DEEP LEARNING BASED SCENE IDENTIFICATION FOR HARD-TO-SEE PEOPLE

A mini project report submitted by

L.CHRISTINA SHERIN with Reg.No(s) :URK19CS1063

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

under the supervision of
Dr.P.Getzi Jeba Leelipushpam, Ph.D ,
Associate Professor

COMPUTER SCIENCE AND ENGINEERING

KARUNYA INSTITUTE OF TECHNOLOGY AND SCIENCES

(Declared as Deemed to be University -under Sec-3 of the UGC Act, 1956)
Karunya Nagar, Coimbatore - 641 114. INDIA
March 2022

DEEP LEARNING BASED SCENE IDENTIFICATION FOR HARD-TO-SEE PEOPLE

A mini project report submitted by

L.Christina Sherin (URK19CS063)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

under the supervision of

Dr. P.Getzi Jeba Leelipushpam,Ph.D,

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

KARUNYA INSTITUTE OF TECHNOLOGY AND SCIENCES
(Declared as Deemed-to-be-under Sec-3 of the UGC Act, 1956)
Karunya Nagar, Coimbatore - 641 114. INDIA

March 2022

Karunya INSTITUTE OF TECHNOLOGY AND SCIENCES
(Declared as Deemed to be University under Sec.3 of the UGC Act, 1956)
A CHRISTIAN MINORITY RESIDENTIAL INSTITUTION
AICTE Approved & NAAC Accredited

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the project report entitled, "**Deep Learning based scene identification for hard to see people**" is a bonafide record of Mini Project work done during the even semester of the academic year 2021-2022 by

L.Christina Sherin (Reg. No: URK19CS063)

in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering of Karunya Institute of Technology and Sciences.

Submitted for the Viva Voce held on ________________

Signature of the Guide

ACKNOWLEDGEMENT

First and foremost, I praise and thank ALMIGTHY GOD whose blessings have bestowed in me the will power and confidence to carry out my project.

I am grateful to our beloved founders Late**. Dr. D.G.S. Dhinakaran, C.A.I.I.B, Ph.D** and **Dr. Paul Dhinakaran, M.B.A, Ph.D**, for their love and always remembering us in their prayers.

I extend my thanks to our Vice Chancellor **Dr. P. Mannar Jawahar, Ph.D** and our Registrar **Dr. Elijah Blessing, M.E., Ph.D,** for giving me this opportunity to do the project.

I would like to thank **Dr. Prince Arulraj, M.E., Ph.D.,** Dean, School of Engineering and Technology for his direction and invaluable support to complete the same.

I would like to place my heart-felt thanks and gratitude to **Dr. J. Immanuel John Raja, M.E., Ph.D.,** Head of the Department, Computer Science and Engineering for his encouragement and guidance.

I feel it is a pleasure to be indebted to, **Dr. J. Andrew, M.E, (Ph.D.),** Assistant Professor, Department of Computer Science and Engineering and **Dr.P.Getzi Jeba Leelipushpam(Ph.D.),Associate Professor(guide)** for their invaluable support, advice and encouragement.I also thank **Dr.Kumudha Raimond,(Ph.D.),Professor** for her invaluable support and guidance while working on this project.

I also thank all the staff members of the Department for extending their helping hands to make this project a successful one. I would also like to thank all my friends and my parents who have prayed and helped me during the project work.

ABSTRACT

To assist the blind people in moving around with confidence ,a deep learning based model has been developed in this project. This model accepts input in terms of classified labeled text , which is fed into the openCV's SSD architecture ,a subdomain of CNN. The model is pretrained with the classified images using VGG-16 architecture. It identifies the pretrained objects in a given scene, associates them with the relevant class label, localizes the identified image in terms of bounding boxes and displays their text and Confidence level inside their respective boxes. This labeled text is then converted into speech and deployed in mobile to guide the blind.

CONTENTS

1. Introduction

1.1 Introduction

To assist the blind people in recognizing the objects infront of them,a deep learning based scene identification system is made using python openCv.This system helps in recognizing objects in a captured mp4 video and also the objects that can be captured through webcamera. The object is identified , converted from text to speech and deployed in mobile for easy convenience of hard-to-see people.

1.2 Objectives

• Develop a semantic segmentation and deep learning based model to identify the objects in a given scene
• Recognize the objects and convert it to audio output
• Deploy it in Mobile to assist the visually impaired people

1.3 Motivation

Scene identification is very important for visually impaired people for recognizing the objects that is in front of them, to avoid the risk of accidents and collisions. This model can help the blind to avoid obstacles and move around with confidence.

1.4 Overview of the Project

- A web camera based object recognition system is built using python Open Cv. The system recognizes predefined objects data given by the user as input.
- Image Segmentation plays three important roles in identifying the objects from the given scene

1.Image Classification

Image classification is about assigning labels to the image that the system recognizes.

2.Object Localisation

The deep learning based model creates a boundary around each object it recognizes and labels them with their respective names.

3.Object Detection

It is used to locate the presence of an object along with an axis aligned bounding box indicating the scale and position of every instance of each object.

System Implementation:

This system is implemented using two ways

1.Object Recognition from real time video using webcam

The day to day objects that any person may come across , are given as the input data set. The model recognizes the objects from the input data , localizes them and labels them. The output is detected using web camera.

2.Object Recognition from preloaded video(Mp4 video)

The input dataset is the objects that are found on roadways. A mp4 video of roadside traffic was given to the model to find the objects from the predefined dataset. The recognized objects are localized and labeled.

1.5 Chapter wise Summary

- **MODULE-1**
  - Module 1 is about the introduction , the overview and the motivation behind the project.
- **MODULE-2**
  - Module 2 is about requirement analysis,design and the architecture behind the project.
- **MODULE-3**
  - Module -3 is about the description of the terms used in the project, how the project is   implemented and tools used in the project.
- **MODULE-4**
  - Module-4 is about verification, validation and testing.
- **MODULE-5**
  - Module-5 is about the applications and future scope of the product.

2. Analysis and Design

2.1 Functional Requirements

User Requirements:

Objective	To assist the hard-to-see people in recognizing the obstacles they come across
Focus	Image segmentation and Object detection using python open Cv
End Result	The Objects in a given scene are detected .
Essentiality	The accuracy of recognizing the object is more as the model is trained with deep learning algorithm.

The user has given a training dataset list. The training dataset are everyday objects and roadside obstacles defined in the form of classes and each detected object from the output are segregated into class variables.

System Requirements:

- The system should have a good quality USB based web camera to capture the scene around them.
- The system should have a preinstalled python Compiler (Latest Version)
- The python version must be able to allow the user to download OpenCV module.

Input: Predefined training datasets in the form of images which are already labeled, predefined class labels.
Output: The images that are recognized from the input are localized with a boundary box around them and the name of the class of the object is displayed around it.

2.2 Non-Functional Requirements

- **Product Requirements:**
  The product must be accessible in any computer.
- **Efficiency Requirements:**
  The efficiency of the model is 80 percent because the model is labeling a series of objects within matter of seconds.
- **Dependency Requirements:**
  Installing Opencv and Tensorflow is mandatory.
- **Security Requirements:**
  There exists some predefined vulnerabilities in open-cv like division by zero error,Out of bound errors,heap buffer overflow and Null pointer dereference which can be resolved by taking proper action.

- **Denial Of Service Attacks**
  Denial of service attack happens in the form of out of bounds error in read function,The validateInputImageSize function allows remote attackers to cause Denial Of Service attack.Segfault error may occur due to vectors involving incorrect chunks leading to DOS error.

- **Environmental Requirements:**
  The system can run on various OS like windows and Mac.
  The system can run only on Computer but progress are undergoing to make it run on mobile.

- **Development Requirements:**
  The programming language used here is python.
  Space Requirements: The dataset is compressed to save disk space. The frozen_inference graph is compressed and ssd_object weights are also compressed and given as input to the program.

2.3 Architecture

MobileNetV3 is used so that the model can be implemented in mobile phone CPUs.

Object Detection

- Object detection is a computer vision and image processing technique that deals with detecting semantic objects of a certain class in video.

- There are many ways to detect Object detection. This model uses **Single Shot MultiBox Detector.**

Single Shot MultiBox Detector

- Single Shot MultiBox detector is a deep learning based model used to detect objects from a video source.

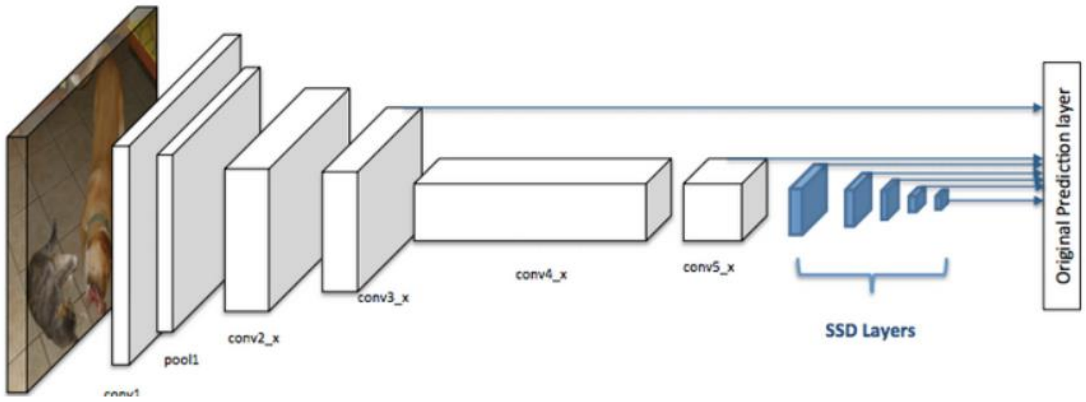
- Several nodes were introduced as a part of Convolution layer. Attribute for each image segment was also declared. The attributes are defined as key and value pair.

- The depth_wiseconv2d applies a different filter to each input, then concatenates the results together.
  - 17 conv_2d layers(hidden layers) are added so that the model can make predictions accurately.

SSD (Single Shot Detector)

SSD architecture is used in this model. SSD has two components.

- Backbone model
- SSD head



SSD Head

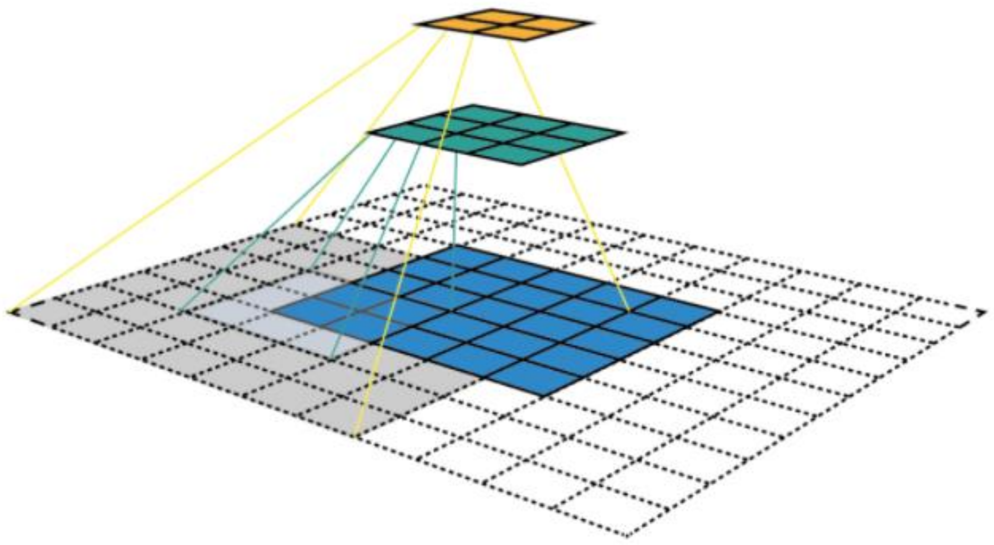
- The SSD head is one or more convolution layers added to the backbone and the outputs are interpreted as the bounding boxes and classes of objects in the spatial location of final layers activation.
- SSD divides the images into a grid and have each grid responsible for detecting objects in the region of the image. Detection is all about predicting the class and location of an object within that region.

- **Anchor Box**
  Each grid cell in SSD can be designed with multiple anchor boxes. The anchor boxes are predefined and each one is responsible for size and shape within a grid cell. The anchor box with the highest degree of overlap with an object is responsible for predicting the object's class and its location.

- **Aspect Ratio**
  The ratio aspect can be used to specify the different aspect ratio of anchor box to account for this.

- **Feature Maps**
  Feature corresponds to 7X7 region on the input image. The kind of green and orange 2d arrays are called feature maps. Features in the same feature map has the same receptive field and look for the same pattern but at different locations.

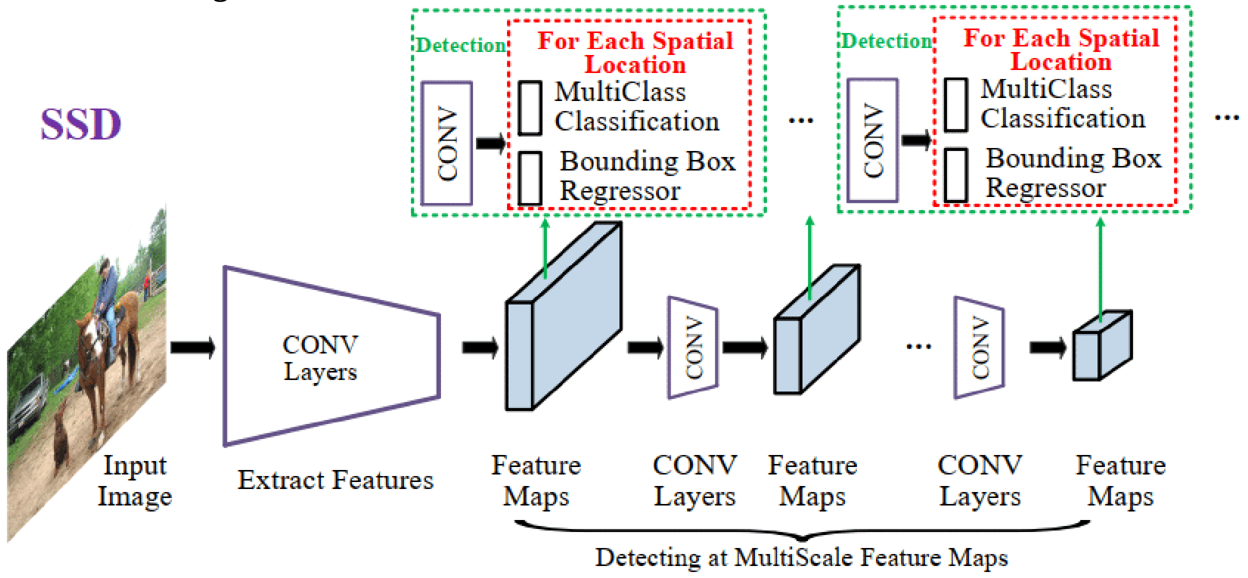
- **Backbone Model**
  Backbone model is a pretrained image classification network which acts as a feature extractor. Usually the fully connected classification layer is removed from the model.



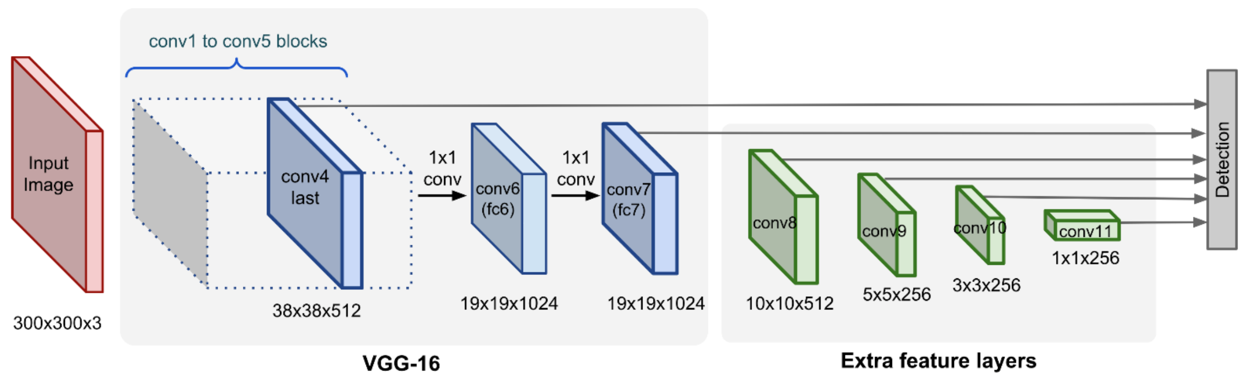
DNN(Deep Neural Network)

- ➢ SSD can be implemented using OpenCV's DNN.
- ➢ DNN can be used to train and test deep learning models. DNN can be trained using CPUs or GPUs.

2.4 Use case diagram



2.5. Sequence Diagram



3. Implementation .

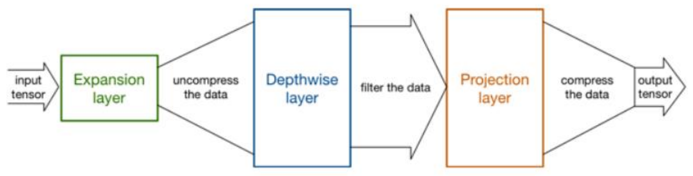
3.1. Modules Description

The model involves getting input from training dataset , process it and identify it in the video given as test output.

A convolution neural network called **MobileNetV3** is used.

Mobile Net V3

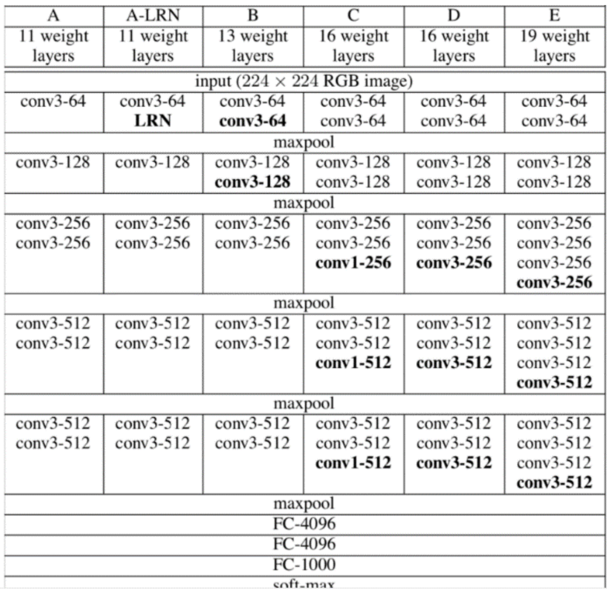
- MobileNetv3 is a convolution neural network that is turned to mobile phone CPUs through a combination of Hardware network Search (NAS) complemented by the **Netadapt algorithm.**It is a depth wis convolution to reduce the number of parameters.The latest version of this model adds squeeze and excitation layers in the initial building block.It is a pretrained convolution neural network.



- The input tensor is given to the model .The data is uncompressed in the Expansion layer. The data is filtered in the Depthwise layer. The data is compressed again in the projection layer. The output tensor is obtained.

VGG-16

VGG is an innovative object recognition model.It supports upto 19 weight layers.



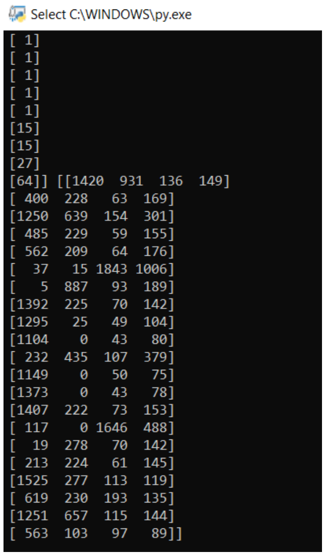
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Fonts in OpenCV:

OpenCV has built-in function to add text on images .Open CV also gives a handful of "HERSHEY-FONTS". This model uses FONT_HERSHEY_COMPLEX which is equivalent to normal size serif font.

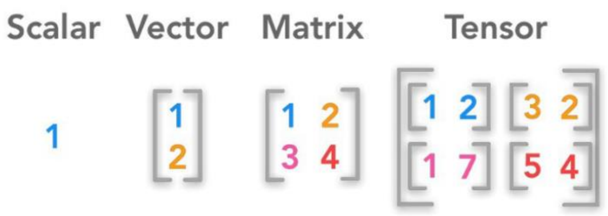
Tensor:

- A tensor is a container which can house data in N dimensions. Often used interchangeably with the matrix, tensors are generalizations of matrices to N-dimensional space .
- The model converts n-Dimensional tensor to 1D tensor.It represents a multilinear relationship and can be denoted as a potentially multidimensional array.



Select C:\WINDOWS\py.exe

```
[ 1]
[ 1]
[ 1]
[ 1]
[ 1]
[15]
[15]
[27]
[64]] [[1420  931  136  149]
[ 400  228   63  169]
[1250  639  154  301]
[ 485  229   59  155]
[ 562  209   64  176]
[  37   15 1843 1006]
[   5  887   93  189]
[1392  225   70  142]
[1295   25   49  104]
[1104    0   43   80]
[ 232  435  107  379]
[1149    0   50   75]
[1373    0   43   78]
[1407  222   73  153]
[ 117    0 1646  488]
[  19  278   70  142]
[ 213  224   61  145]
[1525  277  113  119]
[ 619  230  193  135]
[1251  657  115  144]
[ 563  103   97   89]]
```



- The dimension of a tensor is called its rank. A tensor has shape which is a container that fits our data perfectly and defines the maximum size of our tensor.

Tensor Flow

- Tensor flow first works by building a graph of tf. Tensor objects , detailing how each tensor is computed based on other available tensors and then running the parts of the graph to achieve desired results.

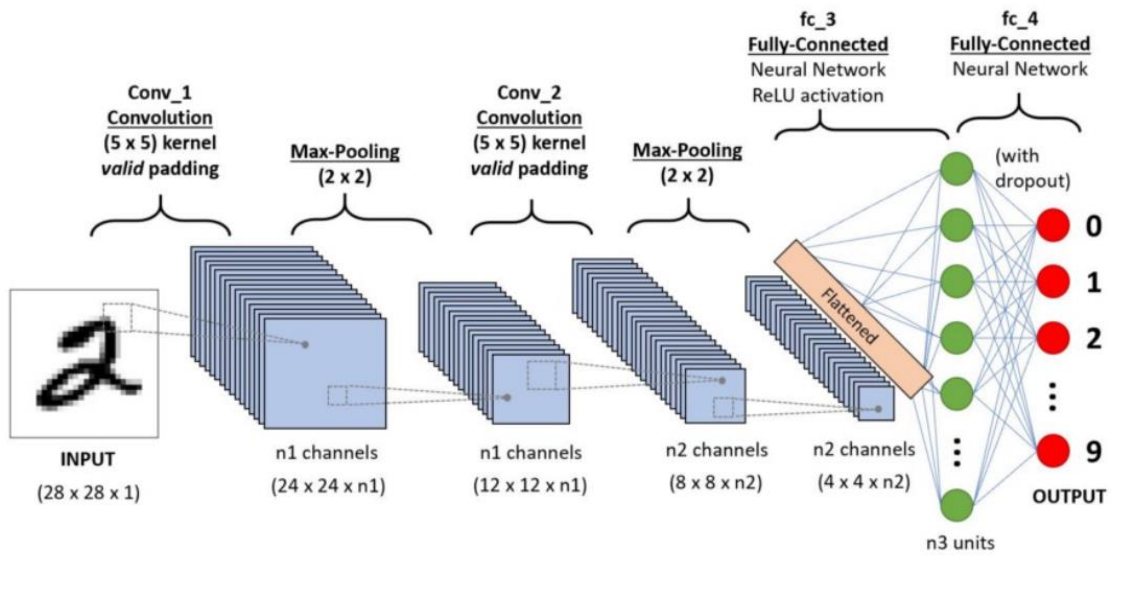
- A layer is a callable object that takes as input one or more tensors and outputs one or more tensors.

Open-CV process

- Images are represented in openCV as numpy arrays.
- Split the images using cv2.split and merge the image back using cv2.merge.

CONVOLUTION LAYER:

Convolution Neural Network consist of multiple layers of Artificial Neurons. Artificial Neurons calculate the weighted sum of multiple inputs and outputs as an activation value. When an image is put in a ConvNet , each layer generates several activation functions that are passed on to the next layer.
The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to next layer which detects more complex features such as corners or combinational edges .As we move deeper it can detect more complex features such as objects and faces. They are regularized versions of multilayer perceptron. CNN extracts the feature from an image



- **2D Convolution Layer(**Spatial convolution over images) creates a convolution kernel that is convolved with layer input to produce a tensor of outputs.
- If activation is none ,it is applied to the output.

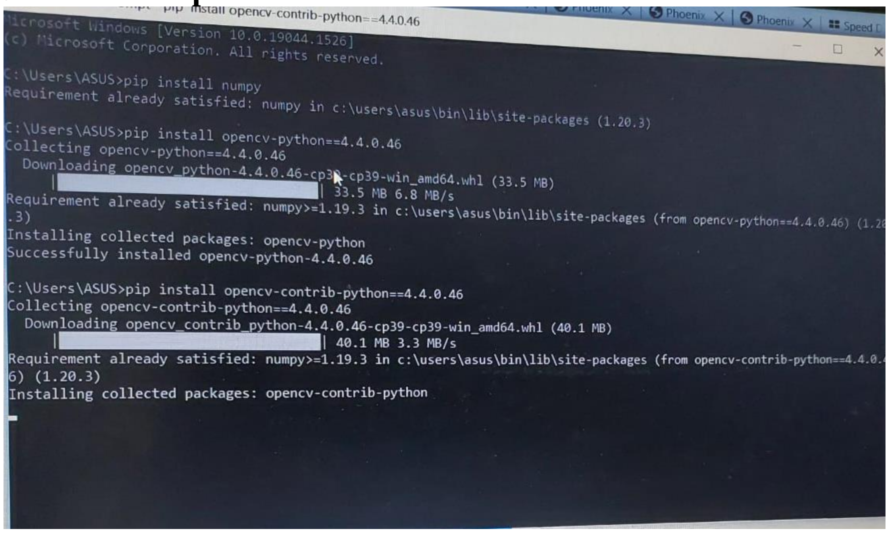
Arguments in 2D Convolution Layer

- **Filters:** Integer,the dimensionality of output space
- **Kernel_size:** A list of 2 integers,specifying the height and width of 2D convolution window.
- **Strides:** A tuple list of 2 integers to specify the value for all spatial dimensions.
- **Padding:** One of “valid” or “same”.”valid” means no padding. ”same” results in padding with zeros evenly to left/right or up/down of input.

- **Kernel_initializer:** Initializer for the kernel wights matrix.Kernel_regulizer:Regularizer function applied to the kernel weight matrix.
- **Kernel_constraint:** Constraint function applied to the kernel matrix.
- **Feature Maps**
  Feature corresponds to 7X7 region on the input image. The kind of green and orange 2d arrays are called feature maps. Features in the same feature map has the same receptive field and look for the same pattern but at different locations.

Implementation Details

- **Configure the developer environment**

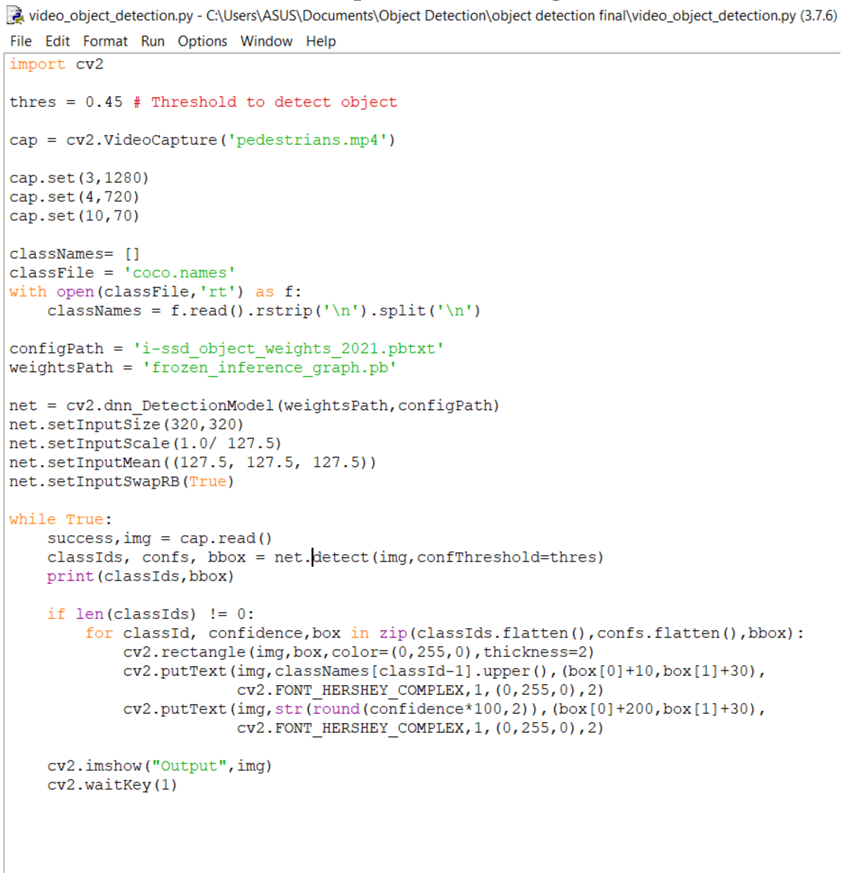


```
pip install opencv-contrib-python==4.4.0.46
icrosoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

:\Users\ASUS>pip install numpy
Requirement already satisfied: numpy in c:\users\asus\bin\lib\site-packages (1.20.3)

:\Users\ASUS>pip install opencv-python==4.4.0.46
Collecting opencv-python==4.4.0.46
  Downloading opencv_python-4.4.0.46-cp39-cp39-win_amd64.whl (33.5 MB)
     |████████████████████████████████| 33.5 MB 6.8 MB/s
Requirement already satisfied: numpy>=1.19.3 in c:\users\asus\bin\lib\site-packages (from opencv-python==4.4.0.46) (1.2
.3)
Installing collected packages: opencv-python
Successfully installed opencv-python-4.4.0.46

C:\Users\ASUS>pip install opencv-contrib-python==4.4.0.46
Collecting opencv-contrib-python==4.4.0.46
  Downloading opencv_contrib_python-4.4.0.46-cp39-cp39-win_amd64.whl (40.1 MB)
     |████████████████████████████████| 40.1 MB 3.3 MB/s
Requirement already satisfied: numpy>=1.19.3 in c:\users\asus\bin\lib\site-packages (from opencv-contrib-python==4.4.0.
6) (1.20.3)
Installing collected packages: opencv-contrib-python
```

  ○

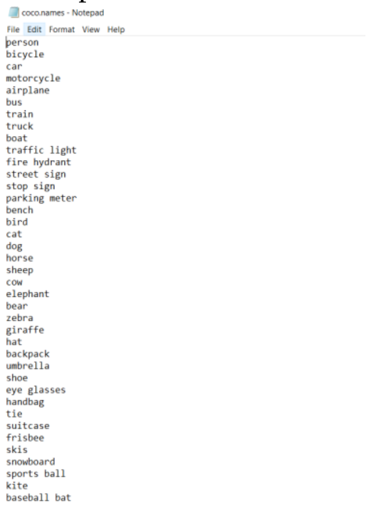
- Pip install opencv
- Pip install numpy
- The model will work only on advanced version of python.The version Python 3.7.6 is used here since this version supports all the libraries from old versions and also from new versions.

- The model uses a predesigned model called **SSD(Single Shot Detector)** architecture. The SSD architecture can be implemented using DNN

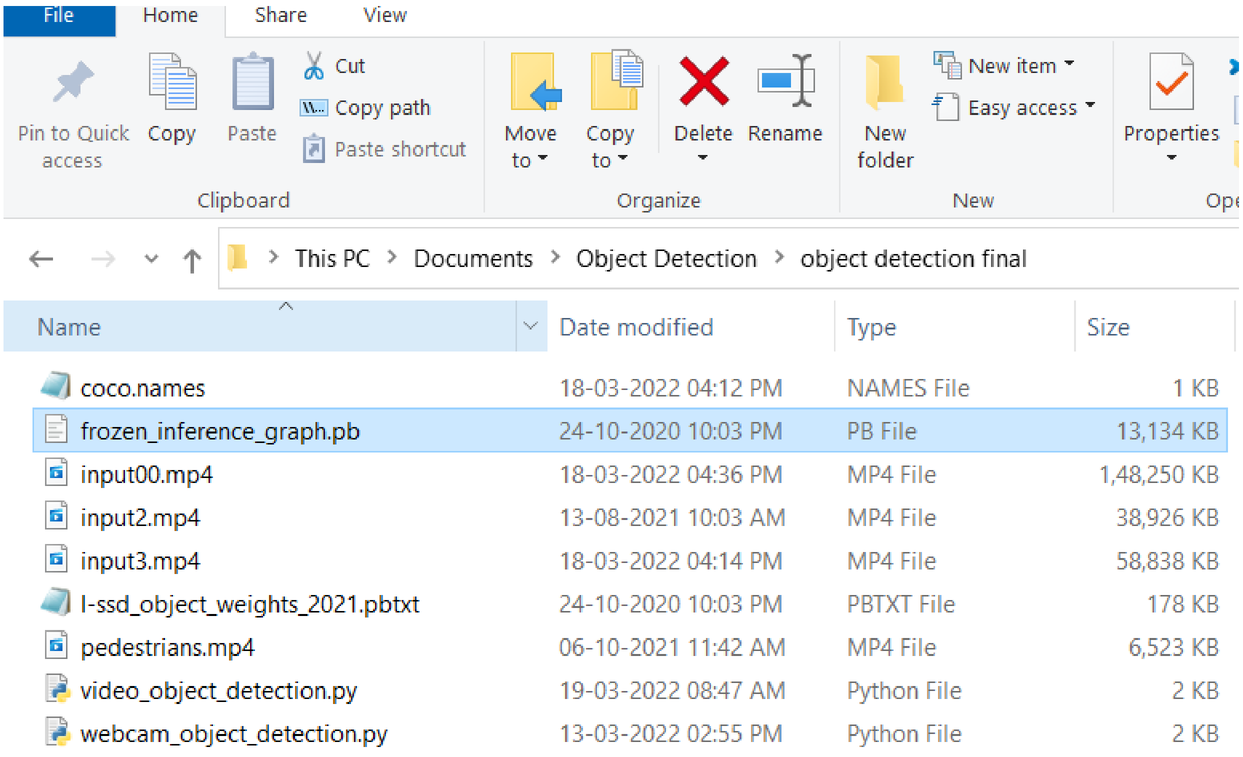

video_object_detection.py - C:\Users\ASUS\Documents\Object Detection\object detection final\video_object_detection.py (3.7.6)
File Edit Format Run Options Window Help

```
import cv2

thres = 0.45 # Threshold to detect object

cap = cv2.VideoCapture('pedestrians.mp4')

cap.set(3,1280)
cap.set(4,720)
cap.set(10,70)

classNames= []
classFile = 'coco.names'
with open(classFile,'rt') as f:
    classNames = f.read().rstrip('\n').split('\n')

configPath = 'i-ssd_object_weights_2021.pbtxt'
weightsPath = 'frozen_inference_graph.pb'

net = cv2.dnn_DetectionModel(weightsPath,configPath)
net.setInputSize(320,320)
net.setInputScale(1.0/ 127.5)
net.setInputMean((127.5, 127.5, 127.5))
net.setInputSwapRB(True)

while True:
    success,img = cap.read()
    classIds, confs, bbox = net.detect(img,confThreshold=thres)
    print(classIds,bbox)

    if len(classIds) != 0:
        for classId, confidence,box in zip(classIds.flatten(),confs.flatten(),bbox):
            cv2.rectangle(img,box,color=(0,255,0),thickness=2)
            cv2.putText(img,classNames[classId-1].upper(),(box[0]+10,box[1]+30),
                        cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
            cv2.putText(img,str(round(confidence*100,2)),(box[0]+200,box[1]+30),
                        cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)

    cv2.imshow("Output",img)
    cv2.waitKey(1)
```

- Using cv2.capture() from CNN , the input video is invoked as argument.The frame size of the boundary boxes are set.The class names are assigned to an empty array and the input data in form of class labels are read from the file coco.names.



```
person
bicycle
car
motorcycle
airplane
bus
train
truck
boat
traffic light
fire hydrant
street sign
stop sign
parking meter
bench
bird
cat
dog
horse
sheep
cow
elephant
bear
zebra
giraffe
hat
backpack
umbrella
shoe
eye glasses
handbag
tie
suitcase
frisbee
skis
snowboard
sports ball
kite
baseball bat
```

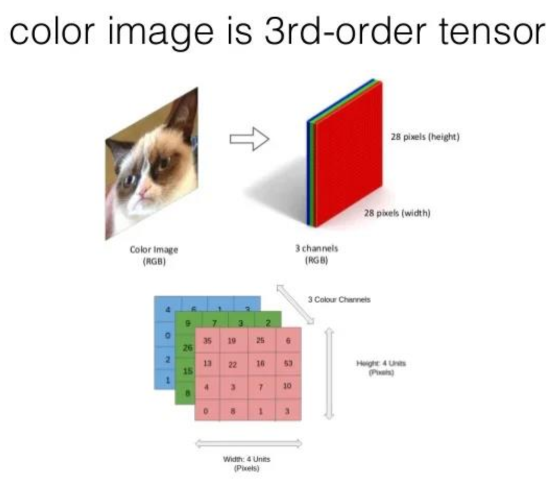


This PC > Documents > Object Detection > object detection final

Name	Date modified	Type	Size
coco.names	18-03-2022 04:12 PM	NAMES File	1 KB
frozen_inference_graph.pb	24-10-2020 10:03 PM	PB File	13,134 KB
input00.mp4	18-03-2022 04:36 PM	MP4 File	1,48,250 KB
input2.mp4	13-08-2021 10:03 AM	MP4 File	38,926 KB
input3.mp4	18-03-2022 04:14 PM	MP4 File	58,838 KB
l-ssd_object_weights_2021.pbtxt	24-10-2020 10:03 PM	PBTXT File	178 KB
pedestrians.mp4	06-10-2021 11:42 AM	MP4 File	6,523 KB
video_object_detection.py	19-03-2022 08:47 AM	Python File	2 KB
webcam_object_detection.py	13-03-2022 02:55 PM	Python File	2 KB

- The frozen inference graph and ssd-object weights are predefined library files that has calculated the weight of each node ,listed all the key ,value pairs along with respective input and output of each node and their function. Both of the files are assigned to a particular variable say configpath, weightpath.

- **Detection_model()** is used from DeepNeuralNetwork architecture wherein weightspath and configpath are invoked. The inputsize,inputscale and Inputmean were set. The InputSwapRB is set to true.**InputswapRB()** is used to split the image in different colours of red,blue and green.



- The image and text is read from the predefined files.detect() is used to detect the image and the confidential level and classId and boxes are printed.

- For the data stored in zip files box and thickness are given. Using putText() function the label and the confidence level of the image is displayed in each boundary boxes. The respective variables are invoked in the putText().The imshow() is used to display the output.

3.3. Tools used

Python, OpenCV, MobileNetSSD prototxt, MobileNetSSD Caffe

4. Test results/experiments/verification

.

4.1. Testing and Verification

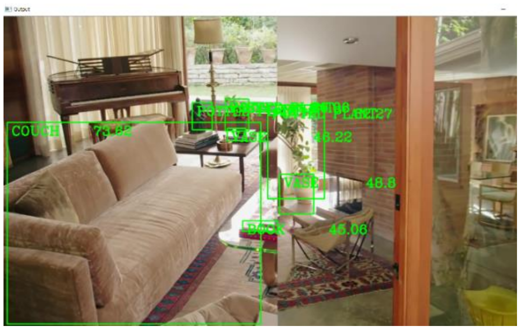
Image classification in Computer Vision takes an image and predicts the object in an image while object detection not only predicts the object but also finds their location in terms of bounding boxes.

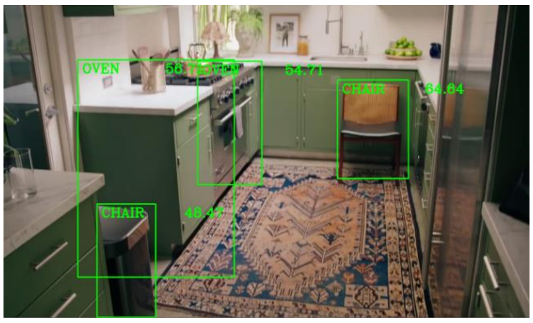
The output of an object detection model include:

- Probablity that there is an object
- Height of the bounding box
- Width of the bounding box
- Horizontal coordinate of the center of the bounding box
- Vertical coordinate of the center point of the bounding point

Phase-1: Testing using first input video

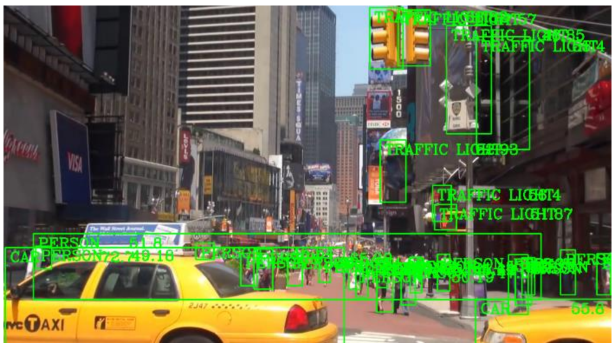
The class datasets are given as input under coco.txt file. The model categorizes the identified items into various classes ,localizes it and displays the class label over it. The model also displays the Confidential level of each item based on their class category.

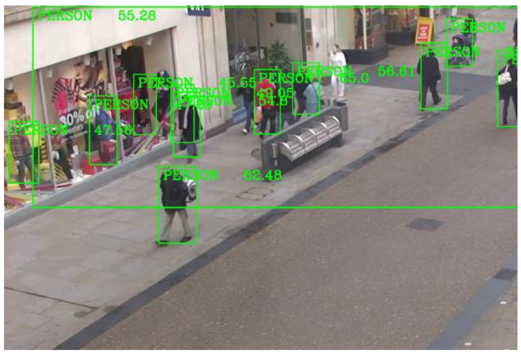




This input video consist of every day objects that any person may come across. The model has successfully identified them based on their classes.

Phase-2: Testing with second input video





The second input video consist of objects that can be identified in roadside area.The model has successfully identified the obstacles any person may come across while walking across the streets.

Phase-3:Testing with webcamera

An USB webcamera was attached with the computer system and the program to detect objects in live video was run.

The objects are detected and the boundary boxes are drawn to differentiate them.

4.2.Result

Thus the model is able to segment images, detect them ,localize them , label them and display their confidence level with good accuracy score.

5.CONCLUSION AND FUTURE SCOPE

Image segmentation is used in variety of applications from healthcare to education industries.

Applications:

- Face Recognition
- Sign Language Detection
- Image Segmentation in Medical Images

Future Scope

The product will help the visually impaired to stand on their own legs without needing any human assistance. The text available in the bounding boxes can be converted to speech using Natural Language Processing in Deep Learning and deployed in mobile to assist the blind. For effective performance, the model can be deployed in a system which has higher Graphical User Interface.

Conclusion

Thus deep learning has found its application in assisting the visually impaired to identify the objects they come across in their day to day life.

REFERENCES

- https://machinelearningmastery.com/object-recognition-with-deep-learning/
- https://mylearningsinaiml.wordpress.com/tensorflow/tensors/
- https://www.tensorflow.org/api_docs/python/tf/keras/layers/Layer
- https://stackoverflow.com/questions/36193553/get-the-value-of-some-weights-in-a-model-trained-by-tensorflow
- https://pyimagesearch.com/2021/01/23/splitting-and-merging-channels-with-opencv/
- https://snyk.io/advisor/python/opencv-python
- https://www.researchgate.net/publication/333248324_Overcoming_Security_issues_using_OpenCV_and_Machine_learning
- https://hackernoon.com/learning-ai-if-you-suck-at-math-p4-tensors-illustrated-with-cats-27f0002c9b32
- https://keras.io/api/layers/convolution_layers/convolution2d/
- https://ieeexplore.ieee.org/document/9522652
- https://iopscience.iop.org/article/10.1088/1742-6596/1883/1/012094/meta
- https://www.alibabacloud.com/blog/part-3-image-classification-using-features-extracted-by-transfer-learning-in-keras_595291
- https://www.codesofinterest.com/2017/07/more-fonts-on-opencv.html
- https://stackoverflow.com/questions/70553077/correct-configuration-path-of-custom-object-detection-model
- https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/
- https://en.wikipedia.org/wiki/Convolutional_neural_network
- https://bdtechtalks.com/2019/12/30/computer-vision-applications-deep-learning/
- https://developers.arcgis.com/python/guide/how-ssd-works/
- https://medium.com/featurepreneur/object-detection-using-single-shot-multibox-detection-ssd-and-opencvs-deep-neural-network-dnn-d983e9d52652