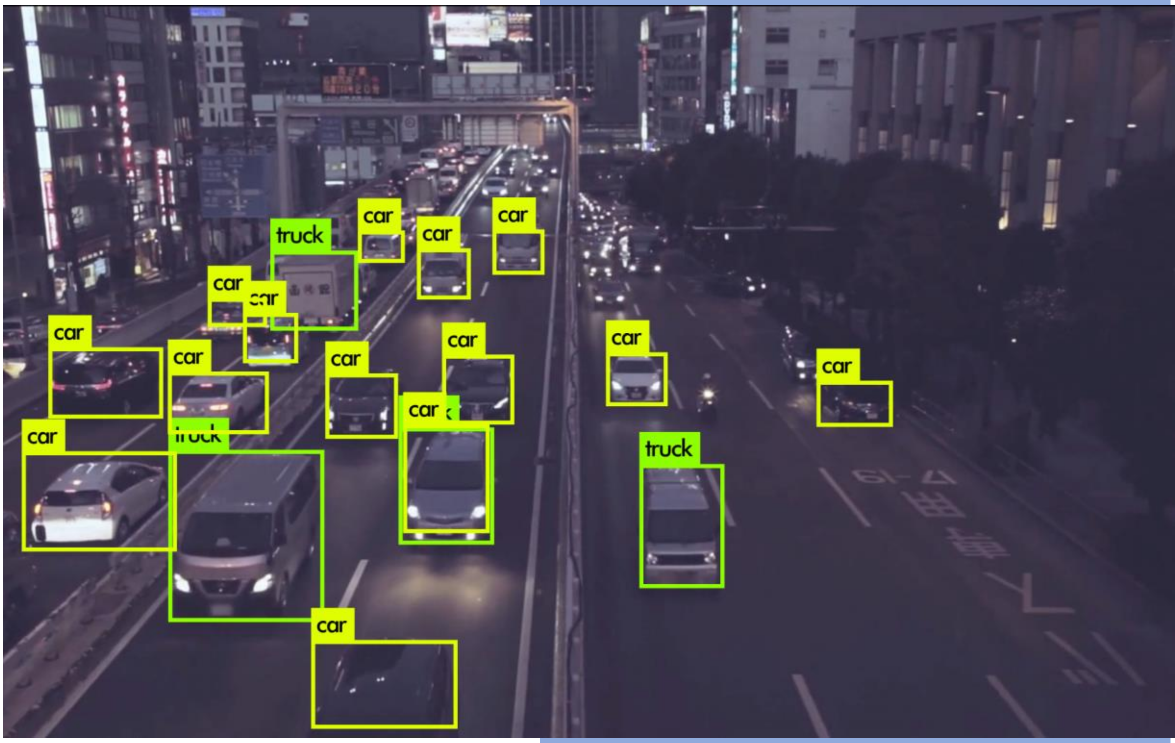
Object Detection- YOLOv3 and OpenCV



Harika Chowdary Digumarthi

Artificial Neural Networks

5/31/2020

Object Detection with YOLOv3 and OpenCV

Object Detection is an important task in Computer Vision and image processing that successfully localizes the objects to draw a bounding box around the object and correctly predict the class of the object.

YOLO – You Only Look Once:

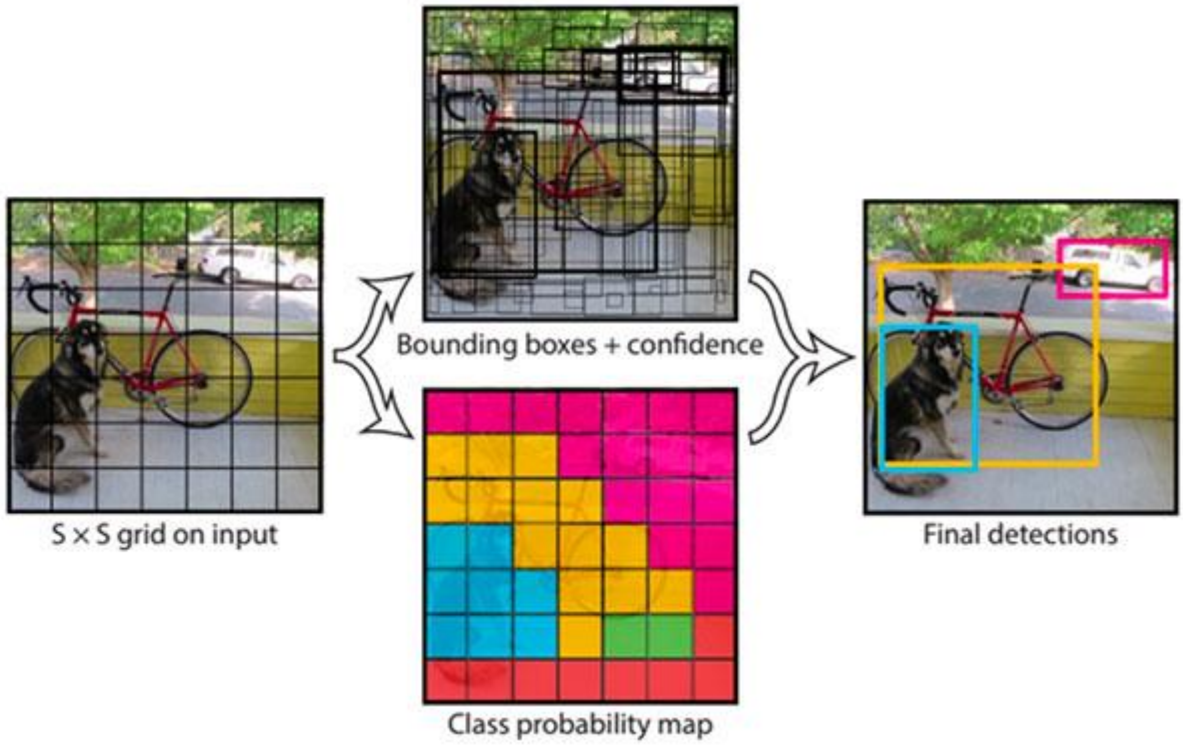
YOLO is a state-of-the-art object detection algorithm, which detects objects and also their locations in the image. YOLO uses convolution neural networks.

YOLO is very faster than R-CNN and Fast R-CNN, achieving object detection in real-time. The predictions are less accurate than R-CNN systems, but are way faster. It uses single layer deep convolution neural network.

Here, we use YOLO version 3, an improved version of its descendants in training and performance.

YOLOV3: Architecture:

Yolo is a single-stage detector. There is no sliding window. The prediction is done by 1x1 convolution. The input (image) is divided into a grid of cells and each cell directly predicts the bounding box involving the x, y coordinate and the width and height and probabilities for each cell for object detection and also class prediction. The resulting bounding boxes weighted by the predicted probabilities are consolidated into a final prediction by a post-processing step. The figure below shows the technique of yolo.



Unlike YoloV2, YoloV3 predicts the bounding boxes with logistic regression. There can be several bounding boxes around the object. A threshold can be defined to restrict the number of bounding boxes with less probability of predicting the object in it. This is known as Non-max Suppression.

The model involves several hypermeters, namely:

- Batch Normalization – which trains the model faster and reduces variance between units.
- Predefined Anchors – which are predefined bounding boxes with useful shapes and sizes that are tailored during training. The anchors are calculated for COCO dataset using k-means clustering. The centre coordinates of the box relative to the location of filter application are predicted using a sigmoid function.
- Activation Functions – Leaky ReLU is mostly used in Yolo, a slight modification of ReLU, which prevents “Dying ReLU” or “vanishing gradient problem”.
- Stride – which is a factor by which the network downsamples the input.

In Yolo, the output is a feature map. Hence, each grid cell or neuron is the dimension of the output. For e.g., if the input image is 416x416, and the stride of the network is 32; then the dimensions of the feature map will be 13x13 and also the image is divided into 13x13 cells.

Dataset: COCO

COCO (Common Objects in Context) is a large-scale object detection, segmentation and captioning dataset, widely available on http://cocodataset.org/#home. It has 80 different object categories, 91 stuff categories and over 330K images.

Here, pre-trained weights that were trained on the COCO dataset will be used.

Darknet:

YOLO uses Darknet as the neural network architecture. Darknet is composed of deep convolutional neural nets with shortcut layers, upsamplings and routes. A shortcut layer is a skip connection, which adds feature maps from the previous layer and the 3rd layer backwards to the shortcut layer to obtain the output. Upsampling upsamples the feature map in the previous layer by a factor of stride. The route layer outputs the feature map of the indicated layer backwards, along just length or along both length and depth. There is another layer called *yolo*, which is responsible for the object detection. It uses anchors to predict bounding boxes in a cell.

Object Detection and Conclusion:

The model detects the objects, along with the defined non-suppression and intersection over union threshold values.

Though YOLO is a state-of-the art algorithm, it has a main drawback of not able to detect the objects grouped together, especially for smaller ones. There are also losses in the algorithm which account for losing the object to be detected in the bounding box with poor confidence.

Github Repository: https://github.com/harika-digumarthi/ANN_Project