Object Detection In Deep Learning

Gaurav Pande
gpande@gatech.edu

Jin Xu
jin.xu@gatech.edu

Tu Xu
xutu@gatech.edu

Abstract

Object detection is a subset in computer vision which deals with automatic methods for identifying objects of interests in an image with respect to the background. In this project we experiment with the existing state of the art object detection algorithms using Deep Learning. Specifically, we implemented and experimented with three models namely Faster-RCNN, Mask-RCNN, YOLOv3(You Only Look Once) and SSD(Single Shot Detection).

1. Introduction

Object detection plays a important role in Computer Vision and Deep Learning. There are two standard approaches for object detection which leads to different sets of algorithms.

- The first category is to deal with region proposal first. This means region which are highly like to contain objects are selected with either traditional CV techniques like selective search or using deep learning techniques such as region proposal network. This category includes algorithms like R-CNN, Fast R-CNN.

- The second category of algorithms looks for object at fixed location with fixed sizes. These location and sizes are strategically selected to cover the object entirely. The original image in these algorithms are separated using fixed size grid regions. For each region these algorithm tries to predict a fixed number of objects of certain, predetermined shape. These are single stage algorithms and YOLO, SSD belongs to this category.

1.1. Background

Image classification and object detection are the most common tasks in computer vision. Image classification takes an RGB image as input and predicts the object in the image with a class and a confidence score. Object detection not only predicts the object in the image but also outputs the location of the object with a bounding box.

Traditional approach: Before the era of deep learning, object detection was done using the sliding window approach with HoG feature representations [4]. In the sliding window approach, a window patch moves around an image and compares it with a HoG template. An SVM is trained to predict whether a current window patch is an object.

R-CNN: With the power of deep learning, convolutional neural networks (CNNs) can achieve extremely high accuracy on image classification tasks. However, it is not feasible to combine directly CNNs with the sliding window approach because there are too many locations with various scales and CNNs are slow to run. To solve this, R-CNN [6] utilizes region proposal algorithm – also called Selective Search. This algorithm reduces the number of candidates to about 2000 region proposals. A typical R-CNN network consists of three major steps. In the first step, the Selective Search algorithm generates a number of Region of Interest (RoI). These proposals are then fed into CNNs. An SVM is then used to predict the class of the object. One major issue of R-CNN is its speed. Running CNN on 2000 region proposal is still time-consuming and uses a lot of space.

Fast and Faster R-CNN: Fast RCNN [5] improves upon R-CNN model. In particular, it achieves end-to-end learning by adding a spatial pooling layer after the ConvNet layer. In 2016, Ren et al. proposed Faster RCNN model[11]. In this model, the Selective Search algorithm is replaced with a new selective search algorithm (Region Proposal Networks) using a convolution neural network. In addition, Faster R-CNN introduces the anchor boxes, which are used to handle different scales of objects.

YOLO: Unlike many other object detection system, such as R-CNN, YOLO frames object detection as a regression problem to spatially separated bounding boxes and associated class probabilities.YOLO uses a single neural network which predicts bounding boxes and class probabilities directly from full images in one evaluation. The advantage of using YOLO is that it reasons globally about the image when making predictions. Unlike sliding window and RCNN techniques, YOLO sees the entire image during training and test time so it implicitly encodes the contextual information about classes as well as their appearance in the image.This also makes YOLO the best choice for real time

object detection method.

SSD: SSD [9] a single-shot detector for multiple categories. It uses a single deep neural network and does not require region proposal methods and therefore is faster than R-CNN, Fast and Faster R-CNN while achieves similar accuracy. SSD is significantly more accurate than YOLO. The most important feature of SSD is to use small convolutional filters applied to feature maps, to predict scores for a fixed set of default bounding boxes .

1.2. Motivation

We have studied image classification problem in class. It is almost intuitive for us to further explore the applications of image classification - object detection. Object detection is currently heavily utilized in a lot of AI applications, such as self-driving cars, video surveillance system, and robotics.

For this project, we aim to explore the problem of object detection using Deep Learning techniques. We studied the existing state of the art object detection algorithms. We then compared and constrast ed different models through implementation and experimentation.

2. Approaches

2.1. Faster R-CNN and Mask R-CNN

Faster R-CNN consists of two major parts: a regional proposal network and a detection network. Based upon a convolutional layer, the regional proposal network (RPN) is used to generating candidates. The output of RPN is fed into the detection network which contains a classifier for object detection and a regressor for bounding box. The architecture of a Faster R-CNN model is shown in Fig. 1. With an extra branch on the top of Faster R-CNN, Mask R-CNN [7] can further do image segmentation. This involves two extra layers of convolutional natural network and a box predictor.

Due to the time constraints and limited computational resources, we took a transfer learning approach. In particular, we took a pre-trained Faster R-CNN trained on the COCO dataset. We modified its architecture to match a Mask R-CNN structure. We then retrained the network for using a small dataset and evaluate its performance.

2.2. YOLO

YOLO [10] uses a single neural network to predict bounding boxes and class probabilities from a single image in one evaluation step, so the advantage offered is that it can be optimized end to end directly. Typically we perform the following steps in yolo:

- Separate the original image into grids of equal sizes.
- For each grid, predict a preset number of bounding boxes with predefined shape centered around the grid center. These boxes are also names as anchor boxes.Each prediction can be associated with a class probability indicating which object class that bounding box belongs to, and an object confidence value indicating whether the box contains an image or is it just the background.
- Finally we select those bounding boxes associated with high confidence value and class probability.

In Fig. 2 we can see that if the center of the object's ground truth bounding box falls in a certain grid cell(i.e. the red one on the bird image), this grid cell is responsible for predicting the object's bounding box. The corresponding objectness score is "1" for this grid cell and "0" for others. For each grid cell, it is assigned with 3 prior boxes of different sizes(these boxes are created using K means algorithm). What it learns during training is to choose the right box and calculate precise offset/coordinate. But how does the grid cell know which box to choose? There is a rule that it only chooses the box that overlaps ground truth bounding box most using the Intersection over union value.

2.2.1 Network architecture

YOLOv3 uses 75 convolution layer, with skip connections, and upsampling layers. There is no pooling done in YOLO, and stride of 2 is used to downsample the features maps. This downsampling helps in reducing the loss which is attributed to pooling.Being a fully convolution network the YOLO is invariant to the size of the input. The loss function in YOLO has multiple parts:

- A bounding box coordinate error measured using mean square error.
- Object error which is a confidence score of whether there is an object or not. When there is an object, we want the score equals to IOU, and when there is no object we want to score to be zero. This is also mean square error.
- classification of object which uses cross entropy loss.

2.3. SSD

Single Shot MultiBox Detector (SSD) [9] is a method for object detection using a single deep neural network. This method is featured with multi-scale feature maps and data augmentation.

The architecture of SSD is shown in Fig. 3 The model is developed from a base network which is a feed-forward convolution network, followed by a non-maximum suppression step. Convolution feature layers are added to the end of the truncated base network. The sizes of these layers decrease through the network. Each layer is able to produce a fixed set of object detection predictions.

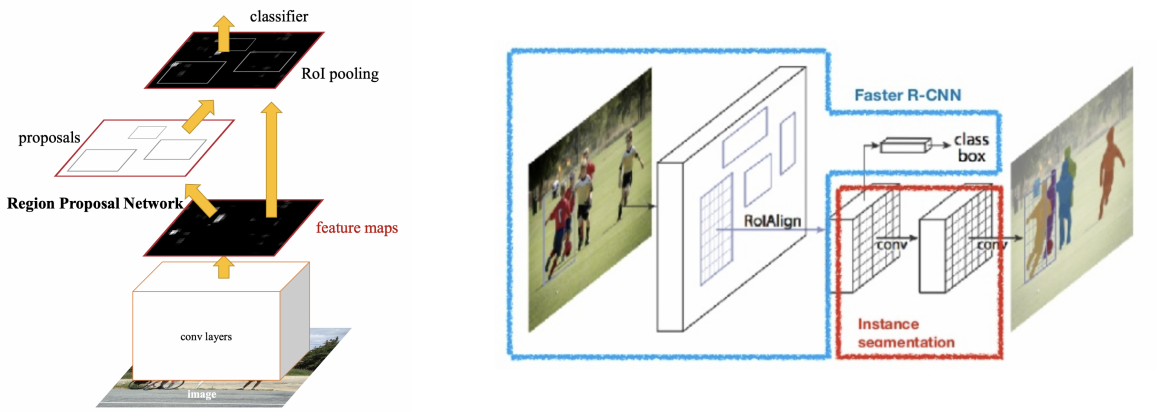

Figure 1. Flow diagram illustrating Faster R-CNN [11] model (left) and Mask R-CNN [7] model(right)

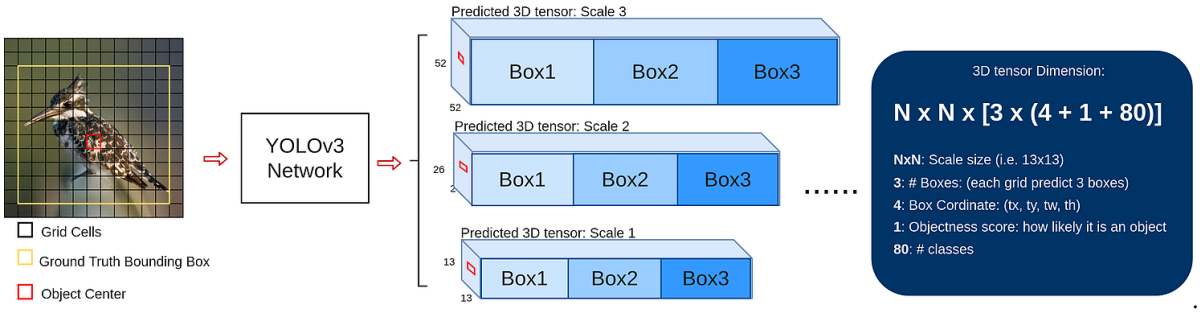

Figure 2. YOLO v3 process flow. [1]

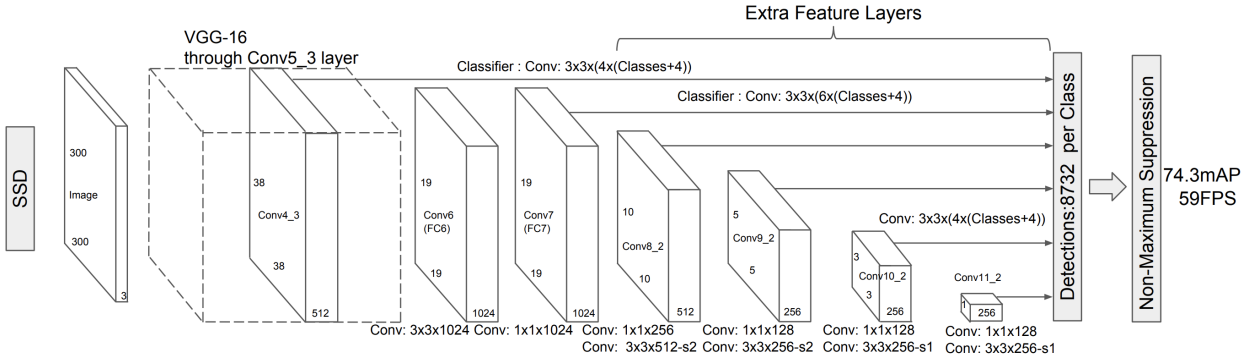

Figure 3. The architecture of the SSD model: several feature layers are added to the end of a base network. The input size of the image is 300×300. [9]

3. Experiments and Results

3.1. Faster R-CNN

We implemented a Faster R-CNN using PyTorch and tested its performance. Due to time constraints and limited computational resources, we did not train the network from scratch. Rather, we utilized a pre-train network on COCO dataset using Resnet50. To show the result, we tested it using three images found online. The Fig. 6 shows the detected objects with corresponding bounding boxes.

3.2. Mask R-CNN

3.2.1 Dataset

We used Penn-Fudan Database for Pedestrian Detection and Segmentation. It contains 170 images in total. Details about this dataset can be found here at https://www.cis.upenn.edu/~jshi/ped_html/ A random sample of 50 were selected for the test set. The rest of the images were used as training data. The training data was augmented using horizontal flip. We did not augment images in the test set.

3.2.2 Training

We used aforementioned pre-trained faster RCNN model and fine tuned it for the Pedestrian detection dataset. The model was trained with 10 epochs, a learning rate of 0.005, a momentum of 0.9, and weight decay of 0.0005. As shown in Fig. 5, training loss drops dramatically after the first epoch and converges towards zero.

3.2.3 Validation

The model was evaluated on the test set. The table below shows detailed results of average precision (AP) and average recall (AR) using COCO IoU metric.

IoU metric: bbox			
IoU	AP	maxDets	AR
0.50:0.95	0.829	1	0.382
0.50	0.990	10	0.871
0.75	0.965	100	0.871
IoU metric: segm			
IoU	AP	maxDets	AR
0.50:0.95	0.768	1	0.351
0.50	0.990	10	0.809
0.75	0.911	100	0.809

3.3. YOLO

3.3.1 Dataset

For training we used Microsoft COCO datase[2]. But since YOLO is incompatible with the coco dataset as it uses the darknet for the backbone network, we used only 2014 coco dataset. The training data was Augumented with horizontal flip only, and no data augmentation is done on the test set. The result can be seen in Fig. 2.

3.3.2 Training

We utilized the Pretrained network on COCO dataset using Darknet. Due to time and compute constraints We did not trained on the full 113000 images of the dataset, rather we used a subset of 35,000 images, with 80 classes of object detection. In Fig. 5, we can see how the loss curve started to fall down with the step size. We trained the model for 50 epochs, with batch size of 8, with momentum=0.9, decay=0.0005, iou-threshold=0.5.

Figure 4. Object detection using Mask R-CNN.

3.3.3 Validation

We used the COCO validation set of 5000 images for validation. The mean average precision we got is 0.58, with detecting all the classes. We can see the results for some of the classes in Table 2.

3.4. SSD

3.4.1 Dataset

COCO [2] is a large dataset used for object detection, segmentation and captioning. In this project, we use COCO 2014 dataset [8] to train the SSD model. The dataset con-

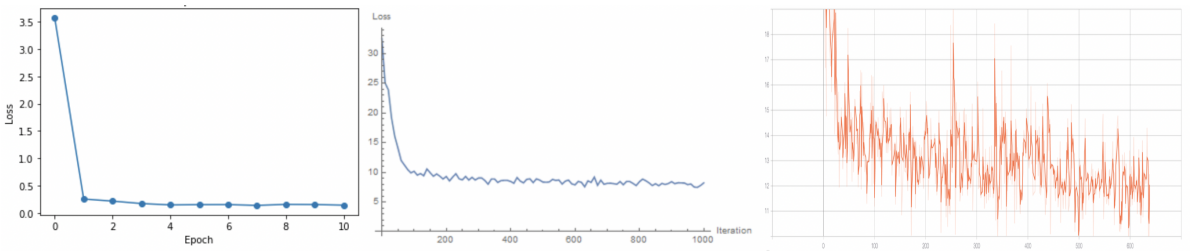

Figure 5. Loss curve for all 3 models(loss against number of iterations) Leftmost Figure is Faster-Rcnn loss curve, middle Figure showing loss for SSD and right most showing loss for YOLO model

Table 1. AP (Average Precision) for different classes of the pretrained Yolo model on COCO 2014 dataset

Class	AP	Class	AP
aeroplane	0.7540	diningtable	0.2885
bicycle	0.5928	dog	0.6976
bird	0.1813	horse	0.8702
boat	0.5299	motorbike	0.7213
bottle	0.4654	person	0.7176
bus	0.9681	pottedplant	0.5069
car	0.8584	sheep	0.7767
cat	0.8568	sofa	0.7894
chair	0.7546	train	0.8623
cow	0.7546	tvmonitor	0.7670

Table 2. AP (Average Precision) for different classes of the pretrained SSD model on VOC 2007 dataset

Class	AP	Class	AP
aeroplane	0.8207	diningtable	0.7906
bicycle	0.8568	dog	0.8566
bird	0.7546	horse	0.8714
boat	0.6952	motorbike	0.8403
bottle	0.5019	person	0.7896
bus	0.8479	pottedplant	0.4724
car	0.6801	sheep	0.5769
cat	0.7504	sofa	0.7500
chair	0.4442	train	0.8523
cow	0.3154	tvmonitor	0.8626

sists of 35,000 images.

3.4.2 Training

The base network weights are based on [12]. We then performed training with a learning rate of 0.001, a momentum of 0.9 and a weight decay of 0.0005. The training loss against iterations is plotted in Fig. 5. The training code is based on PyTorch and is modified on the repo https://github.com/amdegroot/ssd.pytorch.git. Due to the large size of COCO data, we only trained one epoch.

3.4.3 Validation

We use VOC 2007 dataset [3] to perform a precision test of a pretrained SSD model provided by repo https://github.com/amdegroot/ssd.pytorch.git, since we are only able to train one epoch on SSD due to the size of the COCO dataset. The results are shown in Table 2. The mean average precision for the pretrained SSD model is 0.7749.

3.4.4 Examples

Then we use the pretrained SSD model loaded from NVIDIA https://pytorch.org/hub/nvidia_deeplearningexamples_ssd/ to perform object detection on three images randomly selected from internet. The results are shown in Fig. 8.

4. Conclusion

In this project, we study existing object detection models including R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN, YOLO and SSD. For each model, we explore its advantages and disadvantages, as well as its architecture. Specifically, for Mask R-CNN, YOLO and SSD, we successfully perform training on open-source datasets including COCO and Penn-Fudan Database. We also validate the trained model using common metrics. Three randomly selected test images are provided in this project for comparison among the models.

References

[1] Ai: Machine learning: Computer vision: Demos. 3

[2] Common objects in context. 4

[3] The pascal visual object classes. 5

[4] Oscar Déniz, Gloria Bueno, Jesús Salido, and Fernando De la Torre. Face recognition using histograms of oriented gradients. *Pattern Recognition Letters*, 32(12):1598–1603, 2011. 1

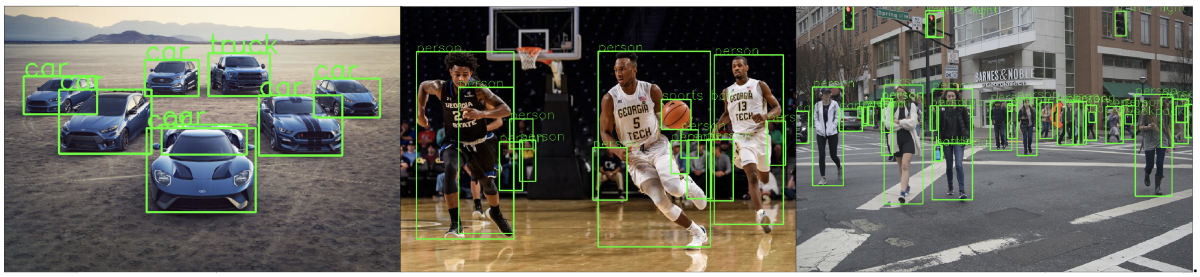

Figure 6. Faster R-CNN object detection examples

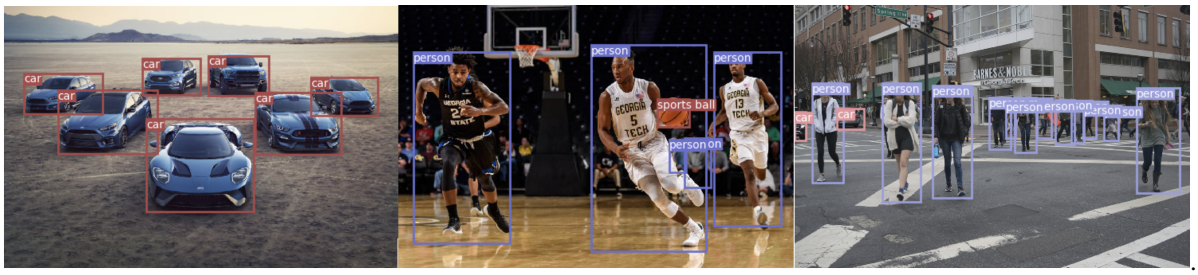

Figure 7. Yolo object detection examples

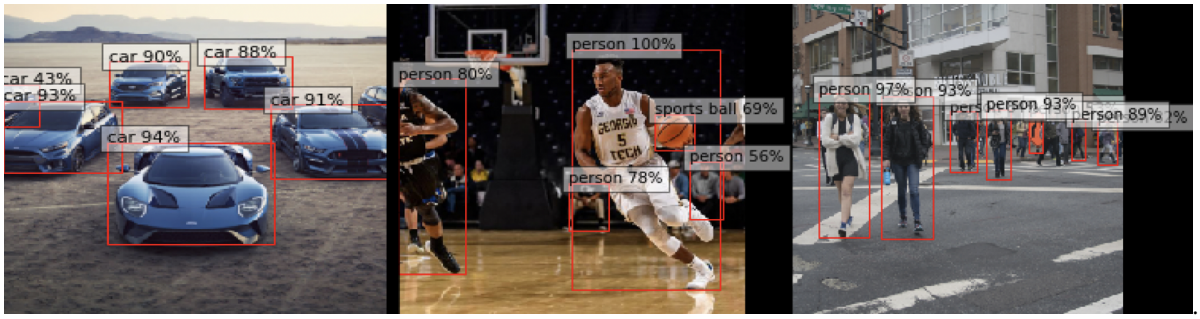

Figure 8. SSD object detection examples

[5] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1

[6] Ross Girshick, 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*, pages 580–587, 2014. 1

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2, 3

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing. 4

[9] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015. 2, 3

[10] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018. 2

[11] 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*, pages 91–99, 2015. 1, 3

[12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5