TÜRKİYE

FIRAT UNIVERSITY

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE

Guns Detection Using YOLO Algorithm

Rayan SULAIMAN KHALAF

Master's Thesis

DEPARTMENT OF SOFTWARE ENGINEERING

May 2021

TÜRKİYE

FIRAT UNIVERSITY

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE

Department of Software Engineering

Master's Thesis

Guns Detection Using YOLO Algorithm

Author

Rayan SULAIMAN KHALAF

Supervisor

Assistant Prof. Dr. Yaman AKBULUT

May 2021

ELAZIĞ

TÜRKİYE

FIRAT UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE

Department of Software Engineering

Master's Thesis

Title: Guns Detection Using YOLO Algorithm

Author: Rayan SULAIMAN KHALAF

Submission date: 8 April 2021

Defense date: 7 May 2021

THESIS APPROVAL

This thesis, which was prepared according to the thesis writing rules of the Graduate School of Natural and Applied Sciences, Fırat University, was evaluated by the committee members who have signed the following signatures and accepted unanimously as a result of the defense exam made open to the academic audience.

Supervisor:	Assistant Prof. Dr. Yaman AKBULUT Firat University, Technology Faculty	Approved
Member:	Associate Prof. Dr. Fatih ÖZYURT Fırat University, Engineering Faculty	Approved
Member:	Assistant Prof. Dr. Ali ARI İnönü University, Engineering Faculty	Approved

This thesis was approved at the meeting of the Administrative Board of the Graduate School on ..... / ..... / 20 .....

Associated Prof. Dr. Kürşat Esat ALYAMAÇ

Director of the Graduate School

DECLARATION

I hereby declare that I wrote this Master's Thesis titled Guns Detection using YOLO Algorithm in consistent with the thesis writing guide of the Graduate School of Natural and Applied Sciences, Firat University. I also declare that all information in it is correct, that I acted according to scientific ethics in producing and presenting the findings, cited all the references I used, express all institutions or organizations or persons who supported the thesis financially. I have never used the data and information I provide here in order to get a degree in any way.

7 / May /2021

Rayan SULAIMAN KHALAF

PREFACE

The most popular branch in modern life (our daily life too) is AI and its branches because most practitioners are searching for new inventions and updates in this branch. Besides, the most paramount side of our life is security. This subject is an idea to make our life more secure, more controllable, more monitored by the government, it is a smart system to detect arms in a fast way before terrorists can make any attacks in public or government institutions. Most great countries are using smart systems to control most branches in their community, like traffics, bus stations, auto-drive cars, medical, industrial and more branches. Our experiment is a Guns detection system in a smart way that can be used in public areas to monitor it from terrorist attacks.

We built the system with simple tools and a normal laptop that was the main reason to be difficult and take more time for us. We are making a system that needs images of guns that may not be available on searching engines, which leads to making a dataset with ourselves and that what we did (making a dataset take much time).

We mentioned that we used simple tools, and the system needs a computer with good RAM, fast CPU, good HARD, with GPU too, these factors make the process easier and faster for us but without having these tools which led to taking much time for the process to be finished. We made this system hoping to be a point to build systems more accurate, monitoring all our life from any terrorist attacks.

TABLE OF CONTENTS

ACKNOWLEDGMENT

I thank everyone who has been a part of this work the supervisor, friends, and special thanks to my family who were the backbone for supporting me.

I dedicate this work to the first and great school, to the soul of my mother who is immortal in my memory.

ABSTRACT

Guns Detection Using YOLO Algorithm

Rayan SULAIMAN KHALAF

Master's Thesis

FIRAT UNIVERSITY
Graduate School of Natural and Applied Sciences

Department of Software Engineering

May 2021, Pages: xvi + 62

This approach focuses on the technology of object detection, the various and innovation methods used in this field of machine learning in general, focusing on object detection in particular. In which we clarify the fundamental things in this field, which have been developed gradually and become the focus of attention to the development of the world at present. The practical aspect of this research explains the use of the You Only Look Once (YOLO) technique, as this branch of object detection is the most developed and most updated branch used in the last couple of years. Our idea is detecting guns in real-time videos or monitoring cameras, so we need the speed of processing that could process frames per second; this property is available in YOLO as the best technique having the speed of the process. We have created a dataset for a specific weapon (AKM-47) as a test for the technique if the camera can detect weapons in real-time or not. We mentioned that despite the sophistication of this technology and its need for some large specifications such as a high-speed memory like SSD's, a good CPU's and also needs GPU's. But it still is very important to process it, because if it is applied on the ground of its art, it will help many people avoid becoming victims of terrorism and murder.

Keywords: computer vision, machine learning, object detection, YOLO, deep learning

ÖZET

YOLO Algoritmasını Kullanarak Silah Tespiti

Rayan SULAIMAN KHALAF

Yüksek Lisans Tezi

FIRAT ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü
Yazılım Mühendisliği Anabilim Dalı
Mayıs 2021, Sayfa: xvi + 62

Bu yaklaşım, nesne algılama teknolojisine, genel olarak bu makine öğrenimi alanında kullanılan çeşitli ve inovasyon yöntemlerine, özellikle de nesne algılamaya odaklanır. Gün geçtikçe gelişen ve günümüzde dünyanın gelişiminin ilgi odağı haline gelen bu alandaki temel şeyleri açıklığa kavuşturuyoruz. Bu araştırmanın pratik yönleri, yalnızca Bir Kez Bakarsınız (YBKB) tekniğinin kullanımını açıklamaktadır, çünkü nesne algılama biliminin bu dalı, son birkaç yılda kullanılan en gelişmiş ve en güncel daldır. Bizim fikrimiz silahları gerçek zamanlı videolarda tespit etmek veya kameraları izlemek, bu nedenle saniyede kare işleyebilecek işleme hızına ihtiyacımız var ve bu özellik, işlem hızına sahip en iyi teknik olarak YOLO'da mevcuttur. Tekniği test etmek için belirli bir silah (AKM-47) için bir veri seti oluşturduk, kamera silahları gerçek zamanlı olarak tespit edebiliyorsa, bu teknolojinin karmaşıklığına ve bazı büyük spesifikasyonlara olan ihtiyacına rağmen bahsettik. SSD'ler gibi yüksek hızlı bir bellek gibi, iyi bir CPU'lara ve ayrıca GPU'lara ihtiyaç duyuyor, ancak onu işlemek için hala çok önemli, çünkü zeminde uygulandığında, birçok kişinin terörizm ve cinayet kurbanı olmasını önleyecektir.

Anahtar Kelimeler: bilgisayarla görme, makine öğrenimi, nesne algılama, YOLO, derin öğrenme

LIST OF FIGURES

Page

LIST OF TABLES

Page

LIST OF ABBREVIATIONS

Abbreviations

Adaboost	: Adaptive Boosting
AI	: Artificial Intelligence
AKM-47	: Automatic relief Kalashnikov Modernize-1947
AMT	: Amazon Mechanical Turk
ANN	: Artificial Neural Network
ATM	: Alien Time Machine
CISCO	: Computer Information System Company
CNN	: Convolutional Neural Network
COCO	: Common Object in Context
DBM	: Deep Belief machine
DPM	: Data Processing Machine
DBN	: Deep Believe Network
DCNN	: Deep Convolutional Neural Network
DEM	: Deep Energy Model
DOG	: Difference of Gaussian
ELM	: Extreme Learning Machine
FPN	: Feature Pyramid Network
FPS	: Frame per Second
GPU	: Graphical Processing Unite
HDD	: Hard Disk Drive
HED	: Heavy Edge Detection
HOG	: Histogram Of oriented Gradient
IMFDB	: Internet Movie Firearms Data Base
IOU	: Intersection of Union
LSVRC	: Large Scale Visual Recognition Challenge
MAP	: Mean Average Precision
RBN	: Restricted Boltzmann Network
R-CNN	: Regional Convolutional Neural Network
RNN	: Recurrent Neural Network

ROI	: Region of Intersection
RP	: Regional Purposed
SIFT	: Scale Invariant Feature Transform
SSD	: Single Shot Detection
SSD	: Solid-State Drive
SVM	: Support Vector Machine
VGG	: Visual Geometry Group
VOC	: Virtual Object Classes
YOLO	: You Only Look Once

DEFINITIONS

Artificial Intelligence: intelligence in public refers to thinking, decision-making and so on. If we talk about human intelligence, we can talk about emotions, judgment, thinking and more decisions. Artificial intelligence, which had appeared in the 1950s, means machine thinking that just transfers a program to some operations, with limited intelligence, high accuracy (depending on the system it is designed for) [22]. Artificial Intelligence (AI) is a branch that is being updated gradually, making machines and computers able to think, making decisions, performing a function, and solving problems like human beings [28]. This development leads to improving AI, makes it able to translate languages, voices, objects recognition and others [29]. We notice that there are different expressions to define Artificial Intelligence; the most likely one is that branch studies algorithms and programming languages trying to improve it continuously.

Support vector machine: is a function or formula that works as a factor to determine and detect the object inside the detection algorithm used, this is the public definition of it. It works as a separator among classes and classifies each object or Regional Proposed (RP) to a specific class it belongs to [28]. Especially, inside object detection SVM works after the step of NN, the object after NN step have some properties, SVM takes these properties and begins to compare it with the properties of classes it is having [30]. We can explain it as a wall between rooms that contain objects. It is called the **maximum margin separator** inside the computer science branch. In object detection is defined as a percentage of similarity between RP and dataset objects. This step leads to minimizing the generalization of the detection, and as a result, this step makes the algorithm works faster with more accuracy [30].

Boosting: an operation of transforming weak hypothesis accuracy to a learnable hypothesis that can be processed through an algorithm [31]. This operation could be a practical step or a program constructing algorithm with good efficiency from one that looks weak, in addition to the theoretical steps by which the general limits of an algorithm must be observed [31]. This operation can be done by running an algorithm or program more times and collecting the majority vote of these repeated results. If a strong learner can solve a problem, a collection of weak learners can do it too [31, 44].

Graphical models: is a traditional object representation, by segmenting an object into graphical parts and making a dependency relationship among them, representing the features of the object inside the CNN's. Then the graphical inference owns the properties that merge well to represent an image for the object [32]. The most famous algorithms used are Bayesian network AI and Markov networks.

Adaboost: is an adaptive boosting that appears after boosting the technique invented, as an advanced adapter for converting the weak classifiers into good classifiers that can be used for object detection [33]. This operation is done by taking an exponential criterion as a cost function and newton's method for optimizing, then an algorithm is going to derive strong classifiers from weak ones [33].

Upsampling operation: This operation was widely used in object detection, semantic segmentation, and instance segmentation to reproduce an image with a bigger size with the same resolution, to use it inside CNN we use it [49]. This operation is representing every pixel inside an image with four pixels with the same value or with zero values for pixels we added them to recreate an image in different size with the

same value [49]. The operation that completely works opposite is known as downsampling operation, which works on pooling an image and making its size smaller with the same resolution.

.

1. INTRODUCTION

Computer vision is an interesting branch to study, not for software engineers only but is for everybody interested in computer science and modern technology [1]. We can define it as the ability of machines to have a vision system just like a human, the operation of a human vision system is very similar to the computer vision system. In the human vision system, firstly, the eye takes images 10-12 image per second, then the nerves take these images as electrical pulses and transmit it to the brain, which in turn begins to analyze pulses and divide them into multiple layers to extract information. This information will be identified by an attention system that will compare it with internal information and take the important part to make a decision and give feedback to the brain again [13, 14, and 47].

The computer vision system is similar somewhat to the human visual system. This system consists of input data (image/video) from memory or camera, takes images/video of the environment outside, then the system process and analyze to extract information from it which that leads to making a decision depending on it [1, 47]. Especially, nowadays, where cameras are inexpensive, computers are available for everyone with good properties that can run this type of processing, and some websites can provide a good dataset for everything you want [1]. Another cause that leads practitioners to work more is widespread of internet users and the global data exchanged around the world. We can see that by looking at the annual report made by Computer Information System Company (CISCO) which gives clear information about this topic, as we can see in figure 1.1 showing the growth:

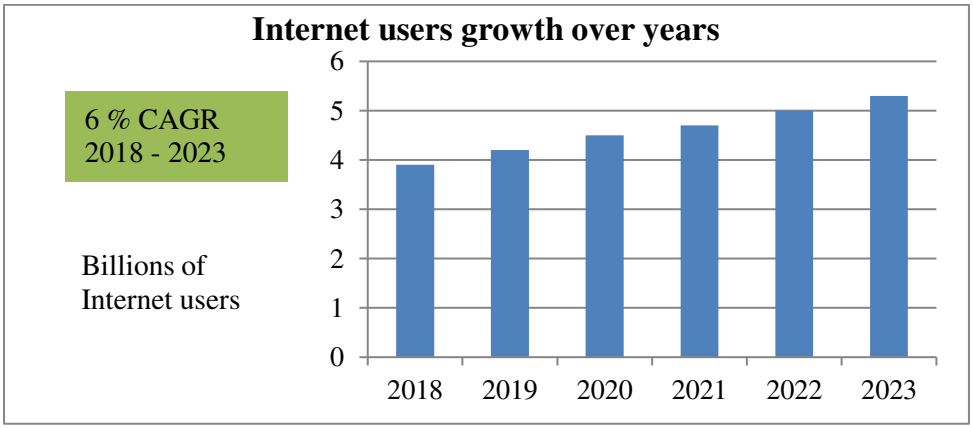

Figure 1.1. Internet user's growth [59]

Every operation inside computers goes through three main steps: input, processing, and output. In computer vision, it is the same thing in which the input may be a voice or image/frame. Processing operation will analyze this voice or image/frame, and then the output will be a decision to make some operations depending on inputted data [4]. We can find computer vision everywhere, For example, self-driven cars, surveillance, traffic, forensic studies, medical systems, monitoring, food industries, etc. This branch made human life easier [1]. We can describe a few applications that are widely used in our life and are of great significance, such as [47]:

- Figure print recognition.
- Machine inspections.
- Retail.
- Medical imaginary.
- Object detection

These modern techniques grow up and get improved because of the continuous work of researchers and institutes that keep making it better and better. This leads practitioners to search for the most updated books and researches to improve their information to analyze and work on what is more modern [1]. The best thing about this branch is that it is interesting for students to study, improve, and test every new technique that appears. The reason is that it is a user face interaction and the practitioner can see the result directly [47].

Computer vision as a process takes images as an array of digits, analyze them depending on filters and extract their features. Then, it classifies them into specific classes or groups [46]. Some problems appear in computer vision such as an image is 2D while the real world is 3D, the difference of point of view, the difference in size (near or far), the difference in colour (like difference clothes), image resolution and size, the noise that appears in images because of the surrounding circumstances, etc. [48, 49].

Computer vision is divaricated into sub-branches that differ from each other in small details like a kind of grouping objects, kind of viewing information, accuracy, speed, algorithms used inside, etc. Here we will take an overview of these branches and how each one is different. As basic branches that are the most famous in computer vision are:

- Classification
- Classification with Localization
- Object detection
- Semantic Segmentation
- Semantic Instance segmentation

Classification is a branch of computer vision that classifies images or groups depending on their components or containing object. The difference is that in this type of vision the system is responsible for classifying images into their categories only without detecting or pointing to the object inside the image, the system is responsible for extracting image features and compare them to predefined classes that are stored in the system then classify it. These features may be colour, size, texture, shape, etc. Then the system checks the similarity with the object inside the image. This type is the base of other types of computer vision like detection and localization [50, 51].

Classification and Localization appeared after that, it is the classification process plus localizing the object itself inside an image as an improvement and a step forward to other enhancement [51]. We should know that the output will be changed from classification because in localization we should refer to a specific object inside an image with a box named bounding box, we can also refer to the classification and localization in a single object inside an image that is different from object detection [51]. This step or enhancement proved that we can build a computer vision system that can detect everything in real-time as we will see in object detection algorithms.

Another type of computer vision is **Object Detection**, one of the most modern and updated branches inside computer vision, we can also say is the most spread in computer vision. Is differ or is an enhancement of localization type, because in object detection we can detect more than one variable in the same image [51]. This type of computer vision has become widespread in multiple branches like medicine, industry, surveillance, and more branches. The basic idea is to detect more objects in one image or frame especially in real-time, and our research will be in this branch (object detection in real-time) exactly in the YOLO algorithm.

The basic processing is going to analyze a set of images by applying a filter of a specific object that we want to make the system detect it, sliding this filter on all the pixels of the image row by row searching on the object we want to detect [51]. In CNN that we will discuss in-depth, more filters differ in size that will apply to the images to make weights that can help the system detect objects [51]. There are more

operations done inside like selective search, loss function, number of layers, accuracy, and more we will discuss in this research brief.

Semantic Segmentation In this type, we take an image as input and the output will be a decision or information about this image, but it will decide for every pixel inside this image. This makes semantic segmentation more complex needing more options and powerful hardware to process it in real-time. The technique will change the inputted image into several parts representing particular objects, and then make a decision depending on it [48]. This type of computer vision is used widely in self-driving cars because of its accuracy in making decisions and accuracy used in the entire system of analyzing [48].

The last type we will explain is **Semantic Instance Segmentation,** which is also a type of computer vision in which the system is responsible for detecting the object and its categories like its shape and size, representing its pixels with a mask that is different from object detection and semantic segmentation in some details [52]. However, it also combines two systems in one, and it is different from object detection because it is not the only detection with a bounding box, but also there is pixel identification and masking the object [52]. It is different from semantic segmentation because it is not a pixel classification only, but also detecting individual properties for objects inside the same class [52]. We can find this type of computer vision in self-driving, robotics, etc.

1.1. Related Works

Technology is available to serve human, everyday scientists, programmers, and practitioners are trying to invent and improve technological inventions to be more adaptable to human life. Computer vision is one of these technologies that give a big service in our life; we can find it everywhere in our daily life in medicine, security, surveillance, traffic and more. Nowadays, we also notice that several terrorist accidents happen throughout the entire world. Hence, an idea has come for an attempt to make the monitoring system able to detect guns through cameras to help security system controlling the state and make it under their control. Our project is detecting guns with one type of guns (AKM-47), with a simple dataset for detection, using the YOLO algorithm. That being said, if this topic gets more support, we think it will be a worthwhile security project for monitoring public spaces as well as government institutions. Also, there are some researches and studies about this topic that we can take advantage of.

There is research named "**use of deep learning for firearms detection in images**", published on 17/9/2019, they made a small dataset containing 608 images for weapons, processed it with a convolutional neural network in YOLO to detect weapons, and the detector has an accuracy rate of 89.15% with a sensitivity reaching 100%, it can be applied on weapons detection on images and real-time videos [53].

"**A HED-optimized Automatic Detection and Tracking Algorithm for Marine Moving Targets based on YOLO V3**", is another paper discussing targets detection using YOLO V3 detection algorithm enhanced by heavy edge detection (HED), this paper was published in the 2019 2nd international Symposium on Power Electronics and Control Engineering (ISPECE 2019) 22–24 November 2019, Tianjin, China published online in Feb-7-2020 [54]. The practitioners used an old dataset of ships detection with a new one that they optimized with the HED algorithm and created a new powerful one [54]. The algorithm applied in YOLO V3 with HED algorithm, using space-based marine target image, with a 1000 images containing divided into 80% training and 20% testing groups, based on VGG net [54]. Inside this research, there are several factors or categories we have found, some of them are positive or advantageous and others are negative or disadvantageous, for example, this method or algorithm gives a piece good information about the environment of targets and its type, which is cleverer than radars used before [54]. Another factor is clouds or unclear weather, this type of detection will be affected and the accuracy will be less [54]. Another factor that we have found in this research is the detection that will be on oceans and seas which will be easier for the processor to detect because of the clear background of the processed image [54].

We have found another paper about research that is superficially similar to our research named **"Developing a Real-Time Gun Detection Classifier"** by Justin Lai and Sydney Maples, published in 2017 by Stanford University [55]. They tried to build a gun detection system using YOLO and Tensorflow-based implementation with a dataset of 2753 images downloaded from the Internet Movie Firearms Database (IMFDB) that contain this type of images, they used 2535 images for the training set and 218 for the test set with 640×480 dimensions, using VGG 16 as a baseline, with accuracy reached to 58% [55]. We think this research is slow in detecting depending on the big dataset that they had. Detection takes 1.3 seconds for every detection process in this situation is not possible to use it in real-time video detection [55]. They also used a dataset downloaded from the internet which is different from our research that we made by hand. This research is similar in working but is different in the building because they downloaded images while our research is all built under our control.

"**Gun detection system using YOLOv3" a** paper was published by a group of students in Proc. of the 2019 IEEE 6th International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA 2019) 27-29 August 2019, Kuala Lumpur, Malaysia. They made their dataset containing guns in multi-positions and different angles to make the dataset rich in different images of guns with help of the Image-net database. The implementation was done on YOLOv3 with one type of guns to explain how YOLOv3 works and how can be determined to make a system of guns detection. The paper did not explain the system clearly; it just compared their dataset with the Image-net benchmark, the difference in detection speed and accuracy [77].

There was another experiment we found while our research study named **"A real-time object detection algorithm for video"** by a group of student, they made a nice idea by mixing frame difference and background difference methods to separate the object from the background and avoid the noise. These two methods are working on video frames to appoint the object by examining the sequential frames and their change from one frame to another (frame difference method), while the background difference method is examining the background factors like changing the light and others [79].

The experiment was done on vehicles; they used videos and processed them to collect 8000 images for processing, they implemented it on the real-time video to detect vehicles. The experiment made YOLO faster and more accurate depending on their results, fps reached 45 with accuracy near to 88% is a good result [79].

Another paper we found named "**YOLO-face: real-time face detector**" a new idea was to build a new algorithm based on YOLO-v3 with an improvement in YOLO loss function, the anchor boxes size and

shape. They depended on Darknet-53 for feature extraction network as well as detection networks (three of them), to increase the speed and accuracy of YOLO-v3 for small objects and especially faces. Concerning improving the anchor boxes, they used the WIDER FACE dataset as a tool to help the procedure to produce not only the standard anchor boxes of YOLO but also other possibilities with other dimensions. After this operation, the IOU will work on possibilities more than YOLO-v3 possibilities producing more classes.

Their dataset contains 32203 images, a big number of images will rich the system with good information, and as a result, the system will be more accurate. The challenge they made was testing their YOLO-face with FDDB, and the result was good and improved the speed reached to 45 fps [80].

2. COMPUTER VISION

The vision operation of the human eye in the field of medicine is specialized to take visual information and compose them into coherent pictures of our world. The visual pathway begins in the eye. It starts when the light passes through the lens of the eye and reaches the retina. The retina is responsible for the formation of the image in the eye. The cornea is the front window in the front of the eyeball. The main function of the cornea is light-ray bending where the light rays pass through the pupil of the eye. The human eye has as many features as the camera has where the lens focus on light the cornea of the eye bends light rays and that leads to the formation of the image on the retina. The retina is the receptive surface inside the back of the eye. The retina inverts the image top to bottom and right to left which is reversed. The axon of the retina ganglion cells exits the retina via the optic nerve. The optic nerve exits the eye from the optic desk. The blind spot is the optic desk area where the optic nerve exits the eye. The axons cross over to the opposite side of the brain. The optic tract is the part of the visual system in the brain that carries retina information from the left side of the eye to the right visual field of the brain and the right optic tract carries retina information to the left visual field of the brain, specifically to the temporal part of the brain. The cells in the suprachiasmatic nucleus are involved in the control of the circadian behavior rhythms related to the light-dark cycle. The lateral geniculate nucleuses in the thalamus receive visual information from the retina and send it to the visual cortex [13, 14]. So, vision is important for all animals and especially for intelligent animals (humans). For the human brain, it is easy to understand what is inside an image because we have a stored dataset about the objects that the image shows; as we know human eyes take one picture every 200 ms of time. For example, if we ask a 5-year-old kid about phone types, he/she probably won't be able to answer us, but if we ask him to point to his/her parents, it will be easy for him/her to do that because he/she has a good dataset about their images.

Concerning computer vision? Computer vision, in short, is the ability to transform or processing images/videos taken from electronic cameras, into a piece of information telling us what is inside that image/video depending on pre-trained images stored in the computer memory. When the camera takes an image/video frame, this image will be inputted to some operations. On the other side, the output will be a result of analyzing this image and detecting what objects are inside it as we can see by looking at Figure 2.1. On the real platform, there are some operations done, such as collecting images, processing these images to be a dataset, making an algorithm for processing date and discovering a result [1].

Everyday computers vision gets improved and enhanced because there are new problems that appear new solutions too, enhanced algorithms, and improved processor. So the computer vision

programmers are working continuously to create a better system, detect more objects, more in speed, more in accuracy and easy to use. For example, nowadays there are monitoring systems with cameras and detectors but without warning system. Programmers are working to make these systems more accurate and ones that detect objects and giving a message that there is something here had been detected [15]. Here we can explain that computer vision science is not correlated with computer science exclusively, but it extends to more branches such as biology, psychology, physics, engineering, mathematics and medicine.

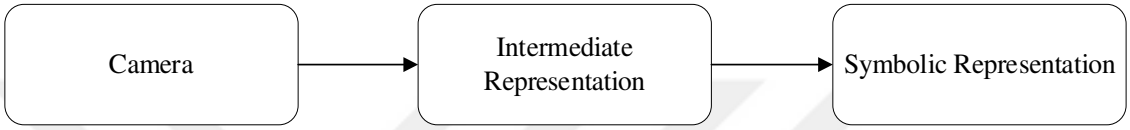

Figure 2.1. Simple representation of computer vision [82]

Generally, to explain the computer vision idea, it was innovated when humans decided to invent machines that can represent the surrounding world with data (image). Here, events that happened in the 19th century, a person named Johann Zahn 1685 invented a simple sketching tool named Camera Obscura (darkroom). It was a box with a pinhole that takes the outside image on the opposite side into the inside room. That was the first attempt to represent the world with data [16]. Figure 2.2 shows the first camera.

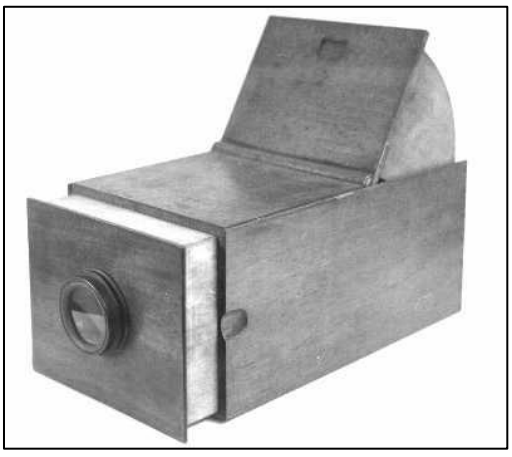

Figure 2.2. Camera obscura [16]

2.1. Computer Vision History

The history of this invention had appeared in the second half of the 19th century; since then, researchers are making studies, researches, and experiments to understand the mechanism of the organism's vision and how they can make a similar experiment on machines to make it see. We made a small research on growing of computer vision and how it has been improved through all years till now.

2.1.1. Fundamental Attempts of Object Representation

The history of this idea had begun many years ago, and exactly in 1959 when Hubel and Wiesel started to study the mechanism of vision in the animal brain (cat) using electrophysiology to analyze it and see how neurons respect the vision process. They stuck some electrodes in the back of a cat's brain, where the primary visual cortex is, and looked at which stimuli make the neurons. They found that these cells respond to the edges when they shed light on the eyes. This experiment was the cornerstone for how brains store objects, by first storing the edges, then by going up to more complexity and more information [2].

After that in 1963, Larry Roberts came with a PhD thesis named "machine perception of the three-dimensional solids" which is widely known as one of the first PhD theses talking about computer vision. The idea was how we can analyze photographs and transform a 2-D image into a 3-D object by computer constructor. By transforming a 2-D into a line drawing then transform line drawing into a 3-D object [3].

In 1966 there was a Massachusetts Institute of Technology (MIT) summer project known as the summer vision project that aimed to use summer workers effectively of a significant part of a visual system. The goal was to divide the problem into parts and process them. This was the first time in which scientists had talked about patterns recognition [4].

After that, and in the 1970s, a scientist named David Marr published a book titled "Vision", in which he wrote his ideas about vision and how we can improve algorithms to get the aim of computer vision aim. He also wrote that if we want to get this aim, we should do several processes, including a "primal sketch" finding the edges, the bars, the ends, the virtual lines, the curves and the boundaries. The next step was what D. Marr called two and a half D sketch, starting to combine the surfaces, the depth, and the layers, of the visual scene. Finally, we got a 3-D object with surfaces, layers, and volumetric shape [5]. This book received good attention from the computer vision branch and scientists.

In the 1970s, computer vision scientists began to think about representing objects with shapes as a way to try to do simple object representation. In 1973 there was a theory of representing objects as geometric pieces connected, and this theory was named “pictorial structure” by Martin A. Fischler and Robert A. Elchlager [6]. The idea of the pictorial theory was to represent objects with pictorial structure. Part of it was to specify and describe the object, depending on it deciding if it is matched or not (detected or not). There was an algorithm that takes the description and the object inside the photograph, making the similarity and giving a decision. Besides, they approved that this object can be recognized. The main aim of this theory was to represent objects in a simple manner [6].

At that time there was another theory named “generalized cylinder” by Brooks and Binford which represents objects with several cylinders. This idea played an important role in computer vision. It is a solid swept and a planer region along an axis, planer region called a cross-section of the GC. This idea got a good spread in computer vision because it can represent the object explicitly and its object-centred coordinates from image data. Most applications used this algorithm like shape recovery, designing a robot vision system, object models, segmentation and detection. Here, we have found that the idea was that object is combined with several figures [7].

In 1987 David Lowe published a paper titled “Three-Dimensional Object Recognition from Single Two-Dimensional Images”. He tried to recognize razors by representing lines and edges as well as constructing them. The computer vision system applied can detect 3-D objects invariant points of view in a grey-scale image. This algorithm is different from other approaches as this algorithm is not needed to reconstruct the depth dimension of the image. Firstly, perceptual organizing to structure the image, reduce the size of the search space using a probabilistic ranking method and match it with the object, and at the end, these special correspondences inputted to projections of three-dimensional models, and matched with the original image [8].

These theories were the fundamentals of representing objects and it was hard to detect objects in this manner, especially human faces, dealing with an image as it is altogether was complex at that time because computers’ speed was slow, there are no more datasets, no GPUs, etc. These theories were the basics of computer vision born.

After that a new idea was invented, it was image segmentation which means dividing an image into a group of pixels into meaningful areas and extracts the pixels belonging to a specific area. That operation called image segmentation and it was a big event in computer vision history, and it was Jianbo Shi and

Jitendra Malik's idea when they published it as a paper named "Normalized Cuts and Image Segmentation". They say that to avoid the problem of grouping segmentation, we can determine a general impression of an image, dealing with an image segmentation as a part of a graph, partitioning the problem, then the normalized cut deals with a total similarity of different groups and the group inside it. They applied their algorithm and there was an enhancement in the results [9].

2.1.2. Viola-Jones Algorithm, 2002

There was another complex thing: face detection. It was a hard thing to detect faces at that time till Paul Viola and Michael Jones published a paper for face detection named "Rapid Object Detection using a Boosted Cascade of Simple Features"; they used simple methods to detect faces. They obtained three contributions for detecting faces [10]:

- Integral images
- Adaboost
- Combining complex classifiers in a cascade structure.

The combination of methods works on a Haar basis (which differs from the previous process depending on the image's intensities), once it computes fewer computations comparing with pixels, and then it can be a base to compute any other part of an image in a constant time. The majority of this approach focuses on several features to determine the detection, and this leads it to be faster than the pixel-based system. So they used three types of features. The first one is a two-rectangle feature that is showing the difference between the sums of pixels inside the same sized and shaped rectangles. The three-rectangle feature calculates the sum of two outside rectangles and subtracted from the one inside or centred. The four-rectangle feature calculates the difference between rectangles on the shape of X [10]. This system was slow in training but it was fast in detection.

2.1.3. SIFT Algorithm

In 2004 David Lowe developed an algorithm known as Scale Invariant Feature Transform (SIFT). The main idea of it was matching the entire object to the target object; the hard work was detecting key points from objects because of the variety of angles, viewpoint, occlusion, lighting and other vectors that can make a difference view of the same object. But his opinion was some features can be detected and can't be changed, and here the task is identifying these features of the object then matching these points with points of the target object [12]. The idea is implemented in four steps: the first one is using Difference Of Gaussian (DOG) for scale-space estimation, then the second step is detecting key points and localize them, the orientation task is a major point depending on the local image hierarchy, and the last step is using descriptor generator to match points of the local image depending on orientation and image gradient magnitude [36].

2.1.4. Histogram of Oriented Gradient (HOG)

A few years after the Viola-jones algorithm was published, a new method for human detection appeared known as a Histogram of oriented gradients for human detection (HOG). It was a paper published by Navneet Dalal and Bill Triggs exactly in 2005. So let's get more information about how this technique works [11].

This technique is using hand-coded features, works as follows:

a) Checking every pixel with its surrounding pixels to show its direction going darker. They reaped this process for every pixel inside the image to get its direction; once we get direction we can replace each pixel with an arrow. These arrows called gradients, and the benefit of these arrows is to show the flow of data from lighter to darker inside an image. So now we checked image pixels and transformed them into arrows named gradients [11].

b) After checking image gradients, the image is subdivided into small quarters known as (cells) exactly 16×16 pixel, checking the gradients inside every cell and replace all pixels with one direction (strongest one) [11].

c) After that, we can combine our neighboring cells to create blocks that give more benefit to normalize our image, this called HOG [11].

d) Now we replaced our image with an array of directions that makes it easier to detect faces inside it [11].

e) Then we can detect faces by checking the similarity of our gradients with a known HOG pattern that is existed inside our database, using conventional Support Vector Machine (SVM) [11].

3. ARTIFICIAL INTELLIGENCE

As a public branch that was invented in 1950's by John McCarthy, when computer science experts tried to invent systems with a thinking memory or clever systems named artificial intelligence [17]. These systems were solving simple or limited problems that do not need more complex algorithms, just a program that solves some computations or applies some rules as chess for example. This type of intelligence known as symbolic artificial intelligence, a simple program that contains rules and data as input and getting answers as output. The public declaration of AI as a human intelligence challenge was in June 1956, when John McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon organized a two-month Dartmouth Summer Symposium on simulating Human Intelligence with Machines [22]. The first AI robot was created during 1954-1961 by George Devol named Unimate #001 [23]. While the first chatbots created was in 1966 by Joseph Weizenbuam [24]. This branch improved for 30 years till in 1980 more branches descended from it in different uses as invitations and applications like Google search engine. One of the more branches that descend from this branch is machine learning [17]. The figure 3.1 shows AI and machine learning as a branch of it. Finally, we should indicate that AI programs can be written in several programming languages like Java, C++, Python, Prolog, Lisp, Matlab, Julia, and others.

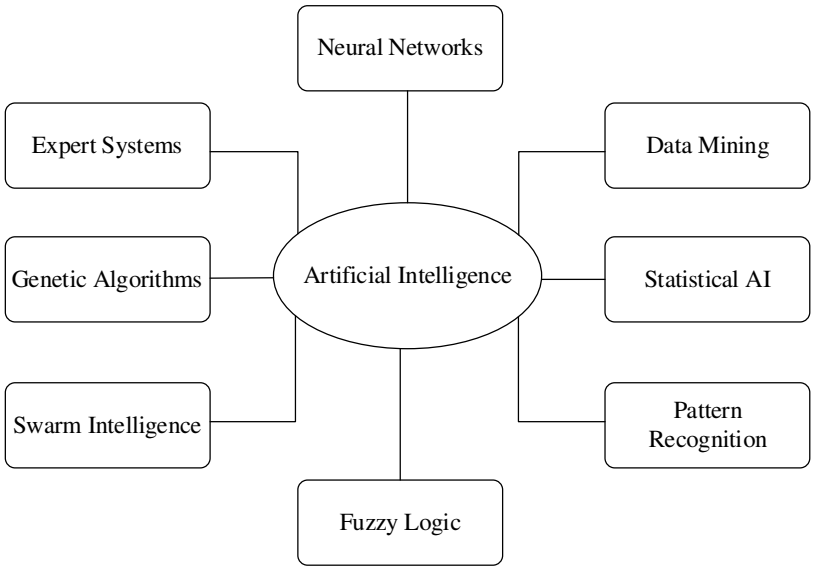

Figure 3.1. AI and its branches [78]

As we can see, machine learning is a branch of AI, but it is less different in its processing, inputs and outputs. The figure 3.2 shows the difference.

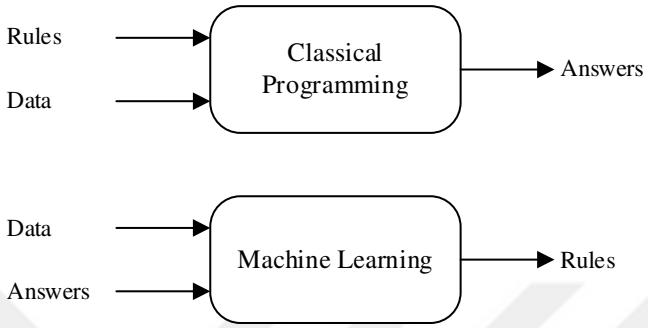

Figure 3.2. The difference between AI and machine learning [17]

3.1. Machine Learning

Machine learning is an algorithm science that makes researchers deal with structured and unstructured data and transform it into information using self-learning algorithms [19]. This branch was invented by Arthur Samuel in 1959; he defined machine learning as a "Field of study that gives computers the ability to learn without being explicitly programmed" [20]. While Tom Mitchell defined it as "A computer program that is said to learn from experience E concerning some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E" [21].

Here, we call it self-learning algorithms, because the main difference between classical AI and machine learning algorithms is that algorithm can improve itself without the involvement of programmers to derive new rules depending on the data. The algorithm itself will be improved between one step to another to make predictions [19]. So machine learning systems are those systems that are trained more than it's being programmed [18]. This branch gives us (practitioners) a big service to deal with data because as we know the computer science branch contains huge data and every day we have new data and new invention [19]. There are a lot of applications in our daily life that use machine learning as using our mobile phone cameras, picture to text transforming applications, voice search on the internet, etc.

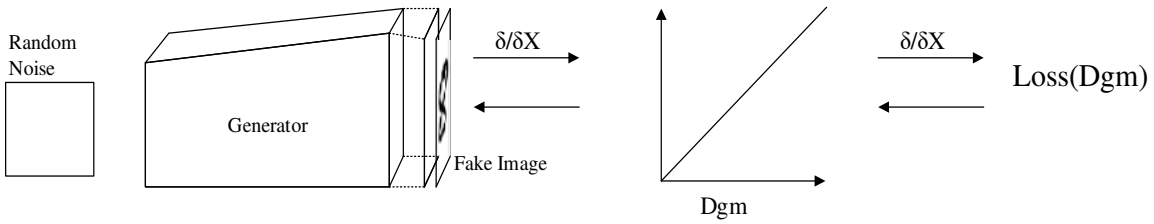

Figure 3.3. Machine learning [18]

We can discover the difference between traditional programming and machine learning, traditional programming is just transforming program sentences to acts like sorting, searching, movement, displaying, etc., is just if → then relationship without making computer calculate or predict results [38]. In machine learning, the story is different as computer works on a dataset and makes a decision depending on an algorithm as we can see in figure 3.3 above. In addition to machine learning, the system updates itself to make decision work better and to improve the system which every time had been in use. [38]. Machine learning must include three major steps to be working normally [38]:

1. Inputted data: maybe image if the system is image detection, or maybe voice if is built for voice recognition, and so on.
2. Predicted output examples: to compare algorithm working output with it.
3. A method to compare the real output with one has been predicted.

3.2. Machine Learning Types

Machine learning algorithms classified into four classes depending on the process and its factors, as the following [17]:

- Supervised learning.
- Unsupervised learning.
- Reinforcement learning.
- Semi-supervised learning.

3.2.1. Supervised Learning

The most used type in machine learning in public and the name came from using labelled data to supervise the output and make the decision as we can see clearly in figure 3.4. Supervised deep learning is used in most applications today, like image classification, object recognition, speech recognition, language translation, text to voice transformation, voice to text transformation, etc. It processes work as learning to map inputted data to get target [17].

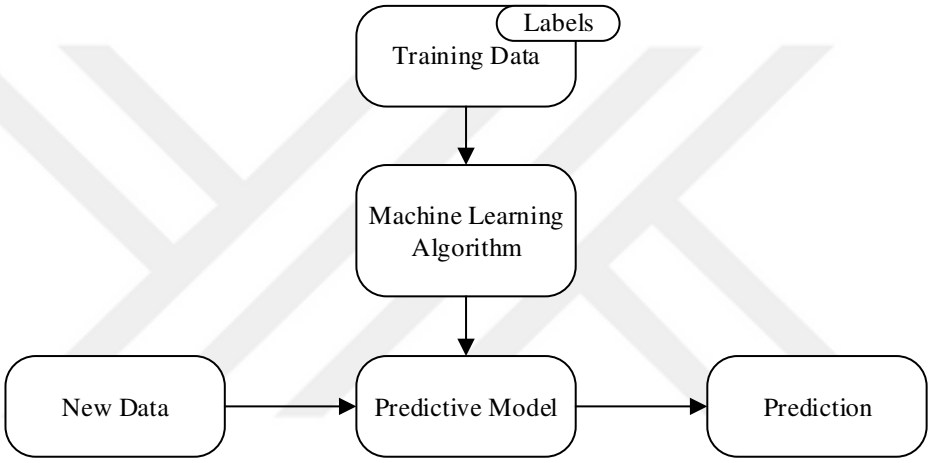

Figure 3.4. Supervised machine learning [34]

Let's take it more clearly. It takes input data and studies them depending on data known as training data to make a decision and predict the output data [25]. So inside an algorithm, there is some data work as a supervisor mapping inputted data until getting the target then output it with a decision tells information about it [27]. Outputted data is named class, every algorithm can be tested and its success rate can be discovered by testing independent data, then checking the true and false classification of an algorithm [26]. An algorithm process will be good and make good decisions whenever it contains a good dataset. For example, a big dataset of images if an algorithm is used for image classification or object detection [27]. Supervised machine learning output may be discretely known as classification task (detecting digits); while the output is a continuous value the system is known as regression analysis. It contains several predictions and continuous output, and the system tries to find a relationship between them that can predict the output [34]. If a supervised learning system has more than one class, then, it will be known as multiclass classification [34].

3.2.2. Reinforcement Machine Learning

This type of machine learning is similar to supervised learning with an advanced property that this type gets feedback after each action or process is done trying to develop the system and improve its decisions and it is called reward signals [34]. Figure 3.5 shows the structure of this type. It is not necessary the reward signal might be a true or false signal, but it is necessary because the system checks how to improve decision making next time when the same circumstances coming up [34]. This operation leads to improving the entire system continuously each time being used. The most popular example of this type of learning is a chess game.

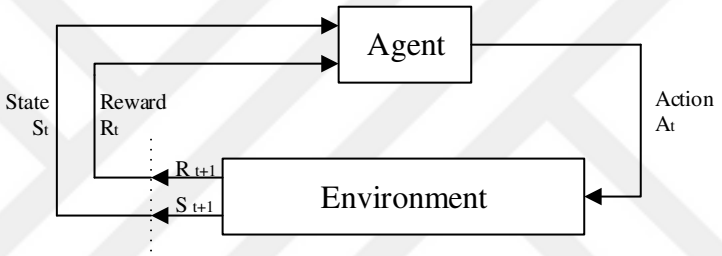

Figure 3.5. Reinforcement machine learning [35]

3.2.3. Unsupervised Machine Learning

This branch of machine learning is called unsupervised learning because it is an unlabeled dataset. This type of learning extracts data properties by structuring data in some figures or groups without having information about the output [34]. Inside this type, we have two types of structures including:

- Cluster groups
- Dimensional reduction groups

Cluster grouping: is a technique used by grouping objects depending on similar properties without knowing the output properties [34]. These structures appear through the analysis of data, meaning that every object in a particular cluster shares similar properties with objects inside the cluster while it is dissimilar

with objects in another cluster. This property helps structure the data in some type makes it easy to know the quantity of each type [34].

Dimensional reduction group: is a technique used for dimensionality reduction, inside object detection we are dealing with data in various and high dimensionality. This technique is used to remove noise inside the data and reduce the dimensionality and make it easy to use and process inside an algorithm leading to making it more productive [34].

3.2.4. Semi-Supervised Machine Learning

The last type of machine learning known as semi-supervised learning which is mixing between supervised and unsupervised machine learning, this type of learning contains labelled and unlabeled data used to train the algorithm [36]. This type of learning is needed when there is a large amount of data that need to be classified. It uses a little labelled data with a large amount of unlabeled data to classify it, for example, internet contents. This type is good to classify because it is not possible to make all internet contents be labelled [36].

3.3. Deep Learning

It was a dream many years ago, inventors for making researches on how to make machines think, but there were a lot of problems at that time like computers speed, slow memories, small datasets, GPUs that were not available, etc. But they were continuing to experiment nowadays artificial intelligence and deep learning which are used in more applications like a deep face in Facebook application, virtual person assistant in Google, Google maps, and more, this improvement done because of the change of circumstances happened, algorithms improved, huge datasets made, CPU's and GPU's with a higher speed invented [17, 39]. **Deep learning** is a modern branch of machine learning that consists of algorithms and datasets, and the model layers learning is named artificial neural network (ANN). The name came from neurons and the name deep came from the number of layers used inside ANN [17, 37]. The basic idea of deep learning is to make computers think like human brains in making decisions depend on a stored dataset [37], the figure 3.6 shows the general structure of deep learning. If we talk about ANN in-depth, several shallow and hidden layers working together to process inputted data to output data and information about it, this type of analyzing data make it easier because it deals with small problems, then it gathers them to produce the main

problem [40]. Firstly, inside ANN, to convert inputted data to meaningful data that is suitable to the system working on. For example, by transforming an image to RGB layers, layers recognize an object features, then compares it with the target object and checks the similarity, and each layer is responsible for a specific property [37]. The advance option deep learning came with is emphasizing successive layers of learning that have a meaningful representation [17].

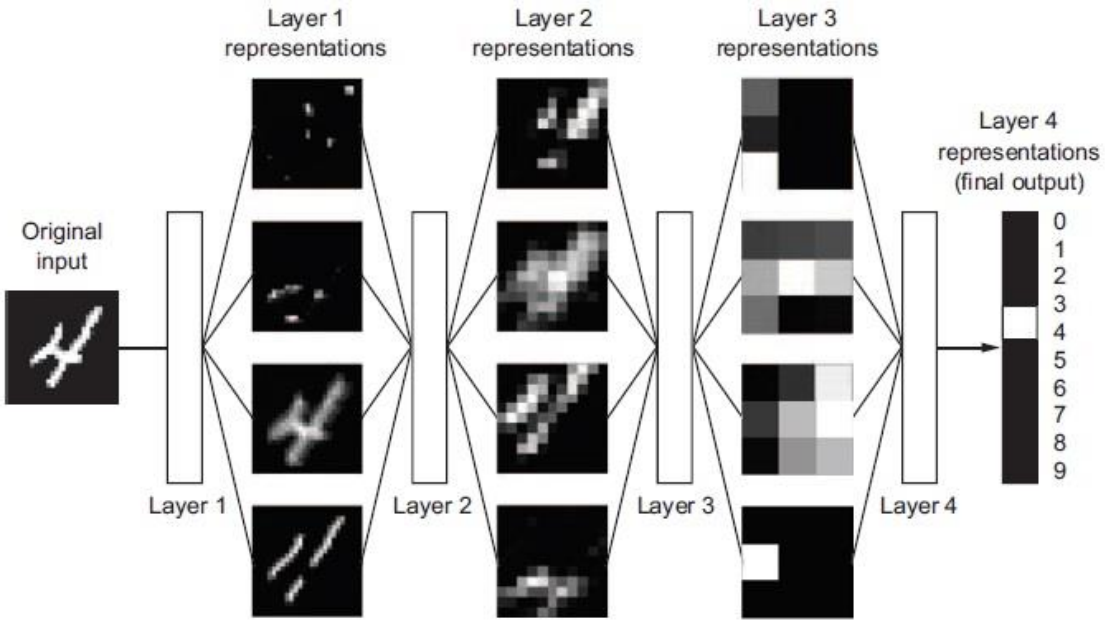

Figure 3.6. Deep learning representations [17]

As we can see above there is a simple representation of how deep learning is working. It divides the picture into parts and layers, then it convolutes it through several layers, each layer is responsible for detecting a property. Then it polls these parts to get the final shape and compare it with a stored dataset. Here it compares with digits and finds the similar one [17]. The deep learning branch is the most famous one used in image/video processing and analyzing techniques [41]. In general, we can divide deep learning into sub-branches [41]:

- Convolutional Neural Network (CNN)
- Recurrent Neural Network (RNN)
- Restricted Boltzmann Machine (RBM)
- Autoencoders
- Extreme Learning Machines (ELM)

3.3.1. Convolutional Neural Network (CNN)

When research or paper or an article is being written about deep learning and processing with layers, the first name coming to reader imagination is CNNs, because it is the most used technique inside this branch [41]. CNN consists of three main steps or layers for processing data, the first one is responsible for convoluting data, the second one is for pooling it, and the last layers are named fully connected layers [41], while the training process going through two processes, forward and backward steps [41]. The forward stage is processing inputted data with weights and loss function then predict the output depending on predictions, backward is taking information from losing function and gradients, updating layers to make it improve, this operation is looping until getting enough time of repetition [41]. The main stages are described as [41]:

Convolution layer: this layer takes an image/frame as input, process it via filters depending on stored weights to merge near pixels to each other depending on its features, by weights technique reducing parameters as possible, locating the object clearly [41].

Pooling layer: using max-pooling to reduce the parameters and minimize feature map measurements [41]. Pooling layers there are three approaches used in varied uses, stochastic pooling, spatial pyramid pooling and def- pooling [41].

Fully connected layers: these layers are the final steps of convolution that contain the max number of parameters, most computational burden through training data [41].

We will explain CNN's in more details, its working, types, and algorithms in the next chapter in a deep way.

3.3.2. Recurrent Neural Network

RNN is different from CNN in which it has more complexity, more parameters compared with CNN, but at the same time, it processes more accurately [41]. RNN deals with sequential data processing that represents time dependency, for example, video, voice, online translating [41]. Also, it has architecture more complex than CNN because of the need for it to process sequential time objects [41].

3.3.3. Restricted Boltzmann Machine

RBM is another type of machine learning that processes miscellaneous and multiple input forms. This technique is found by Minton et al. which is a random NN [41]. In this type, hidden layers and visible layers needed to dipole formation graph [41]. The architecture of it is a double-barreled graph, the hidden layers and visible layers will be correlated with independent relationship [41]. If the hidden layer is H, the visible layer will be V then:

$$P(HV1) = P(H1V1) + (P(H2V1) + \cdots + P(HnV1) \quad (3.1)$$

By this equation and by changing the parameters, the difference between V1 and V2 will be less than it is, and H will give good features of V1 [41]. In RBM it can estimate the corrupted data and save the features, this is done because the data move through more layers of feature extractors [41]. There are several types of RBM like Deep Boltzmann Machine (DBM), Deep Belief Network (DBN), and Deep Energy Model (DEM) [41].

3.3.4. Autoencoders

Autoencoders is a variable type of ANN's, normally used in logical encoding training, the idea in this type of processing is that it learns to re-furnish the input itself and it does not predict output for the current input while training process [41]. Features extracted by restructuring the errors and optimize it and the corresponding code that uses the backpropagation with the same dimensions [41]. We can mention here some of Autoencoders types like **De-noticing autoencoder (DAE), Contractive Autoencoder (CAE)** [41].

3.3.5. Extreme Learning

The most updated branch in machine learning known as **Extreme Learning** posed by Guang-Bin Huang took a wide area of machine learning [41]. The different property that this type has is faster and has generalizing capability than other types, because it is a linear model, meaning that in one step weights from input learns to the output [41]. Inside this type of learning, there is a single solitary layer of masked nodes that takes weights as input in a random form that will not be corrected [41].

3.4. Object Recognition

Object recognition was a property exclusively available for human brains that can recognize faces, things, emotions, cars, etc. in few milliseconds, whether these objects are close or far, different point of view, different colours, and different structure [42]. The researchers tried to make machines also have this property for a long time, but there were always some problems to be found during that operation until object recognition and deep convolutional neural networks appeared. After this technology had appeared, object recognition property appeared for computers too and may have become better than human brains in terms of recognition time [42]. Object recognition is a branch of machine learning that analyzes an image/frame and extracts its features to see it contain a particular object or not depending on some features stored inside the process. These features may be a shape, a colour, a texture, or digital features, or more than one property at the same time, depending on a feature in which inputs are classified by the process [43].

3.4.1. Convolutional Neural Network

CNN One of the most advanced areas in machine learning and image processing, its name has been taken from the analogy of human nerves and how the human brain works from convolutional neural networks [27]. Its history dates back to the 1990s when LeCun et al. used a gradient-based machine learning algorithm on CNN and he got successful outputs of classifying digits righted by hands [45]. As the work of this technique is very similar to the work of the nerves of the human brain in terms of complexity, accuracy, and the speed of treatment and perhaps above the human brain, new techniques in this field have proven superior to the human brain through the speed and accuracy of the human brain in parts of a second [27]. This technique proved its success through more global applications like GoogLeNet, DeepFace, Go game, and others [27]. As we mentioned that CNN's processing consists of convolution, pooling and fully

connected layers that are specialized to deal with different input dimensions with different layers as images, video frames, voices, etc. [44]. There is a function Deep Convolutional Neural Networks (DCNN) uses to process faster than the traditional CNN's more times that is the Called Rectified Linear Unit (RLU) [44]. This success goes to the invention of high-speed CPU's, GPU's, big datasets, improved algorithms as we mentioned latterly, we will explain some of the big datasets as well as improved algorithms used lastly.

We will discuss the commonest architectures and datasets. Architectures are LeNet, Alexnet, ZFNet, and Network in Networks, VGGNet, GoogleNet, Residua Network, Densely Connected Network, FractalNet, Pascal VOC, IMAGNET, IMAGENET ISVRC, and CIFAR [44]. Below are the most important categories of common architectures used in the CNN technique, showed in table 3.1:

Table 3.1. CNN most common architectures [44].

Methods	**LeNET**	**AlexNet**	**Overfeat**	**VGGNet**	**GoogLeNet**	**ResNet**
Top-5 errors	n/a	16.4	14.2	7.4	6.7	5.3
Input size	28×28	227×227	231×231	224×224	224×224	224×224
Number of Convolutional Layers	2	5	5	16	21	50
Size of filter	5	3, 5, 11	3, 7	3	1, 3, 5, 7	1, 3, 7
Feature Maps Number	1,6	3-256	3-1024	3-512	3-1024	3-1024
Stride	1	1, 4	1, 4	1	1, 2	1, 2
Weights Number	26K	2.3M	16M	14.7M	6.0M	23.5M
MACs Number	1.9M	666M	2.67G	15.3G	1.43G	3.86G
FC layers Number	2	3	3	3	1	1
Number of Weights	406K	58.6M	130M	124M	1M	1M
Number of MACs	405K	58.6M	130M	124M	1M	1M
Total Weights	431K	61M	146M	138M	7M	25.5M
Total MACs	2.3M	724M	2.8G	15.5G	1.43G	3.9G

3.4.2. DataSets

Data in general are the steps that every program begins and ends with; it is the input and output of every program, so it is important to understand data types that we will use inside any program we design. Generally, data in Deep Learning are divided into two parts depending on the channels used inside an algorithm, if it is single or multichannel [39]. One of the most important keys that were a reason for improving object detection and the multiple uses of CNN's was the dataset. Dataset is a collection of specific images collected by a researcher or an institute, labelling it, to make it readable by an architecture that uses it. There are two groups of images inside the dataset, one named training data for training the classifier and another named test data [44]. The first step for any practitioner or organization after collecting images for specific object detection labelling these images and processing weights from them [44]. General Datasets are a huge number of images about various objects and things made by an institute, organization or company. Practitioners use them to extract image properties through the object detection process [44]. We will discuss common datasets used inside the Deep Learning field and how the appearance of these datasets impacts the use of CNN's and improves its working and spread. These types are:

- Pascal VOC
- IMAGENET
- IMAGENET LSVRC
- Wordnet

3.4.3. The PASCAL Visual Object Challenge (VOC)

VOC represented a benchmark in the computer vision branch responsible for providing dataset for twenty different objects, which started in 2005 the first annual competition by Mark Everingham, S.M. ALi Eslami, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman [56]. In 2006 it was accepted as a benchmark [56, 57]. Beginning with 2006 to 2012 there was an annual competition responsible for enhancing this benchmark; generally, VOC consists of two parts:

- The public dataset is available for everyone and it is free, images grouped from the Flikcr web site, with software for evaluation.
- There are annual competitiveness and workshop.

In the annual challenge, there are three challenges of classification (an image contains an object or not and if it contains, for which class it belongs to), detection (an image containing object so where is the location of it inside an image), and segmentation (labelling each pixel in an image). This led programmers to be more serious and work more hours to present what is new and most updated [56]. Table 3.2 shows the objects of VOC dataset.

Table 3.2. VOC detecting objects [56].

Vehicles	**Households**	**Animals**	**Others**
Airplane	Bottle	Bird	Person
Bicycle	Chair	Cow	
Boat	Dining table	Cat	
Bus	Potted plant	Horse	
Car	Sofa	Dog	
Motorbike	TV/monitor	Sheep	

The dataset began to grow year after year, and after each annual competition, there were improvements in terms of the dataset until it reached its climax in 2012. The following table 3.3 summarizes the dataset and images used for each object:

Table 3.3. VOC 2012 dataset [58].

	Train		Validation		Train Value	
	Image	**Object**	**Image**	**Object**	**Image**	**Object**
Airplane	327	432	343	433	670	865
Bicycle	268	353	284	358	552	711
Bird	711	560	370	559	765	1119
Boat	260	426	248	424	508	850
Bottle	365	629	341	630	706	1259
Bus	213	292	208	301	421	593
Car	590	1013	571	1004	1161	2017
Cat	539	605	541	612	1080	1217
Chair	566	1178	553	1176	1119	2354
Cow	151	290	152	298	303	588
Dining table	269	304	269	305	538	609
Dog	632	756	654	759	1286	1515
Horse	237	350	245	360	482	710
Motorbike	265	357	261	356	526	713
Person	1994	4194	2093	4372	4087	8566
Potted plant	269	484	258	489	527	973
Sheep	171	400	154	413	325	813
Sofa	257	281	250	285	507	566
Train	273	313	271	315	544	628
TV monitor	290	392	285	392	575	784
Total	5717	13609	5823	13841	11540	27450

The table above shows the big work done by VOC staff, practitioners cooperated to do the best, and make enhancement every day.

3.4.4. Imagenet

The name came from wordnet which was created by George miller in 1990 when he established the first ontological and lexical relationships in natural relationship programming [60]. He organized over 150000 words into 117000 categories called synsets [60]. The idea was first established for English words then transformed into visual recognition. Imagenet came on the same structure of wordnet, but it organizes images instead of words [61]. Imagenet is a large scale space that offers a good resource of good resolution images to large-scale image researches and algorithms also contains training and benchmarks for algorithms [61]. The beginning was in 2009 with 12 object subtrees, explaining 12 objects (mammal, bird, reptile, fish, vehicle, amphibian, tool, geological formation, musical instrument, fruit, furniture, and flower) with 5247 synsets, 3.2 million images, that was clean and the biggest dataset at the time of publishing it [61]. The target was offering 500-1000 clear with good resolution images, downloaded from the internet with reprocessing of some images to be good for the dataset. The process was done using Amazon Mechanical Turk (AMT) services as a platform for global practitioners to do this work [61]. Then year after year they improved the dataset as well as its categories, and the last update of this dataset as we took it from the imagenet web site shows a huge number of images reach 14,197,122, with 21841 synsets, with the most updated categories as shown in table 3.4 [62]:

Table 3.4. IMAGENET dataset statistics [62].

High level category	# synset (subcategories)	Avg # images per synset	Total # images
Amphibian	94	591	56K
Animal	3822	732	2799K
Appliance	51	1164	59K
Bird	856	949	812K
Covering	946	819	774K
Device	2385	675	1610K
Fabric	262	690	181K
Fish	566	494	280K
Flower	462	735	339K
Food	1495	670	1001K
Fruit	309	607	188K
Fungus	303	453	137K
Furniture	187	1043	195K
geological formation	151	838	127K

High level category	# synset (subcategories)	Avg # images per synset	Total # images
Invertebrate	728	573	417K
Mammal	1138	821	934K
musical instrument	157	891	140K
Plant	1666	600	999K
Reptile	268	707	190K
Sport	166	1207	200K
Structure	1239	763	946K
Tool	316	551	174K
Tree	993	568	564K
Utensil	86	912	78K
Vegetable	176	764	135K
Vehicle	481	778	374K
Person	2035	468	952K

The table above shows the last update of IMAGENET LSVRC which is a huge number of synsets of different objects and is available for free for object detection and segmentation practitioners.

3.4.5. SUN Dataset

Practitioners' work on datasets in different branches one of them is the SUN dataset, which was done by Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva and Antonio Torralba. They made a dataset relevant with scenes understanding object categories [64]. The dataset contains 130519 images with 899 categories of different objects [64]. They tried to cover most human scene categories by collecting images for different scene categories with different functionalities, and then they checked how humans can classify scenes. They build a dataset of multiple categories using a kernel combination of many features, and finally, they tested their dataset for scenes detection [64]. The categories were chosen using wordnet vocabularies for most human scenes used. They took 2500 terms from wordnet then chose 899 categories with 130519 images, also labelled with the help of the Amazon Mechanical Turk (AMT) platform [64]. The accuracy is different from one category to another and from 58% to 95% [64].

3.4.6. MS COCO Dataset

A dataset created under the supervision of Microsoft in 2014 containing 91 different objects, with 2.5 million images labelled by pre-instance segmentation [63]. They worked on three topics or problems in researches, detecting non-iconic views, contextual reasoning among objects and accurate 2-dimensional localization of objects. They also worked on detecting multiple objects in one image depending on the context to clarify the ambiguous objects in an image [63]. To collect a large number of images they collected images by new technique with AMT help, labelling images using hierarchical labelling approach [63]. Participants in MS COCO worked on comparing among datasets that are available before MS COCO and they took the benefit of each one. Below is the comparison of MS COCO with other datasets in figures 3.7, 3.8, 3.9, and 3.10 sequentially:

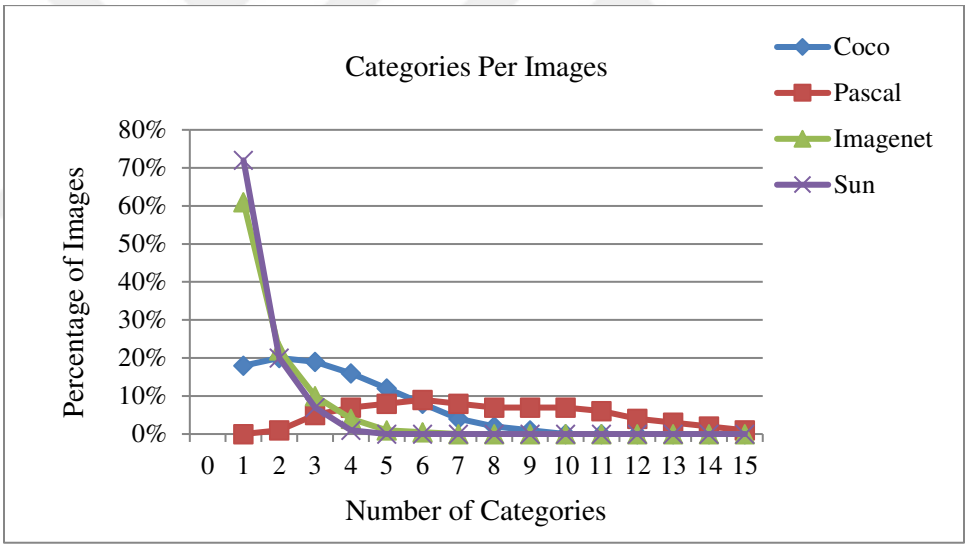

Figure 3.7. Ms coco with other datasets per category [63]

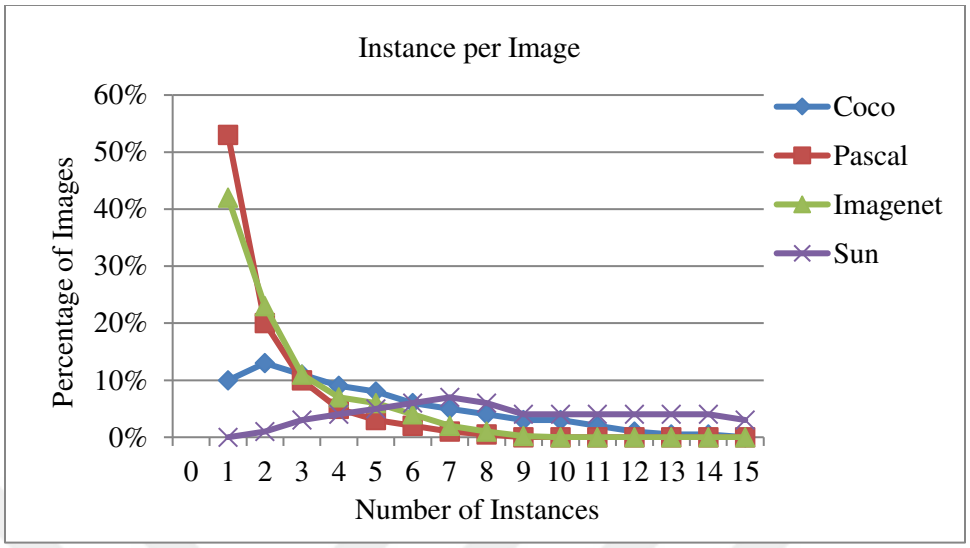

Figure 3.8. Ms coco per instance [63]

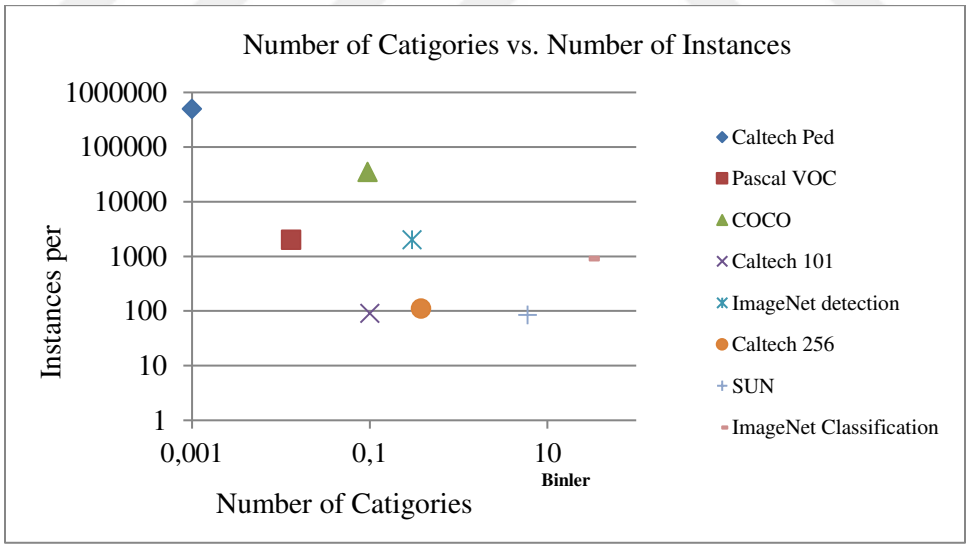

Figure 3.9. Ms coco numbers vs. categories [63]

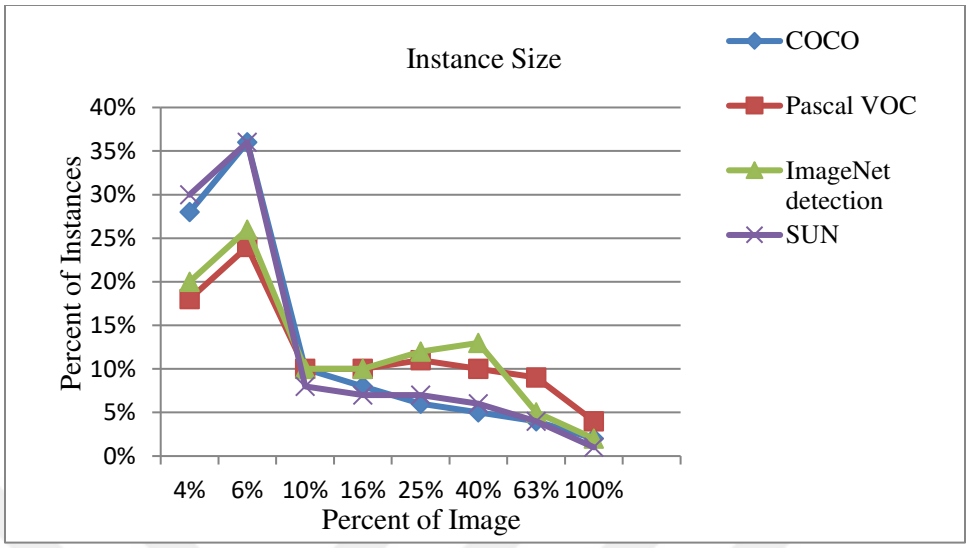

Figure 3.10 Ms coco instances [63]

3.5. Object Classification, Segmentation, and Detection

This branch was invented in the 1990s, but its applications likely spread in the 2000s. That is because in the 1990s we did not have big datasets, good algorithms, good funds specified to do AI studies form companies like google, amazon, Facebook, etc. and good CPUs and GPUs [17]. We found how datasets are made and what the contents of the dataset are; let's take an overview of the most popular algorithms that use these databases to achieve objects detection and classification. The most popular object detection algorithms are:

- R-CNN
- Fast R-CNN
- Faster R-CNN
- Mask R-CNN
- Single Shut Detection
- You Only Look Once

We will discuss them in some details below.

3.5.1. R-CNN

CNN's worked well in the object detection process in the presence of datasets and CPU's/GPU's availability but it was slow and low in MAP value because it takes the whole image and convolutes it pixel by pixel searching for objects. There was an idea to enhance this processing. In 2013 there was an enhancement that made the process of object detection with CNN's faster and more accurate. It is a Region-based Convolutional Neural Network by Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik in 2012 when they published their idea in a paper named "Region-based Convolutional Networks for Accurate Object Detection and Segmentation" [65]. R-CNN was CNN with Region proposed with more speed and improvement of mAP about 50% comparing to algorithms used at that time because the traditional algorithms used low-level images while R-CNN was solving the problem of using high capacity images and they solved the problem of detecting regions of objects inside an image and focus on it for object detection [65]. They used three stages for object detection inside an image, firstly the process finds proposal regions that could find elected objects (they used selective search for predicting RPs). Secondly, an image with its Proposed regions goes as an input to a convolutional neural network to extract its features, and the features are inputted to SVM to detect the class having the same features or similar nearly to the predicted object. Finally, classify the object to a specific class [65]. The vary of R-CNN is that when you use it with a big number of classes, the processing time will be the same as it with low memory consumption compared with other algorithms [65]. Another property is that in R-CNN MAP has a high degree that reaches up to 59% [65]. The most important in R-CNN was that the first trying for the idea if the regional proposed system in object detection world, but still having some problems in training time and cost. It is also slow and there was overlapping between RPs, because of that there were other versions of R-CNN like Fast R-CNN, Faster R-CNN, and Mask R-CNN [30].

We mentioned above that RPs are selected with a process named **selective search**, so the selective search is the technique used to limit the area that should be processed by CNN, by bounding a box around suspected objects detected inside an image as a manner to keep focusing on positive areas inside an image. That process which is called selective search is done by merging cells depending on colour, texture, intensity, or near to each other. That process transforms the single picture into several boxes containing suspected objects, and then these suspected objects go as input to CNN to determine exactly what is inside each box [66].

3.5.2. Fast R-CNN

In R-CNN we found that it works slowly because it carries out a convent forward pass for every predicted object, without sharing computation of predicted objects. In Fast R-CNN it is different because it shares the convolution features through all regions in the convolution stage that will speed up the process. The work of sharing features is computing the convolutional feature Map of the inputted image as a whole then working on classifying the predicted objects [67]. SPPnet idea availability made R-CNN faster in test time as it reached 100× and also made training timeless reached 3× [67]. We can find the enhancement in Fast R-CNN in multiple places like:

- The mAP is higher than R-CNN, the meaning is more accurate
- The training stage is single and uses multitask loss
- There is no need for disk storage to store feature catching

The difference in architecture is available also, in Fast R-CNN it takes the image as a whole for network processing and pooling in multiple layers to extract convolution feature mAP [67]. After that, each predicted object ROI is extracted with feature vectors for each one, each feature vector then inputted to fully connected layers [67]. Then, the output will be softmax probability estimates, and then extract four values for each object to find the bounding box for each object [67]. That was an enhancement of R-CNN, but we will see more enhancements in Faster R-CNN in the next part.

3.5.3. Faster R-CNN

Fast R-CNN was near to real-time object detection but is still slow, and the reason was the time spent in RPs. In Faster R-CNN they worked on reducing this time and found that the most time-consuming was in selective search for finding RPs, as it takes two seconds for each image on CPU. So it is necessary to find a way to make it faster, and the first solution is using GPUs, but it is still not a technical solution [68]. Ross Girshick and his team came up with an idea to reduce the RPs detection time with another version of R-CNN named “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In Faster R-CNN they found another process for finding RPs known as Region Proposal Network (RPN), which is a DCNN fully connected layers that work on finding RPs in less time that reaches 10ms for each image [68]. RPN is designed for RP prediction inefficient ways with various scales and sizes, designed as

an FCN, sharing computation with Fast R-CNN [68]. Making proposed Regions with Anchors, Anchor is the Centre of the movement to produce proposed regions in multiple sizes and scales, this step is only for detecting regions that predicted and contains an object and trying to calculate anchor box dimensions [68]. This improvement leads to Faster R-CNN to be the faster version between R-CNN versions as we can see the difference in the below table 3.5:

Table 3.5. R-CNN versions comparison [69]

	R-CNN	**Fast R-CNN**	**Faster R-CNN**
Test time per image	50 second	2 second	0.2 second
Speed up	1×	25×	250×
mAP (voc 2007)	66.0	66.9	66.9

As the table above shows, the enhancement in detection speed time and how RPN enhanced the speed.

3.5.4. YOLO

A new object detection approach had been invented after multiple improvements of object detection that uses bounding boxes of object detection (as we mentioned in R-CNN) instead of classifying the whole image pixels. Also, it is real-time object detection because the speed of recognition improved that the previous approaches were slower than YOLO so it cannot detect real-time videos with a good speed while YOLO did that, in YOLO algorithm takes a single look at the image and extract the objects that made YOLO much faster but that affects the security too. The table 3.6 below defines object detection approaches according to its born with its properties [70].

Table 3.6 Comparing YOLO with other object detection approaches [70].

Detection frameworks	**Train**	**mAP**	**fps**
Faster R-CNN	VOC 2007+2012	70.0	0.5
Faster R-CNN VGG16	VOC 2007+2012	73.2	7
Faster R-CNN Resnet	VOC 2007+2012	76.4	5
YOLO	VOC 2007+2012	63.4	45
YOLOv2 288×288	VOC 2007+2012	69.0	91
YOLOv2 352×352	VOC 2007+2012	73.7	81
YOLOv2 416×416	VOC 2007+2012	76.8	67
YOLOv2 480×480	VOC 2007+2012	77.8	59
YOLOv2 544×544	VOC 2007+2012	78.6	40

YOLO idea is cleverer than previous approaches like DPM because it takes region proposals inside an image with multiple classes probabilities and processes them simultaneously to get faster, (this is the idea from the name YOU ONLY LOOK ONCE). It takes several steps to do the best [70].

1. Dividing an image into a grid of 13×13 cells
2. Using bounding boxes with probable classes to predict objects
3. Using a single CNN to optimize the RP's
4. Loss function
5. Non-maximal suppression and fast processing get to 45 fps

YOLO's first step is dividing an image into a grid with 13×13 cells; a cell that the object centred in it will be responsible for detecting that object. This cell will predict a bounding five cells.

So we have 13×13=169 cells, and every cell predicting its five bounding cells that lead to having 845 bounding cells. That is a big number of cells that need big time to process, but in YOLO it takes just that boxes having 30% or more of confidentiality (programmer can change this value depending on the accuracy needed). As shown in figure 3.11 and 3.12:

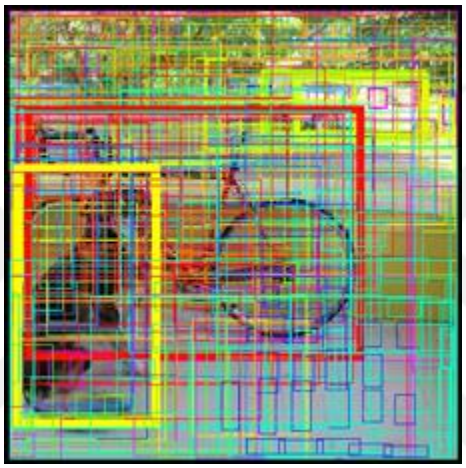

Figure 3.11. All predicted regions in the first step [71]

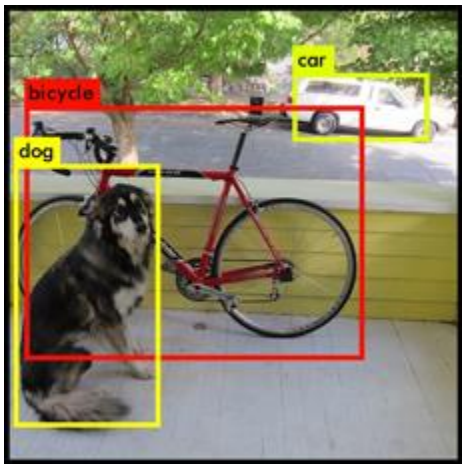

Figure 3.12. Regions remained that contains objects [71]

As we see above from 845 boxes we only detect three boxes that pass the percentage. This predicted boundary gives what size and form of the object are in addition to giving an accurate bounded box (object size). Generally, it defines objects through a form that [71]:

$$pr(object) * iou_{pred}^{truth} \quad (3.2)$$

Where PR denotes the proposal region and IOU denotes the intersection of the union. This formula gives the decision if there is an object or not, if there is no predicted object confidence score will be zero. If there is, it should take the IOU between our predicted box with a ground truth box [70, 71, and 81].

We mentioned that each cell should predict five bounded cells, let them be: x, y, w, h and its. Each cell must predict a probably classes if it predicts an object, Pr(classi|object). After this step we have several predicted classes for each grid of cells, we should multiply probably classes with a specific box that can give a specific class confidence score for each box [70].

$$\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IOU}_{\text{Pred}}^{\text{Truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{Pred}}^{\text{Truth}} \quad (3.3)$$

This step defines each box with a set of classes that are belonging to; this leads to limiting the area of detecting inside an image and also predicts a set of classes for each box [70, 81].

YOLO has a simple architecture, is represented with a simple convolutional network as below in table 3.7:

Table 3.7. YOLO architecture [70].

Layer	Kernel	Stride	Output shape
Input			(416, 416, 3)
Convolution	3×3	1	(416, 416, 16)
Max-pooling	2×2	2	(208, 208, 16)
Convolution	3×3	1	(208, 208, 32)
Max-pooling	2×2	2	(104, 104, 32)
Convolution	3×3	1	(104, 104, 64)
Max-pooling	2×2	2	(52, 52, 64)
Convolution	3×3	1	(52, 52, 128)
Max-pooling	2×2	2	(26, 26, 128)
Convolution	3×3	2	(26, 26, 256)
Max-pooling	2×2	1	(13, 13, 256)
Convolution	3×3	2	(13, 13, 512)
Max-pooling	2×2	1	(13, 13, 512)
Convolution	3×3	1	(13, 13, 1024)
Convolution	3×3	1	(13, 13, 1024)
Convolution	1×1	1	(13, 13, 125)

From the table above we can note that YOLO is using a standard type of layers [70]:

6. Starting with a convolutional layer with 3×3 kernel
7. Followed by a max-pooling layer with 2×2 kernel

These steps were repeated more times to get a result with 125 channels for each box (we have five boundaries, each boundary represented by 25 data members).

In this state we have an input of an image with a 416×416 pixel, going through the conventional network layers, and output will be 13×13×125 tensors representing boxes that entered [70].

After that, we should process only boxes that have more than 30% and drop the others. Figure 3.13 shows the structure of YOLO and it's filters in details.

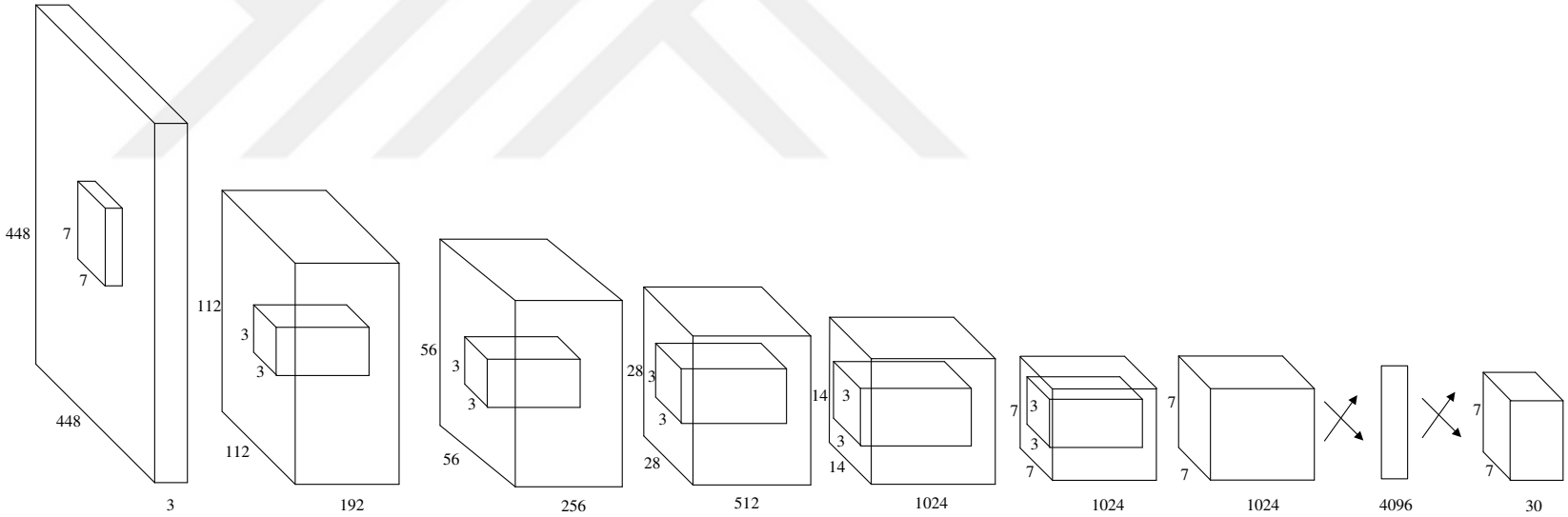

Figure 3.13. YOLO layers [70]

The figure above shows the YOLO CNN detection process that takes 24 convolutional layers followed by two fully connected layers. This system takes a 448×448 image as an input, analyzing it; the output will be 7×7×30 tensors of prediction [70].

After getting predicted boxes it must choose only boxes that contain real objects and similar to ground truth images. In this stage, YOLO uses the loss function to select only boxes that are responsible for an object (boxes have big IOU). There are three steps for localization [72].

8. Classification loss
9. Localization loss
10. Confidence loss

3.5.5. YOLO is Different

After we described how YOLO is working, we noticed that it is different from the previous object detection approaches. This differentiation is represented by [72]:

1. YOLO is faster than all previous approaches that make it able to detect objects inside a video or camera recorder.
2. The accuracy is better because YOLO uses predictions from a single network.
3. YOLO is more generalized from previous approaches
4. RP made YOLO focus on particular parts that predicted containing objects that made it faster.

YOLO, like any other applications, has updated versions such as fast YOLO, YOLOv2 (faster YOLO), tiny YOLO, and YOLOv3

Fast YOLO is a version of YOLO using 9 convolutional layers instead of 24.

3.5.6. YOLO v2

Is a version of YOLO with new improvements (faster, more accurate, etc.) It uses the SSD technique to make detection more stable. We can see the improvement of YOLOv2 as below:

- Accuracy improvement: adding batch convolutional layers for normalization leads to avoiding needing for dropouts increasing mAP 2% [72].
- High-resolution classifier: YOLOv2 differs from YOLO in that it takes an image of 224×224 for training classifier then it returns to 448×448 this step increases mAP by 4% by making detector training easier [72].
- Convolutional with anchor boxes: YOLOv2 takes five anchor boxes instead of five predicting boundaries, and by predicting offset for anchor boxes it can predict a diversity of predictions. This leads it to be more stable [72].
- Dimension Clusters: This property is that putting top-k boundary boxes that contain top clusters of detecting used to detect objects [72]. For example, if the video is recorded in a garden or public place that the general objects are humans, it takes a human object as a top-k cluster.
- Direct location prediction: YOLO predicts offset to the anchors' parameters randomly, for example (tx, ty, tw, th, and to). If it predicts in the right manner, it is ok. If not, it will repeat the prediction.
- Darknet: YOLOv2 replaced CNN with darknet that requires 5.58 billion operations while CNN is requiring 30.69 billion operations this step leads YOLOv2 to be faster and more accurate.

3.5.7. YOLO v3

At the end of our research, we will talk about the last version of YOLO. It is YOLOV3 that is much faster and accurate than the other versions. In this version, we will find small changes, but it makes the detection operation more accurate [73]. Firstly, the bounding boxes are the same as versions before, using anchors for making bounding boxes. The decision of having an object inside the RP or not is taken depending on assigning a bounding box for each object, and if not assigned, it will be incurred [73]. In the prediction of class, they used independent logistic classification instead of softmax prediction that makes it faster than previous versions [73]. In the state of prediction, they used three different scales of boxes and tried on the COCO dataset with the Darknet platform. The below table 3.8 shows the architecture of YOLOV3 with Darknet-53:

Table 3.8. YOLOV3 architecture [73].

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

This mixed network between YOLOV2 and Darknet-19 is worked better than the previous in efficiency and speed, with darknet-53 YOLOV3 made the highest measure float point per second; leads to networking structure will have better utilization on GPU [73].

4. EXPERIMENTAL WORKS

Object detection is a very big branch that as we have explained in our research above there are too many methods and algorithms researchers invented for object detection. There are updates every day. Also, maybe after we made our research there is another algorithm faster and more accurate than YOLOv3 or maybe another version of YOLO will appeared. This is the normal way in computer science, everything is changed every day, and some will appear, while others get disappeared. In this chapter, we will explain how YOLOv3 can be installed on a normal PC with normal characteristics, what is the environment that YOLO needs to be working, what are the factors which affect YOLO, what is made YOLO better, faster, and what makes it slow, and how it is working with custom object detection. We will talk about our experiment, and how we took a gun named AKM-47 as an example of gun detection. We will explain our experiment in a flowchart at first to be clear what we are doing through the steps, figure 4.1 shows the flowchart of our experiment:

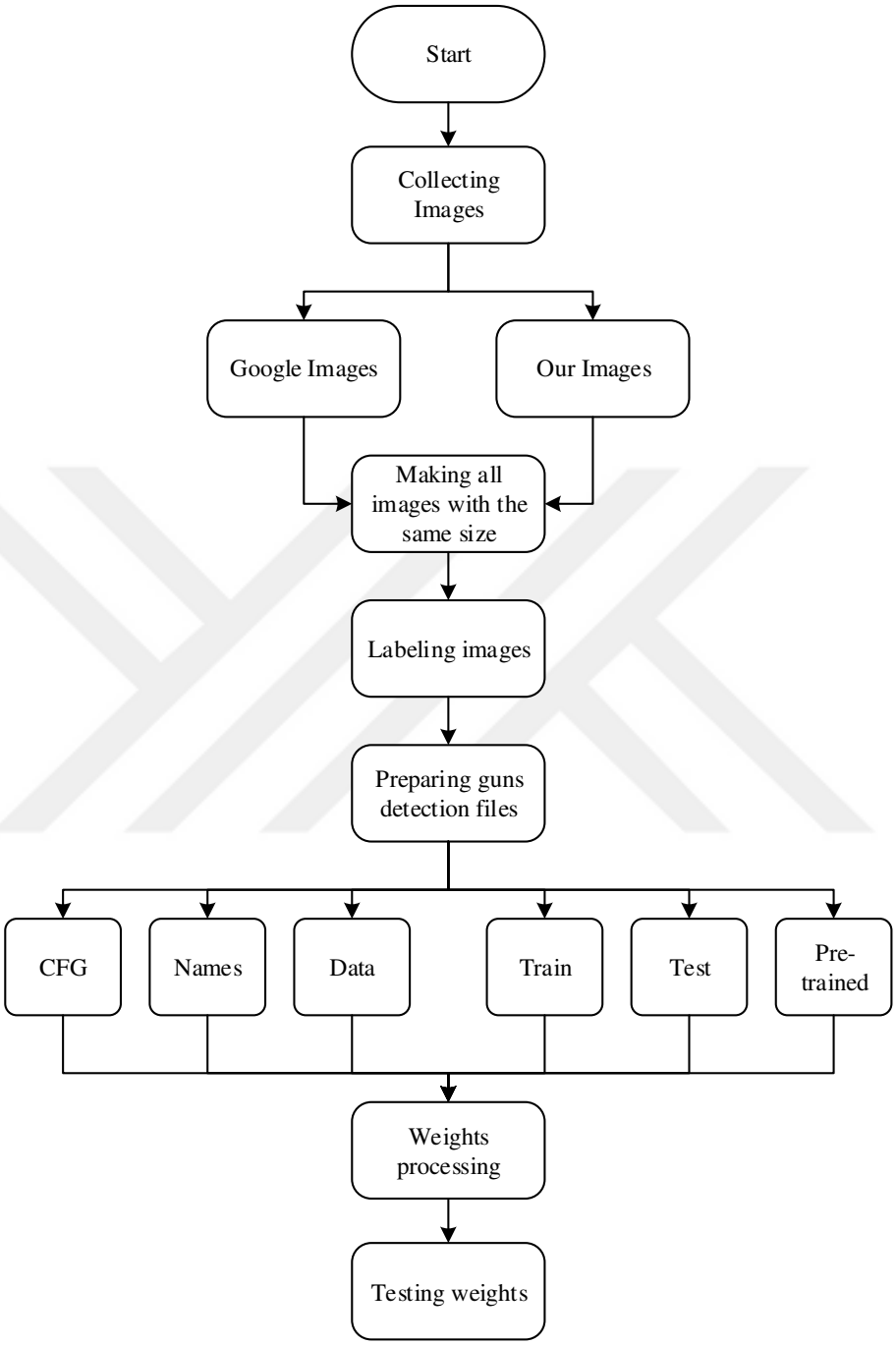

Figure 4.1. Guns detection flowchart

4.1. Installing Tools Supporting YOLO on Windows

Windows is a good and common operating system; we choose it to apply our idea. The idea is going to make a system able to detect guns when it appears inside an image or video, because of that we should prepare it to make YOLO working on it. We need some tools to be adaptive with the YOLO algorithm, as:

1- Git tool: which is an open-source tool designed to help and handle small and big projects [74].
2- Cygwin: This is a tool providing an environment similar to UNIX but on windows. This tool provides some functions that we need to build a Darknet environment like (make, ssh, ssl, gcc-core, vim editor) [75].
3- Installing Darknet: this is an open-source framework that supports neural networks and YOLO as a part of NN's, written in CUDA and C. supporting CPU and GPU too [76].

Now the environment is ready to install the YOLO algorithm.

4.2. Installing YOLO Algorithm

First of all, we must download YOLO pre-trained weights that weights are provided by YOLO. If the installation completed successfully, we should test it before going to build our custom detection system. The most popular picture is below to make sure that the system is installed successfully, as it shown in figure 4.2:

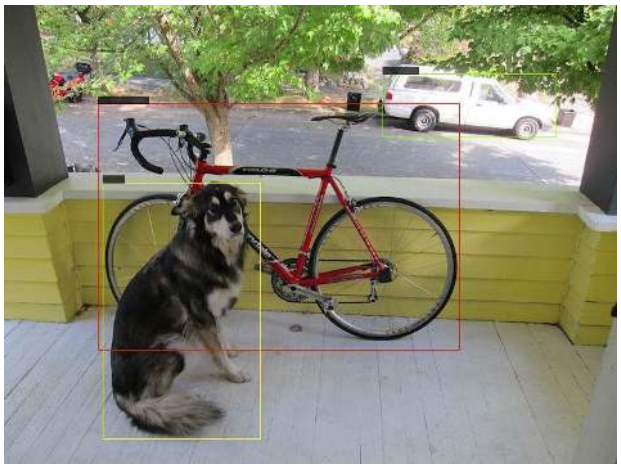

Figure 4.2. YOLO detection [71]

YOLO is working now, the next steps we will build our specific detection system (AKM-47).

4.3. Collecting Images for AKM-47

The most important step when we decide to build a detection system on a specific object is collecting enough images about that object because it is the basic stone of building that system. In our experiment, we searched for images of AKM-47 on Google and we collected 100 different images, but this number of pictures is not enough for our system because of the bad resolution and the similarity. We decided to make our dataset by hand; we took 825 images to the object and prepare it to be ready to be a dataset. As shown in figure 4.3:

Figure 4.3. Akm-47 dataset

We depended on the factors of resolution and a variety of angles in collecting and taking images, to make our dataset includes most of the object angles and position sides. After collecting we need to resize all the images in the same size to make it through the processing. All images resized to 180×180 pixels, the size we have chosen is to make the processor faster through the process of the weights. For resizing images to the specific size we used paint application to prepare it. All images have then been collected in a folder named images to make it ready for the next step.

4.4. Labeling Images

After collecting a dataset for AKM-47 in a specific folder, we labelled all the images. The labelling process means making a txt file for every image, containing the height, width, X, Y for the object, maybe it contains more than one object. The text file tells about the object and its position inside an image and the height, width, x-axis and y-axis. We used a labelling tool showed in figure 4.4. Which is a specific python function, is made for labelling images and is widely used. The function works as shown below:

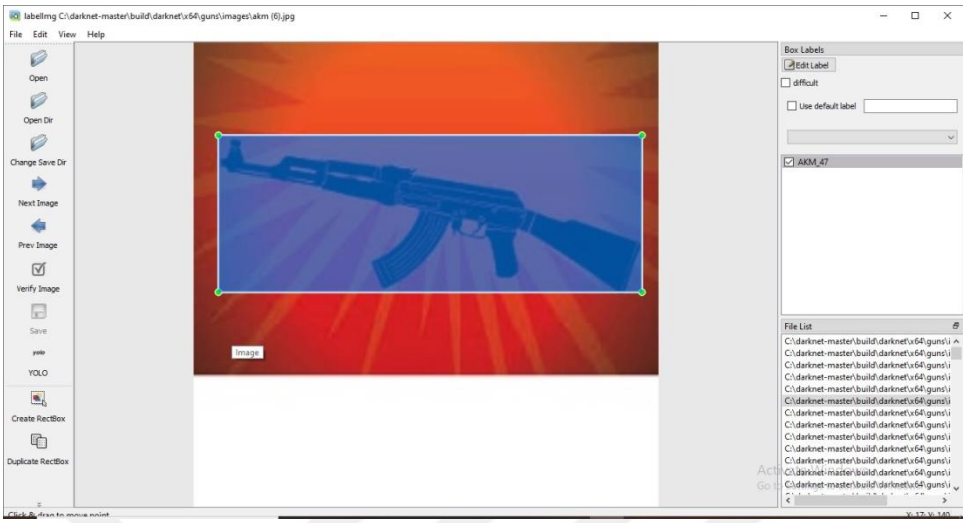

Figure 4.4. Labeling tool

This function is working normally, but sometimes it takes several classes more than your classes and makes mistakes through weights processing. For that reason, the practitioner should note the number of classes appeared in txt files and make sure that it contains the same number of classes that he works on, as we see in figure 4.5:

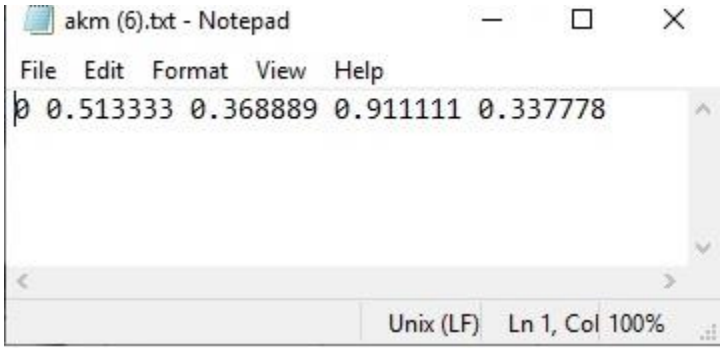

Figure 4.5. Labelling txt file

The first number (0) refers to the number of classes, here, we have a single class for that reason the number is zero, if we have two classes the number will be one and so on. The other numbers are for the x-

axis, y-axis, height, and width of the object inside an image. Txt files must be the same as image files that are referring to, and in the same folder with images.

4.5. Preparing YOLO Custom Detection Files

After collecting and labelling all images for the object, we must provide files that YOLO needs to process weights able to detect the object all files are txt type below we explain every one clearly:

4.5.1. Configuration File

The main and most important file is the one which contains the steps of processing weights step by step and the information and number which we give inside this file must be perfectly accurate because any change to these numbers leads to giving an error or making problems through the detection. We will explain the most important values that affect the process below and the meaning of every value:

5. Batch: means the number of images and label files it takes in iteration.
6. Subdivision: means the number of blocks divided by the batch.
7. Max-batches: means the number of iterations.
8. Filters: means the number of filters inside the algorithm, depending on the classes.
9. Classes: means the number of classes you designed the system for.

4.5.2. Names File

This file contains the names of objects we want to detect to make the system point to each object with its name. For example, our working it contains AKM-47. If the system is designed to detect more than one class, every class must be written in a single line...

4.5.3. Data File

This file contains the number of classes we want to detect, the full path of training and test files (by default these files we put near to the system), and where we must save our new weights (by default we save the weights in the backup folder inside the system).

4.5.4. Training File

This file contains the full path of 70% of the dataset, these images will be used for weights processing. The percentage is prepared by the practitioners; in our experiment, we, put 70% of images to train the weights. As we can see in figure 4.6:

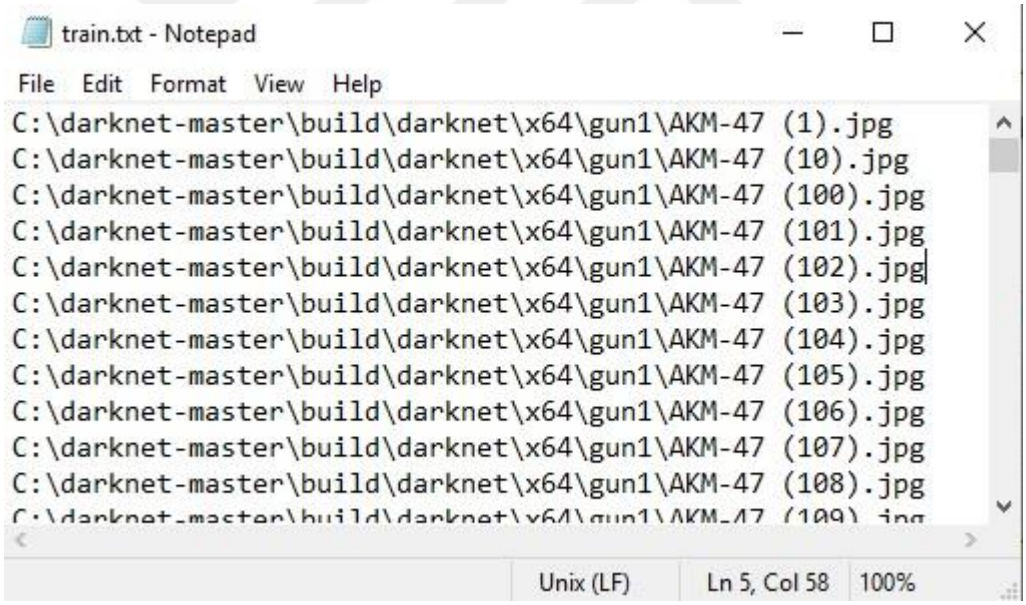

Figure 4.6. Train file for YOLO

4.5.5. Test File

This file is the same as the training file but it contains 30%, of the remaining images as well as the full path of images which are also used for weights processing.

Train and test files must be available near the system folder.

4.5.6. Pre-Trained Weights

After we have made files, YOLO needs them to prepare custom object weights we should download pre-trained weights to help the processor build weights that can detect our custom object, this file is called pre-trained YOLO weight. This file is provided by YOLO to help practitioners build their object detection system. It is available on the same website with YOLO for weights processing, its name is darknet53.conv.74 (155 MB).

These files should be prepared depending on the number of classes we want to detect.

4.6. Weights Processing

All the steps and what we did are just preparing the system to process weights for our specific object detection (AKM-47), we have currently prepared everything for weights processing, the last step is the process and it will be like the figure 4.7:

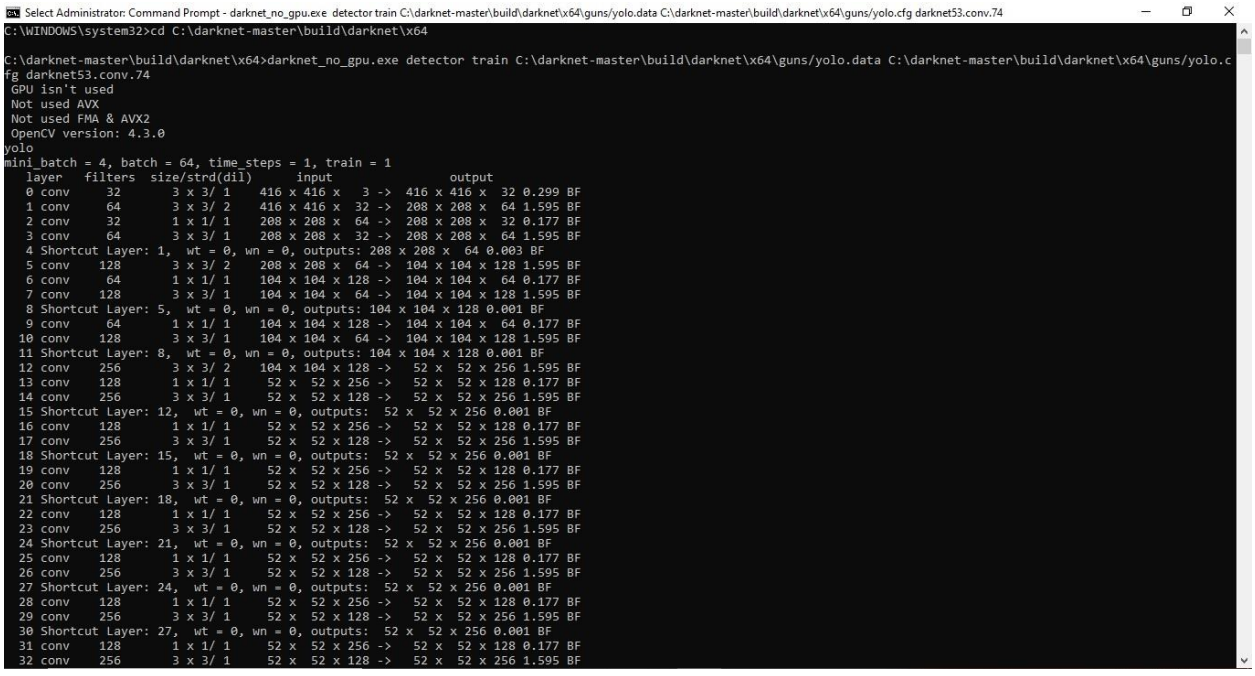

Figure 4.7. Weights training

Through the command or shell process, the instructions are given to the processor as shown below:

(Darknet_no_gpu.exe detector train C:\darknet-master\guns.data C:\darknet-master\guns.cfg C:\darknet-master\darknet53.conv.74)

The instruction takes the data file we prepared with a cfg file for custom detection with the pre-trained weights to process new weights depending on the files we gave.

This step will take a long time depending on the CPU it works on, and if GPU is available or not. It took 14 days if we are using CPU only, as we did in this project. But if there is a GPU the process will finish faster, maybe in 24 hours.

The most important step is loss function and monitoring how it is working because the detection depends on it. As shown in figure 4.8:

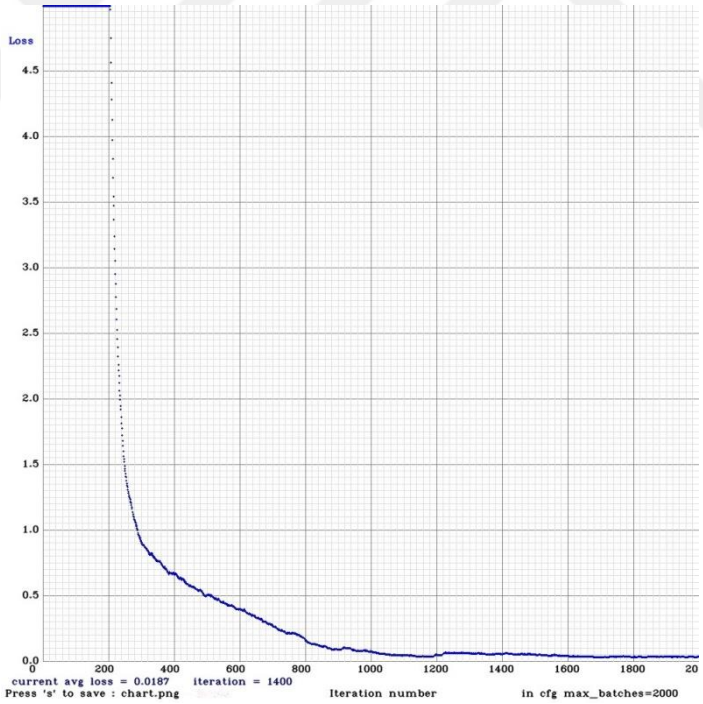

Figure 4.8. Guns loss function

As we see the figure above shows losing function is going to lose the overloaded pixels and detect the specific object. This step takes time depending on PC properties we used for processing, takes from hours to days.

The final step is testing our work by using our weights instead of YOLO weights and test if it is detecting or not. We will give the same instruction we used when we tested YOLO detection but instead of YOLO pre-trained weights, we will use weights we processed step before. As stated below:

Darknet_no_gpu.exe detector demo C:\darknet-master\guns.data C:\darknet-master\guns.cfg C:\darknet-master\guns_1000.weights

This instruction will open the camera and begin to detect objects we designed for it, as we can see in figure 4.9:

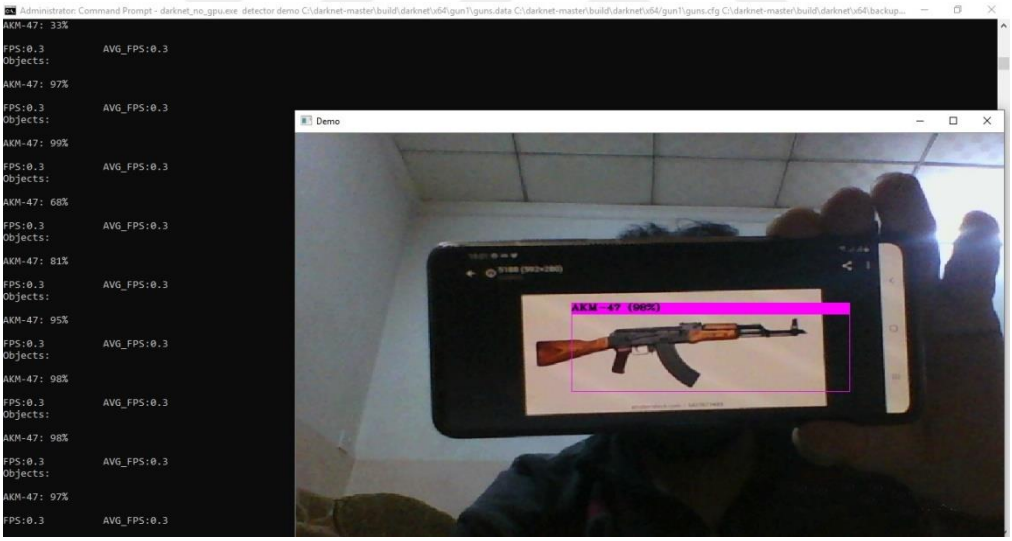

Figure 4.9. Testing Akm-47 detection in real video

This screen was taken for real-time detection of AKM-47, we can notice that it reads 0.3 frames per second, the system is slow but the detection is going in a good way from 68% to 98% detection.

We can perform the same on a picture as we can see in figure 4.10:

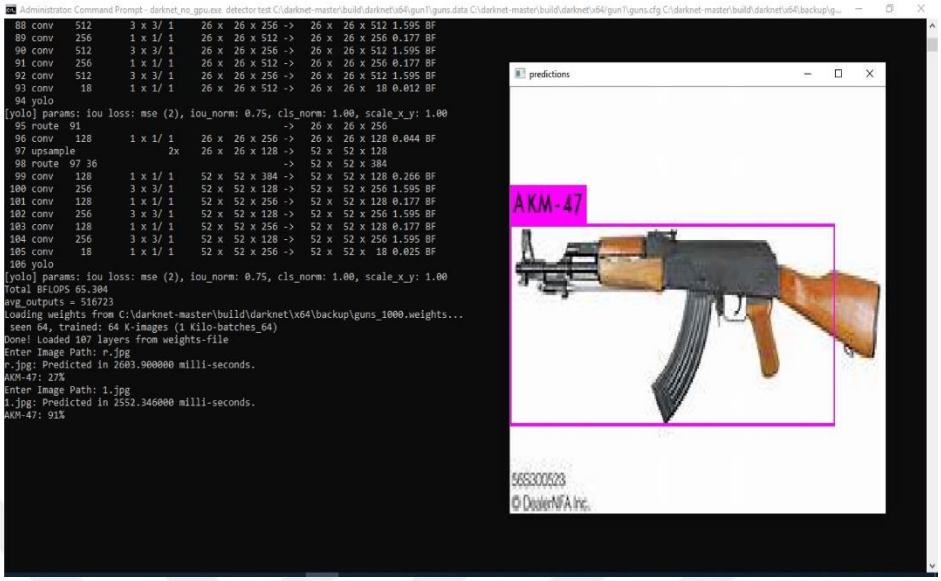

Figure 4.10. Akm-47 detection

As we note above the system detects AKM-47 in 1.5 seconds with accuracy reached 93%. Accuracy and speed depend on dataset and CPU, GPU.

5. CONCLUSIONS

The modern world is led by modern and technological systems using Artificial Intelligence and machine learning, because of that we can notice that several big institutions, universities, and research centre, and military institutions give big care and funds to this branch. Object detection with neural networks works very well and the systems which have been built with these technologies are good and smart systems, we can see this technology in our daily life in hospitals, universities, city streets, or inside our homes, and in the branch of monitoring and surveillance systems in government and military institutions, we can make everything by this technology. The idea of building a smart system for monitoring and detecting guns through monitoring cameras came from the accidents which had happened all around the world from terrorist attacks, and most of them can be controlled if we have a smart detection system connected to the security institutions. Our idea is to build a system that can detect guns (AKM-47) and refer to it; the system must be smart, fast, accurate and simple.

YOLO detection algorithm that we have chosen due to the speed of detection, this algorithm provides, YOLOv3 were the last version of YOLO and were faster than the others. We decided to build our idea on the windows operating system because it is commonly and widely used. The type of gun (AKM-47) that we have chosen because most terrorist operations are done by this type of guns. Collecting images was the hardest step and it takes a lot of time to prepare it and resize it to be ready to use as a dataset. A laptop with normal properties was another dilemma (which does not have a GPU). Installing Darknet and its tools take much time because of the big number of versions available and not all versions are compatible with the environment. Finally, the system was built, then we began our process for building weights, and as we mentioned we did not had a GPU that makes it take too much time to process the weights (near 14 days of the continuous process). The weights are processed then we tested it, the results were good with an accuracy of 93% but it takes near to 2.5 seconds for each image, and it takes 0.3 frames per second and this is slow processing but on a computer with good properties will be fine, needs a GPU 2 GB with a good processor. Our experiment is a good idea if taken by institutions or security companies, with a group of programmers, good computers, and technician persons. Our system is working perfectly if guns are discovered by monitoring cameras, but if a person is hiding his gun under clothes or inside a bag, the system can never detect it.

We hope soon we can make an advanced system that contains more guns objects with good computers and a bigger dataset, and offer it to all military and public areas to control it from any terrorist attacks. Another idea is to make an analyzer before the YOLO process that can analyze object properties to make YOLO more accurate if the system contains more than one object to be detected.

There is another good idea for building a smart security system if it gets good care and support, the idea is connecting an X-ray to scan places and transfer the video to a guns detection system, in this manner the security will be 100% and the system will be smart enough to detect guns at all places, giving good information to the security managers.

There are always new ideas which are being grown up, we hope shortly we will perform smarter systems with better detection and more intelligent.

Finally, thanks to this modern technology, humans have become safer with their help.

REFERENCES

[1] Nixon, M., & Aguado, A. (2019). *Feature extraction and image processing for computer vision*. Academic press.

[2] Hubel, D. H., & Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, *148*(3), 574.

[3] Langdon, J. (1955). The perception of three-dimensional solids. *Quarterly Journal of Experimental Psychology*, *7*(3), 133-146.

[4] Papert, S. A. (1966). The summer vision project., *artificial intelligence group.*

[5] Marr, D. (2010). Vision: A computational investigation into the human representation and processing of visual information. MIT press.

[6] Mplus, M. M. U. Chapman & Hall/CRC Statistics in the Social and Behavioral Sciences Series.

[7] Ballard, D. H., & Brown, C. M. (1982). Computer vision, Prentice Hall.

[8] Lowe, D. G. (1987). Three-dimensional object recognition from single two-dimensional images. Artificial intelligence, 31(3), 355-395.

[9] Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, *22*(8), 888-905.

[10] Viola, P., & Jones, M. (2001, December). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001* (Vol. 1, pp. I-I). IEEE.

[11] Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)* (Vol. 1, pp. 886-893). Ieee.

[12] Lowe, D. G. (1999, September). Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision* (Vol. 2, pp. 1150-1157). Ieee.

[13] Carlson, N. R., & Birkett, M. A. (2017). Physiology of behavior, *Pearson Education, Boston.*

[14] M. G. Armstrong, D. J. Flynn, M. E. Hammond, S. J. E. Jolly and R. A. Salmon, "High Frame-Rate Television," in *SMPTE Motion Imaging Journal*, vol. 118, no. 7, pp. 54-59, Oct. 2009, doi: 10.5594/J15986.

[15] Avery, J. (2020). Seeing the Past with Computers: Experiments with Augmented Reality and Computer Vision for History. Kevin Kee and Timothy Compeau, eds. Ann Arbor, MI: University of Michigan Press, 2019. 254p.

[16] Kofman, S. (1999). Camera obscura: of ideology (No. 43-44). Cornell University Press.

[17] Ketkar, N., & Santana, E. (2017). *Deep learning with python* (Vol. 1). Berkeley, CA: Apress..

[18] Brüel-Gabrielsson, R., Nelson, B. J., Dwaraknath, A., Skraba, P., Guibas, L. J., & Carlsson, G. (2019). A topology layer for machine learning. *arXiv preprint arXiv:1905.12200*.

[19] Raschka, S. (2015). *Python machine learning*. Packt publishing ltd.

[20] Michie, D., Spiegelhalter, D. J., & Taylor, C. C. (1994). Machine learning. Neural and Statistical Classification, 13(1994), 1-298.

[21] Mitchell, T. M. (1999). Machine learning and data mining. *Communications of the ACM*, *42*(11), 30-36.

[22] Li, D., & Du, Y. (2007). Artificial intelligence with uncertainty. CRC press.

[23] Wallén, J. (2008). The history of the industrial robot. Linköping University Electronic Press.

[24] William John, T. (2010). Artificial Intelligence-Agents and Environments, ApS

[25] Raschka, S., Julian, D., & Hearty, J. (2016). *Python: deeper insights into machine learning*. Packt Publishing Ltd.

[26] Witten, I. H., Frank, E., & Hall, M. A. (2011). Data mining: practical machine learning tools and techniques. 3rd edn Amsterdam. The Netherlands: Morgan Kaufmann.

[27] Stuart, J., & Norvig, P. (2010). Artificial Intelligence: a Modern Approach Prentice-Hall. In *A Simon & Schuster Company Englewood Cliffs*.

[28] Russell, S., & Norvig, P. (2020). Artificial intelligence: a modern approach, 4th edition, Pearson Series.

[29] Cycleback, D. (2019). Philosophy of artificial Intelligence, 1st edition, Bookboon.

[30] Stenroos, O. (2017). Object detection from images using convolutional neural networks, Aalto University, Master degree Thesis, Aalto.

[31] Schapire, R. E. The strength of weak learnability, Machine Learning5 (1990), 197-227. *Google Scholar Google Scholar Digital Library Digital Library*.

[32] Zhao, X., Liang, S., & Wei, Y. (2018). Pseudo mask augmented object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4061-4070).

[33] Lee, Y., Han, D. K., & Ko, H. (2013). Reinforced adaboost learning for object detection with local pattern representations. *The Scientific World Journal*, *2013*.

[34] Alpaydin, E. (2020). *Introduction to machine learning*. MIT press.

[35] Eysenbach, B., Salakhutdinov, R., & Levine, S. (2020). C-Learning: Learning to Achieve Goals via Recursive Classification. *arXiv preprint arXiv:2011.08909*.

[36] Karami, E., Shehata, M., & Smith, A. (2017). Image identification using SIFT algorithm: Performance analysis against different image deformations. *arXiv preprint arXiv:1710.02728*.

[37] Deka, P. C. (2019). *A Primer on Machine Learning Applications in Civil Engineering*. CRC Press.

[38] Wagstaff, K. (2012). Machine learning that matters. *arXiv preprint arXiv:1206.4656*.

[39] Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1, No. 2). Cambridge: MIT press.

[40] Nielsen, M. A. (2015). *Neural networks and deep learning* (Vol. 25). San Francisco, CA: Determination press.

[41] Sinha, R. K., Pandey, R., & Pattnaik, R. (2018). Deep learning for computer vision tasks: a review. *arXiv preprint arXiv:1804.03928*.

[42] Gao, M., Jiang, J., Zou, G., John, V., & Liu, Z. (2019). RGB-D-based object recognition using multimodal convolutional neural networks: A survey. *IEEE Access*, *7*, 43110-43136.

[43] Nugraha, N., Madenda, S., Indarti, D., & Agushinta, R. D. (2016, June). Analysis and Recognition of Curve Type as The Basis of Object Recognition in Image. In *Journal of Physics: Conference Series* (Vol. 725, No. 1, p. 012006). IOP Publishing.

[44] Runia, T. F. H. (2015). High-Speed Object Detection: Design, Study and Implementation of a Detection Framework using Channel Features and Boosting, Master Thesis.

[45] Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., ... & Asari, V. K. (2018). The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*.

[46] Kaehler, A., & Bradski, G. (2016). *Learning OpenCV 3: computer vision in C++ with the OpenCV library*. " O'Reilly Media, Inc.".

[47] Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.

[48] Paszke, A., Chaurasia, A., Kim, S., & Culurciello, E. (2016). Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*.

[49] Wang, J., Chen, K., Xu, R., Liu, Z., Loy, C. C., & Lin, D. (2019). Carafe: Content-aware reassembly of features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 3007-3016).

[50] Rawat, W., & Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, *29*(9), 2352-2449.

[51] Zafar, I., Tzanidou, G., Burton, R., Patel, N., & Araujo, L. (2018). *Hands-on convolutional neural networks with TensorFlow: Solve computer vision problems with modeling in TensorFlow and Python*. Packt Publishing Ltd.

[52] Fathi, A., Wojna, Z., Rathod, V., Wang, P., Song, H. O., Guadarrama, S., & Murphy, K. P. (2017). Semantic instance segmentation via deep metric learning. *arXiv preprint arXiv:1703.10277*.

[53] Cardoso, G. V. S., Ciarelli, P. M., & Vassallo, R. F. (2019, September). Use of Deep Learning for Firearms Detection in Images. In *Anais do XV Workshop de Visão Computacional* (pp. 109-114). SBC.

[54] Wang, Z., & Zhou, Y. (2020). A HED-optimized Automatic Detection and Tracking Algorithm for Marine Moving Targets based on YOLO V3. In *Journal of Physics: Conference Series* (Vol. 1449, No. 1, p. 012126). IOP Publishing.

[55] Lai, J., & Maples, S. (2017). Developing a Real-Time Gun Detection Classifier. *Course: CS231n, Stanford University*.

[56] Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, *111*(1), 98-136.

[57] Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, *88*(2), 303-338.

[58] http://host.robots.ox.ac.uk/pascal/VOC/voc2012, access: 18 January 2021

[59] Cisco, U. (2020). Cisco annual internet report (2018–2023) white paper.

[60] Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39-41.

[61] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248-255). Ieee.

[62] http://image-net.org/about-stats, access: 8 March 2020.

[63] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740-755). Springer, Cham.

[64] Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., & Torralba, A. (2010, June). Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition* (pp. 3485-3492). IEEE.

[65] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2015). Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence*, *38*(1), 142-158.

[66] Uijlings, J. R., Van De Sande, K. E., Gevers, T., & Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*, *104*(2), 154-171.

[67] Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448).

[68] Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, *39*(6), 1137-1149.

[69] Lin, W. (2020). *Mask R-CNN*. Korea University. http://kucg.korea.ac.kr/new/seminar/2020/ppt/ppt-2020-11-19.pdf. Access 16 February 2020

[70] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

[71] https://machinethink.net/blog/object-detection-with-yolo, access: 18 April 2020

[72] https://jonathan-hui.medium.com/real-time-object-detection-with-yolo-yolov2-28b1b93e2088, access: 18 March 2020

[73] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint*

arXiv:1804.02767.

[74] https://git-scm.com, access: 2 February 2020.

[75] https://www.cygwin.com, access: 2 February 2020.

[76] https://pjreddie.com/darknet/, access: 2 February 2020

[77] Ruiz-Santaquiteria, J., Velasco-Mata, A., Vallez, N., Bueno, G., Alvarez, J. A., & Deniz, O. (2020). Handgun detection using combined human pose and weapon appearance. *arXiv preprint arXiv:2010.13753*.

[78] Ali, Y. H., Ali, S. M., Rahman, R. A., & Hamzah, R. I. R. (2016). Acoustic emission and artificial intelligent methods in condition monitoring of rotating machine–a review. In National Conference for Postgraduate Research.

[79] Lu, S., Wang, B., Wang, H., Chen, L., Linjian, M., & Zhang, X. (2019). A real-time object detection algorithm for video. *Computers & Electrical Engineering*, *77*, 398-408.

[80] Chen, W., Huang, H., Peng, S., Zhou, C., & Zhang, C. (2020). YOLO-face: a real-time face detector. *The Visual Computer*, 1-9.

[81] Atienza, R. (2020). *Advanced Deep Learning with TensorFlow 2 and Keras: Apply DL, GANs, VAEs, deep RL, unsupervised learning, object detection and segmentation, and more*. Packt Publishing Ltd.

[82] Chella, A., Frixione, M., & Gaglio, S. (2004). Conceptual Spaces for Computer Vision Representations. *Artificial Intelligence Review, 16*, 137-152.

CURRICULUM VITAE

Rayan SULAIMAN KHALAF

PERSONAL INFORMATIONS

[redacted]

RESEARCH EXPERIENCES

- ✓ Computer Programming languages available (C-C++, MATLAB, Visual Basic)
- ✓ Computer programs you can use (MS Office, Photoshop, vb.)

WORK EXPERIENCE

2011-2013: Lecturer at Shekhan Technical Institute

ACADEMIC ACTIVITIES

Paper: Digital Forensics: Focusing on Image Forensics. In 2019 7th International Symposium on Digital Forensics and Security (ISDFS) (pp. 1-5).

Proceedings: In 2019 7th International Symposium on Digital Forensics and Security (ISDFS) (pp. 1-5). IEEE.

Paper: Smart Arms Detection System Using YOLO Algorithm and OpenCV Libraries. In Turkish Journal of Science & Technology 16(1), pp. 129-136, 2021

Proceedings: In Turkish Journal of Science & Technology 16(1), pp. 129-136, 2021