# HOW TO RUN

Just type “python3 submit.py” to run the file

# EXPLANATION OF submit.py

1) Setting up the configuration file. The configuration file needed to be modified to match the computation ability of my gpu(NVIDIA 1050 ti). So i made the necessary changes in the parameters GPU_COUNT, IMAGES_PER_GPU, IMAGE_MAX_DIM and IMAGE_MIN_DIM.

2) The training and validation datasets are prepared by loading the masks and images from the stage1_train folder and are splitted into 90% train and 10% test respectively.

3) A check is made to ensure if the masks ,images ,class_ids and class_names have been properly allocated to the training and validation datasets.

4) model for training mode is created and configuration along the the directory where the model is to be stored is fed into the model.

5) I have used transfer learning to train the resnet101 model by using the pretrained weights of the coco dataset.

6) I have freezed all the layers except the head layer as we are using pretrained bottleneck weights of the coco dataset. The model is trained for 1 epoch comprising of 200 steps .

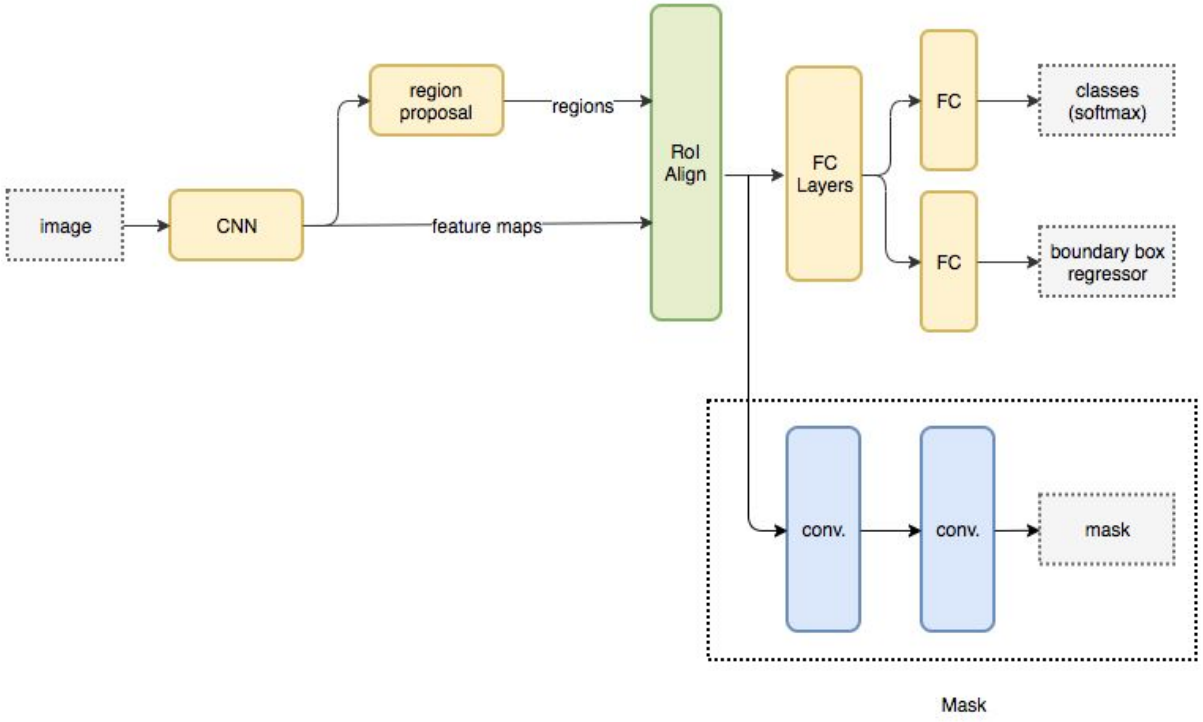
7) The trained weights are then saved into the model directory for future use.

8) As a check on the computed weights the masks and the bounding boxes are created on a random image from the validation dataset.

9) Finally mean average precision is calculated on the validation and train dataset with an iou of 0.5.

# ABOUT MASK_RCNN

Mask R-CNN is just an extension to the already existing Faster R-CNN .It is an improvement on Faster R-CNN as it can correct the displacements of the bounding boxes and also create a mask around the object enclosed within the bounding box.The Faster R-CNN builds all the ground works for feature extractions and ROI proposals. At first sight, performing image segmentation may require more detail analysis to colorize the image segments. By surprise, not only we can piggyback on this model, the extra work required is pretty simple. After the ROI pooling, we add 2 more convolution layers to build the mask.



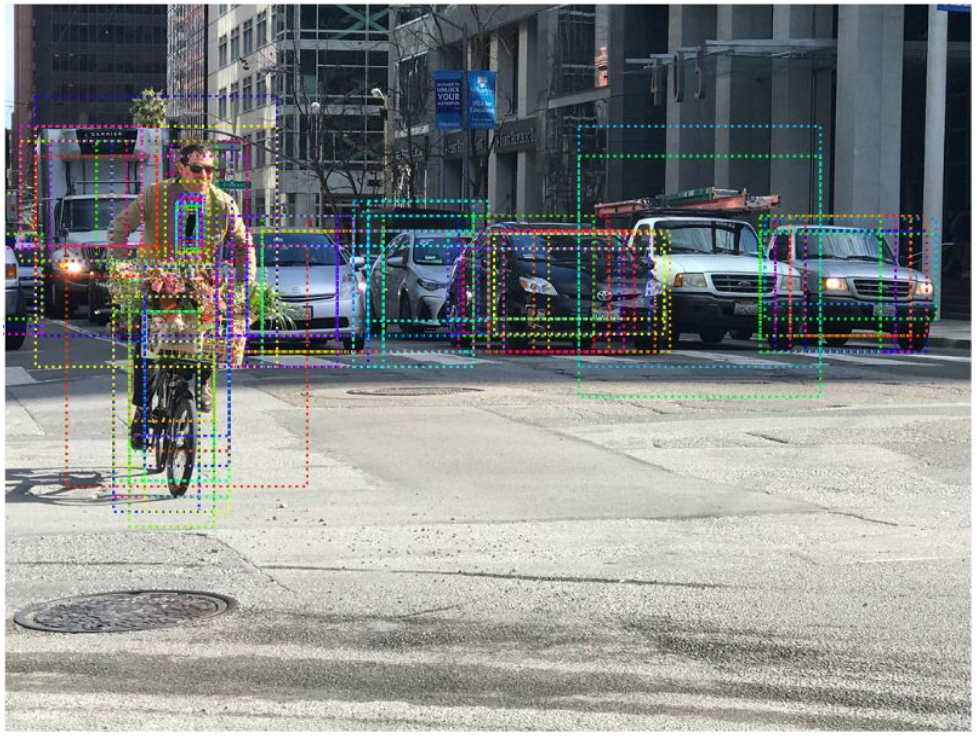

The figure shows the extra 2 CNN layers to compute the mask on the pooled region proposals.

**ROI Align**

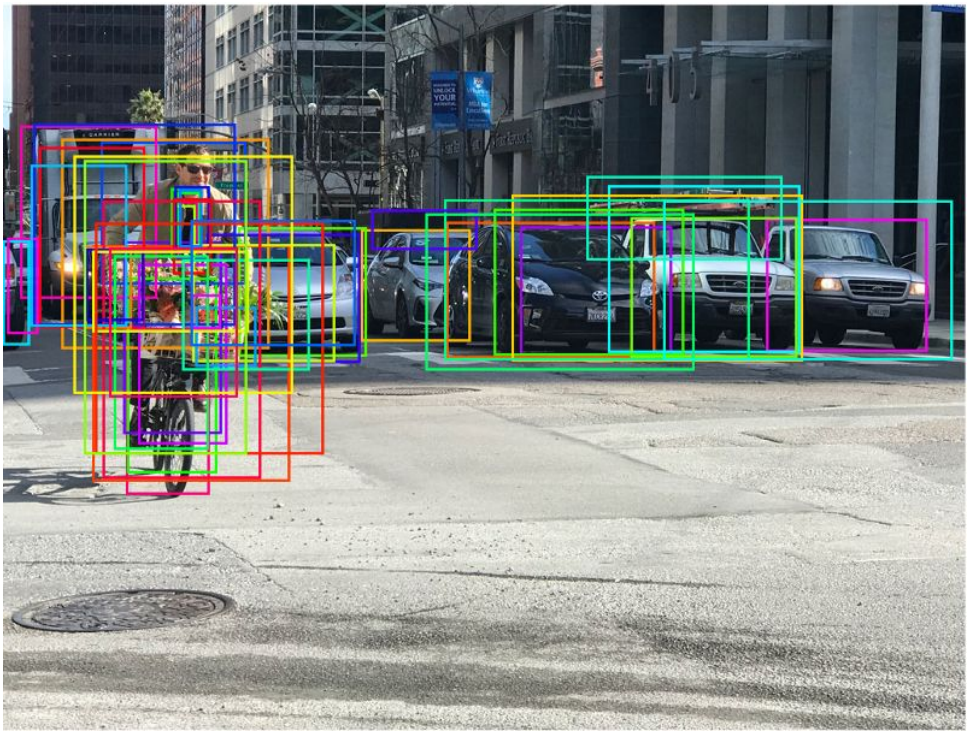
Another major contribution of Mask R-CNN is the refinement of the ROI pooling. In ROI, the warping is digitalized (top left diagram below): the cell boundaries of the target feature map are forced to realign with the boundary of the input feature maps. Therefore, each target cells may not be in the same size (bottom left diagram). Mask R-CNN uses ROI Align which does not digitalize the boundary of the cells (top right) and make every target cell to have the same size (bottom right). It also applies interpolation to calculate the feature map values within the cell better. For example, by applying interpolation, the maximum feature value on the top left is changed from 0.8 to 0.88 now.

# Mask R-CNN visualization

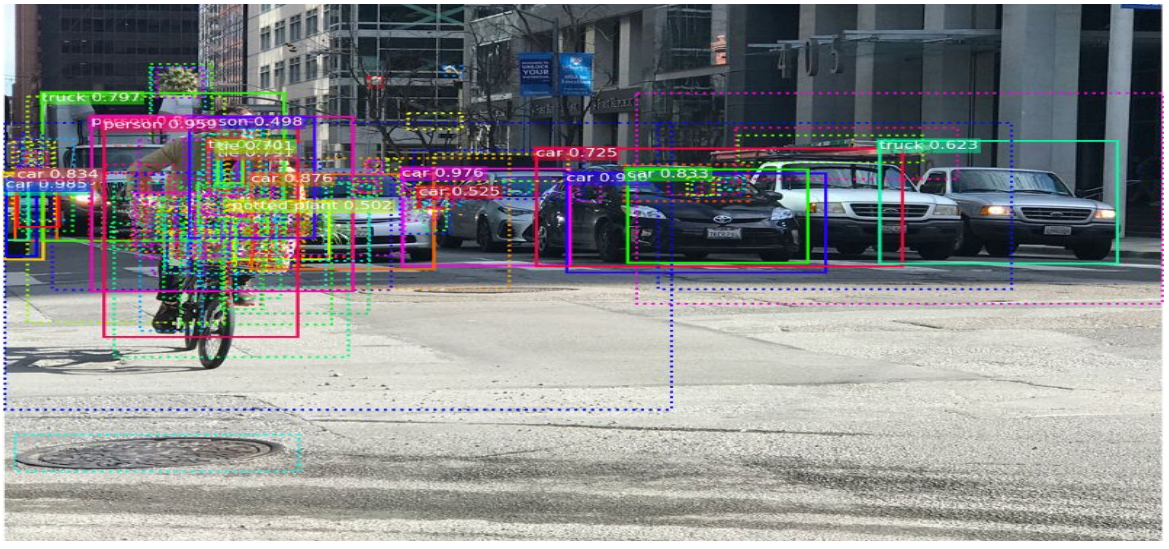
Let’s visualize some of the major steps in Mask R-CNN/Faster R-CNN. Using the region proposal network, we make ROI proposals. The dotted rectangles below are those proposals but, for demonstration purpose, we decide to display those that have high final scores only.



Here are the boxes after boundary box refinements when we make final classification and localization predictions. The boundary box encloses the ground truth objects better.



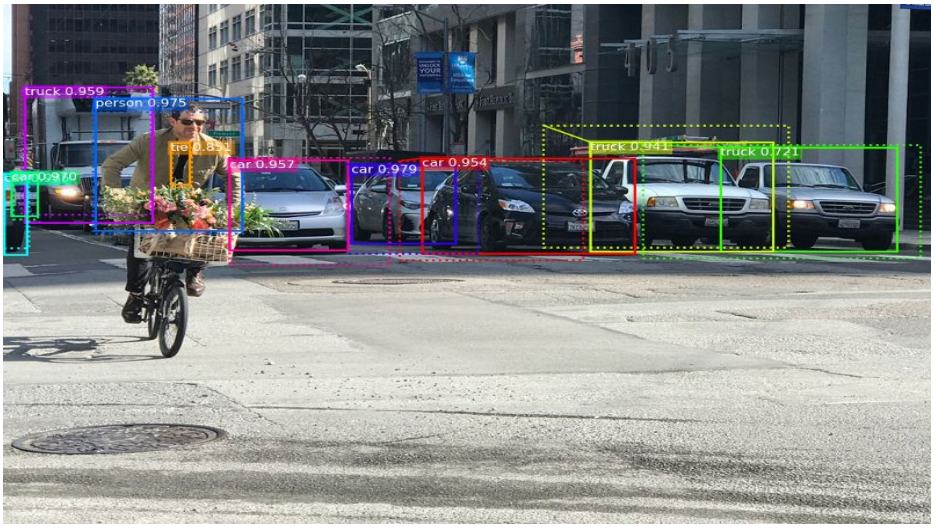
Just like Faster R-CNN, it performs object classification based on the ROIs (dotted lines) from RPN. The solid line is the boundary box refinements in the final predictions.



## Perform the per class non-maximum suppression

(nms)

It groups highly-overlapped boxes for the same class and selects the most confidence prediction only. This avoids duplicates for the same object.




This is top final predictions from Mask R-CNN.