

A PROJECT REPORT ON
OBJECT DETECTION USING OPENCV- PYTHON

**Submitted in partial fulfilment of the
requirement for the award of the degree of**

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by

SEERAM SAI VIVEK

186Q1A0534

Under the esteemed guidance of

Mr. P. RAMAKRISHNA, MTech., (Ph.D.)

Associate Professor



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

**KAKINADA INSTITUTE OF ENGINEERING &
TECHNOLOGY-II**

**Approved by AICTE, Govt. of AP. Affiliated to Jawaharlal Nehru
Technological University Kakinada, Yanam Road, Korangi-533461)**

(2018-2022)

KAKINADA INSTITUTE OF ENGINEERING & TECHNOLOGY-II

(Approved by AICTE, Govt. of AP. Affiliated to Jawaharlal Nehru Technological University Kakinada, Yanam Road, Korangi-533461)



CERTIFICATE

This is certify that the project entitled “**OBJECT DETECTION USING OPENCV-PYTHON**” that is being submitted by **SEERAM SAI VIVEK (186Q1A0534)** in Partial Fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY IN “COMPUTER SCIENCE & ENGINEERING** in the academic year 2021-2022 from **KAKINADA INSTITUTE OF ENGINEERING & TECHNOLOGY-II** affiliated to **Jawaharlal Nehru Technological University Kakinada**, is a record of bonafide work carried out by him/her under my guidance and supervision.

The results embodied in this project have not been submitted to any other University or Institute for the award of any degree.

INTERNAL SUPERVISOR

EXTERNAL EXAMINAR

HEAD OF THE DEPARTMENT

DECLARATION

I hereby declare that the project work entitled “**OBJECT DETECTION USING OPENCV-PYTHON**” submitted to **KAKINADA INSTITUTE OF ENGINEERING & TECHNOLOGY-II** is a record of original done by us under the guidance of **Mr.P. RAMAKRISHNA, MTech., (Ph. D) Associate Professor** in Department of **Computer Science & Engineering** and this project work submitted in the partial fulfilment of requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering**. The results embodied in this thesis have not been submitted to any other university or institute for the award of any degree.

We further declare that we or any other person has not previously submitted this project report to any other institution or university for any other degree or diploma.

Place:

Date:

With Gratitude

SEERAM SAI VIVEK
(186Q1A0534)

ACKNOWLEDGEMENT

We would like to take the privilege of the opportunity to express our gratitude in to the project work of “**OBJECT DETECTION USING OPENCV-PYTHON**” enabled us to express our special thanks to our honorable Chairman of the institution Sir **P.V. VISWAM**.

We are thankful to our honorable Principal **Dr. Ch. BHAVANNARAYANA, Ph.D., Professor.**, who has shown keen interest in us and encouraged us by providing all the facilities to complete our project successfully.

We owe our gratitude to our Beloved head of the Department CSE **Mr.P SUBBARAO, MTech, (Ph.D.), Associate Professor** for assisting us in completing the project work.

We wish our sincere Thanks to our supervisor **Mr.P. RAMAKRISHNA, MTech, (Ph.D.) Associate Professor** who has been a source of inspiration for us throughout our project and for his valuable advices in making our project successful.

We wish to express our sincere Thanks to all teaching staff of Computer Science and Engineering. We wish to express our special Thanks to all the Faculty members of our college of their concern in subjects and their help throughout our course.

We are the Thankful to our parents and all our friends who had given us good co-operation and suggestions thought our project and helped us successful compilation.

With Gratitude

SEERAM SAI VIVEK
(186Q1A0534)

ABSTRACT

Question location is a computer technology connected to computer vision and image preparation that deals with recognizing events of semantic items such as people, buildings, automobiles, and other things in digital pictures and movies. It's widely utilized in applications including image retrieval, security surveillance, advanced driver assistance systems, facial recognition for counting faces, and security purpose locks in banks, mobile phones, and other places. The Open CV Question Discovery API is an open-source framework based on Open CV's greatest features that makes it simple to construct, prepare, and communicate question location models. After installing of all the dependencies, we code this project in Visual Studio Code. We can detect the objects in images and videos as well in real time by using the Open CV. For detecting the objects in the videos, we use the OpenCV dependency.

INDEX

<u>S.NO</u>	<u>CONTENT</u>	<u>PAGE NO</u>
1	INTRODUCTION	01
2	SYSTEM ANALYSIS	03
	2.1. Existing system	
	2.1.1. Disadvantages of existing system	
	2.2. Proposed system	
	2.2.1. Advantages of proposed system	
	2.3. Object detection approaches	
3	SYSTEM REQUIREMENTS	09
	3.1. Hardware requirements	
	3.2. Software requirements	
4	REQUIREMENT ANALYSIS	10
	4.1 Functional requirements	
	4.1 Nom-Functional requirements	
5	SYSTEM DESIGN	11
	5.1. System Architecture	
	5.2. UML diagrams	
6	IMPLEMENTATION	19
	6.1 Technology Description	
	6.2 Python	
	6.3 Why Machine Learning?	
	6.4 Installing Steps	
	6.5 Procedure for Execution	
7	SAMPLE CODING	39
	7.1 Static Object Detection	
	7.2 Real Time Object Detection:	
8	SYSTEM TESTING	45
	8.1 Black Box Testing	
9	OUTPUT SCREENS	49
10	CONCLUSION AND FUTURE SCOPE	54
11	BIBLIOGRAPHY	55

CHAPTER-1

INTRODUCTION

1.1 INTRODUCTION

Open CV could be a computer program library planned by the Google group to implement machine learning and profound learning concepts within the most effortless way. Object detection may be a computer vision strategy in which a package can distinguish and follow the protest from a given picture or video.

Convolutional Neural Systems have revolutionized Design Acknowledgment in final decades. As a result, CNNs are as of now utilized for various applications such as Protest Discovery and Picture Recognition.

The Open CV protest discovery API could be a computer program system utilized for question location assignments. Open CV Protest Discovery API and other comparable APIs (Keras RCNN, YOLO) employments pre prepared CNNs interior the frameworks for expectations. Most of those pretrained models are prepared utilizing, The Open Pictures Dataset, the COCO dataset, and the KITTI dataset.

These models are prepared to identify a few settled sorts of categories of objects such as, people, cars, puppy, toothbrush and etc. If you need to distinguish custom objects, you'll retrain those models utilizing your claim datasets.

The Open CV question discovery API is a framework for constructing a deep learning framework that comprehends protest discovery difficulties. The Open CV Address Area API is an open-source framework based on Open CV that enables creating, planning, and sending address area models simple. Appear Zoo is the name given to the pre-trained models that are now in their framework.

As you'll see underneath there are distinctive models available so what is differing in these models. These diverse models have distinctive designing and, in this way, provide unmistakable correctness's but there's a trade-off between speed of execution and the precision in setting bounding boxes.

1.2 PROBLEM STATEMENT

In the proposed system this project includes image detection in addition with the objects detecting in the video and live stream as well. All we need is an extra dependency and that is OpenCV. The extend “Object Location Framework utilizing Open CV Technique” recognizes objects effectively based on YOLO algorithm and apply the calculation on picture information and video data to identify objects

CHAPTER-2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

The constrained sum of clarified information as of now accessible for question discovery demonstrates to be another significant jump. Protest location datasets regularly contain ground truth illustrations for almost a dozen to a hundred classes of objects, whereas picture classification datasets can incorporate upwards of 100,000 classes. Moreover, crowdsourcing frequently produces picture classification labels for complimentary (for illustration, by parsing the content of user-provided photo captions). Gathering ground truth names in conjunction with exact bounding boxes for question location, in any case, remains fantastically monotonous work.

2.1.1 DISADVANTAGES

- Open CV discharges diverse overhauls each 2-3 month, expanding the overhead for a client to introduce it and tie it with the existing framework.
- Open CV gives homonyms that share comparable names but diverse usage, which makes it befuddling to keep in mind and utilize.
- In spite of the fact that Open CV decreases the length of code and makes it less demanding for a client to get to it, it includes a level of complexity to it utilize. Each code should be executed utilizing any stage for its bolster which increments the reliance for the execution.

2.2 PROPOSED SYSTEM

In the proposed system this project includes image detection in addition with the objects detecting in the video and live stream as well. All we need is an extra dependency and that is OpenCV. We use Open CV API which is an open-source API. Initially User will be given the alternatives to select the sort of the Record to be given to the Framework as an input. In this way, Client can either choose choice of Record Choice or begin the Camera. In the former, Client can select either Picture Record or a Video Record and, in the last mentioned, Client can begin the Camera module. Once the input is chosen Pre- processing is done, where the SXS grids are

shaped. The resulting network-shaped result is sent to the Bounding Box Expectation handle, which draws Bounding Boxes around the detected items. Another result from the previous preparation is supplied to the Course Prediction, which predicts the course of the protest to which it will take place. Then it's forwarded to the discovery handle, where an Edge is established to reduce clumsiness in the yield with a lot of Bounding Boxes and Names in the previous Yield. With Bounding Boxes and Labels, an image or a stream of images is formed at the end for picture and video or camera input individually.

2.2.1 ADVANTAGES

- It is an open-source stage that creates it accessible to all the clients around and prepared for the advancement of any framework on it.
- Nearly each operation can be performed utilizing this stage. With its characteristic of being sent on each machine and graphical representation of a demonstrate permits its clients to create any kind of framework utilizing Open CV.
- Open CV finds it utilize as an equipment speeding up library due to the parallelism of work models. Its employments distinctive dispersion procedures in GPU and CPU frameworks

Open CV acts in numerous spaces such as picture acknowledgment, voice location, movement location, time arrangement, etc. subsequently it suits the prerequisite of a client.

2.3 Object detection approaches:

2.3.1 Traditional approach:

The conventional approach of question location ordinarily has three stages: I) enlightening locale choice, ii) include extraction, and iii) classification of the object. In the primary organize, we attempt to discover the object's area. Objects have a diverse measure and angle proportion and may show up at distinctive areas of a picture. Due to this, we ought to check the complete picture employing a multiscale sliding window. But this strategy is computationally costly, and it produces numerous

unimportant candidates. In the moment step, we do the highlight extraction organize by utilizing methods like Filter, Hoard to extricate the visual include for recognizing the question. These visual highlights give a semantic and vigorous representation. In any case, due to diverse enlightenments conditions, perspective contrasts, and distinctive foundations it is exceptionally troublesome to physically plan a strong highlight descriptor to superbly depict all sorts of objects.

In the third classification organize we utilize Bolster Vector Machine (SVM) or Adbots

2.3.2 Modern approach:

The rise of profound learning can handle a few downsides of conventiona approaches. The profound learning models can memorize more complex highlights as we have seen as of now in picture classification tutorials.

In this article, we'll center on diverse profound learning-based question discovery models.

There are two types of systems accessible in profound learning protest discovery models.

The primary system is locale proposition based and it comprises of models like RCNN, SPP- NET, FRCNN, Faster CNN and the moment system is regression-based and comprises of Multibox, Attention Net, G-CNN, YOLO, SSD, YOLOV2. \

Beat 8 Calculations for Protest Detection

- Fast R-CNN.
- Faster R-CNN.
- Histogram of Situated Angles (HOG)
- Region-based Convolutional Neural Systems (R-CNN)
- Region-based Completely Convolutional Arrange (R-FCN)
- Single Shot Locator (SSD)
- Spatial Pyramid Pooling (SPP-net) □ YOLO (You Merely See Once)

2.3.3 Fast R-CNN:

The Speedy Region-Based Convolutional Organize method, often known as Fast R-CNN, is a planning computation for address finding that is written in Python and C++ (Caffe). This computation essentially eliminates the drawbacks of R-CNN and SPPnet while improving their speed and accuracy. Speedy R-CNN has the following advantages over R-CNN: – Higher area quality (mAP) than R-CNN, SPPnet Preparing is done in a single stage, with a multi-task disaster. All orchestration layers can be upgraded by preparing. Incorporating caching does not need a large amount of disc space.

2.3.4 YOLO (YOU ONLY LOOK ONCE):

You Simply See Once or YOLO is one of the prevalent calculations in protest location utilized by the analysts around the globe.

Agreeing to the analysts at Facebook AI Inquire about, the bound together engineering of YOLO is amazingly quick in way.

The base YOLO demonstrate forms pictures in real-time at 45 outlines per moment, whereas the littler form of the arrange, Quick YOLO forms a bewildering 155 outlines per moment whereas still accomplishing twofold the mAP of other real-time finders. This calculation outflanks the other location strategies, counting DPM and R-CNN, when summing up from normal pictures to other spaces like artwork.

2.3.5 Faster R-CNN:

Speedier R-CNN is an address area calculation that's comparative to R-CNN. This calculation employments the Region Recommendation Organize (RPN) that gives full-image convolutional highlights with the disclosure organize in a cost-effective way than R-CNN and Speedy R-CNN. A Region Recommendation Organize is basically a totally convolutional orchestrate that at the same time predicts the dissent bounds as well as objectless scores at each position of the protest and is ready end-to-end to make high-quality district proposals, which are at that point utilized by Speedy R-CNN for area of objects.

2.3.6 Histogram of Oriented Gradients (HOG):

Histogram of arranged slopes (Hoard) is fundamentally a highlight descriptor that's used to identify objects in picture preparing and other computer vision strategies.

The Histogram of arranged angles descriptor procedure incorporates events of slope introduction in restricted parcels of a picture, such as discovery window, the locale of intrigued (ROI), among others. One advantage of HOG-like highlights is their effortlessness, and it is simpler to get it the data they carry.

2.3.7 Region based convolutional Neural network(R-CNN):

The Region-based Convolutional Organize procedure (RCNN) may be a combination of region suggestion with Convolution Neural Frameworks (CNNs). R-CNN makes a contrast in confining objects with a significant organize and planning a high-capacity illustrate with because it was a small sum of commented on revelation data. It accomplishes extraordinary address revelation accuracy by utilizing a significant Community to classify address proposals. R-CNN has the capability to scale to thousands of dissent classes without turning to derived procedures, checking hashing

2.3.8 Region based Fully convolutional network:

Region-based Totally Convolutional Frameworks or R-FCN may be a region-based discoverer for dissent discovery. Not at all like other region-based locators that apply an costly per-region subnetwork such as Speedy R-CNN or Speedier R-CNN, this region-based discoverer is completely convolutional with about all computation shared on the full picture. R-FCN comprises of shared, totally convolutional structures as is the case of FCN that's known to resign removed way better; a much way better; higher; more grounded; an improved">an moved forward result than the Speedier R-CNN. In this calculation, all learnable weight layers are convolutional and are laid out to classify the ROIs into address categories and establishments.

2.3.9 Single Shot Detector (SSD):

Single Shot Discoverer (SSD) might be a strategy for recognizing objects in pictures utilizing a single significant neural organize. The SSD approach discretizes the abdicate space of bounding boxes into a set of default boxes over unmistakable perspective extents. After discretizing, the methodology scales per incorporate diagram area. The Single Shot Locator organize combines estimates from diverse highlight maps with assorted resolutions to ordinarily handle objects of diverse sizes



2.3.10 Object detection:

Feature extraction is carried out for each segmented rectangular area to predict whether the rectangle contains a valid object.

Overlapping boxes are combined into a single bounding rectangle.

CHAPTER-3

SYSTEM REQUIREMENTS

3.1 Hardware Requirements:

The Hardware consists of the physical components of the computer that input storage processing control, output devices. The kind of hardware used in the project is

- System : Intel core i3, up to 2.4 GHZ.
- Hard Disk : 1 GB.
- Ram : 4 GB.
- Other Tools : Web Cam

3.2 Software requirements:

Software is a set of programs to do a particular task. Software is an essential requirement of computer systems. The kind of software used in the project is

- Operating system : Windows 7,8,9,10
- Coding Language : PYTHON
- Documentation : MS Office
- IDE : VS Code, Anaconda Navigator

CHAPTER-4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

A useful prerequisite could be a necessity concerning a result of behaviour that might be given by a work of the framework. FRs are prerequisites that indicates a work that a framework or framework component must be able to perform. Utilitarian prerequisite depicts a usefulness to be made accessible to the clients of the framework, characterizing somewhat its behaviour as an reply to the jolt that it is subject to. This sort of requirement should not specify any innovative issue, that's, in a perfect world useful necessity must be free of plan and execution viewpoints

4.2 NON-FUNCTIONAL REQUIREMENTS

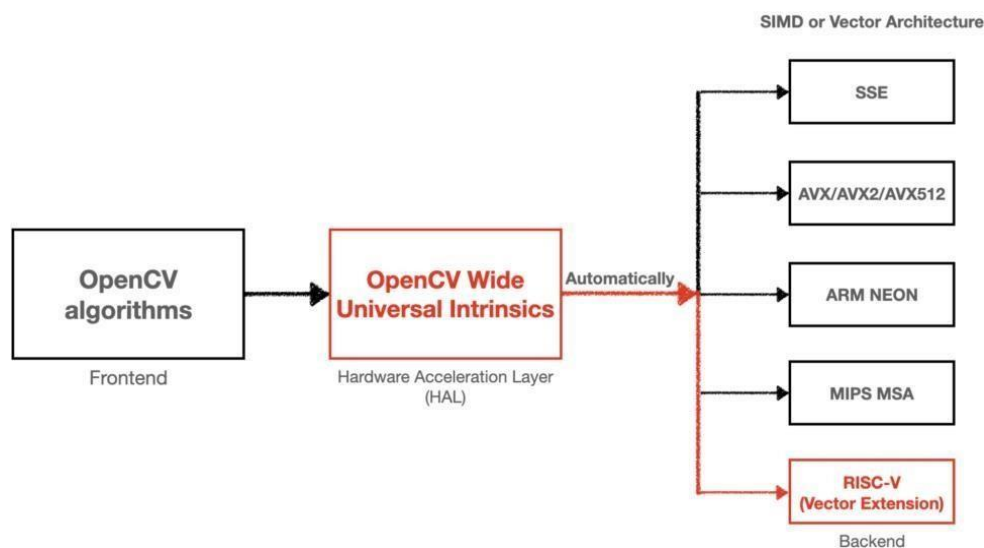
- ❖ **Availability:** Depicts how likely the framework is available for a client at a given point in time
- ❖ **Fault Tolerance:** Degree to which a framework, item or component works as expecting in spite of the nearness of equipment or computer program deficiencies.
- ❖ **Operability:** Degree to which a item or framework has qualities that make it simple to function and control
- ❖ **Performance:** Execution relative to the number of assets utilized beneath expressed conditions.
- ❖ **Portability:** Degree of adequacy and proficiency with which a framework, item or component can be exchanged from one equipment, program or other operational or utilization environment to another.
- ❖ **Usability:** Degree to which a item or framework can be utilized by indicated clients to attain indicated objectives with viability, proficiency, and fulfilment in a indicated setting of utilize.
- ❖ **Scalability:** Degree to which a item or framework can successfully and proficiently be adjusted for distinctive or advancing equipment, computer program or other operational or utilization situations.

CHAPTER-5

SYSTEMDESIGN

5.1 SYSTEM ARCHITECTURE

Framework engineering could be a conceptual show that portrays the structure and behaviour of numerous components and subsystems like different program applications, organize gadgets, equipment, and indeed other machinery of a framework. It centers on the whole framework. Framework design incorporates components of both computer program and equipment and is utilized to empower plan of such a composite framework.



5.1.1 Fundamental Design Concepts

Data Reflection is the process of speaking to basic highlights without counting or explaining the foundation elements of interest. The technique of elaboration is known as refinement. A progression is made by breaking down a clearly evident task explanation in a sequential design till programming dialect explanations are reached. Deliberation and refinement are two principles that work together. Modularity - The design of a program is divided into components known as modules. Software Design - It refers to the program's overall structure and how that structure contributes to a framework's conceptual acuity. Control Chain of Command - A program structure that describes how a program component is organized and infers a control progression.

- **Data Structure** - It's a visual depiction of the logical link between data pieces
- **Software Procedure** - It focuses on each module's processing separately.
- **Information Hiding** - Modules should be described and constructed in such a way that any information stored within them is unavailable to other modules that don't require it.

5.1.2 METHODOLOGY:

1. The first step is taking an input which is collection of data called dataset, Then the second step is analyzing the dataset, we will observe the relationships between features and how they are contributing for the detection of objects, then perform reduction on features which are removed based on the analysis.
2. SSD will break the picture into a few fragments and for each fragment it'll build a few bounding boxes.
3. At that point it'll check each box on the picture for an question of each and each lesson the network is train for
4. at final it'll compare the predictions with the ground truth.
5. Single Shot Finder Is Speedier Than The Previous State-Of-The-Art Techniques (YOLO) And Is Significantly More Accurate.
6. These Highlights Lead To Tall Precision, Even On Moo Resolution Input Picture.

5.2 UML DIAGRAMS

The Unified Modelling Language (UML) is a standard language for specifying, visualizing, constructing, and documenting the software system and its components. It is a graphical language, which provides a vocabulary and set of semantics and rules. The UML focuses on the conceptual and physical representation of the system. It captures the decisions and understandings about systems that must be constructed. It is used to understand, design, configure, maintain, and control information about the systems.

The UML is a language for:

- Visualizing
- Specifying
- Constructing
- Documenting

Visualizing

Through UML we see or visualize an existing system and ultimately, we visualize how the system is going to be after implementation. Unless we think, we cannot implement. UML helps to visualize, how the components of the system communicate and interact with each other.

Specifying

Specifying means building, models that are precise, unambiguous and complete UML addresses the specification of all the important analysis design, implementation decisions that must be made in developing and deploying a software system.

Constructing

UML models can be directly connected to a variety of programming language through mapping a model from UML to a programming language like JAVA or C++ or VB. Forward Engineering and Reverse Engineering is possible through UML

Documenting

The Deliverables of a project apart from coding are some Artefacts, which are critical in controlling, and communicating about a system during its developing requirements, architecture, design, source code, project plans, tests, prototypes releases, etc..

5.2.1 USE CASE DIAGRAM:

A use case diagram in the Unified Modelling Language (UML) is a type of behavioral diagram defined by and created from a use-case analysis, its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

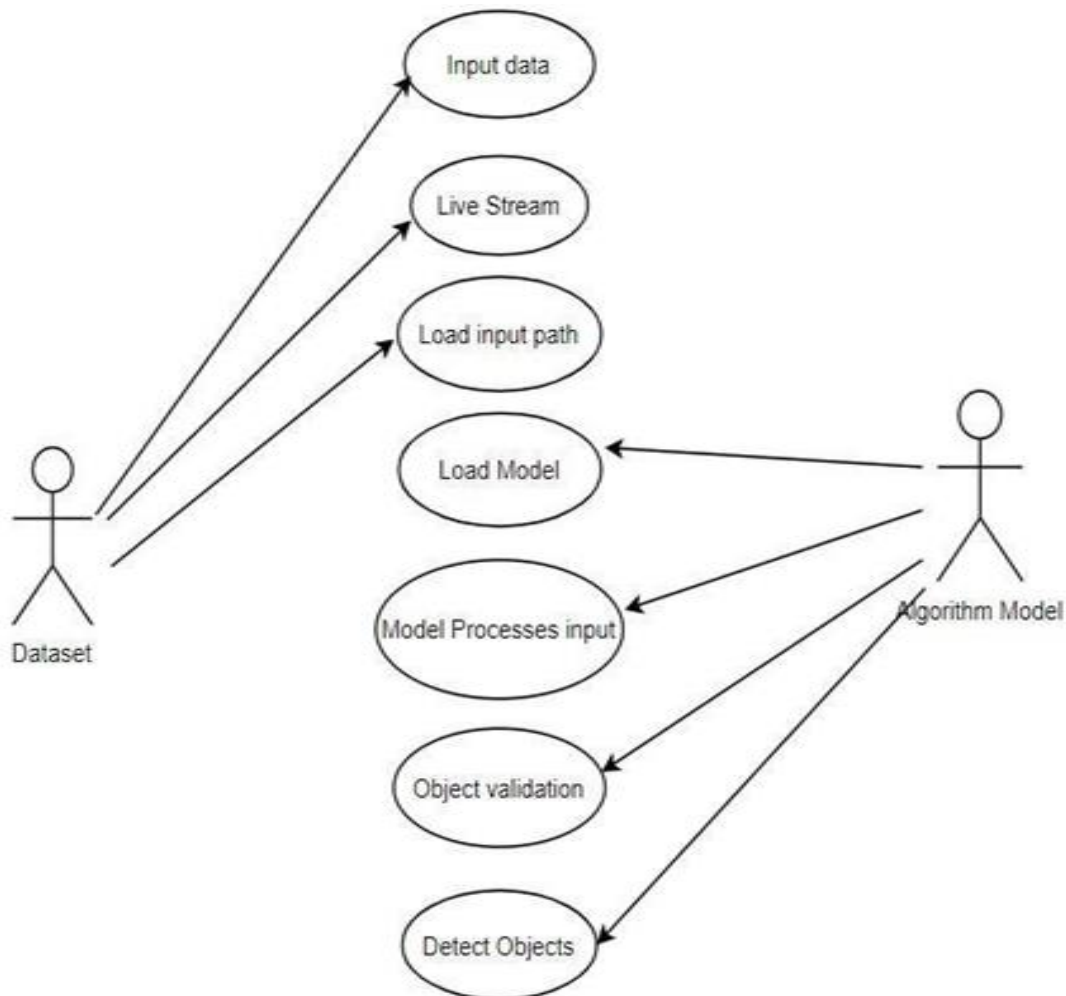


Fig.1 USECASE DIAGRAM

5.2.2 CLASS DIAGRAM:

Class is a category or group of things that has similar attributes and common behavior. A Rectangle is the icon that represents the class it is divided into three areas. The upper most area contains the name, the middle; area contains the attributes and the lowest areas show the operations.

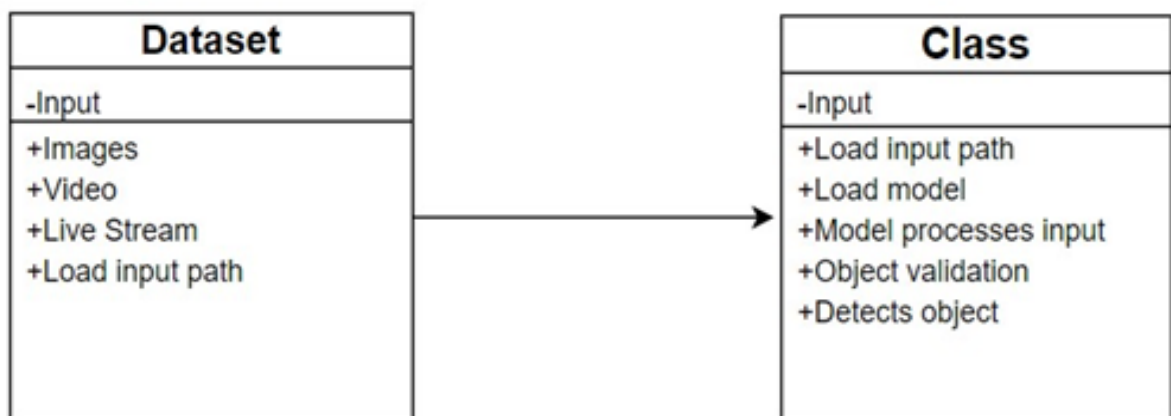


Fig.2 CLASS DIAGRAM

5.2.3 ACTIVITY DIAGRAM:

An activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

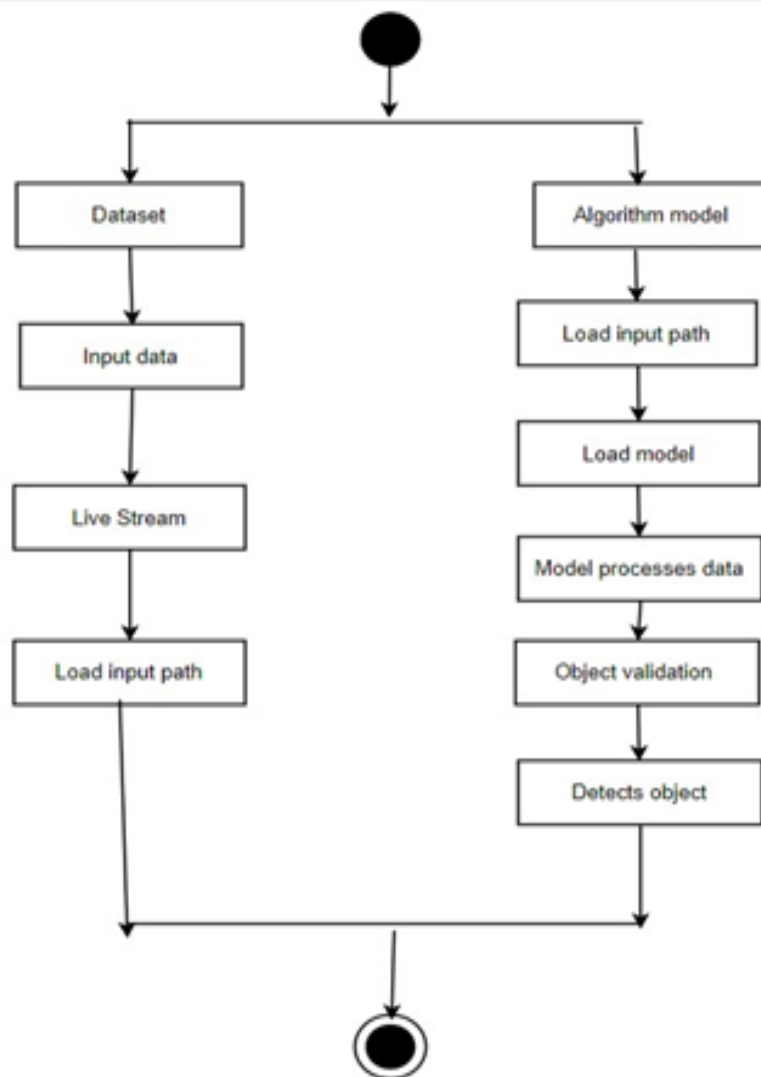


Fig.3 ACTIVITY DIAGRAM

5.2.4 FLOW DIAGRAM:

Information flow diagram is UML behaviour diagram which shows exchange information between system entities at some high levels of abstraction

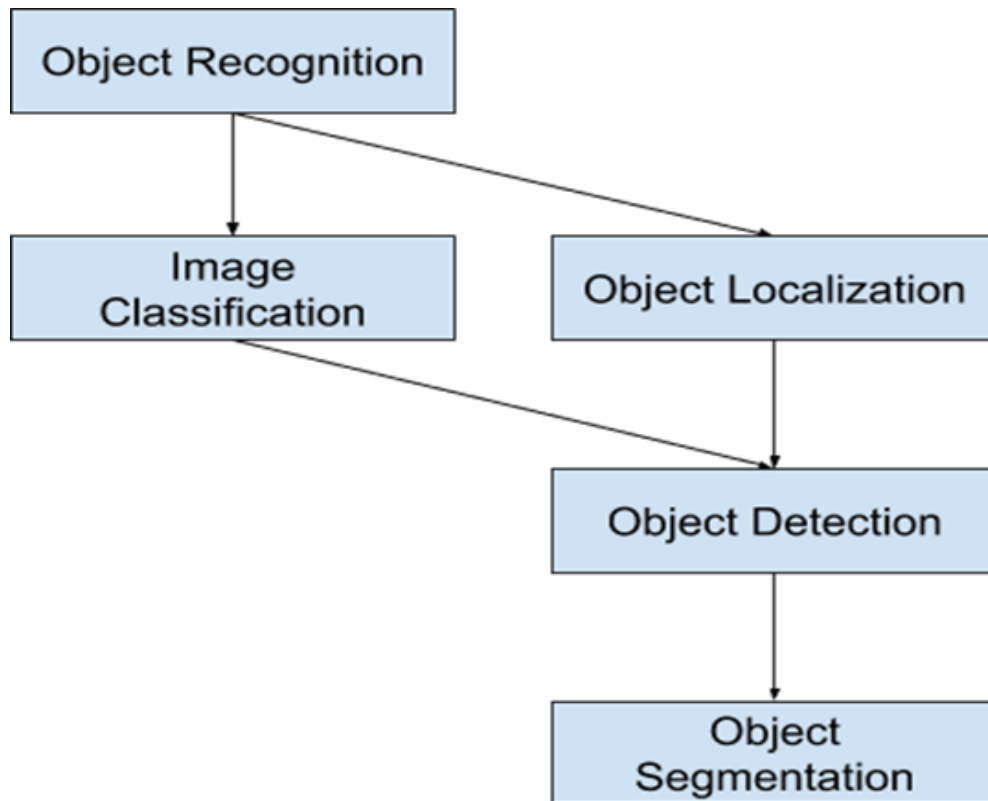


Fig4. FLOW DIAGRAM

While picture classification entails assigning a lesson name to an image, protest localization entails creating a bounding box around one or more of the image's elements. Object discovery is more difficult and combines these two tasks by drawing a bounding box around each point of interest in the image and assigning it a course name. All of these problems are referred to as object recognition. This post contains a sensitive exposition of the topic of question acknowledgment as well as state-of-the-art profound learning models that are being developed to address it.

Object recognition is a term that refers to a group of activities that are used to recognise things in digital pictures.

- The term "protest acknowledgement" refers to a group of related activities for identifying items in digitised pictures.
- RCNNs, or Region-Based Convolutional Neural Systems, are a group of systems for object recognition and localisation.

You Only Have to Look Once, or YOLO, is a group of protest acknowledgment processes designed for speed and real-time use.

- Using a bounding box, determine the proximity of things in a picture and sort or classify the objects identified.
- Input: A photograph or an image containing one or more items.
- Yield: One or more bounding boxes (e.g., with a point, width, and height), as well as a course name for each.

CHAPTER-6

IMPLEMENTATION

6.1 TECHNOLOGY DESCRIPTION

6.1.1 Single shot detector:

SSD has two components: a **backbone** model and **SSD head**. *Backbone* model usually is a pre-trained image classification network as a feature extractor. This is typically a network like ResNet trained on ImageNet from which the final fully connected classification layer has been removed. We are thus left with a deep neural network that is able to extract semantic meaning from the input image while preserving the spatial structure of the image albeit at a lower resolution. For ResNet34, the backbone results in a 256 7x7 feature maps for an input image. We will explain what feature and feature map are later on. The *SSD* head is just one or more convolutional layers added to this backbone and the outputs are interpreted as the bounding boxes and classes of objects in the spatial location of the final layers activations.

In the figure below, the first few layers (white boxes) are the backbone, the last few layers (blue boxes) represent the SSD head.

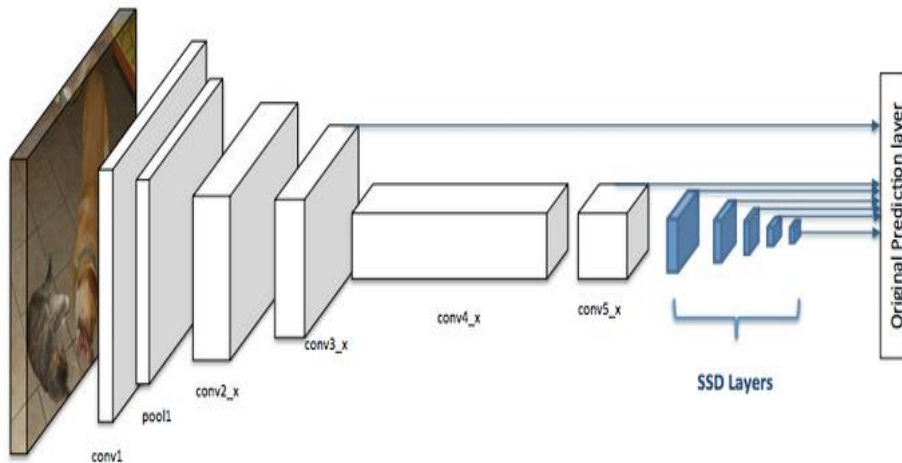


Fig1. Architecture of a convolutional neural network with a SSD detector

On typical datasets like as PascalVoc and COCO, SSD reached new records in terms of execution and accuracy for protest locating missions, earning over 74 percent MAP (mean Average Accuracy) at 59 outlines per moment. Single Shot: This means that the network's query localization and classification assignments are wiped out in a single forward pass. Multibox: Szeged et al. came up with this name for a method for bounding box relapse. Detector: The system is a protest finder that also identifies the things found.

6.1.2 Grid cell

Instead of using sliding window, SSD divides the image using a grid and have each grid cell be responsible for detecting objects in that region of the image. Detection objects simply means predicting the class and location of an object within that region. If no object is present, we consider it as the background class and the location is ignored. For instance, we could use a 4x4 grid in the example below. Each grid cell is able to output the position and shape of the object it contains.



Fig2. Example of a 4x4 grid

6.1.3 Anchor box

Each grid cell in SSD can be assigned with multiple anchor/prior boxes. These anchor boxes are pre-defined and each one is responsible for a size and shape within a grid cell. For example, the swimming pool in the image below corresponds to the taller anchor box while the building corresponds to the wider box.

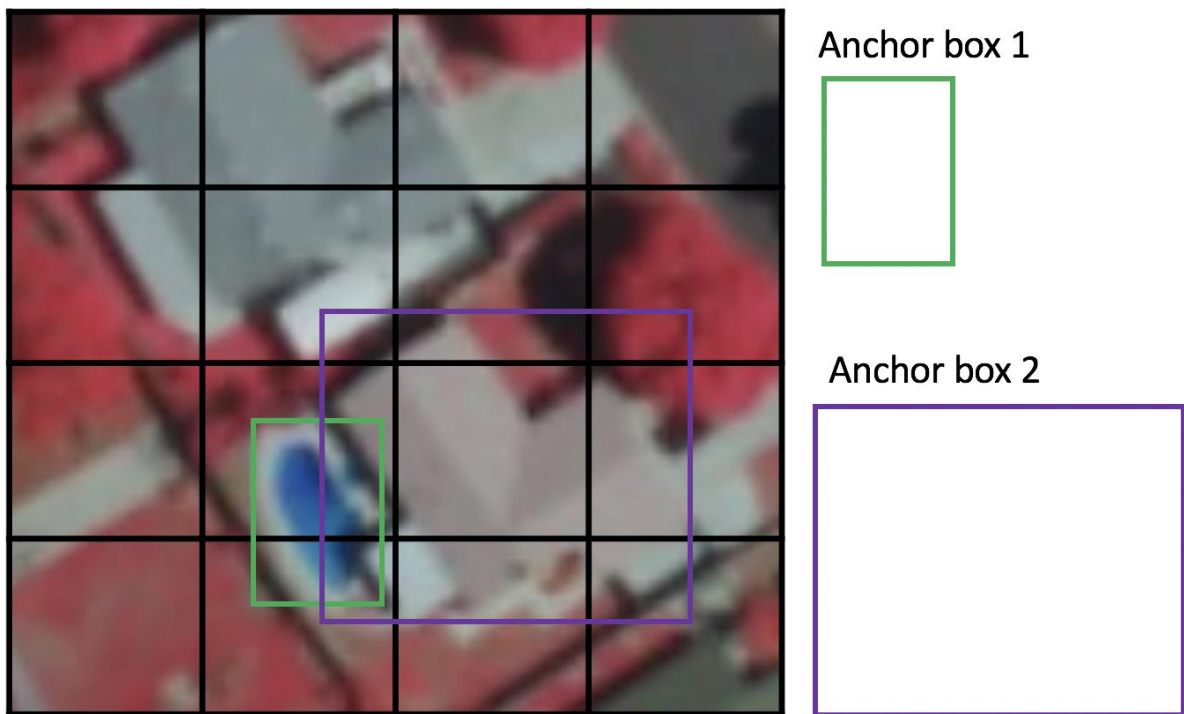


Fig 5. Example of two anchor boxes

SSD uses a matching phase while training, to match the appropriate anchor box with the bounding boxes of each ground truth object within an image. Essentially, the anchor box with the highest degree of overlap with an object is responsible for predicting that object's class and its location. This property is used for training the network and for predicting the detected objects and their locations once the network has been trained. In practice, each anchor box is specified by an aspect ratio and a zoom level.

6.1.4 Aspect ratio

Not all objects are square in shape. Some are longer and some are wider, by varying degrees. The SSD architecture allows pre-defined aspect ratios of the anchor boxes to account for this. The ratios parameter can be used to specify the different aspect ratios of the anchor boxes associates with each grid cell at each zoom/scale level.



Fig 6. The bounding box of building 1 is higher, while the bounding box for building 2 is wider

6.1.4 Zoom level

It is not necessary for the anchor boxes to have the same size as the grid cell. We might be interested in finding smaller or larger objects within a grid cell. The zooms parameter is used to specify how much the anchor boxes need to be scaled up or down with respect to each grid cell. Just like what we have seen in the anchor box example, the size of building is generally larger than swimming pool.

6.1.5 Receptive Field

Receptive field is defined as **the region in the input space that a particular CNN's feature is looking at (i.e. be affected by)**. We will use "feature" and "activation" interchangeably here and treat them as the linear combination (sometimes applying an activation function after that to increase non-linearity) of the previous layer at the corresponding location [3]. Because of the the convolution operation, features at different layers represent different sizes of region in the input image. As it goes deeper, the size represented by a feature gets larger. In this example below, we start with the bottom layer (5x5) and then apply a convolution that results in the middle layer (3x3) where one feature (green pixel) represents a 3x3 region of the input layer (bottom layer). And then apply the convolution to middle layer and get the top layer (2x2) where each feature corresponds to a 7x7 region on the input image. These kind of green and orange 2D array are also called **feature maps** which refer to a set of features created by applying the same feature extractor at different locations of the input map in a sliding window fastion. Features in the same feature map have the same receptive field and look for the same pattern but at different locations. This creates the spatial invariance of ConvNet.

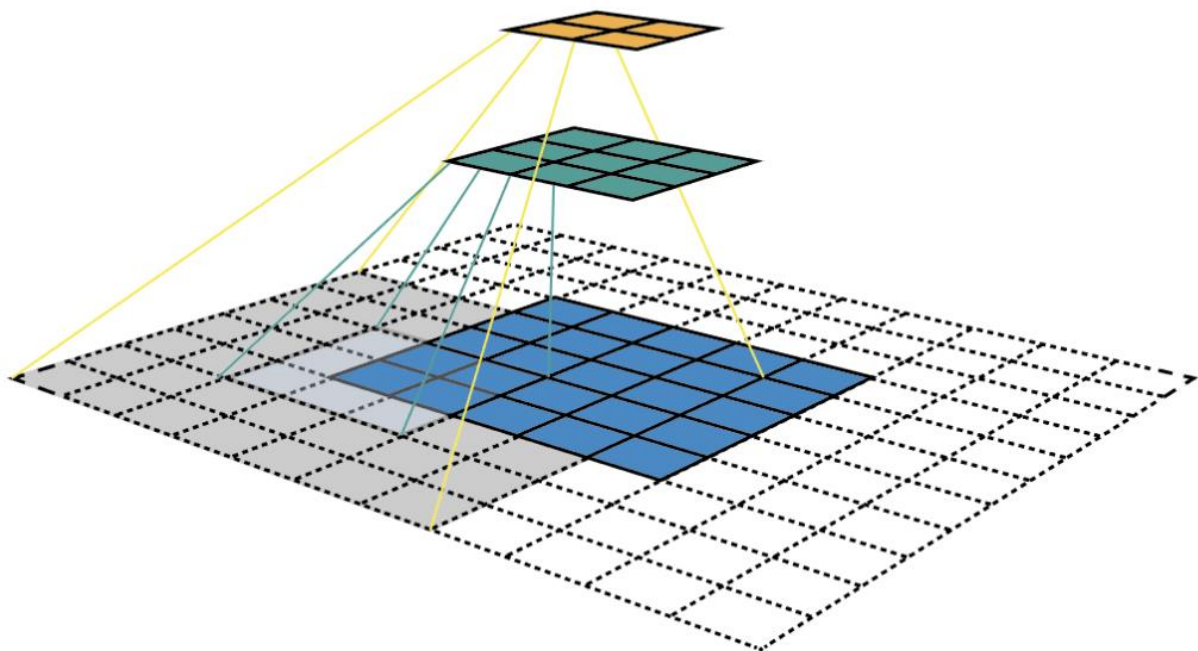


Fig 7. Visualizing CNN feature maps and receptive field

6.1.2 Machine Learning

Machine learning (ML) is the study of computer calculations that progress automatically as a result of experience and the use of data. It's seen as a piece of produced knowledge. Machine learning algorithms create a demonstration based on test data, also known as "preparing data," in order to generate expectations or choices without being explicitly adjusted. Machine learning calculations are used in a variety of applications, including pharmaceutical, mail filtering, speech recognition, and computer vision, when creating routine computations to execute the needed errands is difficult or impossible.

However, not all machine learning is factual learning. A subset of machine learning is closely connected to computational insights, which focuses on producing forecasts using computers; however, not all machine learning is factual learning. In the realm of machine learning, the idea of scientific optimization transmits tactics, hypotheses, and application regions. Information mining is a similar topic of study that focuses on unsupervised learning for exploratory data analysis. Machine learning is sometimes referred to as predictive analytics when applied to business challenges.

It entails computers figuring out how to do tasks without being explicitly programmed to do so. It entails computers learning from information provided in order to carry out certain tasks. It is possible to write calculations directing the machine how to conduct all steps necessary to light the issue at hand for simple chores delegated to computers. There is no need to learn anything about computers. It might be difficult for a person to complete the appropriate computations for more advanced errands. Instead, than having human software programmers identify each essential step, it may be more appealing to aid the computer in developing its own computation.

6.1.2.1 Supervised Learning

Directed learning or supervised learning may be a machine learning approach that's characterized by it utilize of labelled datasets. These datasets are planned to prepare or "supervise" calculations into classifying information or anticipating results precisely. Utilizing labelled inputs and yields, the show can degree its

exactness and learn over time. Supervised learning can be isolated into two sorts of issues

6.1.2.2 Multibox concept

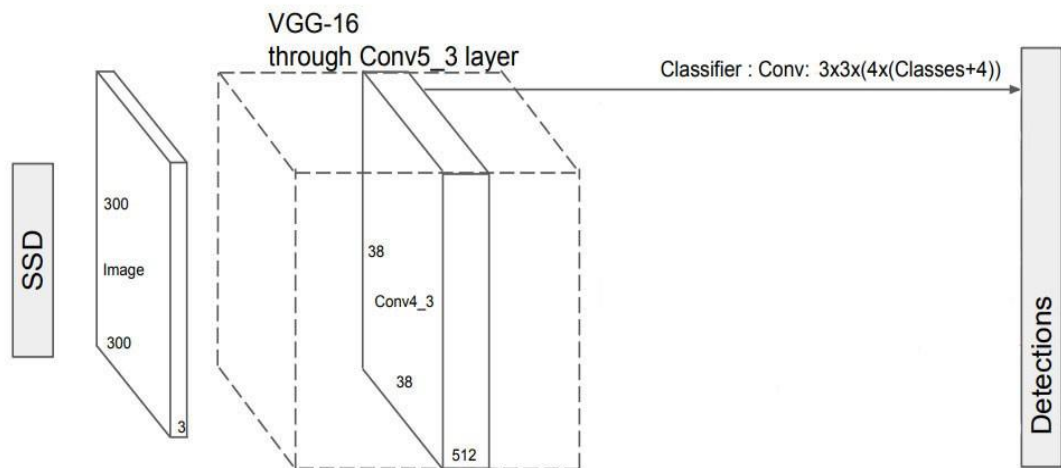
SSD is designed for object detection in real-time. Faster R-CNN uses a region proposal network to create boundary boxes and utilizes those boxes to classify objects. While it is considered the start-of-the-art in accuracy, the whole process runs at 7 frames per second. Far below what real-time processing needs. SSD speeds up the process by eliminating the need for the region proposal network. To recover the drop in accuracy, SSD applies a few improvements including multi-scale features and default boxes. These improvements allow SSD to match the Faster R-CNN's accuracy **using** lower resolution images, which further pushes the speed higher. According to the following comparison, it achieves the real-time processing speed and even beats the accuracy of the Faster R-CNN. (Accuracy is measured as the mean average precision mAP: the precision of the predictions.)

System	VOC2007 test <i>mAP</i>	FPS (Titan X)	Number of Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	~6000	~1000 x 600
YOLO (customized)	63.4	45	98	448 x 448
SSD300* (VGG16)	77.2	46	8732	300 x 300
SSD512* (VGG16)	79.8	19	24564	512 x 512

Performance comparison among object detection networks

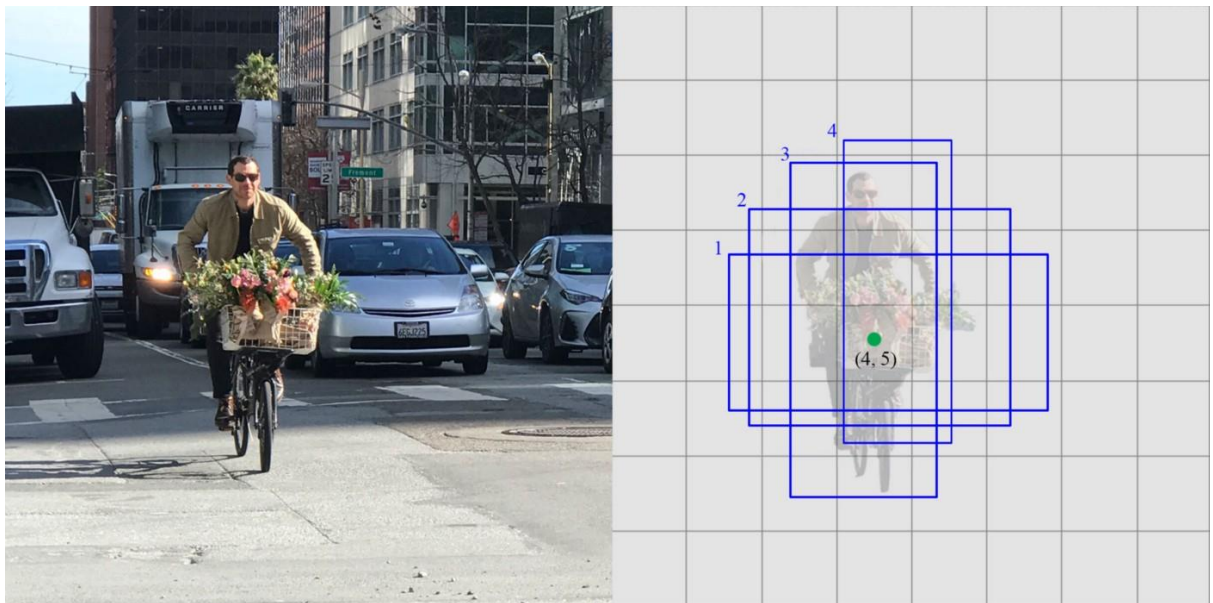
The SSD object detection composes of 2 parts:

- Extract feature maps, and
- Apply convolution filters to detect objects



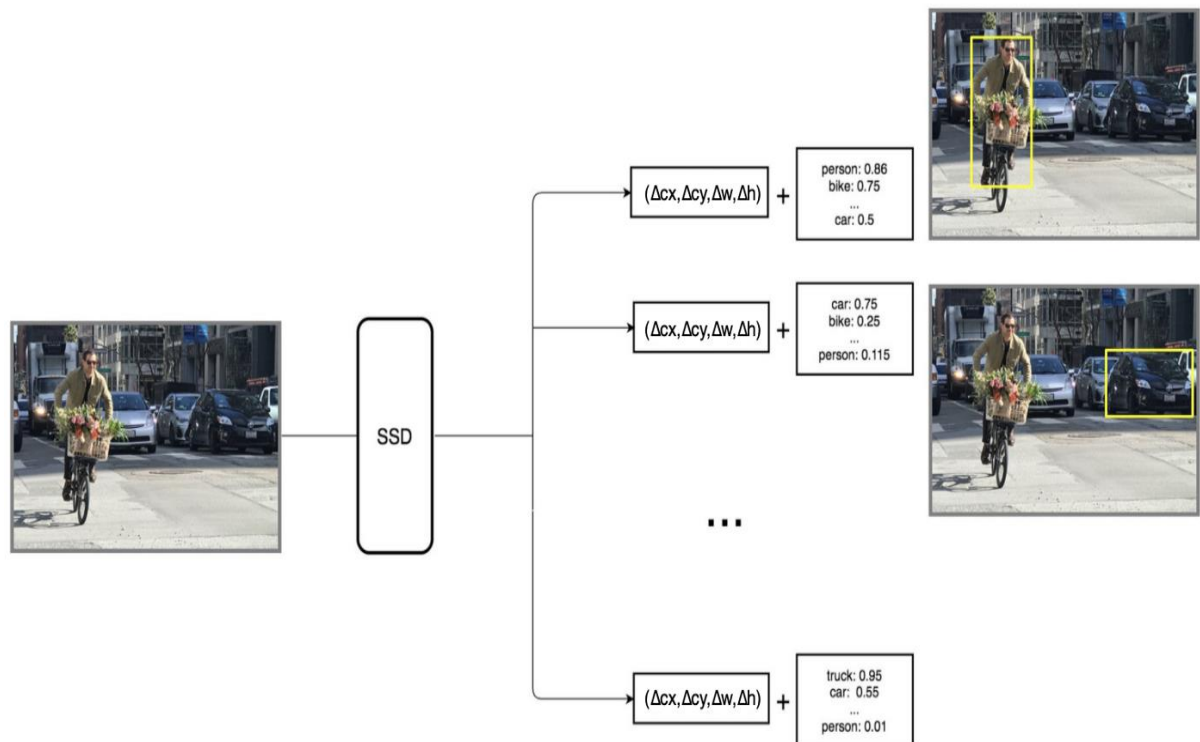
Modified from SSD: Single Shot Multibox Detector.

SSD uses **VGG16** to extract feature maps. Then it detects objects using the **Conv4_3** layer. For illustration, we draw the Conv4_3 to be 8×8 spatially (it should be 38×38). For each cell (also called location), it makes 4 object predictions.



Left: the original image. Right: 4 predictions at each cell.

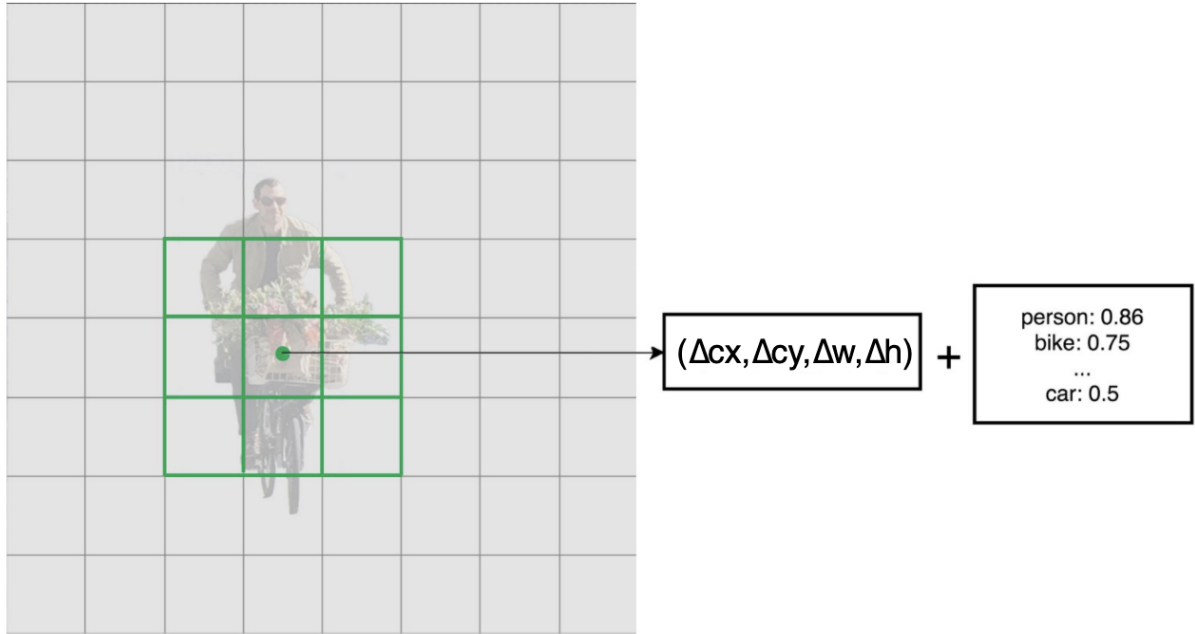
Each prediction composes of a boundary box and 21 scores for each class (one extra class for no object), and we pick the highest score as the class for the bounded object. Conv4_3 makes a total of $38 \times 38 \times 4$ predictions: four predictions per cell regardless of the depth of the feature maps. As expected, many predictions contain no object. SSD reserves a class “0” to indicate it has no objects.



Making multiple predictions containing boundary boxes and confidence scores is called Multibox.

6.1.2.3 CONVOLUTIONAL PREDICTORS FOR OBJECT DETECTION

SSD does not use a delegated region proposal network. Instead, it resolves to a very simple method. It computes both the location and class scores using small convolution filters. After extracting the feature maps, SSD applies 3×3 convolution filters for each cell to make predictions. (These filters compute the results just like the regular CNN filters.) Each filter outputs 25 channels: 21 scores for each class plus one boundary box (detail on the boundary box later).



For example, in Conv4_3, we apply four 3×3 filters to map 512 input channels to 25 output channels.

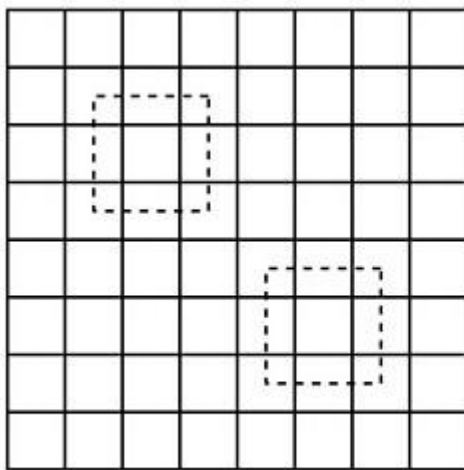
$$(38 \times 38 \times 512) \xrightarrow{(4 \times 3 \times 3 \times 512 \times (21+4))} (38 \times 38 \times 4 \times (21 + 4))$$

6.1.2.4 MULTI-SCALE FEATURE MAPS FOR DETECTION

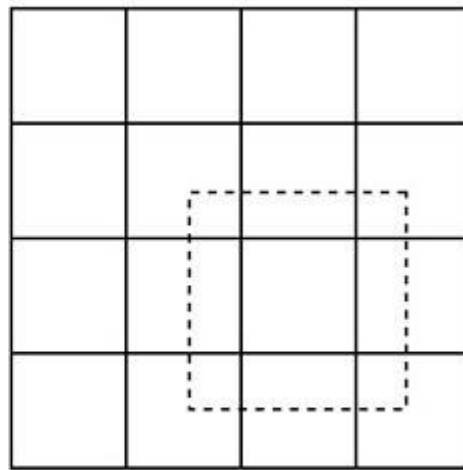
At first, we describe how SSD detects objects from a single layer. Actually, it uses multiple layers (**multi-scale feature maps**) to detect objects independently. As CNN reduces the spatial dimension gradually, the resolution of the feature maps also decreases. SSD uses lower resolution layers to detect larger scale objects. For example, the 4×4 feature maps are used for larger scale object



SCALE MATTER. PERSPECTIVE CHANGES THE SCALE OF THE OBJECTS.



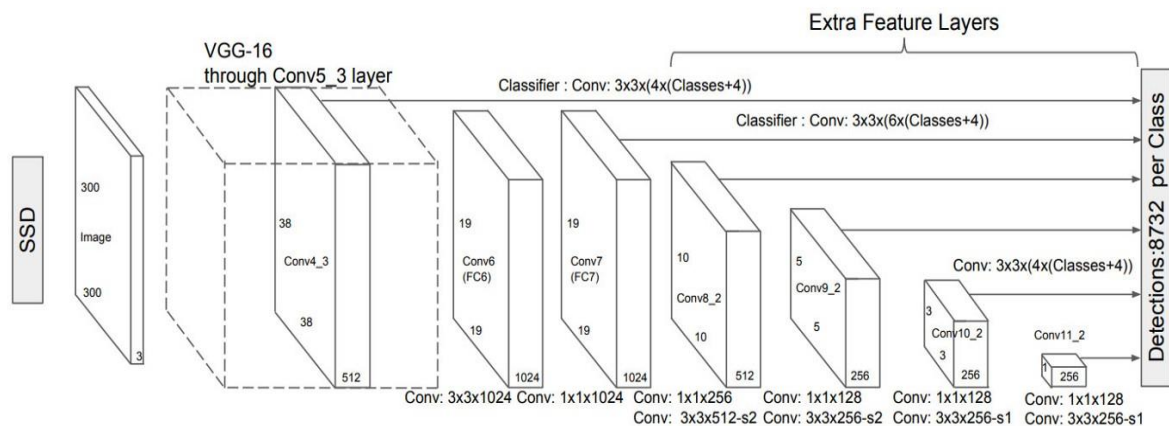
8 x 8 feature map



4 x 4 feature map

LOWER RESOLUTION FEATURE MAPS (RIGHT) DETECTS LARGER-SCALE OBJECTS.

SSD adds 6 more auxiliary convolution layers after the VGG16. Five of them will be added for object detection. In three of those layers, we make 6 predictions instead of 4. In total, SSD makes 8732 predictions using 6 layers.



SOURCE: SSD: SINGLE SHOT MULTIBOX DETECTOR

Multi-scale feature maps improve accuracy significantly. Here is the accuracy with different number of feature map layers used for object detection.

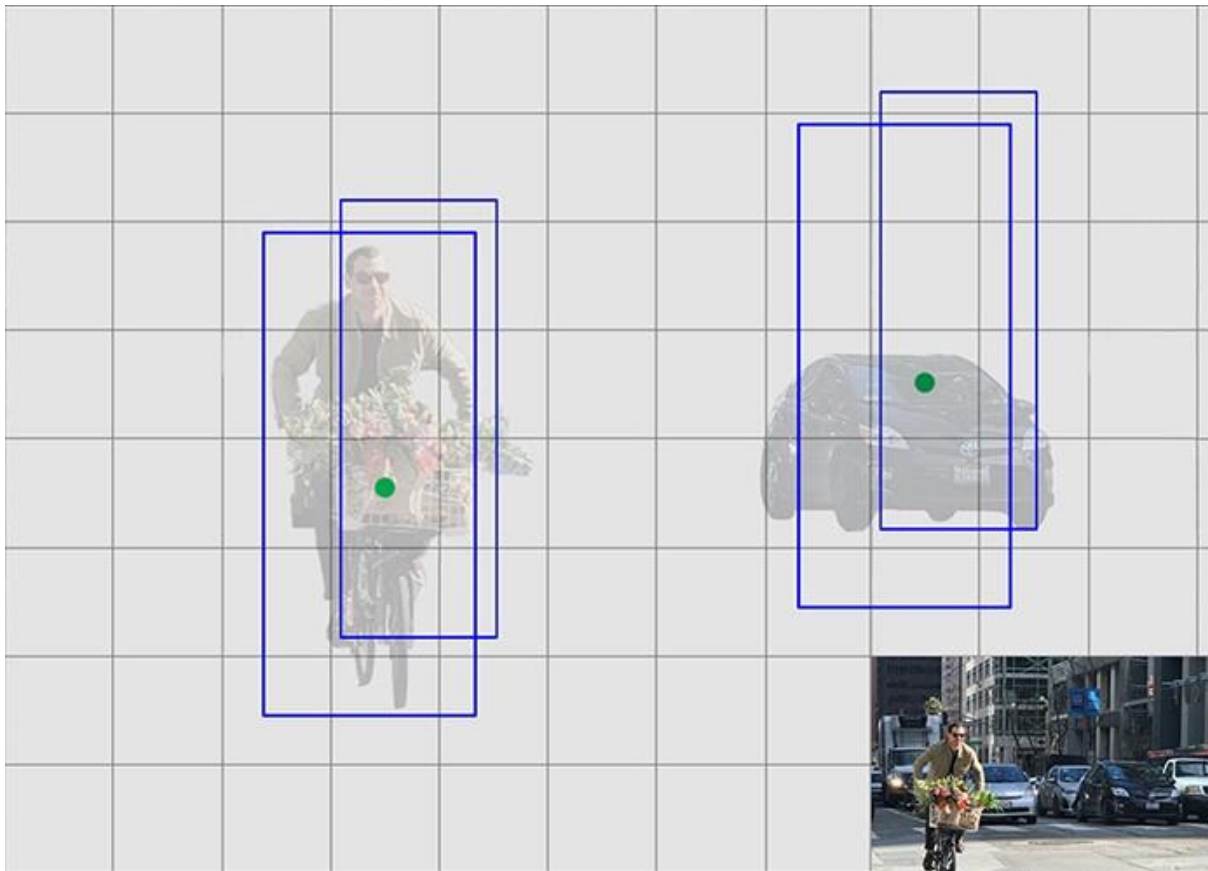
Prediction source layers from:						mAP		# Boxes
38 × 38	19 × 19	10 × 10	5 × 5	3 × 3	1 × 1	use boundary boxes?		
						Yes	No	
✓	✓	✓	✓	✓	✓	74.3	63.4	8732
✓	✓	✓				70.7	69.2	9864
	✓					62.4	64.0	8664

6.1.2.5 DEFAULT BOUNDARY BOX

The default boundary boxes are equivalent to **anchors** in Faster R-CNN.

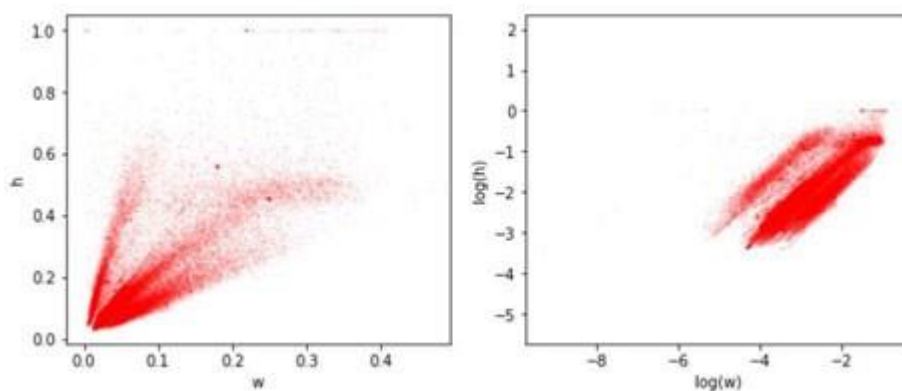
How do we predict boundary boxes? Just like Deep Learning, we can start with random predictions and use gradient descent to optimize the model. However, during the initial training, the model may fight with each other to determine what shapes (pedestrians or cars) to be optimized for which predictions. Empirical results indicate early training can be

very unstable. The boundary box predictions below work well for one category but not for others. We want our initial predictions to be diverse and not looking similar.



If our predictions cover more shapes, like the one below, our model can detect more object types. This kind of head start makes training much easier and more stable

In real-life, boundary boxes do not have arbitrary shapes and sizes. Cars have similar shapes and pedestrians have an approximate aspect ratio of 0.41. In the KITTI dataset used in autonomous driving, the width and height distributions for the boundary boxes are highly clustered.



Default boundary boxes are chosen manually. SSD defines a scale value for each feature map layer. Starting from the left, Conv4_3 detects objects at the smallest scale 0.2 (or 0.1 sometimes), and then increases linearly to the rightmost layer at a scale of 0.9. Combining the scale value with the target aspect ratios, we compute the width and the height of the default boxes. For layers making 6 predictions, SSD starts with 5 target aspect ratios: 1, 2, 3, 1/2, and 1/3. Then the width and the height of the default boxes are calculated as:

$$w = scale \cdot \sqrt{\text{aspect ratio}}$$

$$h = \frac{scale}{\sqrt{\text{aspect ratio}}}$$

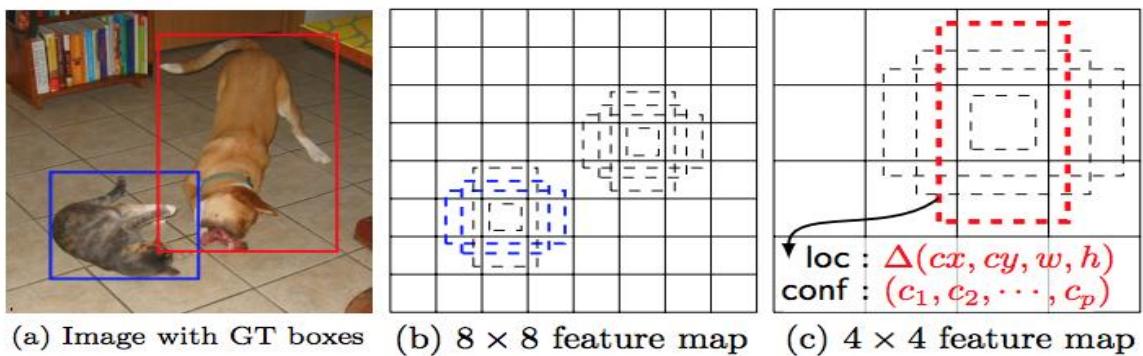
Then SSD adds an extra default box with scale:

$$scale = \sqrt{scale \cdot \text{scale at next level}}$$

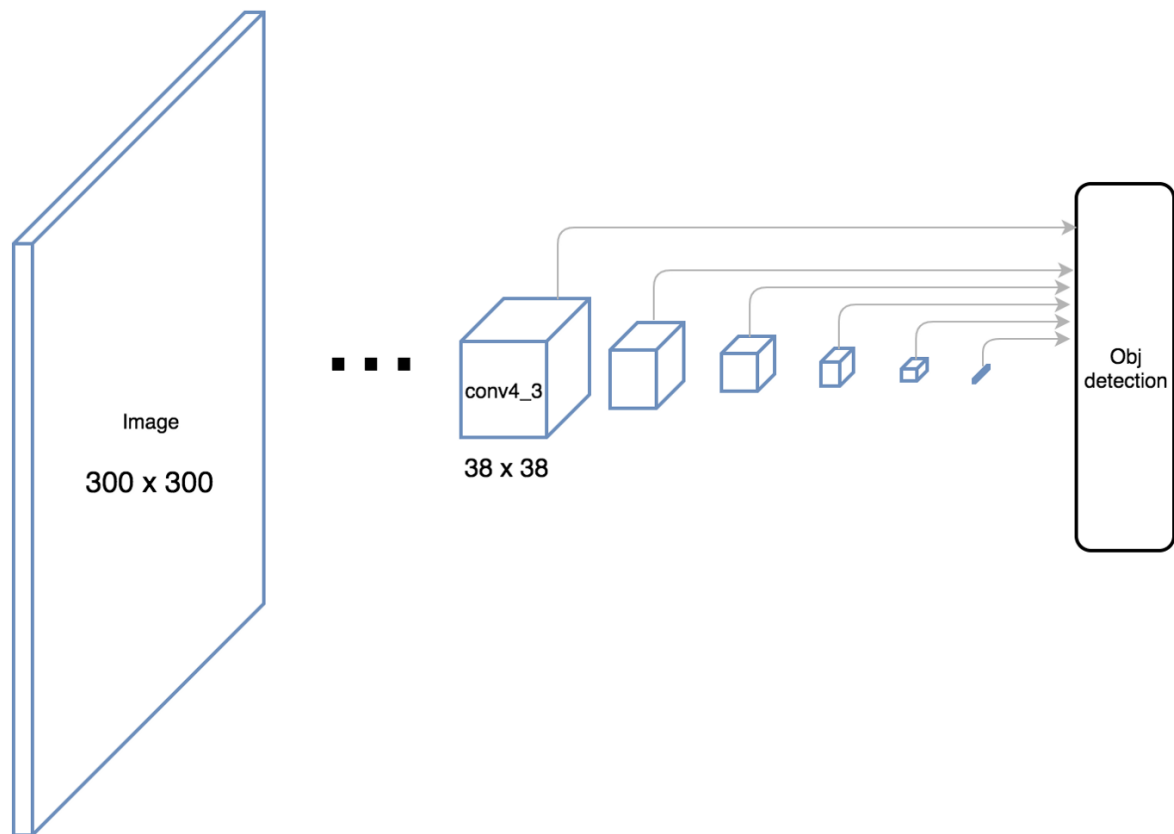
and aspect ratio = 1.

6.1.2.6 MULTI-SCALE FEATURE MAPS & DEFAULT BOUNDARY BOXES

Here is an example of how SSD combines multi-scale feature maps and default boundary boxes to detect objects at different scales and aspect ratios. The dog below matches one default box (in red) in the 4×4 feature map layer, but not any default boxes in the higher resolution 8×8 feature map. The cat which is smaller is detected only by the 8×8 feature map layer in 2 default boxes (in blue).



Higher-resolution feature maps are responsible for detecting small objects. The first layer for object detection *conv4_3* has a spatial dimension of 38×38 , a pretty large reduction from the input image. Hence, SSD usually performs badly for small objects comparing with other detection methods. If it is a problem, we can mitigate it by using images with higher resolution



SSD makes many predictions (8732) for better coverage of location, scale, and aspect ratios, more than many other detection methods. However, many predictions contain no object. Therefore, any predictions with class confidence scores lower than 0.01 will be eliminated.

Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	~ 1000×600
Fast YOLO	52.7	155	1	98	448×448
YOLO (VGG16)	66.4	21	1	98	448×448
SSD300	74.3	46	1	8732	300×300
SSD512	76.8	19	1	24564	512×512
SSD300	74.3	59	8	8732	300×300
SSD512	76.8	22	8	24564	512×512

6.1.2.3 LOGISTIC REGRESSION

The supervised learning classification method logistic regression is used to predict the likelihood of a target variable. Calculated relapse might be a classification calculation based on directed learning that predicts the likelihood of a target variable. The nature of the target or secondary variable is dichotomous, implying that there are two distinct groups. In simple terms, the subordinate variable is binary in nature, with data written as 1 (represents success/yes) or 0 (represents failure/no). A computed relapse model predicts $P(Y=1)$ as a work of X mathematically. It's one of the few machine learning algorithms that can be used to a variety of classification problems, including spam detection, diabetes forecasting, cancer detection, and so on. Types of logistic regression

Calculated relapse usually entails a double calculated relapse with two target factors, but it can also predict two additional categories of target factors. Regression may be classified into the following types according on the number of categories. –

1. Binary: Binomial A subordinate variable in this type of classification will have two possible sorts, either 1 or 0. For example, these variables might indicate success or disappointment, yes or no, victory or disaster, and so on.

2. Multinomial

In this type of classification, the subordinate variable might have three or more possible unordered sorts, or sorts with no quantitative significance. These elements may, for example, speak to "Type A," "Type B," or "Type C."

3. Ordinal

In this type of classification, the subordinate variable can have three or more possible desired sorts or sorts with quantitative significance. For example, these criteria may be categorised as "poor" or "good," "very good," or "excellent," with scores ranging from 0 to 3.

6.1.3 SUPPORT VECTOR MACHINE

Bolster Vector Machine could be a directed classification calculation where we draw a line between two different categories to distinguish between them. SVM is additionally known as the back vector organize. The most assignment of the

calculation is to discover the foremost adjust line, or hyperplane, which separates information into two classes. An SVM is a calculation that gets input information and returns such a isolating line.

6.1.4 DECISION TREE

A choice tree is a flowchart-like structure in which each inner hub corresponds to a highlight test (e.g., whether a coin flip results in heads or tails), each leaf hub corresponds to a course name (decision made after calculating all highlights), and branches correspond to highlight conjunctions. Choice trees are created using an algorithmic technique that identifies distinct ways to divide a data set based on certain variables. It is one of the most widely used and straightforward approaches of supervised learning. Choice Trees are a non-parametric learning approach that may be used for categorization and relapse problems. Classification trees are tree models in which the goal variable can take a discrete set of values. Relapse trees are choice trees in which the goal variable can take any number of values (normally actual numbers). This is referred to as a Classification And Relapse Tree (CART). at the start of the lesson titles The performance is controlled by the routes from root to leaf that speak to categorization.

6.1.5 NAIVE BAYES

It is dubbed Naive because it accepts that the occurrence of one highlight is independent of the occurrence of other highlights. For example, if a natural product is identified based on colour, shape, and flavour, a reddish, round, and sweet natural product is identified as an apple. As a result, each component contributes only to the recognition that it is an apple, without relying on the others.

It's called Bayes since it's based on the Bayes' Theorem concept. Bayes' Theorem:

Bayes' theorem, often known as Bayes' Rule or Bayes' law, is a mathematical formula for calculating the probability of a hypothesis given previous information. It is conditional probability that determines this.

The formula for Bayes' theorem is given as:

Naïve Bayes Classifier Algorithm

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Where,

$P(A|B) = P(A|B) = P(A|B)$ = Probability in the future: On the observed event B, the probability of hypothesis A.

$P(B|A) = P(B|A) = P(B|A)$ = Probability of Likelihood: Given that the likelihood of a hypothesis is true, the probability of the evidence.

6.2 PYTHON

Python is a translated, high-level, and general-purpose programming dialect. Python is powerfully written, and garbage collected. It underpins different programming ideal models counting organized, object-oriented, and useful programming.

It was made within the late 1980s, and to begin with discharged in 1991, by Guido van Rossum. Python mediators are accessible for numerous working frameworks. It right now ties with Java the moment most popular programming dialect within the world.

Python could be a multi-paradigm programming dialect. Object-oriented programming and organized programming are completely upheld, and numerous of its highlights back utilitarian programming and aspect-oriented programming.

For memory management, Python uses active writing, a mix of reference tracking, and a cycle-detecting garbage collector.

It also emphasises active title selection, which connects strategy and variable names together throughout programme execution. The Python translator is easily enhanced with underused capabilities and data.

Python operates in several steps (Windows, Mac, Linux, Raspberry Pi, etc.). Python has a basic linguistic structure that is similar to that of English. Python's sentence structure allows programmers to write programmes in less lines than other programming languages. Python is based on a translation framework, which means that code may be executed quickly once it has been written. As a result, prototyping

may be done quite quickly. Python can be approached in a procedural, object-oriented, or practical manner.

6.2.1 INSTALLING PYTHON MODULES:

As a prevalent open-source improvement extend, Python has an dynamic supporting community of donors and clients that moreover make their program accessible for other Python engineers to utilize beneath open-source permit terms. pip is the favored installer program. Beginning with Python 3.4, it is included by default with the Python parallel installers

6.3 WHY MACHINE LEARNING?

Machine Learning calculations are able of learning the errands with the accessible chronicled information and are able to anticipate the yield utilizing that verifiable information. expressly It can do the errands without any changes. In expansion to this, we will utilize profound learning procedures to perform computer vision errands such as picture preparing, question discovery etc. Since picture handling and protest discovery can be performed productively utilizing machine learning and as they are portion of our venture, we utilized machine learning strategies.

6.4 INSTALLING STEPS

Anaconda

Anaconda is a distribution of python and r, constrictor may be a dispersion of the Python and R programming dialects for logical computing (information science, machine learning applications, large-scale information preparing, prescient analytics, etc.), that points to disentangle bundle administration and sending. The name "boa constrictor" comes from the name of a wind from Ceylon (Sri Lanka), which John Beam depicted in Latin in his Summation Methodica Animalium (1693) as *serpens indicus bubalinus anacandaia zeylonibus, ides bubalorum alarumed jumentorum membra conterens*.

- **Installing Anaconda**
- Go to Anaconda website in internet.
- Check for windows Individual Edition.

- Click on download button available in the window.
- Now run the executable file and install it.
- **Installing required packages**

There are a few python bundles to be introduced some time recently executing the extend. This will be done utilizing pip (favored installer program) command. To introduce a specific python, package its particular pip command needs to be run in Boa constrictor command provoke. The python bundle and its individual pip command were said underneath.

- Open anaconda prompt.□
- Type “pip install numpy” and click enter.
- Type “pip install pandas” and click enter.
- Type “pip install opencv-python” and click enter.
- Type “pip install scikit-learn” and click enter.
- Type “pip install imutils” and click enter.

6.5 PROCEDURE FOR EXECUTION

1. Open the Anaconda prompt.
2. Change the location of prompt to destination file
3. Now type “Jupyter notebook”.
4. Then Jupyter notebook will open.
5. Now click on the file need to be executed.
6. Then file will be opened in the Jupyter note book.
7. In the Menu bar, click on kernel.
8. After clicking kernel, some options will be displayed, then click on Restart and Run all option.
9. Then that Executable file will be executed.

CHAPTER-7

SAMPLE CODING

7.1 STATIC OBJECT DETECTION:

```
# import the necessary packages
import numpy as np
import argparse
import cv2

#chang this comment
# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()

ap.add_argument("-i", "--image", required=True,
                help="path to input image")

ap.add_argument("-p", "--prototxt", required=True,
                help="path to Caffe 'deploy' prototxt file")

ap.add_argument("-m", "--model", required=True,
                help="path to Caffe pre-trained model")

ap.add_argument("-c", "--confidence", type=float, default=0.2,
                help="minimum probability to filter weak detections")

args = vars(ap.parse_args())

# initialize the list of class labels MobileNet SSD was trained to
# detect, then generate a set of bounding box colors for each class

CLASSES = ["background", "aeroplane", "bicycle", "bird", "boat",
            "bottle", "bus", "car", "cat", "chair", "cow", "diningtable",
            "dog", "horse", "motorbike", "person", "pottedplant", "sheep",
            "sofa", "train", "tvmonitor"]

COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))

# load our serialized model from disk
print("[INFO] loading model...")

net = cv2.dnn.readNetFromCaffe(args["prototxt"], args["model"])

# load the input image and construct an input blob for the image
# by resizing to a fixed 300x300 pixels and then normalizing it
# (note: normalization is done via the authors of the MobileNet SSD
# implementation)
```

```

image = cv2.imread(args["image"])
(h, w) = image.shape[:2]

blob = cv2.dnn.blobFromImage(cv2.resize(image, (300, 300)), 0.007843, (300,
300), 127.5)

# pass the blob through the network and obtain the detections and
# predictions
print("[INFO] computing object detections...")

net.setInput(blob)

detections = net.forward()

# loop over the detections
for i in np.arange(0, detections.shape[2]):

    # extract the confidence (i.e., probability) associated with the
    # prediction

    confidence = detections[0, 0, i, 2]

    # filter out weak detections by ensuring the `confidence` is
    # greater than the minimum confidence

    if confidence > args["confidence"]:
        # extract the index of the class label from the `detections`,
        # then compute the (x, y)-coordinates of the bounding box for
        # the object

        idx = int(detections[0, 0, i, 1])

        box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
        (startX, startY, endX, endY) = box.astype("int")

        # display the prediction
        label = "{}: {:.2f}%".format(CLASSES[idx], confidence * 100)
        print("[INFO] {}".format(label))

        cv2.rectangle(image, (startX, startY), (endX, endY),
            COLORS[idx], 2)

        y = startY - 15 if startY - 15 > 15 else startY + 15

```

```
cv2.putText(image, label, (startX, y),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, COLORS[idx], 2)

# show the output image

cv2.imshow("Output", image)
cv2.waitKey(0)

# python deep_learning_object_detection.py --image images/example_01.jpg --
prototxt MobileNetSSD_deploy.prototxt.txt --model
MobileNetSSD_deploy.caffemodel
```

7.2 REAL TIME OBJECT DETECTION:

```
# import the necessary packages
from imutils.video import VideoStreamS
from imutils.video import FPS
import numpy as np
import argparse
import imutils
import time
import cv2

# construct the argument parse and parse the arguments

ap = argparse.ArgumentParser()

ap.add_argument("-p", "--prototxt", required=True,
                help="path to Caffe 'deploy' prototxt file")

ap.add_argument("-m", "--model", required=True,
                help="path to Caffe pre-trained model")

ap.add_argument("-c", "--confidence", type=float, default=0.2,
                help="minimum probability to filter weak detections")

args = vars(ap.parse_args())

# initialize the list of class labels MobileNet SSD was trained to
# detect, then generate a set of bounding box colors for each class

CLASSES = ["background", "aeroplane", "bicycle", "bird", "boat",
            "bottle", "bus", "car", "cat", "chair", "cow", "diningtable",
            "dog", "horse", "motorbike", "person", "pottedplant", "sheep",
            "sofa", "train", "tvmonitor"]

COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))

# load our serialized model from disk
print("[INFO] loading model...")

net = cv2.dnn.readNetFromCaffe(args["prototxt"], args["model"])

# initialize the video stream, allow the cammera sensor to warmup,
# and initialize the FPS counter

print("[INFO] starting video stream...")
```



```

vs = VideoStream(src=0).start()
time.sleep(2.0)
fps = FPS().start()

# loop over the frames from the video stream
while True:
    # grab the frame from the threaded video stream and resize it
    # to have a maximum width of 400 pixels

    frame = vs.read()

    frame = imutils.resize(frame, width=400)

    # grab the frame dimensions and convert it to a blob
    (h, w) = frame.shape[:2]

    blob = cv2.dnn.blobFromImage(cv2.resize(frame, (300, 300)),
        0.007843, (300, 300), 127.5)

    # pass the blob through the network and obtain the detections and
    # predictions

    net.setInput(blob)
    detections = net.forward()

    # loop over the detections
    for i in np.arange(0, detections.shape[2]):
        # extract the confidence (i.e., probability) associated with
        # the prediction

        confidence = detections[0, 0, i, 2]

        # filter out weak detections by ensuring the `confidence` is
        # greater than the minimum confidence

        if confidence > args["confidence"]:

            # extract the index of the class label from the
            # `detections`, then compute the (x, y)-coordinates of
            # the bounding box for the object

            idx = int(detections[0, 0, i, 1])

            box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
            (startX, startY, endX, endY) = box.astype("int")

```

```

# draw the prediction on the frame
    label = "{}: {:.2f}%".format(CLASSES[idx],
                                confidence * 100)

    cv2.rectangle(frame, (startX, startY), (endX, endY),
                  COLORS[idx], 2)

    y = startY - 15 if startY - 15 > 15 else startY + 15

    cv2.putText(frame, label, (startX, y),
                cv2.FONT_HERSHEY_SIMPLEX, 0.5, COLORS[idx], 2)

# show the output frame
cv2.imshow("Frame", frame)

key = cv2.waitKey(1) & 0xFF

# if the `q` key was pressed, break from the loop
if key == ord("q"):
    break

# update the FPS counter
fps.update()

# stop the timer and display FPS information
fps.stop()
print("[INFO] elapsed time: {:.2f}".format(fps.elapsed()))
print("[INFO] approx. FPS: {:.2f}".format(fps.fps()))

# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()

# USAGE
# python real_time_object_detection.py --prototxt
MobileNetSSD_deploy.prototxt.txt --model MobileNetSSD_deploy.caffemodel

```

CHAPTER-8

SYSTEM TESTING

8.1 BLACK BOX TESTING

Black box testing is a type of testing that ignores the system's internal mechanisms and focuses solely on the output created in response to any input and execution. It's also referred to as functional testing.

Black-box testing is a type of software testing that looks at an application's functioning based on its requirements. Specifications-based testing is another name for it. During the software testing life cycle, the Independent Testing Team normally performs this sort of testing.

This method of test can be applied to each and every level of software testing such as unit, integration, system and acceptance testing.

This test technique may be used at all levels of software testing, including unit, integration, system, and acceptability testing.

Behavioral Testing Techniques

There are different techniques involved in Black Box testing.

1. Equivalence Class
2. Boundary Value Analysis
3. Domain Tests
4. Orthogonal Arrays
5. Decision Tables
6. State Models
7. Exploratory Testing
8. All-pairs testing

Black box testing is a sort of software testing when the software's functioning is unknown. The testing is carried out with no knowledge of the items' internal workings.

Black box testing can be done in following ways

- 1. Syntax Driven Testing:** This form of testing is used on systems that can be represented syntactically by a language. Compilers, for example, are a type of language that may be described using context-free grammar. The test cases are created in such a way that each grammatical rule is applied at least once.
- 2. Equivalence partitioning:** It is commonly seen that a variety of inputs act fundamentally in the same way, thus rather than providing them all their own inputs, they can be grouped together and tested as if there were one input of each gather. The idea is to divide the framework's input space into a number of proportionality classes such that each section of the course operates in the same manner, i.e., if one test case in one course results in a few errors, other test cases in other courses will also make the same mistake.

The technique involves two steps:

- 3. Identification of equivalence class:** Divide every input domain into at least two groups: valid and invalid values. Select one valid input, such as 49, and one invalid input, such as 104, if the valid range is 0 to 100.
- 4. Generating test cases:**

I Assign a unique identification number to each valid and incorrect input class. (ii) Create a test case that covers all valid and invalid test cases, keeping in mind that no two invalid inputs are the same. (iii) Equivalence classes will be used to determine the square root of an integer.

Valid inputs:

The result will be an integer if the whole number is a perfect square. The result will be a decimal number if the whole integer is not a perfect square.

Decimals with a positive

value **Invalid inputs :**

Numbers that are negative (integer or decimal).

Characters other than numerals, such as "a,"!",",;" and so on

5. boundary values: Boundaries are a particularly good area for mistakes to happen. As a result, if test cases are created for boundary values of input space, testing proficiency improves and the possibility of identifying errors increases as well. For instance, if the significant range is 10 to 100, test for 10,100.

6. Cause effect Graphing: This approach builds a link between logical input, referred to as causes, and related actions, referred to as effects. Boolean graphs are used to depict the causes and consequences. The following are the measures to take:

Identify the inputs (causes) and outputs (effects) (effect).

Create a cause-and-effect diagram.

Convert the graph to a decision table.

Convert test cases to decision table rules.

Requirement based testing: It entails confirming the requirements specified in the software system's SRS.

Compatibility testing: The test case outcomes are influenced not just by the product, but also by the infrastructure used to supply functionality. It is intended that when the infrastructure settings are modified, it would continue to function appropriately. Processor (Pentium 3, Pentium 4) and number of processors are two factors that influence software compatibility.

Machine architecture and characteristics (32 bit or 64 bit).

Database servers and other back-end components.

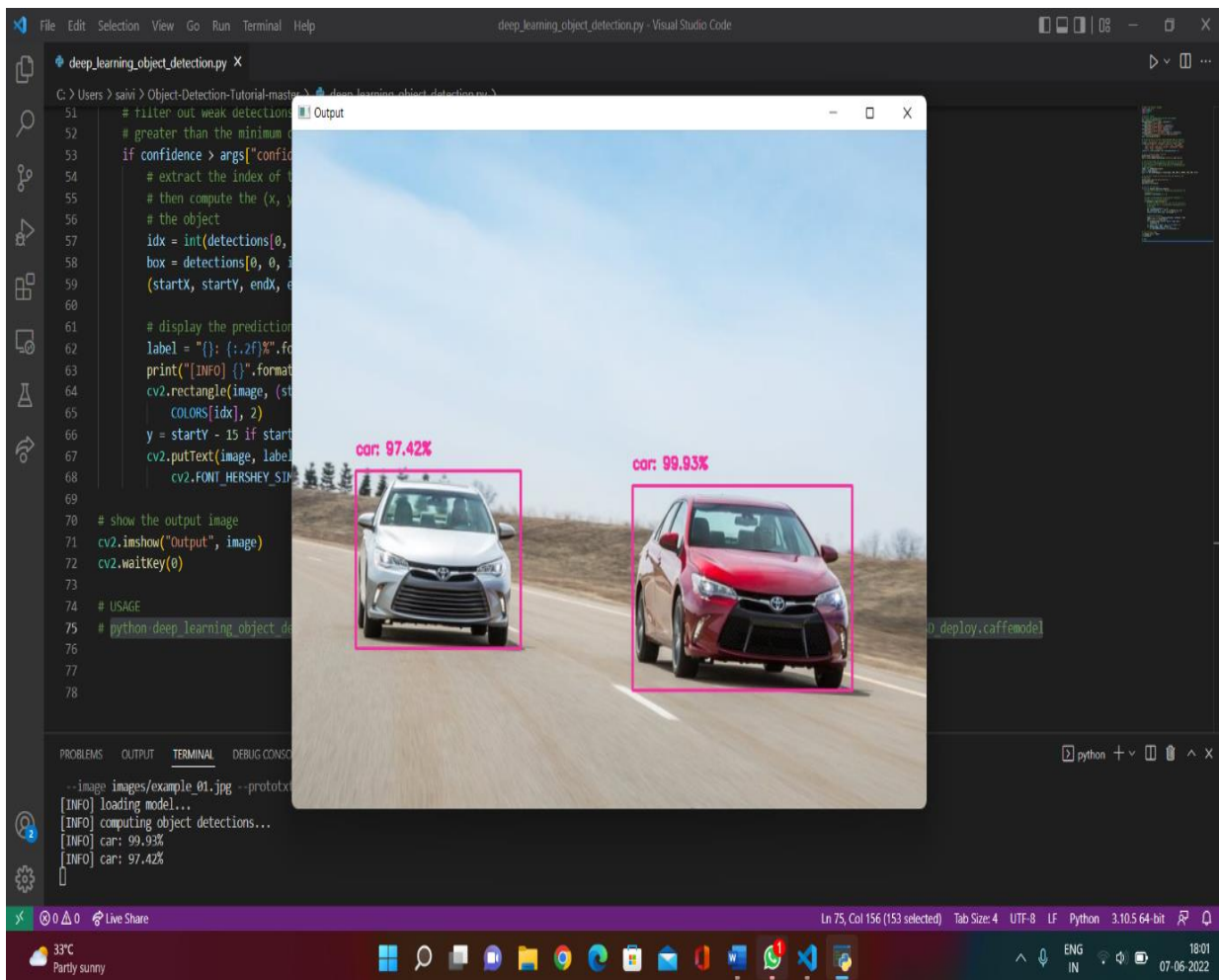
7. System of Operation (Windows, Linux, etc.)

TEST CASE NO:	Test conditions	Anticipated result	Final result
Case 1:	When photo is chosen as an input	Objects are surrounded by a bounding box, and their trajectory is projected.	Successful
Case 2:	When video is chosen as input	Predicted teachings and a video with a bounding box around the items.	Successful
Case 3:	When camera chosen as input	Object detection with bounding box, confidence score, and estimated course in real time.	successful
Case 4:	When black and white image is taken as input	Objects are surrounded by a bounding box, and their trajectory is projected.	successful
Case 5:	Picture with far long object is taken as input	Picture with detected objects	unsuccessful
Case 6:	When picture with overlapping object is Taken as input	Objects are surrounded by a bounding box, and their class is anticipated.	successful
Case 7:	When picture with far objects is taken as input	Objects that were spotted in the image	unsuccessful

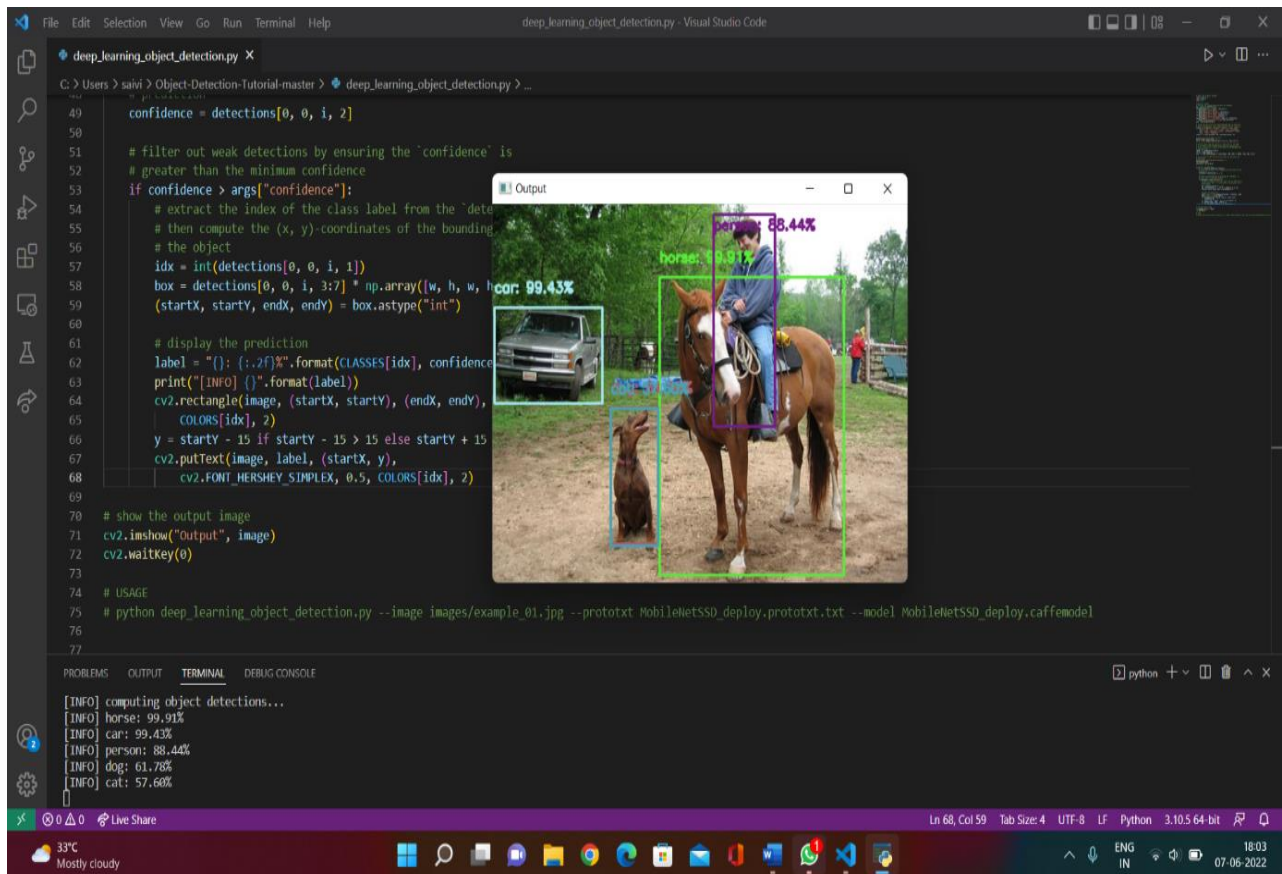
CHAPTER 9

OUTPUT SCREENS

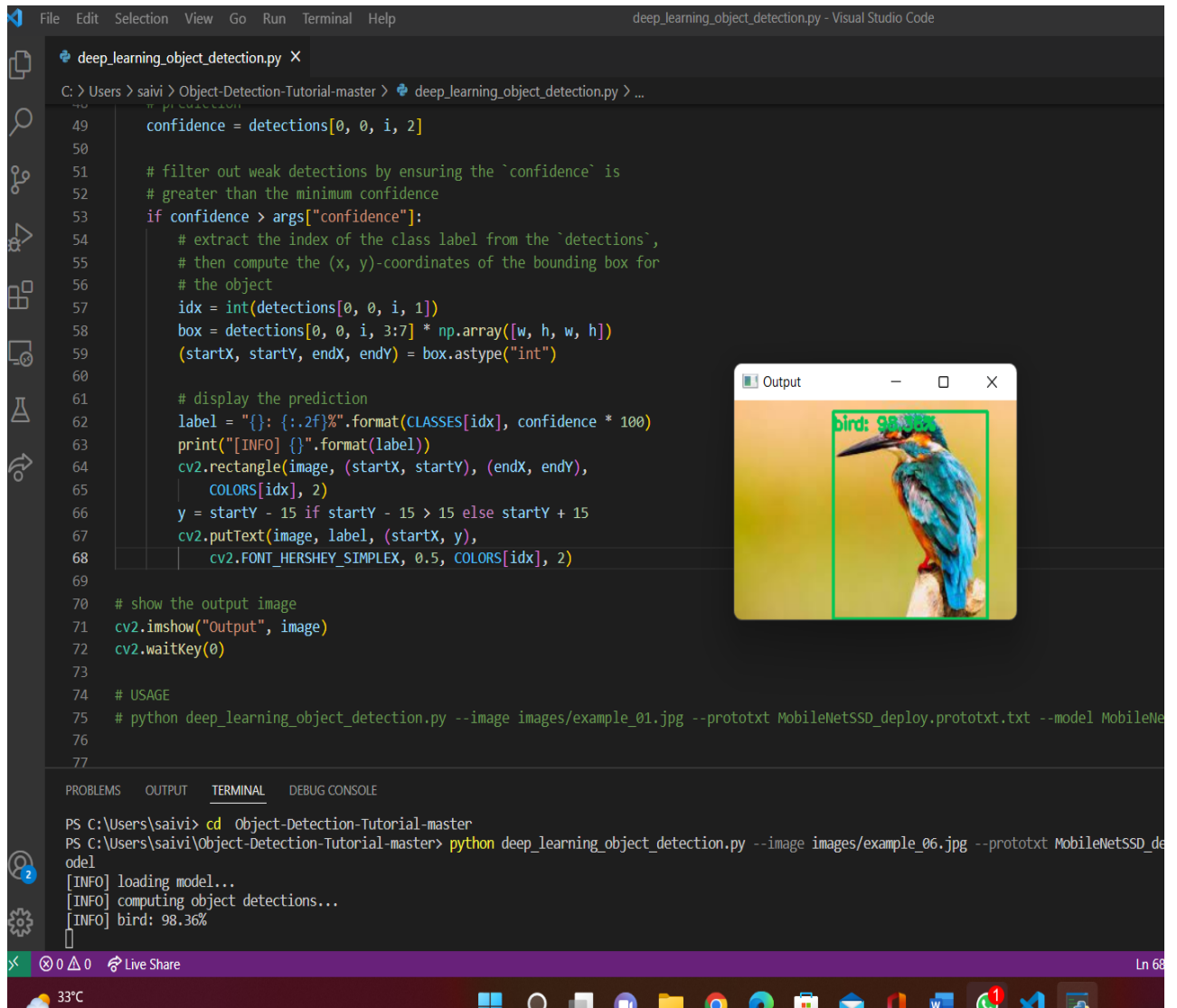
9.1 OUTPUT1:



9.2 OUTPUT2:



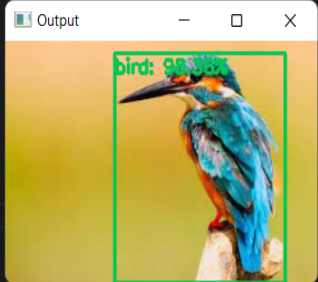
9.3 OUTPUT3:



The screenshot displays the Visual Studio Code interface with a Python script named `deep_learning_object_detection.py` open in the editor. The script is designed to perform object detection using a pre-trained model. It includes comments explaining the steps: filtering weak detections, extracting class labels and bounding boxes, and displaying the results. The script uses OpenCV for image processing and NumPy for array operations. A small window titled "Output" is visible, showing a detected bird with a bounding box and the label "bird: 98.36%". The terminal at the bottom shows the command to run the script and the resulting output, including the model loading and detection process.

```
48 # prediction
49 confidence = detections[0, 0, i, 2]
50
51 # filter out weak detections by ensuring the `confidence` is
52 # greater than the minimum confidence
53 if confidence > args["confidence"]:
54     # extract the index of the class label from the `detections`,
55     # then compute the (x, y)-coordinates of the bounding box for
56     # the object
57     idx = int(detections[0, 0, i, 1])
58     box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
59     (startX, startY, endX, endY) = box.astype("int")
60
61     # display the prediction
62     label = "{}: {:.2f}%".format(CLASSES[idx], confidence * 100)
63     print("[INFO] {}".format(label))
64     cv2.rectangle(image, (startX, startY), (endX, endY),
65                   COLORS[idx], 2)
66     y = startY - 15 if startY - 15 > 15 else startY + 15
67     cv2.putText(image, label, (startX, y),
68                 cv2.FONT_HERSHEY_SIMPLEX, 0.5, COLORS[idx], 2)
69
70 # show the output image
71 cv2.imshow("Output", image)
72 cv2.waitKey(0)
73
74 # USAGE
75 # python deep_learning_object_detection.py --image images/example_01.jpg --prototxt MobileNetSSD_deploy.prototxt.txt --model MobileNet
76
77
```

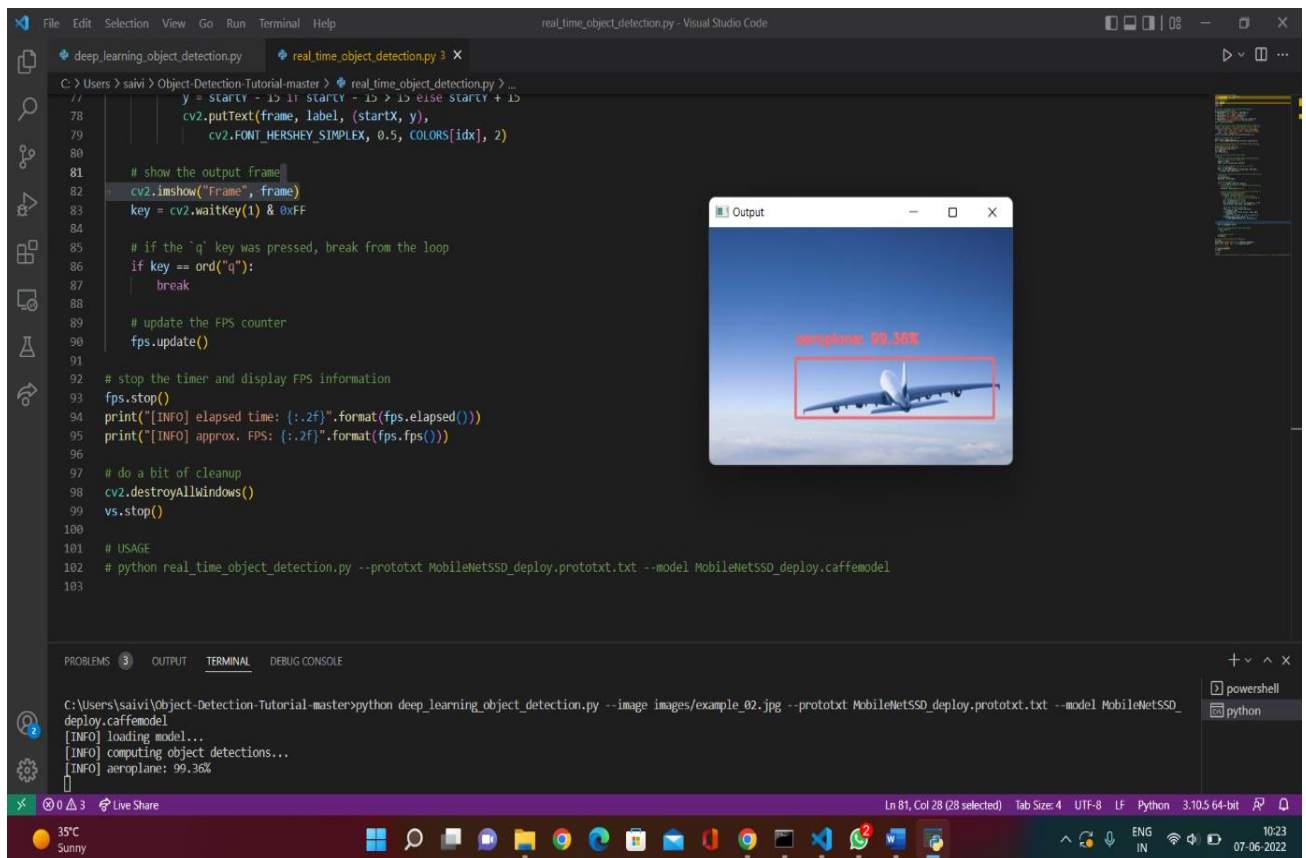
Output window:



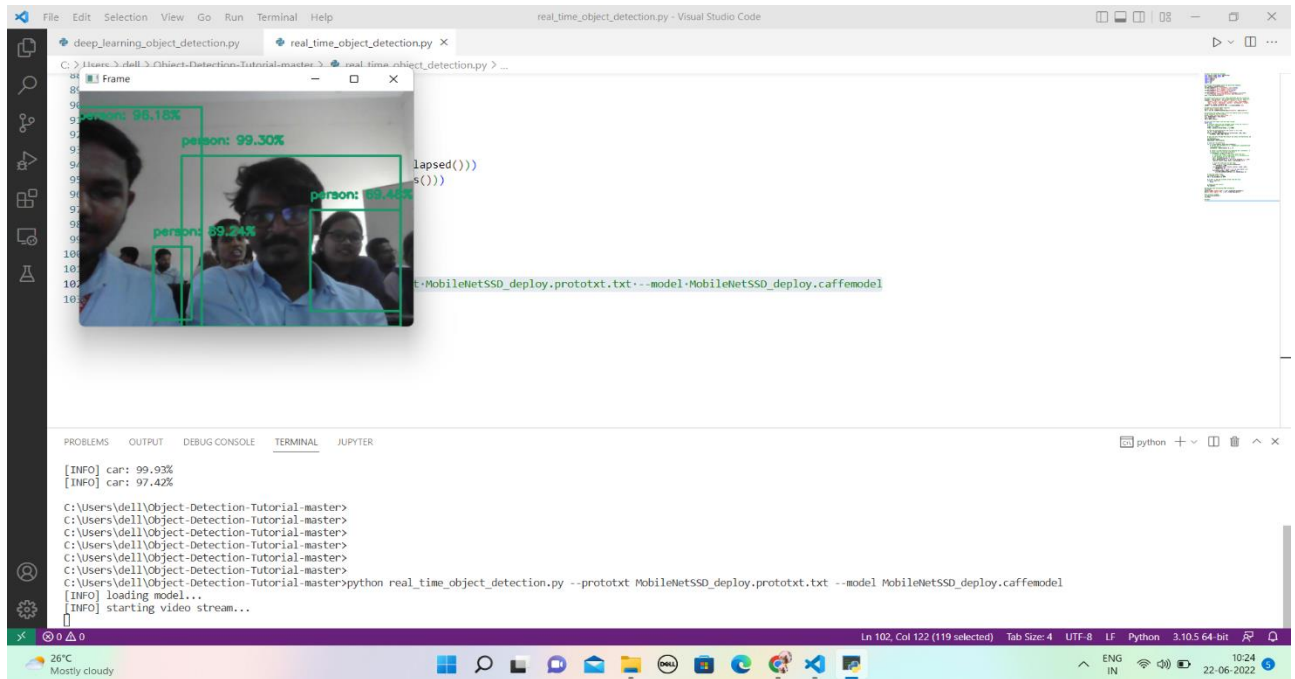
Terminal:

```
PS C:\Users\saivi> cd Object-Detection-Tutorial-master
PS C:\Users\saivi\Object-Detection-Tutorial-master> python deep_learning_object_detection.py --image images/example_06.jpg --prototxt MobileNetSSD_de
odel
[INFO] loading model...
[INFO] computing object detections...
[INFO] bird: 98.36%
```

9.4 OUTPUT4:



9.4 OUTPUT 5:



CHAPTER-10

CONCLUSION AND FUTER SCOPE

10.1 CONCLUSION

Object detection Open CV API is one of the best tool to detect the real-time objects which includes the objects either in static or in dynamic in nature.

As we are using AI application this project will be made with the best accuracy and precision

10.2 FUTURE SCOPE

Machine Learning's use isn't restricted to finance. Rather, it is spreading across a wide range of industries, including banking and finance, information technology, media and entertainment, gaming, and the automobile sector. Because the breadth of Machine Learning is so broad, there are several areas where academics are striving to change the world for the better in the future.

CHAPTER 11

BIBLIOGRAPHY

- [Real-Time Object Detection Using Open CV - Great Learning \(mygreatlearning.com\)](#)
- [Object Detection Tutorial using Open CV | Real-Time Object Detection | Edureka](#)
- [Understanding SSD MultiBox — Real-Time Object Detection In Deep Learning | by Eddie Forson | Towards Data Science](#)