Object Detection

Mini- Project (IPMV)

Submitted in partial fulfillment of the requirement of University of Mumbai
For the Degree of

(Electronics & Telecommunication)

By

1) HEMANTKUMAR GUPTA	**ID No: TU2F1819096**
2) SUDHANSHU SHUKLA	**ID No: TU2F1718102**
3) HITESH GOWDA	**ID No: TU2F1819092**
4) ANIKET GUPTA	**ID No: TU2F1819095**

Under the Guidance of

PRAVIN DERE

Department of Electronics & Telecommunication Engineering
TERNA ENGINEERING COLLEGE
Plot no.12, Sector-22, Opp. Nerul Railway station,
Phase-11, Nerul (w), Navi Mumbai 400706
UNIVERSITY OF MUMBAI

Terna Engineering College

NERUL, NAVI MUMBAI

CERTIFICATE

This is to certify that

1) HEMANTKUMAR GUPTA	**ID No: TU2F1819096**
2) SUDHANSHU SHUKLA	**ID No: TU2F1718102**
3) HITESH GOWDA	**ID No: TU2F1819092**
4) ANIKET GUPTA	**ID No: TU2F1819095**

Has satisfactorily completed the requirements of the ***Mini Project***

Of subject

Image Processing & Machine Vision Lab (IPMV)

As prescribed by the ***University of Mumbai*** *Under the guidance of*

Prof. **PRAVIN DERE**

Subject Incharge **APC** **HOD**

INDEX

TABLE OF CONTENTS		
Caption		**Page No.**
Chapter 1	Introduction	4
Chapter 2	Problem Statement	5-9
Chapter 3	Methodology	10-11
Chapter 4	Implementation	12-15
Chapter 5	Algorithm & Flow-Chart	16-19
Chapter 6	Conclusion	20
	Reference	20

CHAPTER 1

INTRODUCTION

Object Detection is the process of finding and recognizing real-world object instances such as car, bike, TV, flowers, and humans out of an images or videos. An object detection technique lets you understand the details of an image or a video as it allows for the recognition, localization, and detection of multiple objects within an image.

It is usually utilized in applications like image retrieval, security, surveillance, and advanced driver assistance systems (ADAS).Object Detection is done through many ways:

- Feature Based Object Detection
- Viola Jones Object Detection
- SVM Classifications with HOG Features
- Deep Learning Object Detection

Humans can easily detect and identify objects present in an image. The human visual system is fast and accurate and can perform complex tasks like identifying multiple objects with little conscious thought. With the availability of large amounts of data, faster GPUs, and better algorithms, we can now easily train computers to detect and classify multiple objects within an image with high accuracy.

CHAPTER 2

PROBLEM STATEMENT

Here, the HOG feature extraction method is used for both SVM and ANN to derive accuracy and performance. The deep learning method is used for CNN to derive accuracy and performance. The results obtained from the above methods are compared and method with high accuracy and high performance is considered as the best method for handwritten digits. There is a huge number of studies conducted in the field of handwritten digits. Out of them, the parameters which are considered here i.e. SVM, ANN, CNN are the most popular ones. Usually, segmentation and classification phases are the most challenging and play a vital role in the handwritten digit recognition process. Because in segmentation the image is broken into multiple images, each described as an individual digit and proper classification is done to every individual digit

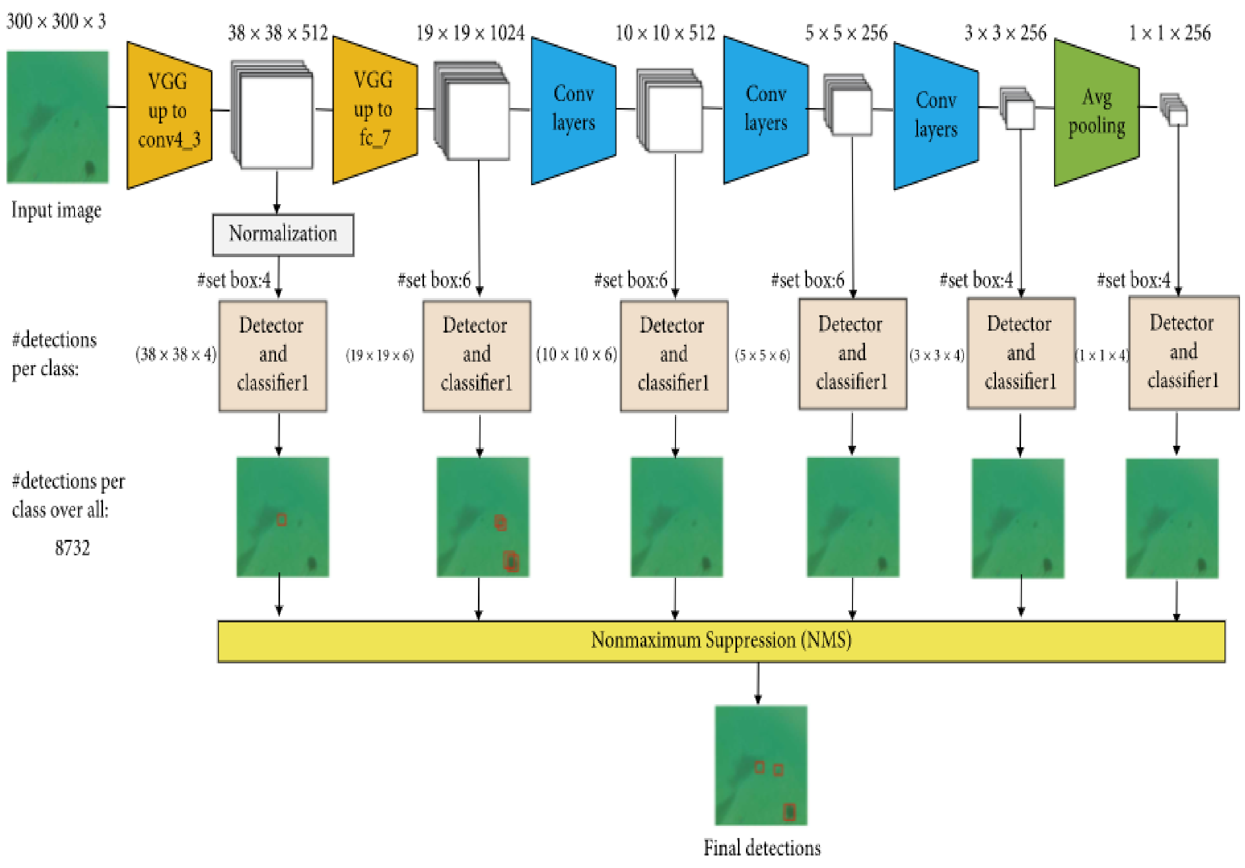
To overcome the challenges faced in segmentation and classification, some rules are implemented to increase the accuracy and performance of both segmentation and classification. The following rules are implemented in segmentation and classification.

- Water reservoir concept is used for segmentation.
- SVM, ANN, CNN concepts are used in classification.

BACKGROUND :

The convolutional neural network is a type of feedforward neural network, and it includes convolution calculation and has a deep structure. It has three core ideas: local network connection, convolution kernel parameter sharing, and pooling. The joint effect of local connection and parameter sharing is to reduce the number of parameters, make the operation simple and efficient, and be able to operate on very large data sets. Pooling is to aggregate the characteristics of different locations to obtain lower dimensions, and it can prevent the problem of overfitting. On this basis, a slight adjustment to the network framework can improve the generalization ability and robustness of the model.

The Single-Shot MultiBox Detector (SSD) algorithm is a one-stage algorithm; it is one of the most real-time and advanced target detection algorithms at present. The SSD algorithm has completed the task of classifying and locating the target using only a full-convolution network. The structural framework of SSD is shown in Figure. It uses VGG16 as the basic architecture and introduces the design concept of a prior frame. On the basis of VGG16, a new cascaded convolution layer is added to obtain multiscale feature maps to detect the target. All the prediction results are merged together, and the final detection result is obtained by Nonmaximum Suppression (NMS).



Machine Learning :

According to Arthur Samuel, "Machine learning is a subfield of computer science which gives computers the ability to learn without being explicitly programmed". This study helps in predicting and learning from the data imported with the help of algorithms implemented. Machine learning is used where there is difficulty in programming tasks instead machine learning algorithms are used to achieve the task. Some of these tasks include Identity Fraud Detection, computer vision, population Growth Prediction, email filtering, Weather forecasting, OCR (optical character recognition), Diagnostics, real-time decisions etc.

Machine learning concepts are classified into three categories:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Supervised Learning:

Consider, a dataset is given as input and assumptions can be made on the output data how it looks like. In supervised learning, there’s a relationship between the input data and the output data. The output can be predicted with the input given.

Unsupervised Learning:

Unsupervised learning is an approach where the algorithm has to identify the hidden patterns in the given input. So, the algorithm works without any guidance as the input data is not labeled or classified.

Reinforcement Learning:

Reinforcement learning is a suitable action to maximize reward in a particular situation. It is to find the best possible behavior or path it should take in a specific situation.

Deep Learning :

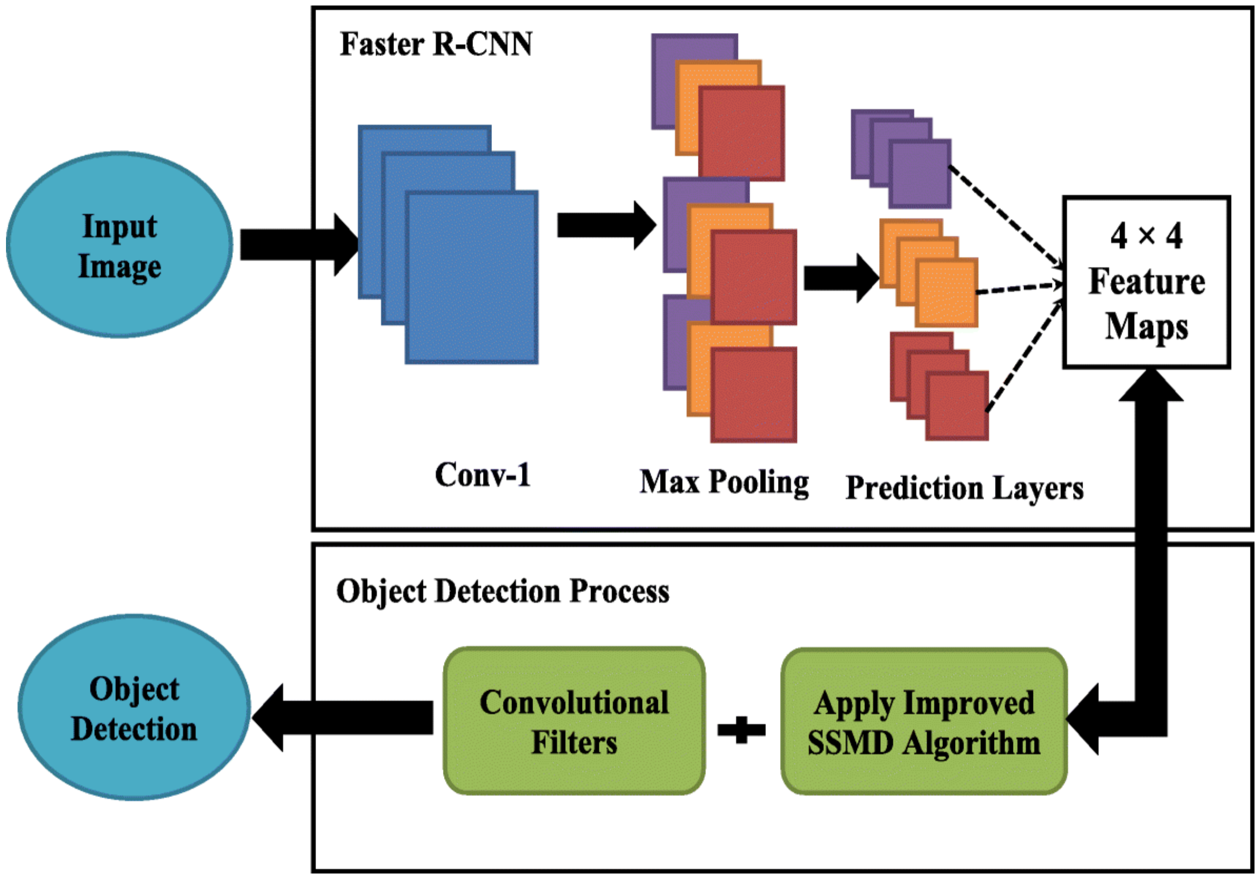
According to Arthur Andrew Ng, “Deep Learning is a superpower. With it, you can make a computer see, synthesize novel art, translate languages, render a medical diagnosis, or build pieces of a car that can drive itself. If that isn’t a superpower, I don’t know what is”.

Deep learning is a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised or unsupervised. It is a set of algorithms in machine learning to learn multiple levels of representation, corresponding to different layers of abstraction that help to make sense of data. Many layers are used to compute nonlinear functions with highly complex data. Each layer gets its input from a preceding layer, then it computes and transforms the data and sends it to the further layers. Each layer in a network consists of neurons and has various modes of connections to other neurons in the same layer as well as to those of other layers depending on the type of network. The whole idea of deep learning is using brain simulations, helping to make learning algorithms more efficient to use and revolutionary advances in machine learning and Artificial Intelligence. Nowadays deep learning gets more attention with development of modern technologies and easy execute it.

CHAPTER 3

METHODOLOGY

This section presents our proposed approach for detecting the objects in real-time from images by using convolutional neural network deep learning process. The previous algorithms such as CNN, faster CNN, faster RCNN, YOLO, and SSD are only suitable for highly powerful computing machines and they require a large amount of time to train. In this paper, we have tried to overcome the limitations of the SSD algorithm by introducing an improved SSD algorithm with some improvement. The proposed scheme uses improved SSD algorithm for higher detection precision with real-time speed. However, SSD algorithm is not appropriate to detect tiny objects, since it overlooks the context from the outside of the boxes. To address this issue, the proposed algorithm uses depth-wise separable convolution and spatial separable convolutions in their convolutional layers. Specifically, our proposed approach uses a new architecture as a combination of multilayer of convolutional neural network. The algorithm comprises of two phases. First, it reduces the feature maps extraction of spatial dimensions by using resolution multiplier. Second, it is designed with the application of small convolutional filters for detecting objects by using the best aspect ratio values. The major objective during the training is to get a high-class confidence score by matching the default boxes with the ground truth boxes. The advantage of having multi-box on multiple layers leads to significant results in detection. Single shot multi-box detector was discharged at the tip of Gregorian calendar month 2016 and thus arrived at a new set of records on customary knowledge sets like Pascal VOC and COCO. The major problem with the previous methods was how to recover the fall in precision, for which SSD applies some improvements that include multi-scale feature map and default boxes. For detecting a small object with higher resolutions, feature maps are used. The training set of improved SSD algorithm depends upon three main sections, i.e., selecting the size of box, matching of boxes, and loss function. The proposed scheme can be understood by the system model given in Fig.

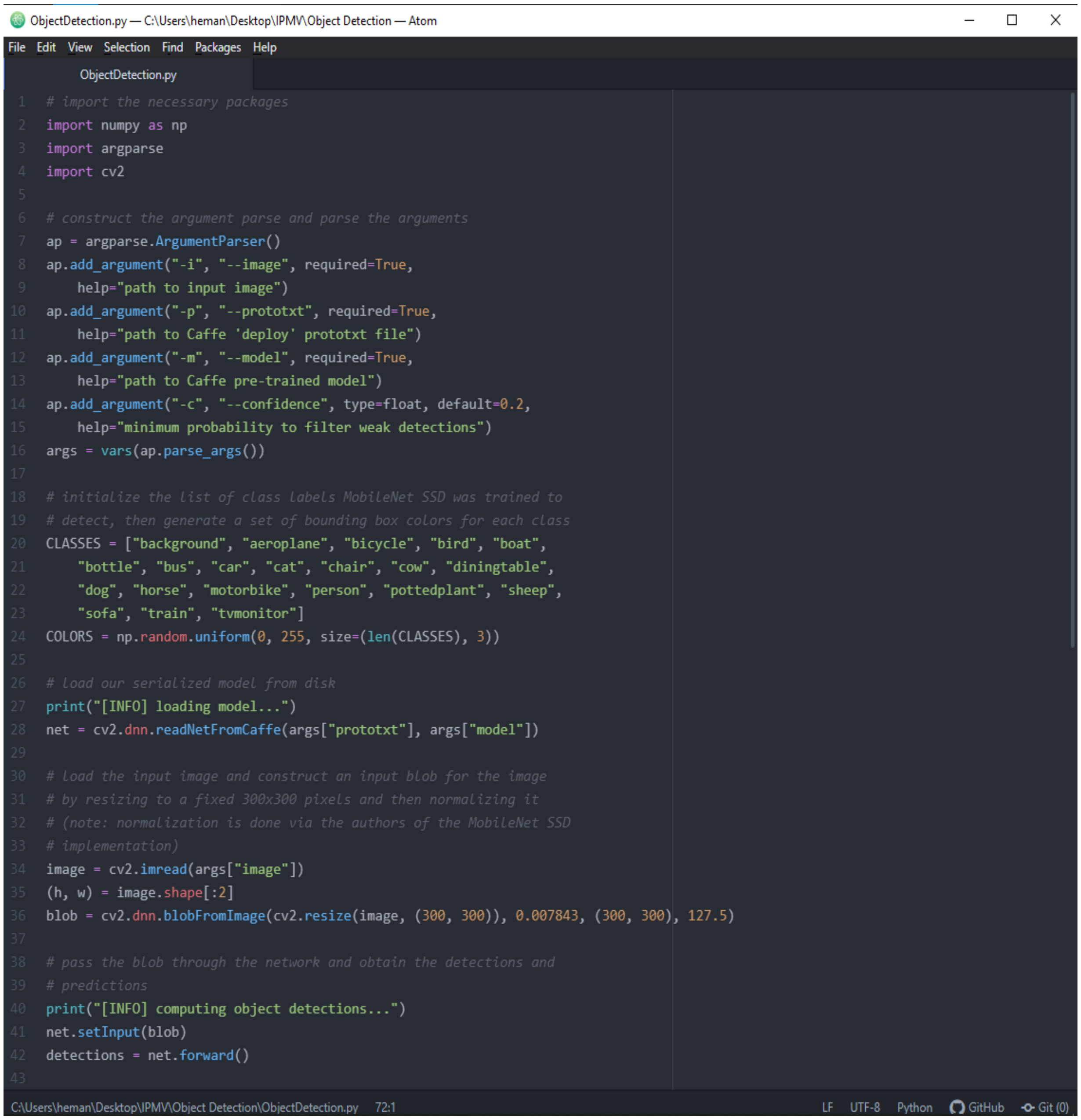


CHAPTER 4

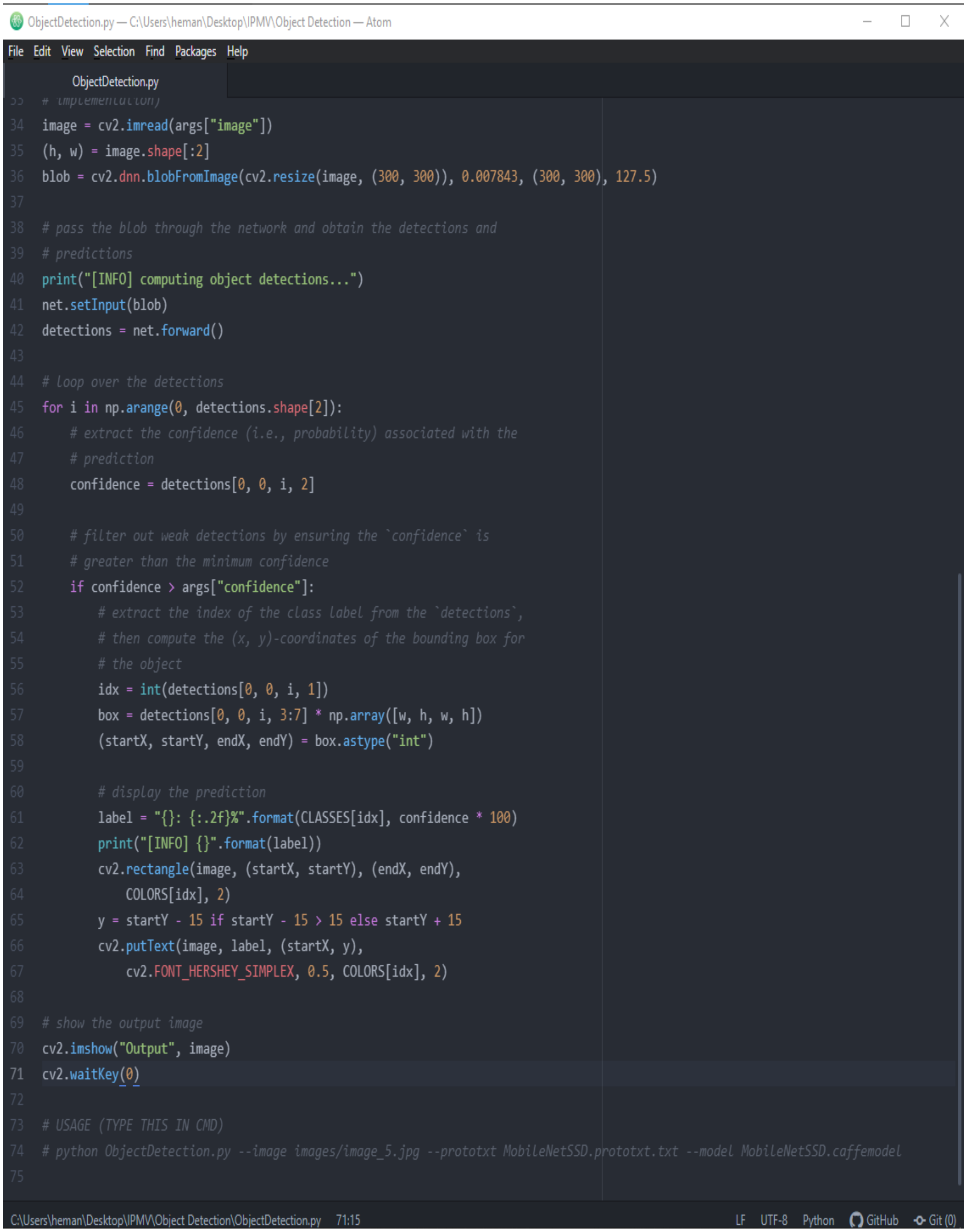
IMPLEMENTATION

PROGRAM : You can download required files from My GitHub account link given below:

https://github.com/hemantcgupta/Object-Detection.git

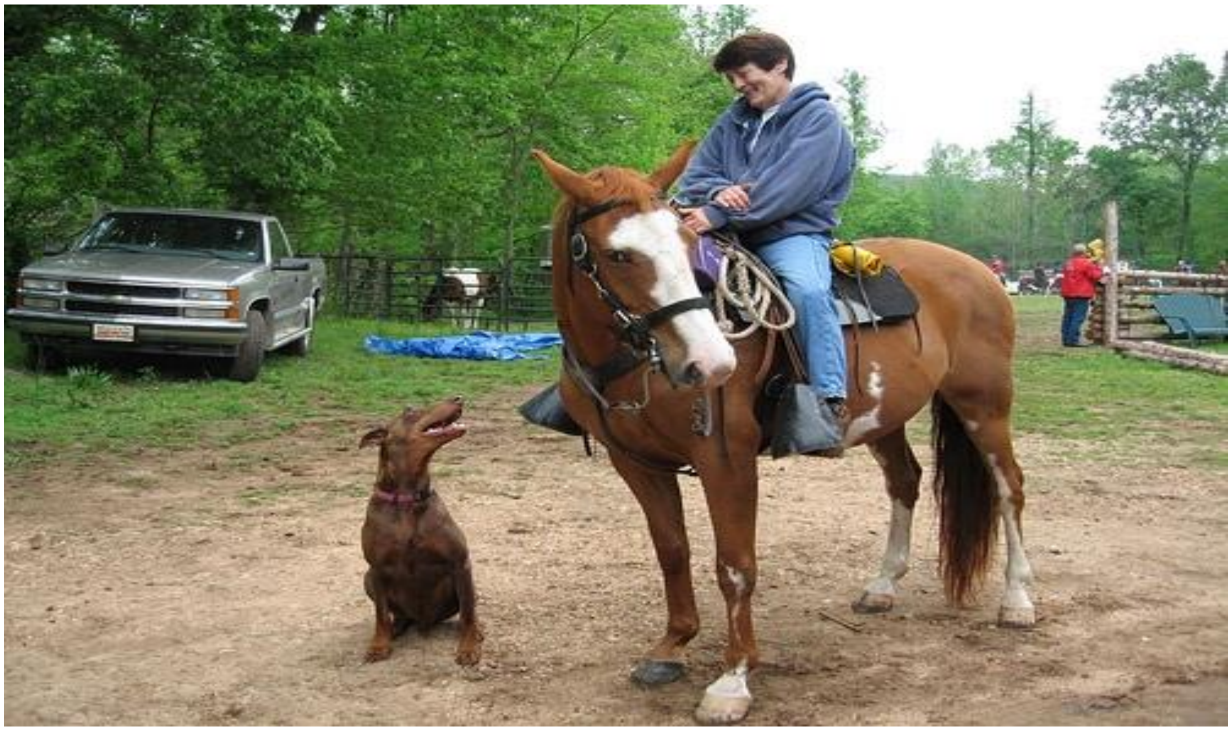

ObjectDetection.py — C:\Users\heman\Desktop\IPMV\Object Detection — Atom
File Edit View Selection Find Packages Help
ObjectDetection.py
C:\Users\heman\Desktop\IPMV\Object Detection\ObjectDetection.py 72:1
LF UTF-8 Python GitHub Git (0)

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

# 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()

```

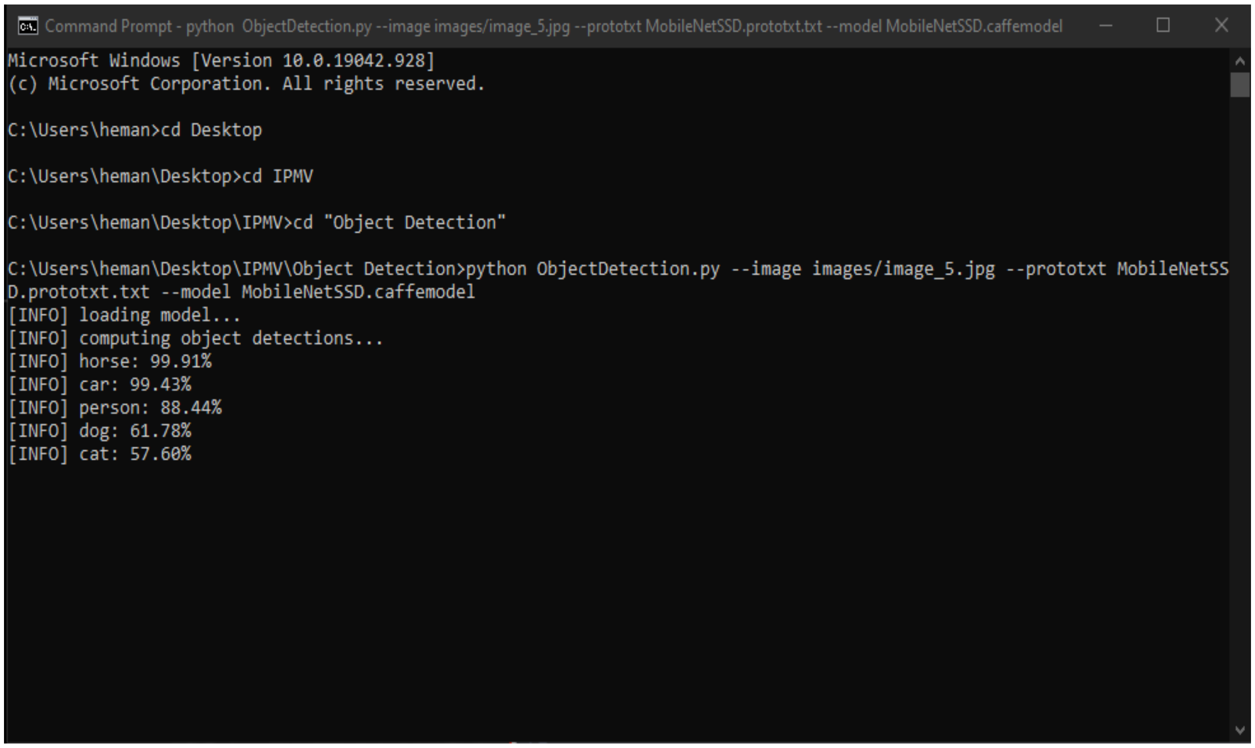

ObjectDetection.py — C:\Users\heman\Desktop\IPMV\Object Detection — Atom
File Edit View Selection Find Packages Help
ObjectDetection.py
C:\Users\heman\Desktop\IPMV\Object Detection\ObjectDetection.py 71:15
LF UTF-8 Python GitHub Git (0)

```python
# 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)

# USAGE (TYPE THIS IN CMD)
# python ObjectDetection.py --image images/image_5.jpg --prototxt MobileNetSSD.prototxt.txt --model MobileNetSSD.caffemodel
```

INPUT :

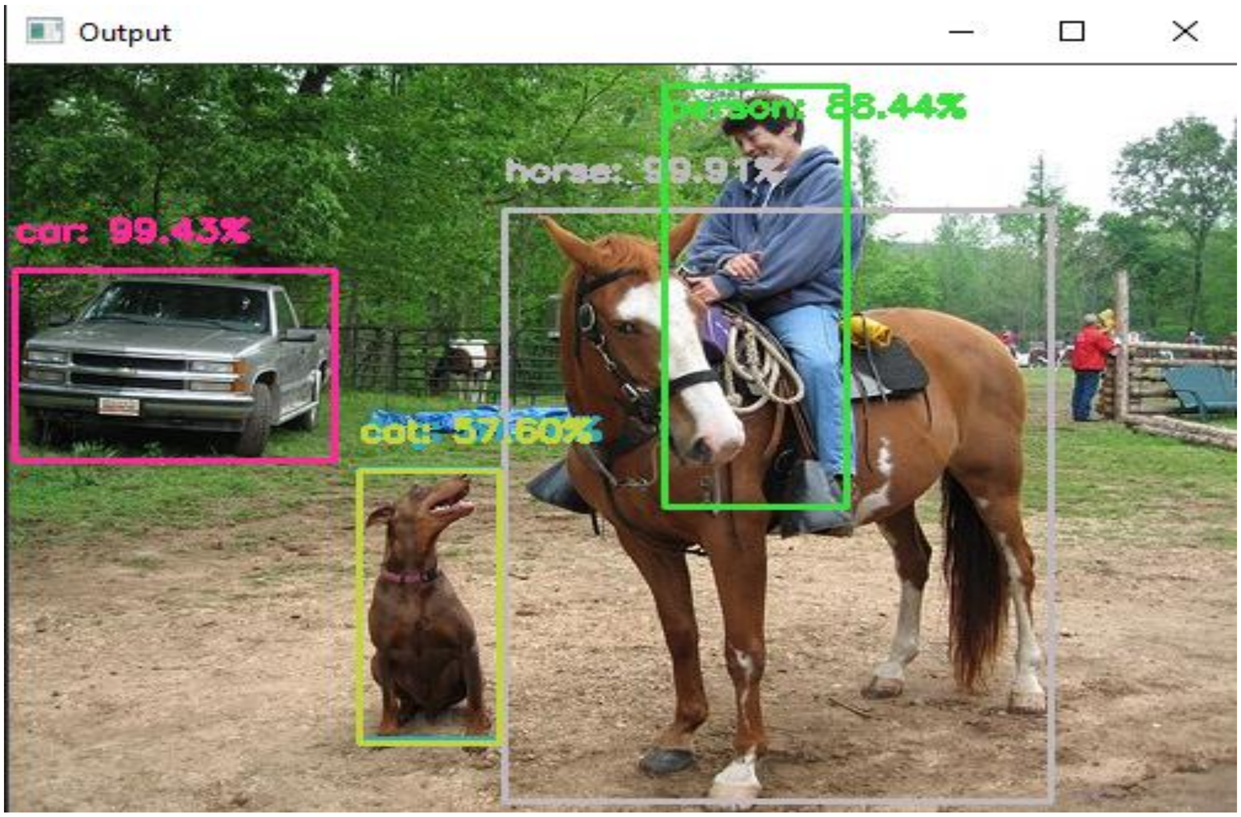


OUTPUT :



Command Prompt - python ObjectDetection.py --image images/image_5.jpg --prototxt MobileNetSSD.prototxt.txt --model MobileNetSSD.caffemodel

```
Microsoft Windows [Version 10.0.19042.928]
(c) Microsoft Corporation. All rights reserved.

C:\Users\heman>cd Desktop

C:\Users\heman\Desktop>cd IPMV

C:\Users\heman\Desktop\IPMV>cd "Object Detection"

C:\Users\heman\Desktop\IPMV\Object Detection>python ObjectDetection.py --image images/image_5.jpg --prototxt MobileNetSS
D.prototxt.txt --model MobileNetSSD.caffemodel
[INFO] loading model...
[INFO] computing object detections...
[INFO] horse: 99.91%
[INFO] car: 99.43%
[INFO] person: 88.44%
[INFO] dog: 61.78%
[INFO] cat: 57.60%
```


Output
person: 88.44%
horse: 99.91%
car: 99.43%
cat: 57.60%

CHAPTER 5

ALGORITHM & FLOWCHART

SSMBD Algorithm :

In order to interpret the role of SSD algorithm, we first formally denote the following concepts.

Single shot:

This means that the tasks of the thing localization and classification are exhausted one passing play of the network.

Multi-box:

Ground truth box and predicted box are the boxes in multi-box. This is introduced by Szegedy.

Detector:

The network is an associate degree object detector that conjointly classifies those detected objects.

Default the size of the boxes:

The selection of boxes is based on the minimum value of convolution layer and maximum values of change in intensity. The first algorithm represents the procedure of producing specified feature maps F(m).

Truth boxes:

After finding the size of boxes, the next phase is matching of the boxes with the corresponding truth boxes. A specific given picture to identify the truth boxes is explained in the second algorithm.

Loss function:

The loss function is unbelievably simple, and it is a methodology of evaluating how well your role models your dataset. If your predictions are entirely of your loss function, it can operate next range. If the output range is less, it means that the model is good. The main objective is to minimize loss function. The loss function is also depending upon the sum of weighted localization and classification loss functions .

When a color image is fed into the input layer, SSD does the following.

Step 1: Image is passed through large number of convolutional layers extracting feature maps at different points.

Step 2: Every location in each of those feature maps uses a 4x4 filter to judge a tiny low default box.

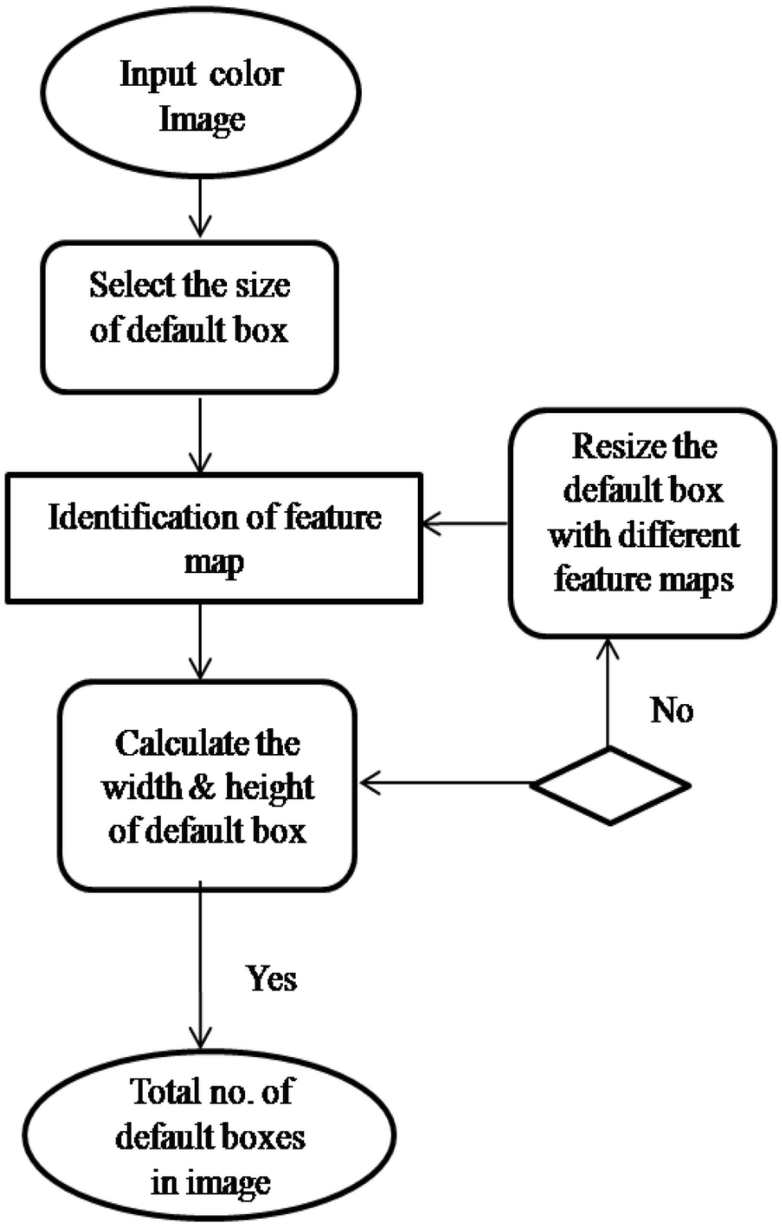
Step 3: Predict the bounding box offset for each box.

Step 4: Predict the class probabilities for each box.

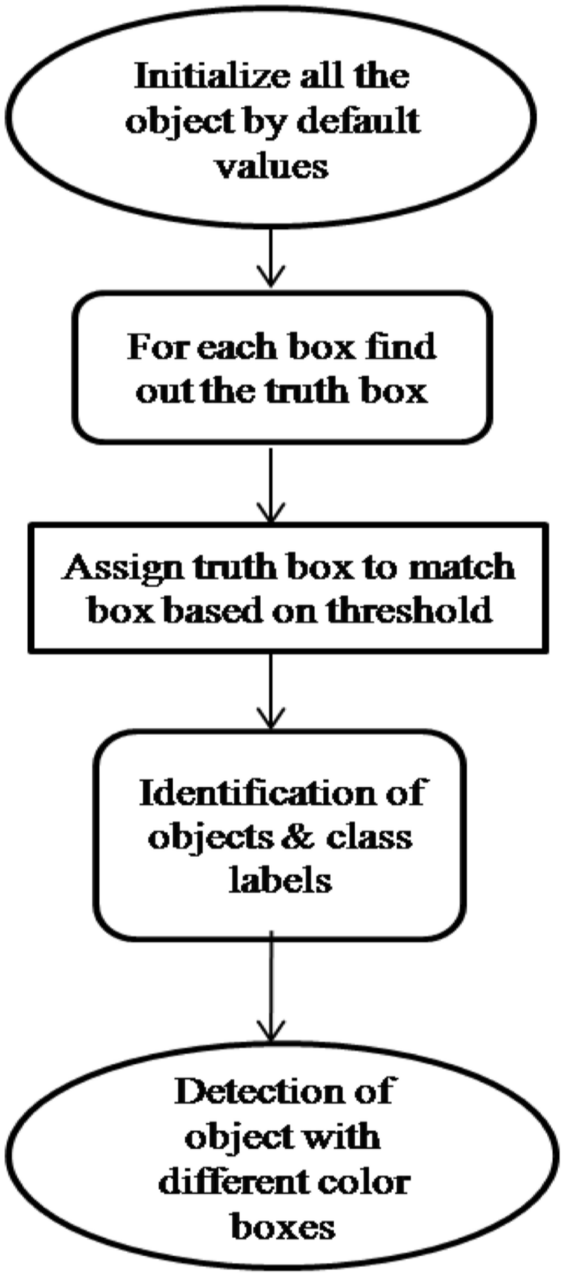
Step 5: Based on IOU, the truth boxes are matched with the predicted boxes.

Step6: Instead of exploiting all the negative examples, the result exploits the best-assured loss for every default box.

Flowchart : For Identifying Total Number of Default Boxes.


Input color Image
Select the size of default box
Identification of feature map
Resize the default box with different feature maps
Calculate the width & height of default box
No
Yes
Total no. of default boxes in image

Flowchart : For Detecting Objects with Different Color Boxes.


Initialize all the object by default values
For each box find out the truth box
Assign truth box to match box based on threshold
Identification of objects & class labels
Detection of object with different color boxes

CHAPTER 6

CONCLUSION :

This study develops an object detector algorithm using deep learning neural networks for detecting the objects from the images. The research uses am improved SSD algorithm along with multilayer convolutional network to achieve high accuracy in real time for the detection of the objects. The performance of our algorithm is good in still images and videos. The accuracy of the proposed model is more than 79.8%. The training time for this model is about 5–6 h. These convolutional neural networks extract feature information from the image and then perform feature mapping to classify the class label. The prime objective of our algorithm is to use the best aspect ratios values for selecting the default boxes so that we can improve SSD algorithm for detecting objects.

REFERENCE :

- My GitHub Account link for Required Files to execute Object Detection Program given below :

https://github.com/hemantcgupta/Object-Detection.git