Second International Conference on Smart Systems and Inventive Technology (ICSSIT 2019)
IEEE Xplore Part Number: CFP19P17-ART; ISBN:978-1-7281-2119-2

Object Detection and Count of Objects in Image using Tensor Flow Object Detection API

B N Krishna Sai,
Department of Computer Science and Engineering,
Amrita School of Engineering,
Bangalore, India
krishnasaibn@gmail.com

Sasikala T.
Department of Computer Science and Engineering,
Amrita School of Engineering,
Bangalore, India
t_sasikala@blr.amrita.edu

Abstract— **Object Detection is widely utilized in several applications such as detecting vehicles, face detection, autonomous vehicles and pedestrians on streets. TensorFlow's Object Detection API is a powerful tool that can quickly enable anyone to build and deploy powerful image recognition software. Object detection not solely includes classifying and recognizing objects in an image however additionally localizes those objects and attracts bounding boxes around them. This paper mostly focuses on detecting harmful objects like threatening objects. To ease object detection for threatening objects, we have got Tensor flow Object Detection API to train model and we have used Faster R-CNN algorithm for implementation. The model is built on two classes of threatening Objects. The model is evaluated on test data for the two classes of detecting threatening objects.**

Keywords— Deep Learning, Object Detection API, Tensor Flow, Threatening Objects, Faster R-CNN, CNN, Computer Vision

I. INTRODUCTION

Computer Vision(CV) is the science of understanding and manipulating digital videos and images. Computer Vision plays a vital role in many applications, which includes Face recognition, image retrieval, industrial inspection, and augmented reality etc. With the emergence of deep learning, computer vision has proven to be useful for various applications. Deep Learning is an Artificial Neural Network (ANN) collection of methods, which is a machine learning branch. On the human brain, ANNs are modelled where nodes are connected to each other that pass data to each other. The use of deep learning for computer vision can be categorized into various categories: generation, segmentation, detection and classification of both videos and images. Image classification labels the image as a whole Finding the position of the object in addition to labelling the object is called object localization. Typically, the position of the object is defined by rectangular coordinates. Finding multiple objects in the image with rectangular coordinates is called detection. Segmentation is detecting exact objects like creating a transparent mask above the object with exact edges. An image classification or model of image recognition merely detects an object's likelihood in an image. In comparison the location of objects relates to the place of an item in the picture. A localization algorithm for object will output the place coordinates of an item with regard to the picture or image.

Object detection is a problem of importance in CV. Similar to image classification tasks, deeper networks have shown better performance in detection. At present, the accuracy of these techniques is excellent. Hence it used in many applications. The difference is the number of objects. In detection, there are a variable number of objects. This small difference makes a big difference when designing the architectures for the deep learning model concerning localization or detection.

II. PRIOR WORK

The Computer Vision is growing exponentially as the technology is growing exponentially. In this area, a lot of work has happened for the continuous growth and improvements in the domain of Computer Vision. To witness the growth and improvements in this area many researchers follow different methods and approaches to a problem. Always researchers will be kept on digging to find improvements. This section explains about the previous works which have done so far using different methodologies followed. Firstly, we start with different applications of object detection and then we got to the background of the implementing algorithm.

Apoorva Raghunandan and et. al [1], has made work on different algorithms such as colour, skin and face detection are simulated and implemented using MATLAB for detecting different objects in video surveillance applications to improve accuracy. Viola jones algorithms are used for face detection. When they had input an image detected face the algorithm detects all the features of a face like eyes, nose etc. In skin pixels, skin detection and the non-skinning pixels has been detected. Skin detection four cases have been considered from a single face and output binary images for different cases has been obtained. When in case of multiple people skin detections, people were made to be seated in various positions. The skin complexion and different colour clothing have been observed. Then colour detection has performed to know the accurate object detections. It helps in classification of different

objects. In simulation, it has detected various shapes and different shades in colour image. Then comes another part of it has been target detection in this fore ground has been cleaned up and foreground shadows are detected with a threshold of 80 and it has been done for different cases with a difference of 20. Finally using various object detection algorithms and simulated in MATLAB has given accuracy of 95%.

Jung Uk Kim and et. al [3], has worked on object detection in road scene. This object detection in road scene drawn very important attention. Occlusion problems occur very frequently road scenes. As the previous research has limitations of not detecting the object properly. They proposed a novel approach of detecting objects which is robust in occlusions. In this, it contains mainly two parts which are using framework and object bounding box. In the framework part, it will perform classification bounding box regression Object detection framework has used VGG16 network which is part of feature encoding. Based on the results of KITTI Vision Benchmark suite dataset has shown that the proposed object network model outperformed different state of art methods.

Shantanu Deshmukh and et. al [4], has come out with object detection solar panel layout generation. Roof with obstacles and edges are marked on panel layout diagram. Generally, user will draw a boundary manually over each and every obstacle in a meticulous and tedious manner. A framework has been built on existing object detection models. which leveraged energy from general traditional edge detection algorithm's, which are fusing with cutting-edge machine based on the frameworks. In the proposed solution an approach termed "Novel" fusion is applied. We firstly put in object detection API then after each detection, put in edge detected algorithms. Object detection API gives bounding boxes on original image. From the edge detection output candidate image fused back original roof image. Exact edges have mapped with obstacles by creating a layout. Proposal described here having significant impact and highly effective on object detection API's. Results in framework is capable of detecting objects. The boundaries in a solar panel are automatically generate pixel count which has a variation of 25 less of the ground truth.

Waritchana Rakumthong and et. al [14], proposed a new method which supports a smart surveillance system which can detect the stolen objects and abandoned in public areas. The system has been implemented using image processing techniques. Immediately it will alert responsible people like guards etc. It has four major components like acquisition, processing, detection and presentation. The experiment has been conducted to detect whether the system is able to access the qualities of usability and correctness. In order to do the experiment, the system has taken a video from the CCTV and then detects the objects using image processing techniques acquired from decision-making. The outcome of processing can be viewed via computer screen or a TV Screen user Interface. In this classification of objects has given approximately 76% and event classification of 83%.

Donghoon Kim and et. al [15], work has been carried out in detection of objects and tracking of underwater robots using template matching. As the environment under water is like noisy and it has very low light so the detection has some cons due to the poor visibility. In this they proposed vision-based tracking techniques for underwater robots using artificial objects and proposed a novel weighted correlation approach using the feature-based performance matching in different illumination conditions. The conventional method has required a different threshold for matching the different shapes where the threshold is more than 0.8. The proposed model has taken a threshold of 0.5. Since underwater is noisy pre-processing has done by using camera calibration and Gaussian smoothing methods to compensate distortion and noise.

From 2012, when Krizhevsky et al.[16] won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) by relea sing AlexNet where deep learning dominated computer vision in various aspects. In 2014, Girshick et al. [13] show cased the advantages of the convolutional neural network for designing new network for detecting the object which was named as Region-based Convolutional neural network or R-CNN. Selective Search algorithm took huge annotated datasets to train a network with R-CNN, but the datasets available for object at that time were scarce. Girshick et al. [13] uses ImageNet 2012 classification dataset to pre-train the CNN which has only image level annotations (no bounding box around image) which solves the scarcity problem. Then, this network is modified and worked with two different datasets, PASCAL VOC 201 and ImageNet 2013 dataset with bounding boxes. To improve the training process of R-CNN, In 2015 Girshick [11] proposed an algorithm called faster object detection algorithm - Fast R-CNN. In Fast R-CNN, an image input is fed to a single CNN having many convolutional layers which generates a convolution feature map. The main advantage of Fast R-CNN involves training an entire image with only one CNN instead of training the images with multiple CNNs for all the region of an image.

III. IMPLEMENTATION

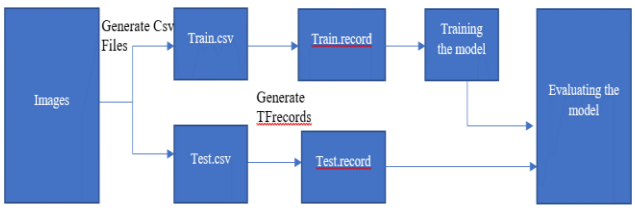

Fig 1 Work Flow

The above Fig 1, shows the work flow of our work. In this paper image data set has been created using threatening object images which is taken from google images. We have taken nearly 78 object images out of which the images splitted into train and test images. We train the model on train images and evaluate model using the test performances. Firstly, we start from labelling the images. We label the images using LabelImg tool we create a rectangular box around the object which gives co- ordinates of the object where it lies. The image data is initially stored in xml format for each image. As the number of images is more we will have a same number of xml files to avoid complexity. So, we create a csv file which has data of all the images. Since we are working with image dataset the size of images is more so using a binary file format to store our data has an important impact on performance of the model. Binary data occupies very little space on disk, consumes less time to copy and can be read more efficiently from disk. So we convert data into record file format. The main advantage of using TFrecord file format is that the data can be optimized in multiple ways. This is an advantage especially for data sets that are too large to store it in memory, as the required data (e.g. a batch) is loaded from the disk as and when required and then processed. After generation of tf records we create a label map which gives a unique id for each of the category to identify. We have used a trained model of Faster RCNN. The default parameters changed to our use model by using config file. Model config block is about configuration of a model. These config files are used to configure parameters to the initial setting for some of the computer codes. Here, each ModelConfig specifies one model to be served, including its name and the path.

After initialization of everything we train our based on our system configuration. When each training phase has begun, the loss will be reported. As the training method progress, it will begin high and get smaller and lower. We are training on model Faster RCNN Inception V2 model. It started at 3 and fell down rapidly. In this we have trained nearly 5000 steps until loss is constantly less than 0.1. It took almost twelve hours to train the model. The model may train faster on powerful cpu and gpu. After completing the training process, we export an inference graph by using checkpoints that are created while training a model. This checkpoint won't contain any description of the computation defined by the model. Weights from these checkpoint files are inserted into variable operations.

After finishing training from Client script, we access the api from our side by giving the test images to detect the objects in an image. These images are led to the tensor flow serving server. TensorFlow Serving is a versatile, high-performance serving system intended to produce environments different machine learning models. TensorFlow Serving makes deploying fresh techniques or algorithms and experiments straight forward while maintaining same server design and APIs. The input image we have given are returned with Bounding boxes around the object. These Bounding boxes are imaginary boxes which are around objects that are being verified for collision, objects like pedestrians on or near to the road, other signs and vehicles. Here we used threatening objects the output image is given around those threatening objects. There is a 2D coordinate system and a 3D coordinate system that are both being used. In digital image processing, the bounding box is only the coordinates of the oblong border that absolutely covers the image when placed over a page, a screen, a canvas, or other comparable bi-dimensional background. These bounding boxes are appeared based on the detection classes and detection scores. Detection classes in object detection are the craft of identifying instances of exact class, like humans, animals and many more in a video or image. To differentiate between two objects in any image or video. Detection score is to interpret the outcome, we can explore for the score and the location of each detected object. The detection score is range between 0 and 1 which gives confidence that the object was genuinely detected. The detected score is nearer to 1, the more confident the model is. We can decide the cut-off threshold and below the threshold will be discarded. The input image fed is given as output based on detection scores where the detection score matches to a label it returns with class on top of bounding boxes.

Fig 2 Test Case 1 as Gun

The above Fig 2, shows one of the test cases that out four objects i.e., out of four guns it has detected only one of them. As we have trained nearly 500 steps of loss. When we checked the result after 500 steps the image with only one gun has detected a greater number of objects. To get the accurate results we need to train more steps.

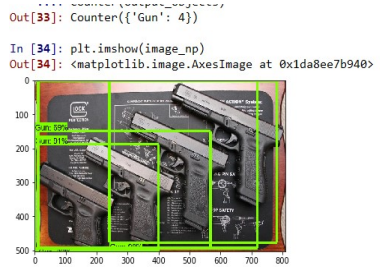

Fig 3 Result after training complete model

In the above fig 3 it shows that four guns has been detected and the count of objects is also given as 4. Fig 3,

shows the output after final model has built with a greater number of steps which gives correct detection and object count in an image.

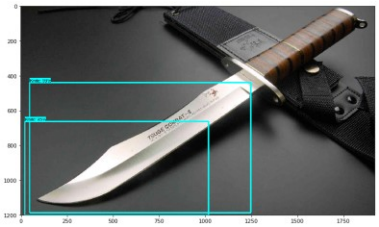

Fig 4 Test Case 2 as Knife

The above Fig 4, shows one of the test cases that out one object i.e, out of One Knife in an image it has detected two are appeared in an image when we have actually in one image. As we have trained nearly 500 steps of loss. When we checked the result after full model is built we greater number of steps the image with only one Knife has detected and the Count of image also given as same.

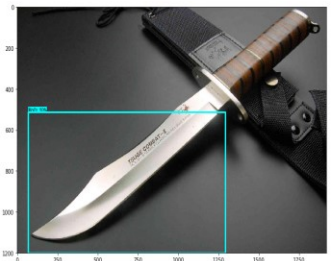

Fig 5 Result after training complete model

In the above Fig 4 it shows that Only Knife has been detected and count is always given from the number of objects detected in image i.e., bounding boxes number in an image.

IV. RESULTS

The network model is tested with our model with the test image as input using Faster R-CNN. As shown in the Fig. 5, and Fig. 6, the model which we built method detected objects using Faster R-CNN. We have already seen some test cases in implementation part. From test cases in implementation we know that more you train better the performance of model. In test we have taken 11 objects out of which 7 are gun and 4 are Knife 6 of them are predicted as True Positive which means actual and predicted are same 1 is wrongly classified similarly for knife objects 1 is wrongly classified which gives an accuracy of 81.81 % accuracy of the model. Now we test the model with some random images.

The below Fig. 5, shows results of the images with detected objects and count of objects in an image. In the First fig we have given as input to the system of image which containing two images. The output has predicted as knife with a very good accuracy and the count of objects is same as the in the input image.

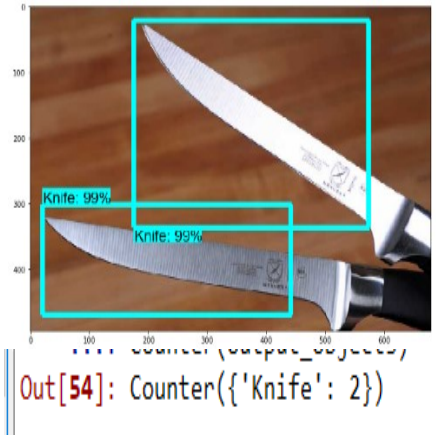

Fig 6 Detected Objects and Count in Image

As we have two classes then we have input with different class which is gun. The same image we have given as input previously when we stopped to train the model with a smaller number of steps where it has predicted three guns in the same image now it shows only two after the model has trained.

Fig 7 Detected Objects and Count in Image

V. CONCLUSION

In this paper, we built a model using Faster R-CNN which considers the detection of threatening objects in an image. We built the model using Object Detection API. We trained model nearly 4500 steps to get a loss under 0.1 which took twelve hours and when we test the model with test images it performs well by giving better results. The future work includes to further enhancing efficiency of model by training big number of images and to train model a higher number of steps for better results.

REFERENCES

[1] Raghunandan, Apoorva, Pakala Raghav, and HV Ravish Aradhya. "Object Detection Algorithms for video surveillance applications." In 2018 International Conference on Communication and Signal Processing (ICCSP), pp. 0563-0568. IEEE, 2018

[2] Yu, Liyan, Xianqiao Chen, and Sansan Zhou. "Research of Image Main Objects Detection Algorithm Based on Deep Learning." In 2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC), pp. 70-75. IEEE, 2018.

[3] Kim, Jung Uk, Jungsu Kwon, Hak Gu Kim, Haesung Lee, and Yong Man Ro. "Object Bounding Box-Critic Networks for Occlusion-Robust Object Detection in Road Scene." In 2018 25th IEEE International Conference on Image Processing (ICIP), pp. 1313-1317. IEEE, 2018

[4] Deshmukh, Shantanu, and Teng-Sheng Moh. "Fine object detection in automated solar panel layout generation." In 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 1402-1407. IEEE, 2018.

[5] Abhilash, M. S. K., Amrita Thakur, Deepa Gupta, and B. Sreevidya. "Time Series Analysis of Air Pollution in Bengaluru Using ARIMA Model." In Ambient Communications and Computer Systems, pp. 413-426. Springer, Singapore, 2018.

[6] Aki, Aravindh, D. Krishna Mohan Reddy, Y. Koushik Reddy, C. R. Kavitha, and T. Sasikala. "Analyzing the real time electricity data using data mining techniques." In 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon), pp. 545-549. IEEE, 2017.

[7] Venkataraman, D., Nandina Vinay, TV Vamsi Vardhan, Sai Phanindra Boppudi, R. Yogesh Reddy, and P. Balasubramanian. "Yarn price prediction using advanced analytics model." In 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), pp. 1-8. IEEE, 2016.

[8] Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "Ssd: Single shot multibox detector." In European conference on computer vision, pp. 21-37. Springer, Cham, 2016.

[9] Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." In Advances in neural information processing systems, pp. 91-99. 2015.

[10] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2015.

[11] Girshick, Ross. "Fast r-cnn." In Proceedings of the IEEE international conference on computer vision, pp. 1440-1448. 2015.

[12] Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang et al. "Imagenet large scale visual recognition challenge." International journal of computer vision 115, no. 3 (2015): 211-252.

[13] Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580-587. 2014.

[14] Rakumthong, Waritchana, Natpaphat Phetcharaladakun, Wichuda Wealveerakup, and Nawat Kamnoonwatana. "Unattended and stolen object detection based on relocating of existing object." In 2014 Third ICT International Student Project Conference (ICT-ISPC), pp. 115-118. IEEE, 2014.

[15] Kim, Donghoon, Donghwa Lee, Hyun Myung, and Hyun-Tak Choi. "Object detection and tracking for autonomous underwater robots using weighted template matching." In OCEANS, 2012-Yeosu, pp. 1-5. IEEE, 2012

[16] Alex Krizhevsky, Ilya Sutskever, and Geo_rey Hinton. Imagenet classi_cation with deep convolutional neural networks. In Advances in Neural Information Processing Systems, pages 1097-1105, 2012.

[17] Koh, Jia Juang, Timothy Tzen Vun Yap, Hu Ng, Vik Tor Goh, Hau Lee Tong, Chiung Ching Ho, and Thiam Yong Kuek. "Autonomous Road Potholes Detection on Video." In Computational Science and Technology, pp. 137-143. Springer, Singapore, 2019.

[18] Oyebode, Kazeem, Shengzhi Du, Barend Jacobus Van Wyk, and Karim Djouani. "A sample-free Bayesian-like model for indoor environment recognition." IEEE Access 7 (2019): 79783-79790.