Deep Learning and Neural Networks

Demetrio Labate

March 25, 2024

Part 4
Object Recognition Networks

Visual object recognition

Visual **object recognition** or **object detection** refers to the ability to identify the objects in view based on visual input.

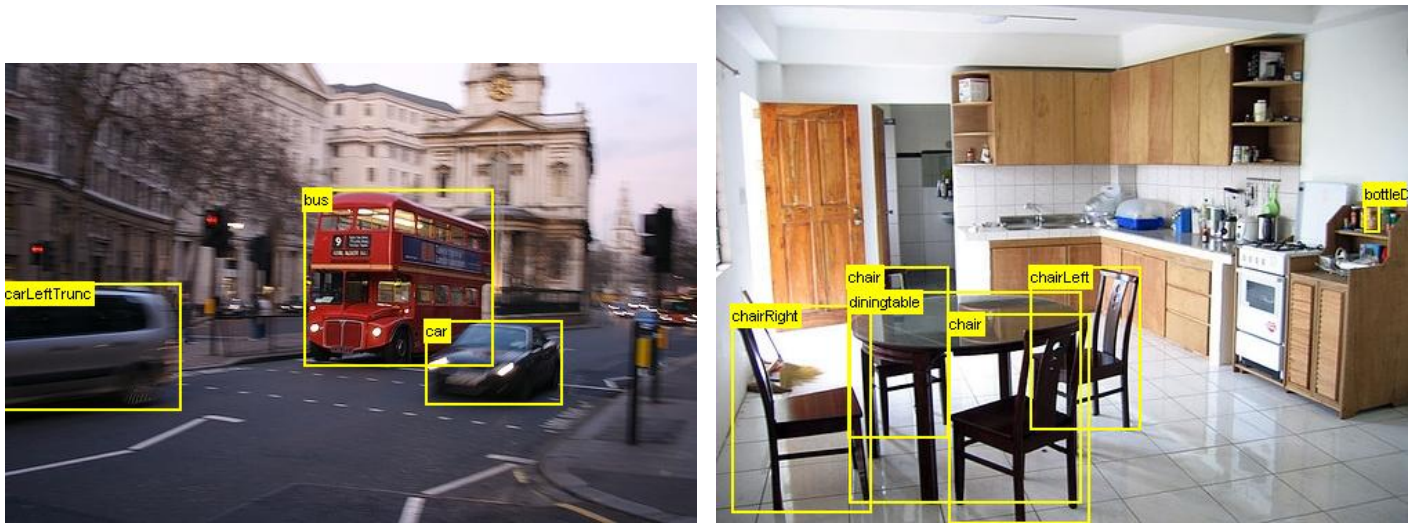


Figure: Images from the **PASCAL VOC** (Visual Object Classes) dataset

The primary aim of visual object recognition is *object invariance*, that is, the ability to identify objects across changes in the context in which objects are viewed, including changes in illumination, object pose, and background.

Visual object recognition

While CNNs displayed remarkable improvements in image classifications tasks during 2000-2014, performance on object detection and visual recognition tasks improved very slowly during the same time.

Key difference between object detection algorithms and classification algorithms:

- in detection algorithms, we try to draw a bounding box around the object of interest to locate it within the image;
- there could be many bounding boxes representing different objects of interest within the image and you would not know how many beforehand

Visual object recognition

Why not employing standard convolutional layers followed by a fully connected layer for object detection?

Main challenge is that the length of the output layer is variable as the number of occurrences of the objects of interest is not fixed.

A naive approach to solve this problem would be to take different regions of interest from the image, and use a CNN to classify the presence of the object within that region.

The problem with this approach is that the objects of interest might have different spatial locations within the image and different aspect ratios. Hence, you would have to select a huge number of regions and this would blow up the computational burden.

Region Based CNNs

Region Based CNNs (R-CNNs) were introduced to solve the problem of **visual object recognition** using deep learning architectures .

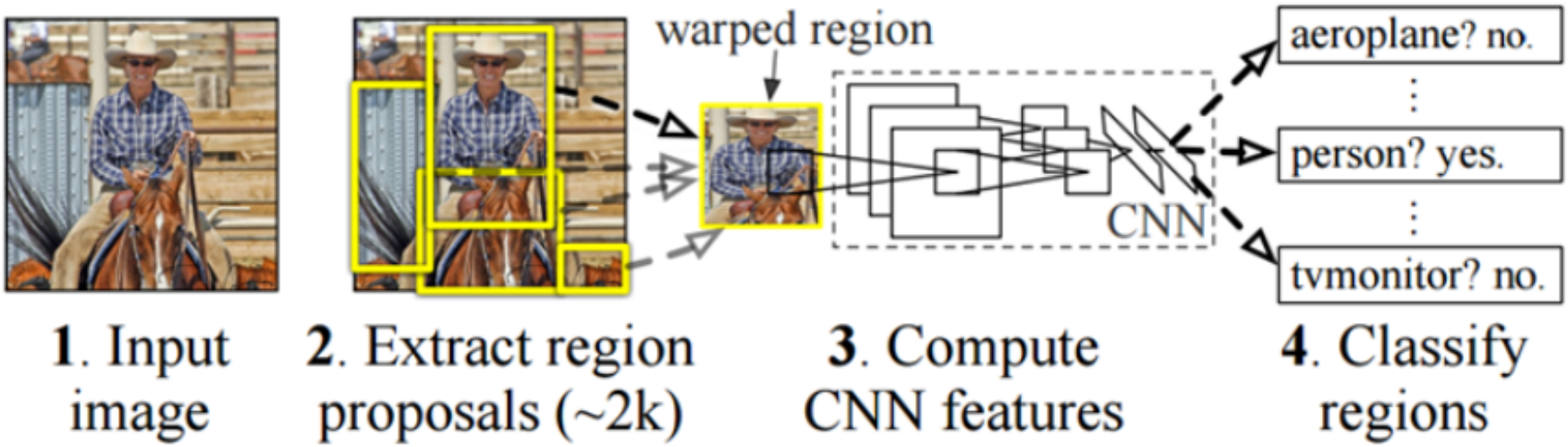
The original publications [Girshick, Donahue, Darrell, Malik, 2014] spurred very rapid advancements in computer vision and was followed by **Fast R-CNN** [Girshick, 2015] and **Faster R-CNN** [Ren, He, Girshick, Su, 2016] to make the model faster and better suited for modern object detection tasks.

The advent of R-CNN has been extremely impactful with the original paper being cited over 35,000 times.

Region Based CNNs

Given a certain image, we want to **draw bounding boxes over all of the objects.**

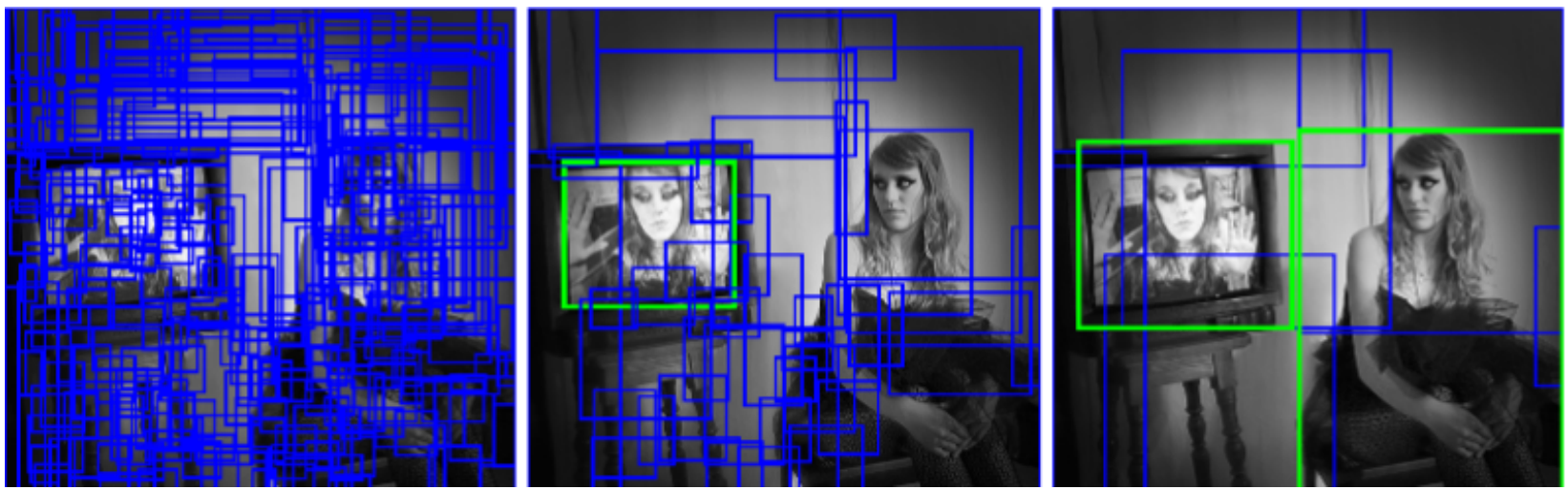
The process can be split into two general components: (i) the region proposal step and (ii) the classification step.



Region Based CNNs

To find regions of interest, the R-CNN algorith adopts a method called **Selective Search** (cf. Selective Search for Object Recognition by Uijlings et al, 2012).

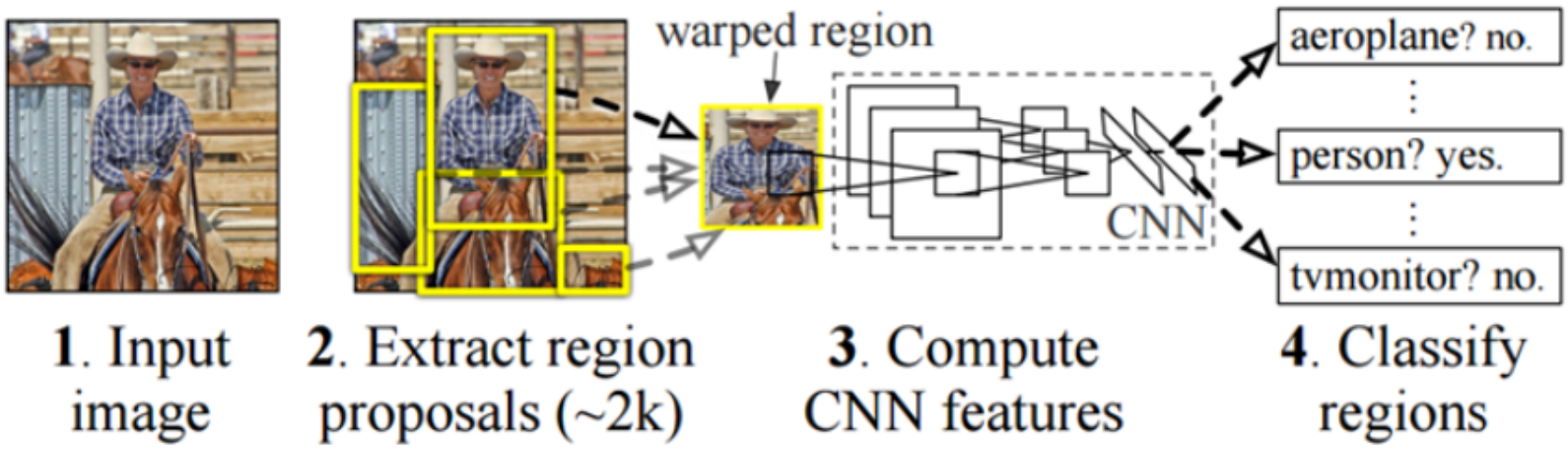
Selective Search performs the function of generating 2000 different regions over multiple scales and locations that have the highest probability of containing an object.



Selective Search generates regions of different scales to find objects that may have diferent sizes

Region Based CNNs

After the R-CNN algorithm has selected a set of region proposals, these proposals are "warped" into an image size that can be fed into a trained CNN (AlexNet) that extracts features vector for each region.



This vector is then used as the input to a set of linear SVMs that are trained for each class and output a classification. The vector also gets fed into a bounding box regressor to obtain the most accurate coordinates.

Non-maxima suppression is then used to suppress bounding boxes that have a significant overlap with each other.

Fast R-CNN

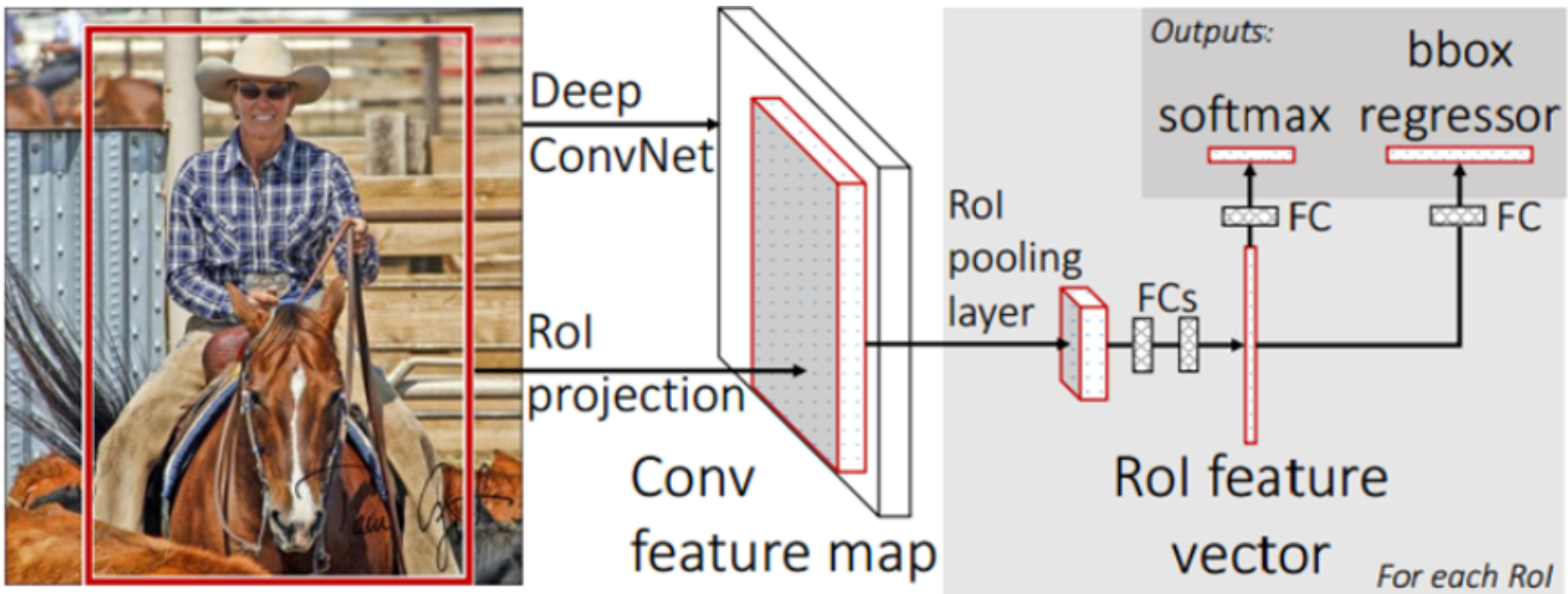
The original R-CNN approach has limitations:

- Training took multiple stages: ConvNets to SVMs to bounding box regressors.
- Method was computationally expensive.
- Processing was extremely slow: RCNN took 53 seconds per image.

Fast R-CNN was able to solve the problem of speed by basically sharing the computation cost of the convolutional layers between different proposals and by swapping the order of generating region proposals and running the CNN.

Fast R-CNN

In the Fast R-CNN model, instead of feeding the region proposals to the CNN, we (i) feed the input image to the CNN; next (ii) from the convolutional feature map, we identify the region of proposals and warp them into squares and (iii) by using a RoI pooling layer we reshape them into a fixed size so that it can be (iv) fed into a FC layer. From the RoI feature vector, (v) we use a softmax layer to predict the class of the proposed region and also the offset values for the bounding box.



Fast R-CNN

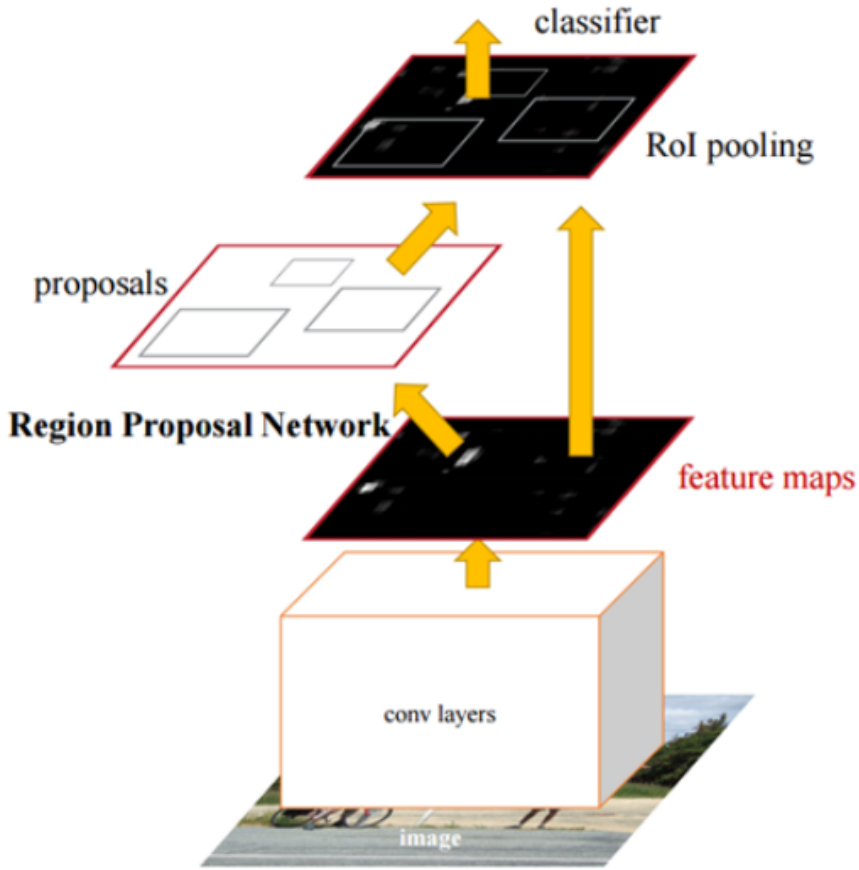
The reason Fast R-CNN is faster than R-CNN is because it does not feed 2000 region proposals to the CNN every time.
Instead, the convolution operation is done only once per image and a feature map is generated from it

Both of the R-CNN and Fast R-CNN uses selective Search to find out the region proposals.
Selective search is a slow and time-consuming process affecting the performance of the network. Faster R-CNN eliminates the selective search algorithm and lets the network learn the region proposals.

Faster R-CNN

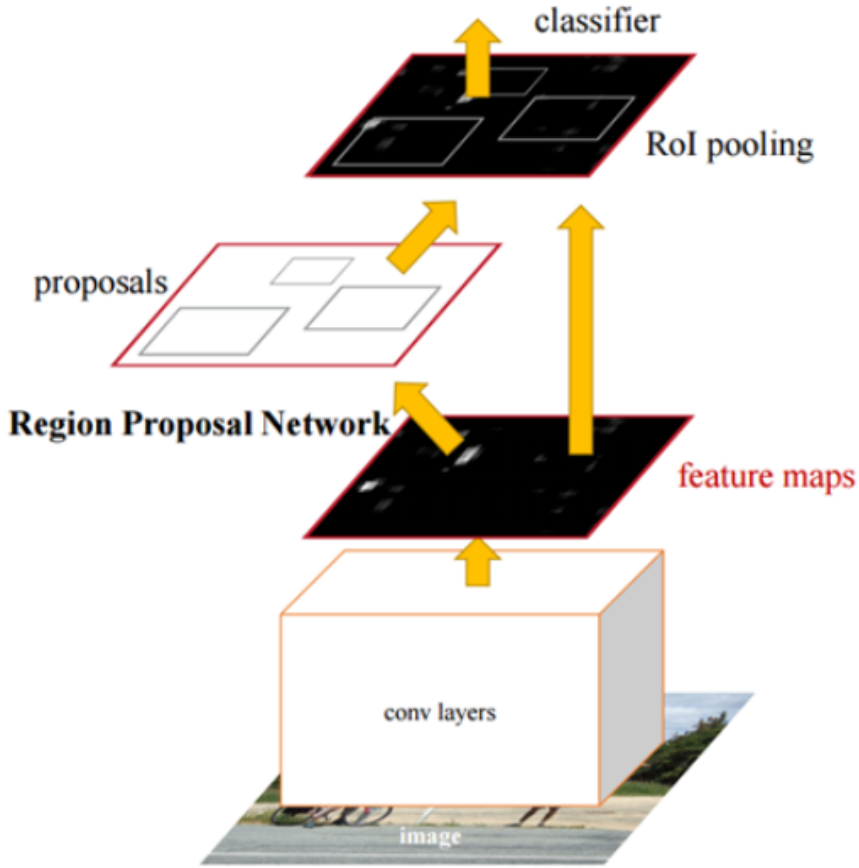
Faster R-CNN improve the somewhat complex training pipeline of both R-CNN and Fast R-CNN.

Instead of using Selective Search on the feature map to identify the region proposals, a separate network, the **Region Proposal Network (RPN)**, is used to predict the region proposals

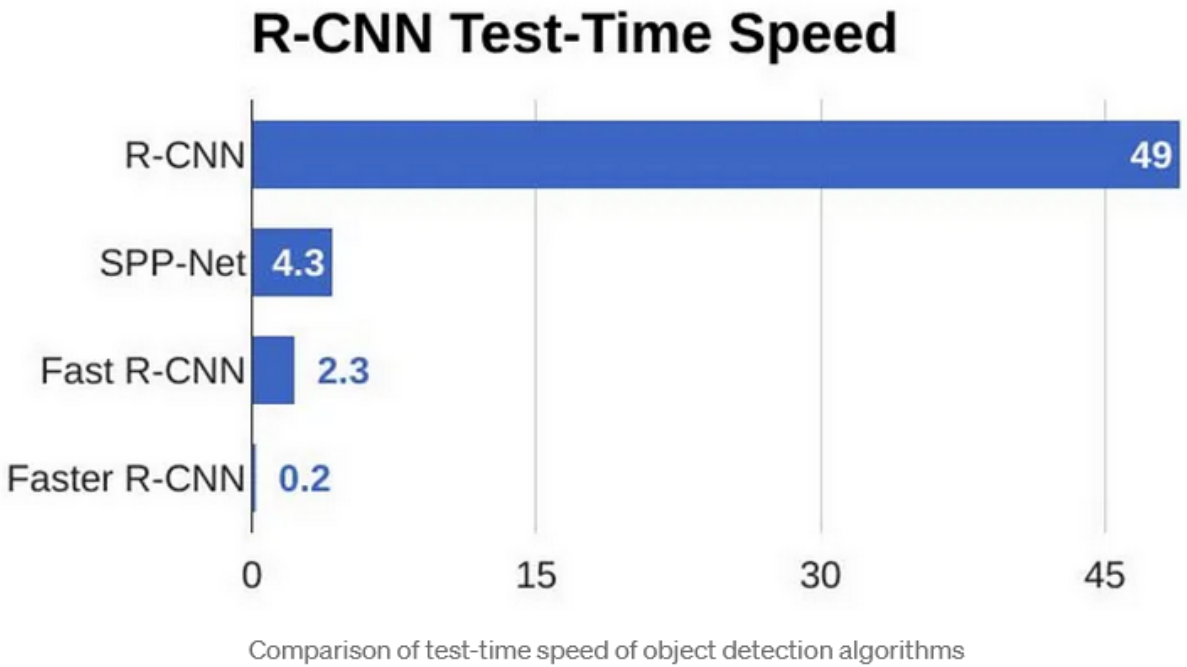


Faster R-CNN

After generating the region proposals, the rest of the Faster R-CNN pipeline is the same as the R-CNN. The predicted region proposals are reshaped using a RoI pooling layer which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes.



R-CNN, Fast R-CNN, Faster R-CNN



Comparison of test-time speed of object detection algorithms

Faster R-CNN became the standard approach for object detection about 2016.

YOLO

All object detection algorithms presented so far use regions to localize the object within the image. The network does not look at the complete image. Instead, parts of the image which have high probabilities of containing the object.

YOLO (You Only Look Once) [Redmon, Divvala, Girshick, Farhadi, 2016] introduced a one-stage approach that is very different from the two-stage region-based algorithms seen above.

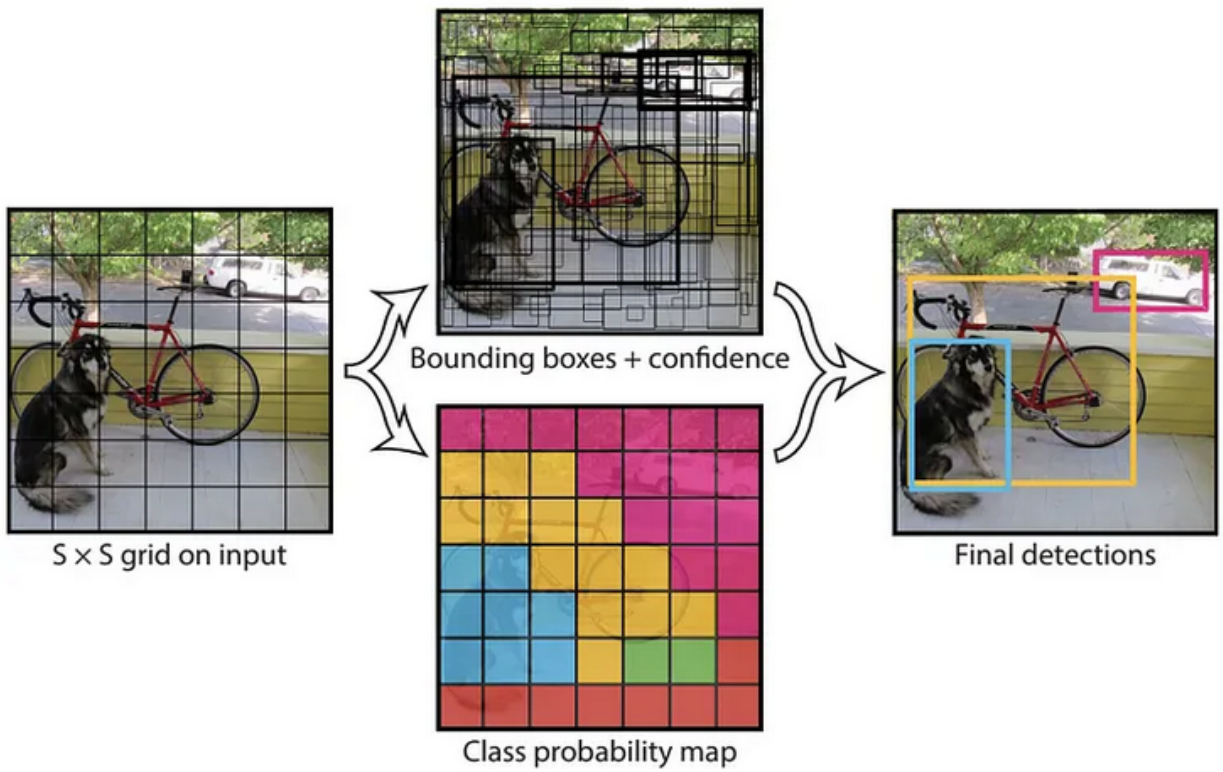
YOLO processes an image once through a CNN to simultaneously predict object positions and classes, transforming the object detection task into an end-to-end regression problem.

The YOLO team (and other contributors) implemented various optimizations into the YOLO algorithm, culminating in the development of eight upgraded versions so far: YOLOv2 through YOLOv9 (released in 2024).

YOLO

Here is the (first) YOLO workflow:

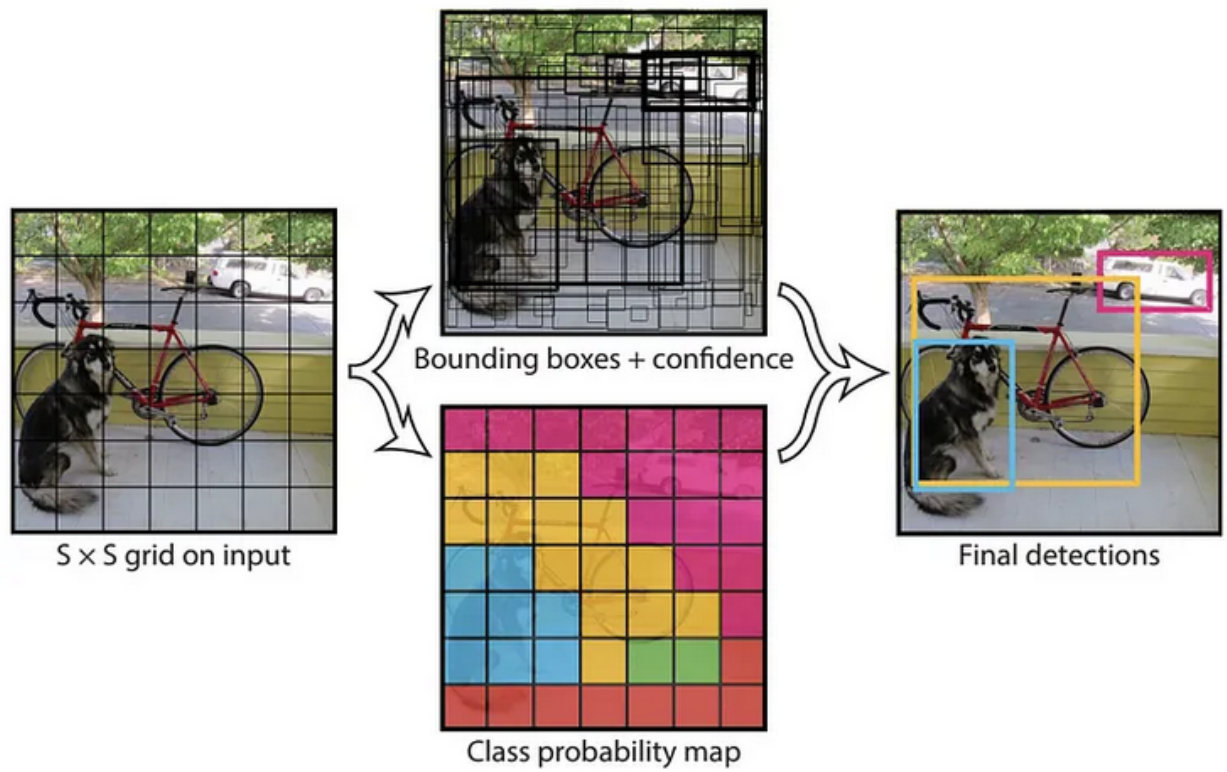
1. resize the input image to a standard size and overlay a grid on it;
2. within each of the grid we take m bounding boxes;



YOLO

YOLO workflow (continue):

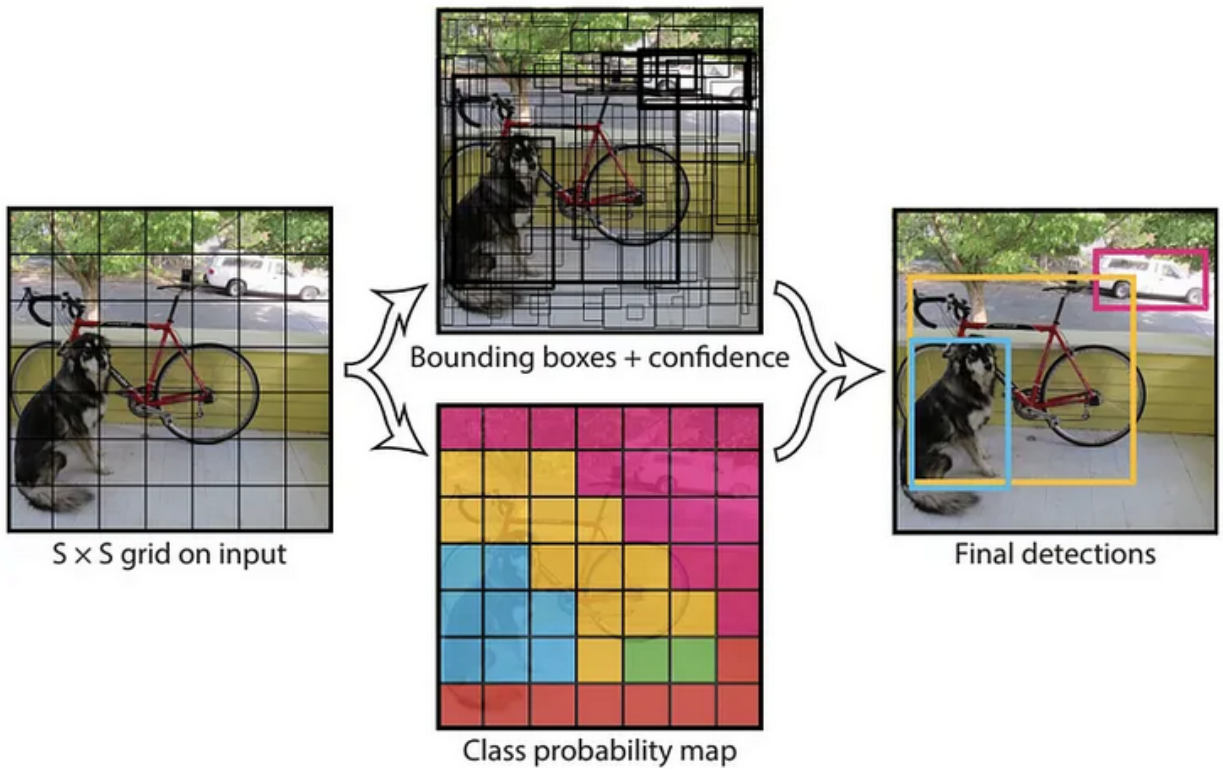
3 extract image features using a CNN and use the grid cells to perform regression predictions with fully connected layers. Each grid cell predicts K boxes, each with five regression values. Four of these values represent the box's position, and the fifth indicates the confidence of the box containing an object and the accuracy of its position;



YOLO

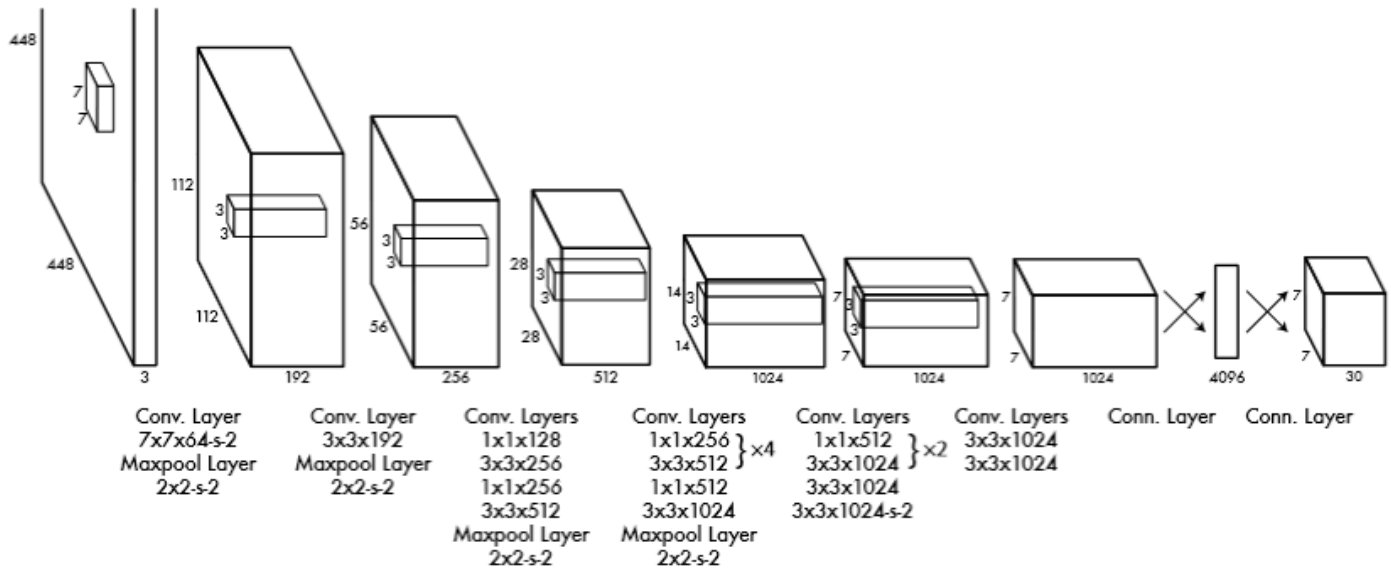
YOLO workflow (continue):

4 bounding boxes having the class probability above a threshold value are selected and used to locate the object within the image.



YOLO

The network architecture of (original) YOLO was inspired by the GoogLeNet.



The network has 24 convolutional layers followed by 2 fully connected layers. Instead of the inception modules used by GoogLeNet, it uses a 1×1 reduction layers followed by 3×3 convolutional layers

YOLO

YOLO is orders of magnitude faster than the R-CNN object detection algorithm and their faster versions, being able to process 45 frames per second.

As a result, it can be employed for **real-time object detection applications** such as Identifying intruders in a building, monitoring vehicle movement at a location, Understanding traffic patterns on a road.

Nonetheless, the (original) YOLO is limited by a coarse grid division can hinder its performance in detecting small objects. So, its detection performance is not as good as Faster R-CNN

YOLO - Subsequent versions

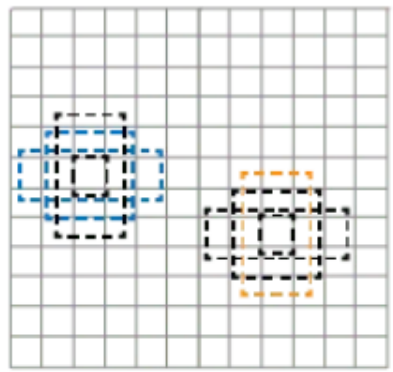
The YOLO team implemented various improvements into the YOLO algorithm.

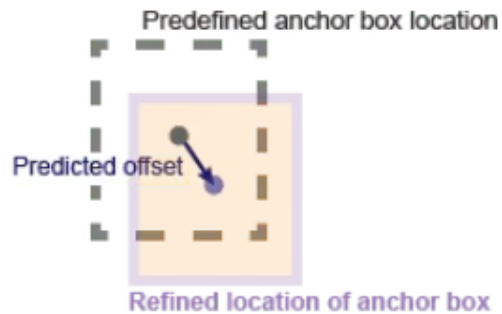
- **YOLOv2** introduced the employment of Batch Normalization, high-resolution image classification, the use of anchor boxes, clustering for extracting scale information, constraints on predicted bounding box positions, passthrough layers for detecting fine-grained features, and hierarchical classification.
- **YOLOv3** integrated ideas from SSD (another object detection algorithm), utilizing multi-scale features for object detection. It also added an objectness score to bounding box prediction and made predictions at three separate levels of granularity to improve performance on smaller objects.

YOLO - Subsequent versions

Anchor boxes are pre-defined boxes with specific sizes, aspect ratios, and positions that are used as reference templates during object detection.
These anchor boxes are placed at various positions across an image, often in a grid-like pattern, to capture objects of different scales and shapes.


Ground truth image and bounding boxes


Anchor boxes at each predefined location in each feature map



During training and inference, anchor boxes are used to predict the locations and shapes of objects relative to these reference boxes.
[video]

Object Detection Algorithms

Current state-of-the-art object detection algorithms include:

- **YOLO v8, v9** exploit YOLO one-stage approach, which ensure fast processing times, combined with several improvements to provide accurate detection for smaller objects.
- **RetinaNet** [Lin, Goyal, Girshick, He, Dollaris, 2017] a single-stage detector that uses Focal Loss to handle the issue of class imbalance, which is a key challenge in object detection tasks. RetinaNet also uses a Feature Pyramid Network as its backbone to efficiently detect objects at different scales and the classification subnet to classify the objects.
- **DETR (DEtection TRansformer)** [Carion, Massa, Synnaeve, Usunier, Kirillov, Zagoruyko 2020] uses CNN to extract image features and adds position encoding, a technique used in natural language processing, to generate a serialized set of data. It abandons the use of anchor boxes, region proposals, and non-maximum suppression. Instead, it treats object detection as a direct set prediction problem, eliminating post-processing steps such as anchor adjustment. [YOLO vs DETR review]