

**SHIP DETECTION AND CLASSIFICATION USING TYPE OF CONVOLUTION
NEURAL NETWORKS**

**A THESIS SUBMITTED TO
THE GRADUATE SCHOOL NATURAL AND APPLIED SCIENCES OF
ÇANKAYA UNIVERSITY**

BY

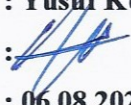
YUSUF KUNT

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF
MASTER OF SCIENCE
IN
ELECTRONIC AND COMMUNICATION ENGINEERING
DEPARTMENT**

JULY 2020

ÇANKAYA UNIVERSITY
THE GRADUATE SCHOOL NATURAL AND APPLIED SCIENCES

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : **Yusuf KUNT**
Signature : 
Date : **06.08.2020**

ABSTRACT

SHIP DETECTION AND CLASSIFICATION USING TYPE OF CONVOLUTION NEURAL NETWORKS

KUNT, Yusuf

M.Sc., Department of Electronic and Communication Engineering

Supervisor: Asst. Prof. Selma ÖZAYDIN

July 2020, 74 Pages

Ship classification and detection systems are used in many areas that may cause problems such as national and local defense in countries with intense coastal and strait crossings, ship traffic control, illegal fishing, pirate, human traffickers and the global trade chain.

In order to meet these kinds of needs easily, solutions with many different methods have been developed related to ship classification and detection. Research on these methods is divided into two groups intensely. These are the methods in used in Satellite and SAR images. Although the number of studies on these sources is high, the images that are come are not public. In addition, their resolution is not enough for ship classification and detection. As an alternative solution to these methods, CNNs with deep learning technology, have become quite popular.

Over the past 30 years, computer vision technologies have had a hard time helping people in visual tasks. However, major advances in deep learning technology today have enabled computers to process images as successfully and even better as a person.

One of the reasons why this thesis is based on CNN architectures is that the number of classes can be easily expanded according to the scenarios in the classification and detection problems of CNN models.

In this thesis, the most popular deep learning CNN architectures in the literature such as VGG16, VGG 19, DenseNet121 and Xception are used for ship classification. Also, CNN architecture YOLOv3 with high performance is preferred for ship detection. The results of these models are compared in this thesis. Our data set of 6 different ship types, created by us, is used for training and testing. Since CNN models require a large number of training data sets, the transfer learning technology is applied in these models in order to eliminate the problem of lack of data.

The contribution of our thesis to literature can be listed as comparing CNN model architectures for ship classification problems, investigating the effects of layer numbers in CNN model architecture and choosing the appropriate hyper parameter for this problem. For ship detection, we can evaluate the number of labeled data as an impact on performance.

Keywords: Convolutional Neural Networks, Transfer Learning, Own Dataset, Ship Classification, Ship Detection

ÖZ

EVRIŞİMLİ SİNİR AĞLARI KULLANILARAK GEMİ TANIMA VE SINIFLANDIRMA

KUNT, Yusuf

Yüksek Lisans, Elektronik Haberleşme Mühendisliği Anabilim Dalı

Tez Yöneticisi: Asst. Prof. Selma ÖZAYDIN

Temmuz 2020, 74 sayfa

Gemi sınıflandırma ve tanıma sistemleri, denize kıyısı ve boğaz geçişleri yoğun olan ülkeler de ulusal ve yerel savunma, gemi trafik kontrolü, kaçak balıkçılık, korsan, insan tacirleri ve küresel ticaret zinciri gibi sorun oluşturabilecek birçok alanda kullanılmaktadır. Bu sorunlara erken müdahale edebilmek için gemi sınıflandırma ve tanıma ile ilgili farklı farklı çözüm metodları geliştirilmiştir. Bu metodlar üzerinde yapılan araştırmalarda yoğun olarak iki gruba ayrılmıştır. Bunlar Uydu ve SAR görüntüleri üzerine yapılan çalışmalardır. Bu kaynaklar üzerinde yapılan çalışma sayısı fazla olsada gelen görüntüler herkesin kullanımına açık değildir. Ayrıca çözünürlükleri gemi sınıflandırma ve tanıma için yetersiz kalmaktadır. Bu metodlara alternatif çözüm olarak derin öğrenme teknolojisi olan CNN' ler oldukça popüler hale gelmiştir.

Geçtiğimiz 30 yıl boyunca, bilgisayar görme teknolojileri görsel görevlerde insanlara yardım etmekte zorlandı. Ancak, bugün derin öğrenme teknolojisindeki büyük ilerlemeler, bilgisayarların görüntüleri bir kişi kadar başarılı ve hatta daha iyi işleme yeteneğini sağlamıştır.

Tezimizin CNN mimarileri üzerine kurulmasının sebeplerinden birisi de CNN modellerin sınıflandırma ve tanıma problemlerindeki senaryolara göre kolaylıkla sınıf sayılarının genişletilebilir olmasıdır.

Bu tezde gemi sınıflandırma için VGG16, VGG 19, DenseNet121, Xception gibi literatürdeki en popüler derin öğrenme CNN mimarileri kullanılmıştır. Ayrıca gemi tanıma için ise YOLOv3 gibi yüksek performans elde edilen CNN mimari tercih edilmiştir. Tezimizde bu modellerin sonuçları karşılaştırılmıştır.

Eğitim ve test için kendimizin oluşturduğu 6 adet gemi den oluşan veri setimiz kullanılmıştır. CNN modelleri çok fazla sayıda eğitim seti gerektirdiği için öğrenme aktarım teknolojisi, veri yetersizliği sorununu ortadan kaldırmak amacıyla bu modellerde uygulanmıştır.

Tezimizin literatüre olan katkısını, gemi sınıflandırma problemleri için CNN model mimarilerin karşılaştırılması, CNN model mimarisindeki katman sayılarının etkilerinin araştırılması ve bu probleme yönelik uygun hiper parametre seçimi olarak sıralayabiliriz. Gemi tanıma için ise etiketli veri sayısının performans üzerindeki etki olarak değerlendirebiliriz.

Keywords: Convolutional Neural Networks, Transfer Learning, Own Dataset, Ship Classification, Ship Detection

ACKNOWLEDGMENTS

I want to thank Selma ÖZAYDIN for her guidance, advice, criticism, encouragements, and insight throughout the research.

Furthermore, I would like to thank my wife Seda KUNT and my son Demir KUNT for their patience and never-ending support. My dear colleague Mustafa DEMİR, for his helpful feedbacks on my thesis and for being the best communicator.

TABLE OF CONTENTS

ABSTRACT	iv
ÖZ	vi
ACKNOWLEDGMENTS	viii
TABLE OF CONTENTS	ix
LIST OF FIGURES	xii
LIST OF ABBREVIATIONS	xv
CHAPTER I	1
INTRODUCTION	1
1.1. OVERVIEW OF THE OBJECT CLASSIFICATION AND DETECTION	1
1.2. PROBLEM DEFINITION AND MOTIVATION	2
1.3. STRUCTURE OF THE THESIS ORGANIZATION	4
CHAPTER II	5
TECHNICAL BACKGROUND	5
2.1. RELATED WORK EXPERIENCE	5
2.2. MACHINE LEARNING (ML)	7
2.3. DEEP LEARNING	8
2.4. NEURAL NETWORK	9
2.5. CONVOLUTIONAL NEURAL NETWORK	10
2.5.1. Object Classification and Detection	12
2.6. BACK PROPAGATION	20
2.7. TRANSFER LEARNING	21
2.8. IMAGENET DATABASE	22
CHAPTER III	23
CNN HYPERPARAMETERS	23
3.1. SELECTED HYPER PARAMETERS	23

3.2. ACTIVATION FUNCTIONS	24
3.2.1. Rectified Linear Unit (ReLU) Function	24
3.2.2. Softmax Function	25
3.2.3. Sigmoid Function	25
3.3. OPTIMIZATION FUNCTIONS.....	26
3.4. PERFORMANCE EVALUATION METRIC.....	28
3.4.1. Confusion Matrix.....	29
3.4.2. Accuracy	29
3.4.3. Precision.....	29
3.4.4. Recall.....	29
3.4.5. F1-Score	30
3.4.6. Mean Average Precision (mAP)	30
CHAPTER IV	31
EXPERIMENTAL STUDIES FOR SHIP CLASSIFICATION AND DETECTION .	31
4.1. HARDWARE AND SOFTWARE	32
4.2. DATASET PREPARATION	32
4.3. WORKING FLOWCHART	37
4.4. SHIP CLASSIFICATION AND DETECTION TASKS	37
4.4.1. Ship Classification Experimental Methodology, Result and Analysis	38
4.4.2. Ship Detection Experiments' Methodology, Result and Analysis	45
4.5. COMPARISON RESULTS WITH ANOTHER SIMILAR THESIS	48
REFERENCES	51

LIST OF TABLES

Table 1 Used Optimization Functions	27
Table 2 Hardware and Software Specifications	32
Table 3 Train/Validation/Test Data Set of Ship Images	34
Table 4 Performances' comparison	39
Table 5 Ship classification performance for the modified models retrained on our dataset using transfer learning	40
Table 6 Optimization Hyperparameters	43
Table 7 Comparison Number of Labeling Ship Image performance	48

LIST OF FIGURES

Figure 1 Ship Traffic Density [9]	3
Figure 2 Number of Commercial Ships [21]	5
Figure 3 Relationship between Artificial Intelligence, Machine Learning and Deep Learning [36]	8
Figure 4 A simple example of a perceptron neuron structure [68]	10
Figure 5 LeNet CNN Architecture [47]	11
Figure 6 A Sample of Object Classification Output	12
Figure 7 VGG19 Architecture [77]	14
Figure 8 VGG16 Architecture [77]	14
Figure 9 Connection strategy of DenseNet [17]	15
Figure 10 DenseNet Architecture	16
Figure 11 Xception Workflow [80]	17
Figure 12 Xception Architecture [77]	18
Figure 13 A Sample of Object Detection Output	19
Figure 14 Darknet-53 Image [32]	20
Figure 15 The images which are sample data sets taken from ImageNet data set .	22
Figure 16 ReLu Graph	25
Figure 17 Sigmoid Function	26
Figure 18 Distribution on the data set	28
Figure 19 The example Python crawler code	33
Figure 20 The distribution of the classes in our own Data Set	35
Figure 21 Randomly selected sample pictures	36
Figure 22 Sample ship labeling image using LabelImg Tool	46
Figure 23 LabelImg Tool output format	46
Figure 24 YOLOv3's architecture to work with Darknet [4]	47

LIST OF EQUATIONS

Equation 1 Formulation of Multi-Layered Structure of CNN [74]	11
Equation 2 Activation Function	24
Equation 3 ReLu Function	24
Equation 4 SoftMax Function	25
Equation 5 Sigmoid Function	26
Equation 6 Accuracy	29
Equation 7 Precision	29
Equation 8 Recall	29
Equation 9 F-1 Score	30
Equation 10 Formula of mAP	30

APPENDICIES

APPENDIX A: FLOWCHART	64
APPENDIX B: VGG16/VGG19/XCEPTION/DENSENET121 CONFUSION MATRIX	65
APPENDIX C: EVALUATION PREDICTION FOR VGG16	66
APPENDIX D: EVALUATION PREDICTION FOR VGG19	67
APPENDIX E: EVALUATION PREDICTION FOR XCEPTION	68
APPENDIX F: EVALUATION PREDICTION FOR DENSENET121	69
APPENDIX G: FULLY CONNECTED LAYER NEURON SIZE PERFORMANCE COMPARISON FOR MODIFIED MODELS	70
APPENDIX H: I USE HYPERPARAMETERS FOR EACH MODIFIED MODELS	71
APPENDIX H: I USE HYPERPARAMETERS FOR EACH MODIFIED MODELS (Continued)	72
APPENDIX I: TEST RESULTS ON OUR OWN 3000 DATASET	73
APPENDIX J: TEST RESULTS ON OUR OWN 6000 DATASET	74

LIST OF ABBREVIATIONS

AE: Auto-Encoder
AI: Artificial Intelligence
AIS: Automatic Identification System
BN: Batch Normalization
CFAR: Constant False-Alarm Rate
CNN: Convolutional Neural Network
CONV: convolutional
CPU: Central Processing Unit
FN: false negative
FP: False Perceived
GPU: Graphic Processing Unit
HOG: Histogram of Oriented Gradients
IDE: Integrated Development Environment
ILSVRC: ImageNet Large Scale Visual Recognition Challenge
IMO: International Maritime Organization
IR: Infrared
LR: Learning Rates
mAP: Mean Average Precision
MASATI: Maritime Satellite Imagery
ML: Machine Learning
MLP: Multi-Layer Perceptron
NN: Neural Network
PCA: Principal Component Analysis
RADAR: Radio Detection and Ranging
RBF: Radial Basis Function
RBM: Restricted Boltzmann Machine

R-CNN: Region-based Convolutional Neural Networks

ReLU: Rectified Linear Unit

SAR: Synthetic-Aperture Radar

SGD: Stochastic Gradient Descent

SIFT: Scale-Invariant Feature Transform

SSD: Single Shot Multibox Detector

SVM: Support Vector Machine

TP: True positive

VGG: Visual Geometry Group

WWF: World Wide Fund



CHAPTER I

INTRODUCTION

For countries with seaside and strait crossings, the classification and detection of ships are very important for country defense, supervision and control of critical areas such as maritime safety and traffic. Because these countries supply the basic requirements such as nutrition, shelter, protection, health, tourism, and ship trading and transportation by sea. According to report of WWF (World Wide Fund) in 2015, activities related to the maritime sector have a major impact on the global economy [1].

Real time information flow is necessary for correct maritime trade and transportation. This real time data flow provides that the sea trade and transportation is done correctly and taking precautions against the problems that may occur in the seas. As an example, it has become more important to develop anti-collision early warning systems, especially for the Bosphorus crossing points, where the local traffic density is high.

The purpose of this study is to examine how can we contribute using CNN technology for maritime safety and traffic control. In addition, it aimed to create infrastructure of a useful system for maritime practices and to give fast and precise results for the operations of supervision and surveillance of the seas. Despite of a limited number of our own ship data set, it is intended to show that the pre-trained models can be trained with the ship data set I created via transfer learning method.

1.1.OVERVIEW OF THE OBJECT CLASSIFICATION AND DETECTION

Hubel and Wiesel have found out the basis for the functioning of Object classification and detection systems during their cats' brains study [2]. With this study, they discovered that while examining cats' vision systems, some cells respond to lines with certain orientation.

With this study, they laid the foundation of object classification and detection. The ability to identify and classify objects in an image is one of the most basic requirements if it is desired to interact with the environment or scene where that object is located. Object classification is the identification and labeling of a dominant object in an image.

Object detection is the process of labeling the dominant object in the image and then placing the image in the image by determining the bounding box of the area occupied by the object [3].

Popular solution for object classification and detection in 2000s; scale-invariant feature transform (SIFT) [4] and histogram of oriented gradients (HOG) [5] developed by David Lowe. Today, object classification and detection has become popular according to the results obtained in the field of deep learning technologies [6].

There are three main reasons behind this popularity. The first is the use of large amounts of data sets, the second reason is the development of powerful graphic processing units, and third reason is the worldwide competition on object classification and detection such as ILSVRC, PASCAL VOC and Microsoft COCO [7].

1.2.PROBLEM DEFINITION AND MOTIVATION

The serious increase in ship density causes an increase in ship traffic density, compared to previous years and shown Figure 1 [8]. Therefore, real-time early warning systems have become important issues such as ship collisions, human trafficking, military areas, crossing traffic, pirate attacks, fishing, and local sea traffic. In addition, although marine and coastal surveillance has environmental importance, it has become an action that it motivates in socio-economic factors [1]. It has become imperative to seek alternatives for the solution of such problems in the shipping industry.



Figure 1 Ship Traffic Density [9]

Apart from Electro Optic and Camera sensor systems stationed on land or aboard, there are two systems in literature for ship detection and classification. The first is the ship detection and classification system with SAR images. SAR is an important system for oversight of maritime transport and the presence of ship docks [10]. However, this system is not enough to sensitively determine the shape of ships. Speckle noise and spatially varying reflectivity differences in the SAR images make it difficult to find the outer border of the ships. As the ships in the sea seem brighter than the water surface, segmentation is easier. However, segmentation of the ships that are standing on the shore, in the port or next to each other are quite difficult [11].

Another system for ship detection and classification is usage of satellites. Ship detection and classification is performed from different satellite images with growing of space technology and increasing number of satellites [12]. For satellites, stable and effective remote area detection and classification can be made in stagnant water, open air or marine conditions [13]. However, satellite images are not usually real-time data that are easily accessible [14]. Also, if whether is cloudy or seas are wavy, the number of false alarms in satellite images increases [13]. Therefore, ship detection and classification efficiency are very low for satellite images.

The techniques which are mentioned above have the performance problem in themselves for some classification and detection. The reason of motivation that the development of high-density data-based technologies such as deep learning has become attractive with the increase in the amount and quality of images obtained by image sensors (Electro Optics, Camera) in the maritime field. With these technologies, it has become easier

to develop CNN based ship images detection and classification algorithms [15]. Other motivation is that the CNN model works with high performance and accuracy against the noisy and blurred images.

1.3.STRUCTURE OF THE THESIS ORGANIZATION

In general, in the rest of this thesis, research of technical details, literature review and deep learning which is a sub-branch of machine learning are discussed in Chapter 2.

Object Classification and Detection, one of the deep learning technologies, are explained. In addition, information is given about the CNN model types used in this thesis and backpropagation, transfer learning techniques, which form the basic structure of CNNs. In this section, used ImageNet dataset for pre-trained models is additionally explained.

In Chapter 3, the hyper parameter types that are used in this thesis for CNN optimization and their working principles are explained. In addition, used performance evaluation metrics to evaluate the results of this thesis are explained.

Chapter 4 shows how constructed the experimental infrastructure of this thesis, the technologies used in the hardware and software, the stages of creating our own data set and the workflow diagram that is used in our experiments in general. In addition, methods of experiments for ship classification and detection are specified. In Chapter 5, the results and analyzes of the experiments mentioned in the previous section are discussed. In the last Chapter 6, the gains in ship classification and detection experiments are indicated. In addition, future studies have been mentioned to shed light on the next studies.

CHAPTER II

TECHNICAL BACKGROUND

In this chapter, results of research relevant with ship detection and classification are firstly discussed. Introduction to deep learning and basic techniques of deep neural networks is briefly explained. Existing methodologies are explained for Description of major technologies such as Backpropagation and Transfer Learning, Object classification (VGG16 [16], VGG19 [16], Densenet121 [17], Xception [18]) and Object detection (YOLOv3 [19]).

2.1. RELATED WORK EXPERIENCE

There are lots of studies for ship detection and classification. Because the number of ships in the world exceeds 50,000. As of January 2018, there were 53,732 ships in commercial fleets worldwide [20]. The figure 2 gives the ship increase rate in the world between 1980 and 2015.

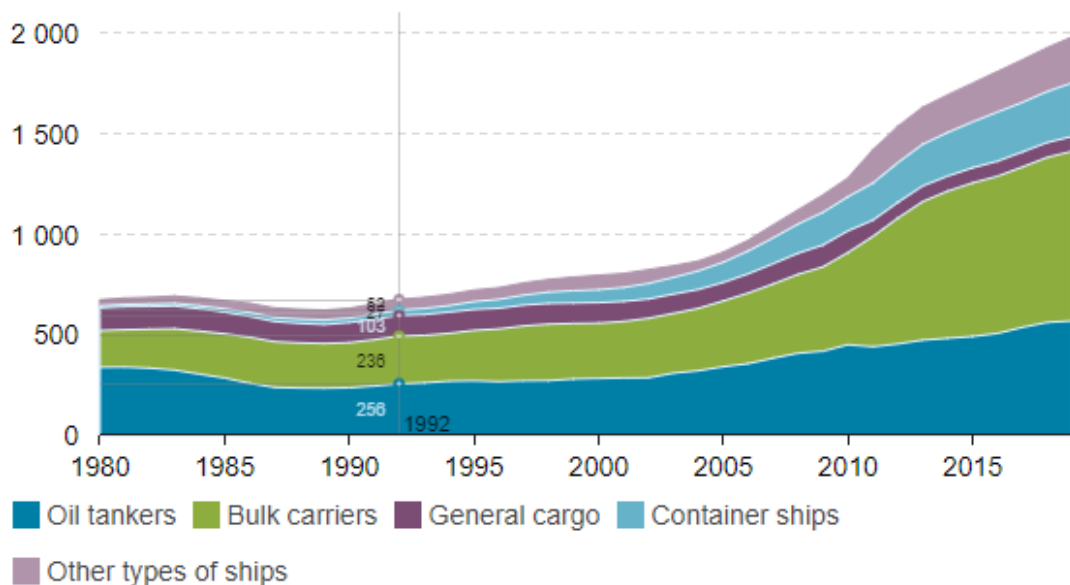


Figure 2 Number of Commercial Ships [21]

The quantity of ships at this rate is an indication that it is worth doing work on ship detection and classification as shown in the Figure 2 [21]. When performing literature

research, it is seen that there are lots of methods and trials for ship detection and classification. These methods were mainly created for images from SAR, satellite, land-based video camera systems and images collected from web sites [22].

One of the studies is detection of SAR Picture images-based CNN which is presented in. In this study, the data set is created with approximately 43,816 labelled SAR images. According to the experimental results, although there is no land-ocean segmentation, the rate of low number false alarms on their own trained is created and to obtain better ship detection model by modifying the SSD model with its datasets. With creating object detection models named Modified SSD-300 and Modified SSD-512 based on CNN Models, large scale ships are detected [23].

In the study in [24], aim is to produce a low false alarm by creating its own (MT-CNN) model on SAR labelled picture images provided by the GF-3 satellite. The main aim is to classify the targets in the sea and to detect the large-scale SAR images [24].

Nowadays, SAR technology is not affected by weather conditions, clouds, and sea waves. But it produces low resolution pictures. Despite that, satellite images are high resolution. This is big importance for Ship detection and classification [25] [26].

Ramani S, et al. [14] [5] have Ramani et al. developed a deep learning-based algorithm to automatically detect and segment ships from satellite images with their work. For automatic detection, they used MASK R-CNN architecture. For trained architecture weight functions, MS-COCO data set was used [14].

On the other hand, MASATI (Maritime Satellite Imagery) [27] data set was used to detect and classify small objects on the sea from satellite images based on CNN [28].

Besides, according to MASATI data set, compression's' performance of models, which are Retina Net [29] , Faster R-CNN [30] , YOLOv2 [31], YOLOv3 [32] was performed for object detection.

A distance metric learning depth function and deep convolutional neural network to classify 3965 different ships with 60 percent accuracy was used by Gundogdu S., et al [33]. MARVEL which is named as ship dataset is created. The MARVEL has 2 million images including 109 subclasses, 26 super classes [34].

2.2. MACHINE LEARNING (ML)

The term machine learning term is often the wrong term confused with artificial intelligence (AI), but machine learning is a sub-term of artificial Intelligence, [35] Relationship between AI, ML and Deep Learning is shown in Figure 3 [36] .Machine learning is an area that focuses on the creation of algorithms that make data-based predictions [37].

Machine learning technology has innovative applications in a variety of fields such as engineering, medicines, agriculture, astronomy, sports, education, etc. According to the researchers, problems in these application areas are usually solved as prediction and classification with labeled and unlabeled data [38] [39] [40].

Machine learning is divided into three: Supervised, unsupervised and reinforcement learning [41]. Supervised learning defines a problem class which involves using a model to learn a mapping between input samples and the target variable [42]. Unsupervised learning defines a problem class that involve using a model to define or extract relationships in data [43]. Reinforcement Learning defines a class problem where an agent works in an environment and needs to learn to work using feedback [41].

Deep learning is a subfield of Machine Learning methods based on the representation of learning data as shown in Figure 3 [44]. It is also a kind of ML method in which deep learning hierarchical architectures are used to classify patterns and to learn feature or representation learning [45]. Recently, it has been used extensively in deep learning research and has been widely used with many different applications, including detection of attacks, image classification, extraction and analysis of data from video files, analysis of social network data, data mining and information security [46]. As a machine learning tool, deep neural networks are very effective in understanding high-dimensional data such as images [43]. Deep learning will be explained in detail in the continuation of this thesis.

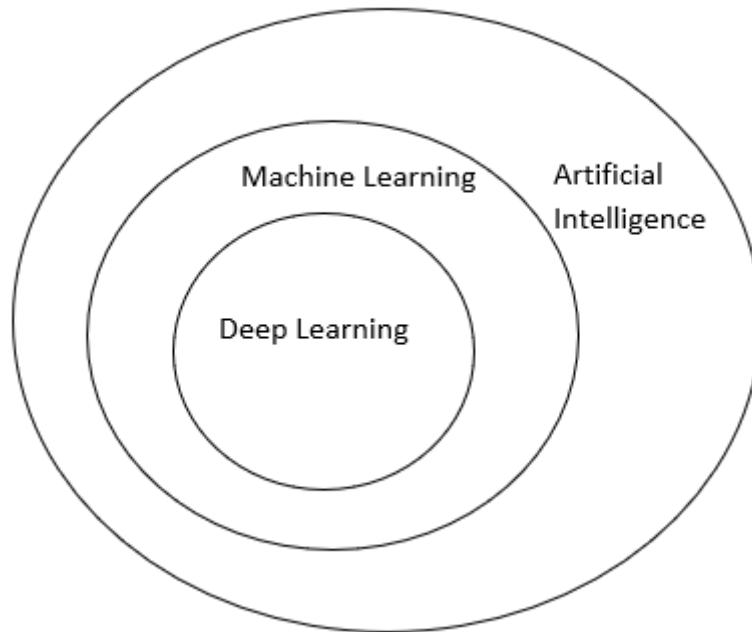


Figure 3 Relationship between Artificial Intelligence, Machine Learning and Deep Learning [36]

2.3. DEEP LEARNING

Deep learning architectures are popular in these times because of some technological developments in the Computer Vision studies. At the beginning of these technologies' developments, back propagation of errors in multiple layers of deep architectures is the most important deficiency of learning [47].

Back propagation algorithms have been introduced in previous years. But Yann Le Cun and his colleagues [48] developed the first successful deep neural network application on mailbox articles. This Deep neural network was successfully completed but this training took 3 days completely. This long time was questioned the appropriateness of the study. After this study, Yann Le Cunn and his colleagues [49] performed classification of Handwritten numbers (MNIST) with this deep learning network "LeNet" which was developed before and Back propagation algorithms.

Back-propagation algorithms were developed by Hinton between the 1980s and 1990s, but topics such as overfitting, data shortage and limited computing have reduced the popularity of deep learning in the early 2000s [50].

The popularity of deep learning started again in 2006 [51]. The reasons why deep learning is popular in the field of computer vision are stated in order; the creation of large-scale datasets such as ImageNet [52] and its spread as open source. High performance parallel computing rapid development, such as GPU systems, has increased approximately 1000 times over a 10-year period [53].

Significant advances in the design of deep learning network structures and changing the training styles of networks and in addition to developing techniques such as Auto-Encoder (AE) [54], Restricted Boltzmann Machine (RBM) [55], dropout [56], data augmentation [57], batch normalization (BN) [58], AlexNet [59], GoogLeNet [60], VGG [16], ResNet [61] made an important contribution to the development of popular CNN architectures.

2.4. NEURAL NETWORK

Computer Vision problems are based on object detection. For this reason, to be successful of object detection, valuable information in pictures and videos must be extracted. The result of solving this problem is associated with many applications [50]. These applications are image classification [62], human behavior analysis [63], face recognition [64] and autonomous driving [65]. The basis of these above applications is based on Neural Network (NN) algorithms [66].

NN was developed in the 1950s and 1960s by scientist Frank Rosenblatt and inspired by earlier studies by Warren McCulloch and Walter Pitts. Neural Networks consist of sequential layer sequences and these layers are composed of neurons. Therefore, neurons are the main computing units in neural networks. These neurons are called perceptron's [67]. Artificial neurons are created based on the neurons that make up the human nervous system. Neurons receive input signals from one or more sources and produce a single output based on these inputs. The steps for creating the output are as follows; First, each multiple neuron entry and weights are multiplied. Then these values are summed up and added bias value. Then the production to be passed through the activation function is completed (see Figure 4). These real weight numbers use to explain the importance of neurons' different input signals [68].

Weights are real numbers that are used to express the relative importance of different input signals to that neuron. It is these weights, and the biases, that are updated during CNN training to create a model suited to a specific task such as the ship classification and detection tasks in this thesis. An activation function is then used to determine the final output of the neuron [68].

A simple example of a perceptron neuron structure is shown below in Figure 4. In this example, three inputs which are indicated as “ x_0 , x_1 and x_2 ” are available. Then these inputs are multiplied by their weights which are shown as “ w_0 , w_1 , w_2 ” and sum of all and added to bias(b). Then these values pass through the activation function (f). With these processes, the final output is created [68].

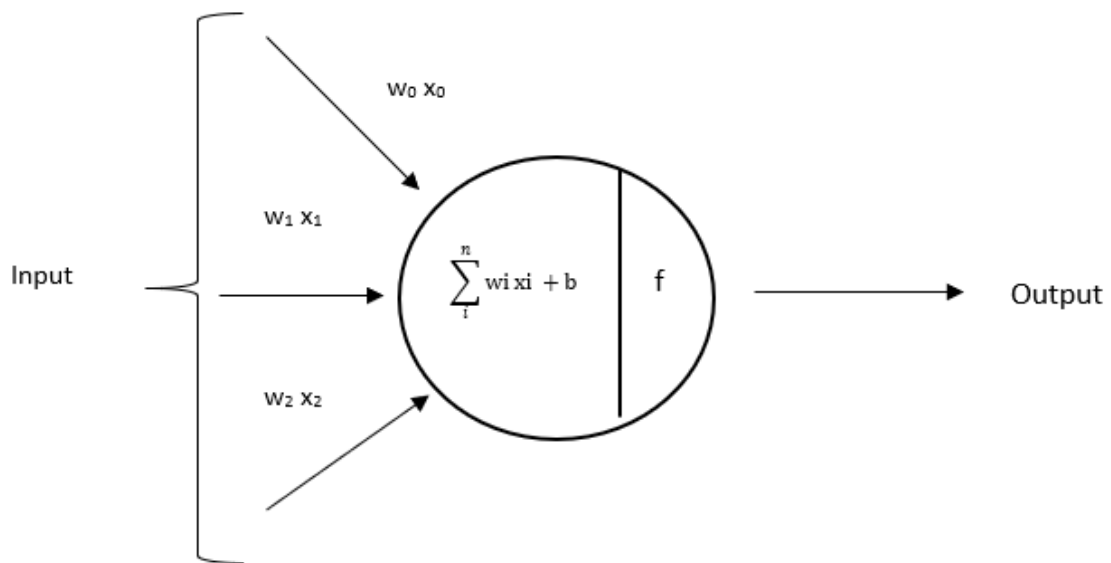


Figure 4 A simple example of a perceptron neuron structure [68]

2.5. CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks (CNN) have achieved significant success in the perception and detection of the image [69].

Convolution algorithm has been developed inspired by the visual mechanism of animals. The cells in the seeing process are divided into some structures according to their duties. For example, simple nerve cells detect edge-like features, while complex nerve cells are concentrated on the whole image [70]. The presentation of such visual neural networks

with a layered structure is called a Multi-Layer Perceptron (MLP). In other words, it is called the Convolutional Neural Network [71]. If we mathematically interpret the convolution process, it is known as the response of a neuron to its own stimuli [72]. The first CNN network developed by Yan LeCun, LeNet is the architecture that emerged in 1988. The development of this architecture lasted until 1998 [73].

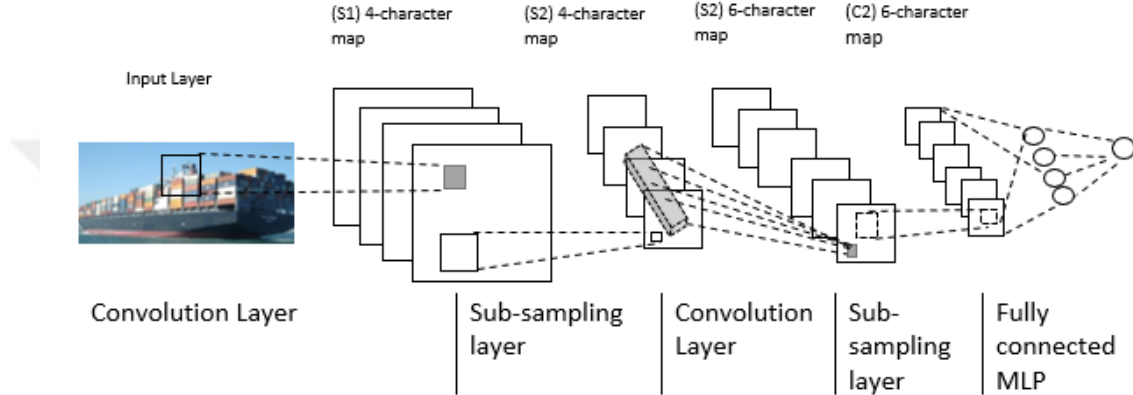


Figure 5 LeNet CNN Architecture [47]

The network indicating in the figure 5 is LeNet CNN Architecture. The input layer consists of the sub-sampling layer and fully connected layers. The function of each layer is to produce their own feature inference maps [47].

Convolutional Neural Networks consist of one or more layers. These layers are the input layer, sub-sampling, and fully connected layer [47].

Convolution Neural Networks are popular because they have a multi-layered structure. These layers consist of input, multiple hidden and output layers. The formulation of these layers is as follows [74].

$$X_j^l = f \left(\sum_{i=1}^d X_i^{l-1} W_{ij}^l + b_j^l \right)$$

Equation 1 Formulation of Multi-Layered Structure of CNN [74]

In this formula, the “i” value is the index value of the input layer. “J” is the index value in hidden layers, W_{ij}^l indicates the weight between the input layer unit “i” and the hidden layer unit “j”, “ b_j^l ” indicates the additional bias given by each layer. $f(\cdot)$ indicates the output activation function [74].

2.5.1. Object Classification and Detection

This section is explained as two main topics as “Object Classification” and Object Detection.

2.5.1.1. Object Classification

Object classification is to reduce the information in an image to a single number by determining the class of the object, as shown in the example pictures below. The object class expression here refers to an abstract class such as "the class of all ships". The object classification approach is to focus on the object to classify and sort the objects in the image according to some features such as brightness, contrast, and position. They do not need to locate the classified objects on the image, such as object detection.



Ship Type : Military
Confidence=0.96



Ship Type : Container
Confidence=0.98



Ship Type : Tanker
Confidence=0.88



Ship Type : Yacht
Confidence=0.96



Ship Type : Fishing
Confidence= 0.92



Ship Type : Cruise
Confidence= 0.94

Figure 6 A Sample of Object Classification Output

2.5.1.1.1. VGG

VGGNet was developed by the Visual Geometry Group at Oxford University and the Model name consists of the initials of the group. VGG-16 - Karen Simonyan and Andrew Zisserman introduced the VGG-16 architecture in Very Deep Convolution Network articles for Large Scale Image Recognition [6].

In this model architecture they introduced, they wanted to investigate the effect of CNN depth on the accuracy of large-scale image recognition settings [75]. The network architecture they created includes 13 convolutional (conv) layers, 3 fully connected layers, 3 max-pooling layers, and a SoftMax conv for classification [76].

2.5.1.1.1.1. VGG-16

VGG-16, a variant of VGGNet, consists of an architectural structure as seen in the picture below. The details of this architect used in this thesis are as follows. The first and second layers consist of 64 feature kernel filters and the filter size is 3 x 3, and the next two layers (stride) are passed to the maximum pooling layer. In the third and fourth convoluted layers, 124 feature kernel filters and filter size are 3 x 3, and the next 2 strides (maximum stretch) layers are passed. The fifth, sixth and seventh layers consist of 256 feature kernel filters and the filter size is 3 x 3, and the next 2 layers of stride pass to the maximum pooling layer. Between the eighth and thirteenth layers, 512 feature kernel filters and filter size are formed as 3 x 3, and then the maximum 1-layer (stride) pooling layer is continued. Fourteen and fifteen layers are fully connected layers consisting of 4096 units and finally the Sixteenth layer consists of a SoftMax output layer consisting of 1000 units. VGG-16 model architecture is created in this way [76].

2.5.1.1.1.2. VGG-19

VGG-19 models are one of the 19-layer deep convolutional neural networks that increase the depth of the network and emphasize the concepts of design and depth [1]. One of the differences between VGG-16 and VGG-19 is that VGG-16 has 138 million and VGG-19 has 144 million computational parameters [75]. If we examine the architectural structure of VGG-19; Unlike VGG-16, it is more than 3 Convolutional Layers. The features of the other layers are not different from the VGG-16 architecture. To better understand this architectural difference, the following picture has been created.

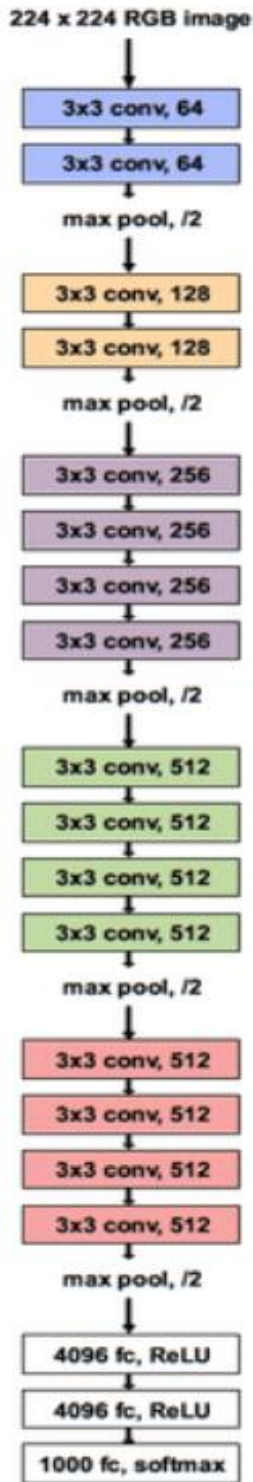


Figure 7 VGG19 Architecture [77]

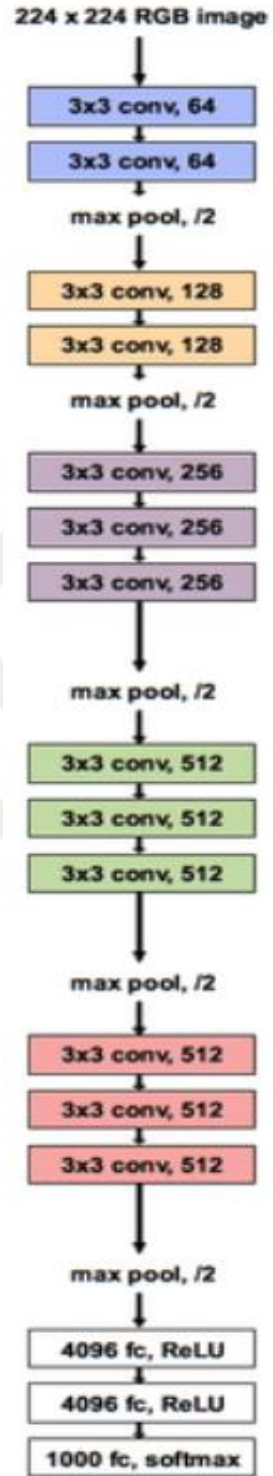


Figure 8 VGG16 Architecture [77]

2.5.1.1.2. DenseNet

DenseNet was released by Huang et al in 2017 [17]. Their aim in creating this model has shown that it is more accurate and efficient to start the creation of convolutionary network structures in a hierarchical structure and to train the models with this approach, the closer to the input and output layers [78].

The basic idea of DenseNet, which they create on these approaches, is that every layer in the network is directly connected to the front layers [2]. Connection strategy of DenseNet is shown in Figure 9 [17]. The advantages of this architecture are: Less parameters, Overfitting prevention capability, being able to create deeper layered architectures with fewer parameters [79].

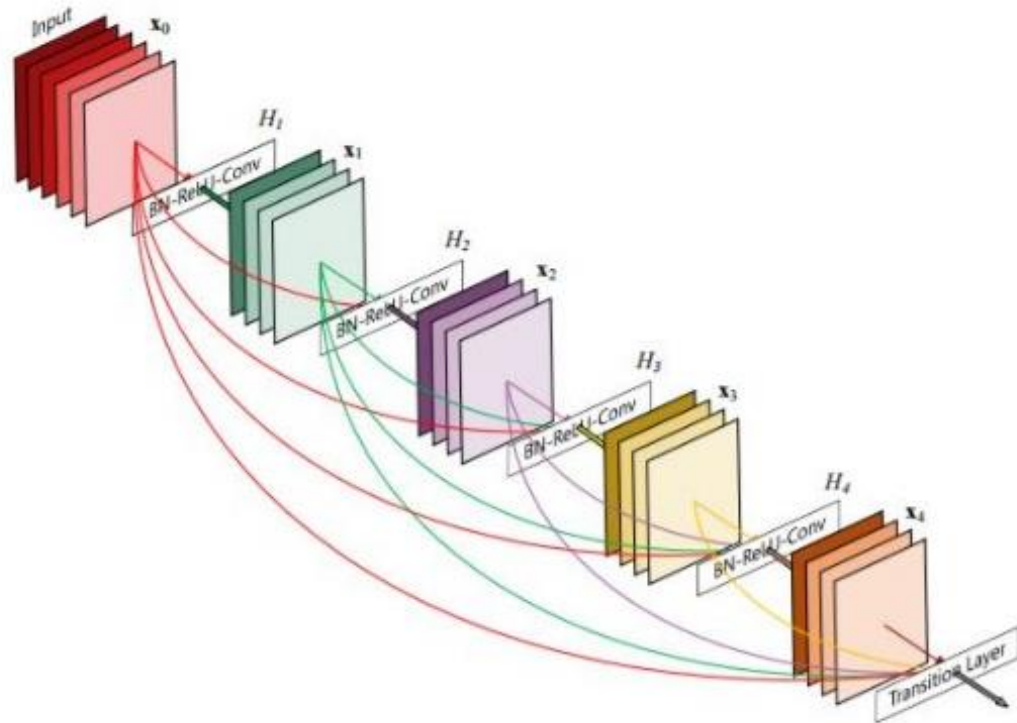


Figure 9 Connection strategy of DenseNet [17]

In Figure 10, DenseNet architecture is given in detail. The entire network consists of a first layer, three transition layers, and four layers of layers.

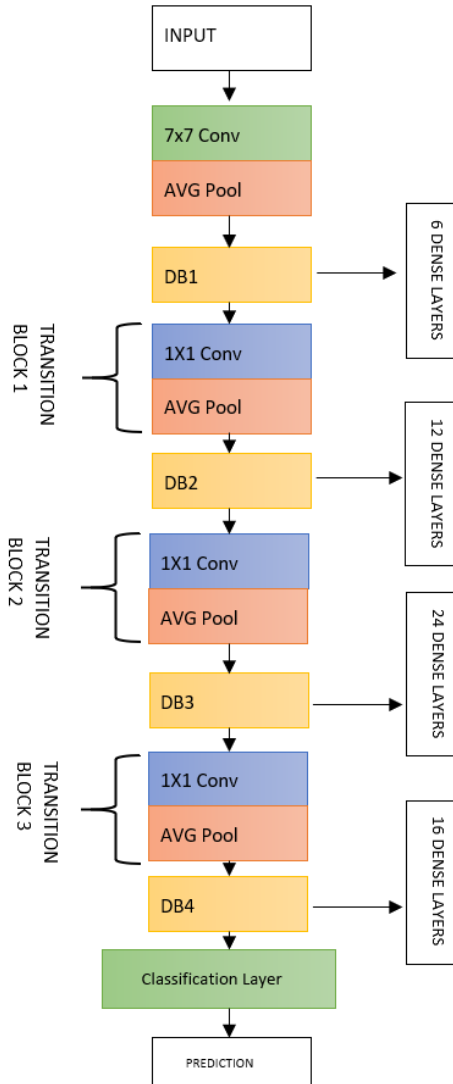


Figure 10 DenseNet Architecture

2.5.1.1.3. Xception

Xception was created by Chollet in 2017 and is a deep convolutional neural network architecture with in-depth detachable convolutions [80]. Xception is an extension of the inception architecture that replaces standard Inception modules with deeply separable convolutions [81]. In the experiment with the ImageNet dataset, it performed better than the Inception architecture [80].

The general architecture of the model is as seen in Figure-12. The work of the architecture goes through Entry Flow, then Middle Flow, which is repeated eight times, and

finally Exit Flow. As stated in Figure 11, Xception is structured in 14 modules with 36 non-linear residual connections, except 36 Convolutional layers and first and last modules [80]. It was observed that it expanded transversely according to Xception, DenseNet and VGG architectural structures.

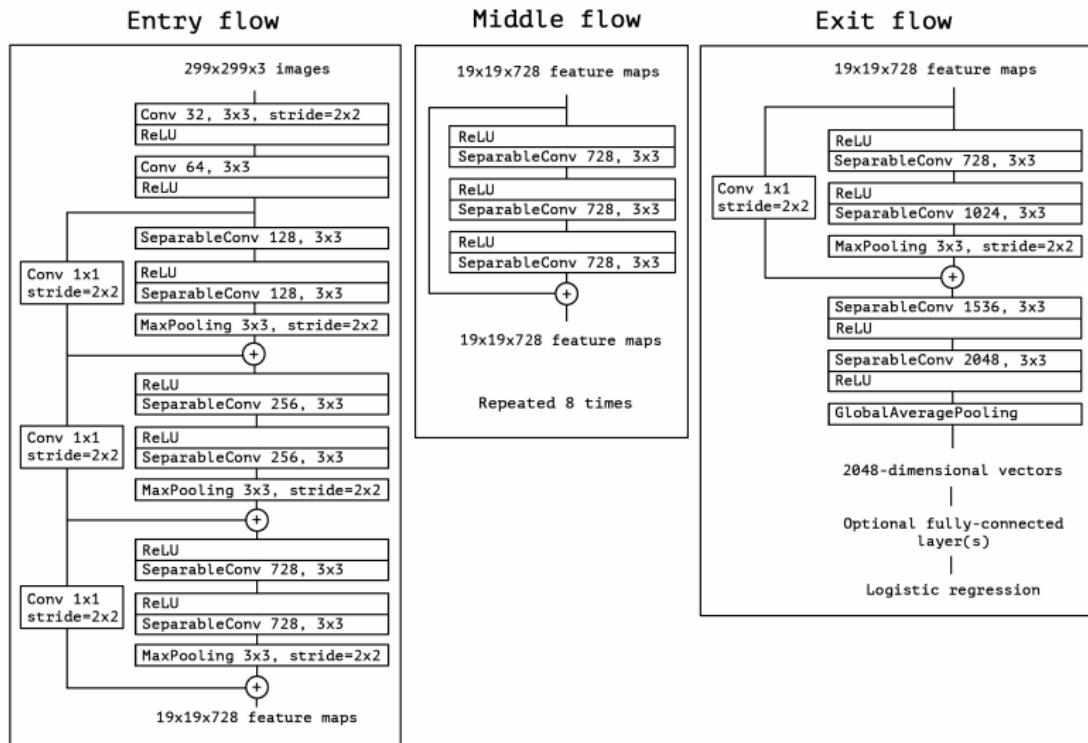


Figure 11 Xception Workflow [80]

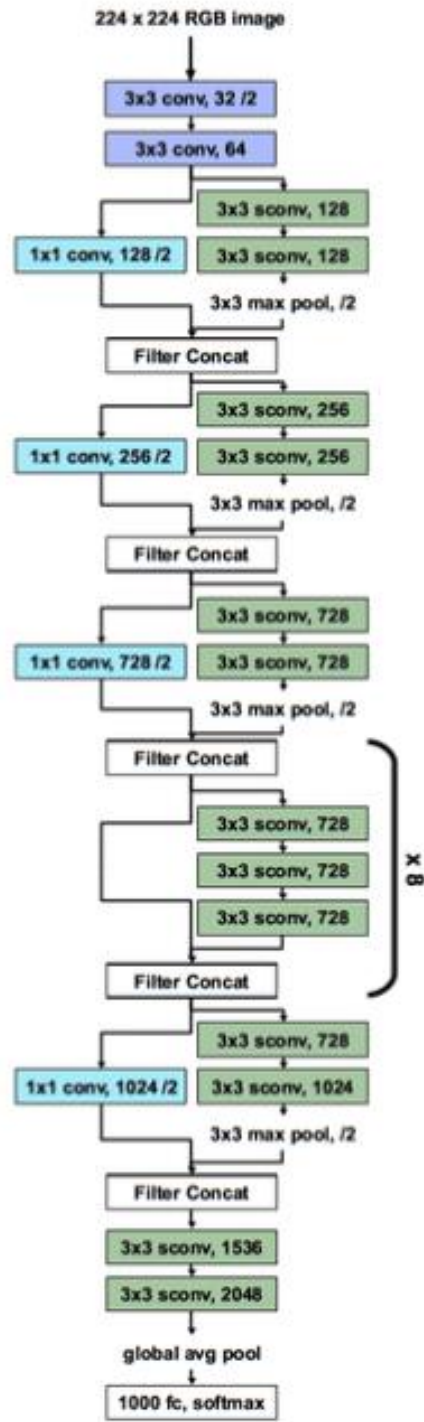


Figure 12 Xception Architecture [77]

2.5.1.2. Object Detection

Object detection finds the location of an object in the image, then guesses and classifies the object. As seen in Figure 13, the areas of the ship classes in the picture have been determined and classified as a frame.




Figure 13 A Sample of Object Detection Output

2.5.1.2.1. YOLOv3

YOLO is one of the most advanced object detection and localization solutions [31]. This object detector, which appeared in 2015, has three versions today: YOLO [82], YOLOv2 [31] and YOLOv3 [32]. The release reasons for these versions should perform differently. The version used in this thesis is YOLOv3. YOLOv3 is a one-stage object identification system that focuses heavily on extraction speed and works in real-time. One-stage detectors that appeared in recent years with YOLOv3 are SSD [83] and RetinaNet [29]. The general features of the one-stage detectors take an image and create a bounding box with direct class estimates [84]. This method allows one-stage detectors to run faster. The network structure of Yolo v3 consists of 3 layers. These are called backbone network Darknet-53, the up-sampling network, and detection layer or YOLO layer [85]. Darknet-53 or the backbone is used to extract features from the input image and consists of 53 convolutional layers as

seen in Figure-14 and there are 5 residual layers in total, each residual layer consists of 3 x 3 and 1 x 1 convolutional layers.

Up sampling network is combined with the previous layer and detection of small objects is used. Detection Layer is where the objects are detected and created by the bounding box. Detection layer and up sampling layer use the features of Darknet53 in 3 different scales for object detection.



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	128 × 128
	Convolutional	64	3 × 3	
	Residual			
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	64 × 64
	Convolutional	128	3 × 3	
	Residual			
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	32 × 32
	Convolutional	256	3 × 3	
	Residual			
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	16 × 16
	Convolutional	512	3 × 3	
	Residual			
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	8 × 8
	Convolutional	1024	3 × 3	
	Residual			
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 14 Darknet-53 Image [32]

2.6. BACK PROPAGATION

The image information spreads throughout the network through the neuron system, inlet, secret, and forward during training in a CNN network. This spread is called forward propagation. This propagation continues until it generates a cost in the output node [86].

Back-propagation is the main body of the learning algorithm during the training of CNN. It gives weight outputs of a neuron during training and compares the expected outputs with these outputs in back-propagation and calculates a measure of error. By taking partial

derivatives of this calculation process, it monitors the change and trend in each step. Based on these calculations, it is seen how much affects the final loss in each neuron. These weight changes vary depending on the optimizer methods [49].

In a neural network system, weights of all neurons are selected and trained. This approach shows the error rates of weights in target value during the training. One of the methods to optimize these error rates is gradient descent. It can take quite a while to calculate error rates in a gradient descent network and does not guarantee to minimize error rates, but in this thesis, it worked well in the proper configuration. At the beginning of the training of the neural network, random values are assigned to the weights of the network neurons. An input vector spreads forward along the neural network and cost is obtained. This is called forward propagation. The output obtained because of the advanced training of the neural network is compared with the desired output using a loss function. The result is called the error value. Error values are then propagated to update neuron weights and are called back-propagation. Finally, neuron weights are updated by calculating the gradient of the weights and subtracting a ratio of the gradient from the weights. This rate is called the learning-rate. Back propagation algorithm continues to update neuron weights and error-rate after each new input vector. The main purpose of this process is to reach optimum weight values [43] [42] [87].

2.7. TRANSFER LEARNING

In this thesis, used pretrained models on ImageNet are VGG-16, VGG-19, DenseNet121, Xception and YOLOv3.

The reason for preferring transfer learning technology is that it is suitable to work on a small set. Therefore, this technology has provided great convenience for those who work with a small data set. Transfer learning, by definition, is to create a new model by training it on a previously trained model with our own data set [88]. Models such as VGG, Xception are models trained on data sets such as ImageNet. The advantage of transfer learning is to save time and achieve good performance. The reason we save time is that pre-trained models do not have to repeat previously removed features [89].

One of the advantages of transfer learning is that training a small number of examples without the need for a new classifier for object classification in the field of computer vision

and will significantly eliminate the risk of overfitting [90]. In this thesis, Transfer learning technology is the technology that we use as basic technology like backpropagation. In the continuation of the thesis, details will be given about the pre-trained models that we use and compare.

2.8. IMAGENET DATABASE

The data set for the pre-training of the VGG16, VGG19, DenseNet-121, Xception and YOLOv3 models used in this thesis is ImageNet. ImageNet was developed in 2009 by Deng and colleagues. This assists researchers due to its accuracy, diversity, hierarchical structure, and a free data set [91]. The first version has a total of 5247 synsets, 12 subtrees and a total of 3.2 million images. Today, there are 21841 synsets, and about 14 million images [92]. These subtrees generally consist of categories such as mammal, bird, fish, reptile, amphibian, vehicle, furniture, musical instrument, geological formation, vehicle, flower, fruit [91].

The purpose of our selection of this data set is that it includes the ship classes in our problem. The following are examples of ship pictures in Figure 15 ImageNet data set.



Figure 15 The images which are sample data sets taken from ImageNet data set

CHAPTER III

CNN HYPERPARAMETERS

3.1. SELECTED HYPER PARAMETERS

While designing CNN models, the algorithms or techniques used in the model bring some parameters that the designer should decide what should be, and these parameters are called Hyper parameters [93]. In this thesis, the Random Search method published by Bengio in 2012 was first used for hyper-parameter tuning [94]. The purpose of choosing this method; Instead of trying all combinations between hyper parameters, it is to work on randomly selected parameters. It is thought that this method will be faster when determining the performance rates of the models. The hyper parameters used in the thesis when creating our CNN models are batch size, training epochs, learning rate, dropout, activation functions (ReLU, Softmax and Sigmoid) and optimization functions (Adam, Adadelata, SGD and Adamax).

Batch size is divided into pieces according to the value set as the batch value and training of the model is performed on this iteration with each iteration [95]. While the model is being trained, all the data is not attended at the same time. They take part in education in a certain number of pieces. This process is repeated in each training step and the most suitable weight values are tried to be calculated for the model. Each of these training steps is called “epoch” [96].

Dropout is called randomly dropping of data passing through a layer in CNN models. In this way, the trained network does not learn by memorizing the values of the input parameters [97].

The learning rate is a small number that determines which part of the gradients should be used for weight update. Selecting the learning rate ratio as small or large may cause the update rates to decrease and increase on the network, and these approaches directly contribute to the learning of the network [42].

3.2. ACTIVATION FUNCTIONS

The activation function, a neuron in a neural network, is the basic calculation unit that receives “n” inputs and produces a single output [9]. An artificial neuron calculates the sum of the weight’s “w” of the “x” inputs and adds a bias value to produce a “y” output as follows [98]. As shown in the equation below, a “y” output is obtained. “ α ” is the activation function, “x” is the input value, “w” is the weight of the neuron, “b” is bias. The activation function is used here to control the “y” value, that is, to decide whether a neuron will be active or not [43].

One of the tasks of the activation function is to make predictions with the function itself in the feed forward step. Another task is to contribute to learning with derivative of the function in the back-propagation step [99]. Activation functions used in this thesis are Relu, Softmax and Sigmoid.

$$y = \alpha(w_1 x_1 + w_2 x_2 + \dots + w_n x_n + b)$$

Equation 2 Activation Function

Where “ α ” is the activation function, where “x = input”, “w = weights”, “b = biases”. The functions that can be used for activation functions are explained below.

3.2.1. Rectified Linear Unit (ReLU) Function

ReLU is the most commonly used activation function today [100]. ReLU is often used in convolutional neural networks and deep learning [99]. In this function, when the input value is below zero, the output is zero, but if the input value is above zero, the output is equal to the input value and a linear relationship occurs with the dependent variable [99]. Any negative value is returned by ReLU as shown in Equation 3 and Figure 16 in the graph. Converting all negative values to 0 prevents the model from being properly trained. For this reason, the model cannot be trained with negative values [98]. It performs better when compared to the sigmoid and tanh activation functions [101].

$$f(x) = \max(0, x) = \begin{cases} x_i, & \text{if } x_i \geq 0 \\ 0, & \text{if } x_i < 0 \end{cases}$$

Equation 3 ReLu Function

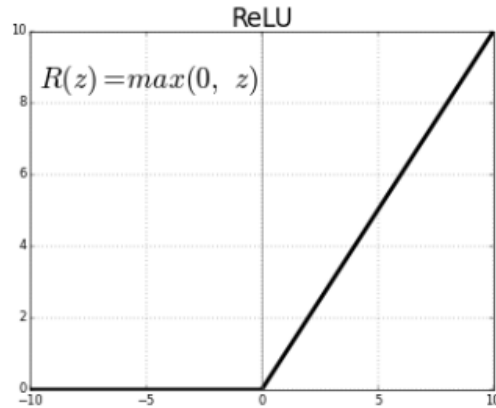


Figure 16 ReLu Graph

3.2.2. Softmax Function

Softmax is another activation function. As a result of Softmax, probability value is generated for the similarity of the test input for each class [43]. As a result of Softmax, probability value is generated for the similarity of the test input for each class and the score values given by the artificial neural network model as output are non-normalized values [102]. As stated in Equation 2, Softmax normalizes these values and converts them into probability values. This means that the probability of the input belonging to a class is determined by producing values in the range of 0-1 [98]. It is preferred in models that need more than two classifications and especially in the output layer of deep learning models [103] [59].

$$\text{Computation Equation } f(x_i) = \frac{\exp(x_i)}{\sum_j \exp(j)}$$

Equation 4 SoftMax Function

3.2.3. Sigmoid Function

Sigmoid function is a continuous derivative function and it is a type of function used in artificial neural network applications because it is not linear as stated in the equation [104]. As shown in the figure, it produces a value between 0 and 1 for each of the input values [99]. The major disadvantage of this function is that in deep networks consisting of many layers, the elements that derive after a certain point approach towards zero [105].

Computation Equation $f(x) = \left(\frac{1}{1+exp^{-x}}\right)$

Equation 5 Sigmoid Function

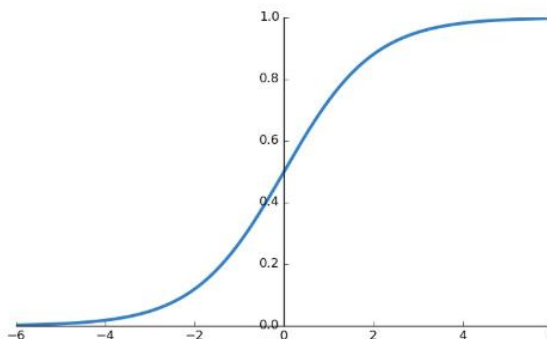


Figure 17 Sigmoid Function

The results of the activation functions used in the experiment section of this thesis will be discussed.

3.3. OPTIMIZATION FUNCTIONS

The general objectives of optimization functions are to minimize loss function in CNN models and update with new weight changes from existing samples [106]. When I explain it in detail, the CNN weights are started at a certain value, and the output layer creates a prediction system according to the calculations that occur in hidden layers. This output is compared to the value expected from a Loss function. For calculating of gradient formed in Output, an optimization function use used [106] [107] [108].

The main optimization functions used in this thesis are detailed below Table 1.

Table 1 Used Optimization Functions

Optimization Function	Description	Formula
Adaptive Moment Estimation (Adam)	<p>“Adam” is a more efficient, adaptive optimization algorithm that can be used instead of the classical stochastic gradient reduction method. That is, it dynamically updates the learning rate for each parameter. It is efficient as a computing load and needs low memory requirements [108].</p>	$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\frac{v_t}{1 - \beta_2^t} + \epsilon}} \frac{m_t}{1 - \beta_1^t}$
Adadelta	<p>In Adadelta optimization method, the parameters have their own learning speeds and these learning speeds are gradually decreasing and the system cannot perform learning after a certain point [109].</p>	$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$
Stochastic Gradient Descent (SGD)	<p>In SGD algorithm, calculation is made on a training sample instead of all training data. Thus, problems of memory deficiency that may occur are also prevented [110].</p>	$\theta_{t+1} = \theta_t - \eta \cdot \nabla_{\theta} J(\theta_t; x^{(i)}; y^{(i)})$
Adamax	<p>Adamax, a variant of the Adam optimization function, scales the vt factor in the Adam update rule, the slope is inversely proportional to the l2 norm and the current slope $\frac{ g_t }{\sqrt{ g_t ^2}}$ [108].</p>	$\theta_{t+1} = \theta_t - \frac{\eta}{(1 - \beta_1^t)} \frac{m_t}{u_t}$

3.4. PERFORMANCE EVALUATION METRIC

Some performance criteria are needed to evaluate overfitting and underfitting in addition to measuring the accuracy of the CNN Model [111] [112] [113]. If the selected model has worked more than necessary on data set used for training and started to memorize, or if the training set is uniform, it is called overfitting in the literature [43]. Unlike overfitting, if a model has insufficient learning, it means that the model does not comply with the training data and therefore miss the trends in the data and is underfitting [114]. Measuring a single performance metric for multi-classification CNN models may not be sufficient to evaluate the model. For this reason, some measurement parameters are needed to obtain a more reliable result from the model. In order to get an accurate result in this verification metric, our own created data set is divided into train, test, validation [115] [116]. The test and train data have not been made the same since the CNN Model has not been used to test the model and the model used for testing.

In this thesis, the distribution on this data set is 80% train, 10% validation, 10% test. This division rate also affects the performance of the model. While creating this distribution, the distribution of ship pictures in each category is randomly realized. We will explain the details of our dataset that we have created in the following chapters.



Figure 18 Distribution on the data set

The train data set is used to train the model [3]. The validation part used here is used to give a better idea of how well the model will perform on invisible data during the train [117]. The name of this method is referred to as “cross validation” in the literature [118]. This method is one of the most preferred methods to determine the performance rate of the model [12]. The test part is used to measure its performance when the model completes training [117]. It has been observed that the Accuracy value is not enough to accurately measure the performance of the model. Therefore, the following performance metrics are needed. Detailed descriptions of these metrics are given below.

3.4.1. Confusion Matrix

Confusion Matrix is a summary of the estimation results for the classification problem, that is, the number of correct and false estimates are summarized by count values and separated by each class, and also shows when your classification model is mixed in the estimates [115]. As a result of the confusion matrix, accuracy, recall, precision, F1-score evaluation outputs are obtained by defining True positive (TP) as objects that are truly perceived and object is negative, and is predicted to be negative(TN), objects that are not detected as false negative (FN) and objects that are false perceived (FP) [119]. The calculation (1,2,3,4) equations of these outputs will be specified below. these metrics will be used in other parts of the thesis to compare the performances of ship classification models.

3.4.2. Accuracy

The accuracy value is calculated by the ratio of the areas correctly predicted in the model to the total data set.

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

Equation 6 Accuracy

3.4.3. Precision

It shows how many of the values estimated as Positive are really Positive.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Equation 7 Precision

3.4.4. Recall

It is a metric that shows how much of the transactions that should be estimated as positive are estimated as Positive.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Equation 8 Recall

3.4.5. F1-Score

It shows the harmonic average of Precision and Recall values.

$$\text{F1 Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Equation 9 F-1 Score

3.4.6. Mean Average Precision (mAP)

Each image in an object detection problem can have different objects from different classes. As mentioned earlier, both classification and localization of a model need to be evaluated [120]. Therefore, the standard sensitivity metric used in image classification problems cannot be applied directly here. In this case, mAP (Minimum Average Precision) is used [121]. To calculate the mAP, we need to calculate the average precision (AP) above mentioned precision value for each class [122]. Formula of mAP is shown below equation 10 and we compute the AP for each class and average them.

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^N \text{AP}_i$$

Equation 10 Formula of mAP

CHAPTER IV

EXPERIMENTAL STUDIES FOR SHIP CLASSIFICATION AND DETECTION

In the first part of this chapter, hardware and software information which are used in our experiments are explained detailly. In the second part, steps of preparation of Data set are mentioned. Our own data set has six ships' classes like "Container, Cruise, Fishing, Military, Tanker, Yacht" as shown below Table 3. In all experiments, this data set distribution in all ship classes is 80% train, 10% validation, 10% test rate. After these parts, work flow chart is detailed.

Last part of this section, Ship classification and detection experiments are explained. Firstly, ship classification experiments are given. They are detailed as "Methodology" and "Result and Analysis". In the second part, ship detection experiment is mentioned as "Methodology" and "Result and Analysis".

In the experiments "Methodology" sections, how we perform the experiments and which methods are preferred, are explained.

In the experiments' "Result and Analysis" sections, results of our studies are analyzed. When looking at the performance reviews of deep learning techniques for machine learning, object classification and detection according to literature research, it is recommended to use CNNs in solving such problems. For this reason, in this study, it is observed that open source architectures such as VGG16, VGG19, DenseNet121, Xception for ship classification performed higher than our own data set. Besides that, for ship detection, YOLOv3 architecture is recommended. In addition, transfer learning techniques have been used in pre-trained models to deal with the problem of data deficiency and less computation time in architectures.

4.1. HARDWARE AND SOFTWARE

The hardware and software infrastructure used in this thesis has been specified in Table 2 with its versions. Current versions have been used. The reason for the use of the Ubuntu operating system is that the development environment and libraries are compatible with the used systems. Using a powerful graphic card has shortened the training period of the models.

Table 2 Hardware and Software Specifications

Hardware and Software Specifications	
Operating System	Ubuntu 18.04 LTS
Graphics Card	GeForce RTX 2080 Ti
Software	Python version 3.7.7
Framework	Keras 2.3.1
IDE	Anaconda 4.2.0 Jupyter 1.0.0
Using Library	Sciki-Learn 0.22.1 Numpy 1.18.1 Matplotlib 3.1.3

4.2. DATASET PREPARATION

Below Data Set in Table 3 is created for ship classification by taking advantage of crawler in python language. Websites which are “<https://www.vesselfinder.com>” and “<https://www.marinetraffic.com/>” are used for ship images. For insufficient data sets, more images are downloaded from search engines such as Google, Yandex, Yahoo etc. After download processes, some noisy images are deleted. because the more accurate picture data, the higher the efficiency of CNN models. With these data collection methods, 54030 ship pictures are collected for a total of six ship classes. These classes are “Container, Cruise, Fishing, Military, Tanker, Yacht”. The distribution of the classes in our own data Set is shown in the graphics in Figure 19. Randomly selected sample pictures in the Data Set are shown below Figure 20.

The example Python crawler code is given as below Figure18 for class “Container”. Also, in the table below, numbers are given as "train, test, validation" for each class. This separation is divided into 80%, 10% and 10% respectively.

The reason for this distinction is allocated to train the model overfitting (memorization), underfitting (lack of data) and performance metric analysis. These metrics of the later stages of the thesis will be mentioned.

```
import request

from bs4 import BeautifulSoup

import urllib.request

for pageNumber in range (1, 10000):

    url = "https://vesselfinder.com/vessels?page="+str(pageNumber)+"&type=401"

    requestFirst = urllib.request.Request(url, headers={'User-Agent' : 'Mozilla/5.0
(Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/41.0.2228.0 Safari/537.36'})

    htmlFirst = urllib.request.urlopen(requestFirst)

    soup = BeautifulSoup(htmlFirst, "html.parser")

    for shipName in soup.find_all("a", attrs={"class": "ship-link"}):

        ship_page = "https://www.vesselfinder.com"+shipName.get("href")

        requestSecond = urllib.request.Request(ship_page, headers= {'User-Agent' :
'Mozilla/5.0 (Window NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/41.0.2228.0 Safari/537.36'})

        htmlSecond = urllib.request.urlopen(requestSecond)

        beautifulSoup = BeautifulSoup(htmlSecond, "html.parser")

        for shipPicture in beautifulSoup.find_all("img", attrs={"class": "main-photo"}):

            print(shipName.text.strip()[1:-1]+"jpg")

            urllib.request.urlretrieve(shipPicture["src"],
            "~/Ship_Dataset/Container/" + shipName.text.strip())
```

Figure 19 The example of Python crawler code

Table 3 Train/Validation/Test Data Set of Ship Images

Total Ship Data Set separated %80 Train, %10 Validation, %10 Test	
Train Set of 43224 Images	
Cruise	5640
Container	5664
Tanker	8816
Military	4400
Fishing	8696
Yacht	10008
Validation Set of 5403 Images	
Cruise	705
Container	708
Tanker	1102
Military	550
Fishing	1087
Yacht	1251
Test Set of 5403 Images	
Cruise	705
Container	708
Tanker	1102
Military	550
Fishing	1087
Yacht	1251

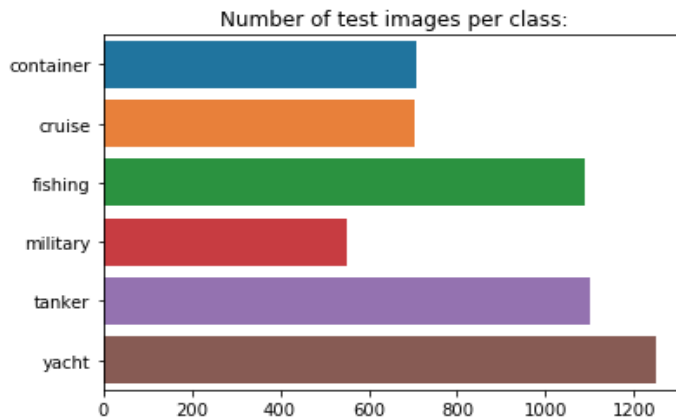
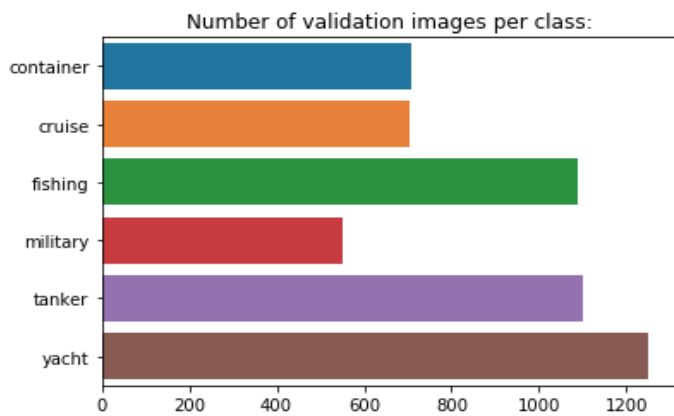
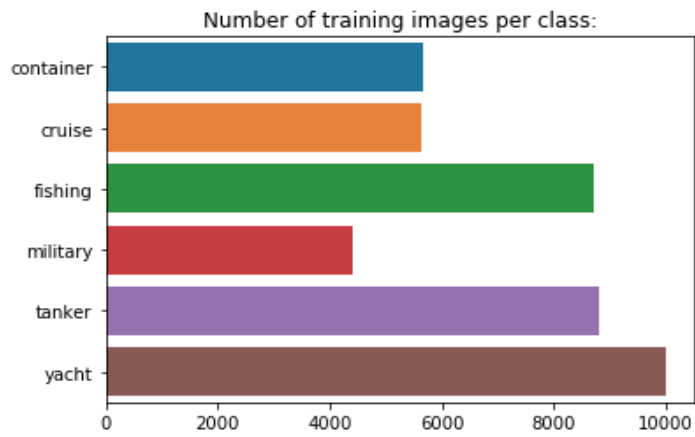


Figure 20 The distribution of the classes in our own Data Set

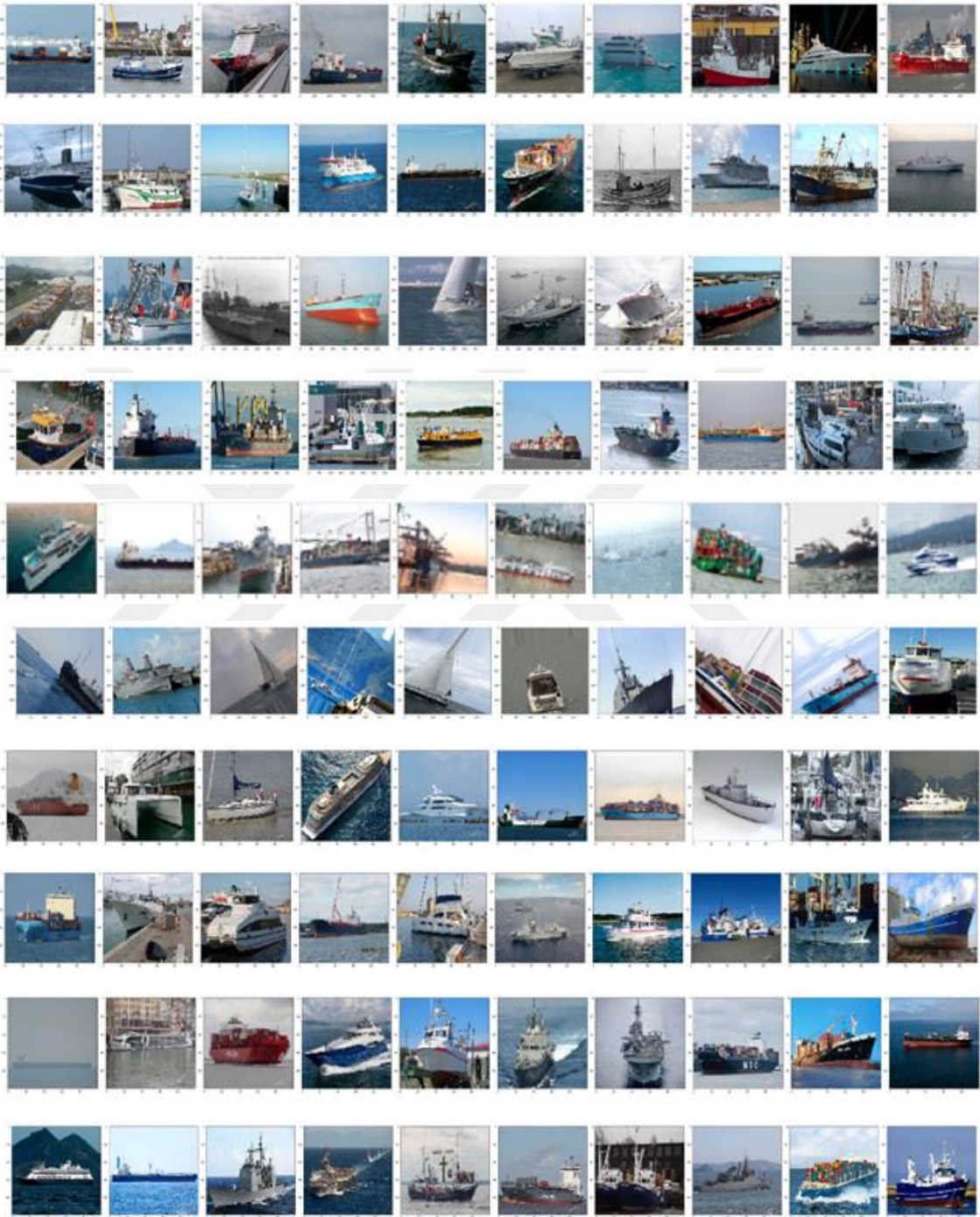


Figure 21 Randomly selected sample pictures

4.3. WORKING FLOWCHART

Basically, the progress steps of the experiments done on the models are specified in the work flowchart as shown ANNEX I. Our experiments usually have 3 main stages of progress. These are model selection, optimization, and evaluation stages.

4.4. SHIP CLASSIFICATION AND DETECTION TASKS

In this section, our main goal is to classify and detect for a particular ship image among six ship classes and different hyper parameters have been tried to increase efficiency in these experiments.

Firstly, Ship classification performance for the modified models VGG16, VGG19, DenseNet121, Xception retrained on our own dataset. These experiments are “Pre-trained models performance comparison”, “Effects of layers on modified models with Transfer Learning”, “Optimization with Hyperparameters”.

Secondly, Ship detection performance for modified model YOLOv3 experience, which are trained on different numbers of labelled data sets are mentioned.

In our experiment environment, TensorFlow and Keras frameworks are used to build, train, evaluate and predict our models. Anaconda and Jupyter are used as environment to run and compile these algorithms. As the software language, high level language Python is preferred.

In general, setups have been made to design algorithms, process data, draw shapes and evaluate the results and use the relevant libraries. The libraries used are NumPy, Pandas, Scikit-learn and Matplot, respectively. The data set is prepared with manual examination to ensure that the data is properly distributed throughout the data set and handle the variations of the data specified which are mentioned in Section “Data Set Preparation”. Some evaluation methods to get higher efficiency for our models during the experimental processes are used. These evaluation methods are multi-class confusion matrix, accuracy, recall, precision, F-1 score, mAP, which are explained detail in chapter 3. In the following sections, experiments for ship classification and detection are described separately.

4.4.1. Ship Classification Experimental Methodology, Result and Analysis

4.4.1.1. Experiment 1 Pre-trained models performance comparison

In this experiment performances of modified pretrained models on our own ship data set are compared and given in below Table 5.

4.4.1.1.1. Methodology

Four models which are “VGG16”, “VGG19”, “DenseNet121”, “Xception” and trained on ImageNet are selected, each with a different model architecture, accuracy, and performance. One of factors in choosing these models is the performance comparison of the models which are trained with ImageNet. Performances’ comparison is shown below Table 4. The architectural structure of the pretrained models which are used in these experiments were explained in detail in Chapter 2.

Brief information about the models is mentioned below. VGG16 has shallower architecture than other CNN networks. Despite this shallow architecture, it shows high performance. VGG19 has a similar characteristic to VGG16. VGG19 has three extra layers according to VGG16. Xception is an extension of Inception architecture which replaces standard Inception Modules with deeply separable folds.

Deeply Separable Convolutions are an alternative to conventional convolutions that are considered to be much more efficient in terms of computation time. DenseNet121 requires fewer parameters than conventional CNN according to connection shapes. Because there is no need to learn extra feature maps.

Table 4 Performances' comparison

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	88MB	0.790	0.945	22,910,480	126
VGG16	528 MB	0.712	0.901	138,357,544	23
VGG19	549 MB	0.713	0.900	143,667,240	26
DenseNet121	33 MB	0.750	0.923	8,062,504	121

All the models which are mentioned above are pre-trained in the ImageNet dataset and are used with the Tensorflow and Keras API. Information about ImageNet is given in chapter 2 in detail. The main reason for choosing the ImageNet data set is that there are too many different ship classes and sub-categories of images.

4.4.1.1.2. Result and Analysis

In this part, performances of modified pretrained models on our own ship data set are displayed. All the architectures used in the thesis are adapted to our own dataset. As a result, the best performance comparisons given in Table 5 are shown. In these performance evaluations, the number of classes is the same in every experiment as “Military, Container, Cruise, Fishing Ship, Tanker”. In addition, for each modified model, trainings are carried out the rate of "train, validation and test" as explained in Data Preparation section. The test data used for the evaluations here are the ship class data that the model has not seen before. ImageNet dataset is used for pretraining of modified models, and the data set used for ship classification is used on these models. The results obtained in the evaluation of the models are consistent with the estimates of the ship images shown to manually modified models.

Table 5 Ship classification performance for the modified models retrained on our dataset using transfer learning

Modified Model	Accuracy	Precision	Recall	F1-Score
VGG16 + [ImageNet] + [Own Dataset]	0.91	0.91	0.91	0.91
VGG19 + [ImageNet] + [Own Dataset]	0.86	0.87	0.87	0.87
Xception + [ImageNet] + [Own Dataset]	0.84	0.84	0.81	0.82
DenseNet121 + [ImageNet] + [Own Dataset]	0.79	0.82	0.80	0.80

Methods of obtaining the best performance, how many layers are added on the pretrained model and which hyper parameters are used, will be explained in detail in other experimental sections. This section compares the best performance outputs with our ship's dataset. As a result, as stated in ANNEX II, when Accuracy, Precision, Recall, F1-Score performance metrics are examined on our own data set, VGG16 model shows the best performance with 91 percent performance output. The worst performance is the modified DenseNet121. In the results of this experiment, overfitting is not observed since I obtain results close to Accuracy ratio Precision, Recall, F1-Score.

Confusion matrixes indicate an indication of misclassified images per classes, as noted in the figures above. As shown in the figures, when comparing between modified CNN models, the wrong classification rate is the lowest VGG16, whereas the highest misclassification is seen as DenseNet121. ANNEX III, ANNEX IV, ANNEX V, ANNEX VI show evaluation prediction for VGG16, VGG19, Xception and DenseNet121 modified models.

The models we modified make false predictions in some images, as seen in their estimates. To increase this accuracy even more, it is necessary to increase the number of data.

4.4.1.2. Experiment 2 Effects of layers on modified models with Transfer Learning

In this experiment, according to the transfer learning technique, the effect of the last layer removed from the pre-trained models and the number of layers added in our training again with our own data set has been observed.

4.4.1.2.1. Methodology

Weight layers of pre-trained models affects the reduction of training times and reduction of error rates with the transfer learning technique mentioned in Chapter 2 [2]. For these reasons, this transfer learning technique against the problems of data inefficacy and long training period in the experiments is preferred to use in this thesis. In our experiments, the full connected layer which is the last layer of pre-trained models is removed with using transfer learning. Because ImageNet the model trained on ImageNet creates an output which has 1000 classes. So, for training with our own data set, the last Dense layer as six classes is changed. For example, our VGG16 pre-trained model consists of 5 block with 13 curved layers, a flatten, two full connected layers and a layer of layers.

The process of adapting the new model to transfer learning technique consists of two stages. At the first stage, the full connected and Dense layers are removed from the pre-trained model, and all layers except for these layers are not processed. In the second stage, instead of these discarded layers, our own layers are added and adapted to our own data set with trying the different hyper parameters. In our experiments, expressions like " f_n " indicate the number of layers added to the full connected layer in our new model. The number "n" increases numerically as much as the number of layers added.

4.4.1.2.2. Result and Analysis

The purpose of the experiments in this section is to observe the effects of the number of layers we use for pre-trained models on performance when we use transfer learning technique. In Methodology part, we explain how we used the transfer learning technique. Also, we mention that the number of neurons in the layers where we apply transfer learning in models is stated as " f_n " for each layer.

Below ANNEX VII gives the neuron numbers and accuracy rates used in the layers for VGG16, VGG19, Xception, DenseNet121. The same data set and classes are used in this experiment as in other ship classification experiments.

As it can be seen in ANNEX VII, in our experiments for VGG16 and VGG19, it has been observed that increasing neuron number and layer number increases performance, and exact opposite for Xception and DenseNet121, decreasing accuracy rates. The best result is a modified VGG16 with a number of 3 layers of 256, 128, 64 neurons, while the worst result is a modified DenseNet121 with a number of 3 layers of 512, 256, 128.

4.4.1.3. Experiment 3 Optimization with Hyperparameters

In this experiment, effect of some hyper parameters are investigated on our modified CNN models.

4.4.1.3.1. Methodology

Some basic hyper parameters are used for achievement of better performance and optimization than our CNN models. One of the biggest challenges of CNN models is to achieve an optimal combination of hyper parameters to the model. The hyper parameters used in our experiments are indicated in the table below with their values and are also described in detail in chapter III.

Table 6 Optimization Hyperparameters

Variables	Values
Batch size	32
Learning rate	0.01, 0.001, 0.0001
Fully Connected Layer Activation Function	ReLu
Dense Layer Activation Function	Softmax, Sigmoid
Drop out	0.2, 0.25, 0.5
Optimizer	SGD, Adam, Adadelata, Adamax
Input Picture Size	(200,200), (300,300), (400, 400), (600,600)

Hyper parameters based on our literature reviews are determined and some hyper parameters are used as default during the experiments. One of the hyper parameters used as the default is batch size. Batch size is used as constant value 32. Our model training is stopped at 30 epochs when it is exposed to overfitting. During the experiments, activation function which is used in full connected layer is ReLu. Softmax and Sigmoid activation functions are used in Dense layer. SGD, Adam, Adadelata, Adamax are used as optimizers on our modified models. Besides this, values between 0.01, 0.001, 0.0001 is used as learning rate. Evaluations on modified models related to different input pic sizes are made.

4.4.1.3.2. Result and Analysis

The hyper parameters mentioned in Chapter 3 directly contribute to the performance and optimization of models. In this thesis, different hyper parameters are used for modified models and different results are obtained. For selecting of hyperparameters, Random Search method is used. And for each selected hyperparameters, before starting of experiments, testing in short term model trainings is performed with 10 epochs. Then, number of epochs is increased according to the appropriate hyper parameter selections. With this method, we aim to achieve the most appropriate hyper parameter selection for our models in a shorter time. During the trainings, the training character of the model according to the train and

validation data set outputs in each epoch is determined and the model training is ended when the overfitting problem started to occur.

In all of our model trainings, batch size is 32 and activation function is Relu in full connected layer. Also, in most of our experiments, in Dense layer, activation function Softmax is preferred. Detailed descriptions of the hyper parameters used in this section can be found in Chapter 3. ANNEX VIII shows the hyper parameters which are used in the experiments and their outputs.

In our Hyper Parameter adaptation experiments, different values are used for each modified model. In these experimental processes, the three best performing experiments among each modified model are shown in ANNEX VIII. Among our modified VGG16 model hyperparameter experiments, the model which is named “VGG16 + [ImageNet] + [Own Dataset]-1” has the highest performance rate. In our first experiments on this model, when “SGD” is used as an optimizer, we observe that the training time is very long compared to “Adam”. Also, "Adam" gives better results in terms of performance than other optimizers. When the image size is increased for the input picture size, we observe that the model performance ratio does not change after a certain size and we determine the optimum size as 600 x 600. During our experiments, the learning rate value is applied in the range of 0.1 and 0.00001 and observe that the optimal rate is 0.001. The reduction made in the drop out value directly contributes to the performance of our model and the optimum is determined as 0.2.

Among our modified VGG19 model hyperparameter experiments, the model which is named “VGG19 + [ImageNet] + [Own Dataset]-1” has the highest performance rate. In this model, Adadelta is observed as the best optimizer compared to experiments on VGG16. Increasing the input picture size does not contribute to the performance of the model in any way. The optimum value of the learning rate ratio for the model is observed as 0.01.

Among our modified Xception model hyperparameter experiments, the model which is named “Xception+ [ImageNet] + [Own Dataset]-1” has the highest performance rate. Increasing the input picture size does not contribute to the performance of the model in any way. Adadelta is observed as the best optimizer according to SGD. Increasing the drop out value contributes positively to our model's performance. The best performing Learning rate on the model is 0.1.

Among our modified DenseNet121 model hyperparameter experiments, the model which is named “DenseNet121 + [ImageNet] + [Own Dataset]-1” has the highest performance rate. It has been observed that SGD optimizer has a direct contribution to the performance of this model compared to other model experiments. Learning rate shows the best performance at 0.01 and drop out at 0.5. Changing the input picture size values does not directly affect the performance of the model and the optimum value found is observed as 200x200.

According to the results of the experiment 1, 2, 3, the CNN architecture VGG16 best suited to our own data set is observed as the worst performing DenseNet121. The evaluations of the models will be explained in detail in the last Chapter.

4.4.2. Ship Detection Experiments’ Methodology, Result and Analysis

In our second experiment set, our experiments on ship detection are explained. Our main purpose in this experiment is to detect the ship in an image and find out which ship subcategory it belongs to. The following sections describe our approach to the ship detection problem and what methods used for the solution in our experiments. The analysis of these experimental results will be explained in the next chapter. Ship detection experiments have six ship classes like as ship classification.

The LabelImg [3] tool shown in the figure below has been used to create labeled ship images. This tool allows us to limit the position information of the ship in a picture in a rectangular shape. The reason for this data labelling process is to train the data labeled on YOLOv3. This labelling process is carried out for each class totally. I labelled approximately 1000 images for each ship classes and totally 6000 ship labelled images.

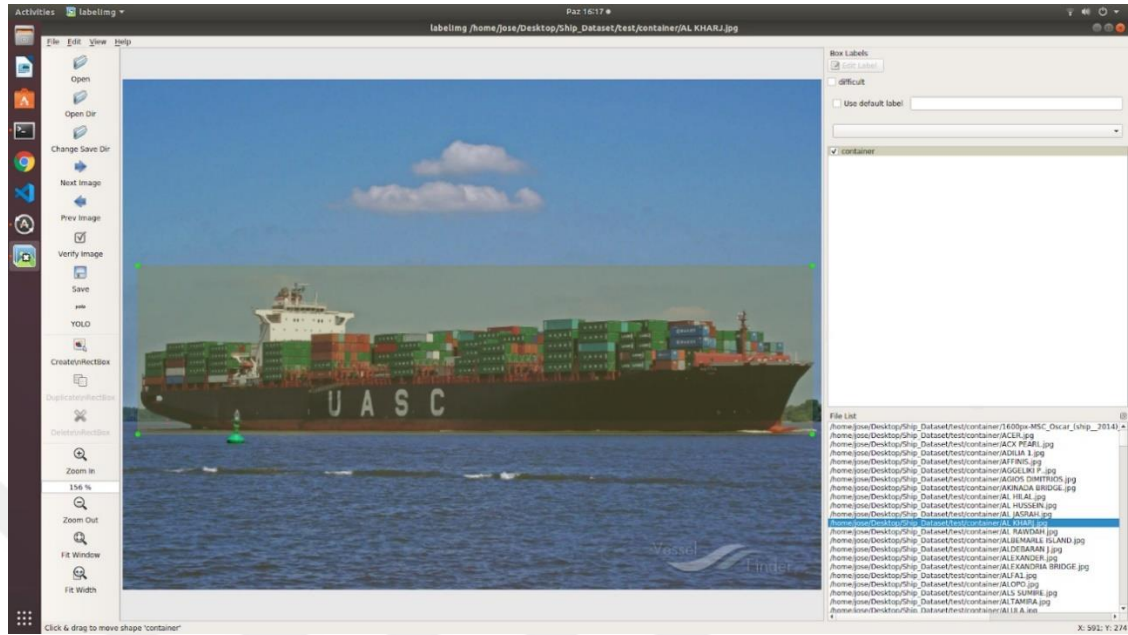


Figure 22 Sample ship labeling image using LabelImg Tool

The ship images labeled in the LabelImg tool shown in Figure-5000 produce a file with a .txt extension as output, and an output is obtained as format “<object-class-id> <x-centre> <y-centre> <width> <height>” from this file.

The example of the output format is given in Figure 22 below. To explain this output format: <object-class-id> represents class names as integer starting from 0. <x-centre> <y-centre> <width> <height> represents float values in the range of 0.0 and 1.0, and the screen specifies the coordinates of the area to be labeled.

```
1 0.716797 0.395833 0.216406 0.147222
0 0.687109 0.379167 0.255469 0.158333
2 0.420312 0.395833 0.140625 0.166667
```

Figure 23 LabelImg Tool output format

4.4.2.1. Experiment 4 Number of Labeled Pictures Analysis

The purpose of this experiment; for ship detection, the effects of the labelled ship images numbers are observed.

4.4.2.1.1. Methodology

YOLOv3 is a high-tech object detection system and in experience I have made some trials by adapting it to our own dataset [5]. Detailed explanations about YOLOv3 are mentioned in chapter 2. In this thesis, I use the pre-trained Darknet-53 classifier which is the part of YOLOv3 architecture in the system which is modified by us for ship detection. In addition, this classifier is trained on the ImageNet dataset like models used in ship classification. As seen in figure 24 below, the architecture of YOLOv3 and DarkNet is specified. The working structure of the architecture; darknet which is the first part of YOLOv3 extracts the properties in the input image and sends the second part. The second part analyzes the output of Darknet-53 and classifies what kind of object is inside the boxes after creating the bounding boxes in the picture.

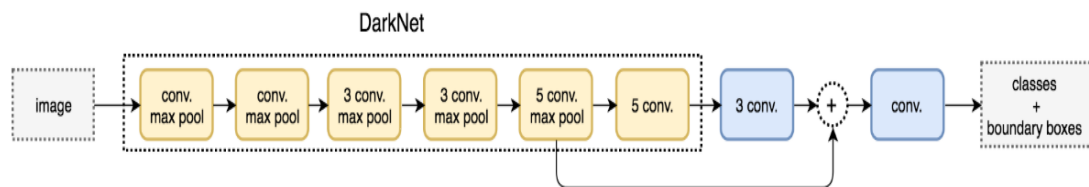


Figure 24 YOLOv3's architecture to work with Darknet [4]

There are two stages in our experiment. In the first stage, our ship detection system, which consists of 6 classes, is trained on a dataset consisting of 3000 labeled images. Then this number is increased to 6000. These labeled ship data sets are divided into 80% train and 20% test data. Our purpose in these experiments is to investigate the effects of YOLOv3 on the number of data sets. The hyper parameters used during all experiments are batch size 64, subdivisions 16, learning rate 0.001, and input pic size 416x416. The trainings are completed in 12000 iterations in total and mAP values are measured together with the test data.

4.4.2.1.2. Result and Analysis

The purpose of this experiment is to evaluate its performance against the number of data labeled in the YOLOv3 object detection system on the 6 class ship our own data set. In

the first experience, 500 labeled data sets are created from each ship's classes as “Military, Container, Cruise, Fishing Ship, Tanker” and trained on the YOLOv3 system. In the second experience, 1000 labeled datasets are created and trained on YOLOv3 for each ship class.

In the table below, mean average precision (mAP) ratios of this number of labeled data are given. Mean Average Precision is explained in chapter 2 in detail. When the results are examined, as the number of labeled data increases, its performance on the model increases. The test results on our ship detection system are given in ANNEX IX, ANNEX X. The trained model does not see the ship data in these tests. Experimental infrastructure and hyper parameters used are described in chapter IV.

Table 7 Comparison Number of Labeling Ship Image performance

Number of Labeling Ship Image	Mean Average Precision(mAP)
3000	0.58
6000	0.63

4.5. COMPARISON RESULTS WITH ANOTHER SIMILAR THESIS

The closest study to our thesis is seen as the thesis called “Fine-tuning Convolutional Neural Networks For Maritime Vessel Classification, Verification And Recognition” written by C. Deniz Gürkaynak [123]. In their experiments, AlexNet, VGGNet and ResNet, which are three convolutional neural networks, used architectures. The MARVEL data set is used for training and testing [34]. In addition, they have included transfer learning technology in their theses, just like us. With this study in 2018, more than 2 million ship images were studied and they achieved the best performance on VGG 16 architecture. They achieved Accuracy 93.67% and F1 Score 93.58% as the best performance rates.

We achieve that VGG 16 architecture gives the best performance on 54030 ship images which are created by ourselves. The best performance rates are Accuracy 91% and F1 Score 91%. With the same architecture but a different and small data set, it is seen that results close to the results of Gürkaynak were obtained.

CONCLUSION AND FUTURE WORK

While technology is developing rapidly, state-of-the-art technologies such as artificial intelligence, machine learning and deep learning are beginning to be a solution to real problems in today's world. Among these solutions, CNN models with deep learning technique come to the fore against object classification and detection problems.

Ship detection and classification is performed with using CNN models. This thesis focusses on maritime problems, and we have achieved some results. If we list these results, first and foremost, the pre-trained CNN models have better performances with the transfer learning method against small data set problems. By means of this, we saved time by using training models instead of training from scratch the models with our own data set. Also, high performance models are created. We have collected all the ship images which are used in this thesis. It took me a lot of time to organize, clean, and adapt this data to our experiments.

Choosing random search which are the most appropriate hyperparameter finding method for training of these CNN models enabled me to progress faster in this thesis. The main purpose in applying these hyper parameter methods is to deal with a small data set. Because the basic working principle of deep learning architectures is that they work based on a lot of data. In this thesis, we show that deep learning architectures can give a good result on a small data set by getting good performance on a small data set.

For ship classification, CNN Models VGG16, VGG19, Xception and DenseNet121 which are trained in the ImageNet dataset are used. Among these models, VGG16 model gave the best performance according to validation and test data with an accuracy rate of 91%. Looking at the best optimizer training time and performance, Adam comes to the fore. The number of used layers for transfer learning differs for each model. For example, when we increased the number of neurons in the fully connected layers for VGG16, it was observed that we got better results, while we experienced the opposite in VGG19. According to the experimental results, Softmax gave the best result as the activation function used in the Dense layer in the solution of multi-class ship classifier problems.

As a result, the ability to perform transfer learning integration, which is a deep learning technique on a small data set and to produce high accuracy models, shows that CNN models give good results in ship classification problems. Using this technique, ship classification models can be realized for different types of ships. In this experiment, we observed that the more we increase the number of labeled data, the more the model improves in performance. Also, we have seen from the results that the YOLOv3 architecture can be adapted to our ship classification problem. As a result, increasing the number of labeled data from 3000 to 6000 increased the mAP value of the model performance by 5%.

There are many fields that we cannot experience within this thesis. If we list them; For ship classification and detection, we plan to classify and detect ships with different ship classes in an image. In addition, it will be interesting to classify and detect ships with mask segmentation of ships in the images with the CNN technique called Mask R-CNN.

Object classification architectures such as VGG16, VGG19, Xception and DenseNet121 can be expanded by retraining on relevant datasets for non-ship vehicles and land vehicles. Finally, apart from object detection systems such as YOLOv3, it would be useful to compare the performance on object detection architectures such as SSD, Faster R-CNN and RetinaNet on our own data set.

REFERENCES

- [1] **Hoegh-Guldberg, (2015)**, "Reviving the Ocean Economy: the case for action - 2015," *WWF Report 2015*, vol. 1, no. Report 2015, pp. 5-18.
- [2] **D. H. H. a. T. N. Wiesel, (1959)**, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of physiology*, vol. 148, no. 3, pp. 574-591.
- [3] **A. Kloss, (2015)**, "Object Detection Using Deep Learning-Learning where to search using visual attention," Eberhard Karls Universität Tübingen, Tübingen.
- [4] **D. Lowe, (1999)**, "Object recognition from local scale-invariant features," *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, pp. 1150-1157.
- [5] **N. A. T. B. Dalal, (2005)**, "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference*.
- [6] **A. RamPathak, M. Pandey and S. Rautaray, (2018)**, "Application of Deep Learning for Object Detection," *Science Direct*, no. 132, pp. 1706-1717.
- [7] **A. Krizhevsky, I. Sutskever and G. Hinton, (2012)**, "Imagenet Classification with Deep Convolutional Neural Networks," *In Advances in Neural Information Processing Systems*, pp. 1097-1105.
- [8] **J. Tournadre, (2014)**, "Anthropogenic pressure on the open ocean: The growth of ship traffic revealed by altimeter data analysis," *Geophysical Research Letters*, vol. 41, no. 22, pp. 7924-7932.
- [9] "Marine Traffic," Marine Traffic, [Online]. Available: <https://help.marinetraffic.com/hc/en-us/articles/204062548-Live-Map>. [Accessed 10 06 2020].

- [10] **A. Tabasco and M. Gerard, (2011),** "Ship Classification in Single-Pol SAR Images Based on Fuzzy Logic," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 3129-3138.
- [11] **E. Akyilmaz, C. Demirkesen, F. Nar, E. Okman and M. Çetin, (2013),** "Interactive Ship Segmentation in SAR Images," Haspolat, Turkey.
- [12] **U. Kanjir, H. Greidanus and K. Ostir, (2018),** "Vessel detection and classification from spaceborne optical images: A literature survey," vol. 207, no. 7.
- [13] **H. LI, L. Chen, F. Li and M. Huang, (2019),** "Ship detection and tracking method for satellite video based on multiscale saliency and surrounding contrast analysis," vol. 13, no. 2.
- [14] **R. S, P. N, K. R and R. S, (2019),** "Real Time Detection And Segmentation Of Ships In Satellite Images," vol. 8, no. 12.
- [15] **R. Zhanga, J. Yao, K. Zhang, C. Feng and J. Zhang, (2016),** "S-CNN-BASED SHIP DETECTION FROM HIGH-RESOLUTION REMOTE SENSING," *Remote Sensing and Spatial Information Sciences*, vol. XLI, no. 7.
- [16] **K. Simontan and A. Zisserman, (2014),** "Very Deep Convolutional Networks for Large-Scale Image Recognition," *Computer Science*, no. 6.
- [17] **G. Huang, Z. Lui, L. Maaten and K. Weinberger, (2016),** "Densely Connected Convolutional Networks," *Computer Science*, vol. 1, no. 2, p. 3.
- [18] **F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions, (2017),** " in *IEEE Conference on Computer Vision and Pattern Recognition*.
- [19] **J. Redmon and A. Farhadi,** "arXiv.org," Cornell University, 8 Apr 2018. [Online]. Available: arXiv:1804.02767.
- [20] S. R. Department, "www.statista.com," Statista Research Department, 23 03 20. [Online]. Available: <https://www.statista.com/statistics/264024/number-of-merchant-ships-worldwide-by-type/>. [Accessed 06 2020].

- [21] UNCTAD, "<https://stats.unctad.org/>," UNCTAD, 2019. [Online]. Available: <https://stats.unctad.org/handbook/MaritimeTransport/MerchantFleet.html>. [Accessed 1 05 2020].
- [22] **E. Statistics, (2015)**, "The world merchant fleet - statistics from Equasis," EMSA.
- [23] **Y. Wang, C. Wang, H. Zhang, Y. Dong and S. Wei, (2019)**, "A SAR Dataset of Ship Detection for Deep Learning," *Remote Sensing*, vol. 11, no. 765, pp. 6-11.
- [24] **M. Ma, J. Chen, W. Liu and W. Yang, (2018)**, "Ship Classification and Detection Based on CNN Using GF-3 SAR Images," *Remote Sensing*, vol. 2043, no. 10, pp. 2-22.
- [25] **M. Ma, J. Chen, W. Liu and W. Yang, (2018)**, "Ship Classification and Detection Based on CNN Using GF-3 SAR Images," *Remote Sensing*, vol. 10, no. 2043, pp. 3-10.
- [26] **H. Li, L. Chen, F. Li and M. Huang, (2019)**, "Ship detection and tracking method for satellite video based on multiscale saliency and surrounding contrast analysis," *Remote Sensing*, vol. 13, no. 2, pp. 6-10.
- [27] **J. Gallego, A. Pertusa and P. Gill, (2018)**, "Automatic Ship Classification from Optical Aerial Images with Convolutional Neural Networks" *Remote Sensing*, vol. 10, no. 4, p. 511.
- [28] **S. Alashhab, A.-J. Gallego, A. Pertusa and P. Gil, (2019)**, "Precise Ship Location With CNN Filter Selection From Optical Aerial Images," *IEEE Access*, vol. 7, pp. 96568-96570.
- [29] **T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, (2018)**, "Focal Loss for Dense Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [30] **S. Ren, K. He, R. Girshick and J. Sun, (2015)**, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Neural Information Processing Systems (NIPS)*.

- [31] **K. Redmon and A. Farhadi, (2017),** "YOLO9000: Better, Faster, Stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu.
- [32] **J. Redmon and A. Farhadi, (2018),** "YOLOv3: An Incremental Improvement," *University of Washington*.
- [33] **E. Gündoğdu, B. Solmaz, A. Koç, V. Yücesoy and A. Alatan, (2017),** "Deep distance metric learning for maritime vessel identification, " in *2017 25th Signal Processing and Communications Applications Conference, SIU 2017, Antalya*.
- [34] **E. Gundogdu, B. Solmaz, V. Yücesoy and A. Koç, (2017),** "MARVEL: A Large-Scale Image Dataset for Maritime Vessels," in *Asian Conference on Computer Vision*, Ankara.
- [35] **D. Hashimoto, G. Rosman, D. Rus and O. Meireles, (2018),** "Artificial Intelligence in Surgery: Promises and Perils," *PMC*, vol. 1, no. 268, pp. 70-76.
- [36] "<https://www.ibm.com/>," IBM, 5 12 2019. [Online]. Available: <https://www.ibm.com/blogs/systems/ai-machine-learning-and-deep-learning-whats-the-difference/>. [Accessed 05 2020].
- [37] **S. Nordin, (2018),** "A Deep Learning Prediction Model for Object Classification," Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of, DELFT.
- [38] **R. Carbonneau, K. Laframboise and R. Vahidoc, (2008),** "Application of machine learning techniques for supply chain demand forecasting," *European Journal of Operational Research*, vol. 3, no. 184, pp. 1140-1154.
- [39] **I. Lenz, H. Lee and A. Saxena, (2015),** "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705-724.
- [40] **A. M. Pinto, L. Rocha and P. Moreira, (2013),** "Object recognition using laser range finder and machine learning techniques," *Robotics and Computer Integrated Manufacturing*, vol. 1, no. 29, pp. 12-22.

- [41] **R. Sutton and A. Barto,(2014,2015)** Reinforcement Learning: An Introduction, London, England: A Bradford Book The MIT Press,.
- [42] **C. Bishop,(2006)** in *Pattern Recognition and Machine Learning*, Cambridge U.K., SPRINGER, p. 3.
- [43] **I. Goodfellow, Y. Bengio and A. Courville (2016)** Deep Learning, An MIT Press book,.
- [44] **A. Azawii, S. F. Al-Janabi and B. Al-Khateeb, (2019)**, "Survey on Intrusion Detection Systems based on Deep Learning," *Periodicas of Engineering and Natural Sciences*, vol. 7, no. 3, pp. 1074-1095.
- [45] **S.Puyanfar, (2018)**, "A Survey on Deep Learning: Algorithms, Techniques, and Applications,"," *ACM Comput. Surv.*, vol. 51, no. 5, p. 92.
- [46] **B. Durokovic, (2017)**, "Design of experiments application, concepts, examples: State of the art," *Period. Eng. Nat. Sci.*, vol. 5, no. 3.
- [47] **A. Şeker, B. Diri and H. H. Balık, (2017)**, "DERİN ÖĞRENME YÖNTEMLERİ VE UYGULAMALARI HAKKINDA BİR İNCELEME," *Gazi Mühendislik Bilimleri Dergisi*, vol. 3, no. 3, pp. 47-64.
- [48] **Y. Lecunn, (1989)**, "Backpropagation Applied to Handwritenn Zip Code Recognition," *Neural Comput.* , vol. 1, no. 4, pp. 541-551.
- [49] **Y. Lecunn, B. Boser, J. Denker, D. Henderson, W. Hubbard and L. Jackel, (1989)**, "Handwritten Digit Recognition with a Back-Propagation Network," in *NIPS*.
- [50] **Z. Q. Zhao, P. Zheng, S.-t. Xu and X. Wu, (2019)**, "Object Detection with Deep Learning: A Review," *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS*, pp. 2-3.
- [51] **G. Hinton and R. Salakhutdinov, (2006)**, "Reducing the Dimensionality of," *SCIENCE AMG*, vol. 313, pp. 504-507.
- [52] **J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, (2009)**, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA.

- [53] **J. Schidhuber**, "Deep learning in neural networks: An overview, (2015), " *Neural Networks*, vol. 61, pp. 85-117.
- [54] **L. Deng, M. Seltzer and D. Yu**, (2010), "Binary coding of speech spectrograms using a deep auto-encoder," in *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association*, Makuhari, Chiba, Japan.
- [55] **G. Dahl, M. Ranzato and A.-r. Mohamed**, (2010), "Phone Recognition with the Mean-Covariance Restricted Boltzmann Machine.," in *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010*, Vancouver, British Columbia, Canada.
- [56] **A. Krizhevsky, I. Sutskever and G. Hinton**, "ImageNet Classification with Deep Convolutional Neural Networks," 2012 07 3. [Online]. Available: <https://arxiv.org/abs/1207.0580>. [Accessed 04 2020].
- [57] **G. Hinton, N. Srivastava and I. Sutskever**, "Improving neural networks by preventing coadaptation of feature detectors," 2012. [Online]. Available: <https://arxiv.org/abs/1207.0580>. [Accessed 04 2020].
- [58] **S. Ioffe and C. Szegedy**, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," 11 02 2015. [Online]. Available: <https://arxiv.org/abs/1502.03167>. [Accessed 05 2020].
- [59] **A. Krizhevsky, I. Sutskever and G. E. Hinton**, (2012), "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in neural information processing systems*, vol. 25, no. 2, pp. 1-2.
- [60] **C. Szegedy , W. Liu, Y. Jia, P. Sermanet and S. Reed**, (2015), "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA.
- [61] **K. He, X. Zhang, S. Ren and J. Sun**, (2016), "Deep Residual Learning for Image Recognition," in *IEEE*, Las Vegas, NV, USA.
- [62] **Y. Jia, E. Schelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama and T. Darrel**, "Caffe: Convolutional Architecture for Fast Feature

- Embedding," 20 06 2014. [Online]. Available: <https://arxiv.org/abs/1408.5093>. [Accessed 04 2020].
- [63] **Z. Cao, T. Simon, S.-E. Wei and Y. Sheikh, (2017),** "Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields," in *IEEE*, Honolulu, HI, USA.
- [64] **Z. Yang and R. Nevatia, (2016),** "A multi-scale cascade fully convolutional network face detector," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, Cancun, Mexico.
- [65] **C. Chen, A. Seff, A. Kornhauser and J. Xiao, (2015),** "DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Araucano Park, Las Condes, Şili.
- [66] **A. Dundar, J. Jin and E. Culurciello, (2017),** "Embedded Streaming Deep Neural Networks Accelerator With Applications," *Computer Science, Medicine*, vol. 28, no. 7, pp. 1572-1583.
- [67] **M. Nielsen,** "<http://neuralnetworksanddeeplearning.com/>," Lambda, TinEye, VisionSmart, 2015. [Online]. Available: <http://neuralnetworksanddeeplearning.com/>. [Accessed 04 2020].
- [68] **K. Rice, (2018),** "CONVOLUTIONAL NEURAL NETWORKS FOR DETECTION AND CLASSIFICATION OF MARITIME VESSELS IN ELECTRO-OPTICAL SATELLITE IMAGERY," Dudley Knox Library, California, USA.
- [69] **W. Liu, L. Ma and H. Chen, (2018),** "Arbitrary-Oriented Ship Detection Framework in Optical Remote-Sensing Images," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 6, pp. 937-941.
- [70] **D. Hubel and T. Wiesel, (1968),** "Receptive fields and functional architecture of monkey striate cortex," *The Journal of Physiology*, vol. 195, no. 1, pp. 215-243.
- [71] **K. Fukushima, (1980),** "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, pp. 193-202.

- [72] **Y. Lecunn, L. Bottou, Y. Bengio and P. Hafner, (1998)**, "Gradient-based learning applied to document recognition," *Proceeding of the IEEE*, vol. 86, no. 11, pp. 2278 - 2324.
- [73] **Y. Lecunn, L. Jackel and B. Boser, (1989)**, "Handwritten digit recognition: applications of neural network chips and automatic learning," *IEEE Communication Magazine*, vol. 27, no. 11, pp. 41-46.
- [74] **R. Duda, P. Hart and D. Stork,(2000)**, Pattern Classification, 2nd Edition, Wiley-Interscience,.
- [75] **K. Simonyan and A. Zisserman, (2015)**, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *International Conference on Learning Representations*, Oxford.
- [76] **S. Tammina, (2019)**, "Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images," *International Journal of Scientific and Research Publications*, vol. 9, no. 10, pp. 143-150.
- [77] **M. M. Leonardo, T. J. Carvalho, E. Rezende, R. Zucchi and F. A. Faria, (2018)**, "Deep Feature-based Classifiers for Fruit Fly Identification (Diptera: Tephritidae)," in *Conference on Graphics, Patterns and Images*, SIBGRABI.
- [78] **G. Huang, Y. Sun and Z. Liu, (2016)**, "Deep Networks with Stochastic Depth," in *European Conference on Computer Vision*, Cornell University, Ithaca, USA.
- [79] **J. Zhang, C. Lu and X. Li, (2019)**, "A full convolutional network based on DenseNet for remote sensing scene classification," *Mathematical Biosciences and Engineering*, vol. 16, no. 5, pp. 3345-3367.
- [80] **F. Chollet, (2017)**, "Xception: Deep Learning with Depthwise Separable Convolutions," in *IEEE*, Honolulu, HI, USA.
- [81] **C. Szedey, W. Liu, Y. Jia, P. Sermanet and S. Reed, (2015)**, "Going deeper with convolutions," in *IEEE*, Boston, MA, USA.
- [82] **J. Redmon, S. Divvala, R. Girshick and A. Farhadi, (2016)**, "You Only Look Once: Unified, Real-Time Object Detection," in *IEEE*, Las Vegas, NV, USA.

- [83] **W. Liu, D. Anguelov, D. Erhan, C. Szedgedy, S. Reed, C.-Y. Fu and A. Berg, (2016)**, "SSD: Single Shot MultiBox Detector," in *The European Conference on Computer Vision (ECCV)*, Amsterdam.
- [84] **J. Huang, V. Rathod and C. Sun, (2017)**, "Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Hawaii.
- [85] **H. Honda**, "www.medium.com," www.medium.com, 1 02 2019. [Online]. Available: <https://medium.com/@hirotoschwert/reproducing-training-performance-of-yolov3-in-pytorch-part1-620140ad71d3>. [Accessed 05 2020].
- [86] **D. E. Rumelhart, G. E. Hinton and R. J. Williams, (1986)**, "Learning representations by back-propagating errors," *nATURE*, vol. 323, no. 533.
- [87] **D. Wilson and T. R. Martinez, (2003)**, "The general inefficiency of batch training for gradient descent learning," *Elsevier Computer Science*, pp. 1429-1451.
- [88] **E. S. Olivas, J. D. M. Guerrero, M. Martinez-Sober, J. R. Magdalena-Benedito and A. J. Lopez ,(2009)**, *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, Scopus.
- [89] **S. J. Pan and Q. Yang, (2009)**, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345-1359.
- [90] **Y. Yao and G. Doretto, (2010)**, "Boosting for transfer learning with multiple sources," in *Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Francisco.
- [91] **J. Deng, W. Dong and R. Socher, (2009)**, "ImageNet: a Large-Scale Hierarchical Image Database," in *Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami, Florida, USA.

- [92] ImageNet, "Image-net.org.," Stanford Vision Lab, Stanford University, Princeton University , 2016. [Online]. Available: <http://www.image-net.org/>. [Accessed 01 04 2020].
- [93] **L. Xie and Yuille, (2017),** "Genetic CNN," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy.
- [94] **J. Bergstra and Y. Bengio, (2012),** "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281-305.
- [95] **P. Radiuk, (2018),** "Impact of Training Set Batch Size on the Performance of Convolutional Neural Networks for Diverse Datasets," *Information Technology and Management Science*, vol. 20, no. 1, pp. 20-24.
- [96] **S. Sharma,** "Towards Data Science," Towards Data Science, 23 09 2017. [Online]. Available: <https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9>. [Accessed 15 04 2020].
- [97] **N. Srivastava, G. Hinton, A. Krizhevsky , I. Sutskever and R. Salakhutdinov, (2014),** "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929-1958.
- [98] **C. Kusuma and A. Afiahayati, (2018),** "Suitable CNN Weight Initialization and Activation Function for Javanese Vowels Classification," *Procedia Computer Science*, vol. 144, pp. 124-132.
- [99] **P. Ramachandran and B. Zoph, (2018),** "Searching for Activation Functions," in *ICLR 2018 Conference*.
- [100] **V. Nair and G. Hinton, (2010),** "Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair," in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel.
- [101] **M. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang and P. Nguyen, (2013),** "On rectified linear units for speech processing," in *IEEE*, Vancouver, BC, Canada.
- [102] **C. E. Nwankpa, W. Ijomah, A. Gachagan and . S. Marshall,** "Activation Functions: Comparison of Trends in Practice and Research for Deep Learning," 11

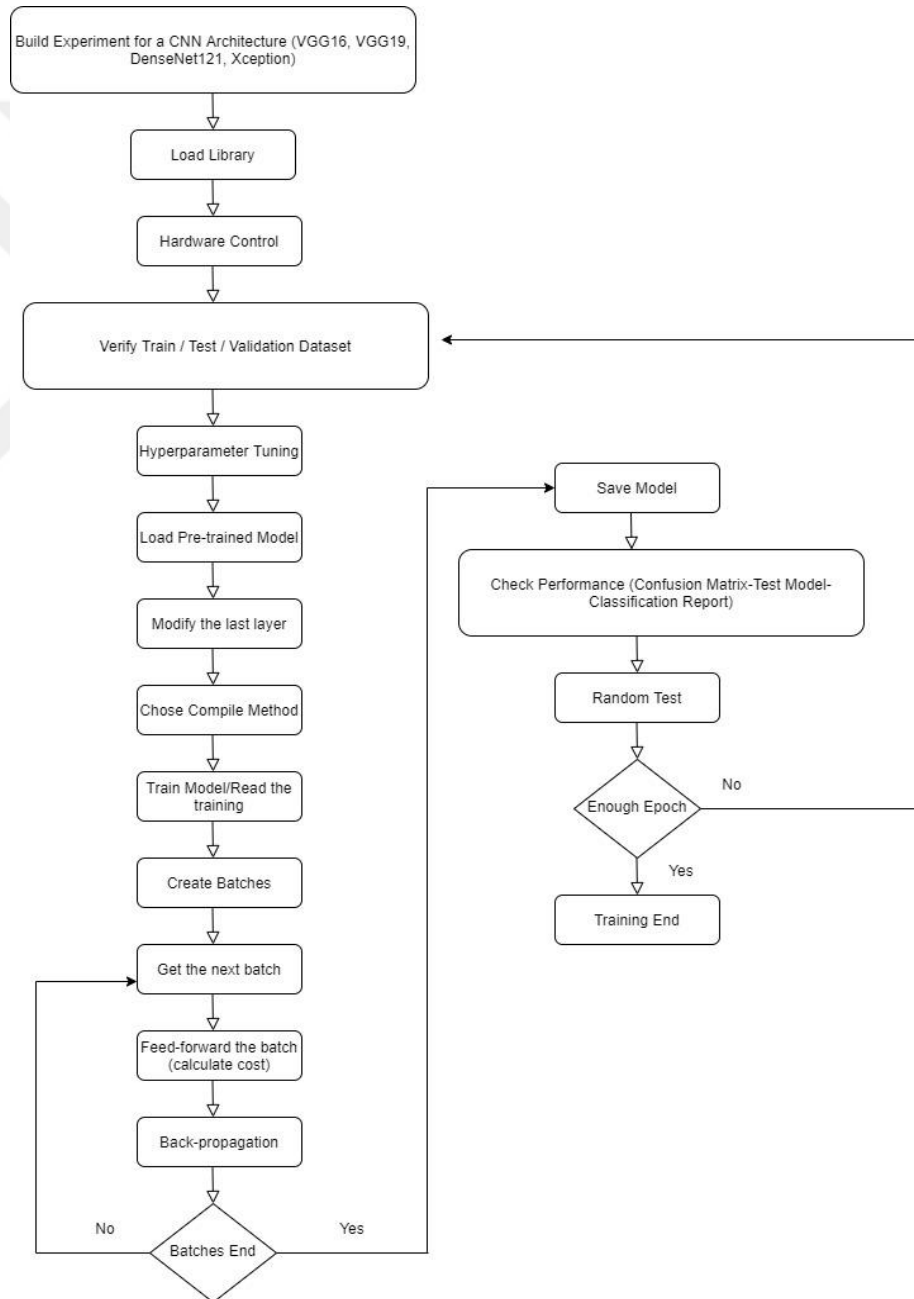
- 08 2018. [Online]. Available: <https://arxiv.org/pdf/1811.03378.pdf>. [Accessed 12 05 2020].
- [103] **V. Badrinarayanan, A. Kendall and R. Cipolla, (2017)**, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481-2495.
- [104] **J. Hann and C. Moraga, (1995)**, "The Influence of the Sigmoid Function Parameters on the Speed of Backpropagation Learning," *IWANN '96: Proceedings of the International Workshop on Artificial Neural Networks: From Natural to Artificial Neural Computation*, pp. 195-201.
- [105] **X. Glorot and Y. Bengio, (2010)**, "Understanding the difficulty of training deep feedforward neural networks," *Journal of Machine Learning Research*, vol. 9, pp. 249-256.
- [106] **I. Sutskever, J. Martens, G. Dahl and G. Hinton, (2013)**, "On the importance of initialization and momentum in deep learning," *Proceedings of Machine Learning Research*, vol. 28, no. 3, pp. 1139-1147.
- [107] **L. Bottou, (2010)**, "Large-Scale Machine Learning with Stochastic Gradient Descent," in *Proceedings of COMPSTAT'2010*, USA.
- [108] **D. Kingma and J. Ba**, "Adam: A method for stochastic optimization," 30 01 2017. [Online]. Available: <https://arxiv.org/abs/1412.6980>. [Accessed 15 04 2020].
- [109] **M. Zeiler**, "ADADELTA: An adaptive learning rate method," 01 12 2012. [Online]. Available: <https://arxiv.org/pdf/1212.5701.pdf>. [Accessed 20 03 2020].
- [110] **S. Ruder**, "An overview of gradient descent optimization algorithms," 15 06 2017. [Online]. Available: <https://arxiv.org/abs/1609.04747>. [Accessed 20 03 2020].
- [111] **R.-Z. Liang, W. Xie, W. Li, H. Wang, Y. Wang and L. Taylor, (2016)**, "A Novel Transfer Learning Method Based on Common Space Mapping and

- Weighted Domain Matching," in *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, San Jose, CA, USA.
- [112] **R.-Z. Liang, L. Shi, H. Wang, J. Meng and Y. Wnag, (2016)**, "Optimizing top precision performance measure of content-based image retrieval by learning similarity function," in *23rd International Conference on Pattern Recognition (ICPR)*, Cancun, Mexico.
- [113] **Q. Li, X. Zhou, A. Gu, Z. Li and R.-Z. Liang, (2016)**, "Nuclear norm regularized convolutional Max Pos@Top machine," *Neural Computing and Applications* , vol. 30, pp. 463-472.
- [114] **R. Reed and R. MarksII, (1999)** *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks*, A Bradford Book,.
- [115] **G. James, D. Witten, T. Hastie and R. Tibshirani, (2017)** *An Introduction to Statistical Learning*, Los Angelas: Springer,.
- [116] **B. Ripley, (1996)** *Pattern Recognition and Neural Networks*, Oxford: Cambridge University Press.
- [117] **S. Russell,(2016)**, *Artificial Intelligence: A Modern Approach*, Pearson Education.
- [118] **R. Kohavi, (1995)**, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," *IJCAI'95: Proceedings of the 14th international joint conference on Artificial intelligence*, vol. 2, pp. 1137-1143.
- [119] **M. Sokolova, N. Japkowicz and S. Szpakowicz, (2006)**, "Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation," in *Australasian Joint Conference on Artificial Intelligence*, Berlin.
- [120] **S. Birch and S. Pine, (2019)**, "Tree species classification," KTH ROYAL INSTITUTE OF TECHNOLOGY, Stockholm, Sweden.
- [121] Wikipedia, "WIKIPEDIA, The free encyclopedia," Wikipedia, [Online]. Available:
[https://en.wikipedia.org/wiki/Evaluation_measures_\(information_retrieval\)#Mean_average_precision](https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)#Mean_average_precision). [Accessed 10 03 2020].

- [122] **M. Schwarz, (2017),** "Object Localization for Warehouse Automation,"
Universität Bonn, BONN.
- [123] **D. Gürkaynak, (2018),** "Fine-Tuning Convolutional Neural Networks For
Maritime Vessel Classification, Verification And Recognition," The Republic Of
Turkey Bahcesehir University, Istanbul.
- [124] **S. Ramani, Prabakaran N, Rajkumar S. (2019),** "Real Time Detection And
Segmentation Of Ships In Satellite Images," vol. 8, no. 12.
- [125] C. Research, "unctadstat.unctad.org," UNCTAD, 2019. [Online]. Available:
unctadstat.unctad.org. [Accessed 06 2020].
- [126] **G. Huang, (2017),** "Densely connected convolutional networks" *Proceedings of
the IEEE conference on computer vision and pattern recognition*, vol. 1, no. 2.
- [127] **S. Ruder, (2016),** "An overview of gradient descent optimization algorithms,".

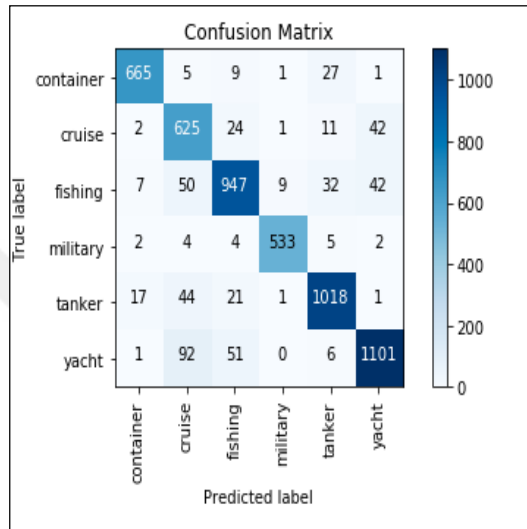
APPENDICES

APPENDIX A: FLOWCHART

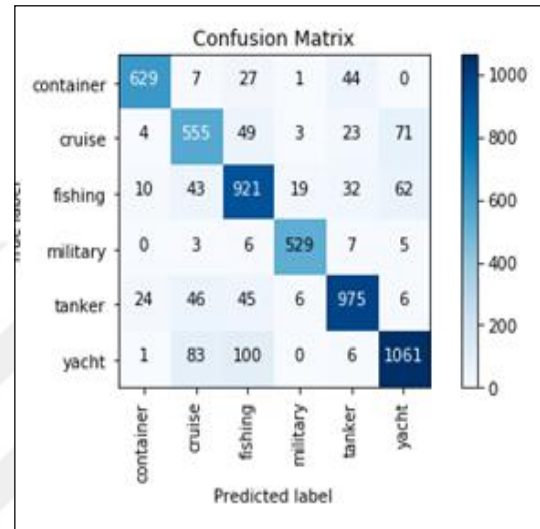


APPENDIX B: VGG16/VGG19/XCEPTION/DENSENET121 CONFUSION MATRIX

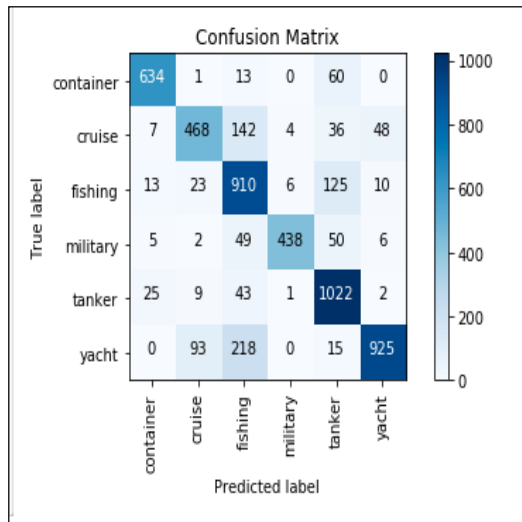
VGG16



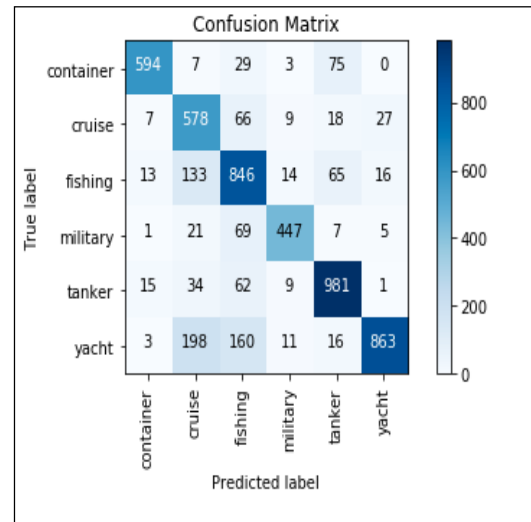
VGG19



Xception



DensetNet121



APPENDIX C: EVALUATION PREDICTION FOR VGG16



Confidence:%89
Label: Container
Prediction: Container



Confidence:%91
Label: Cruise
Prediction: Cruise



Confidence:%90
Label: Fishing
Prediction: Fishing



Confidence:%89
Label: Military
Prediction: Military



Confidence:%89
Label: Tanker
Prediction: Tanker



Confidence:%89
Label: Yatch
Prediction: Yatch

APPENDIX D: EVALUATION PREDICTION FOR VGG19



Confidence:%89

Label: Container
Prediction: Container



Confidence:%86

Label: Cruise
Prediction: Cruise



Confidence:%85

Label: Fishing
Prediction: Fishing



Confidence:%89

Label: Military
Prediction: Military



Confidence:%89

Label: Tanker
Prediction: Tanker



Confidence:%83

Label: Yatch
Prediction: Yatch

APPENDIX E: EVALUATION PREDICTION FOR XCEPTION



Confidence:%81

Label: Container
Prediction: Container



Confidence:%77

Label: Cruise
Prediction: Cruise



Confidence:%83

Label: Fishing
Prediction: Fishing



Confidence:%77

Label: Military
Prediction: Military



Confidence:%56

Label: Tanker
Prediction: Container



Confidence:%81

Label: Yatch
Prediction: Yatch

APPENDIX F: EVALUATION PREDICTION FOR DENSENET121



Confidence:%69

Label: Container
Prediction: Container



Confidence:% 53

Label: Cruise
Prediction: Fishing



Confidence:% 75

Label: Fishing
Prediction: Fishing



Confidence:%80

Label: Military
Prediction: Military



Confidence:%75

Label: Tanker
Prediction: Cruise



Confidence:%78

Label: Yatch
Prediction: Yatch

APPENDIX G: FULLY CONNECTED LAYER NEURON SIZE PERFORMANCE COMPARISON FOR MODIFIED MODELS

Modified Model	fc1-layer	fc2-layer	fc3-layer	Accuracy
VGG16 + fc1+ fc2	512	256		0.91
VGG16 + fc1+ fc2 + fc3	256	128	64	0.87
VGG16 + fc1+ fc2	256	128		0.84
VGG16 + fc1+ fc2 + fc3	128	64	32	0.83
VGG19 + fc1	1024			0.85
VGG19 + fc1 + fc2	1024	512		0.86
Xception + fc1	1024			0.84
Xception + fc1 + fc2	1024	512		0.83
Xception + fc1 + fc2 + fc3	1024	512	256	0.79
DenseNet121 + fc1	32			0.79
DenseNet121 + fc1 + fc2	512	256		0.75
DenseNet121 + fc1 + fc2 + fc3	512	256	128	0.71

APPENDIX H: I USE HYPERPARAMETERS FOR EACH MODIFIED MODELS

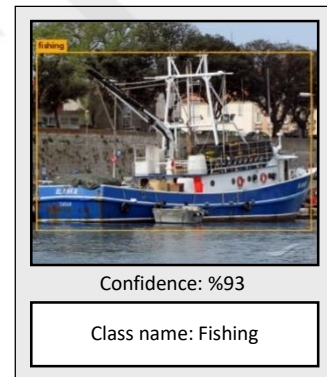
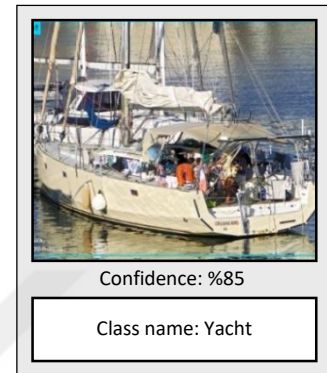
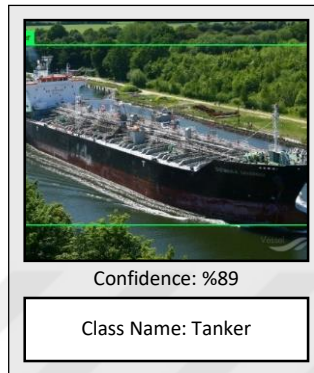
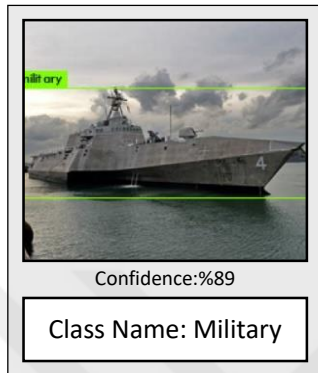
Modified Model	Input Picture Size	Learning Rate	Dense Layer Activation	Drop out	Optimizer	Accuracy	Precision	Recall	F1-Score
VGG16 + [ImageNet] + [Own Dataset]-1	600x600	0.001	soft max	0.2	Adam	0.91	0.91	0.91	0.91
VGG16 + [ImageNet] + [Own Dataset]-2	300x300	0.0001	soft max	0.2	SGD	0.87	0.84	0.84	0.84
VGG16 + [ImageNet] + [Own Dataset]-3	400x400	0.0001	soft max	0.5	SGD	0.84	0.85	0.84	0.84
VGG19 + [ImageNet] + [Own Dataset]-1	400x400	0.01	soft max	0.5	Adadelata	0.86	0.87	0.87	0.87
VGG19 + [ImageNet] + [Own Dataset]-2	600x600	0.01	soft max	0.5	Adadelata	0.85	0.86	0.86	0.86
VGG19 + [ImageNet] + [Own Dataset]-3	200x200	0.01	soft max	0.5	Adamax	0.82	0.84	0.83	0.83
Xception + [ImageNet] + [Own Dataset]-1	300x300	0.01	soft max	0.5	Adadelata	0.84	0.84	0.81	0.82
Xception + [ImageNet] + [Own Dataset]-2	400x400	0.1	soft max	0.5	Adadelata	0.81	0.84	0.82	0.82

APPENDIX H: I USE HYPERPARAMETERS FOR EACH MODIFIED MODELS

(Continued)

Modified Model	Input Picture Size	Learning Rate	Dense Layer Activation	Drop out	Optimizer	Accuracy	Precision	Recall	F1-Score
Xception + [ImageNet] + [Own Dataset]-3	400x400	0.1	soft max	0.25	Adadelata	0.80	0.83	0.81	0.81
DenseNet 121 + [ImageNet] + [Own Dataset]-1	200x200	0.01	soft max	0.5	SGD	0.79	0.82	0.80	0.80
DenseNet 121 + [ImageNet] + [Own Dataset]-2	200x200	0.01	Sig moid	0.5	Adam	0.75	0.79	0.78	0.77
DenseNet 121 + [ImageNet] + [Own Dataset]-3	200x200	0.01	soft max	0.5	SGD	0.69	0.72	0.72	0.70

APPENDIX I: TEST RESULTS ON OUR OWN 3000 DATASET



APPENDIX J: TEST RESULTS ON OUR OWN 6000 DATASET

