iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** III **Month of publication:** March 2022

DOI: https://doi.org/10.22214/ijraset.2022.40755

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue III Mar 2022- Available at www.ijraset.com

Real Time Object Detection using SSD and MobileNet

Tejal Palwankar[1], Kushal Kothari[2]
[1, 2] *Department of Computer Engineering, Rajiv Gandhi Institute of Technology*

Abstract: ***This Accurate Real-time object detection needs faster computation power to identify the object at that specific time. The accuracy of object detection has increased drastically with the advancement of deep learning techniques. We incorporate a state-of-the-art method for object detection to achieve high accuracy with real-time performance. The state-of-the-art methods are subdivided into two types. The first is one-stage methods that prioritize inference speed, and example models include YOLO, SSD, and RetinaNet. The second is two-stage methods that prioritize detection accuracy, and its example models include Faster R-CNN, Mask R-CNN, and Cascade R-CNN. Among all these, Faster-RCNN and SSD have better accuracy, while YOLO performs better when speed is given preference over accuracy. A major challenge in many of the object detection systems is that it is dependent on the other computer vision techniques for helping the deep learning-based approach, which results in slow and non-optimal performance. In this paper, we have used a deep learning-based approach to solve the matter of object detection in an end-to-end fashion. Deep learning combines SSD and Mobile Nets to perform the efficient implementation of detection and tracking.SSD eliminates the feature resampling stage and combined all calculated results as a single component. MobileNet is a lightweight network model that uses depth-wise separable convolution for the places which lacks computational power like mobile devices (eg: laptop, mobile phones, etc). This algorithm performs efficient object detection while not compromising on the performance The main purpose of our research is to elaborate the accuracy of an object detection method SSD and the importance of the pre-trained deep learning model MobileNet. The resultant system is fast and accurate, thus aiding those applications which require object detection***

Keywords: Object Detection, CNN, TensorFlow object detection API, SSD with MobileNet

I. INTRODUCTION

Many issues in computer vision were saturating on their accuracy before a decade. However, with the increase of deep learning techniques, the accuracy of those problems drastically improved. One of the key problems was that of image classification, which is defined as predicting the class of the image. A rather complicated problem is that of image localization, where the image contains one object and therefore the system should predict the class of the location of the object within the image (a bounding box around the object). The more complicated problem, of object detection, is that it involves both classification and localization. In this case, the input to the system is going to be an image, and therefore the output is going to be a bounding box corresponding to all the objects within the image, along with the class of objects in each box. We implemented a system which detects objects fast and operates at higher FPS while consuming less computational power than the current systems.

II. PROPOSED SYSTEM

There are mainly three steps in an object detection framework.

1) Firstly, an algorithm or a model is employed to get regions of interest or region proposals. These region proposals are nothing but a large set of bounding boxes spanning the full image.
2) The second step consists of visual features that are extracted for each of the bounding boxes, they are further evaluated and it is determined whether and which objects are present in the proposals are based on visual features (i.e., an object classification component).
3) In the third step which is the final post-processing step, overlapping boxes are combined into a single bounding box (that is, non-maximum suppression).

A. Region Proposals

Several different approaches exist to generate region proposals. Initially, the 'selective search' algorithm was used to generate object proposals. A selective search is a clustering-based approach that tries to group pixels and then generates proposals based on the generated clusters. Several other approaches use more complex visual features extracted from the image to generate regions (for example, based on the features from a deep learning model) or acquire a brute-force approach to region generation. These brute-force approaches are similar to a sliding window that is applied to the image, over various ratios and scales.

These regions are generated automatically, without taking into consideration the image features. An important trade-off that's made with region proposal generation is the number of regions vs. the computational complexity. We find the object based on the regions we generate. On the contrary, if we exhaustively generate all possible proposals, it won't be possible to run the object detector in real-time, for instance. Sometimes, it is possible to use problem-specific information to reduce the number of ROI's. For example, pedestrians have a ratio of approximately 1.5, so it is not useful to generate ROI's with a ratio of 0.25.

III.METHODOLOGY

The Tensorflow Detection API brings together a lot of the foregoing ideas together in a single package, allowing to quickly repeat over different configurations using the Tensorflow backend. With the API, we define the object detection model using configuration files and therefore the Tensorflow Detection API is responsible for structuring all the necessary elements together.

A. SSD (Single Shot Multibox Detector)

The researchers from Google published an SSD architecture in 2016. It presents an object detection model employing a single deep neural network combining regional proposals and feature extraction. A set of default boxes over various aspect ratios and scales is used and applied to the feature maps. The feature maps are computed by passing an image through an image classification network, thus the feature extraction for the bounding boxes are extracted in a very single step. Scores are generated for every object category in every of the default bounding boxes. To better fit the bottom truth boxes, adjustment offsets are calculated for every box.

Different feature maps within the convolutional network correspond with different receptive fields and are utilized to naturally handle objects at different scales. As all the computation is enclosed in a single network and fairly high computational speeds are achieved (for example, for 300 × 300 input 59 FPS). For the usage, we are going to investigate the various sample configuration files for SSD. Several parameters are important when leveraging the SSD architecture and we will re-evaluate them one by one.

First, various classification networks have different strengths and weaknesses (see this blog post for an overview). The Inceptionv3 network as an example is trained to detect objects well at different scales, while on the contrary, the ResNet architecture achieves very high accuracy overall. Mobilenet on the other hand is a network that was trained to minimize the specified computational resources. The number of parameters, the performance of the feature extraction network on ImageNet, and therefore the original dataset it had been trained on are a good proxy for the performance/speed trade-off. The feature extractor is defined within the 'feature_extractor' section.

The second set of parameters is the settings for the default boxes and aspect ratios. Depending on the kind of problem, it's worthwhile to analyse the scales of the bounding boxes of the labeled data and the various aspect ratios. Setting the aspect ratios and scales will ensure that the network doesn't do unnecessary calculations. We will be able to tweak these within the 'ssd_anchor_generator' section. Note that adding more scales and aspect ratios will result in better performance, but typically with diminishing returns.

Thirdly, when training the model, it's important to set the image size and data augmentation options within the 'data_augmentation_options' and 'image_resizer' sections. Larger image sizes will perform better as small objects are often hard to detect, but they'll have a significant computational cost. Data augmentation is very important within the context of SSD to be able to detect objects at different scales (even at scales that could not be present in the training data). Finally, tweaking the 'train_config', setting the learning rates and batch sizes is very important to reduce overfitting, and can highly rely on the size of the dataset we've got.

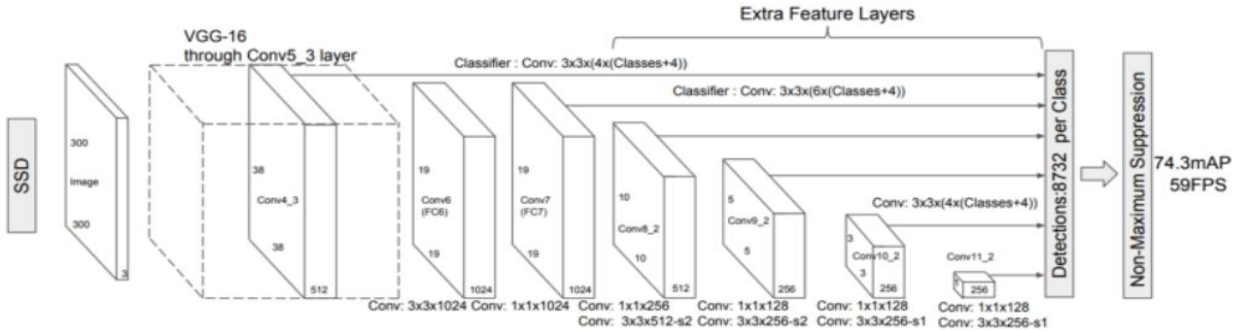


Fig. 1 SSD Architecture

B. Non-Maximum Suppression

A typical Object detection pipeline has one component for generating proposals for classification. Proposals are nothing but the candidate regions for the thing of interest. Most of the approaches employ a sliding window over the feature map and assign foreground/background scores depending on the features computed in this window. The neighbourhood windows have similar scores to some extent and thus are considered as candidate regions. This results in hundreds of proposals. Because the proposal generation method should have a high recall, we keep loose constraints in this stage. However, processing these many proposals throughout the classification network is complex. This results in a technique that filters the proposals based on some criteria called Non-maximum Suppression.

Fig. 2 Before and after Non-Maximum Suppression

1) Firstly, it discards all those cells where the probability of the thing being present (calculated in final softmax layer) is <= 0.6
2) Then the following step is to take the cell with the largest probability among candidates for an object as a prediction
3) Lastly, we discard any remaining cell with Intersection over union value >= 0.5 with the prediction cell.

C. Faster R-CNN

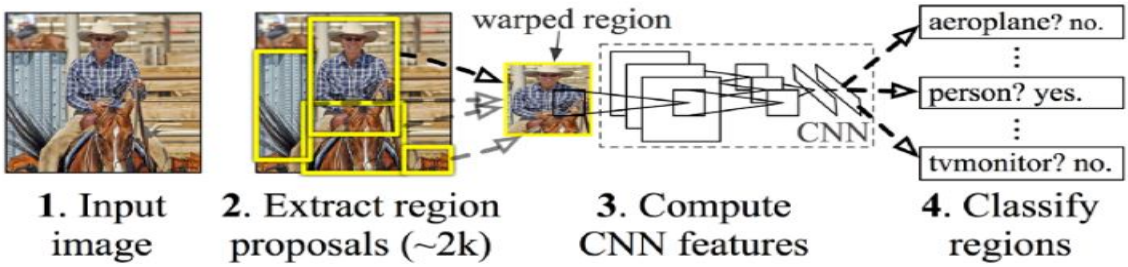


Fig. 3 Faster RCNN Object Detector

Faster R-CNN was developed by researchers at Microsoft. it's based on R-CNN which uses a multi-phased approach to object detection. R-CNN used Selective search to determine region proposals, pushed these through a classification network, then used an SVM to classify the different regions. Faster R-CNN, almost like SSD, is an end-to-end approach. rather than using default bounding boxes, Faster R-CNN includes a Region Proposal Network (RPN) to come up with a fixed set of regions. The RPN uses the convolutional features from the image classification network, enabling almost cost-free region proposals. The RPN is duly implemented as a fully convolutional network and it predicts the objectness scores and objects bounds at each position. Note that the RPN incorporates a similar setup because of the SSD network (i.e. it doesn't predict bounding boxes out of thin air). The RPN network works well with sliding windows over the feature maps. At every sliding-window location or anchor, a set of proposals are computed with different scales and aspect ratios. just like SSD, the result of the RPN is 'adjusted' bounding boxes, based on the anchors. The various components are combined in a single setup and are trained either end-to-end or in multiple phases (to improve stability). An additional interpretation of the RPN is that it guides the network's attention to interesting regions. Most of the usage details of Faster R-CNN are just like those for SSD. In terms of raw mAP, Faster R-CNN typically surpasses SSD, but it requires significantly more computational power. An important section for the Fast-RCNN detector is the 'first_stage_anchor_generator' defines the anchors generated by the RPN. The strides during this section define the steps of the sliding window. Keep a note, especially when attempting to detect small objects (if the stride is too large, we will miss them). Although no extensive data augmentation was employed by the authors of the Faster-RCNN paper, it's still advised to use it when working with smaller datasets.

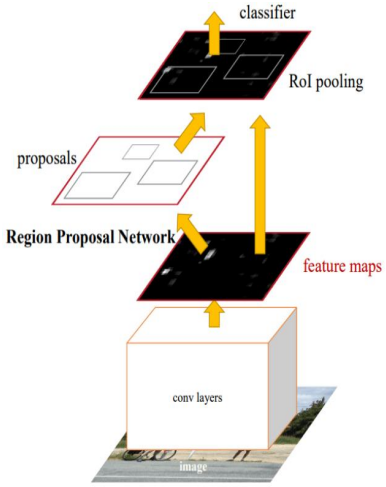


Fig. 4 Network Structure of Faster R-CNN

D. Model Architecture Overview

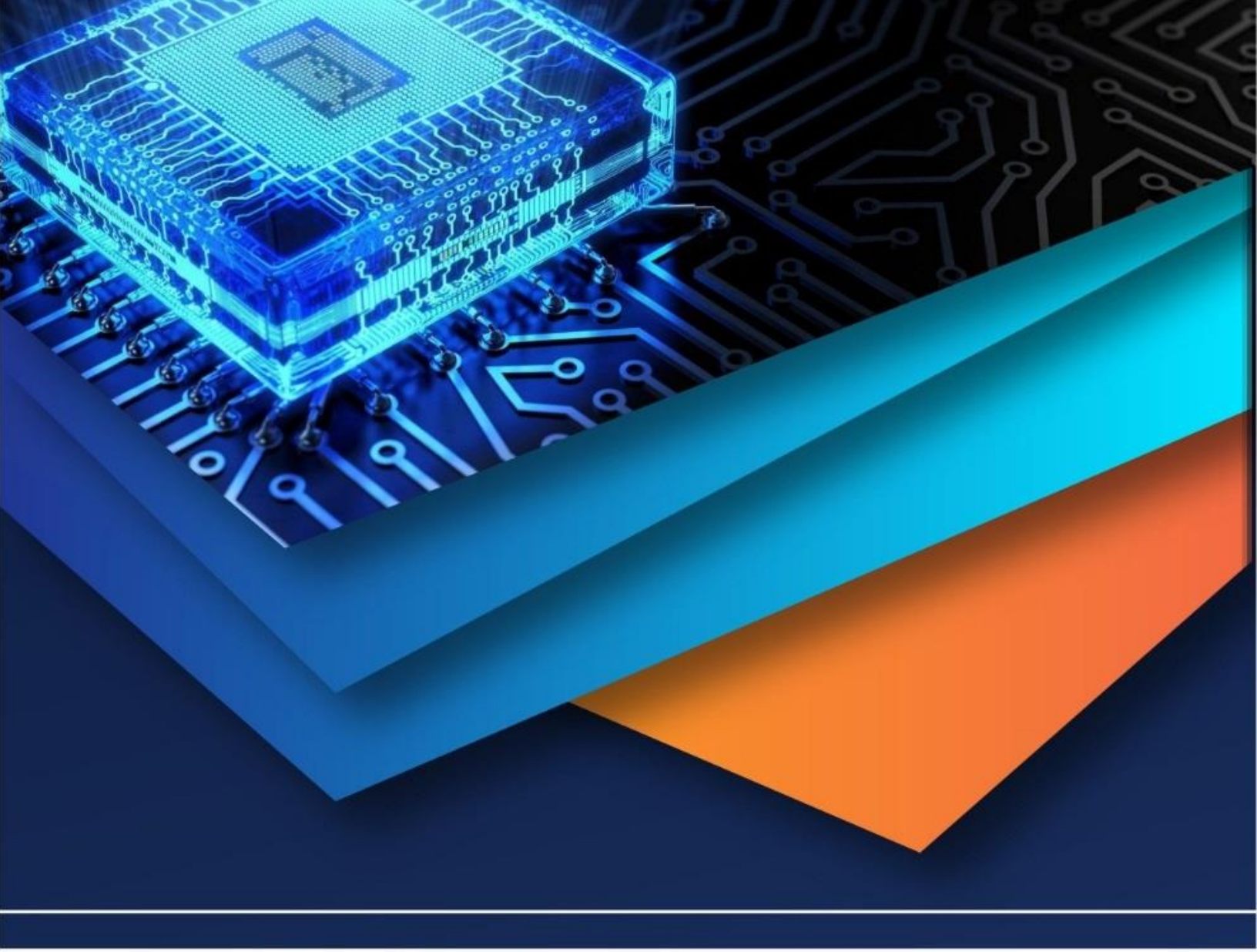
R-CNN, Faster R-CNN, Mask R-CNN Several popular object detection models belong to the R-CNN family. Short for region convolutional neural network, these architectures are supported the region proposal structure discussed above. Over the years, they have become more accurate and more computationally efficient. Mask R-CNN is the latest iteration. It was developed by researchers at Facebook, and it makes a good place to begin for server-side object detection models. YOLO, MobileNet + SSD, SqueezeDet There is a variety of models that belong to the single-shot detector family. The main difference between these variants is their encoders and also the specific configuration of predetermined anchors. MobileNet + SSD models feature a MobileNet-based encoder, SqueezeDet borrows the SqueezeNet encoder, and also the YOLO model features its convolutional architecture. SSDs make an excellent choice for models destined for mobile or embedded devices.

IV. CONCLUSIONS

The Real-time object detection and tracking on video streams is a very crucial topic of surveillance systems in field applications. We implemented SSD with the MobileNet detection tracking method. Algorithms work well for detection and tracking. A high accuracy object detection procedure has been achieved by using the MobileNet and the SSD detector for object detection. The proposed system is tested with many objects and it can detect and identify the objects quite accurately. This system can detect the items within its dataset, such as a car, bicycle, bottle, chair, etc. By using SSD (single Shot Detector), we can detect objects in the fastest manner. It helps to identify individual objects in the image in the fastest way.However, SSD is also the fastest ObjectDetection Model in detecting smaller objects with accuracy whereas other models such as RCNN are not able to detect smaller objects accurately. Finally, we can perform Real-Time Object Detection using SSD with MobilNet v1 which is faster and more efficient than traditional methods of object detection. The objective of our paper is to develop an Object Recognition system to identify the 2-D and 3-D objects in the image. Moreover, the Object Detection systems can be further improved by increasing the global or local features so that the efficiency of these systems can be increased. The goal of this research is to develop an autonomous system where the recognition of objects and scenes helps the community to make the system interactive and attractive. For future work, this work will be primarily deployed to identify the item with better features in the external environment.

REFERENCES

[1] Ross Girshick, Je Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014
[2] Ross Girshick. Fast R-CNN. In International Conference on Computer Vision (ICCV), 2015.
[3] Shaoqing Ren, 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 (NIPS), 2015.
[4] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Uni ed, real-time object detection. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
[5] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In ECCV, 2016.
[6] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY