoya and Iwasaki proposed a method based on sparse representation and Hough voting (SR-Hough). This method detects instances of an object class or a specific object in remote sensing images. Prasad et al. presented a study that investigates various challenges in maritime surveillance, such as occlusion, variations in orientation and scale, a large number of object classes, and weather changes. Yu et al. also presented a machine learning approach for detecting small and dim objects in a Forward Looking Infrared (FLIR) image using a context-driven Bayesian saliency model.

Traditional machine learning techniques, on the other hand, necessitate the explicit definition of object features. Learning-based features have drastically changed the conventional approach, thanks to recent advancements in deep learning fields and computer vision.

Krizhevsky et al, the ImageNet Large Scale Visual Recognition Challenges (ILSVRC) winner, popularised the use of deep Convolutional Neural Network (CNN) for image recognition, detection, and especially classification. Since then, much research on CNNs has been conducted, with advancements in various factors such as activation function, optimization technique, network architecture, and regularisation mechanism.

Tang et al.[3] used SPOT-5 dataset images to detect and classify ships using a combined compressed-domain framework, deep neural network, and extreme machine learning. Zhang et al. propose the sequential convolutional neural network

(S-CNN) method, which combines CNN with the saliency detection method, to achieve the same goal. They discovered that their method outperforms the R-CNN. A concise overview of the most recent saliency object detection method can be found. Liu et al. used Google Earth images to implement the CNN approach for ship classification and reported better results than support vector machine and standard neural network. The work in, on the other hand, took the same approach but specialised in navy applications.

Tang et al.[3] used SPOT-5 dataset images to detect and classify ships using a combined compressed-domain framework, deep neural network, and extreme machine learning. Zhang et al. propose the sequential convoluting method to achieve the same goal. The amount of data stored in the modern era is growing by the day. The researchers are constantly working on improvements to existing algorithms, which usually occur as the number of datasets grows. A typical deep learning classifier is made up of several layers of CNN between the input and output layers, allowing for complex nonlinear information processing.

Chua et al. compared three classical machine learning algorithms, histogram of oriented gradient (HOG), exemplar-SVM, and latent-SVM, to determine their specific advantages, and discovered that exemplar SVM is good for specificity measure. Yu et al. introduced a detection approach that uses a fully convolutional neural network (FCNN) segmentation method and then detects the object through bounding box regressions, where the class is labelled by CNN classifier, to overcome the issue of scale variance and complicated background.

In 2012, Krizhevsky et al. proposed a deeper and broader CNN model than the original LeNet, and they won the most difficult ImageNet challenge (ILSVRC). Against all traditional machine learning algorithms, their AlexNet achieved state-of-the-art recognition accuracy. In 2013, Zeiler and Fergus improved AlexNet by tweaking network parameters, and they won the 2013 ILSVRC. Simonyan and Zisserman investigated the effect of convolutional on large scale images in 2014. GoogleNet, the 2014 ILSVRC winner, introduced several parallel CNNs with varying kernel sizes.

1.3 Objectives and scope

The main goal of this project is to implement and experimentally evaluate and compare deep neural networks for detecting ships from satellite images. Object detection is one of the problems of computer vision which is discussed in detail in Chapter 2.The intelligent detection and recognition of ships is quite important for maritime security and civil management.
The objectives can be defined as:

1. Understand neural networks technique for computer vision.
2. Build an object detector from scratch using selective search algorithm and finding region of interest (ROI).
3. Build using existing models and fine tune to get accurate results (transfer learning).
4. Present a comparitive analysis of three famous object detection models i.e. YOLOv3, Faster RCNN, SSD.

Chapter 2

UNDERSTANDING

This chapter takes a top-down approach in presenting the current problem. Understanding the data is an important step in machine learning. The chapter starts with the explanation of machine learning and deep learning followed by brief introduction to object detection. The core of object detection i.e., Convolutional Neural Network is explained. The chapter ends with dataset review and dataset anotation.

2.1 Machine Learning and its types

Machine learning is an AI application that enables computers to learn on their own and complete tasks without the need for human intervention. Machine learning techniques have numerous applications in the field of computer vision. Machine learning has simplified the formulation of some of the most difficult problems. Several computer programmes that were previously written by humans, sometimes by hand, are now programmed without any human input thanks to the use of machine learning. Machine learning has become increasingly common in our daily lives in recent years, thanks to a remarkable increase in the availability of massive amounts of data and the affordability of processing resources.

2.1.1 Supervised learning:

Supervised learning is the most important class of machine learning algorithms. These algorithms, as the name implies, necessitate direct supervision. The data labeled/annotated by humans is spoon-fed to the algorithm in this type of learning. The classes and locations of the objects of interest are contained in this data. After the training process is completed, the algorithm learns from the annotated data and predicts the annotations of new data that was previously unknown to the algorithm.

2.1.2 Unsupervised learning :

Unsupervised learning occurs when an algorithm attempts to learn and identify useful properties of classes from annotated data without the assistance or intervention of a human. Unsupervised learning algorithms that are commonly used include the Apriori algorithm, K-means clustering, and others.

2.1.3 Reinforcement Learning:

In this type of learning, the machine is allowed to continually train itself through trial and error. As a result, the machine learns from previous experience and attempts to capture the most accurate knowledge possible [18]. Some examples of reinforcement learning include the Markov Decision Process, Q-learning, Temporal Difference, and others.

2.2 Deep Learning

It is a sub field of machine learning. It deals with algorithms which are inspired by the working of brain. It actually tries to replicate the work of brain. Like the nervous system which connected by neurons and passes information, deep learning also has connected neurons and passes information. Deep learning models work in layers and a basic model consists of at least three layers. Each layer gets some information from previous layer and is passed onto the next layer. A typical model is shown in fig:2.2:

Neural networks can be thought of as classifiers which extract features from a raw data. The intermediate neurons in the hidden layer train and learn using optimization methods. The user gives the cost function and loss function to be optimized for and accordingly the parameters are updated. The updating of weight takes place by propagating the error back into network. This is known as back-propagation algorithm. There are basically three types of supervised deep learning networks- Artificial Neural Network, Recurrent Neural Network and Convolutional Neural Network. CNN is basically used for images.

2.3 Object Detection

Object detection involves detecting objects from specified scenes or images. Before the deep learning era, the methods of object detection were established using mathematical models. At present, the common classical methods in object detection are as follows: Hough transform method, frame-difference method, background subtraction method, optical flow method, sliding window model method

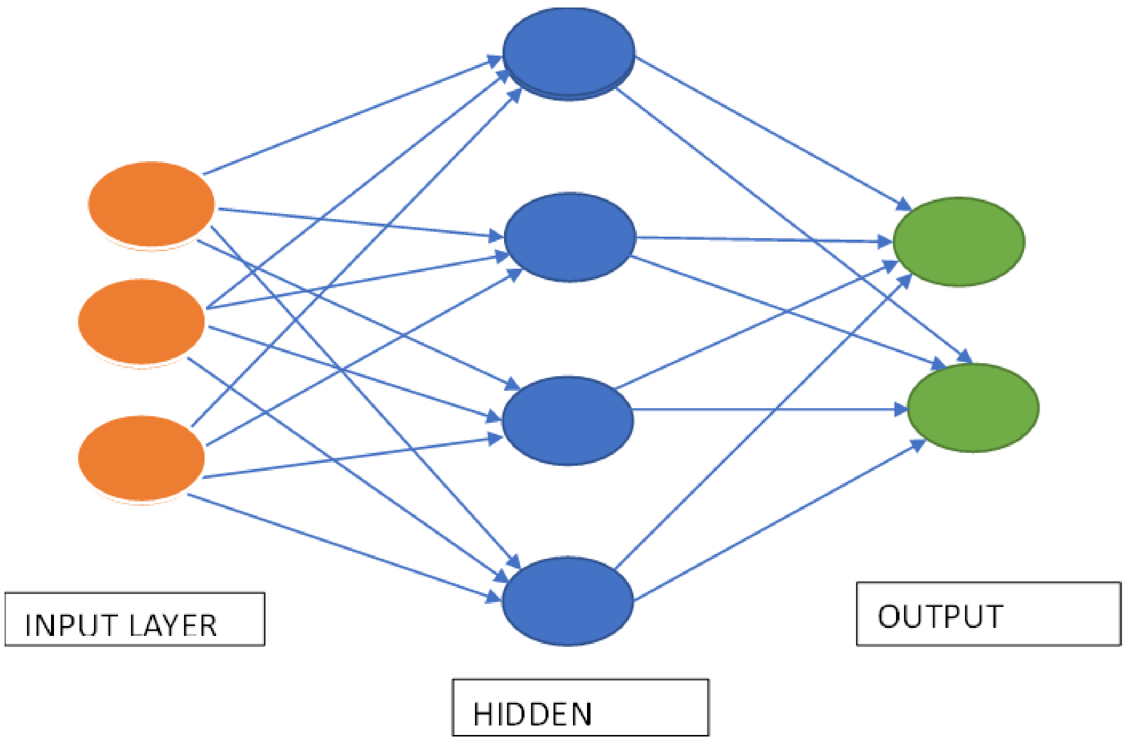


Figure 2.1: Simple neural network

and deform-able part model method. Object detection is combination of two tasks- Image classification and object localization to detect multiple objects in a single image. Classification and localization are generally used for a single object in an image.

Object detection has gone through mainly two historical periods i.e traditional object detection (before 2014) and deep learning based detection (after 2014).This is an important task in computer vision which answers the question what objects are where. From the application point of view, object detection are grouped into two classes- general object detection and detection applications where the former is used for detecting various objects in an image and compare different detecting methods and the latter one is application specific such as text detection, pedestrian detection. This project is also based on one of detection application.

2.4 Back Propogation

The idea of neural network is old even if they have contributed a lot in development in recent years. The neural networks, formerly known as 'perceptrons,' have been used since the 1940s. They were not as popular as they are now because they were single-layered and required a lot of computer power and data, both of which were scarce at the time. They rose to prominence as a result of the invention of a process known as 'Backpropagation.'The weights are updated based on difference in result obtained from previous steps on propagating it backwards. Rumelhart et al. were the first to propose the approach in 1986. If the result differs from the intended output, networks can use this strategy to change the weights of hidden layers. To modify the weights according to the demand, the error is calculated and backpropagated to all levels of the network.

2.5 Convolutional Neural Network(CNN)

Convolutional Neural Network (CNN) is a type of deep neural networks which is mainly used in recognition of image, image classification, object detection, etc. The positive improvements in computer vision with deep learning has been developed with time particularly over one algorithm i.e., a convolutional neural network. The CNN based models are used everywhere such as Google uses it for photo search, Amazon uses it for product recommendation and Facebook for photo tagging.

In CNN, an image is given as input and importance is assigned to various features found in that image. This helps in differentiating one from another. The processing before giving into model required in CNN is much lesser as compared to other classification algorithms.

A CNN consist of three layers: a convolutional layer, pooling layer, and fully connected layer.

- Convolutional layer:

A convolutional layer represents the core of a CNN, and it is where the majority of computation happens. It requires three components, which are input data, a filter, and a feature map. Assume that the input is a colour image, which is made up of pixels in 3D. This means that the input will have three dimensions, which correspond to RGB. The feature detector, also known as a kernel or a filter, moves across the receptive fields of the image to determine whether the feature is present. The process is called convolution. The feature detector uses two-dimensional (2-D) weighted arrays of weights that represent parts of the image. While their dimensions can vary, the filter size is typically a 3x3 matrix; the receptive field size is also determined by the filter size. After shifting by a step, the filter repeats the process until the kernel has swept the entire image. A feature map, activation map, or convolved feature is the final output of the series of dot products from the input and the filter.

• Pooling Layer:
The pooling layer is placed in between two convolutional layers. It is used to achieve shift invariance by reducing the resolution of the feature maps. Average pooling and maximum are widely used pooling methods. Max pooling gives the most prominent feature present in a patch of the feature map, whereas average pooling gives the average of the features present in the patch.

• Fully connected layer:
Fully connected layers receive their input from the final pooling or convolutional layer, which is flattened and then fed into the fully connected layer.

2.6 Hyperparameters

The selection of hyperparameters is a key aspect in developing a NN architecture. Before the actual training (optimization) process, hyperparameters are variables that are set to specified values. There are several approaches for determining these values:

1. Manual: Hyperparameters are established by hand, usually by combining existing issue information with educated guesses regarding parameter values. Parameters are then modified as necessary, until a usable set of parameters are found.

2. Search algorithms: To find suitable hyperparameter ranges, a grid search or random search technique might be used. After that, the network is trained on several models using all of the parameter combinations that are accessible in these ranges. A random search technique is suggested here because it has been found to work better than other methods in general.

3. "Hyper" Optimization: The goal here is to develop an automated method for optimising the model's performance based on the task. The generalisation performance of a network is modelled in such a way that the search algorithm's parameter selection after an experiment is optimised.

In the training phase, one of three ways is typically used to provide data to the neural network:

1. Batch Gradient Descent: The gradient of the cost function is computed across the entire dataset.

2. Mini-Batch Gradient Descent: A portion of the training dataset (referred to as a mini-batch) is fed into the network, and each mini-batch is updated separately.

3. Stochastic Gradient Descent: For each training example, parameter adjustments are conducted.

The learning rate is defined as the rate at which the parameters' gradient updates occur in the gradient direction. Model convergence takes a long period when this rate is too low. If is too great, however, the model diverges, and the loss may fluctuate forever. To ensure optimal learning, an initial learning rate of 0 is specified first (0:01 is a widely acknowledged norm), and then the rate is updated periodically by scaling with a decay factor, based on the mini-batch size and number of iterations.

-Mini batch sizes:

Mini-batch is preferred over batch and stochastic gradient descent updation rules because it combines the benefits of both while reducing the drawbacks. Following that, the size of the mini-batch to be employed is determined based on the available processing resources.

-Regularization:

Setting regularisation strength and p for the dropout probability can be done with the use of a validation set. During training, the model is evaluated on the validation set to determine the regularisation strength. is usually determined by the loss function and varies between 10-3 and 104. Dropout is kept to a reasonable default of 0:5, which has been shown to be effective.

-Initialization of the weight:

The starting approach for the weight matrices has a big impact on the local minimum attained by the training procedure. While biases can be set to 0, weights are frequently initialised to vary in a random 0 mean distribution, taking into consideration a neuron's fan-in.

2.7 Tensorflow object detection API

API is an abbreviation for Application Programming Interface. An API provides developers with a set of common operations that eliminates the need for them to write code from scratch.APIs are great time savers in one sense. In many cases, they also provide users with convenience.

The TensorFlow object detection API is a framework for building a deep learning network that can detect objects.They already have pretrained models in their framework, which they call Model Zoo. This includes a collection of pretrained models trained on the COCO, KITTI, and Open Images datasets. If we are only interested in categories in this dataset, we can use these models for inference. They can also be used to initialise your models when training on a new dataset. When examining the 'protos folder,' which contains the function definitions, to gain a better understanding of the various supported components. When fine-tuning a model, the train, eval, ssd, faster rcnn, and preprocessing protos are especially important.

2.8 Dataset Review

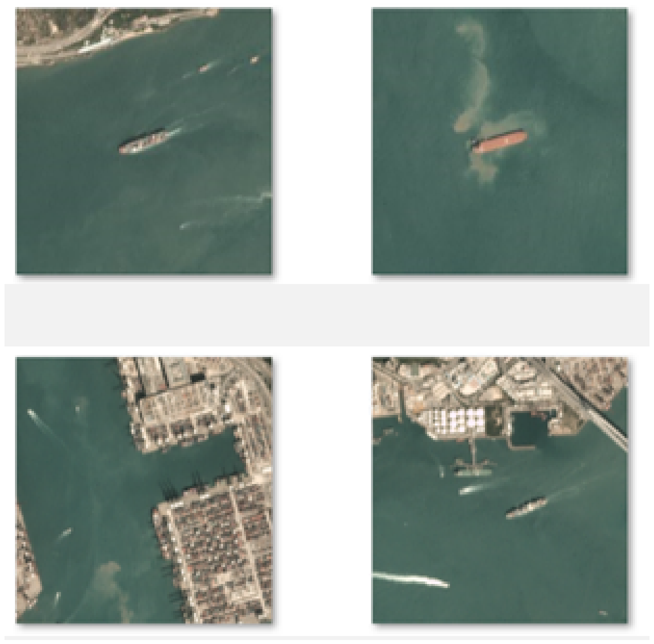


Figure 2.2: Images from dataset showing ships in ocean or near port

Dataset is taken from a Github profile (https://github.com/amanbasu/ship-detection/tree/master/dataset). The dataset was originally downloaded from www.planet.com which provides satellite data that helps researchers, businesses to understand the physical world and take a decision. It can be downloaded using planet API and AOI geometry in GeoJSON format. The images are changed to fixed dimensions, here 512 x 512. The dataset can be annotated using LabelImg tool and annotated in YOLO format. Dataset consists of 185 training images consisting of one or more than one ship. Ground truth dataset is an already annotated dataset which help us to benchmark the accuracy of the model predictions.

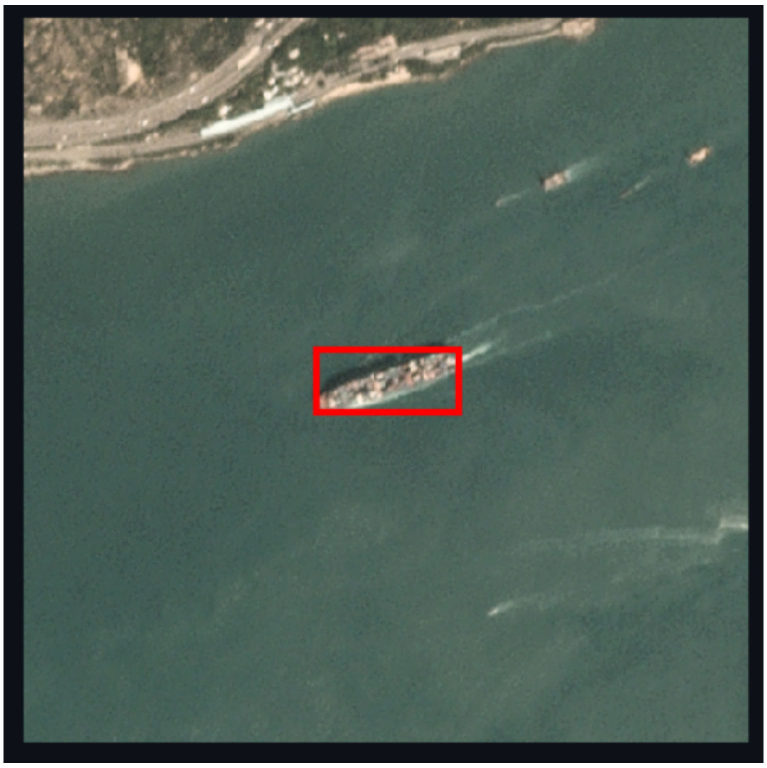


Figure 2.3: Ground truth label

Ground truth is a term used in statistics and machine learning to refer to the process of comparing machine learning results to the real world. The term comes from meteorology, where ”ground truth” refers to information gathered on the spot. The term alludes to a sort of reality check for machine learning algorithms. ”Ground truth” is a meteorological term for independent confirmation at a site of information obtained through remote sensing. A storm spotter reporting a tornado in the field that a meteorologist is tracking on Doppler radar is one example. Machine learning researchers also use a type of ”ground truth,” comparing the classifications made by machine learning algorithms to what they know in reality.

2.9 Dataset Annotation

Data annotation refers to the human activity of tagging content such as text, images, and videos so that machine learning models can recognise them and use them to generate predictions. When we label data elements, ML models accurately understand what they are going to process and retain that information in

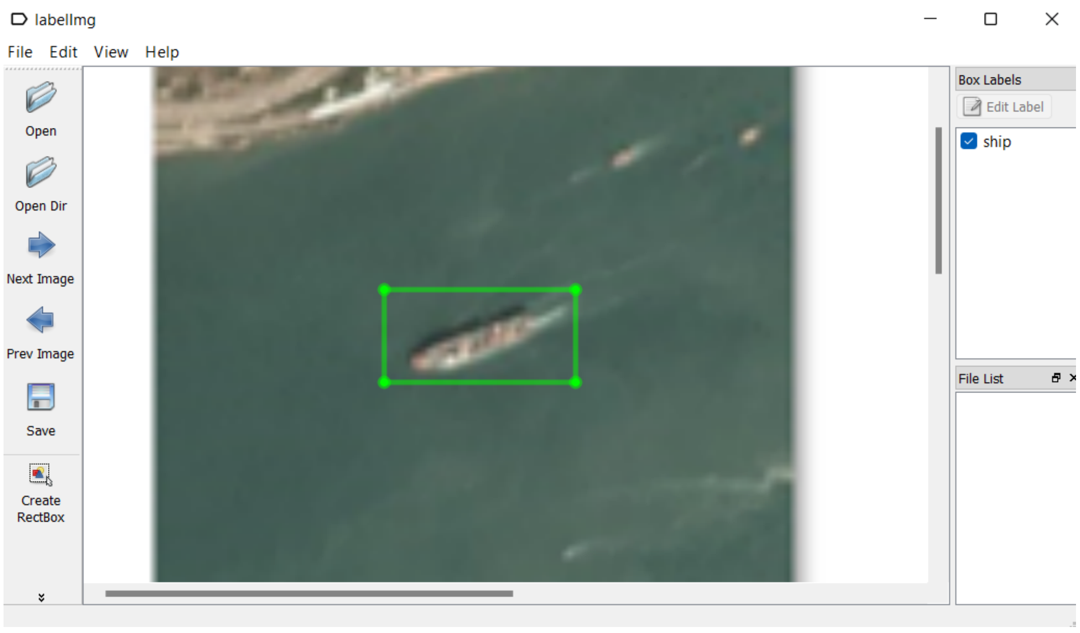


Figure 2.4: Annotation using LabelIMG

order to automatically process the available information and make decisions based on existing knowledge. The process of attributing, tagging, or labelling data is referred to as data annotation. To summarise, data labelling and data annotation both deal with labelling or tagging relevant information/metadata in a dataset so that machines can recognise it.

LabelImg is a free and open source tool for labelling images graphically. It's written in Python and has a graphical interface built with QT. It's a quick and easy way to label a few hundred images for your next object detection project.

2.10 Summary

- This chapter gives an understanding of basics of classification in detail.

- CNN is basically used for images. CNN is one of the types of deep learning.

- Deep Learning is the subset of Machine Learning.

- When a model is developed certain hyperparameters need to be set based on which the performance of model differs.

- The dataset selected is from github profile which was originally collected from planet website.

- Ground truth data is used to validate the precision of predicted bounding boxes.

- Dataset is annotated using LabelIMG application.

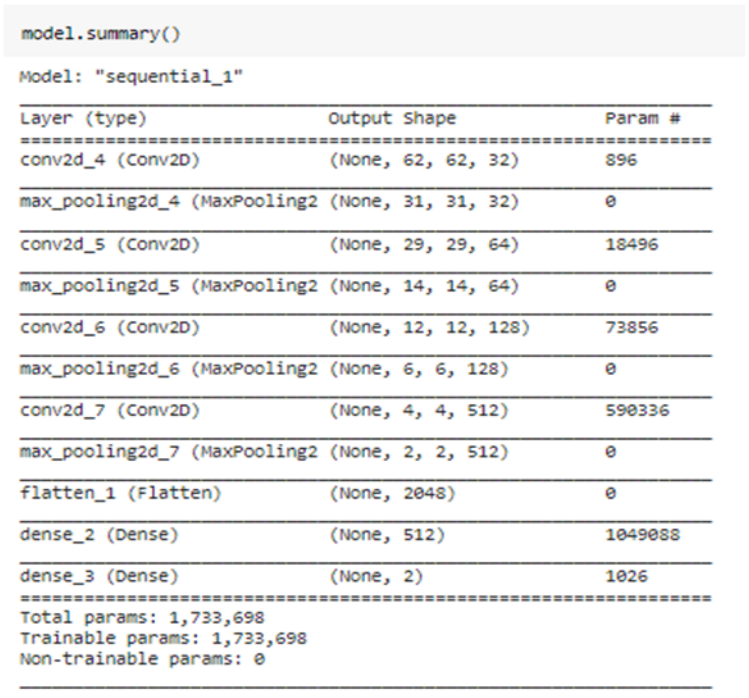
Chapter 3

DEVELOPMENT AND APPROACH

This chapter gives an understanding of the first approach followed to solve a detection problem from scratch.

3.1 Object Detection from scratch

The initial model was built with the aim of understanding the implementation of neural network to do a classification or detection problem. It consists of an image classifier which detects whether the given image consists of a ship or no ship. Selective search algorithm is used to find all the objects in an image and image classifier classifies it as ship or no ship. The image classifier consists of 4 convolutional layers with max pooling layers and a final fully connected layer. The model summary is given in fig:3.1.
The dataset selected for this is 'Ships in satellite imagery' obtained from Kaggle website. The dataset was collected over the San Francisco Bay and San Pedro Bay areas of California and extracted from Planet Satellite imagery. It includes 4000 images labelled with either a "ship" or "no-ship" classification having size 80 x 80. OpenCV's selective search algorithm is widely adopted in state-of-the-art algorithms such as RCNN and Fast-RCNN. It merges super pixels in a hierarchical fashion based on five key similarity measures:
1. Colour Similarity
2. Texture Similarity
3. Size similarity
4. Shape similarity
5. Metasimilarity



```
model.summary()

Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_4 (Conv2D)            (None, 62, 62, 32)        896
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 31, 31, 32)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 29, 29, 64)        18496
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 14, 14, 64)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 12, 12, 128)       73856
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 6, 6, 128)         0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 4, 4, 512)         590336
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 2, 2, 512)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2048)              0
_________________________________________________________________
dense_2 (Dense)              (None, 512)               1049088
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 1026
=================================================================
Total params: 1,733,698
Trainable params: 1,733,698
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.1: Summary of model

Selective search is a type of region proposal algorithm. A region proposal algorithm identifies possible objects within an image by using segmentation. Segmentation groups adjacent regions that have similar characteristics such as colour or texture. With the sliding window approach, the object is looked at all pixel locations and at all scales. With the region proposal algorithm, pixels are grouped into a smaller number of segments, so there is a much smaller number of proposals generated. These generated region proposals have different scales and aspect ratios, which reduces the number of image patches we need to classify. The output of selective search is given as input to binary classifier.

The general idea behind non-maximum suppression is to limit the number of detections in a frame to the number of objects actually present. If the object in the frame is fairly large and over 2000 object proposals have been generated, it is very likely that some of these will have significant overlap with each other and the object.

3.2 Learning Curve

For algorithms that learn (optimise their internal parameters) incrementally over time, such as deep learning neural networks, learning curves are widely used.

The learning metric could be maximising, which means that higher scores (larger numbers) indicate more learning. One example is classification accuracy.It is more common to use a minimising score, such as loss or error, where better scores (smaller numbers) indicate more learning and a value of 0.0 indicates that the training dataset was perfectly learned with no mistakes.

The current state of a machine learning model at each step of the training algorithm can be evaluated during training. It can be tested against the training dataset to determine how well the model is "learning." It can also be tested on a separate validation dataset that is not included in the training dataset. The validation dataset is used to determine how well the model "generalises."

-Train Learning Curve: A learning curve derived from the training dataset that indicates how well the model is learning.

-Validation Learning Curve: A learning curve derived from a hold-out validation dataset that shows how well the model generalises.

In some cases, creating learning curves for multiple metrics is also common, such as in classification predictive modelling problems, where the model may be optimised using cross-entropy loss and model performance is measured using classification accuracy. In this case, two plots are generated, one for each metric's learning curves, and each plot can display two learning curves, one for each of the train and validation datasets.

-Optimization Learning Curves: Learning curves calculated on the metric used to optimise the model's parameters, such as loss.

-Performance Learning Curves: Learning curves calculated on the basis of performance.

The shape and dynamics of a learning curve can be used to diagnose the behaviour of a machine learning model and, as a result, suggest the type of configuration changes that could be made to improve learning and/or performance.Learning curves are likely to exhibit three common dynamics, which are as follows:

3.2.1 Underfit

Underfitting occurs when a model is unable to learn from the training dataset. Only the learning curve of the training loss can be used to identify an underfit model. It may display a flat line or noisy values of relatively high loss, indicating that the model failed to learn the training dataset at all.An underfit model can also be identified by a decreasing training loss that continues to decrease at the end of the plot. This indicates that the model is capable of additional learning and improvement, and that the training process was terminated prematurely. Underfitting is indicated by a plot of learning curves if:
-Regardless of training, the training loss remains constant.
-The training loss decreases until the end of training.

3.2.2 Overfit

Overfitting is defined as a model that has learned the training dataset too well, including statistical noise or random fluctuations. Overfitting has the disadvantage that the more specialised the model becomes to training data, the less well it can generalise to new data, resulting in an increase in generalisation error. The model's performance on the validation dataset can be used to quantify this increase in generalisation error. This is common when the model has more capacity than is required for the problem and, as a result, too much flexibility. It can also happen if the model is trained for an inordinately long period of time.

3.2.3 Good fit

The learning algorithm seeks a good fit between an overfit and an underfit model.A good fit is identified by a training and validation loss that decreases to a stable point with a small difference between the two final loss values.

The model's loss is almost always lower on the training dataset than on the validation dataset. This implies that there will be some disparity between the train and validation loss learning curves. This is known as the ”generalisation gap.”

3.3 Results from above approach

Confusion matrix gives the performance of a classifier. It shows the output based on each class. Here it is a binary classifier which classifies whether an image is

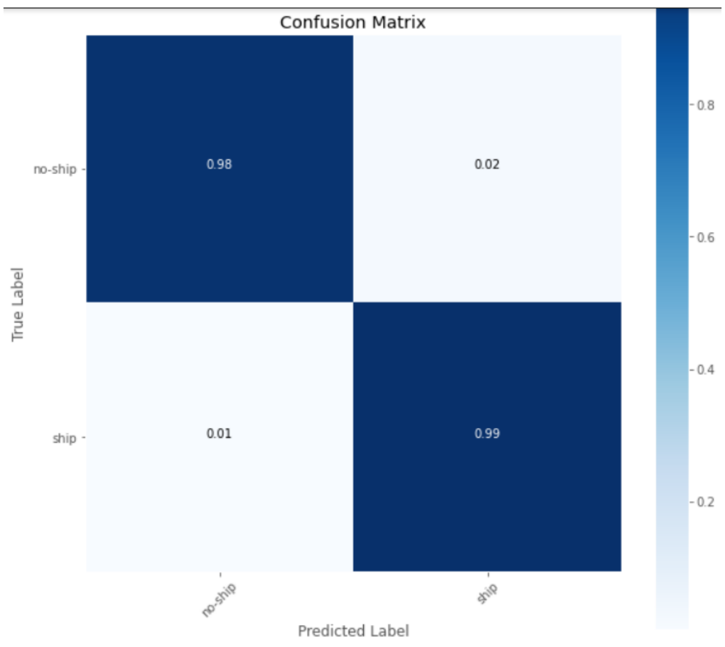


Figure 3.2: Confusion Matrix of binary classifier

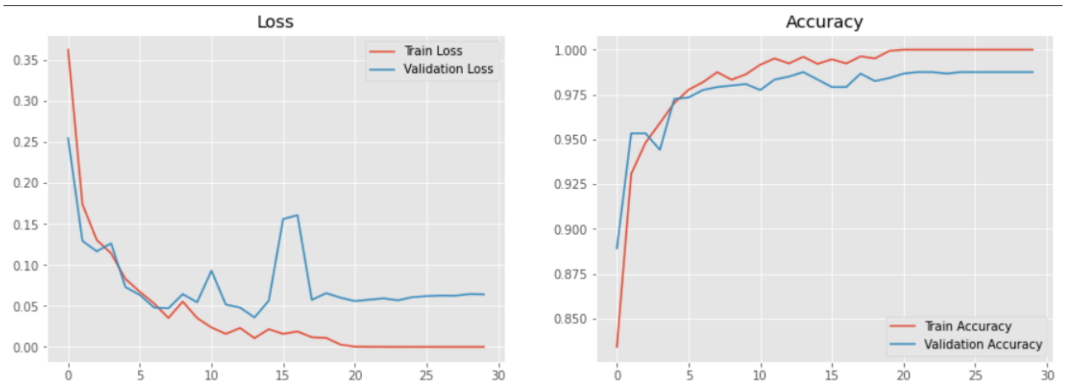


Figure 3.3: Loss curve and Accuracy curve

ship or no ship. The classifier output can be as follows:

- True Positives (TP): Ships are classified correctly.
- True Negatives (TN): No ships are classified correctly.
- False positives (FP): No ship is classified mistakenly as ship.
- False Negatives (FN): Ships are classified as no ship.

The confusion matrix for this is shown in given fig:3.2.

After every epoch, loss function is calculated after every epoch. It generally

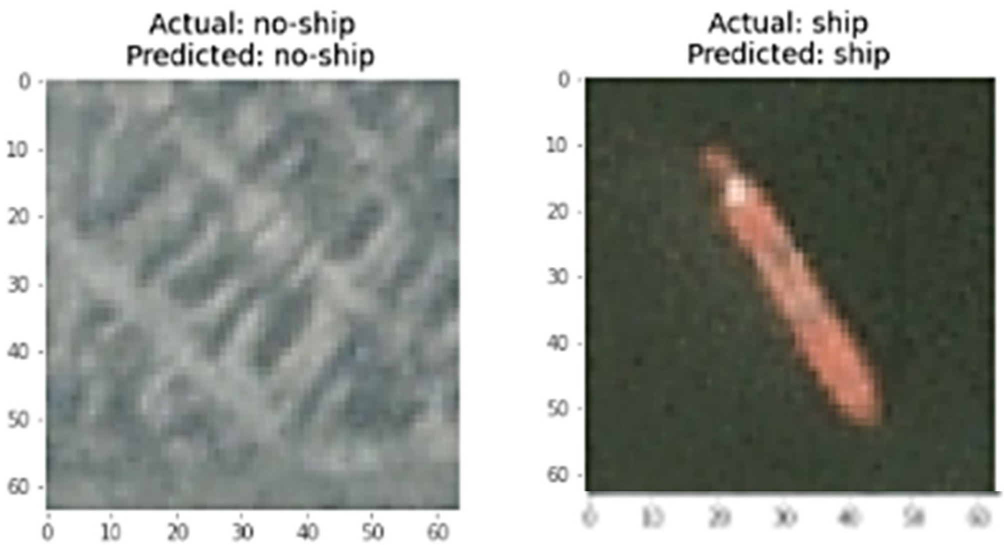


Figure 3.4: Prediction of binary classifier

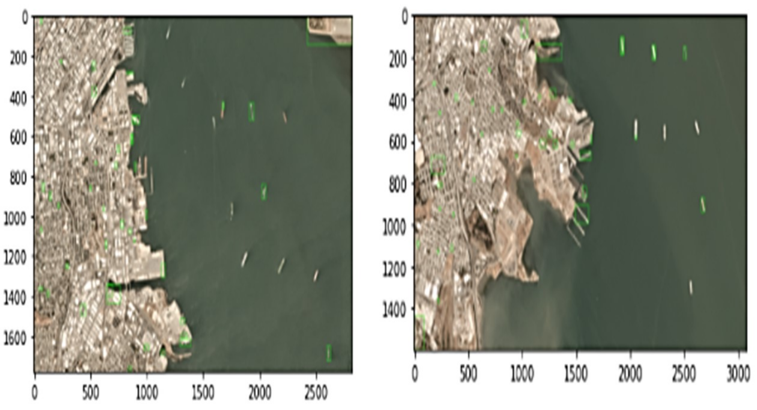


Figure 3.5: Output of object detection from scratch

gives an idea of calculated loss at different epoch on a subset of data. More detailed understanding can be obtained from plotting validation loss along with training loss. Accuracy graph is mostly used to understand the progress of neural network. The gap between validation accuracy and training accuracy gives an idea of overfitting. The overfitting is high if gap is large. The predictions versus actual of the binary image classifier is shown in fig:3.4.

3.4 Summary

- In this chapter a review of object detection from scratch is given.
    - The dataset is taken from kaggle website.
    - The selective search algorithm is used to find the region of interest.
    - The results obtained in above approach is given.

Chapter 4

TRANSFER LEARNING

This chapter gives the idea of doing the same problem with transfer learning using YOLOv3, Faster RCNN and SSD. A comparative analysis of result obtained using all 3 models is also given.

4.1 Transfer Learning

The transfer learning process means using a model developed for one problem on a second, related problem. Transfer learning is effectively a type of weight initialization scheme in which weights from re-used layers are used to start the training process and then adapted for the new problem. The state of art model present in object detection are RCNN, Faster RCNN, SSD, YOLO, RetinaNet, etc. These can be classified into two genres- 'two stage detection' and 'one stage detection'. Two stage detection frames the detection as coarse to fine while one stage detection frame it as completes in one step. RCNN, SPPNet, Fast RCNN, Faster RCNN and feature pyramid networks come under two stage detectors and YOLO, SSD and RetinaNet comes under one stage detectors.

A saved network that was previously trained on a large dataset, typically on a large-scale image-classification task, is referred to as a pre-trained model. You can either use the pretrained model as is or use transfer learning to tailor it to a specific task.

The idea behind transfer learning for image classification is that if a model is trained on a large and diverse dataset, it can effectively serve as a generic model of the visual world. These learned feature maps can be used instead of starting from scratch by training a large model on a large dataset.

1. Feature Extraction: Use previous network representations to extract meaningful features from new samples. Simply layer a new classifier, trained from scratch, on top of the pretrained model to reuse the feature maps learned previously for the dataset. The base convolutional network already has features that are useful for classifying images in general. The final, classification part of the pretrained model, on the other hand, is specific to the original classification task, and thus to the set of classes on which the model was trained.

2. Fine-tuning: Unfreeze a few of the top layers of a frozen model base and train both the newly added classifier layers and the base model's final layers concurrently.

4.2 Transfer Learning using YOLOv3

"You Only Look Once," or YOLO, is a group of deep learning models aimed at detecting objects quickly. YOLO comes in three different flavours: YOLOv1, YOLOv2, and YOLOv3.The first version presented a generic architecture, the second version refined the design and used predefined anchor boxes to improve the bounding box proposal, and the third version improved the model architecture and training process even more.In a single assessment, a single neural network predicts bounding boxes and class probabilities directly from entire images. Because the entire detection pipeline is a single network, it can be adjusted directly on detection performance from beginning to end.

The inputs are a set of form images (m, 512, 512, 3). This image is sent to a convolutional neural network by YOLO v3 (CNN). The output volume is obtained by flattening the final two dimensions of the above output.
The result is a list of bounding boxes together with the classes that have been identified. Six numbers are assigned to each bounding box (pc, bx, by, bh, bw, c). Finally, to prevent selecting overlapping boxes, we use the IoU (Intersection over Union) and Non-Max Suppression techniques. YOLO v3 is based on a Darknet variation that was originally trained on Imagenet and has a 53-layer network. For detection, another 53 layers are added, giving YOLO v3 a 106-layer fully convolutional underlying architecture. Detection in YOLO v3 is accomplished by using 1 x 1 detection kernels on feature maps of three different sizes at three different locations in the network. 1 x 1 x (B x (5 + C) is the detection kernel's shape. B is for the number of bounding boxes a feature map cell can predict, '5' for the four bounding box attributes and one object confidence, and C stands for the number

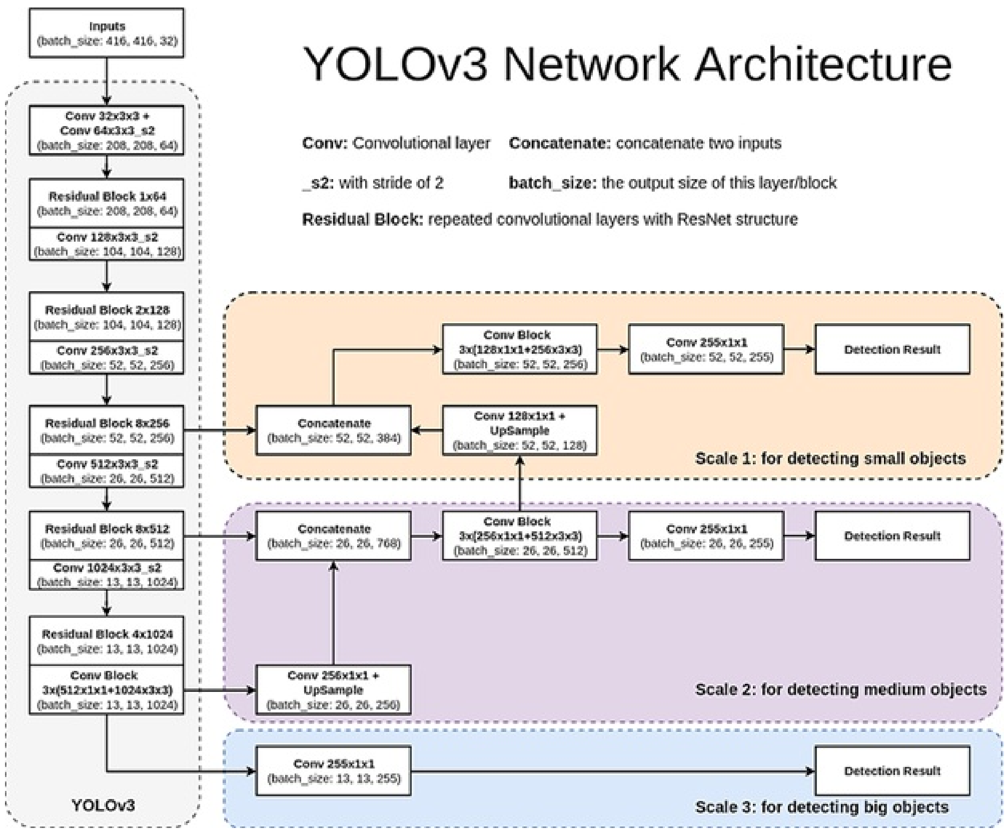

Inputs (batch_size: 416, 416, 32)
YOLOv3 Network Architecture
Conv: Convolutional layer
Concatenate: concatenate two inputs
_s2: with stride of 2
batch_size: the output size of this layer/block
Residual Block: repeated convolutional layers with ResNet structure
Conv 32x3x3 + Conv 64x3x3_s2 (batch_size: 208, 208, 64)
Residual Block 1x64 (batch_size: 208, 208, 64)
Conv 128x3x3_s2 (batch_size: 104, 104, 128)
Residual Block 2x128 (batch_size: 104, 104, 128)
Conv 256x3x3_s2 (batch_size: 52, 52, 256)
Residual Block 8x256 (batch_size: 52, 52, 256)
Conv 512x3x3_s2 (batch_size: 26, 26, 512)
Residual Block 8x512 (batch_size: 26, 26, 512)
Conv 1024x3x3_s2 (batch_size: 13, 13, 1024)
Residual Block 4x1024 (batch_size: 13, 13, 1024)
Conv Block 3x(512x1x1+1024x3x3) (batch_size: 13, 13, 1024)
YOLOv3
Conv Block 3x(128x1x1+256x3x3) (batch_size: 52, 52, 256)
Conv 255x1x1 (batch_size: 52, 52, 255)
Detection Result
Concatenate (batch_size: 52, 52, 384)
Conv 128x1x1 + UpSample (batch_size: 52, 52, 128)
Scale 1: for detecting small objects
Concatenate (batch_size: 26, 26, 768)
Conv Block 3x(256x1x1+512x3x3) (batch_size: 26, 26, 512)
Conv 255x1x1 (batch_size: 26, 26, 255)
Detection Result
Conv 256x1x1 + UpSample (batch_size: 26, 26, 256)
Scale 2: for detecting medium objects
Conv 255x1x1 (batch_size: 13, 13, 255)
Detection Result
Scale 3: for detecting big objects

Figure 4.1: Architecture of YOLOv3

of classes.

Hyper-parameters were employed. Class threshold defines the predicted object's probability threshold. Non-Max Suppression Threshold aids in overcoming the issue of an object being detected many times in an image. It accomplishes this by selecting boxes with the highest likelihood and suppressing nearby ones with lower probabilities (less than the predefined threshold). Input height and input shape are the image sizes to be used. It has 53 convolutional layers, each of which is followed by a batch normalisation layer and the activation of the Leaky ReLU. Numerous filters are convolved on the pictures using the convolution layer, which results in multiple feature maps. There is no pooling and the feature maps are downsampled using a convolutional layer with stride 2. It aids in the prevention of low-level feature loss, which is frequently linked to pooling.

YOLO (You Only Look Once) is a type of regression algorithm. To identify elements like height, width, and centre, and object classes, this algorithm uses a single bounding box regression. YOLOv3 was proposed in 2018. This showed better performance in detecting tiny objects because it is possible to run predictions at three different levels of granularity.

In this the model divides the image into grid and each grid is responsible for detecting an object or part of object. Each grid cell predicts bounding boxes and confidence score for each box. If none of the object to be detected is present then the confidence score should be zero otherwise it should be equal to the intersection over union (IOU) between the predicted box and the ground truth. Each bounding box outputs 5 predictions: x, y, w, h and confidence. The x, y coordinates represent centre of the box. Width w and height h are predicted with respect to the whole image.

YOLO predicts the probability of an object by using bounding box regression. IOU describes the overlap of bounding boxes and ensures that final prediction gives unique bounding boxes for each object. Coding was done in Colab Pro. Programming language used was Python. The GPU allocated was Tesla P-100. YOLOv3 requires graphic card compactible with CUDA. Since darknet framework was used initially to train the YOLO.

4.3 Transfer Learning using Faster RCNN

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun created the Faster R-CNN object detection architecture. It was developed as a response to the complexity necessary for accurate localization in object identification tasks, as well as an enhancement over the R-CNN and Fast R-CNN algorithms. The majority of object detection algorithms rely on a guess or hypothesis about where the object to be identified is located. As a result, region proposal algorithms were developed, and the accuracy of most object identification methods currently depends heavily on the effectiveness of their region proposal. While this is true, it was later discovered that the region proposal technique might make object identification incredibly slow because each proposal is subjected to a CNN.

The architecture of Faster RCNN is shown in below fig: 4.2. The Faster R-CNN was created as a solution to this problem, with a Region Proposal Network (RPN) that shares full-image convolutional features with the detection network. There are two sections to the Faster R-CNN. A CNN that provides possible places for the object to be found. This is where the RPN comes in. It tells the Faster R-CNN where it thinks it should seek for the object in the second stage of the algorithm.

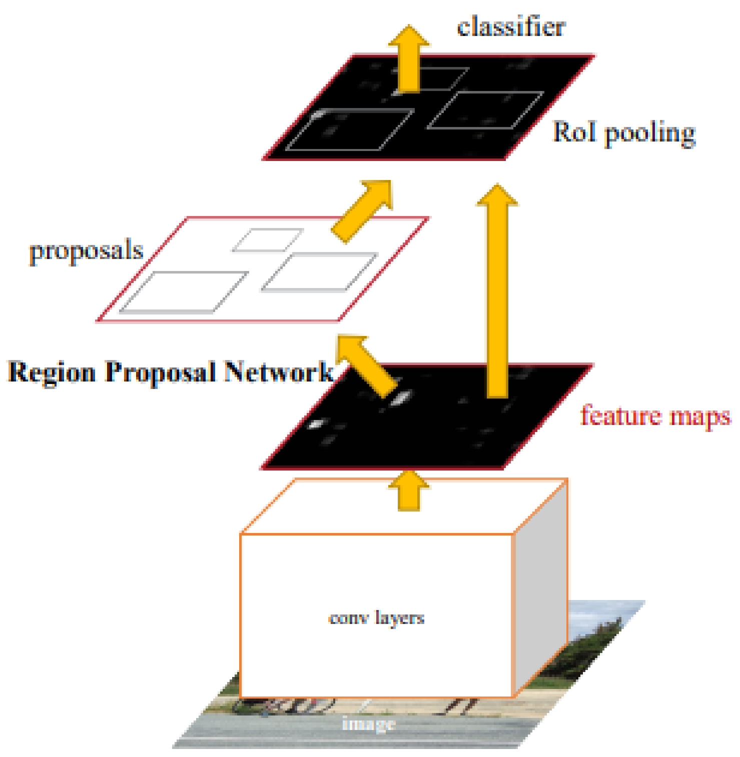


Figure 4.2: Architechture of Faster RCNN

The RPN takes an image as input and outputs rectangular objects that are its suggestions for the object's position. It also produces a value that indicates how likely it is to belong to a specific class compared to the rest of the image. In this case, search selective is used to split portions with similar textures and colours into different boxes. These selections are classified using the Softmax activation function, and they are output. Each layer's bounding box localizations are likewise created using a linear regressor. This is the method for extracting features.

The Fast R-CNN object detector is the second part of the Faster R-CNN. It takes the feature maps that the initial CNN produced and performs ROI pooling on them. ROI pooling extracts the regions of interest from the first CNN's output of a set of regions. These ROIs are then passed on to a fully connected layer, where they are flattened before being categorised and given bounding boxes in the output layer.

One of the main differences between the RPN and the Faster R-CNN is that the entire system works as a single network and that the RPN and the Faster R-CNN share convolutional layers. This eliminates the need for two different networks to be trained. TensorFlow's Faster R-CNN Inception V2 Model from TensorFlow's model zoo will be used in this project. Learning rate, epochs, validation period, batch size, GPU devices (or a TPU), and so on are among the model's setup requirements. The manual contains the configuration parameters.

4.4 Transfer Learning using SSD

SSD is a single-shot detector for object detection that does not use a substitute or proxy region proposal network (RPN). It envisions the classes and boundary boxes being extracted directly from the feature maps in a single pass. Its accuracy has been improved by incorporating small convolutional filters (multi-scale features) to forecast object classes and offsets to original (default) boxes. These are the advancements that have allowed SSD to compete with R-CNN accuracy rates while utilising lower resolution images, which also increases speed. SSD is important in the two key tasks of object detection, namely classification and localization. The image's adaptability improves detection robustness by taking into account windows of various sizes.

The SSD architecture is based on the VGG-16 architecture, although it does not have the completely connected layers. VGG-16 was chosen as the basic network due to its great performance in high-quality image classification tasks and its widespread application in issues where transfer learning might assist improve results. A number of auxiliary convolutional layers (from conv6 onwards) were added in place of the original VGG fully connected layers, allowing for the extraction of features at many scales while gradually decreasing the size of the input to each succeeding layer. The architecture of SSD is shown in fig below.

-MultiBox Szegedy's work on MultiBox, a method for quick class-agnostic bounding box coordinate recommendations, influenced SSD's bounding box regression algorithm. Surprisingly, a neural network modelled after Inception is used in the MultiBox research. Because the number of dimensions decreases (although "width" and "height" remain the same), the 1x1 convolutions shown below aid in dimensionality reduction.

The loss mechanism in MultiBox also included two important components that made their way onto SSD:
Confidence Loss: this metric indicates how confident the network is in the computed bounding box's objectness. This loss is calculated using categorical cross-entropy.
Location loss: The distance between the network's predicted bounding boxes and

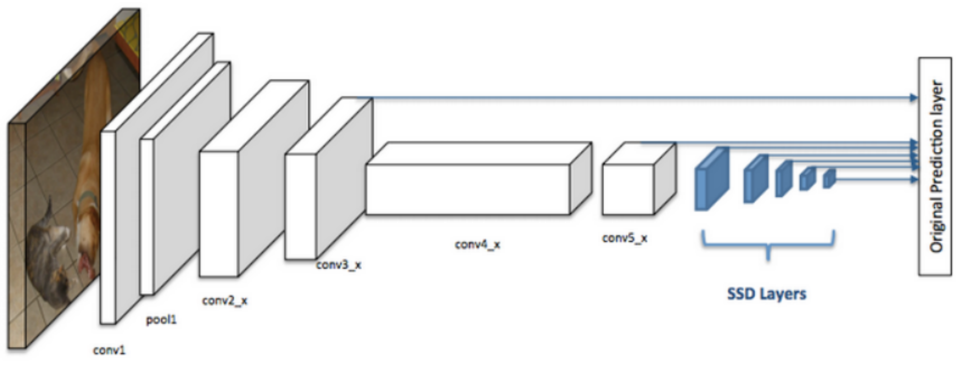


Figure 4.3: Architecture of SSD

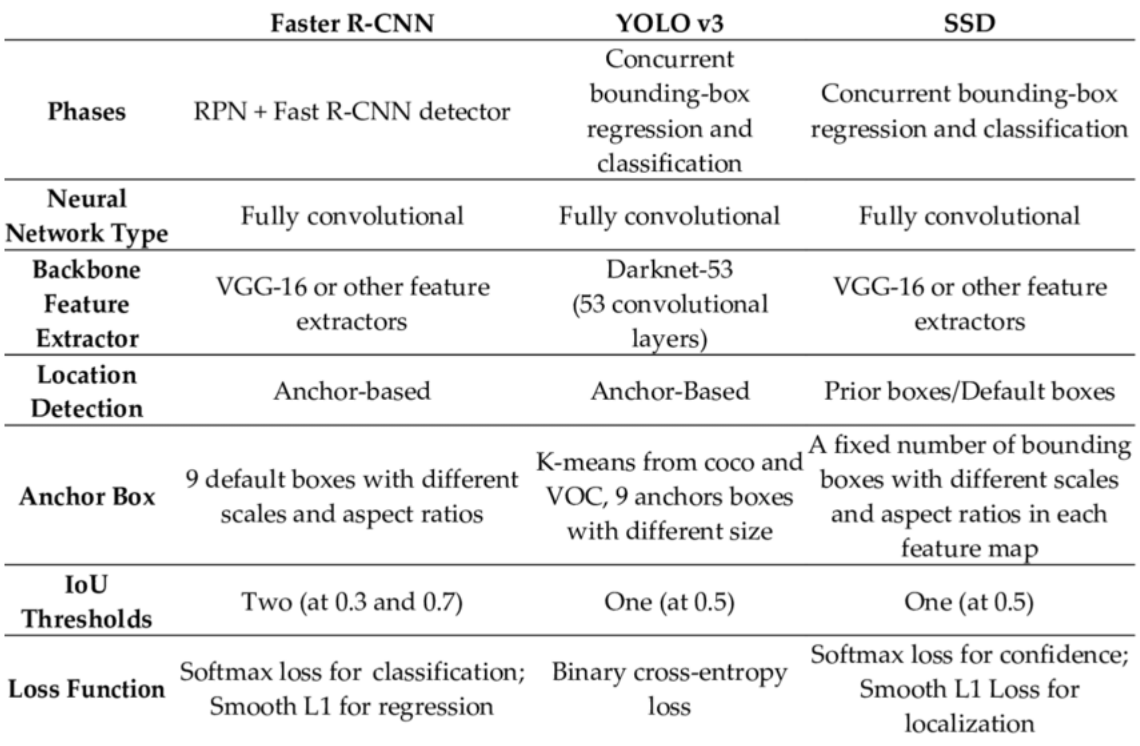
the ground truth ones from the training set is measured by location loss. Here, L2-Norm is employed.

Without getting too technical, the loss, which indicates how far off our prediction "landed," is expressed as follows:

$$confidenceloss + alpha * locationloss = multiboxloss$$

SSD is intended for real-time object detection. Faster R-CNN creates boundary boxes using a region proposal network and then uses those boxes to classify objects. While it is considered cutting-edge in terms of accuracy, the entire process runs at 7 frames per second. Far below the requirements of real-time processing. By eliminating the need for the region proposal network, SSD speeds up the process. SSD implements a few improvements, such as multi-scale features and default boxes, to compensate for the drop in accuracy. These enhancements enable SSD to match the accuracy of the Faster R-CNN using lower resolution images, further increasing the speed. According to the comparison below, it achieves real-time processing speed and even outperforms the accuracy of the Faster R-CNN.

A delegated region proposal network is not used by SSD. Instead, it comes down to a very simple solution. It uses small convolution filters to compute both the location and class scores. SSD makes predictions using 3 3 convolution filters for each cell after extracting the feature maps. (These filters compute the results in the same way that regular CNN filters do.) Each filter generates 25 channels, with 21 scores for each class and one boundary box.



	Faster R-CNN	YOLO v3	SSD
Phases	RPN + Fast R-CNN detector	Concurrent bounding-box regression and classification	Concurrent bounding-box regression and classification
Neural Network Type	Fully convolutional	Fully convolutional	Fully convolutional
Backbone Feature Extractor	VGG-16 or other feature extractors	Darknet-53 (53 convolutional layers)	VGG-16 or other feature extractors
Location Detection	Anchor-based	Anchor-Based	Prior boxes/Default boxes
Anchor Box	9 default boxes with different scales and aspect ratios	K-means from coco and VOC, 9 anchors boxes with different size	A fixed number of bounding boxes with different scales and aspect ratios in each feature map
IoU Thresholds	Two (at 0.3 and 0.7)	One (at 0.5)	One (at 0.5)
Loss Function	Softmax loss for classification; Smooth L1 for regression	Binary cross-entropy loss	Softmax loss for confidence; Smooth L1 Loss for localization

Figure 4.4: Comparison of Faster R-CNN, YOLO v3 and SSD

4.5 Summary

- A detailed explanation of all three transfer learning models and their architecture is given in this section.

- The differences in three models are given in the fig: 4.4.
- Faster RCNN is two stage detector.
- SSD and YOLOv3 are one stage detectors.

Chapter 5

RESULTS AND EVALUATION

This chapter presents the final results of the models developed during this project. A comparative analysis of three models is provided, followed by a discussion. Furthermore, samples of the predicted images are displayed, along with a qualitative discussion.

5.1 Metrics

Mean average precision (MAP) is a popular metric used for assessing the performance of document retrieval and object detection algorithms. It is sometimes referred to as Average precision (AP). The concept of intersection over union (IOU) is used to detect objects. IOU is used to measure how much our predicted boundary overlaps the ground truth (the real boundary of the object).

Precision is measured by how accurate our predictions are i.e., the percentage of our predictions that are correct while recall gives how well we find all positive cases. For example, in our top K predictions, we find 80% of the possible positive cases.

$Precision = TP/(TP + FP)$
$Recall = TP/(TP + FN)$
where, TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative
The general definition for the Average Precision (AP) is finding the area under the precision-recall curve above. The MAP or AP obtained after detection is 89

5.2 Evaluation

5.2.1 YOLOv3

Yolov3 was trained for 766 iterations reducing the average loss to 0.877.On evaluating on an image the output obtained is given in fig: 5.1.The speed of detection comes around 90 ms.

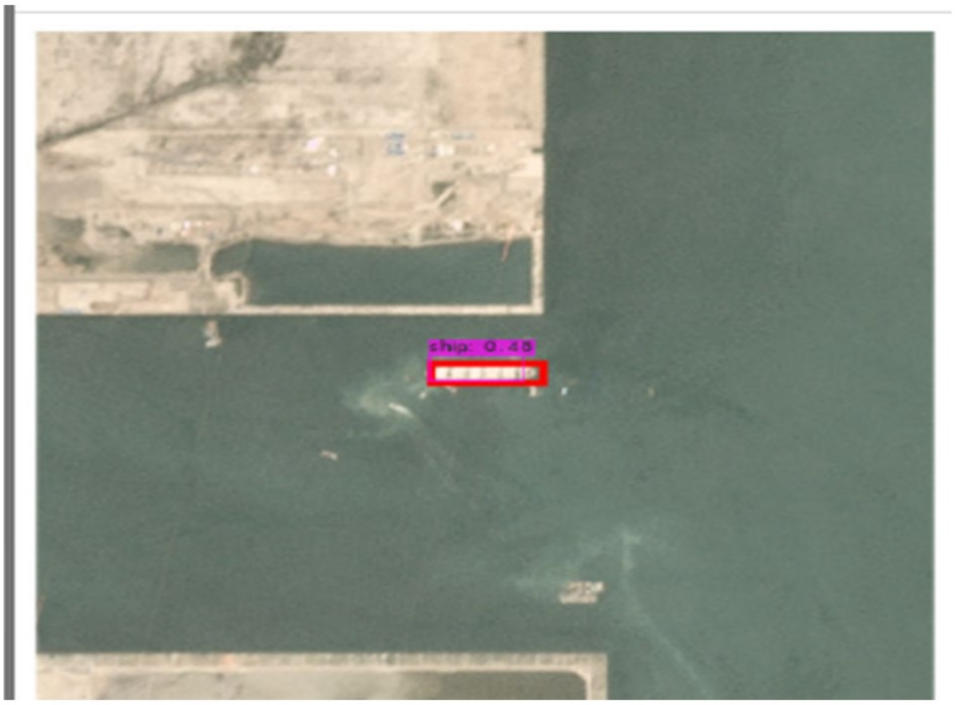
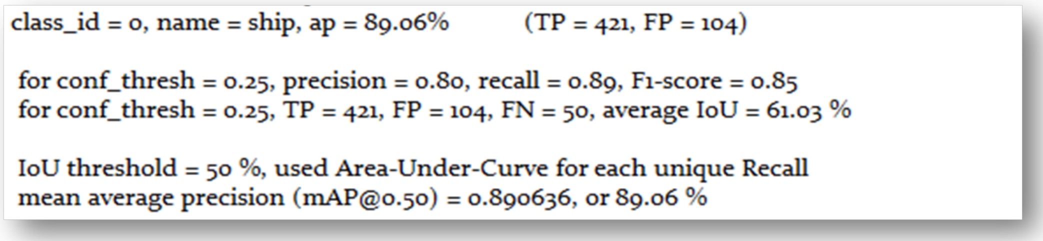


Figure 5.1: Output obtained from YOLOv3

The details about Mean Average Precision is given in figure below. 80% MAP is obtained in YOLOv3.

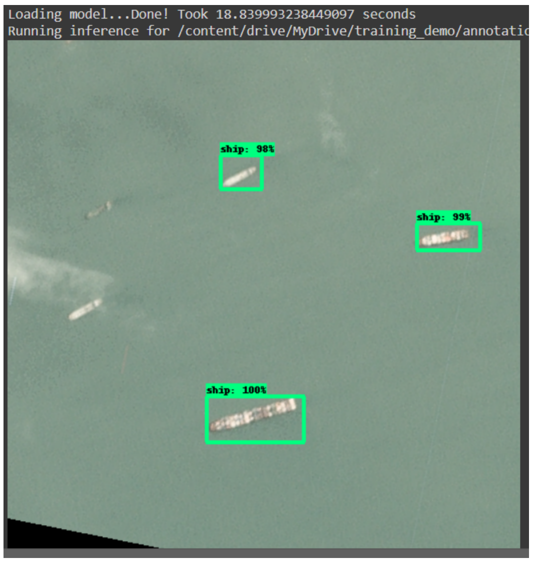


```
class_id = 0, name = ship, ap = 89.06%          (TP = 421, FP = 104)

 for conf_thresh = 0.25, precision = 0.80, recall = 0.89, F1-score = 0.85
 for conf_thresh = 0.25, TP = 421, FP = 104, FN = 50, average IoU = 61.03 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.890636, or 89.06 %
```

Figure 5.2: MAP @0.50 in YOLOv3

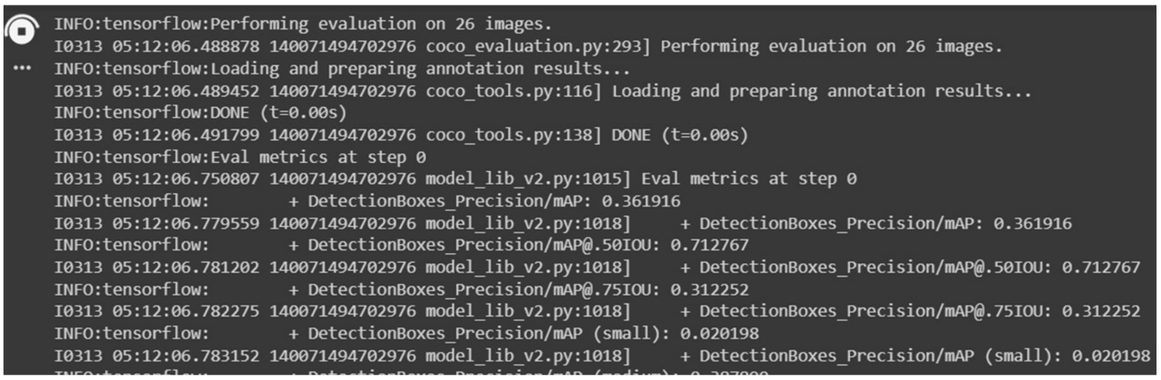
5.2.2 Faster RCNN

The training was done on Faster RCNN with ResNet 50 model. The learning rate was set 0.04. The output obtained on image is shown in figure below. The time taken to detect is 18 seconds and the overall MAP obtained is 71%.



```
Loading model...Done! Took 18.839993238449097 seconds
Running inference for /content/drive/MyDrive/training_demo/annotatic
```

Figure 5.3: Output obtained from Faster RCNN

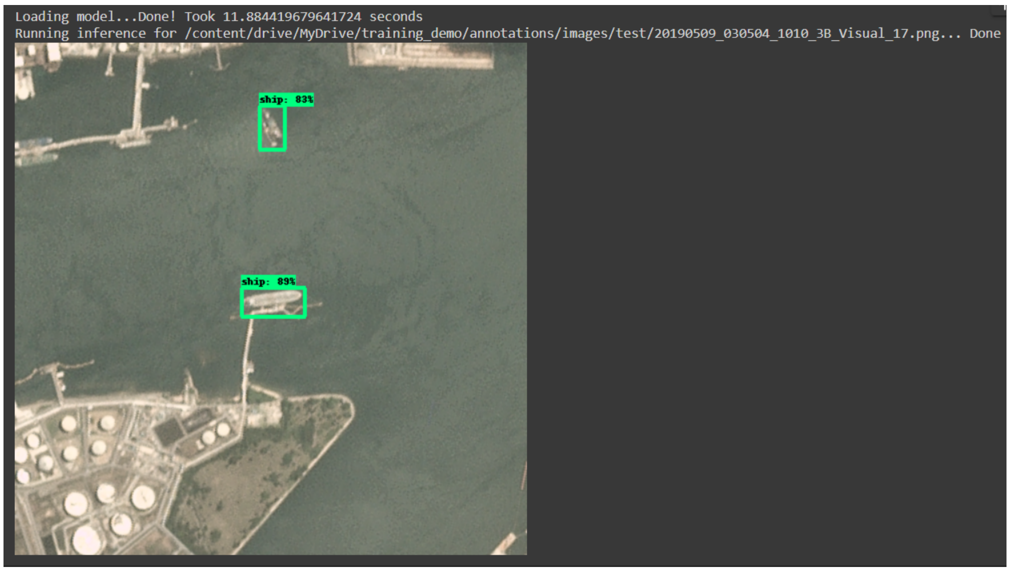


```
INFO:tensorflow:Performing evaluation on 26 images.
I0313 05:12:06.488878 140071494702976 coco_evaluation.py:293] Performing evaluation on 26 images.
INFO:tensorflow:Loading and preparing annotation results...
I0313 05:12:06.489452 140071494702976 coco_tools.py:116] Loading and preparing annotation results...
INFO:tensorflow:DONE (t=0.00s)
I0313 05:12:06.491799 140071494702976 coco_tools.py:138] DONE (t=0.00s)
INFO:tensorflow:Eval metrics at step 0
I0313 05:12:06.750807 140071494702976 model_lib_v2.py:1015] Eval metrics at step 0
INFO:tensorflow:        + DetectionBoxes_Precision/mAP: 0.361916
I0313 05:12:06.779559 140071494702976 model_lib_v2.py:1018]     + DetectionBoxes_Precision/mAP: 0.361916
INFO:tensorflow:        + DetectionBoxes_Precision/mAP@.50IOU: 0.712767
I0313 05:12:06.781202 140071494702976 model_lib_v2.py:1018]     + DetectionBoxes_Precision/mAP@.50IOU: 0.712767
INFO:tensorflow:        + DetectionBoxes_Precision/mAP@.75IOU: 0.312252
I0313 05:12:06.782275 140071494702976 model_lib_v2.py:1018]     + DetectionBoxes_Precision/mAP@.75IOU: 0.312252
INFO:tensorflow:        + DetectionBoxes_Precision/mAP (small): 0.020198
I0313 05:12:06.783152 140071494702976 model_lib_v2.py:1018]     + DetectionBoxes_Precision/mAP (small): 0.020198
```

Figure 5.4: MAP obtained from FRCNN

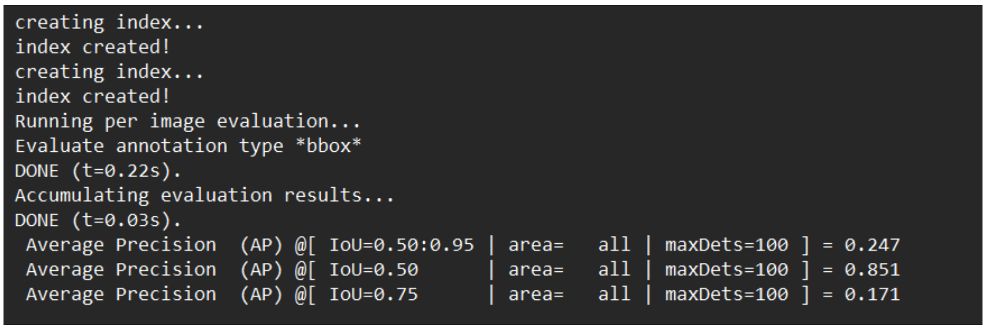
5.2.3 SSD

The training was done on SSD with ResNet 50 model. The learning rate was set to 0.0399. The output obtained on image is shown in figure below. The time taken to detect is 11 seconds.



```
Loading model...Done! Took 11.884419679641724 seconds
Running inference for /content/drive/MyDrive/training_demo/annotations/images/test/20190509_030504_1010_3B_Visual_17.png... Done
```

Figure 5.5: Output obtained from SSD



```
creating index...
index created!
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=0.22s).
Accumulating evaluation results...
DONE (t=0.03s).
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.247
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.851
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.171
```

Figure 5.6: MAP obtained from SSD

5.3 Comparative Analysis

The comparative analysis of three models are given in table below:

Models	MAP	Speed of detection
YOLOv3	89%	90 ms
FRCNN	71%	18 s
SSD	85%	11 s

Table 5.1: Comparative Analysis of three models

YOLOv3 outperforms both the models in the context of both speed of detection and Mean Average Precision.

Conclusion & Future Work

In this report, object detection is performed using two methods. One is using selective search algorithm to find the region of interest and second is using transfer learning (using YOLOv3, Faster RCNN and SSD). Applying transfer learning increases the speed of detection and also improves the accuracy of detection. The initial approach was taken to understand the detection and classification problem in depth. Training and testing were implemented using Python language. The Mean average Precision obtained for the dataset using yolov3 is 89% which was better than all other models.

Future Work

There are many methods to improve the original yolov3 algorithm such as data augmentation, etc. Those methods can be explored to get better results. By automatically generating synthetic data, it is possible to gain easy access to ground truth data. An object mask could be used instead of saving an object's bounding box. Only the pixels of the object are marked by an object mask. The reason for saving an object mask rather than the bounding box is that the bounding box contains background noise, whereas an object mask only contains the interesting pixels, i.e. the object. It will be interesting to see if this improves the accuracy even more.

Bibliography

[1] Planet Website,
`https://www.planet.com/`.

[2] Kanjir U, Greidanus H and Ostir K 2018 Vessel detection and classification from spaceborne optical images: A literature survey Remote Sensing of Environment 207 1-26.

[3] Lin Y, He H, Tai H, Chen F and Yin Z 2017 Rotation and scale invariant target detection in optical remote sensing images based on pose-consistency voting Multimed. Tools Appl.

[4] Tang J, Deng C, Huang G and Zhao B 2015 Compressed-domain ship detection on spaceborne optical image using deep neural network and extreme learning machine IEEE Trans. Geosci. Remote Sens. 53 1174-85

[5] Jinlei Ma, Z. Z., “Ship Detection in Optical Satellite Images via Directional Bounding Boxes Based on Ship Center and Orientation Prediction”.

[6] Feng Yang, Qizhi Xu, Feng Gao, Lei Hu, “Ship detection from optical satellite images based on virtual search mechanism”-IEEE, 2015.

[7] Zhengxia Zou, Zhenwei Shi, Yuhong Guo and Jieping Ye, “Object Detection in 20 years: A Survey”.

[8] Joseph Redmon, Ali Farhadi, “YOLOv3: An Incremental Improvement”.

[9] Yulian Zhang, Lihong Guo, Zengfa Wang, Yang Yu, Xinwei Liu and Fang Xu, “Intelligent Ship Detection in Remote Sensing Images Based on Multi-Layer Convolutional Feature Fusion”.

[10] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, and A.W.M. Smeulders, “Selective Search for Object Recognition”.

[11] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun,”Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, 2015.

[12] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, SSD: Single Shot MultiBox Detector,2015.

[13] Selective search,
`https://learnopencv.com/selective-search-for-object-detection-cpp-python/`

[14] YOLO,
`https://pjreddie.com/darknet/yolo/`

[15] Ship detection from scratch,
`https://medium.com/intel-software-innovators/`
`ship-detection-in-satellite-images-from-scratch-849ccfcc3072`

[16] Medium,
`https://jonathan-hui.medium.com/map-mean-average-precision-for-object-|`
`|detection-45c121a31173` .