

Object Detection and Instance Segmentation

Group 12

Muhammad Ibrahim - BSCS19003

Muhammad Sufian Saeed - BSCS19013

Muhammad Muneeb Ur Rahman - BSCS19057

Moin ud Din - BSCS19078

1. Introduction

Object detection is one of the fundamental problems of Computer Vision which is basically based on two tasks. Firstly draw a bounding box on each of the interest objects in the image. Secondly, classify each object into some class. This problem forms the basis of many other computer vision problems like object tracking, instance segmentation, etc.

2. Literature Review

2.1 Slow R-CNN/R-CNN

There has been a lot of work done on this problem in the last decade with different deep learning-based approaches. One of the early models was RCNN[1] which process the image in three separate parts. First, it uses a pre-trained model to propose regions, then it uses Caffe implementation of the CNN described by Krizhevsky to extract a 4096-dimensional feature vector. Finally, it trains class-specific linear SVMs.

However there are a few drawbacks in R-CNN, it trains on a multi-stage pipeline, training is expensive in space and time because with deep networks like VGG16 it takes 2.5 GPU-days for 5k images and hundreds of gigabytes of storage, and Object detection is slow which takes approximately 47s/image on GPU. So, Fast R-CNN [2] improves it.

2.2 Fast R-CNN

Fast R-CNN[2] takes an entire image and a set of object proposals as input. The network first passes the image from several convolution and max-pooling layers to produce a conv feature map. Then region of interest (ROI) pooling layer extracts a feature vector for each object proposal. These feature vectors are passed through a sequence of fully-connected (FC) layers and then uses two output layers to predict results. One for softmax probabilities for K class and one for bounding box (x, y, w, h) .

In Fast R-CNN, we take image and object proposals as input. The overall network is slow because the object proposals are obtained from another network for example selective search[3] which takes almost 2s which is much slower than the further processing. This makes the overall network slow

2.3 Faster R-CNN

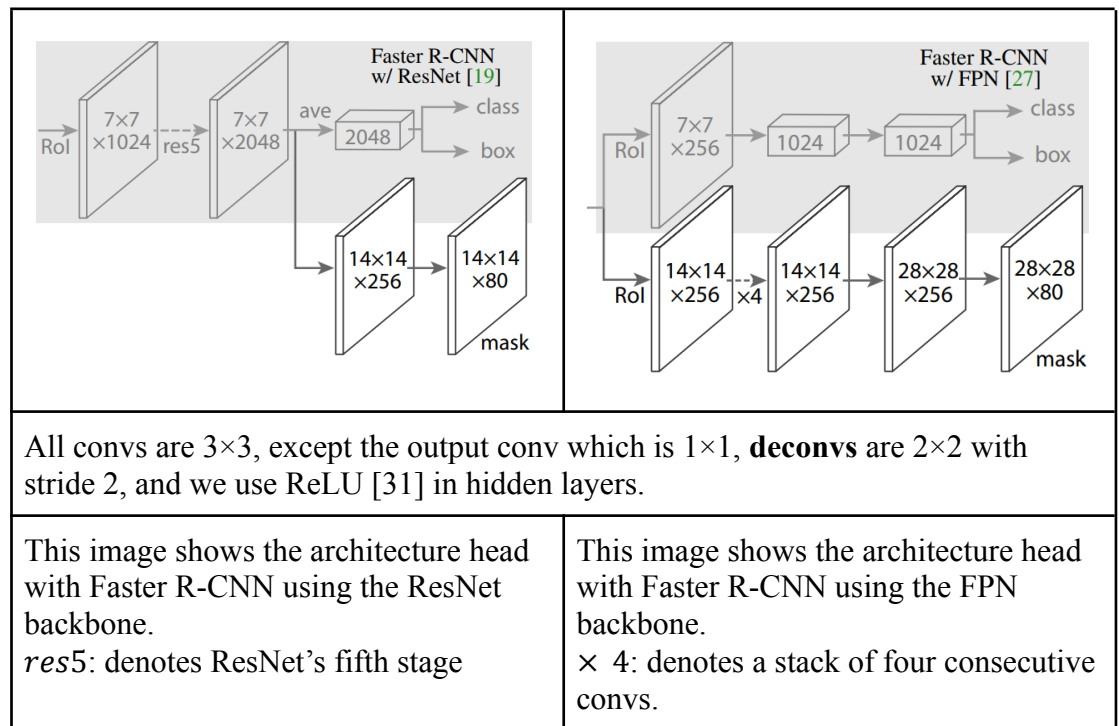
Faster R-CNN[4] proposed a unified network for region proposals and object detection. It introduced the region proposal network (RPN), where a feature map, obtained from convnet, is passed to a small network to output lower dimension features. These features are fed into two sibling fully connected layers - a box-regression layer (*reg*) and a box classifier layer (*cls*).

3. Methodology and Network Details

3.1 Architecture

Mask R-CNN[5] extend the Faster R-CNN[2] by adding a network block to predict the mask of $m \times m$ size.

- a) Backbones
 - i) ResNet50 (ResNet-50-C4)
 - ii) ResNet101
 - iii) Feature Pyramid Network (FPN)
- b) Heads

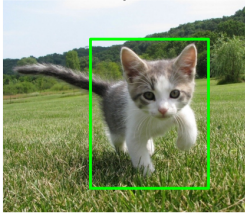
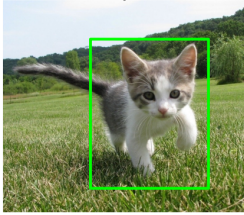
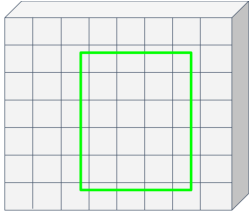
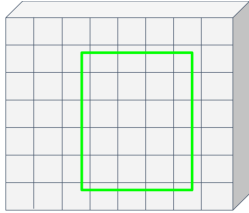
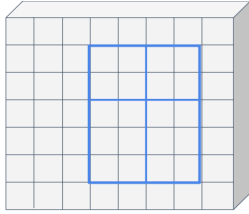
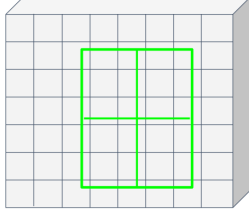
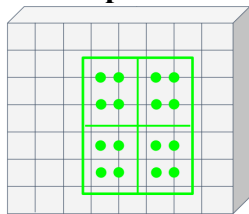
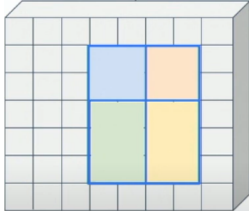
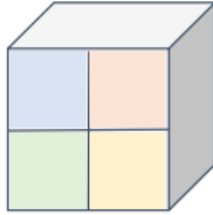
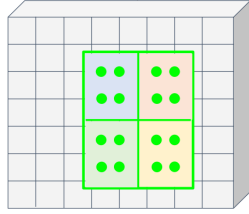
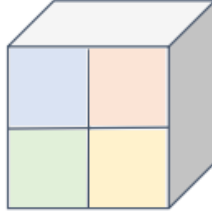


3.2 Mask

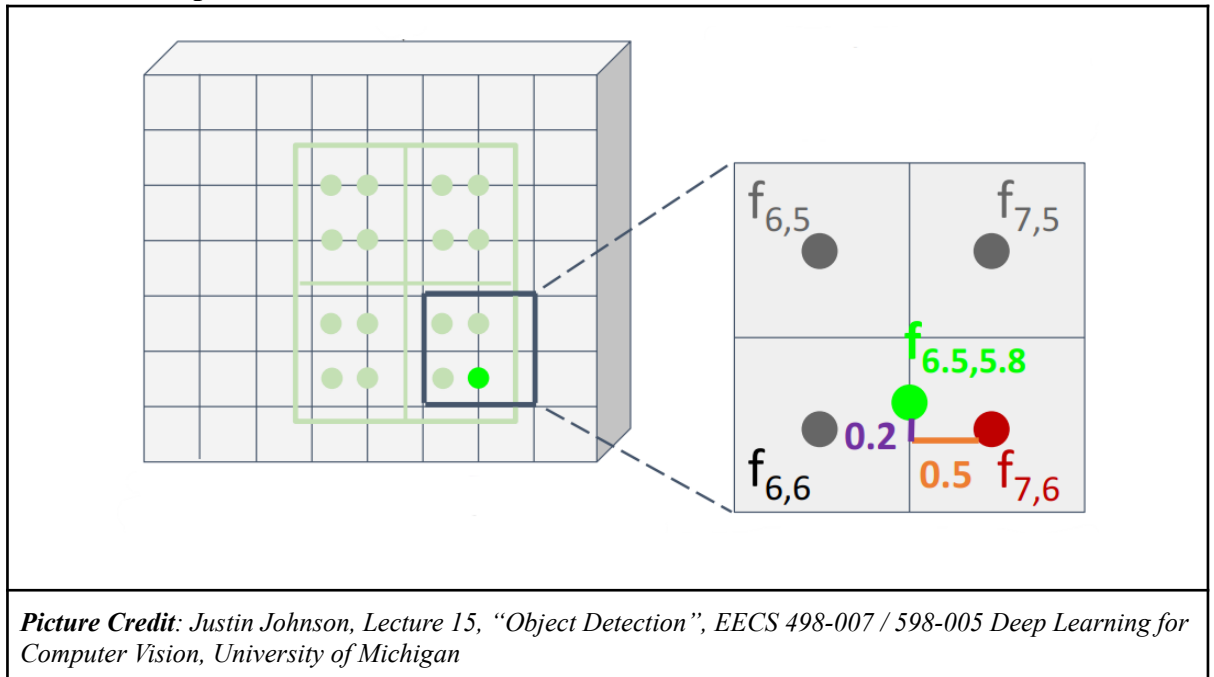
The layer added for mask outputs K masks for each class of size $m \times m$. For each pixel, the sigmoid is applied to classify it as a foreground or background.

3.3 RoIPool vs RoIAlign

For predicting an accurate mask we need to maintain pixel-by-pixel spatial correspondence between the mask and the region of interest. In Faster R-CNN, the RoIPool layer performs quantization on the Image Feature. Due to which spatial correspondence distorts. So, Mask R-CNN replace it with RoIAlign that uses bilinear interpolation from discrete cell to continuous values

	RoIPool	RoIAlign
Input Image ($3 \times 640 \times 480$)		
Image Features ($512 \times 20 \times 15$)	Project proposal from the input image to features. 	Project proposal from the input image to features. 
	Divide into 2x2 grid roughly (7x7 in paper) 	Divide into equal size region 
		Sample features at regularly-spaced points in each subregion using bilinear interpolation 
Region Feature ($512 \times 2 \times 2$)	MaxPool to obtain 2x2  	MaxPool to obtain 2x2  
Picture Credit: Justin Johnson, Lecture 15, “Object Detection”, EECS 498-007 / 598-005 Deep Learning for Computer Vision, University of Michigan		

3.4 Bilinear Interpolation:



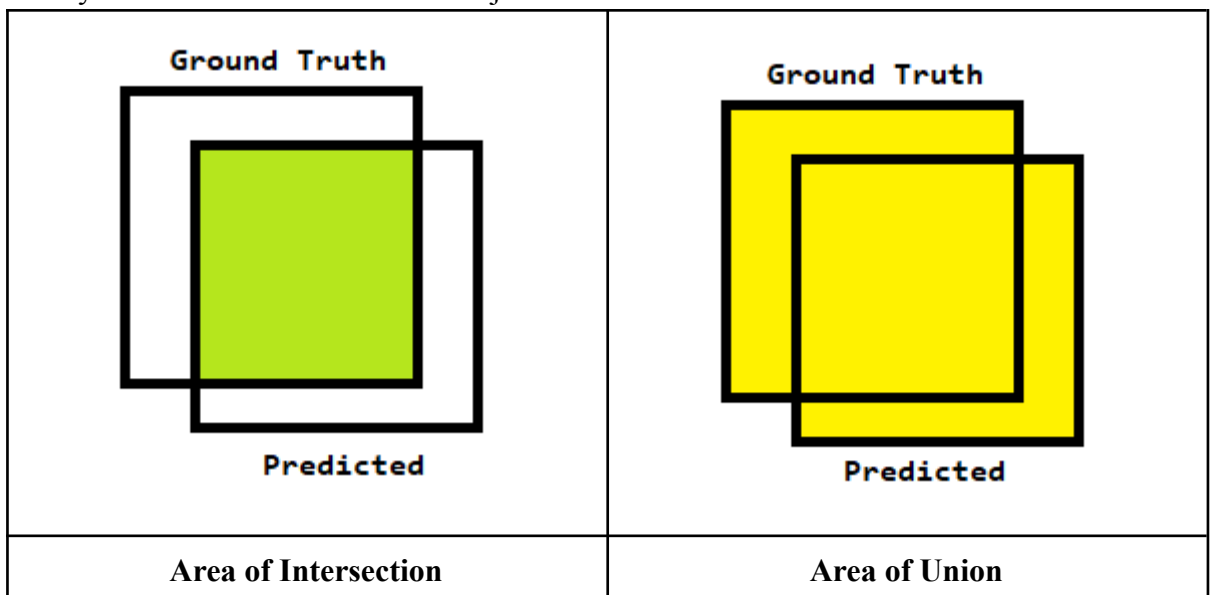
$$f_{xy} = \sum_{i,j=1}^2 f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_j|)$$

$$f_{6.5,5.8} = (f_{6,5} * 0.5 * 0.2) + (f_{7,5} * 0.5 * 0.2) + (f_{6,6} * 0.5 * 0.8) + (f_{7,6} * 0.5 * 0.8)$$

This example shows how bilinear is used to find continuous points in the RoIAlign Layers.

3.5 Intersection Over Union (IOU)

There are N bounding boxes(bbs) predicted by the algorithm and then we have to map them to the ground truth. For which we need to define some kind of matching measure with which we say whether this bb contains an object or not and which one.



$$\text{Intersection over Union} = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$

a) bb has an object:

If $IOU < 0.5(\text{common})$ for all the ground truths then bb does not contain any object.

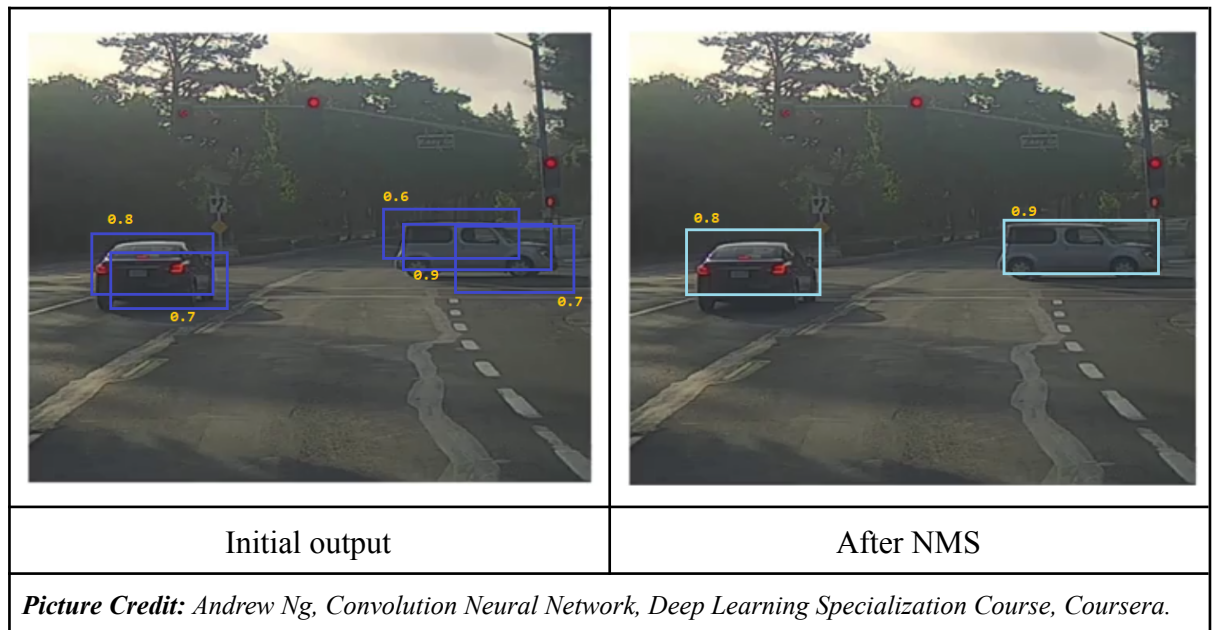
b) bb has which object

If $IOU \geq 0.5(\text{common})$ then pair the bb with the ground truth which it has maximum IOU.

This IOU parameter(0.5) is set by the user and depends on the particular problem that is being solved.

3.6 Non-Maximum Suppression (NSM)

In the output, the algorithm detects multiple bbs for a single object with a slight movement of the bb window. But we need to show one bb per object. To solve this problem we use NSM.



Algorithm:

Repeat until all objects are detected:

- Select the bb with maximum probability (0.9 in the first iteration).
- Calculate IOU between this bb and all other bbs (5 others in the first iteration).
- Discard bbs with $\text{IOU} > k$, commonly $k = 0.5$

3.7 Loss

For training multi-task loss is calculated for each RoI as:

$$L = \text{Loss}_{cls} + \text{Loss}_{box} + \text{Loss}_{mask}, \text{ where}$$

Loss_{cls} and Loss_{box} are similar to those used in Fast R – CNN

Loss_{mask} is average binary cross-entropy loss per pixel.

$$\text{BCE} = \frac{1}{N} \sum_{i=1}^N (y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$$

3.8 Training

Following Fast R-CNN, an RoI is considered positive if its IOU is at least 0.5 with the ground-truth box and negative otherwise. The mask loss Loss_{mask} is defined only of positive RoIs because they only contain an object for which the mask needs to be predicted. The mask targets to find the intersection between RoI and its associated ground-truth mask.

- Images are resized such that their scale (shorter edge) is 800 pixels
- Each mini-batch has 2 images per GPU and each image has N sampled RoIs.
- N is 64 for the C4 backbone and 512 for FPN.
- Model is trained on 8 GPUs (so effective minibatch size is 16) for 160k iterations, with a learning rate of 0.02 which is decreased by 10 at the 120k iteration (0.002).
- ResNeXt, we train with 1 image per GPU and the same number of iterations, with a starting learning rate of 0.01.
- The RPN anchors span 5 scales and 3 aspect ratios.

3.9 Testing

For testing there are a few changes in the model as follows:

- The proposal number is 300 for C4 backbone and 1000 for FPN.
- Box prediction branch runs on these proposals, followed by non-maximum suppression.
- The mask branch is then applied to the highest scoring 100 detection boxes.
- The sequence of models during testing is Box Predictions \rightarrow NMS \rightarrow Mask Branch.
- Although this differs from the parallel computation used in training, it speeds up inference and improves accuracy (due to the use of fewer, more accurate RoIs).

4. Datasets and Accuracy Measure

4.1 PASCAL Visual Object Classes (VOC) Dataset

- VOC2005 to VOC2012.
- Contains 20 classes.
- Used for object detection, and segmentation.
- Used Mean Average Precision (MAP) for Accuracy Measure.

4.2 COCO Dataset

- Latest Dataset.
- Contains almost 80 classes.
- Use Average over multiple IOU (AP) for Accuracy Measure.
- AP@[.50:.05:.95]**: corresponds to the average AP for IoU starting from 0.5 to 0.95 with a step size of 0.05.

5. Results

5.1 Main Results

These result are shown with some other state-of-art instance segmentation algorithms comparison. And Mask R-CNN achieve best results with the deep network like ResNeXt-101-FPN.

	backbone	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
MNC [10]	ResNet-101-C4	24.6	44.3	24.8	4.7	25.9	43.6
FCIS [26] +OHEM	ResNet-101-C5-dilated	29.2	49.5	-	7.1	31.3	50.0
FCIS+++ [26] +OHEM	ResNet-101-C5-dilated	33.6	54.5	-	-	-	-
Mask R-CNN	ResNet-101-C4	33.1	54.9	34.8	12.1	35.6	51.1
Mask R-CNN	ResNet-101-FPN	35.7	58.0	37.8	15.5	38.1	52.4
Mask R-CNN	ResNeXt-101-FPN	37.1	60.0	39.4	16.9	39.9	53.5

Table 1. **Instance segmentation mask AP** on COCO test-dev. MNC [10] and FCIS [26] are the winners of the COCO 2015 and 2016 segmentation challenges, respectively. Without bells and whistles, Mask R-CNN outperforms the more complex FCIS+++, which includes multi-scale train/test, horizontal flip test, and OHEM [38]. All entries are *single-model* results.

6. References

- [1] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.
- [2] R. Girshick. Fast R-CNN. In ICCV, 2015.
- [3] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. IJCV, 2013.
- [4] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In NIPS, 2015.
- [5] K. He, G. Gkioxari, P. Dollar, R. Girshick. Mask R-CNN. In IEEE, 2017