International Research Journal of Computer Science (IRJCS) ISSN: 2393-9842
Issue 08, Volume 09 (August 2022) https://www.irjcs.com/archives

IMAGE CHARACTER RECOGNITION USING CONVOLUTIONAL NEURAL NETWORKS

Prithvi Raj R
Dept. of ISE
Vemana Institute of Technology
Bengaluru, India
Rohith M
Dept. of ISE
Vemana Institute of Technology
Bengaluru, India
Ravichandra A R
Dept. of ISE
Vemana Institute of Technology
Bengaluru, India
Shafien Ulla Khan
Dept. of ISE
Vemana Institute of Technology
Bengaluru, India

Publication History
Manuscript Reference No: IRJCS/RS/Vol.09/Issue08/SPAUCS10109
Research Article | Open Access
Peer-review: Double-blind Peer-reviewed
Article ID: IRJCS/RS/Vol.09/Issue08/SPAUCS10109
Volume 2022 | Article ID SPAUCS10109 http://www.irjcs.com/volumes/Vol09/iss-08/29.SPAUCS10109.pdf
Received: 25, July 2022 | Accepted: 04, August 2022 | Published: 12, August 2022
doi: https://doi.org/10.26562/irjcs.2022.v0908.29
Citation: Prithvi,Rohith,Ravichandra,Shafien(2022).Image Character Recognition using Convolutional Neural Networks. International Research Journal of Computer Science (IRJCS), Volume IX, Issue VIII, 2022, Pgs 304-311
BibTeX key:
Academic Editor-Chief: Dr.A.Arul Lawrence Selvakumar

Abstract: Object Recognition is a challenging and exciting task of computer Vision. Object recognition is to describe a collection of related computer vision tasks which involves identifying objects in images. Image classification involves predicting the class of objects in an image. Object localization refers to identifying the location of objects in an image. Image Annotation in Machine Learning is the process of creating bounding boxes around the localized images. Now with the advance of deep learning and neural networks, we can finally tackle these problems without coming up with various heuristics real time. By now, there are many different algorithms and concepts for object detection published. Just a few are lightweight enough to run them on live interface that results in an appropriate frame rate. Object detection networks are for example the You Look Only Once (in short: YOLO) network or the Faster R- CNN network. In this project our aim was to develop a software application that takes an image and produce the Informative links about the objects present in the image. Our aim in this project is to develop a software product that runs the SSD to identify objects in an image and the identified images are searched in Google API to give appropriate informative links of those objects
Keywords: Deep Learning, Visual Geometric Group, Convolution Neural Network

I. INTRODUCTION

Vision and perception come naturally to us humans. We never think how hard visual intelligence can be. It was up until the mid-20th century that vision and perception was not considered to be an intelligent task. Vision and perception were considered an easy approach as compared to reasoning and planning. Back then artificial intelligence was mostly a theory. With further advancements in artificial intelligence, when attempts were made, in the mid- 20th century to build AI with reasoning and planning and AI with vision and perception, it was realized that vision and perception can be often harder than reasoning (may be not planning!). We have a very limited understanding of human/animal visual perception abilities.

It is so, we believe artificial visual intelligence is very hard. A more in depth understanding of the human/visual perception system may help us build more efficient systems with much more ease. ANN was an approach in this direction. Invented by Geoffrey Hinton it was built to loosely mimic the human brain. ANN was a very successful approach in artificial intelligence. With back propagation (learning, error and correction method), ANN produced very impressive results in every aspect of artificial intelligence. However, it was not considered a correct path to artificial general intelligence. The primary reason was hardware limitations. In the mid-2000s the problem of hardware was overcome with very powerful GPUs. Since then we have seen the fastest growth in the field of AI. CNN was invented by the student of Geoffrey Hinton, YannLecun. But, can artificial general intelligence be solved with neural networks? Maybe, but we still do not understand a lot about intelligence and intelligent models, so maybe not. It is very difficult to answer this question precisely. So why this approach of neural networks? Because this produces results. We have seen results with CNN that were not possible to imagine a decade ago. Facial recognition, image classification, Automatic Image Annotation a single model to classify and label up to 1000, synthetic data generation with GANs. As of now we believe this a direction of study which needs to be extensively Although YOLO does seem to be the best algorithm to use if you have an object detection problem to solve, it comes with several limitations. YOLO struggles to detect and segregate small objects in images that appear in groups, as each grid is constrained to detect only a single object. Small objects that naturally come in groups, such as a line of ants, are therefore hard for YOLO to detect and localize.

II. SCOPE

The information age has witnessed the rapid development of wireless network technology, which has attracted the attention of researchers and practitioners due to its unique characteristics such as flexible structure and efficiency. As wireless network technology continues to evolve, it has brought great convenience to people's life and work with its powerful technical capabilities. Wireless networks have gradually facilitated the main stream of people's online life. At the same time, the advent of 5G network will further enable the greater development and more advanced applications of wireless network technology. The future generations of wireless networks will provide strong support for related applications such as Internet of Things (IoT) and virtual reality (VR). Many of these applications connect to each other and transmit information within networks based on the detection of specific target objects. In order to achieve a comprehensive network connection between people and people, things and people, and things and things, one of the key tasks of future applications is to identify the target in a real-time manner in the wireless networks.

III. CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Network (CNN) was used to achieve some breakthrough results and win well-known contests. The application of convolutional layers consists in convolving a signal or an image with kernels to obtain feature maps. So, a unit in a feature map is connected to the previous layer through the weights of the kernels. The weights of the kernels are adapted during the training phase by back propagation, in order to enhance certain characteristics of the input. Since the kernels are shared among all units of the same feature maps, convolutional layers have fewer weights to train than dense FC layers, making CNN easier to train and less prone to overfitting. Moreover, since the same kernel is convolved over the entire image, the same feature is detected independently of the locating translation invariance. By using kernels, information of the neighborhood is taken into account, which is a useful source of context information. Usually, a non-linear activation function is applied on the output of each neural unit. If we stack several convolutional layers, the extracted features become more abstract with the increasing depth. The first layers enhance features such as edges, which are aggregated in the following layers as motifs, parts, or objects.
The following concepts steps are important in the context of CNN:

1) Initialization: It is important to achieve convergence. We use the Xavier initialization. With this, the activations and the gradients are maintained in controlled levels; otherwise back-propagated gradients could vanish or explode.

2) Activation Function: It is responsible for non-linearly transforming the data. Rectifier linear units (ReLU), defined asf(x)= max(0, x),were found to achieve better results than the more classical sigmoid, or hyperbolic tangent functions, and speed up training. However, imposing a constant 0 can impair the gradient flowing and consequent adjustment of the weights. We cope with these limitations using a variant called leaky rectifier linear unit (LReLU) that introduces a small slope on the negative part of the function. This function is defined asf(x)= max(0, $x)+\alpha\min(0,x)$where is the leakiness parameter. In the last FC layer, we use softmax.

3) Pooling: It combines spatially nearby features in the feature maps. This combination of possibly redundant features makes the representation more compact and invariant to small image changes, such as insignificant details; it also decreases the computational load of the next stages. To join features it is more common to use max-pooling or average-pooling.

4) Regularization: It is used to reduce over fitting. We use Dropout in the FC layers. In each training step, it removes nodes from the network with probability.

In this way, it forces all nodes of the FC layers to learn better representations of the data, preventing nodes from co-adapting to each other. At test time, all nodes are used. Dropout can be seen as an ensemble of different networks and a form of bagging, since each network is trained with a portion of the training data.

5) Data Augmentation: It can be used to increase the size of training sets and reduce over fitting. Since the class of the patch is obtained by the central voxel, we restricted the data augmentation to rotating operations.

NETWORK ARCHITECTURE:

Image-Input Layer:

An image Input Layer is the place you initialize the size of input image, here, 128-by-128-by-1 is used. These numbers represent height, width, and the number of channels. In this case, input data is a grayscale image, hence the number of channel is 1.

Convolutional Layer:

Input arguments for this layer are filtering size, the number of filters, and padding. Here, the filter of size 10 is used, which determines 10 x 10 filter. The number of channels used is 10, means 10 neurons are connected. Padding of 1 specifies that the size of the output image is same as that of an input image.

ReLU Layer :

ReLU (rectified linear unit) layer is a batch normalization layer, which is placed after initializing a nonlinear activation function. Importance of this layer is to decrease the sensitivity and increase the pace of the training.

Max Pooling Layer:

Max pooling layer is one of the down sampling technique which is used for convolutional layers. In this architecture, poolSize is set to 3 and training function's step size is 3.

Fully Connected Layer:

Fully connected layers follow max pooling layer. In this layer, all the neurons of all layers are interconnected to the previous layer. The given input argument for this layer is 10, which indicate 10 classes.

Softmax Layer:

Fully connected layers are followed by softmax layer, which is normalization technique. This layer generates positive numbers as output such that the sum of numbers is one. Classification layer uses these numbers for classification.

Classification Layer:

Classification layer is the final layer of the architecture. This layer classifies the classes based on probabilities obtained from softmax layer and also calculate cost function.

Training Options:

The maximum number of epochs set to 100 and initial learning rate is 0.001.

Architecture (AlexNet):

This architecture was one of the first deep networks to push ImageNet Classification accuracy by a significant stride in comparison to traditional methodologies. It is composed of 5 convolutional layers followed by 3 fully connected layers, as depicted in Figure.

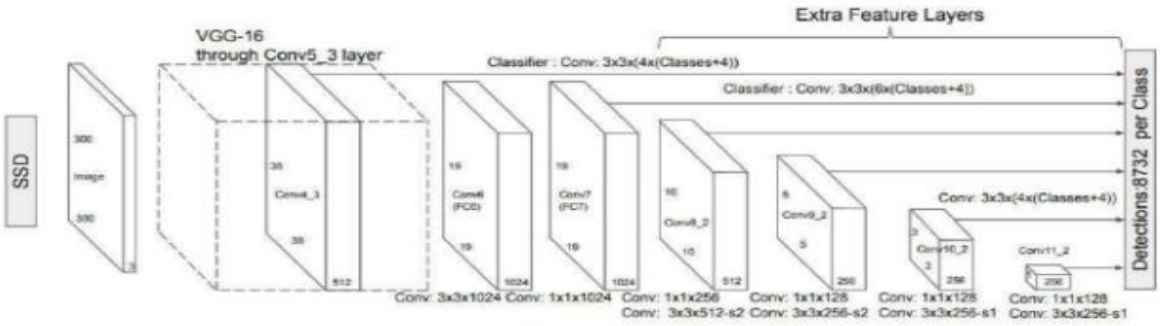

Fig1: SSD Model

SSD is built on the VGG-16 but discards the fully connected layers. The features maps obtained by VGG is used to detect the objects through regression. A set of auxiliary convolutional layers were added, thus enabling to extract features at multiple scales and multiple aspect ratios and progressively decreasing the size of the input to each subsequent layer. Each prediction of SSD comes in boundary box and 81 scores for each class of the object (one extra for background), and we have to filter out based on the object threshold.

Conv4_3 is the filter that is used to apply on the feature maps and it makes a total of 38*38*4 predictions. SSD uses multi scale feature maps to detect the items independently. The CNN has reduced the spatial dimension gradually; the resolution of the feature maps also gets decreased. To detect the larger items SSD uses lower resolution layers. Also, SSD adds more supplementary convolutional layers after the VGG16. Five of them will be used to detect the objects. Each image is divided into multiple cells and the SSD applies the convolutional filters for each cell. Each filter produces 85 channels: 81 classes and one boundary box

IV. PROPOSED METHODOLOGY

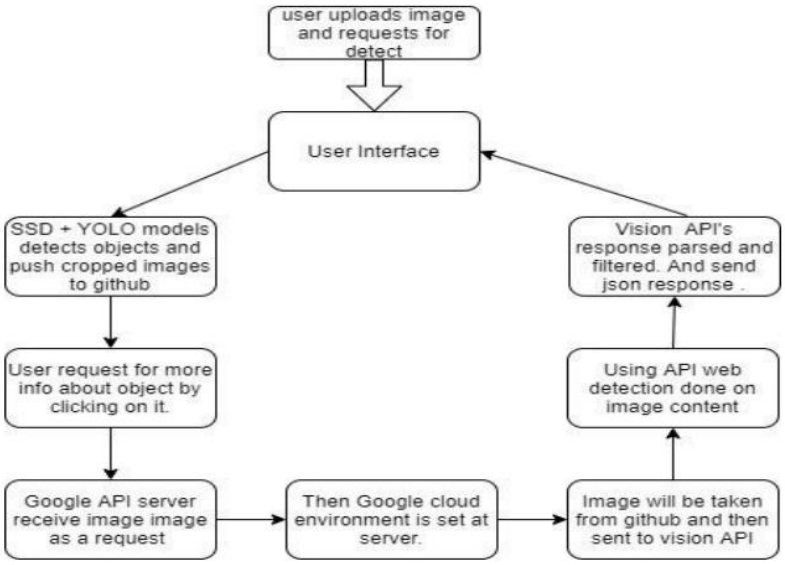

Figure 4.1: Data Flow Diagram

Fig 2: Data flow diagram

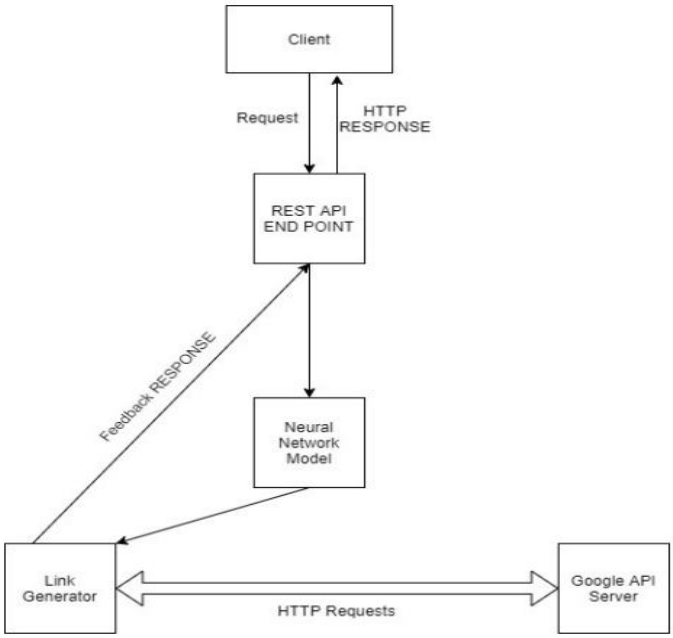

Figure 4.2:System architecture

Fig 3: System architecture

Training and validating system should be fast with all good resources. Network should identify as many objects as possible from the given image. Input image should be clear minimum resolution of 150*150 pixels. Small objects in an object are difficult to identify and objects in image which are not in the dataset are not identified There should not be dependency between the server and client. A client should be able request server through IP or hostnames The user should be connected to internet to send the requests and get the response. The user should be able to upload an image or capture an image through the inbuilt camera. The first part contains a Client – Server Interface where a user can upload the image and server receives the image for processing. The server has the tasks of cross verifying the image size and format with the required specification. After pre-processing server send the image to SSD model. The second part of the design has the convolutional neural network model SSD. SSD model after reading the image using OpenCV parses the image through the neural network layers. The model generates the bounding box coordinates and it is sent back to the server. The server after receiving the image takes the bounding box and annotations.

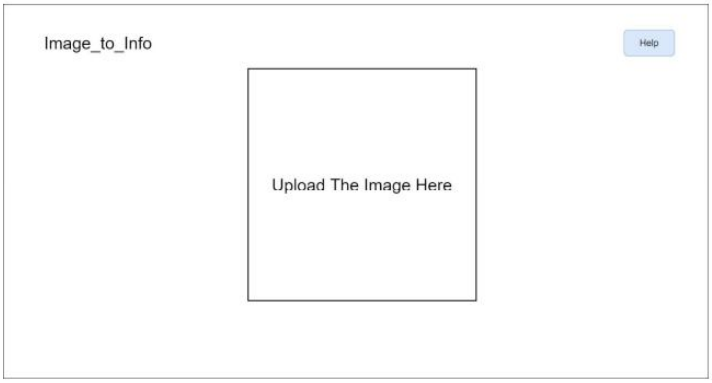

Figure 4.5(a): Client side UI Format in the web browser

Fig 4 : Client side UI Format in the web browser

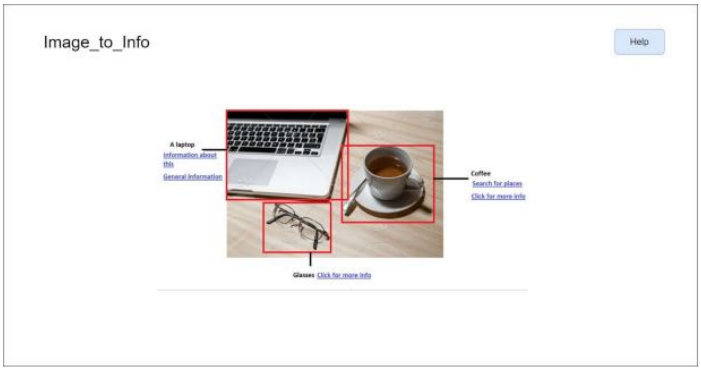

Figure 4.5(b): Client side UI Format in the web browse

Fig 5 : Client side UI Format in the web browser

The current proposed system will be of user friendly UI with accurate results of the objects in an image with the help of google API. The system provides the features which is useful for the users who query the images and understand the components in it The design is intended for real-time image querying application development. The architecture has a front-end web interface with an option to upload or click images through the camera. The application sends the image data to the backend service which is the REST API endpoint which handles the processing of the image i.e., idention.

Request Handling by the Object Detection Server

When we first run the server, we initialize with required dependencies and read the weights of SSD¬ model and YOLO model and starts listening to the incoming requests. Once the request is received by the server, it reads all the bytes sent by the user and decodes it using¬ base64 algorithm.

After converting the server in its folder saves the image into a file¬ A function called run_ssd() is called with the parameter as the file path of the image that was saved¬ by the main function The function run_ssd() will read the image and passes the image through the SSD neural network¬ which we trained and gets the detection. Based on the threshold we filter the results which are higher than the required threshold and append those to a json and send back the response to the main function in the dictionary format. The main function will now call the function run_yolo() with the parameter as the original image¬path. The run_yolo() function also passes the image into a pretrained neural network which was trained¬ on COCO 2014 dataset. The results obtained from yolo model is filtered based on the threshold and are sent to the main¬ function in the dictionary format. The results of both the model are passed to a function called crop_images() .¬ The crop_images will receive the detections of both the model. Based on the number of objects¬ detected in each class we take the maximum detections from one of the models and append it to the output JSON. Example: If yolo detected 5 persons 1 bottle and SSD detected 4 persons, 1 bottle and 1 cat, the final combined result would be like 5 persons of yolo detection,1 bottle of SSD detection, 1 cat of SSD Detection. After appending all the detections to output JSON, we parse it to crop the original image into the¬ images and save the file with the class name followed by timestamp. Once all the images have been saved, we call the push_github () function.¬ The push_github () function will add all the newly cropped images and commits to the git¬ repository which we created to push the images.

V. RESULTS

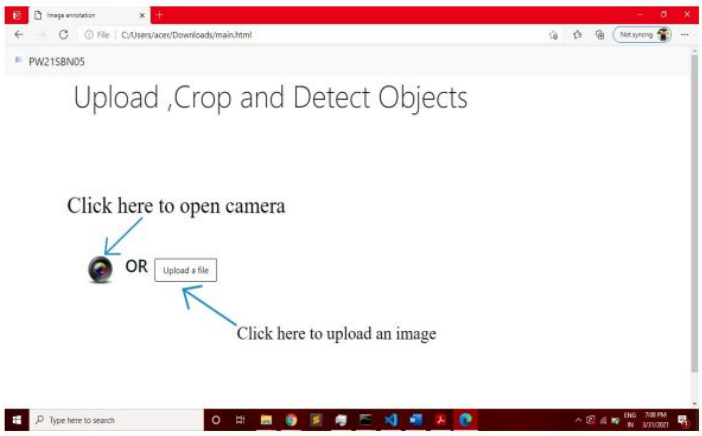

Figure 6.1: User Guide

Fig 6: User Guide

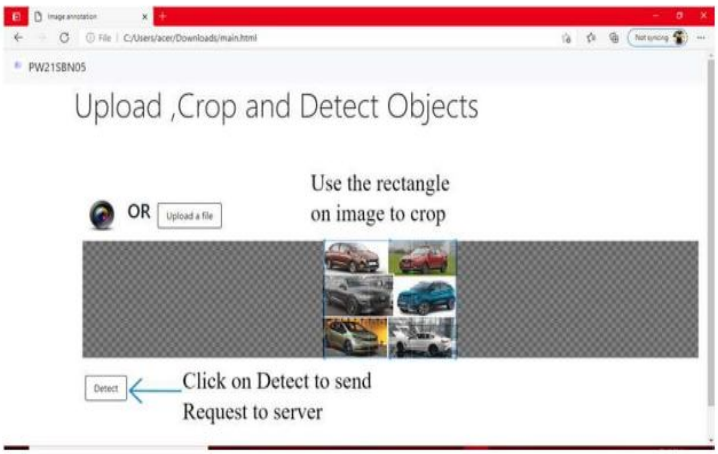

Figure 6.2: Detecting object

Fig 7: Detecting Object

The user upon loading the web pages shown this interface where there are two choices: Click photo using camera Upload an image Click photo through the camera will prompts the user to allow the permission to access the web camera attached to the device. While in smart phones upload an image will also asks for the photo to be taken via the camera. The access to camera will generate the display of camera feed in are tangular div upon which the button take snapshot will be visible through which we can take photo. After uploading an image or taking a photo through the camera user can crop the image based on his interest. The cropped image is saved internally in our canvas. The click on the button called 'DETECT' will hide all the elements and popsup a loading animation which was created. In the background the listener on the button Detect will send the POST request to the user by reading all the bytes in the image cropped and sends to the user in a base 64 encoded format through file upload request in the ajax.

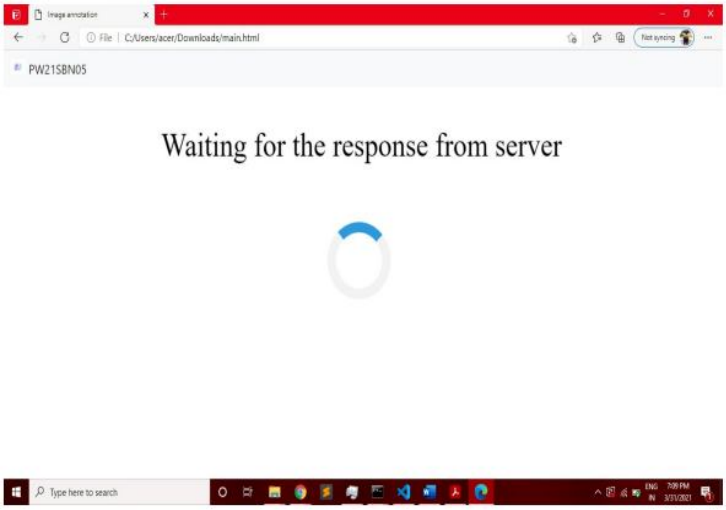

Fig 8: Waiting for Response from server

VI. CONCLUSION

In the project, we have trained the SSD model on PASCAL VOC dataset. Our SSD model was built on pytorch library of python. Starting off with the Literature survey, followed by dataset collection and model training. In the second phase of our project, we have trained our model on PASCAL VOC 2012 Dataset which had 20 classes. In the second phase of the capstone, we had trained our model on 80 different classes of objects through COCO 2019 dataset which has 328,000 images. We have implemented front-end User interface in the form of a HTML page with the required javascript code which has the functionality to communicate with the server and render the results. We have completed the project as per the requirements and specifications provided. We have built the full stack application which can be deployed easily through simple steps and low-cost in-house hardware.

VII. FUTURE WORK

We have to evaluated our model on the validation set of COCO 2019 and got the expected accuracy of our model. With AP50 of 0.65 the SSD model is the best-in-class object detection model that has been developed so far. It beats all the existing object detection models with its accuracy and speed. Our goal was to create two python servers one for object detection and another for the information retrieval from google and Wikipedia. Two python flask servers are REST API based servers which handles our requests and sends the appropriate response. We achieved all the specifications and the accuracy that the model should satisfy in a given environment.

REFERENCE

1. Karen Simoyan, Andrew Zisserman "Very Deep Convolutional Neural Networksfor Large Scale Image Recognition" 2015
2. ShaoquinRen, Kaiming He, Ross Girshick, Jian Sun "Faster R-CNN Towards Real time object Detection" 2015
3. Joseph Redmon, Santosh Divvalay, Ross Girshick, Ali Farshadi "You Only Look Once: Unified, Real-Time Object Detection" 2016
4. Wei Liu, DragomirAngelov, DumitruErhan, Christian Szegedy, Scott Reed, Cheng Yang Fu, Alexander C Berg "SSD :Single Shot MultiBox Detector" 2016

5. Yong Lui, Ruiping Wang "Structural Inference Net: Object Detection Using Scene Level Context and Instance-level Relationship" 2018
6. Vincent Dumoulin, Francesco Visin "A guide to convolution arithmetic for deep learning" 2018
7. KaranbirChahal, KuntalDey "A Survery of Modern Object Detection Literature using Deep Learning
8. Bell, S., Zitnick, C.L., Bala, K., Girshick, R.: Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: CVPR. (2016)
9. Tsung-Yi Lin, Michael Maire, Serge Belongie, LubomirBourdev, Ross Girshick, James Hays PietroPerona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollar "Microsoft COCO: Common Objects in Context"
10. Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan "PyTorch : An Imperative Style, High Performance Deep Learning Library"
11. Mark Everingham, Luc Van Gool, Christopher K.I. Williams, John Winn, Andrew Zisserman "The PASCAL Visual Object Classes (VOC) Challenge

