**MINOR MID-EVALUATION REPORT**

# Instance Segmentation using Faster-RCNN

PANEL JURY:

Dr. PAWAN UPADHYAY
Ms. Sonal

SUBMITTED BY:

AKSHAT UPADHYAY
HARSH PANDEY
SHASHWAT SINGH

MINOR PROJECT MENTOR:
Dr. ANUJA ARORA

PANEL & GROUP NUMBER:
PANEL 03-GROUP-05

## Problem Statement:

Instance segmentation is the problem of detecting and identifying each distinct object of interest in an image. Image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as image objects). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

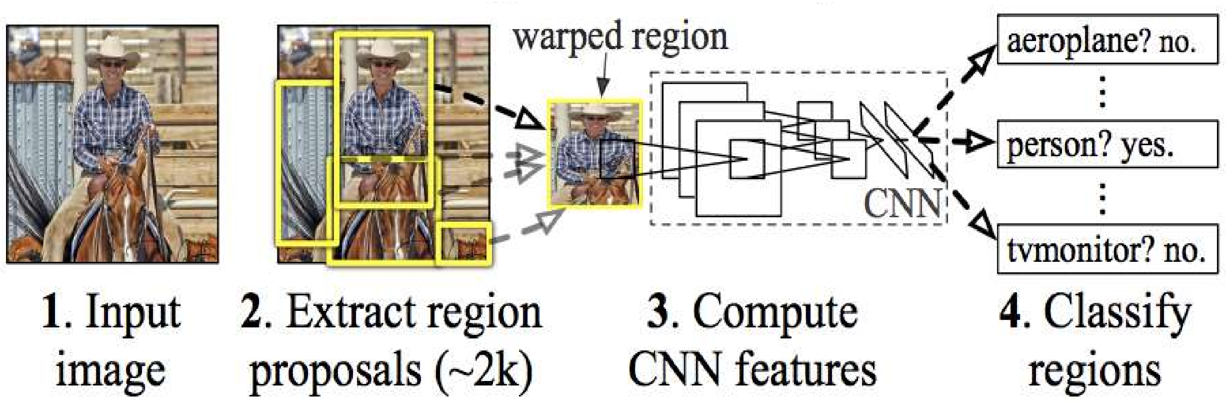
Instance segmentation finds its applicability in various sectors few of which are mentioned below:

- Self-Driving Cars for obstacle identification and avoidance
- Implementation of bokeh effect in smartphone cameras
- Unwanted object removal in video editing softwares
- 3D reconstruction using aerial images

But the problem is how to precisely segment instances in an image precisely. Current instance segmentation approaches consist of a combination of modules that are trained independently of each other, thus missing opportunities for joint learning.

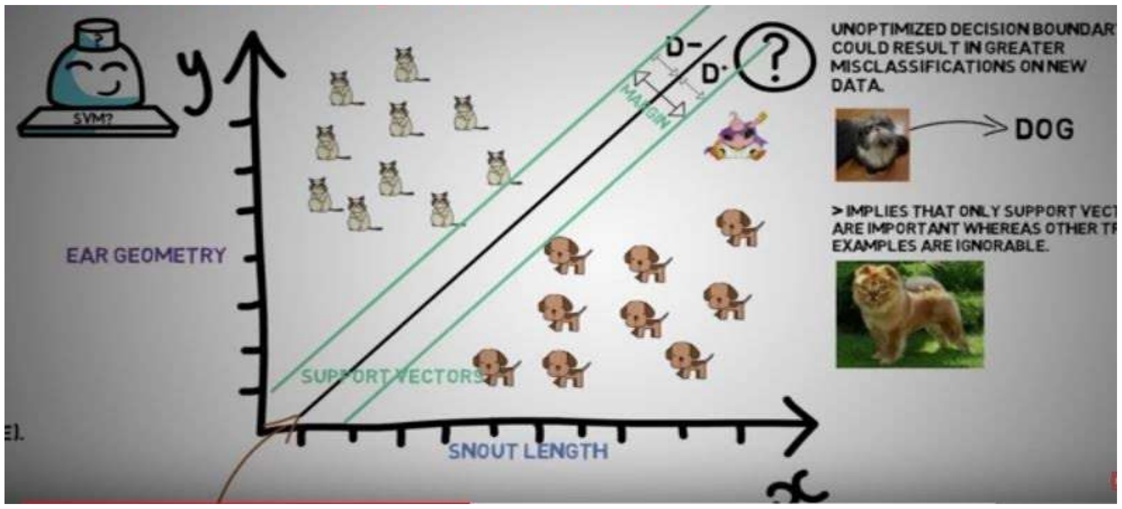
## Literature Survey:

### Real time Object Detection using R- CNN



Thepaper has considered 3 major modules in their object detection model:

1. First module -- generates region proposals independent of categories.
2. Second module -- large convolutional neural network which extracts fixed size feature vectors from the individual region proposed.
3. Third module -- set of class specific linear SVMs(*Support Vector Machine*).


SVM?
y
EAR GEOMETRY
SNOUT LENGTH
x
D-
D
MARGIN
SUPPORT VECTORS
UNOPTIMIZED DECISION BOUNDAR
COULD RESULT IN GREATER
MISCLASSIFICATIONS ON NEW
DATA.
DOG
> IMPLIES THAT ONLY SUPPORT VECT
ARE IMPORTANT WHEREAS OTHER TR
EXAMPLES ARE IGNORABLE.

Support Vector Machine(**SVMs**)

## Region Proposals

They are using selective search so that they can have a check on their work with respect to various prior works done in this field.

## Feature Extraction

They first convert image data into fixed size image of *227 X 227  pixel* size RGB image and then this image in forward propagated through five convolutional layers and two fully connected layers.Every bounding boxes  pixels are also wrapped to smaller size irrespective of their original size to new size of *p* pixels  Here they have used *p = 16.*

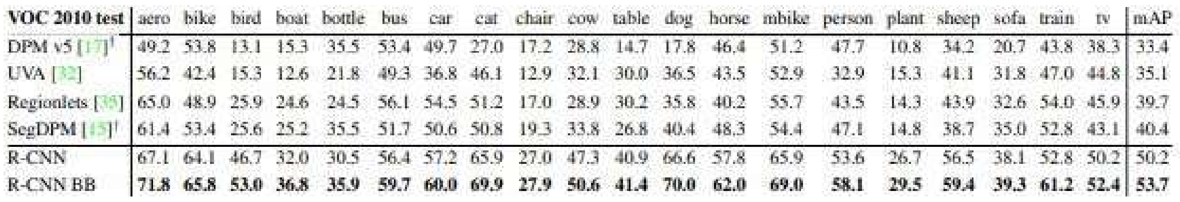
## Binary Classification

It's obvious that Region proposal enclosing a desired object image should be positively classified whereas Region proposal enclosing only natural scene background  must be negatively classified. But if  Region Proposals which partially includes a desired object image should be either classified  positively or negatively is decided using **IoU** (Intersection Over Union).
If IoU value is <0.3 is considered a negative example, and IoU >0.3 is considered a positive example. Various  threshold has been considered for this purpose but if threshold is changed to 0.5 decreases mAP*(mean Average Precision)* by 5 points similarly decreasing it to 0.0 decreases mAP value by 4 points.

Once features are extracted image data is then passed through the third module to give final output.

## Summary

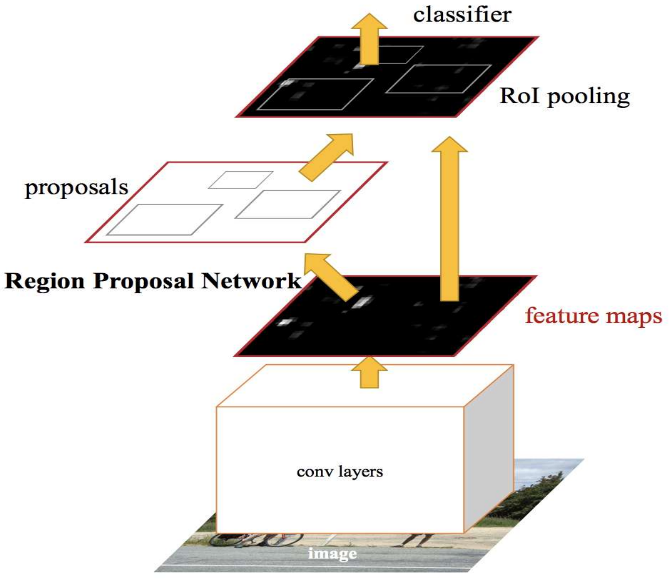
- Generate a Set of Proposals
- Runs the image through pre-trained Alexnet
- Run the box through a linear regression model



| **VOC 2010 test** | aero | bike | bird | boat | bottle | bus | car | cat | chair | cow | table | dog | horse | mbike | person | plant | sheep | sofa | train | tv | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DPM v5 [17]† | 49.2 | 53.8 | 13.1 | 15.3 | 35.5 | 53.4 | 49.7 | 27.0 | 17.2 | 28.8 | 14.7 | 17.8 | 46.4 | 51.2 | 47.7 | 10.8 | 34.2 | 20.7 | 43.8 | 38.3 | 33.4 |
| UVA [32] | 56.2 | 42.4 | 15.3 | 12.6 | 21.8 | 49.3 | 36.8 | 46.1 | 12.9 | 32.1 | 30.0 | 36.5 | 43.5 | 52.9 | 32.9 | 15.3 | 41.1 | 31.8 | 47.0 | 44.8 | 35.1 |
| Regionlets [35] | 65.0 | 48.9 | 25.9 | 24.6 | 24.5 | 56.1 | 54.5 | 51.2 | 17.0 | 28.9 | 30.2 | 35.8 | 40.2 | 55.7 | 43.5 | 14.3 | 43.9 | 32.6 | 54.0 | 45.9 | 39.7 |
| SegDPM [15]† | 61.4 | 53.4 | 25.6 | 25.2 | 35.5 | 51.7 | 50.6 | 50.8 | 19.3 | 33.8 | 26.8 | 40.4 | 48.3 | 54.4 | 47.1 | 14.8 | 38.7 | 35.0 | 52.8 | 43.1 | 40.4 |
| R-CNN | 67.1 | 64.1 | 46.7 | 32.0 | 30.5 | 56.4 | 57.2 | 65.9 | 27.0 | 47.3 | 40.9 | 66.6 | 57.8 | 65.9 | 53.6 | 26.7 | 56.5 | 38.1 | 52.8 | 50.2 | 50.2 |
| R-CNN BB | **71.8** | **65.8** | **53.0** | **36.8** | **35.9** | **59.7** | **60.0** | **69.9** | **27.9** | **50.6** | **41.4** | **70.0** | **62.0** | **69.0** | **58.1** | **29.5** | **59.4** | **39.3** | **61.2** | **52.4** | **53.7** |

*Above table is the mAP (mean Average Precision) score of R-CNN model on data set VOC 2010*

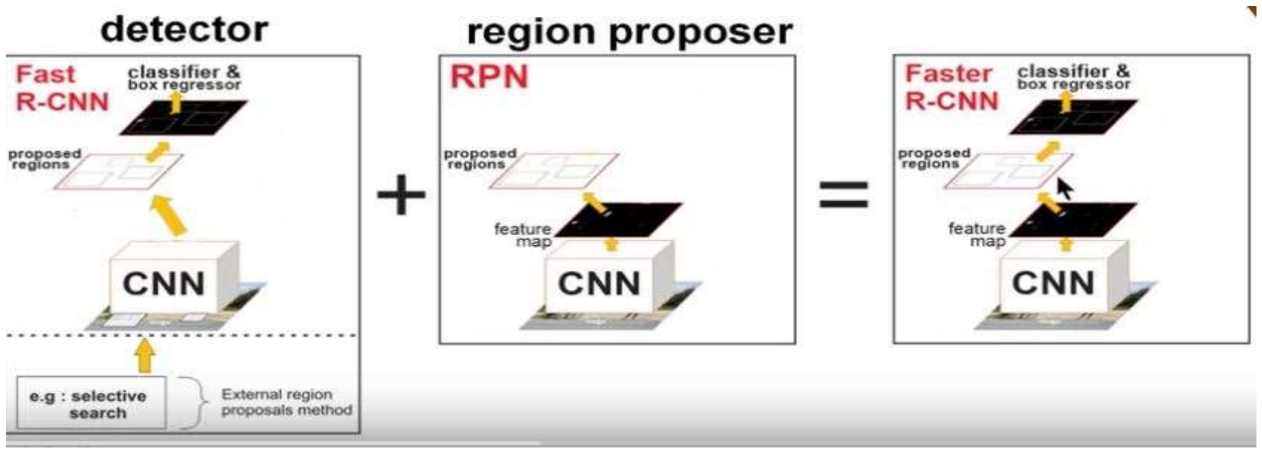
## Real time Object Detection using Faster R-CNN



In this architecture, In spite of passing image data to Region Proposal module and then applying module of convolutional neural network they first passing image data through their convolutional layers and then through Region Proposals Network (*RPN*).

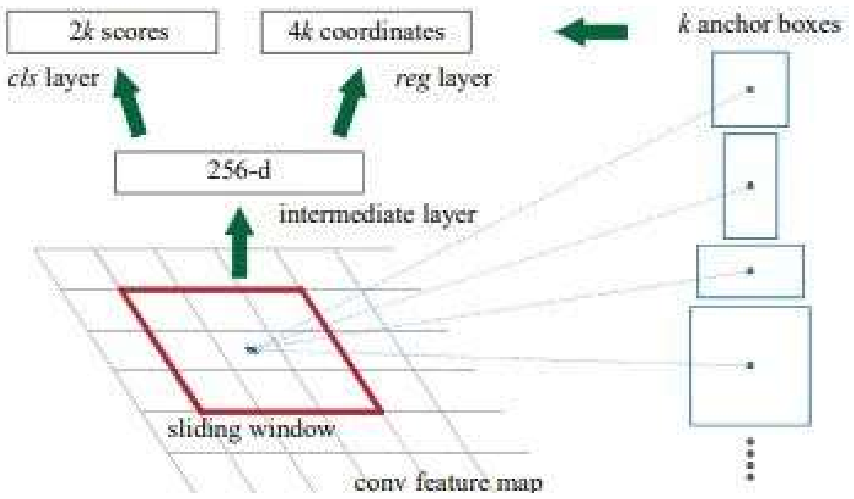
They have considered 3 major modules in their object detection model:

1. Convolutional Layers.
2. Region Proposal Network.
3. Fast RCNN part
    a. Extract features using ROI pooling
    b. Forms binary classification



## Region Proposal Network

The time cost of generating region proposals is much smaller in RPN than selective search. In this module they have used ZFnet (*Improvement of AlexNet*) which consists of 5 shareable convolutional layers.To generate region proposals they slide small window over the feature map that is generated by the last layer of ZFnet. This feature is fed into two sibling fully connected layers—a box-regression layer (reg) and a box-classification layer (cls). As box regression layers contain 4 inputs coordinates hence total number of data with this layer is 4k coordinates and box-classification contains 2k scores.



But here k is taken as to be 9 hence we have in total 4*9 coordinates in regression layer and 2*9 scores in cls layer. To overcome the problem that each Region Proposal network can detect only one object we use the idea of Anchor boxes whose count is represented as k in the above figure

.

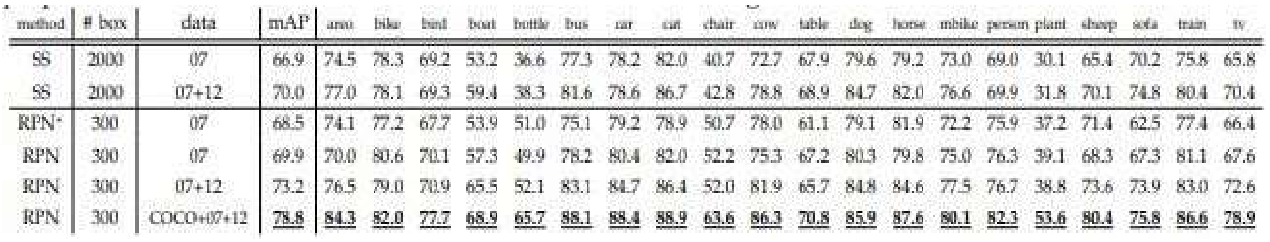
For Region Proposals containing more than one objects anchor boxes of different dimensions are tested on the region and classes are selected using the formulae:

$$\text{IoU} = \frac{\text{Anchor\_box} \cap \text{Ground\_truth}}{\text{Anchor\_box} \cup \text{Ground\_truth}}$$

If IoU >0.7 object
If IoU >0.3 not object

Hence in

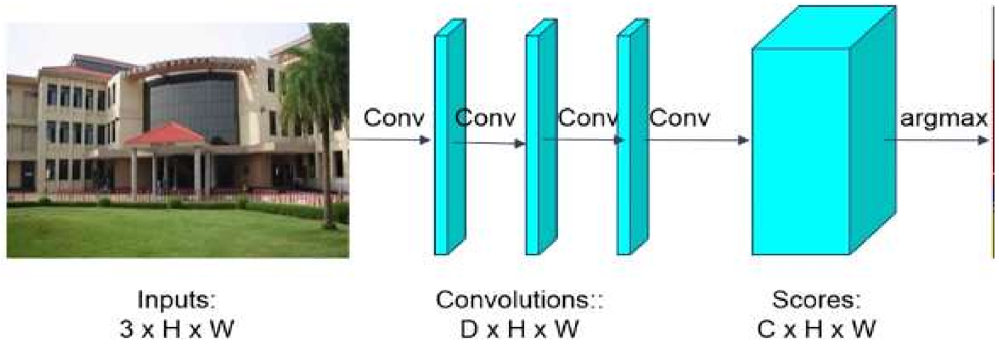
- **Reg layer = [** $X_{mid}$ $Y_{mid}$ H W **] * 9**
- **Cls layer = [** $Score_{yes}$ $Score_{no}$ **] * 9**



| method | # box | data | mAP | areo | bike | bird | boat | bottle | bus | car | cat | chair | cow | table | dog | horse | mbike | person | plant | sheep | sofa | train | tv |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SS | 2000 | 07 | 66.9 | 74.5 | 78.3 | 69.2 | 53.2 | 36.6 | 77.3 | 78.2 | 82.0 | 40.7 | 72.7 | 67.9 | 79.6 | 79.2 | 73.0 | 69.0 | 30.1 | 65.4 | 70.2 | 75.8 | 65.8 |
| SS | 2000 | 07+12 | 70.0 | 77.0 | 78.1 | 69.3 | 59.4 | 38.3 | 81.6 | 78.6 | 86.7 | 42.8 | 78.8 | 68.9 | 84.7 | 82.0 | 76.6 | 69.9 | 31.8 | 70.1 | 74.8 | 80.4 | 70.4 |
| RPN* | 300 | 07 | 68.5 | 74.1 | 77.2 | 67.7 | 53.9 | 51.0 | 75.1 | 79.2 | 78.9 | 50.7 | 78.0 | 61.1 | 79.1 | 81.9 | 72.2 | 75.9 | 37.2 | 71.4 | 62.5 | 77.4 | 66.4 |
| RPN | 300 | 07 | 69.9 | 70.0 | 80.6 | 70.1 | 57.3 | 49.9 | 78.2 | 80.4 | 82.0 | 52.2 | 75.3 | 67.2 | 80.3 | 79.8 | 75.0 | 76.3 | 39.1 | 68.3 | 67.3 | 81.1 | 67.6 |
| RPN | 300 | 07+12 | 73.2 | 76.5 | 79.0 | 70.9 | 65.5 | 52.1 | 83.1 | 84.7 | 86.4 | 52.0 | 81.9 | 65.7 | 84.8 | 84.6 | 77.5 | 76.7 | 38.8 | 73.6 | 73.9 | 83.0 | 72.6 |
| RPN | 300 | COCO+07+12 | **78.8** | **84.3** | **82.0** | **77.7** | **68.9** | **65.7** | **88.1** | **88.4** | **88.9** | **63.6** | **86.3** | **70.8** | **85.9** | **87.6** | **80.1** | **82.3** | **53.6** | **80.4** | **75.8** | **86.6** | **78.9** |

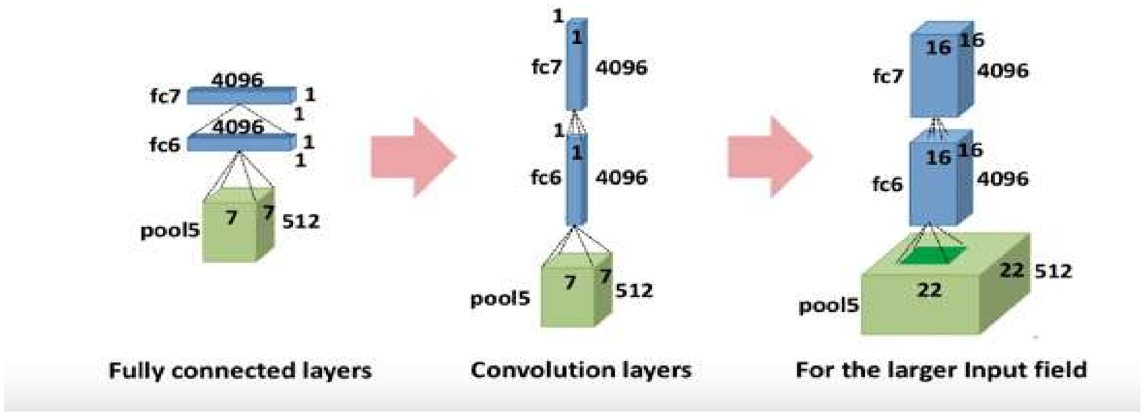
*Above table is the mAP (mean Average Precision) score of FAster R-CNN model on data set VOC 2007*

# Fully Convolutional Networks For Semantic Segmentation

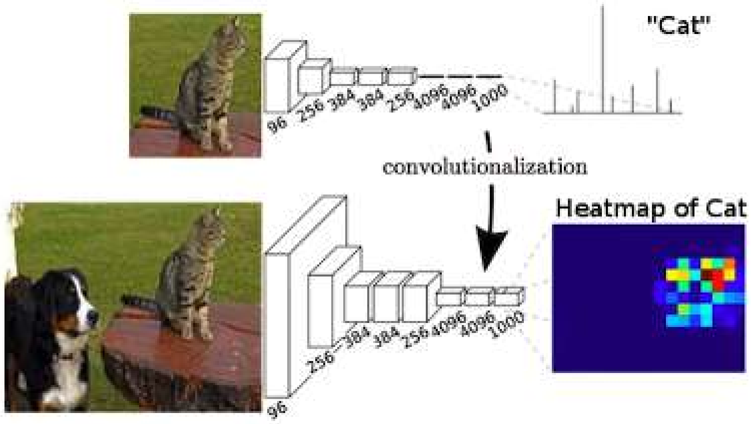


In this paper a fully convoluted network is used where the first layer is the image, with pixel size h × w, and d color channels. Locations in higher layers correspond to the locations in the image they are path-connected to, which are called their receptive fields. Each layer of data in a convnet is a three-dimensional array of size h × w × d, where h and w are spatial dimensions, and d is the feature or channel dimension.Their basic components (convolution, pooling, and activation functions) operate on local input regions. An FCN naturally operates on an input of any size, and produces an output of corresponding (possibly resampled) spatial dimensions.

## Fully connected layer to Fully convoluted layer



All the remaining fully connected layers are converted by introducing kernels that cover their entire input region. Basically fixing the number of channels (say 4096) what we get is a w x h feature map.



Transforming fully connected layers into convolution layers enables a classification net to output a heatmap with every pixel showing the probability of belonging to that object. But the output is coarse as it is downsampled and so are the probabilities. Therefore outputs of multiple convolutional layers go through 1x1 convolution operation to generate multiple feature maps and then upsample them and add, this upsampling is learned.

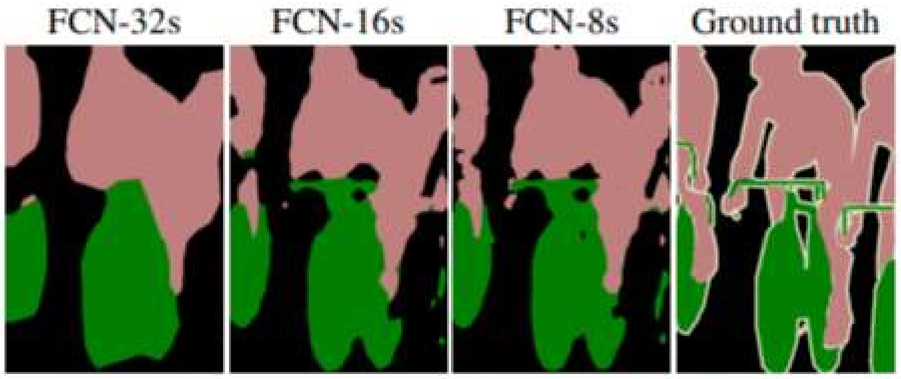


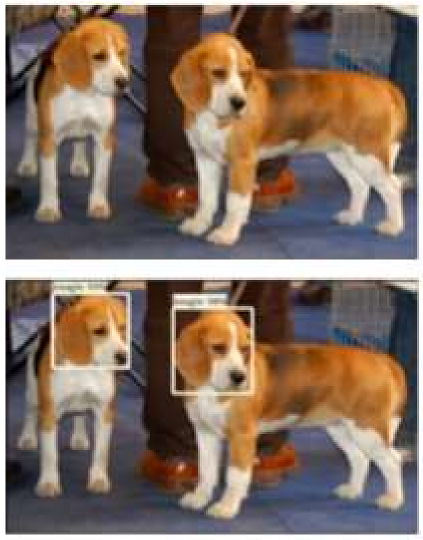
Figure 4. Refining fully convolutional nets by fusing information from layers with different strides improves segmentation detail. The first three images show the output from our 32, 16, and 8 pixel stride nets (see Figure 3).

The 32 pixel stride at the final prediction layer limits the scale of detail in the upsampled output. We address this by adding skips that combine the final prediction layer with lower layers with finer strides, combining coarse and fine layers help model make better predictions that are similar to original image.

## Work Flow

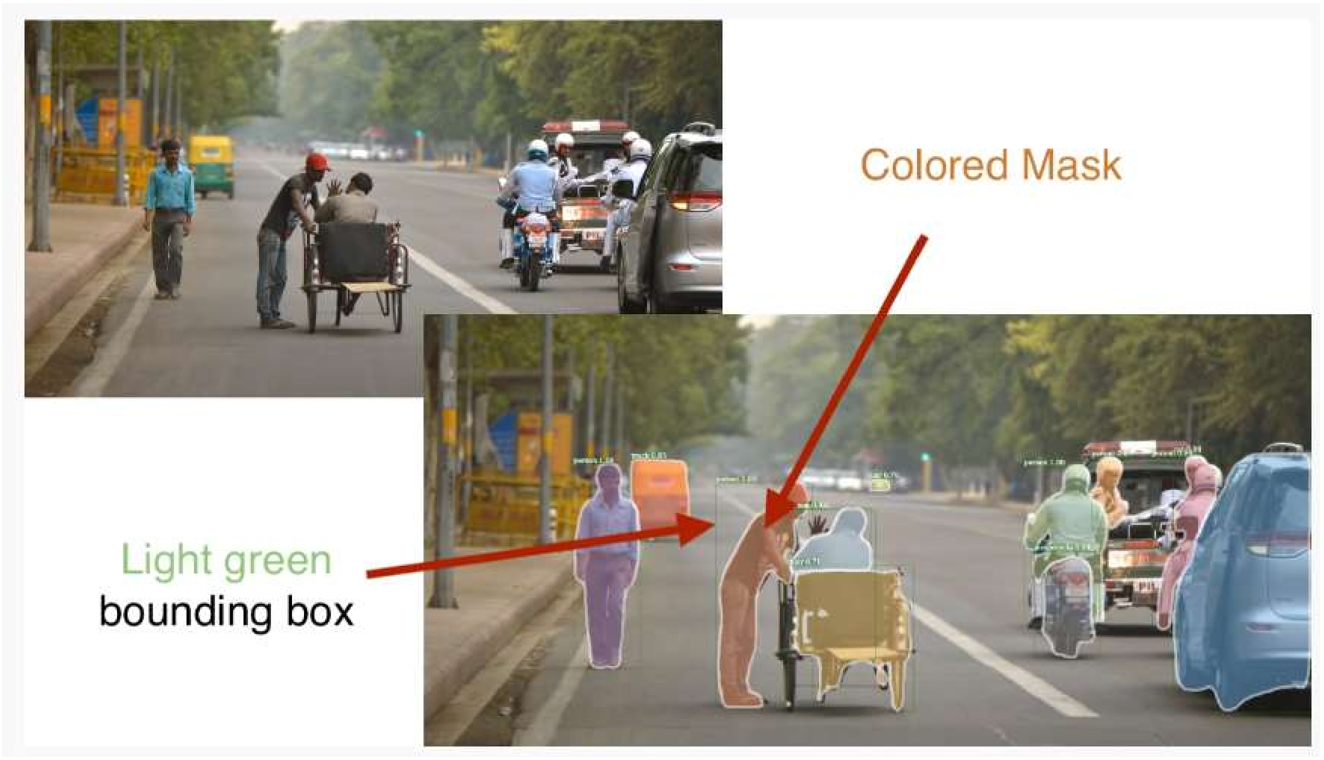
There are two sub problems:

1) **Object Detection -**



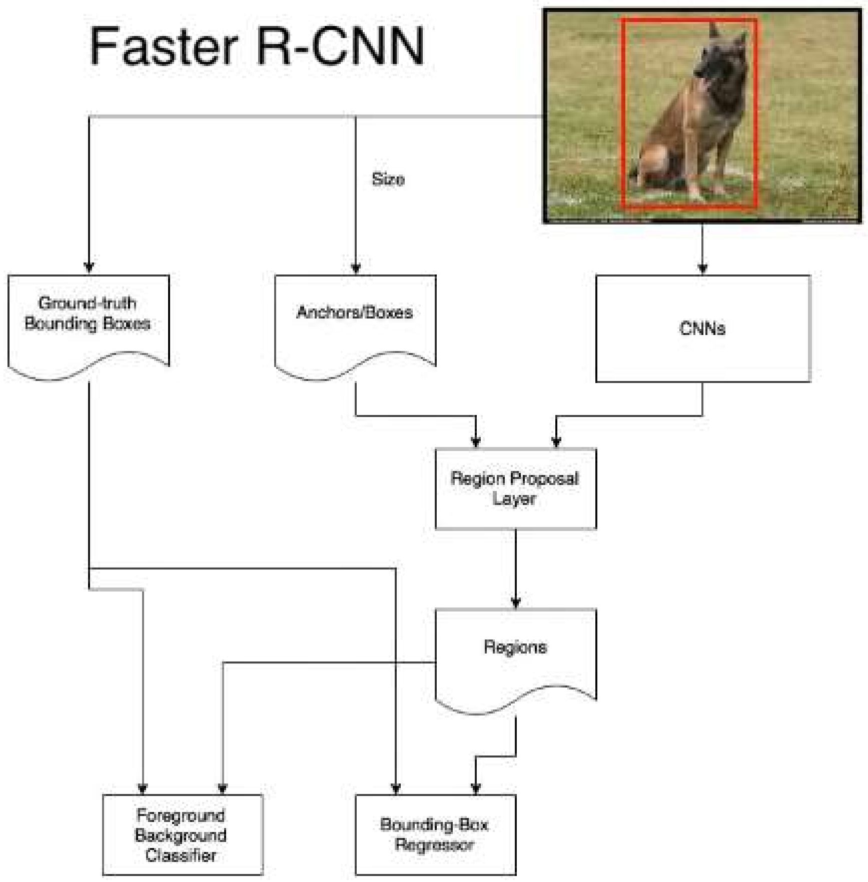
2) **Semantic Segmentation-**

Combining both results in **instance segmentation**-



The bounding boxes are a result of object detection and colored regions are output of semantic segmentation.

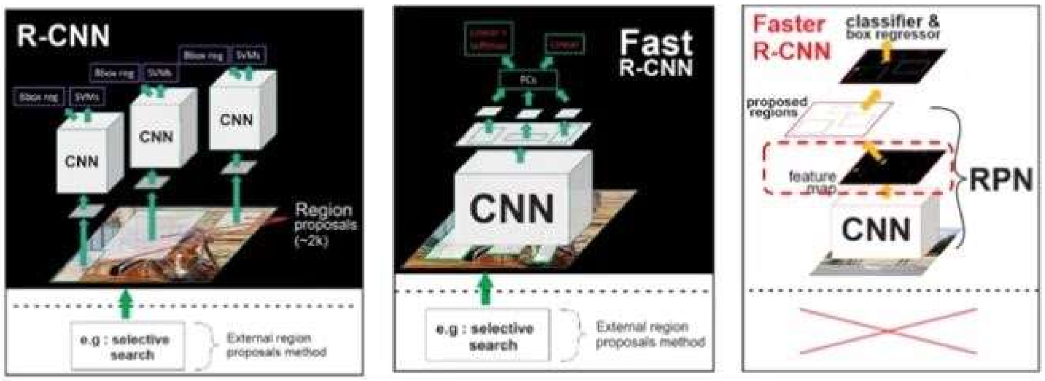
For Object Detection we will use Faster RCNN -

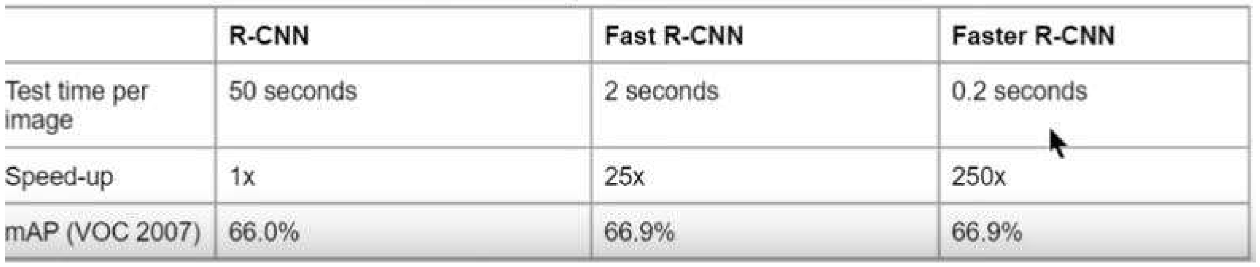


The only difference will be in the final stage where a binary mask will also be the output along with the box coordinates and class for that region of interest in parallel.

## Tools and techniques used

Faster RCNN over  RCNN  takes a huge amount of time to train the network as you would have to classify 2000 region proposals per image.The network does not look at the complete image. Instead, parts of the image which have high probabilities of containing the object. YOLO or You Only Look Once is an object detection algorithm much different from the region based algorithms seen above. In YOLO a single convolutional network predicts the bounding boxes and the class probabilities for these boxes.





| | R-CNN | Fast R-CNN | Faster R-CNN |
|---|---|---|---|
| Test time per image | 50 seconds | 2 seconds | 0.2 seconds |
| Speed-up | 1x | 25x | 250x |
| mAP (VOC 2007) | 66.0% | 66.9% | 66.9% |

2)The backbone of this architecture is ResNet50
3) Training will be done on MS COCO dataset.
4) Dependencies:

- Numpy
- Scipy
- Scikit-image
- Tensorflow
- Keras

## Other References

1. https://www.youtube.com/watch?v=4tkgOzQ9yyo
2. https://medium.com/@smallfishbigsea/faster-r-cnn-explained-864d4fb7e3f8
3. https://medium.com/coinmonks/review-r-cnn-object-detection-b476aba290d1
4. https://medium.com/@andrewjong/how-to-use-roi-pool-and-roi-align-in-your-neural-networks-pytorch-1-0-b43e3d22d073
5. https://www.youtube.com/watch?v=a9_8wqMxVkY&t=1140s
6. https://medium.com/@kegui/what-is-the-difference-between-nearest-neighbor-bilinear-interpolation-and-cubic-convolution-9589c58e2f17
7. https://www.youtube.com/watch?v=v5bFVbQvFRk
8. https://medium.com/@purnasaigudikandula/a-beginner-intro-to-convolutional-neural-networks-684c5620c2ce
9. https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33
10. https://medium.com/@14prakash/understanding-and-implementing-architectures-of-resnet-and-resnext-for-state-of-the-art-image-cf51669e1624