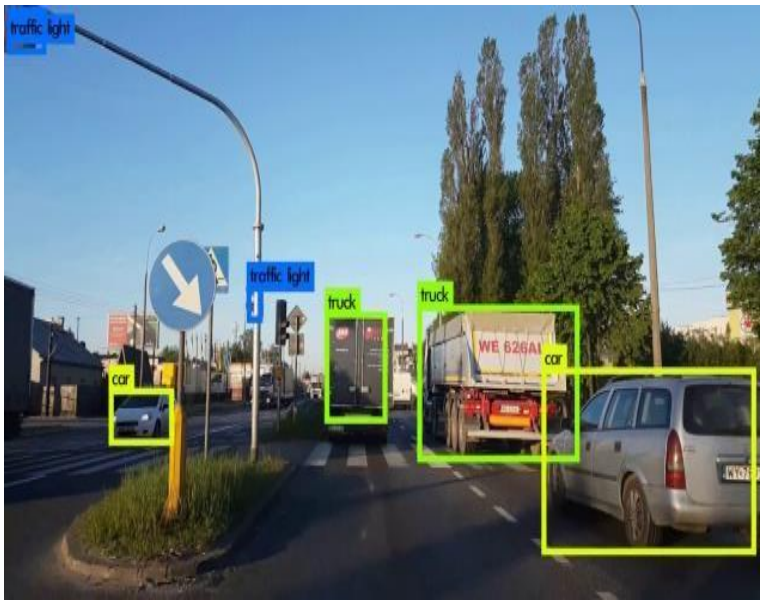
Object Detection and Recognition using In Real Time Camera and Video

AISHWARYA MURKUTE

YIMIN WANG



Abstract

Object recognition is one of the most popular and trending topics in computer vision. This technique usually requires the presence of a data-set with annotations with location information of the objects, which is in the form of bounding boxes around the objects. In this project, we have implemented and custom trained our object to detect them in real time using computer vision using 30 objects. Besides that, we also used the ssd_mobilenet dataset to detect the other objects. We have used a Convolutional Neural Network (CNN) based architecture to perform this task. We have further investigated the suitability of this idea to another application of object counting using supervised training i.e. by providing information of location of objects too (strong labels). To further explore the topic more, we implemented object detection and recognition using one of the most popular algorithms YOLO (You look only once). Using Yolo, we gave the input file as a video to detect the objects in the video. Having an efficient and accurate system which detects the objects accurately is very important in computer vision., and with the help of the various machine learning algorithms it had become possible to do so. The primary aim of our project is to successfully detect the objects in an image and achieve high accuracy in real time.

Introduction

Object Detection is one of the meta-heuristic problems in the real world. Object detecting and recognition is under research in the computer vision technology. One of the best approaches to solve this problem is to use machine learning and cognitive thinking.

Recently, computer vision is one of the most popular research areas in the field of deep learning. Computer vision is actually a cross-disciplinary discipline, including computer science, mathematics, engineering, biology and psychology and so on.

Many scientists believe that computer vision has opened the way for the development of artificial intelligence.
Object detection is an important topic in computer vision, and has great theoretical and practical merits in applications such as visual surveillance, autonomous driving, and human-machine interaction

Object recognition deals with training the computer to identify a particular object from various perspectives in different conditions. It deals with detecting and identifying the presence of various individual objects in an image and categorize them.

Various approaches have been made by computer scientists and engineers to impart vision to the machine. A lot of interest has been shown towards object recognition, object detection, object categorization etc. There were various challenges that we faced during developing this project like the dependency of the computer vision to detection the objects with the help of deep learning algorithms. Not all the algorithms used have the same efficiency and accuracy, this leads to non-optimal performance. In this project, we use a completely focus on using the deep learning-based approach to solve the problem of object detection. The network is trained on the most challenging publicly available dataset (PASCAL VOC) and also use our custom objects to train and detect the objects, on which an object detection challenge is conducted annually. The resulting system is fast and accurate, thus aiding those applications which require object detection the primary goal of our project is to identify the objects in real time and using a live video camera. The project identify about 90 objects (classes) that we trained.

Problem Statement: Object Detection and Recognition using In Real Time Camera and Video

Most of the computer vision are lacking the accuracy. However, with the rise of deep learning techniques, the accuracy of these problems drastically improved. Predicting the class of the image is one of the major problems in object detection. A slightly complicated problem is that of image localization, where the image contains a single object and the system should predict the class of the location of the object in the image (a bounding box around the object). The more complicated problem (this project), of object detection involves both classification and localization.

Literature Survey:

Sr No.	Title	Author and Dates	Contributions
1	Deep Neural Networks for Object Detection	Christian Szegedy Alexander Toshev Dumitru Erhan Google, Inc. {szegedy, toshev, dumitru}@google.com	Research on Deep learning network

2	Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks	Shaoqing Ren∗ Kaiming He Ross Girshick Jian Sun Microsoft Research {v-shren, kahe, rbg, jiansun}@microsoft.com	Faster RCNN learning network
3	Application of deep learning in object detection	Xinyi Zhou ; Wei Gong ; WenLong Fu ; Fengtong Du	Deep learning in object detection task

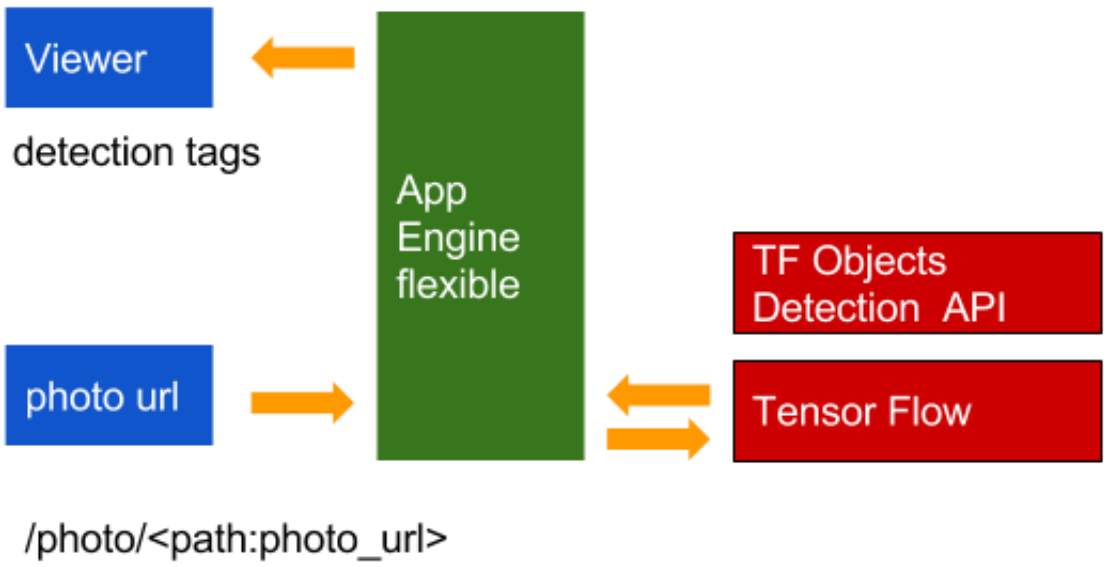
Methodology

There are various approaches for object detection. In this project we are using Google's TensorFlow which is an open source machine learning framework to over-come this problem and identify the object to classify and recognize them. It is of the most popular API's which used for the identification of objects in real world. Also, TensorFlow uses faster RCNN, i.e. Convolutional neural networks to identify the objects in an image**.**

A unified model for object detection is used to identify the objects in an image. The model is easy to construct where it will be trained on a loss function which is directly be proportional to detection and performance of the entire model combined together.

TensorFlow a API is an open source machine learning library which is developed by researchers and engineers at Google's Machine Intelligence research organization which runs on multiple computers to distribute the training workloads.

An object detection API is a an open source framework built on top of TensorFlow which makes it easy to construct, train, and deploy object detection models.



TensorFlow API and Object Detection API

We have used the OpenCV computer vision library for the identification of object in real time and the detection of the objects.

A unified model for object detection is used to identify the objects in an image. The model is easy to construct where it will be trained on a loss function which is directly be proportional to detection and performance of the entire model combined together.

RCNN is used for Selective Search or Edge boxes. It all replaces all the selective search with a very small convolutional network called Region Proposal Network to generate regions of Interest.

Image Classification using RCNN

In place of predicting the class of object from an image, we now have to predict the class as well as a rectangle (called bounding box) containing that object. It takes 4 variables to uniquely identify a rectangle. So, for each instance of the object in the image, we shall predict following variables:

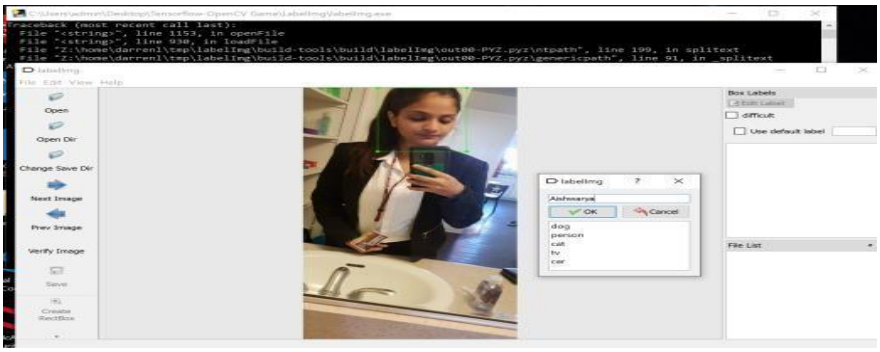
Label 1: class_name,

Variable 1: bounding_box_top_left_x_coordinate,

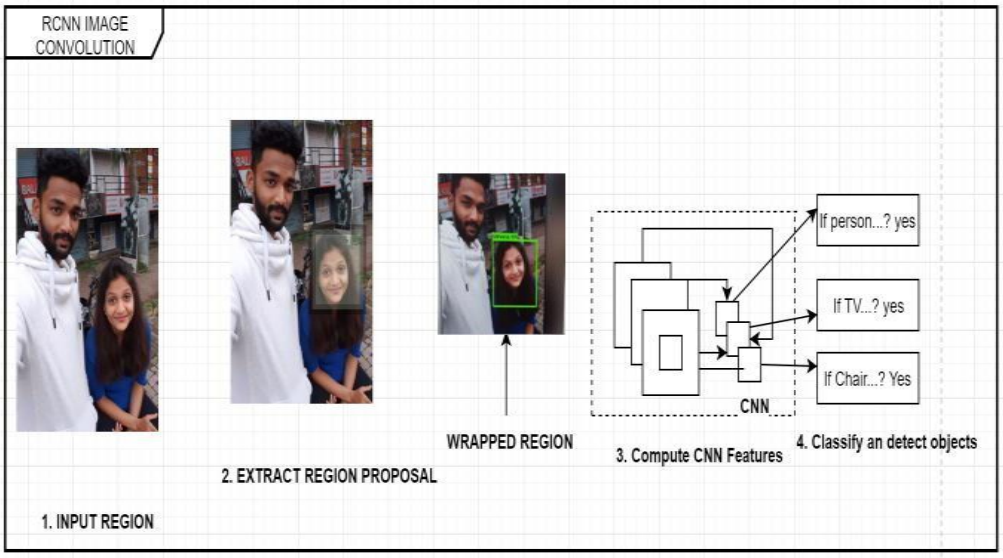
Variable 2 :bounding_box_top_left_y_coordinate,

Variable 3 :bounding_box_width,

Variable 4: bounding_box_height



Drawing bounding boxes to show area of interest

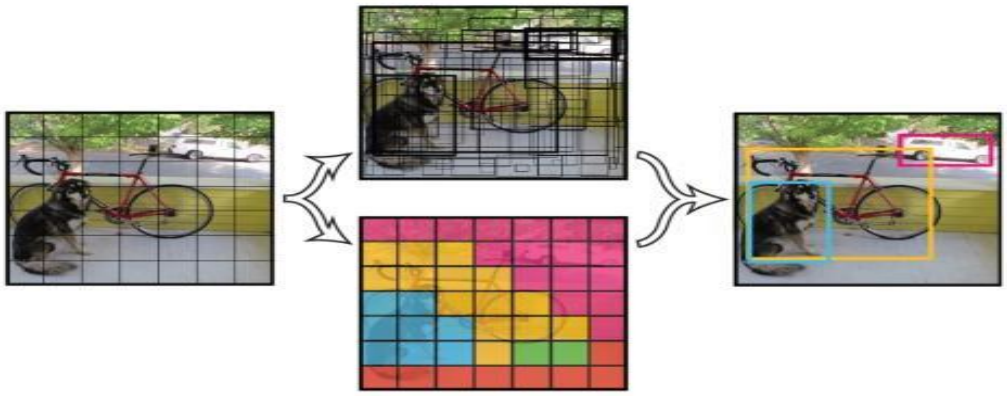


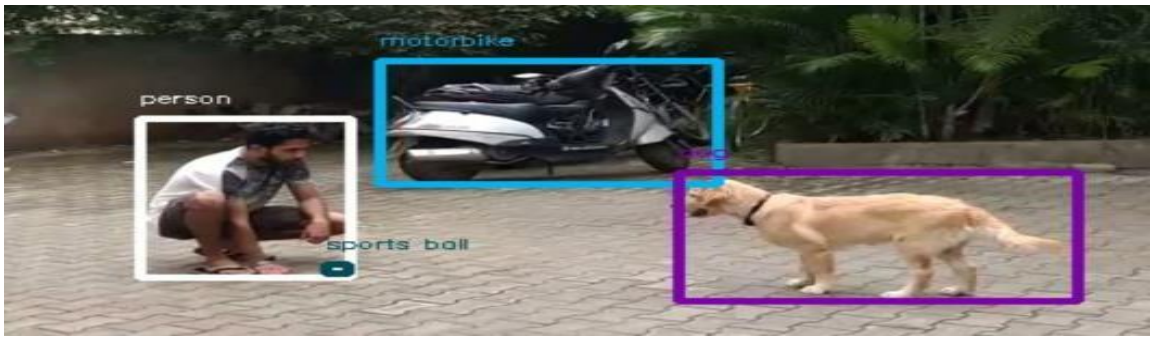
Additional Topics Explored YOLO Algorithm

In this project, we use a completely machine learning based approach to solve the problem of object detection in real time using the most popular technologies. Use YOLO algorithm to detect object when use a MP4 video as an input. We also use YOLO (You Look Only Once) algorithm which is an open source neural network for the detection of object when a MP4 video file is given as an input.

YOLO's entire series are like one-stage detection method. The whole detection framework is based on generating potential bounding boxes, then running a classifier on each proposal box.

YOLO uses an 80-class classifier, and can process video object detection at about 60 FPS on a Nvidia Titan XP, which can be used in real-time applications. The algorithm will output a bounding box on object detected and their classes.





Other Applications

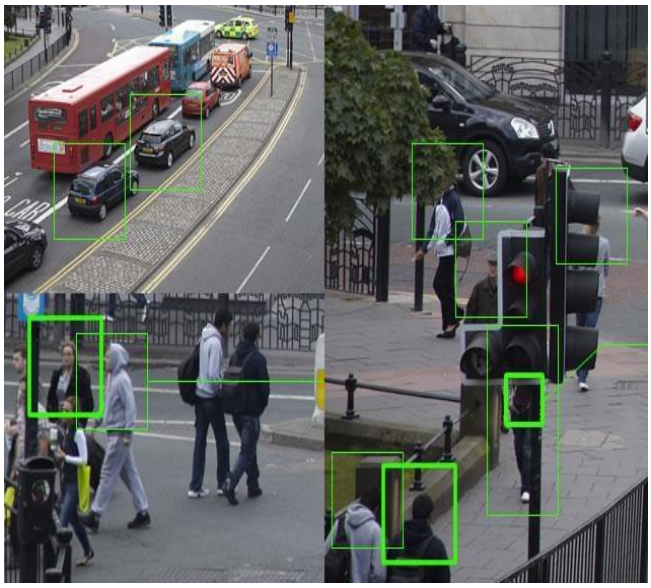
This research field has spawned a large number of fast-growing, practical applications, such as:

Face Recognition: Snapchat, Facebook and other companies are using face detection algorithms to identify faces.

Smart cars: Computer vision is still the primary source of information for detecting traffic signs, lights and other visual features.

This research field has spawned a large number of fast-growing, practical applications, such as:

Monitoring: Surveillance cameras for monitoring suspicious behavior are spread across major public spaces.



Results and analysis

After the successful detection of the various objects there were various observations that were made form the training results and the datasets used. For Convolutional neural networks it was observed that the images can be trained effectively and it reduced the top-5 error from 26.2 % to 15.3 % for large scale visual recognition. The experiments were conducted using TensorFlow and OpenCV. The computation results of deep learning convolutional neural networks are complex.

The results and analysis were seen on the Tensor Board. Tensor Board is used to visualize the quantitative metrics. It is tool used to visualize the TensorFlow graph and plot the execution of the graph and show additional data. It gives a summary of the loss_function, learining_rate and tf. histograms.

We used our own dataset of 300 images to train on the TensorFlow API. The image is converted from. JEPG and .PNG format to XML format and convert the XML format to .CSV format. Following table shows the images with converted to the .CSV format.

	A	B	C	D	E	F	G	H
1	filename	width	height	class	xmin	ymin	xmax	ymax
2	172.JPG	4608	2592	Hand	2448	806	2893	1437
3	194.JPG	2841	2548	Hand	1035	1174	1806	2039
4	195.JPG	3664	2748	Hand	2503	685	3120	1336
5	196.JPG	3664	2748	Hand	2443	782	3000	1285
6	221.JPG	1920	1080	Hand	920	395	1528	896
7	245.JPG	1920	1080	Hand	696	300	1284	988
8	313.jpg	640	640	Hand	43	158	364	581
9	320.jpg	1280	853	Hand	624	299	874	591
10	323.jpg	640	640	Hand	153	82	354	367
11	339.jpg	960	960	Hand	32	9	685	377
12	391.png	1080	1920	Hand	364	780	790	1188
13	725.jpg	1600	1600	Hand	355	430	871	1180
14	726.jpg	1200	1200	Hand	391	421	657	882
15	731.jpg	640	640	Hand	261	227	409	408
16	807.jpg	1280	720	Hand	283	239	578	629
17	808.jpg	1200	1200	Hand	338	519	550	810
18	855.jpg	659	659	Hand	206	92	424	381
19	881.JPG	750	1000	Hand	161	157	381	519
20	882.JPG	750	1000	Hand	401	148	681	594
21	883.JPG	750	1000	Hand	343	125	645	502
22	890.JPG	750	1000	Hand	112	274	390	683
23	B612-2016	768	1024	Hand	132	8	648	652
24	B612-2016	768	1024	Hand	415	366	719	797
25	B612-2016	768	1024	Hand	100	122	637	647
26	B612-2016	768	1024	Hand	74	75	683	677
27	B612-2016	768	1024	Hand	65	57	652	760
28	B612-2016	768	1024	Hand	87	61	604	695
29	B612-2016	768	1024	Hand	115	19	681	578
30	B612-2016	768	1024	Hand	79	118	500	645
31	B612-2016	768	1024	Hand	31	148	398	675
32	B612-2017	768	1024	Hand	90	37	627	620
33	B612-2017	768	1024	Hand	105	93	575	706
34	B612-2017	768	1024	Hand	60	54	406	667

Fig 1: Image data and labelling in CSV format

Since we are using the TensorFlow API, and google TensorFlow does not accept the input for training in the .CSV forma hence we have to convert these files to. TFrecord format.

We feed these. TFrecord file for training and following are the results that we get form the training.

The following graph shows the loss function of the trained model. In order obtain more accuracy we trained the model up to 6000 epochs. (Epochs is the number of steps in the training). If the number of steps is more, the accuracy of the trained model is also more.

The loss function keeps on dropping, based on the number of epochs.

The loss function keeps dropping as the model is trained as seen in graph below form figure 1. This happens because the model is training and learning. The part where we are training the model

is where the machine actually learns and detect the gesture as per the labels. This is what is called as machine learning.

The learning rates increase as the network is trained on various images. It learns more after every step of the training. The following graph shows the initial stages of the training after 300 epochs it is seen that the loss is significant.

Screenshot of training results

Figure 1: Loss Function after 300 epochs

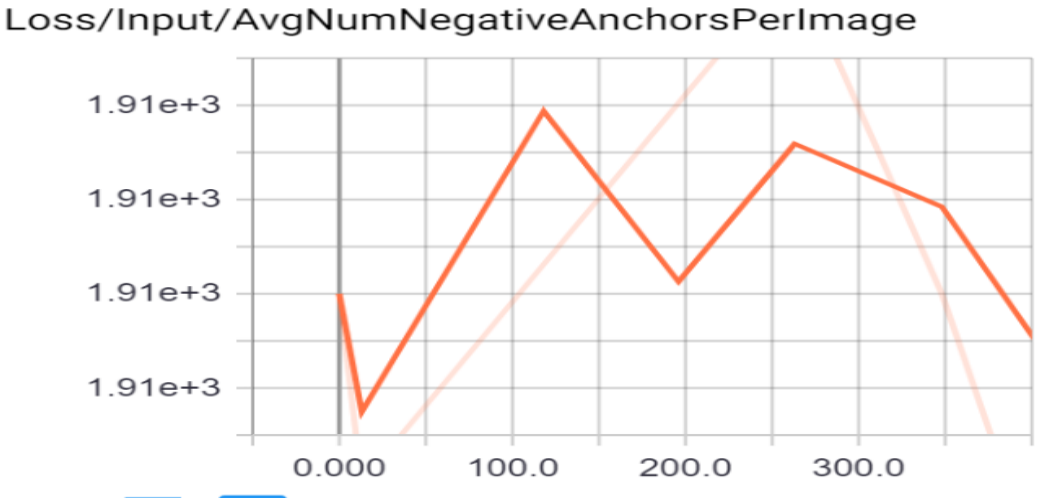


Figure 2: Loss Function after 300 epochs

The above graph shows the results of the loss per input i.e. per image this value is fluctuating initially. Hence, we reach to a conclusion to train the model more.

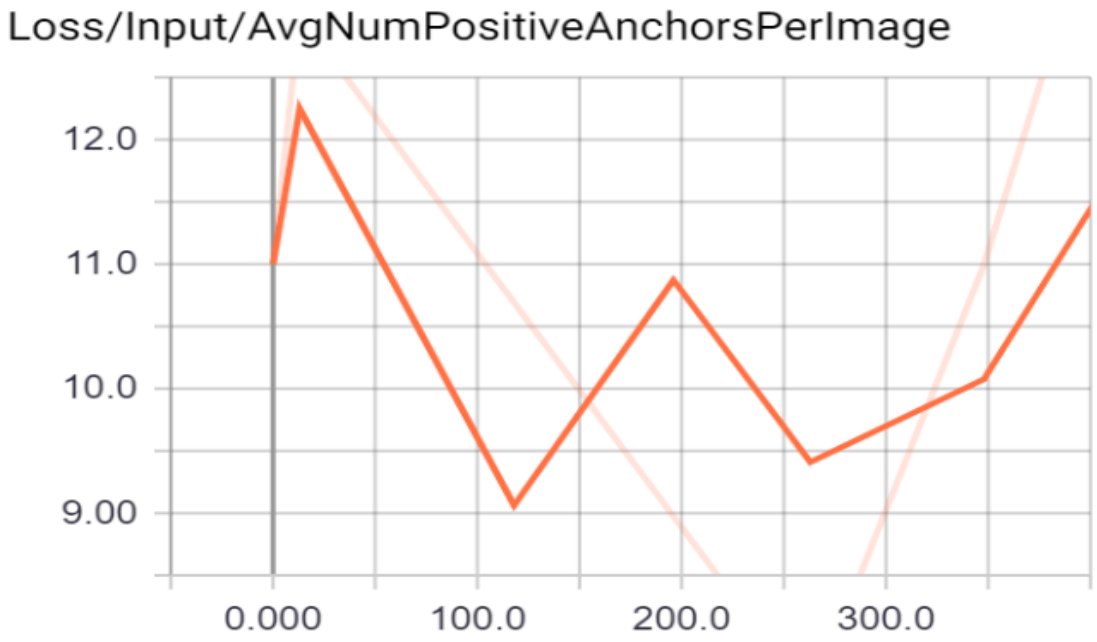


Figure 3: Loss Function after 300 epochs (Fluctuating)

As we see that during the training the model initially is fluctuating a lot hence, we continue to train the model. After, we continue to train the model for more epochs (6000 epochs) it improves the accuracy of the model, as seen in figure 4 because the loss function keeps on dropping.

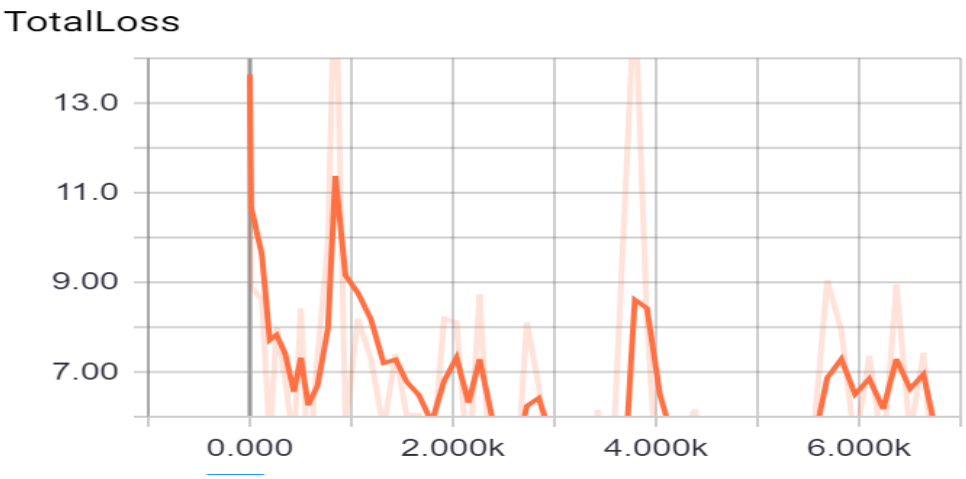


Figure 4: Loss function graph after 6000 epochs

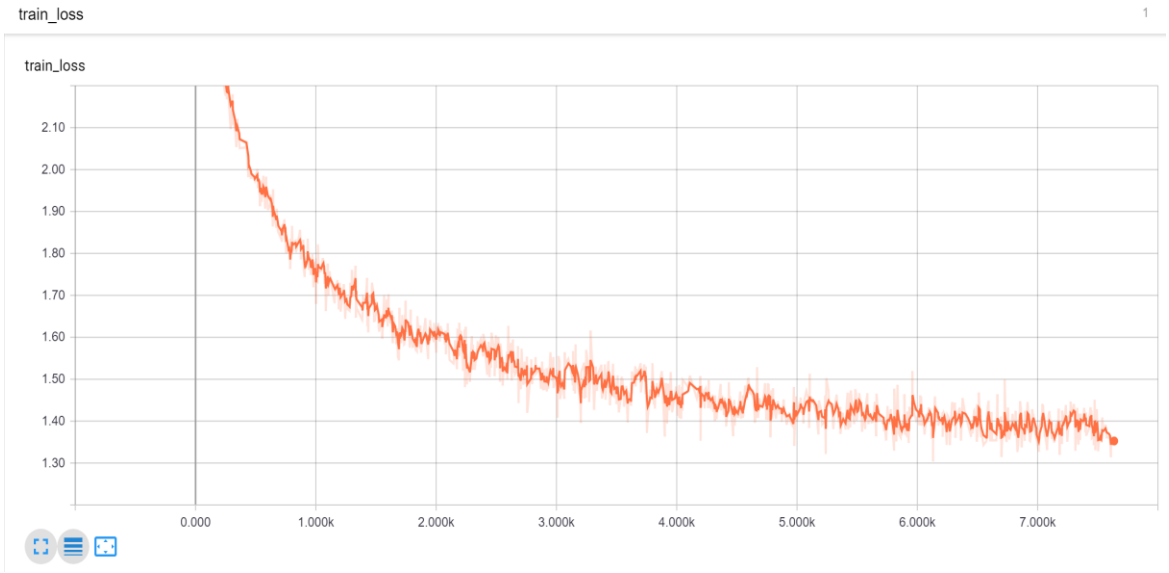


Figure 5: Loss function graph

In figure 5 we can see that there is no significant loss even after 6000 epochs hence the training was terminated after 6000 epochs.

In the below graph it shows the loss per image (or loss per input given to the neural network). We can see that the loss is -0.2 the loss function does not drop beyond this for most of the inputs given to the network. Figure 6 gives the fine-tuned (smoothed value) of the per image loss.

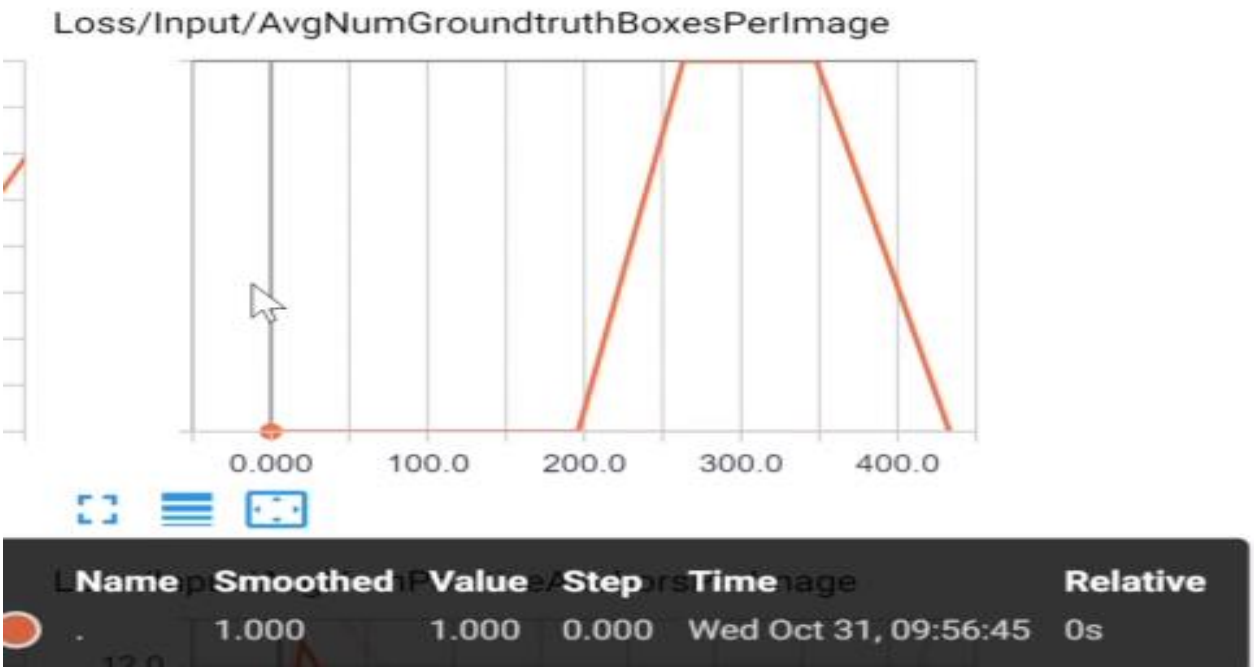


Figure 6: Loss function graph

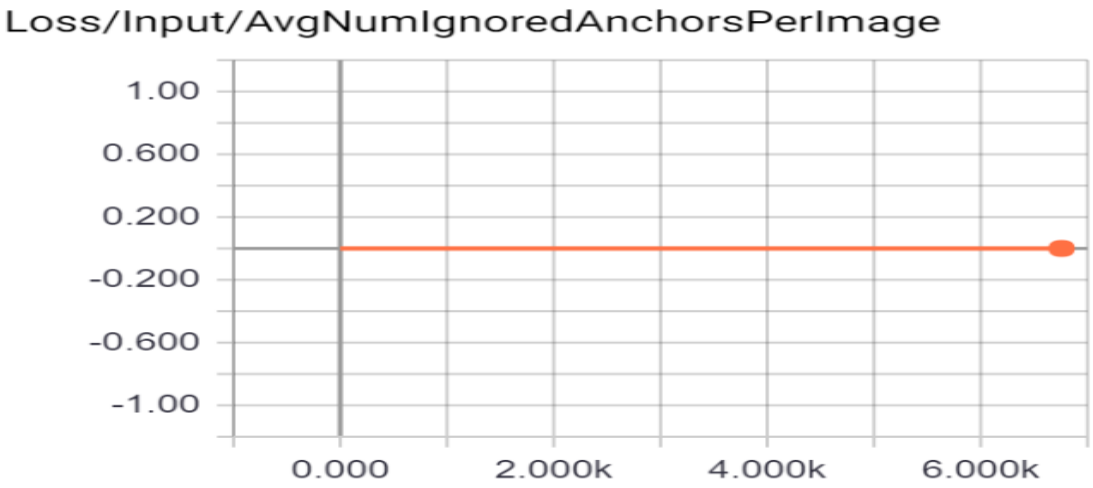


Figure 7: Fine tuning of Loss function graph

In most of the machine learning the learning rate is always inversely proportional to the loss_function during the training. As the loss function keeps dropping the model learns more. Hence the learning rate is inversely proportional to the loss function. The more the neural net learns greater is the recognition accuracy.

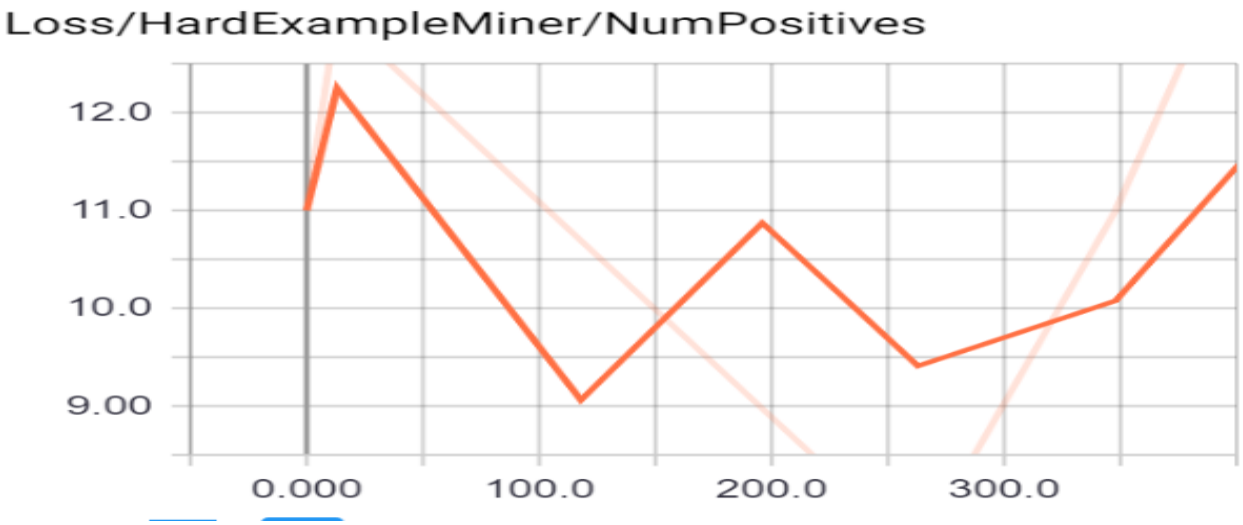


Figure 7: Learning during initial stages of the training

The above graph is the result of the initial stages of training. From figure 2 we know that the loss function during the initial stages is fluctuating and hence is the learning rate of the model (i.e. learning rate after 300 epochs)

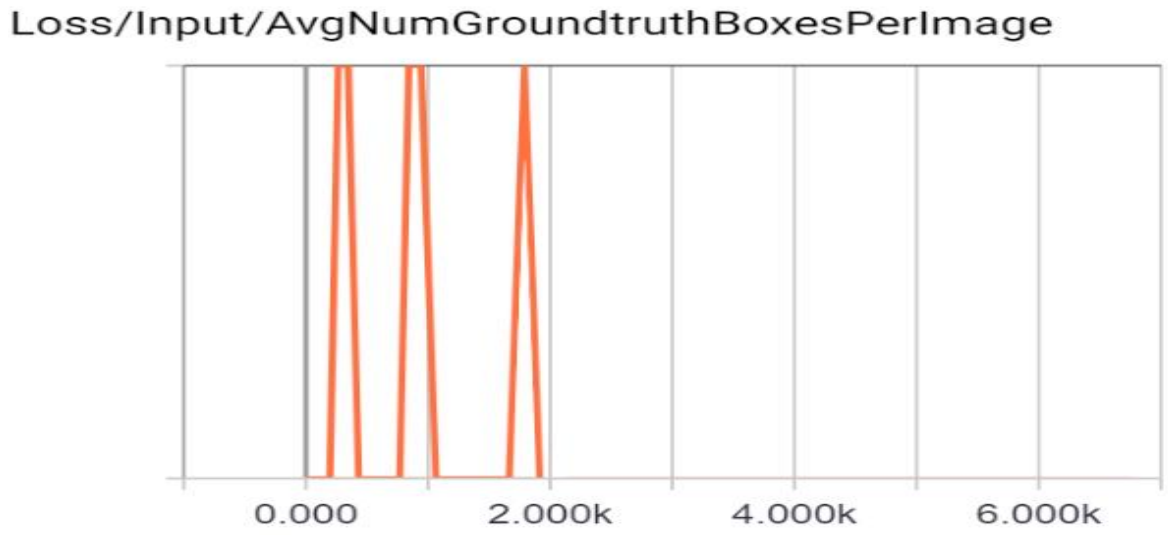


Figure 8: Fluctuating learning rate during initial stages of the training

Following graph shows the batch size for training during the initial stages of training. The batch size is the number of images that are fed to the neural net per epochs. Hence to improve the learning rate we increase the batch size to improve the accuracy.

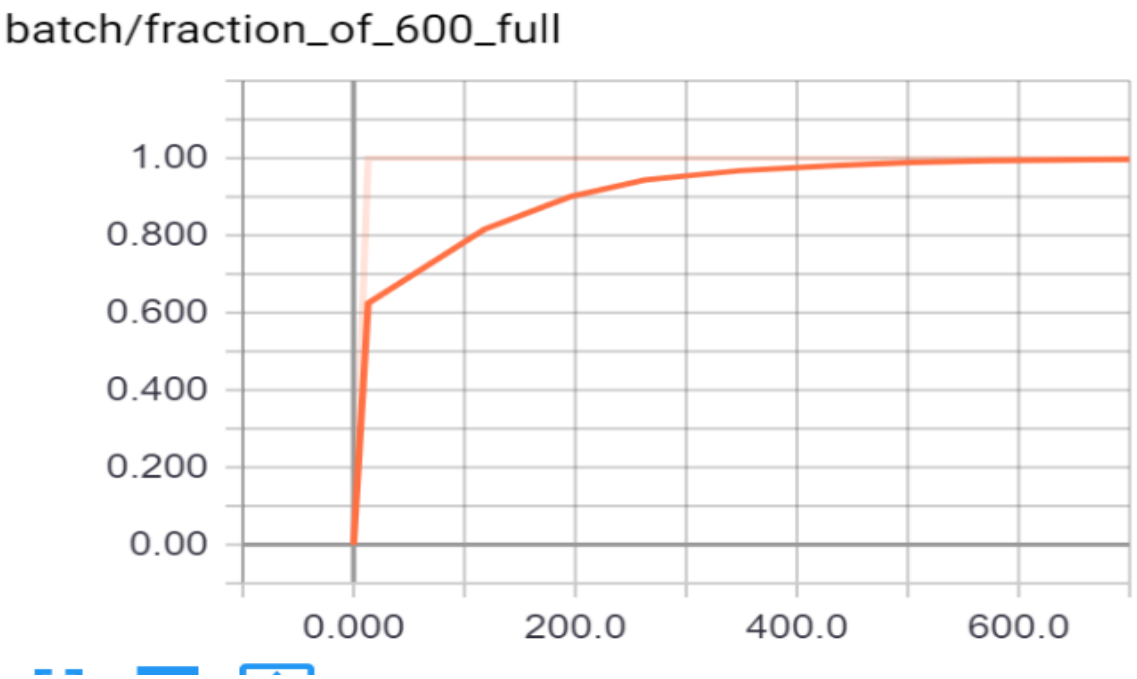


Figure 9: number of images per epochs (i.e. batch size)

Hence to improve the learning rate we increase the epochs. As seen in figure 10

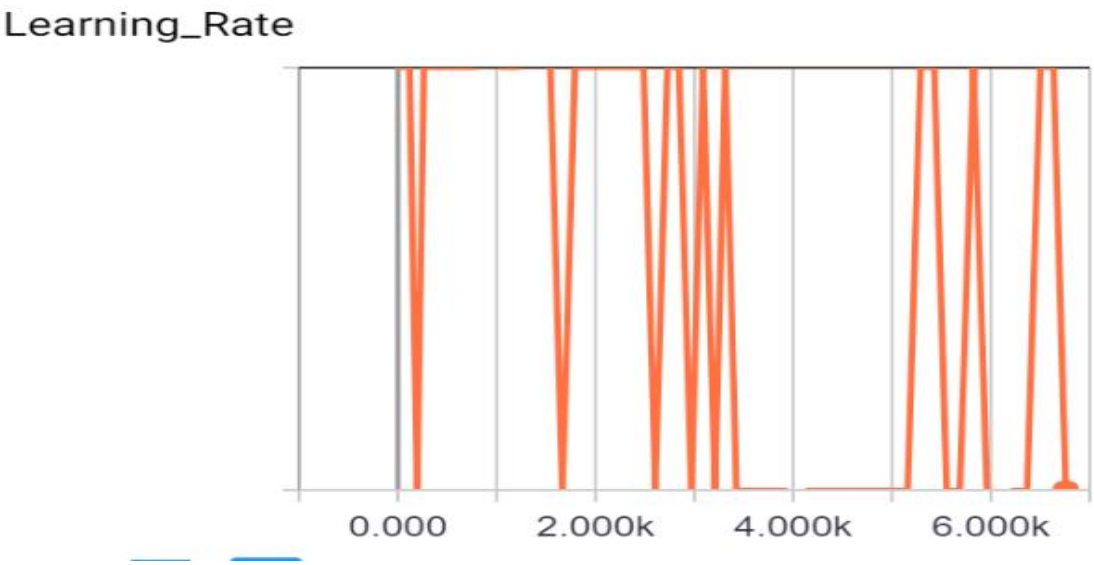


Figure 10: Learning rate

The learning rate increase as the number of epochs increase. As seen in figure 11.

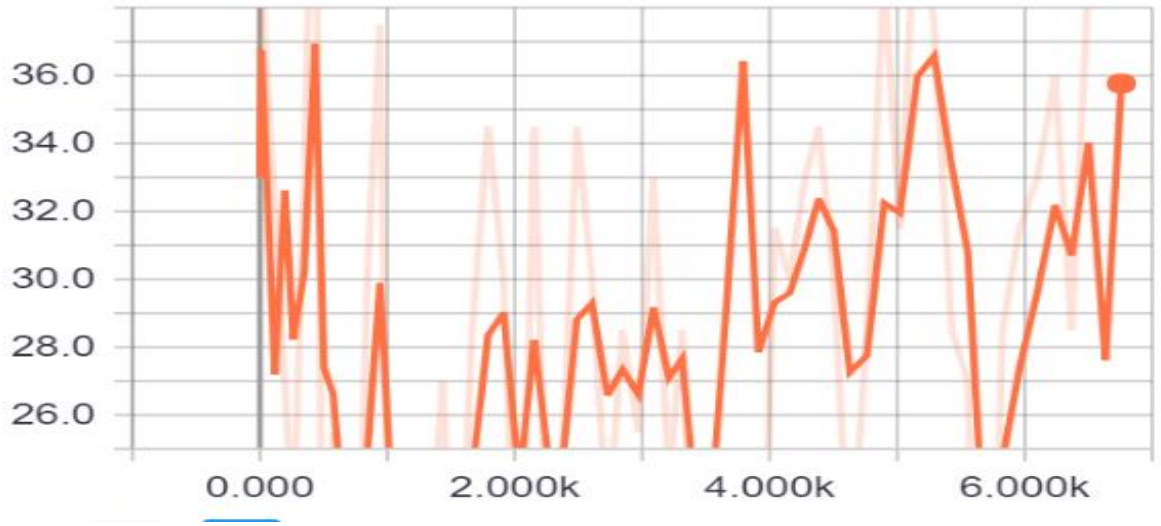


Figure 11 Learning rate

Object Classification and detection performance parameters

For neural networks there are various parameters on which the performance of the model depends. The predictive performance measures are as follows;

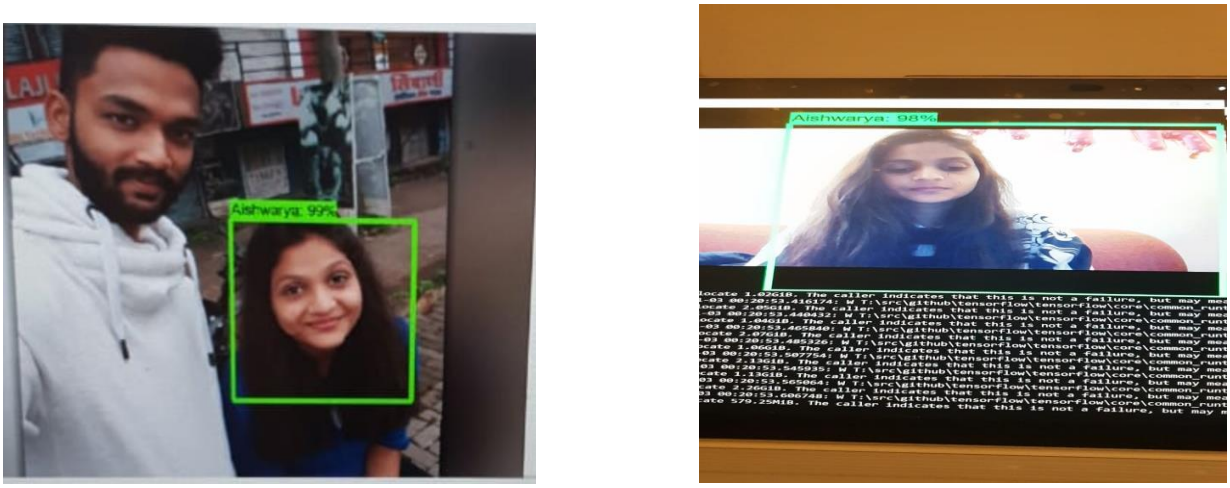
RMSE: Root mean squared value, it used to determine the difference between the actual value and the predicted values (In our case it is the actual input given vs the recognized output.)

$R^{2:}$ It is used to determine the reduction variance of the model.

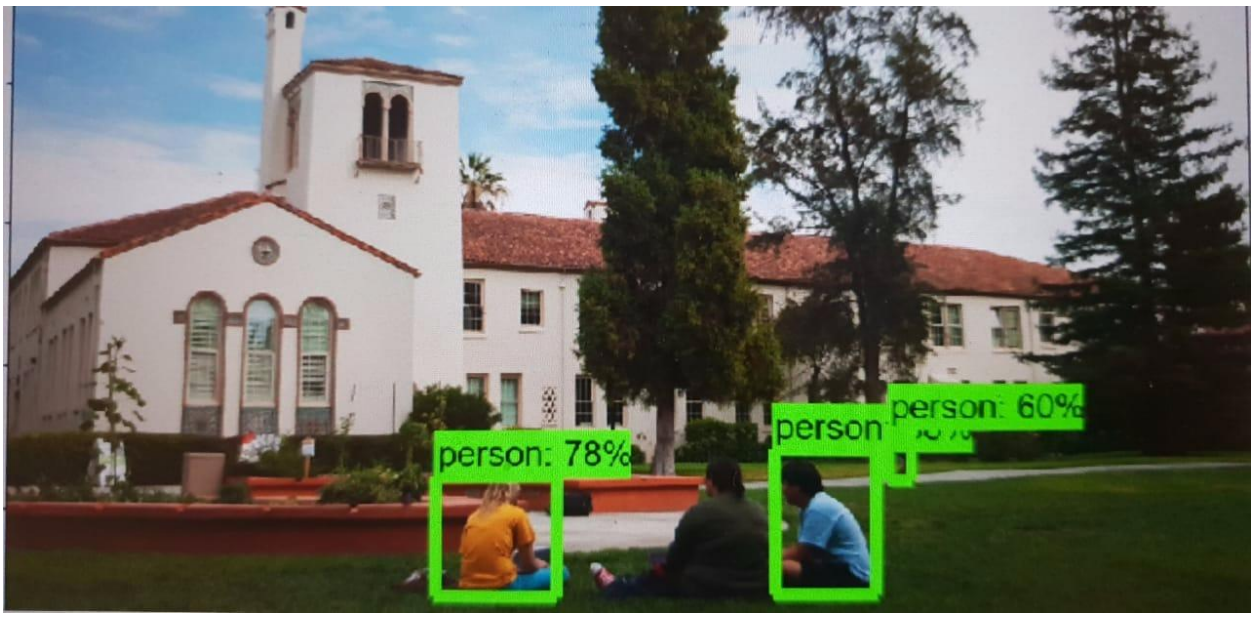
Fig 12: Performance Parameters graph

Results of Object Detection Using Open CV

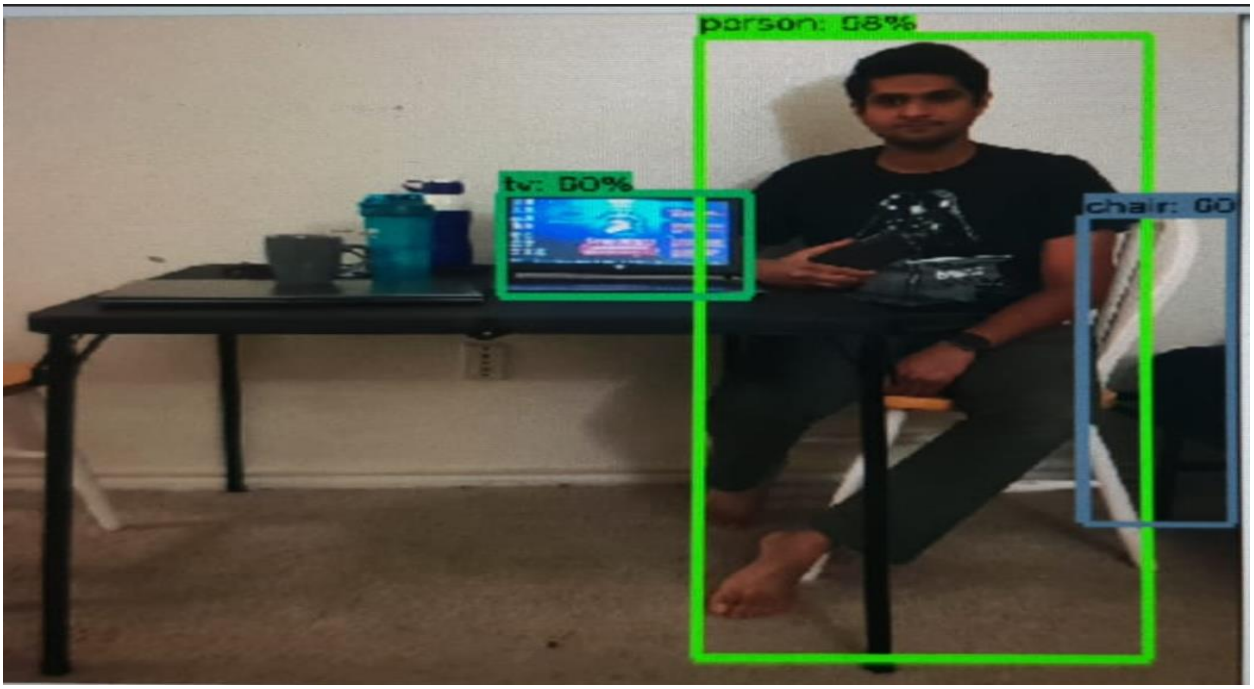
Following are the results that are generated using the OpenCV, computer vision library, it generates the bounding box and the contours of the detected object using CNN it is used for Selective Search or Edge boxes. It all replaces all the selective search with a very small convolutional network called Region Proposal Network to generate regions of Interest (ROI)The OpenCV is used, which is an open source computer vision library which is popularly used for at real-time computer vision.



OpenCV Results: Detection of custom training using image and using live feed from camera.



OpenCV Results: Detection of custom training using image and using live feed from camera.



OpenCV Results: Detection of custom training using image and using live feed from camera.

Conclusion

Hence, we have proposed one of the most versatile methods for object detection and recognition. Further, the method would require a small completely supervised training set so that it can deal with unsupervised datasets that may be added to the system at any stage. The foundation of the proposed project has been laid and the discussion of existing methods and future path adequately demonstrate the feasibility of the proposed approach

References

[1] **ImageNet Classification with Deep Convolutional:** Neural Networks Alex Krizhevsky University of Toronto kriz@cs.utoronto.ca,Ilya Sutskever University of Toronto ilya@cs.utoronto.ca,Geoffrey E. Hinton University of Toronto

[2] **XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks** https://pjreddie.com/media/files/papers/xnor.pdf

[3] **Playing around with RCNN, State of the Art Object Detector** https://cs.stanford.edu/people/karpathy/rcnn/

[4] J. Cabrera and P. Meer, "Unbiased estimation of ellipses by bootstrapping," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, pp. 752-756, 1996.

[4] Mask R-CNN Kaiming He Georgia Gkioxari Piotr Doll´ar Ross Girshick

[5] R. K. K. Yip, P. K. S. Tam, and D. N. K. Leung, "Modification of hough transform for object recognition using a 2-dimensional array," Pattern Recognition, vol. 28, pp. 1733-1744, 1995.

Object Detection and Recognition using In Real Time Camera and Video