**Aero2Astro**

**YOLO V2 and V3**

**Report**

**By**

**Anshul Singh**

**Research intern-Inspect**

**YOLO V2**

Following were some of the improvements in YOLO v1 architecture that leads us to YOLO v2:

**Batch Normalization:**

By adding batch normalization on all of the convolutional layers in YOLO we get more than 2% improvement in mAP. By applying batch normalization followed by all YOLOv2's convolution layers. This technique not only reduces training time but also increases the generalization of the network. The network also does not need to use additional Dropouts to avoid overfitting.

**High Resolution classifier:**

In YOLOv2, after completing the training phase of the feature extractor with the 224 × 224 input image, the model continued to train the feature extractor for more 10 epochs with 448 × 448 input image before using the architecture for training object detector. This high-resolution classification network gives an increase of almost 4% mAP.

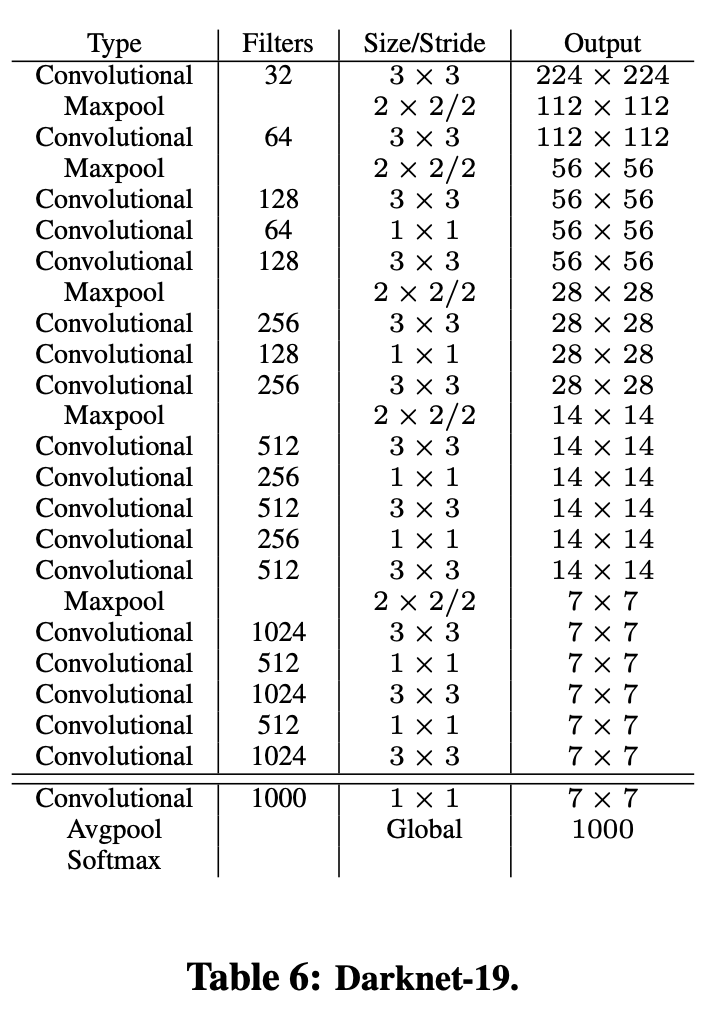
**Convolutions with Anchor Boxes**

* YOLOv2 removes all fully connected layers and uses anchor boxes to predict bounding boxes.
* One pooling layer is removed to increase the resolution of output.
* And 416×416 images are used for training the detection network now.
* And 13×13 feature map output is obtained, i.e. 32× downsampled.
* Without anchor boxes, the intermediate model got 69.5% mAP and recall of 81%.
* With anchor boxes, 69.2% mAP and recall of 88% are obtained. Though mAP is dropped a little, recall is increased by large margin.

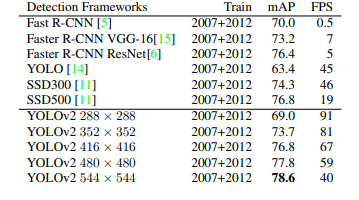
**Multi scale training:**

To aid problem of multi scale training, Instead of fixing the input image size we change the net-work every few iterations. Every 10 batches our network randomly chooses a new image dimension size.

Following is the newly proposed YOLO v2 architecture.



Results of comparison of YOLO V2 with other architectures:



**YOLOv3**

YOLOv3 came up with a better architecture where the feature extractor used was a hybrid of YOLOv2, Darknet-53 (53 convolutional layers), and Residual networks (ResNet).

**The most salient feature of v3 is that it makes detections at three different scales.**

 In YOLO v3, **the detection is done by applying 1 x 1 detection kernels on feature maps of three different sizes at three different places in the network.**

**Better at detecting smaller objects**

**Detections at different layers helps address the issue of detecting small objects, a frequent complaint with YOLO v2. The upsampled layers concatenated with the previous layers help preserve the fine grained features which help in detecting small objects.**

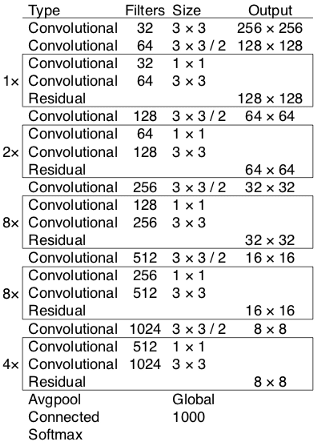
**The 13 x 13 layer is responsible for detecting large objects, whereas the 52 x 52 layer detects the smaller objects, with the 26 x 26 layer detecting medium objects.**

**Steps for object Detection using YOLO v3:**

* The input is a batch of images of shape (m, 416, 416, 3).
* YOLO v3 passes this image to a convolutional neural network (CNN).
* The last two dimensions of the above output are flattened to get an output volume of (19, 19, 425):
  + Here, each cell of a 19 x 19 grid returns 425 numbers.
  + 425 = 5 \* 85, where 5 is the number of anchor boxes per grid.
  + 85 = 5 + 80, where 5 is (pc, bx, by, bh, bw) and 80 is the number of classes we want to detect.
* The output is a list of bounding boxes along with the recognized classes. Each bounding box is represented by 6 numbers **(pc, bx, by, bh, bw, c)**. If we expand c into an 80-dimensional vector, each bounding box is represented by 85 numbers.
* Finally, we do the IoU (Intersection over Union) and Non-Max Suppression to avoid selecting overlapping boxes.

**architecture:**

* YOLO v3 uses a variant of Darknet, which originally has 53 layer network trained on Imagenet.
* For the task of detection, 53 more layers are stacked onto it, giving us a 106 layer fully convolutional underlying architecture for YOLO v3.
* In YOLO v3, the detection is done by applying 1 x 1 detection kernels on feature maps of three different sizes at three different places in the network.
* The shape of detection kernel is 1 x 1 x (B x (5 + C)). Here B is the number of bounding boxes a cell on the feature map can predict, '5' is for the 4 bounding box attributes and one object confidence and C is the no. of classes.
* YOLO v3 uses binary cross-entropy for calculating the classification loss for each label while object confidence and class predictions are predicted through logistic regression.



Darknet 53 architeture