Single Shot Detector Heroku Project technical report

Jiacheng Shi

Department of Electrical and Computer Engineering

University of Michigan

jiachs@umich.edu

1.Introduction:

With the rapid development of deep learning and computer vision, the field of object detection is getting more and more attention, they are widely used in, for example, face recognition, medical image processing, security surveillance and so on. So I would like to design a website that can let novices quickly see the detection results of object detection models and give them a preliminary understanding of object detection algorithms. The detection network uses the one-stage Single Shot Detector[1] network, which has the advantages of faster training and inference and higher accuracy compared to the two-stage target detection network, like Faster-RCNN[2]. This model is trained on COCO dataset[3] based on the Pytorch framework and is eventually using AWS to finish the inference part and use Heroku to deploy.

2.Data:

MS COCO, whose full name is Microsoft Common Objects in Context, originated from the Microsoft COCO dataset, which Microsoft funded to annotate in 2014, and is considered one of the most popular and prestigious competitions in computer vision, along with the ImageNet competition. The images include 91 classes of targets, 328,000 images and 2,500,000 labels, and it is the largest dataset with semantic segmentation, it provides 80 classes, over 330,000 images, 200,000 of which are labeled, and over 1.5 million images in the entire dataset. The reason I use the COCO dataset as a training set is because it is a comprehensive and generalizable dataset that can make our model more robust. In this project, I first downloaded the pytorch pre-trained model of SSD[4] on the COCO dataset and then transformed it into the onnx model.

3. Key of Architecture in SSD:

3.1 Use of multi-scale feature maps for detection:

Multi-scale uses feature maps of different sizes. CNN networks generally have a relatively large feature map in the front and gradually use stride=2 convolution or pool to reduce the feature map size later, which is as shown in Figure 1, a relatively large feature map and a relatively small feature map, which are both used for detection. The advantage of this is that the larger feature map is used to detect relatively small targets, while the smaller feature map is responsible for detecting large targets, as shown in Figure 2, the 8x8 feature map can be divided into more units, but the prior frame scale of each of its units is relatively small.

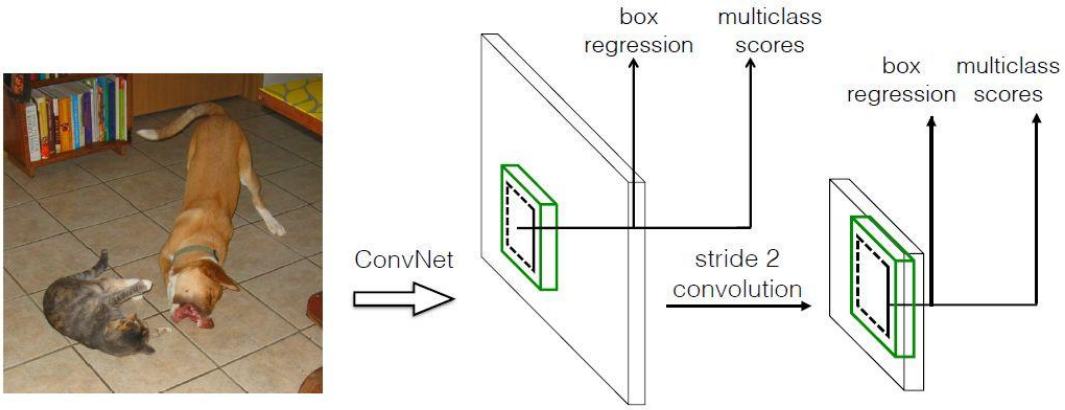

Figure 1 : multi-scale feature maps prediction

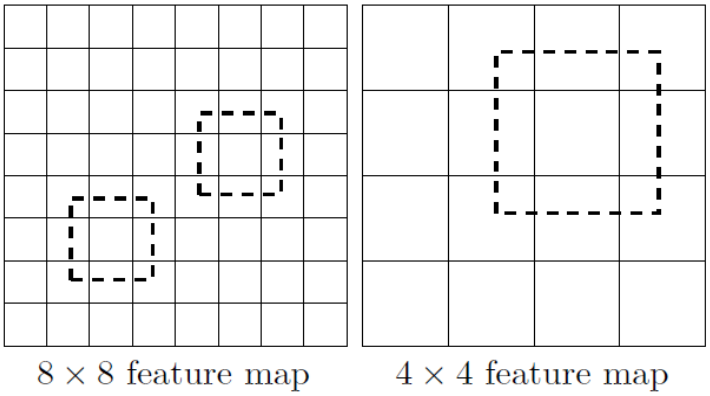

Figure 2: receipt field with different size

3.2 Convolutional predictors for detection

Unlike Yolo which finally uses a fully connected layer in Figure3 , SSD directly uses convolution for different feature maps to extract the detection results. For feature maps of shape (M,N,P) feature map, only a relatively small convolution kernel like (3,3,P) is needed to obtain the detection value.

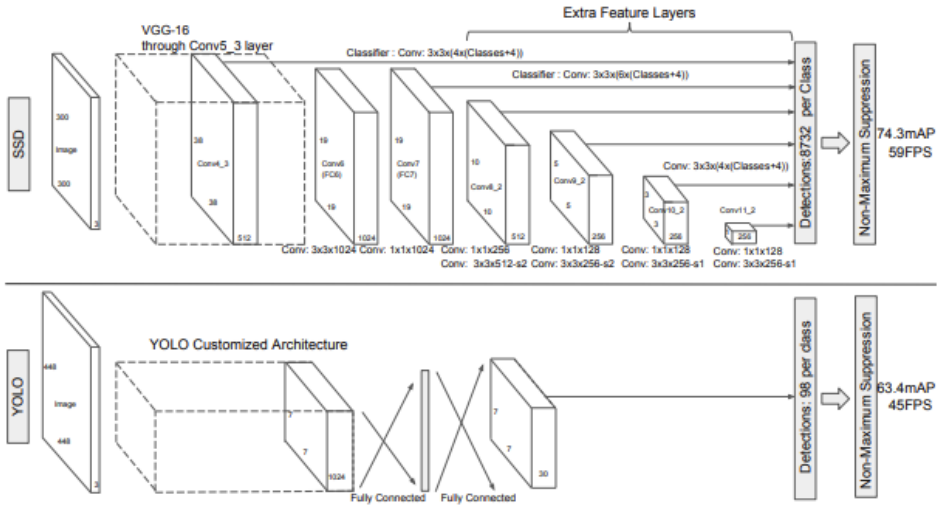

Figure 3: A comparison between SSD and YOLO models

4. Pipeline in Cloud

4.1 S3 bucket in AWS:

I put the onnx ssd model into a zip file, along with the corresponding environment packages and library. Moreover, I put the demo images into another s3 bucket so that we could show the demo picture by the drop down menu.

4.2 Lambda function and API gateway in AWS:

The first lambda function I create is the ssd-inference function that is responsible for the inference stage of the ssd model and finishing the class and boundbox prediction. I also created an API gateway with POST method to send the image to the inference function to detect the image, and I will send the image back to the front-end after interencing it.
The second lambda function I create is to interact with the drop-down menu, and also I create two corresponding API gateway which has a POST and GET method to get the list of demo files from s3 bucket and down the specific image from it. Like Figure 4.

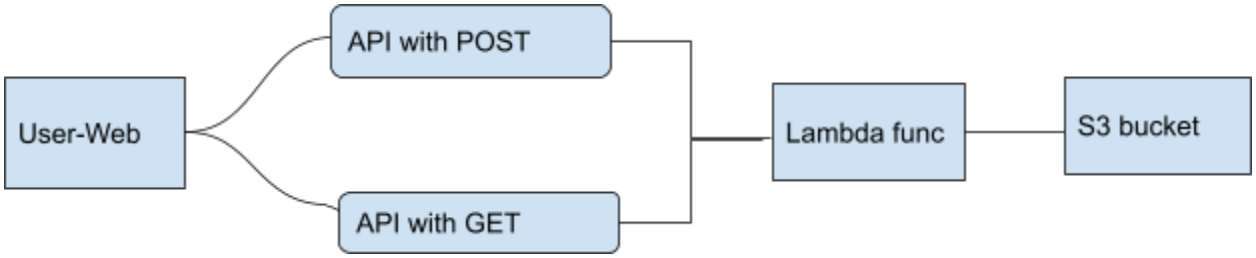

Figure 4: the pipeline of lambda function interact with dropdown menu and API

4.3 Image working well

In most simple scenes such as large objects and a single number of objects, SSD has a relatively good detection effect and high confidence, I manually tested 100 images, and the probability of fully detecting all objects in a single image reached 91%.

Original Image

Detect Result

Original Image

Detect Result

4.4 Image not working well

In more complex scenes, such as pictures of many people walking on the street, it is more difficult for the model to detect small-sized pedestrians in the distance. The problem may arise in the feature pyramid, where the size of feature map for predicting small objects is limited, resulting in an insufficient number of prediction bounding box in the original image, which makes it difficult to cover all small objects.

Detect Result

My website link:https://jiacheng-ssd-app.herokuapp.com

Citation:

[1]: Liu, Wei, et al. "Ssd: Single shot multibox detector." *European conference on computer vision*. Springer, Cham, 2016.
[2]:Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems* 28 (2015).
[3]:Lin, Tsung-Yi, et al. "Microsoft coco: Common objects in context." *European conference on computer vision*. Springer, Cham, 2014.
[4]: https://github.com/amdegroot/ssd.pytorch