ABSTRACT

We consider the problem of image classification in order to enhance the performance of object detection of images. Typically, Image classifier is used to describe only one object but object detection is used to find location of the multiple objects in the input image and then classified it. In this project, we introducing Region Proposal Network (RPN) that can shares most computations of full-image convolutional features with the object detection network. An RPN is a fully convolutional network that can predicts object bounds and objectness scores at each position. These RPN is trained end-to-end to generate high-quality region proposals, which are used by Fast R-CNN for detection. We used R-CNN algorithm to classify the image by using bounding box for faster object detection.

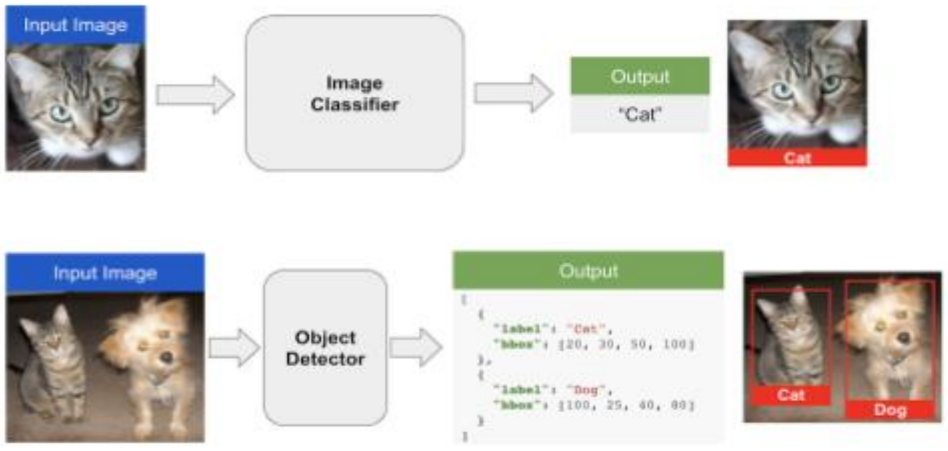
Keywords: Region Proposals, Object Detection, Convolutional Neural Network

CHAPTER 1
INTRODUCTION

The Object detection is a cornerstone of computer vision. It can be connected to both Image Recognition and Image Segmentation. Where the Image Recognition can give outputs as a classification label for an identified object and Image Segmentation can be creates a pixel level understanding of objects in the image, then the object detection are locates the objects within images. The Object detection can find the location of the objects in particular image and classify them. In that sense, the object detection is above and beyond image classification.

Recent advances in object detection, one of the most popular object detection methods is the R-CNN which was developed by Ross Girshick et al in 2014. Then it can improvement depends on with Fast R-CNN and then finally with Faster R-CNN. The differentiating approach that makes Faster R-CNN is better and more faster is the introduction of Region Proposal Network (RPN). These RPN can be used for generating region proposals and a network using these proposals to detect objects. RPN is a more computational fully convolutional of the detection network, it is trained end-to-end, that can be predicts object boundaries and object scores at each detection. With help of RPN being so important to Faster R-CNN, it can be continues which is one of the best object detection frameworks available to researchers, these can be focus on the RPN design and also the concept of anchor boxes and non-maximum suppression.

The Most classical computer vision techniques for object detection like HAAR cascades and HOG + SVM use a sliding window approach for detecting objects. A sliding window is moved over the image. All the pixels inside that sliding window are cropped out and sent to an image classifier. If the image classifier identifies a known object, the bounding box and the class label are stored. Otherwise, the next window is evaluated. The sliding window approach is computationally very expensive. To detect objects in an input image, sliding windows at different scales and aspect ratios need to be evaluated at every pixel in the image. Due to the computational costs, sliding windows are used only when we are detecting a single object class with a fixed aspect ratio. For example, the HOG + SVM or HAAR based face detector in OpenCV uses a sliding window approach. Interesting to note, the famous Viola Jones face detection uses sliding windows. In case of a face detector, the complexity is manageable because only square bounding boxes are evaluated at different scales.



```
[
  {
    "label": "Cat",
    "bbox": [20, 30, 50, 100]
  },
  {
    "label": "Dog",
    "bbox": [100, 25, 40, 80]
  }
]
```

Fig 1 Image classification versus Object detection

Image classification is a good fit in applications where there is only one object in the image. There could be multiple classes (e.g. cats, dogs, etc.) but usually, there is only one instance of that class in the image. In most applications with multiple objects in the input image, we need to find the location of the objects, and then classify them. We use an object detection algorithm in such cases. The Object detection is a two-step process. First step is to find bounding boxes containing objects such that each bounding box has only one object. The second step is to classify the image inside each bounding box and assign it a label.

1.1 APPLICATIONS:

Object detection is breaking into a wide scope of enterprises, with use cases extending from individual security to efficiency in the working environment. Object detection is applied in numerous territories of image processing, including picture retrieval, security, observation, computerized vehicle systems and machine investigation. Critical difficulties remain in the field of object detection. The potential outcomes are inestimable with regards to future use cases for object detection.

1.1.1 Person Detection:

Person detection is necessary and critical work in any intelligent video surveillance framework, as it gives the essential data to semantic comprehension of the video recordings. It has a conspicuous augmentation to automotive applications because of the potential for improving security frameworks. Person detection is undertakings of Computer vision frameworks for finding and following individuals. Person detection is the task of finding all examples of individuals present in a picture, and it has been most broadly achieved via looking through all areas in the picture, at all potential scales, and contrasting a little region at every area with known layouts or examples of individuals. Person detection is commonly viewed as the initial procedure in a video surveillance pipeline and can take care of into more significant level thinking modules, for example, action recognition and dynamic scene analysis.

1.1.2 Automated CCTV Surveillance:

Surveillance is a necessary piece of security and watch. Ongoing advances incomputer vision innovation need to prompt the improvement of different programmed surveillance systems. Be that as it may, their viability is influenced by numerous factors and they are not totally dependable. This examination researched the capability of an automated surveillance system to diminish the CCTV administrator outstanding task at hand in both discovery and following exercises. Typically CCTV is running inevitably, so we need a huge size of the memory framework to store the recorded video. By utilizing an object discovery framework we can mechanize CCTV so that in the event that a few items are detected, at that point the record is going to begin. Utilizing this we can diminish the over and over account a similar picture outlines, which expands memory effectiveness. We can diminish the memory prerequisite by utilizing this object detection system.

Fig 1.1.2 Automated CCTV Surveillance

1.1.3. Vehicle Detection:

Vehicle Detection is one of the most important part in our daily life. As the world is moving faster and the numbers of cars are keep on increasing day by day, Vehicle detection is very important. By using Vehicle Detection technique we can detect the number plate of a speeding car or accident affected car. This also enables for security of the society and decreasing the number of crimes done by car. By using Vehicle Detection Technology Pixel Solutionz have successfully detected the speed of the vehicle and we have also detected the number plate of the car using Optical Character Recognition (OCR). By detecting the Number plate, Pixel Solutionz managed to measure the speed of the vehicle and for and oil company we have successfully developed a Safety Alert System with collision detection warning alert.

1.1.4. People Counting:

Object detection can be additionally utilized for People counting. It is utilized for dissecting store execution or group measurements during festivals. These will, in general, be progressively troublesome as individuals move out of the frame rapidly (likewise in light of the fact that individuals are non-inflexible objects).

1.1 5. Tracking Objects:

An item/object detection framework is additionally utilized in tracking the objects, for instance tracking a ball during a match in the football world cup, tracking the swing of a cricket bat, tracking an individual in a video. Object tracking has an assortment of uses, some of which are surveillance and security, traffic checking, video correspondence, robot vision and activity.

1.2 MOTIVATION:

The main motivation of this project object detection is to recognize and locate (localize) all known objects in a particular image. It will overdo all the physical tasks. Robotics and smart systems are buzzing around the world. Object recognition reduces human efforts and provides efficiency. It is of interest as it may help humans to be aware of minute information about particular objects and reduce human tasks.

1.3 PROBLEM STATEMENT:

The Aim of the project is to detect the objects with faster speed. The disadvantage of previous models are slower to detect the objects. Here, we can perform by finding bounding boxes and classify the image. We can do multiple objects that are identifiable within a particular image by using Regional Proposal Network (RPN). Convolutional Neural Network(CNN) is used to significant improvement in the speed of object detection.

1.4 OBJECTIVE:

The objective of object detection is to find the location of an object in a particular input image accurately and mark the object with the appropriate category. It can detects very faster speed in multiple objects of image. This project can develop as a performance measures through fast time speed of object detection in particular image.

1.5. ORGANIZATION OF REPORT:

The central idea of this project is to develop a perfect model that could provide a faster speed and to develop a flexible and effective to detect and classify the multiple objects. **Chapter 1** deals with introduction to the project, applications, motivation, problem statement, objective and organization of report. **Chapter 2** gives a detailed literature survey and the information useful for understanding of the thesis. **Chapter 3** gives the methodology containing system architecture and system requirements. **Chapter 4** deals with implementation of proposed system. **Chapter 5** deals with analysis and results. **Chapter 6** gives the conclusion and future scope followed by the **BIBILOGRAPHY.**

CHAPTER 2

LITERATURE SURVEY

This chapter describes a survey of existing system on various mechanisms used by R-CNN to identify object detection. After the survey of existing system, limitations of it is mentioned which the proposed system can overcome and provide required solution. Few of the techniques that are used in the literature for advantage are mentioned below.

2.1 OBJECT DETECTION

This paper aim is to detect all instances of objects from one or several known classes, such as people, cars or faces in an image. Typically only a small number of objects are present in the image, but there is a very large number of possible locations and scales at which they can occur and that need to somehow be explored. Object detection approaches typically fall into one of two major categories, generative methods and discriminative methods. A generative method consists of a probability model for the pose variability of the objects together with an appearance model. The model parameters can be estimated from training data and the decisions are based on ratios of posterior probabilities. A discriminative method typically builds a classifier that can discriminate between images (or sub-images) containing instances of the target object classes and those not containing them. The parameters of the classifier are selected to minimize mistakes on the training data, often with a regularization bias to avoid overfitting. Convolutional Neural Network (CNN) for reducing computation and Dynamic programming methods can be searched over the spaces of arrangements efficiently. The Object detection which performed well on COCO dataset and this performance has been steadily increasing and deployed in various applications as augmented and virtual reality scenarios.

2.2 DEEP NEURAL NETWORK FOR OBJECT DETECTION

Deep Neural Networks (DNNs) have recently shown outstanding performance on image classification tasks. In this paper we go one step further and address the problem of object detection using DNN's that is not only classifying but also precisely localizing objects of various classes. We present a simple and yet powerful formulation of object detection as a regression problem to object bounding box masks. We define a multi-scale inference procedure which is able to produce high-resolution object detections at a low cost by a few network applications. These results come at some computational cost at training time as one needs to train a network per object type and mask type. As a future work we aim at reducing the cost by using a single network to detect objects of different classes and thus expand to a larger number of classes. State-of-the-art performance of the approach is shown on Pascal VOC dataset.

2.3 Faster R-CNN

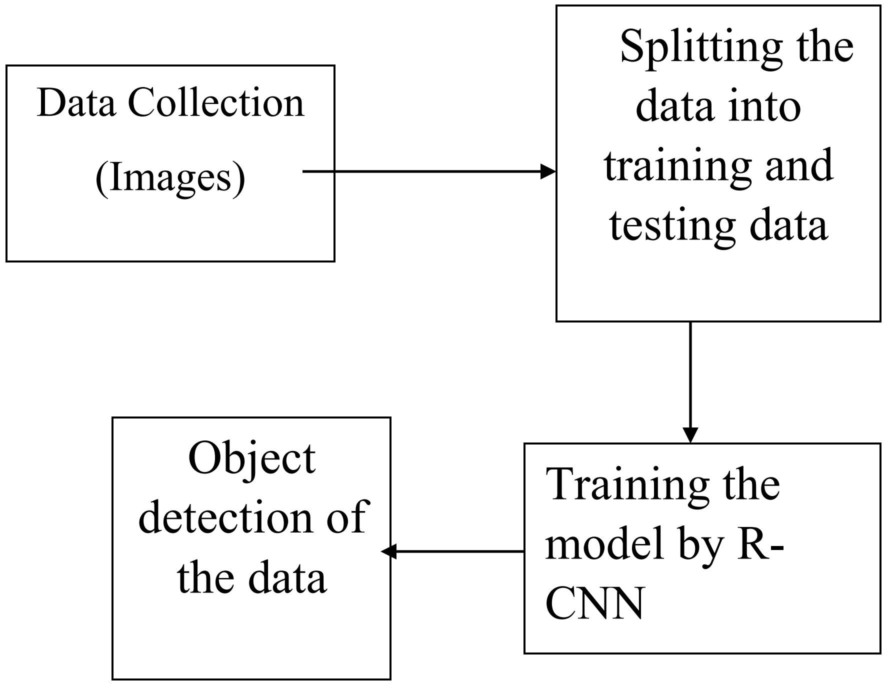
This paper proposes a Fast Region-based Convolutional Network method (Fast R-CNN) for object detection. Fast R-CNN builds on previous work to efficiently classify object proposals using deep convolutional networks. Compared to previous work, Fast R-CNN employs several innovations to improve training and testing speed while also increasing detection accuracy. It can proposed as a Fast R-CNN, a clean and fast update to R-CNN and SPPnet. In addition to reporting state-of-the-art detection results, we present detailed experiments that we hope provide new insights. The Sparse object proposals appear to improve detector quality. It was too costly (in time) to probe in the past, but becomes practical with Fast R-CNN. It can be applied Fast R-CNN (with VGG16) to the MS COCO dataset to establish a preliminary baseline and they are trained on the 80k image training set for 240k iterations and evaluated on the "test-dev" set using the evaluation server. The PASCAL-style MAP is 35.9%; the new COCO-style AP, which also averages over IOU thresholds, is 19.7%. .Faster R-CNN uses the softmax classifier learnt during fine-tuning instead of training one-rest linear SVMs 1446 post-hoc, as was done in R-CNN and SPPnet. To understand the impact of this choice, they are implemented post-hoc SVM training with hard negative mining in Fast R-CNN.

S. NO	**Title of Research Paper**	**Name of Journal & Year of publication & Authors**	**Techniques Or Methodology used**	**Performance Parameters used**	**Advantages**	**Drawbacks**
1	Object detection	Springer, Yali Amit, Pedro Felzenszwalb, Ross Girshick Year: 2020	• Convolutional Neural Network (CNN) for reducing computation • Dynamic programming methods can be searched over the spaces of arranagements efficiently	• Sparse features • Cascade method • Generative and Discriminative models	The performance has been steadily increasing & Deployed in various application as augmented and virtual reality scenarios.	Since these results are less accurate and it is not much faster to detect the multiple objects.

2	Deep Neural Network for Object detection	Paper nips,Christian Szegedy, Alexander Toshev, Dumitru Erhan Year: 2019	• Deep Neural Network (DNN) based detection • DNN Regression • DNN-generatedmasks	• Object Localization • DNN Localizer	It present very simple and yet powerful formulation of object detection	The results can come at some computational cost in training time
3	Fast R-CNN	IEEE international conference ,Ross Girshick, Microsoft Research Year: 2015	• RoI Pooling layer • R-CNN (Region-based Convolutional Neural Network)	• SGD (Stochastic Gradient Descent) hyper-parameters • SVD (Singular Value Decomposition)	Fast R-CNN is clean and fast update to R-CNN and SPPnet	• Object detection is slow • Training is a expensive in space and time

CHAPTER 3
METHODOLOGY

The block diagram of the overall project of the object detection is shown as:



Data Collection (Images)
Splitting the data into training and testing data
Training the model by R-CNN
Object detection of the data

Fig 3 Block Diagram of Object detection

To develop a model for object detection first the images are collected i.e., the first step of the process, Data collection also called Image of the objects from the dataset of MS COCO where we recognize and detect the multiple objects in particular image. Then the collected images can splitting into training data of 60% and testing data of 40%. Then again trained the model by R-CNN and later predict the model and by pipleline for the object detection of pretrained the model, finally the result as the image is displayed.

In this paper, the training the model by R-CNN and RPN and it includes faster R-CNN. It can be predict the model of multiple objects and finally the image is obtained and later it compare inference time of CPU and GPU as it can faster speed to detect the multiple objects in a single input image.

3.1 R-CNN Object Detector:

Every object detector has an image classifier at its heart, the invention of a CNN based object detector became inevitable. Researchers started working on a new idea of training a machine learning model that could propose locations of bounding boxes that contained objects. These bounding boxes were called **Region Proposals** or **Object Proposals**. Region proposals were merely lists of bounding boxes with a small probability of containing an object. It did not know or care which object was in the bounding box.A region proposal algorithm outputs a list of a few hundred bounding boxes at different locations, scales, and aspect ratios. Evaluating the image classifier at a few hundred bounding boxes proposed by the region proposal algorithm is much cheaper than evaluating it at hundreds of thousands or even millions of bounding boxes in case of the sliding window approach. Hence, the region proposal algorithm is still useful and handy at times. One of the first approaches that used region proposals was called **R-CNN** (Regions with CNN features) by Ross Girshick et al. They used an algorithm called Selective Search to detect 2000 region proposals and ran a CNN + SVM based image classifier on these 2000 bounding boxes.The accuracy of R-CNN at that time was the state of the art, but the speed was still very slow (18-20 seconds per image on a GPU).

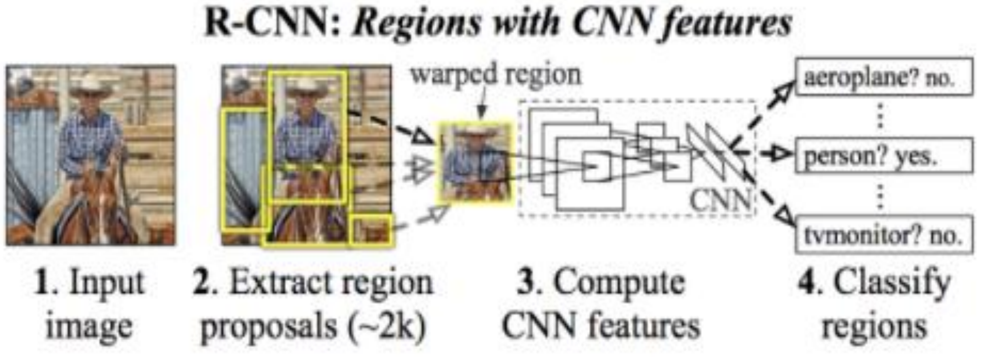
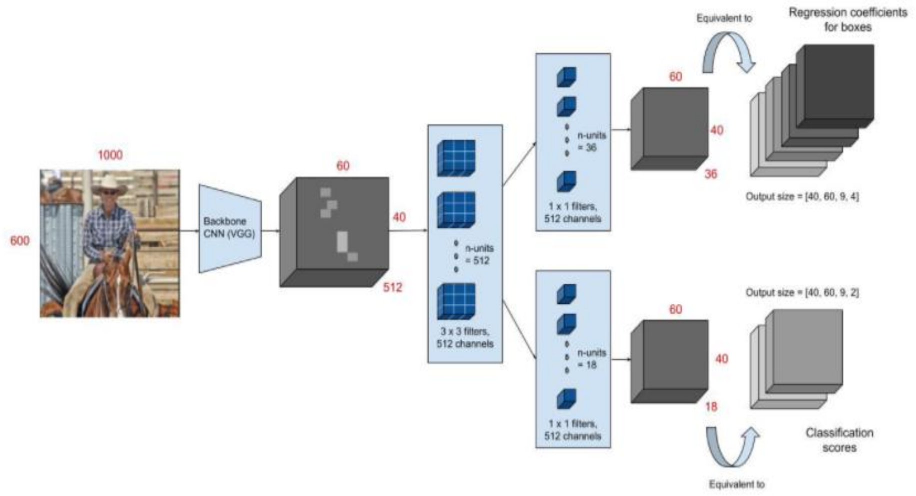


Fig 3.1 R-CNN object detector

3.2 RPN:

RPN is the backbone of faster R-CNN. It has purpose is to propose multiple objects that are identifiable within a particular image. It can be generate the bounding boxes of objects in the image. RPN has a specialized and unique architecture in itself. RPN has a classifier and a regressor. The authors have introduced the concept of anchors. Anchor is the central point of the sliding window. For ZF Model which was an extension of AlexNet, the dimensions are 256-d and for VGG-16, it was 512-d. Classifier determines the probability of a proposal having the target object. Regression regresses the coordinates of the proposals.



1000
600
Backbone CNN (VGG)
60
40
512
n-units = 512
3 x 3 filters, 512 channels
n-units = 36
1 x 1 filters, 512 channels
60
40
36
Equivalent to
Regression coefficients for boxes
Output size = [40, 60, 9, 4]
n-units = 18
1 x 1 filters, 512 channels
60
40
18
Output size = [40, 60, 9, 2]
Classification scores
Equivalent to

Figure 3.2 RPN

3.3 Faster R-CNN:

In Fast R-CNN, even though the computation for classifying 2000 region proposals was shared, the part of the algorithm generating the region proposals did not share any computation with the part that performed image classification. In the follow up work called Faster R-CNN, the main insight was that the two parts as first is calculating region proposals and second is image classification could use the same feature map and therefore share the computational load. A Convolutional Neural Network was used to produce a feature map of the image which was simultaneously used for training a region proposal network and an image classifier. Because of this shared computation, there was a significant improvement in the speed of object detection.

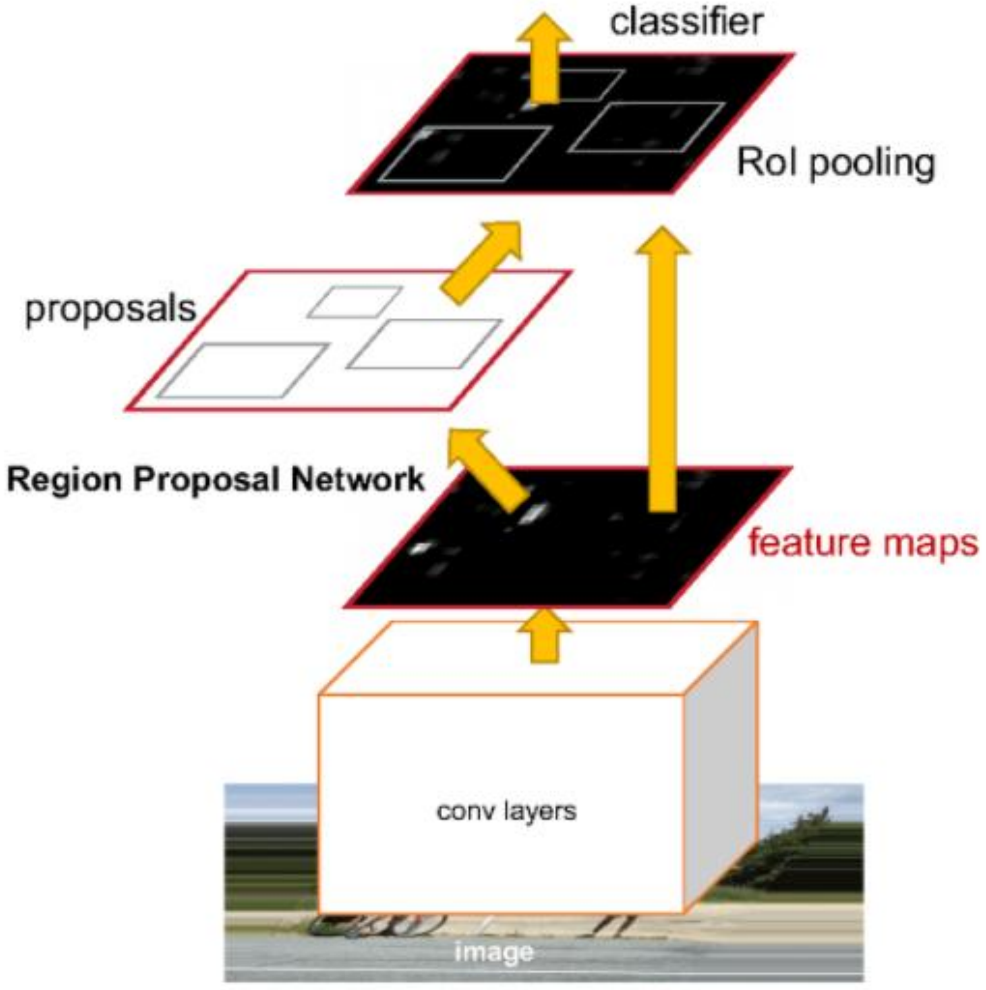


Fig 3.3 Faster R-CNN

The image is provided as an input to a convolutional network which provides a convolutional feature map. Instead of using selective search algorithm on the feature map to identify the region proposals, a separate network is used to predict the region proposals. The predicted region proposals are then reshaped using a RoI pooling layer which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes.

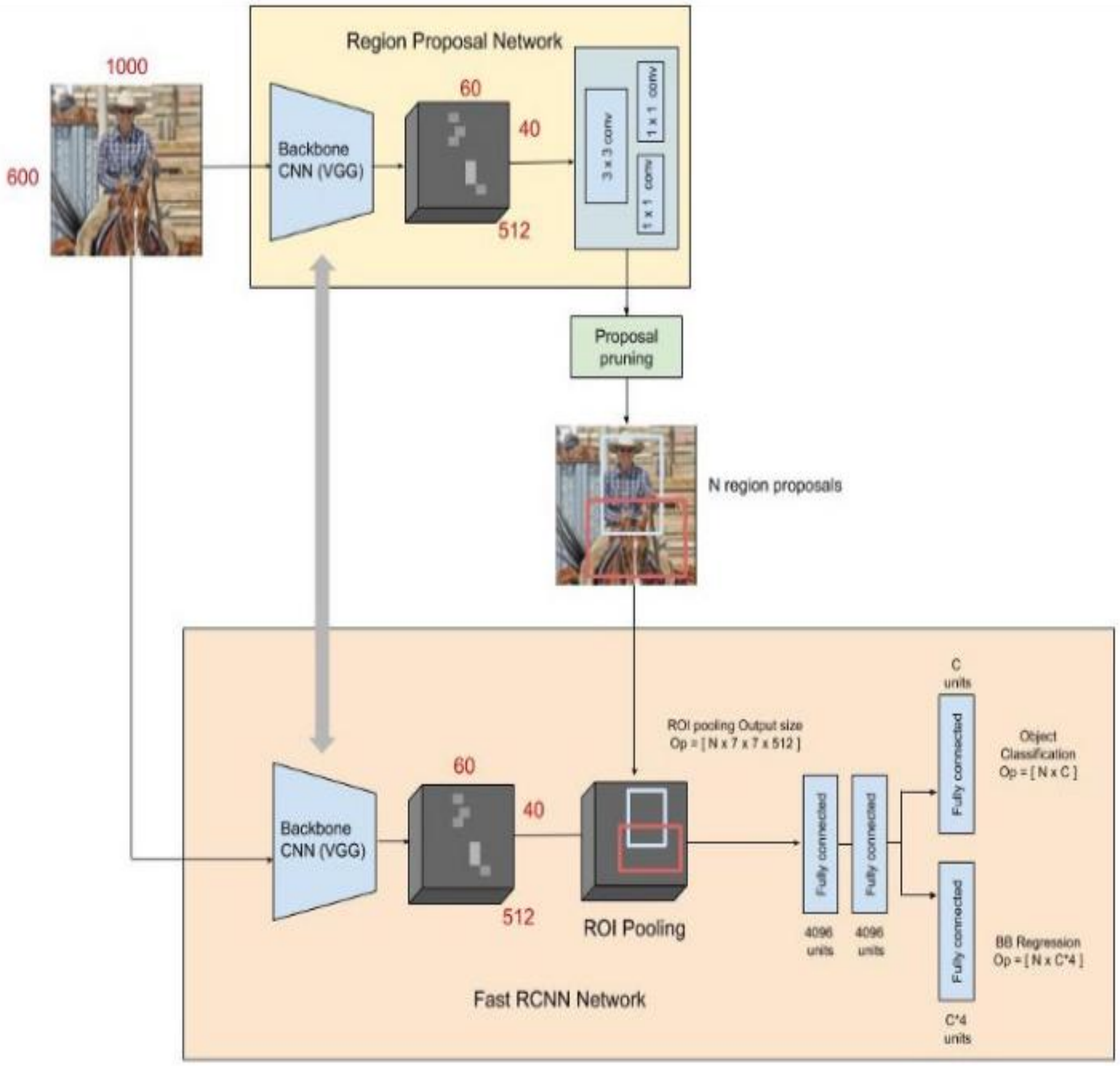


Fig 3.3.1 The RPN for region proposals and fast R-CNN as a detector in the Faster R-CNN detection pipeline

The learned regression output t_i can be applied to its corresponding anchor box (that is predicted positive), and the *x, y, w, h* parameters for the predicted object proposal bounding box can be back-calculated from:

$$t_x^* = (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a, \quad t_w^* = \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a)$$

3.4 SYSTEM REQUIREMENTS:

System Requirements consists of the both hardware and software requirements which are given below.

3.4.1 Software Requirements:

• Operating System	:	Windows 10
• Coding language	:	Python
• Integrated Development Environment	:	Anaconda Jupyter

3.4.2 Hardware Requirements:

• System	:	Intel i3 processor
• Hard Disk	:	1TB
• Monitor	:	15 VGA Color
• RAM	:	8GB

CHAPTER 4

IMPLEMENTATION

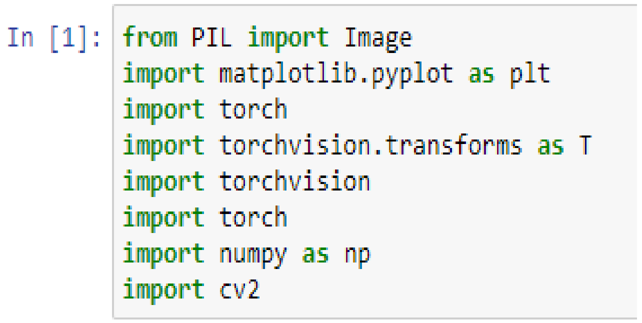
Object detection of faster R-CNN using Region Proposals Network (RPN) can be implemented in following stages . We take the MS COCO dataset as input and perform the pre-trained the model by collecting images. Then it can prediction of the model and later by pipeline of the object detection can display the image and evaluate the performance of the model. All these phases are imported and implemented Faster R-CNN object detection using RPN model.

We will use the pre-trained model included with torchvision and faster R-CNN object detector with PyTorch. Details of all the pre-trained models in PyTorch can be found in torchvision.models.

4.1 Collection of the Image(inputs):

The pretrained Faster R-CNN ResNet-50 model that we are going to use expects the input image tensor to be in the form [n, c, h, w] and have a min size of 800px, where "n" is the number of images, "c" is the number of channels for RGB images its 3, "h" is the height of the image, "w" is the width of the image.

We should import all library functions which can included in this project as shown below:

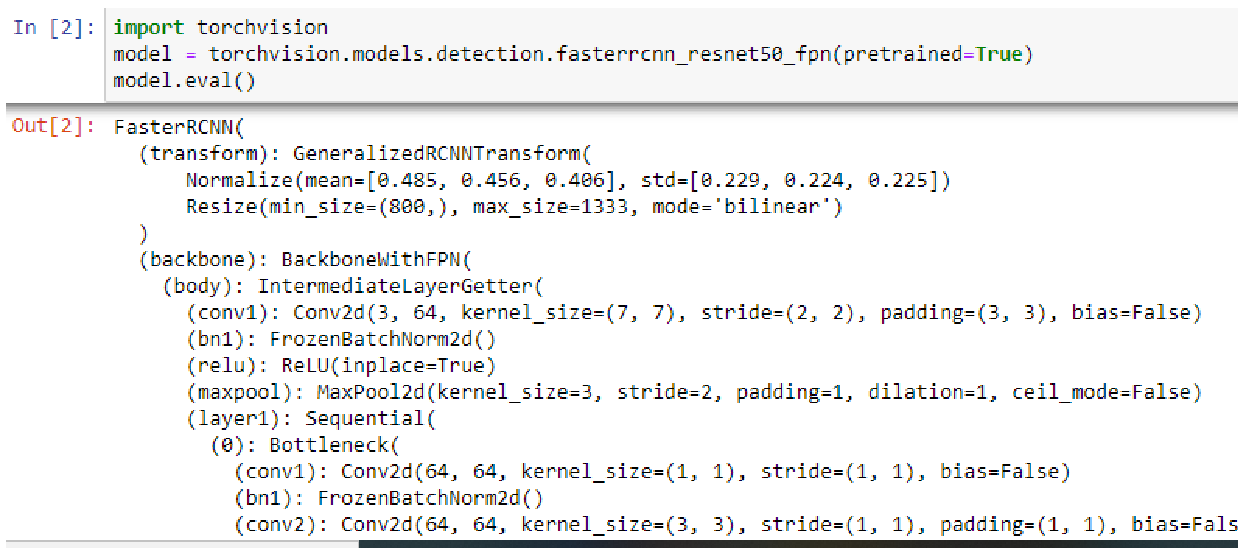


```
In [1]: from PIL import Image
        import matplotlib.pyplot as plt
        import torch
        import torchvision.transforms as T
        import torchvision
        import torch
        import numpy as np
        import cv2
```

where PIL is Python Imaging Library. Matplotlib is a plotting library as a pyplot of a numerical mathematics extension Numpy. Torchvision is a package consists of model architectures and common image transformations for computer vision.

4.2 Pre-trained the Model:

We can download the pre-trained model from torchvision with the following code:

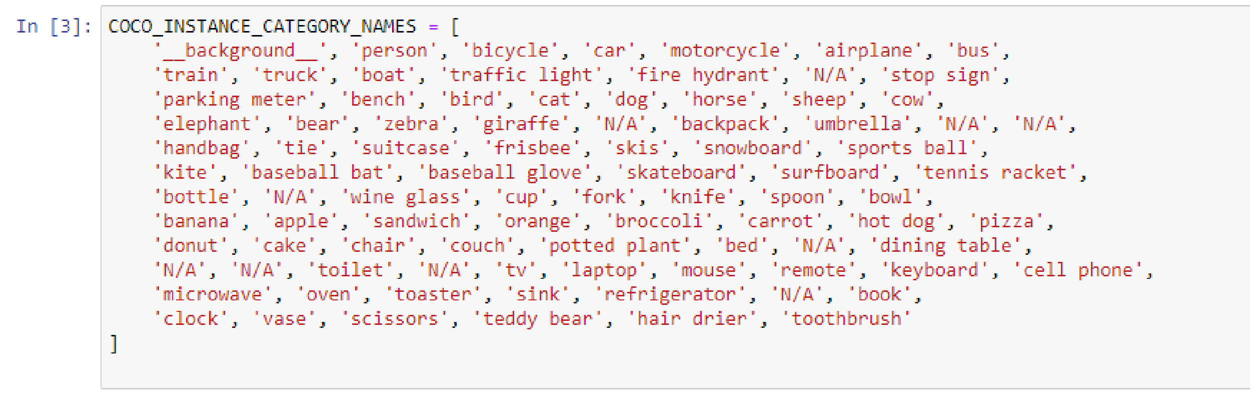


```
In [2]: import torchvision
        model = torchvision.models.detection.fasterrcnn_resnet50_fpn(pretrained=True)
        model.eval()

Out[2]: FasterRCNN(
          (transform): GeneralizedRCNNTransform(
              Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
              Resize(min_size=(800,), max_size=1333, mode='bilinear')
          )
          (backbone): BackboneWithFPN(
            (body): IntermediateLayerGetter(
              (conv1): Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3), bias=False)
              (bn1): FrozenBatchNorm2d()
              (relu): ReLU(inplace=True)
              (maxpool): MaxPool2d(kernel_size=3, stride=2, padding=1, dilation=1, ceil_mode=False)
              (layer1): Sequential(
                (0): Bottleneck(
                  (conv1): Conv2d(64, 64, kernel_size=(1, 1), stride=(1, 1), bias=False)
                  (bn1): FrozenBatchNorm2d()
                  (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=Fals
```

A Line 2 will download a pre-trained Resnet50 Faster R-CNN model with pre-trained weights.

We can define the class names given by PyTorch's official documents.

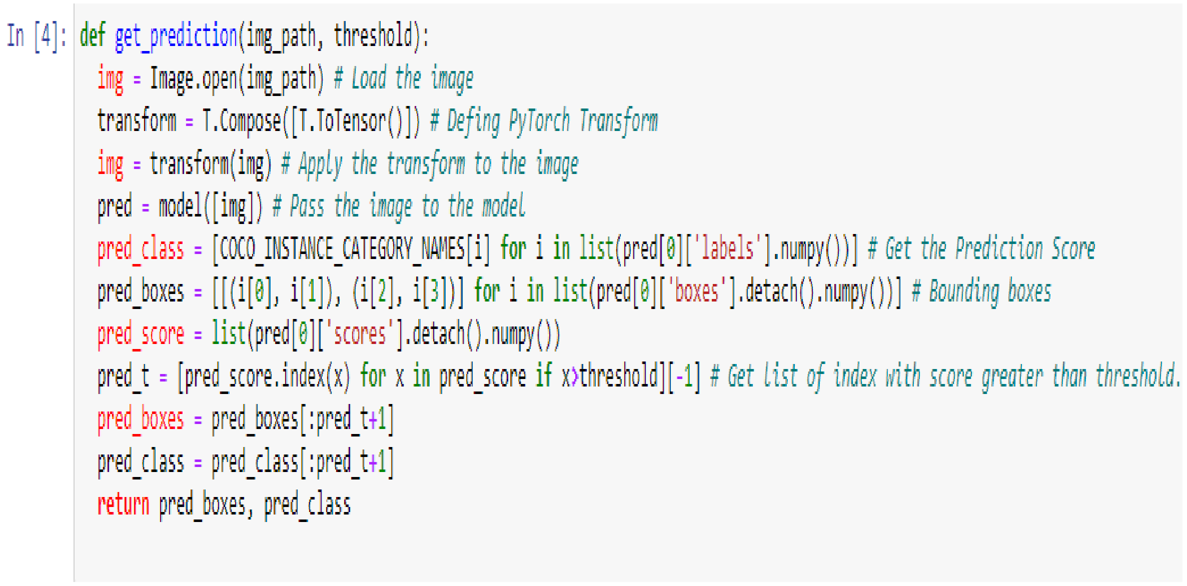


```
In [3]: COCO_INSTANCE_CATEGORY_NAMES = [
            '__background__', 'person', 'bicycle', 'car', 'motorcycle', 'airplane', 'bus',
            'train', 'truck', 'boat', 'traffic light', 'fire hydrant', 'N/A', 'stop sign',
            'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'cow',
            'elephant', 'bear', 'zebra', 'giraffe', 'N/A', 'backpack', 'umbrella', 'N/A', 'N/A',
            'handbag', 'tie', 'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball',
            'kite', 'baseball bat', 'baseball glove', 'skateboard', 'surfboard', 'tennis racket',
            'bottle', 'N/A', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl',
            'banana', 'apple', 'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza',
            'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed', 'N/A', 'dining table',
            'N/A', 'N/A', 'toilet', 'N/A', 'tv', 'laptop', 'mouse', 'remote', 'keyboard', 'cell phone',
            'microwave', 'oven', 'toaster', 'sink', 'refrigerator', 'N/A', 'book',
            'clock', 'vase', 'scissors', 'teddy bear', 'hair drier', 'toothbrush'
        ]
```

We can see some N/A's in the list, as a few classes were removed in the later papers. We will go with the list given by PyTorch.

4.3 Prediction of the Model:

Let's define a function to get the image path and get the prediction of the image by the model.

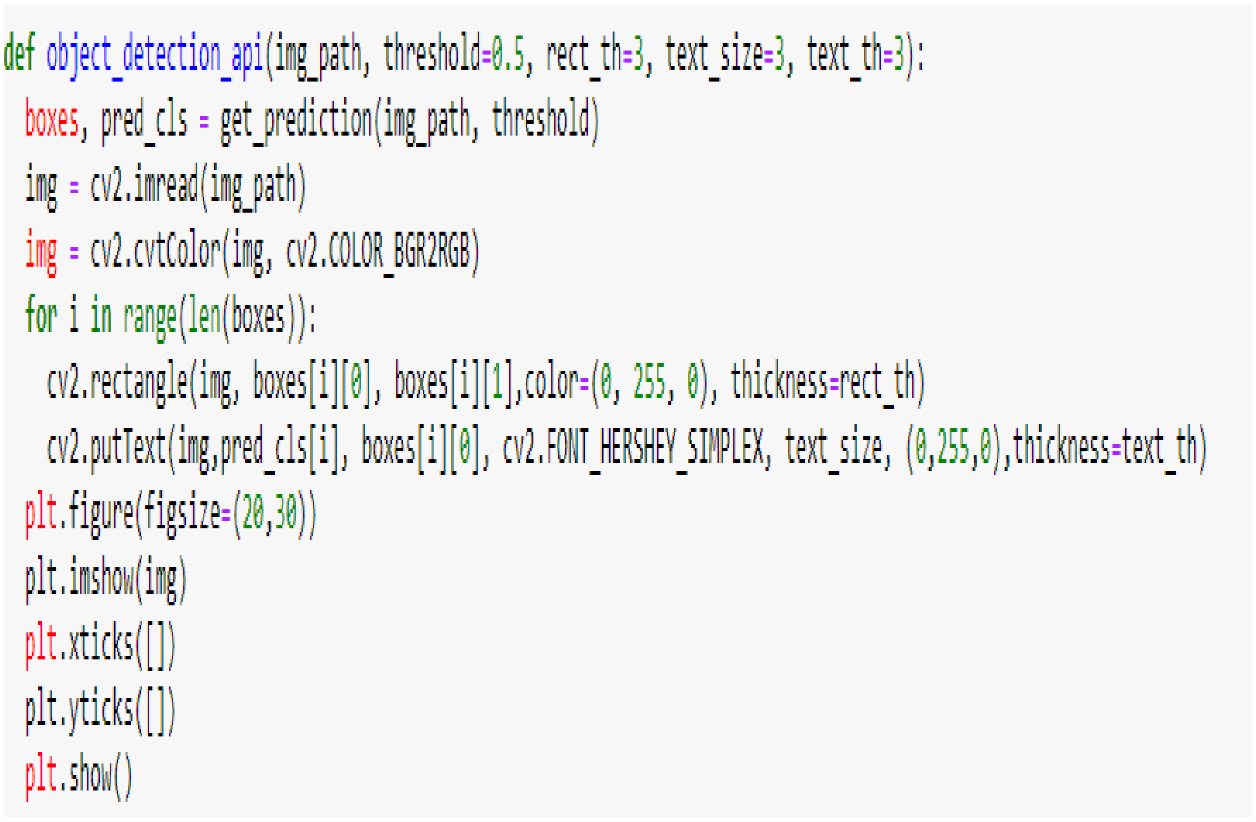


```
In [4]: def get_prediction(img_path, threshold):
          img = Image.open(img_path) # Load the image
          transform = T.Compose([T.ToTensor()]) # Defing PyTorch Transform
          img = transform(img) # Apply the transform to the image
          pred = model([img]) # Pass the image to the model
          pred_class = [COCO_INSTANCE_CATEGORY_NAMES[i] for i in list(pred[0]['labels'].numpy())] # Get the Prediction Score
          pred_boxes = [[(i[0], i[1]), (i[2], i[3])] for i in list(pred[0]['boxes'].detach().numpy())] # Bounding boxes
          pred_score = list(pred[0]['scores'].detach().numpy())
          pred_t = [pred_score.index(x) for x in pred_score if x>threshold][-1] # Get list of index with score greater than threshold.
          pred_boxes = pred_boxes[:pred_t+1]
          pred_class = pred_class[:pred_t+1]
          return pred_boxes, pred_class
```

- Image is obtained from the image path
- The image is converted to image tensor using PyTorch's Transforms
- The image is passed through the model to get the predictions
- Class, box coordinates are obtained, but only prediction score > threshold are chosen.

4.4 Pipleine For the Object detection:

Next we will define a pipeline to get the image path and get the output image.



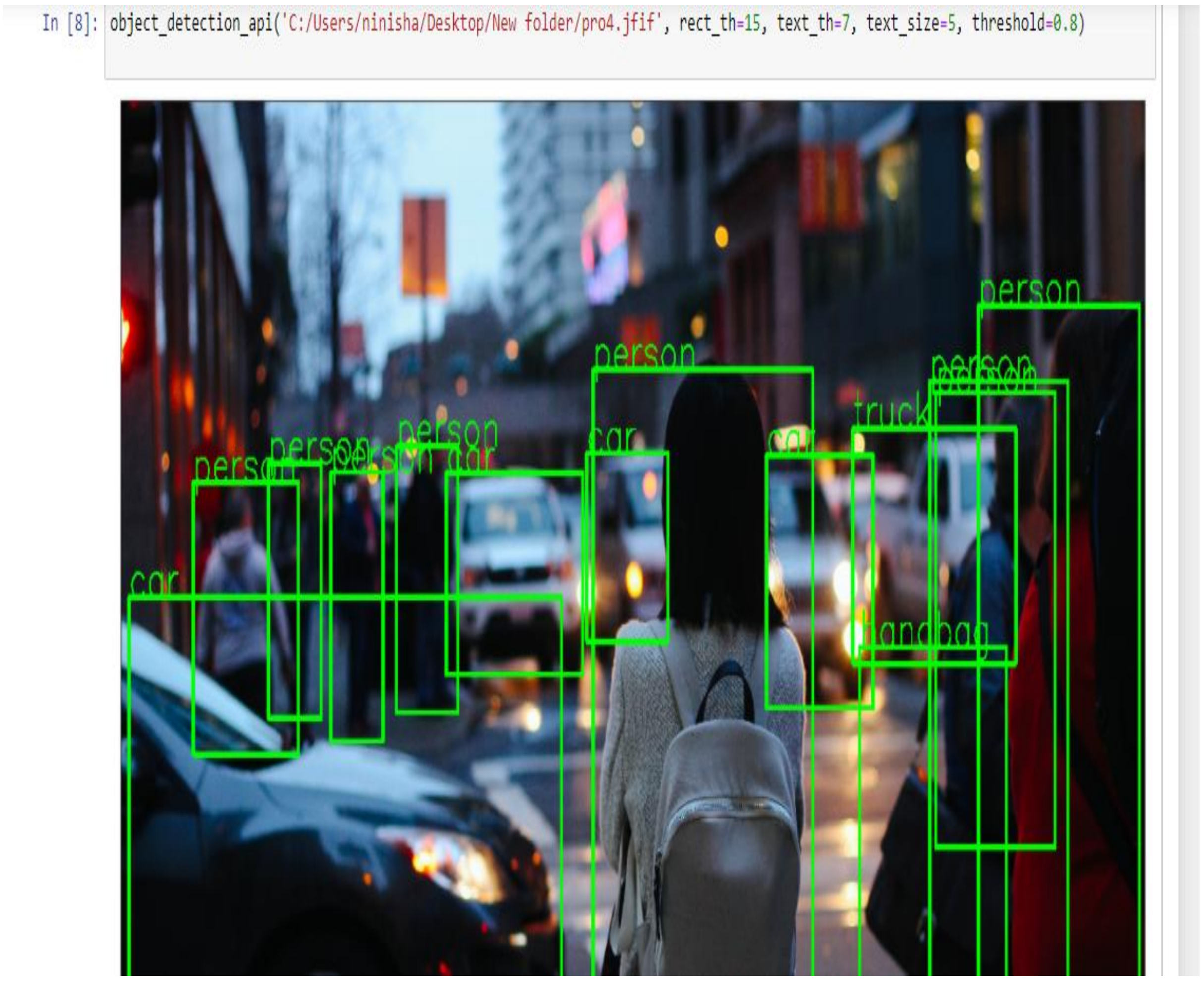
```python
def object_detection_api(img_path, threshold=0.5, rect_th=3, text_size=3, text_th=3):
  boxes, pred_cls = get_prediction(img_path, threshold)
  img = cv2.imread(img_path)
  img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
  for i in range(len(boxes)):
    cv2.rectangle(img, boxes[i][0], boxes[i][1],color=(0, 255, 0), thickness=rect_th)
    cv2.putText(img,pred_cls[i], boxes[i][0], cv2.FONT_HERSHEY_SIMPLEX, text_size, (0,255,0),thickness=text_th)
  plt.figure(figsize=(20,30))
  plt.imshow(img)
  plt.xticks([])
  plt.yticks([])
  plt.show()
```

- prediction is obtained from get_prediction method
- for each prediction, bounding box is drawn and text is written with opencv
- the final image is displayed

CHAPTER 5

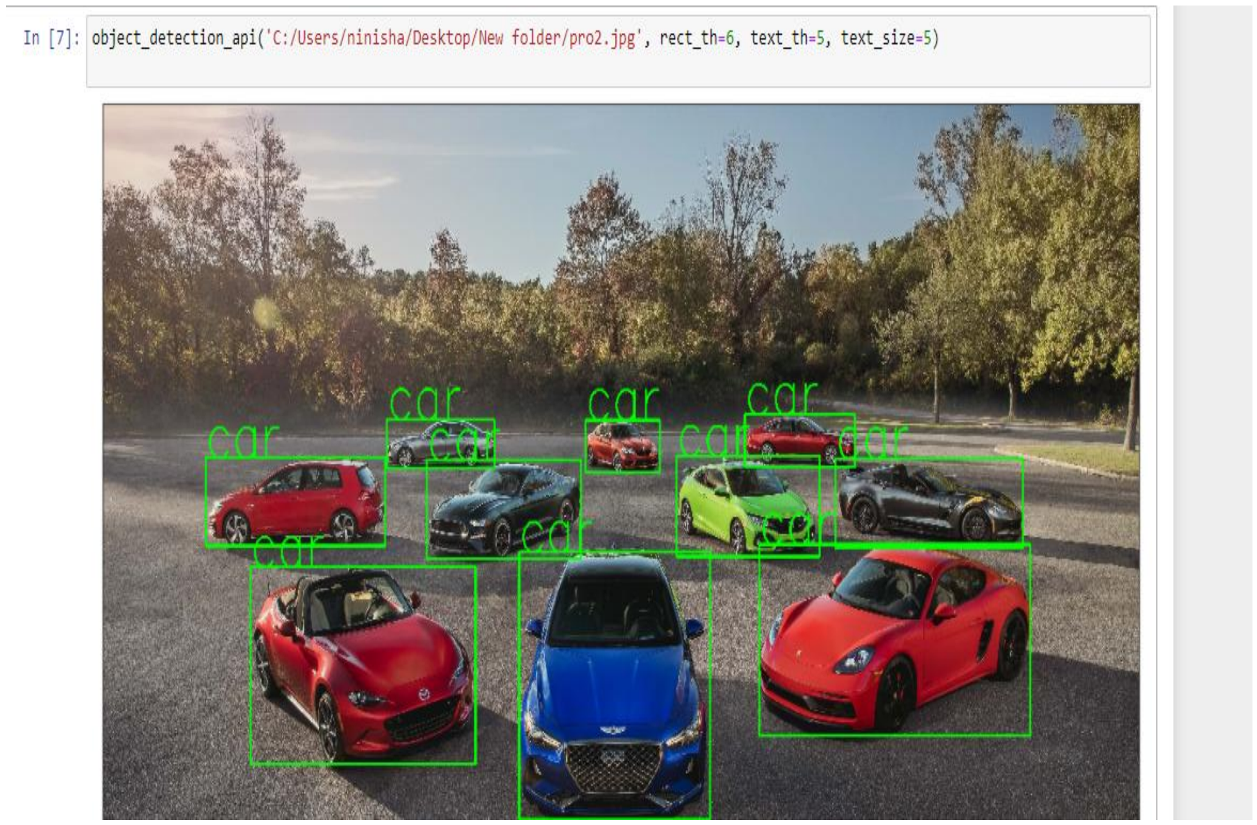
TESTING AND RESULTS

5.1 Output Screens:



```
In [8]: object_detection_api('C:/Users/ninisha/Desktop/New folder/pro4.jfif', rect_th=15, text_th=7, text_size=5, threshold=0.8)
```

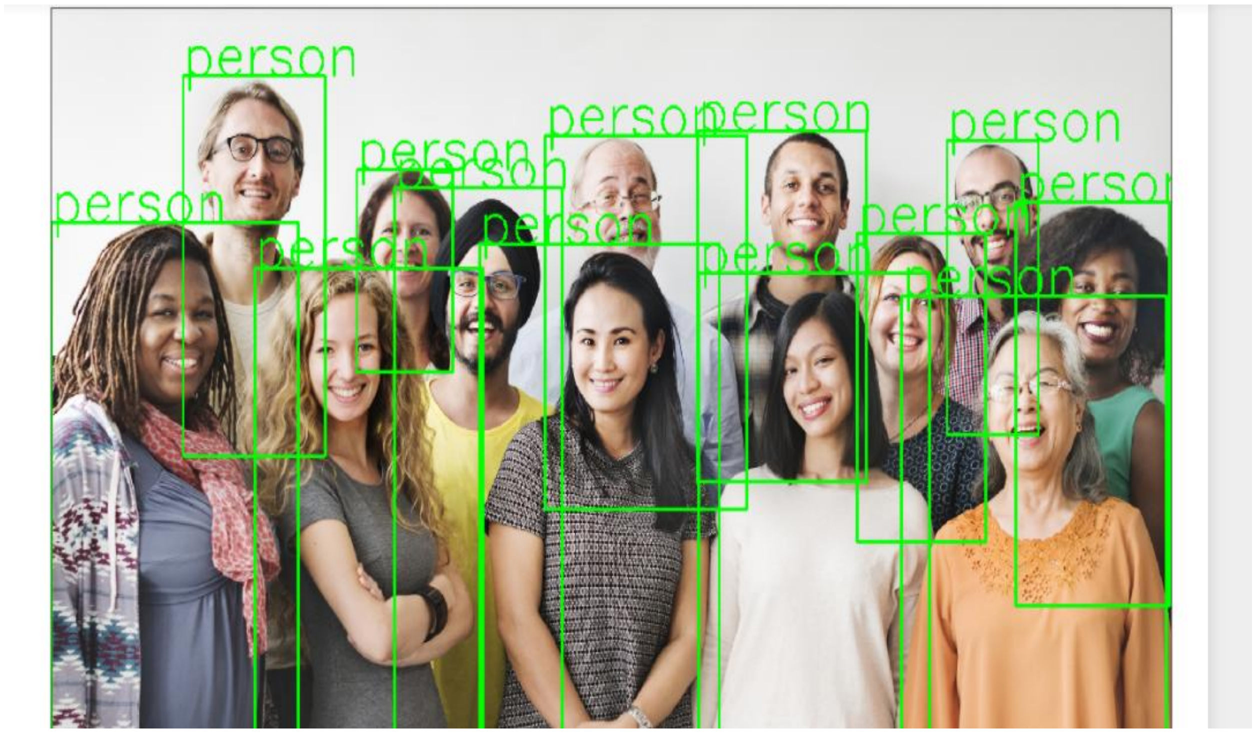
It can detect the multiple objects in single image of JFIF format. It can be fast detecting the different objects like person, car, handbag, truck,..as shown in above output to each category of objects.



```
In [7]: object_detection_api('C:/Users/ninisha/Desktop/New folder/pro2.jpg', rect_th=6, text_th=5, text_size=5)
```

From the above output (figure), it can detect multiple objects as Car of similar objects to each one in single JPG format.

The below output (figure) are detect multiple persons of different gender in single image in JPEG format as same as output 5.2 but different ways of objects.



5.2 Comparing the inference time of model in CPU & GPU:

We would like to know the inference time of each of the models in CPU and GPU. We measure of the time taken by the model to predict the output for an input image. ie: time taken for **prediction = model(image)**

Models	CPU	GPU
Faster R-CNN ResNet-50 FPN	8.45859 s	0.15356 s

The API pipleine which we built to detect object in some images. The pretrained Model takes around 8 seconds for inference in CPU and 0.15 second in NVIDIA GTX 1080 Ti GPU.

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

We have presented RPNs for efficient and accurate as faster speed detection of region proposal generation. By sharing Convolutional features with the down-stream detection network, the region proposal step is nearly cost-free. The result can be displayed multiple objects in single input image effectively. The learned RPN also improves region proposal quality and thus the overall object detection accuracy. We can be improvement the performances as a more faster than R-CNN and SPP of the object detection The RPN can generate high-quality regional proposals and more accurate. We can enables a unified, deep-learning-based object detection system to run at near real-time frame rates. An Efficient less time consuming objects of detection methods is projected which performed on single input image.

The Faster R-CNN object detection is a one of the most widely used concept in the field of Artificial Intelligence. It has a great scope in future for the development. There are comes as possibilities to future use cases for object detection as digital watermark, robotics, online images, activity recognition, so on. The performance requirements of object detection applications have continuously increased the computing power of implementation platforms, especially when they are executed under real time constraints.

BIBILOGRAPHY

[1] Zhaowei Cai and Nuno Vasconcelos ,“Cascade R-CNN: Delving into High Quality Object Detection”,2018.

[2] Ross Girshick, “Faster R-CNN”,Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015.

[3] Yali Amit, Pedro Felzenszwalb and Ross Girshick ,”Object detection” , IEEE journal published , 2020.

[4] Paper nips,Christian Szegedy, Alexander Tosheva and Dumitru Erhan, ”Deep Neural Network”, IEEE journal published,2019.

[5]Zhong-Qiu Zhao and Shou-Tao Xu,“Object Detection with Deep Learning”, IEEE Journal published,2019.

[6]Stephan Brehm,Anton winschel and Dan Zrcha,” Small object detection in faster R-CNN”,IEEE journal published,2018.