Some notes on Region-based CNNs

by
Dr. Maryam Khanian

R-CNN (using CNNs to localize objects in regions proposed by SS)

Fast R-CNN (using deeper CNNs and RoI pooling layer mechanism to classify objects in proposed regions by SS)

Faster R-CNN (proposing an RPN instead of other methods (e.g. SS) to extract region proposal using conv features shared with detection network that includes RoI pooling layers)

Note:

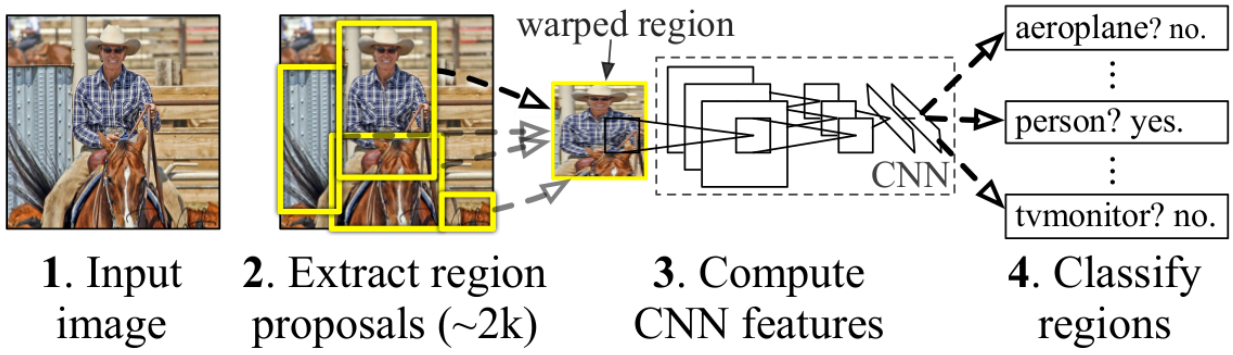
Bad Sequence:

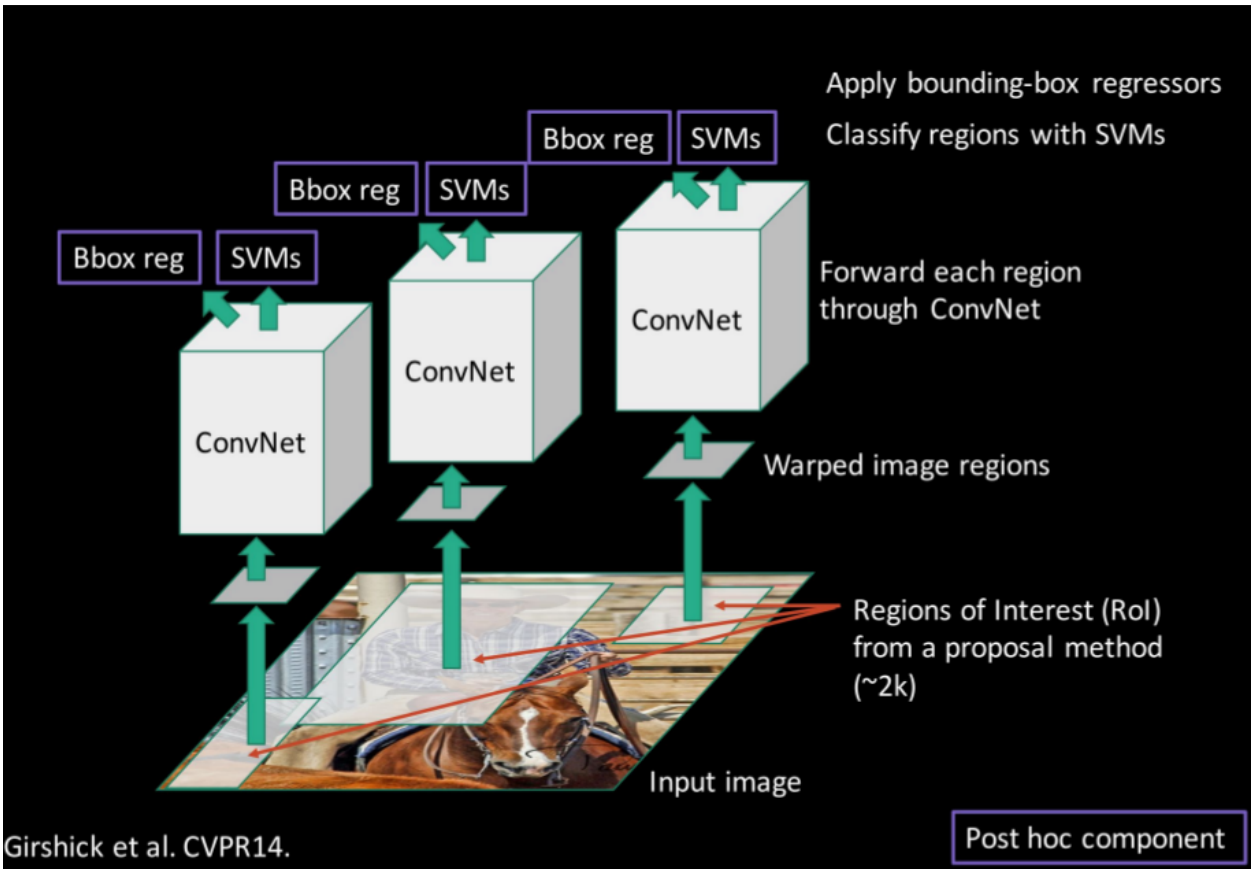
1. Region extraction
2. Generating feature maps for each region using a CNN
   (one forward pass through CNN for each region, expensive)

Good Sequence:

1. Generate an overall feature map for all the image input using a CNN
2. Use an extra fully convolutional network (FCN) to generate region proposals (coordinates & scores)

R-CNN (using CNNs to localize objects in regions proposed by SS)





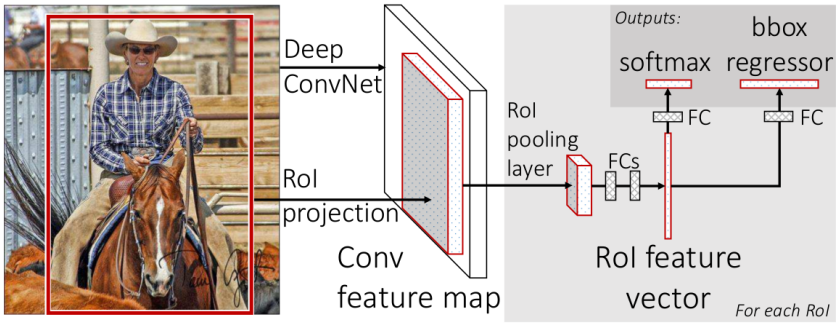
It has three main steps:

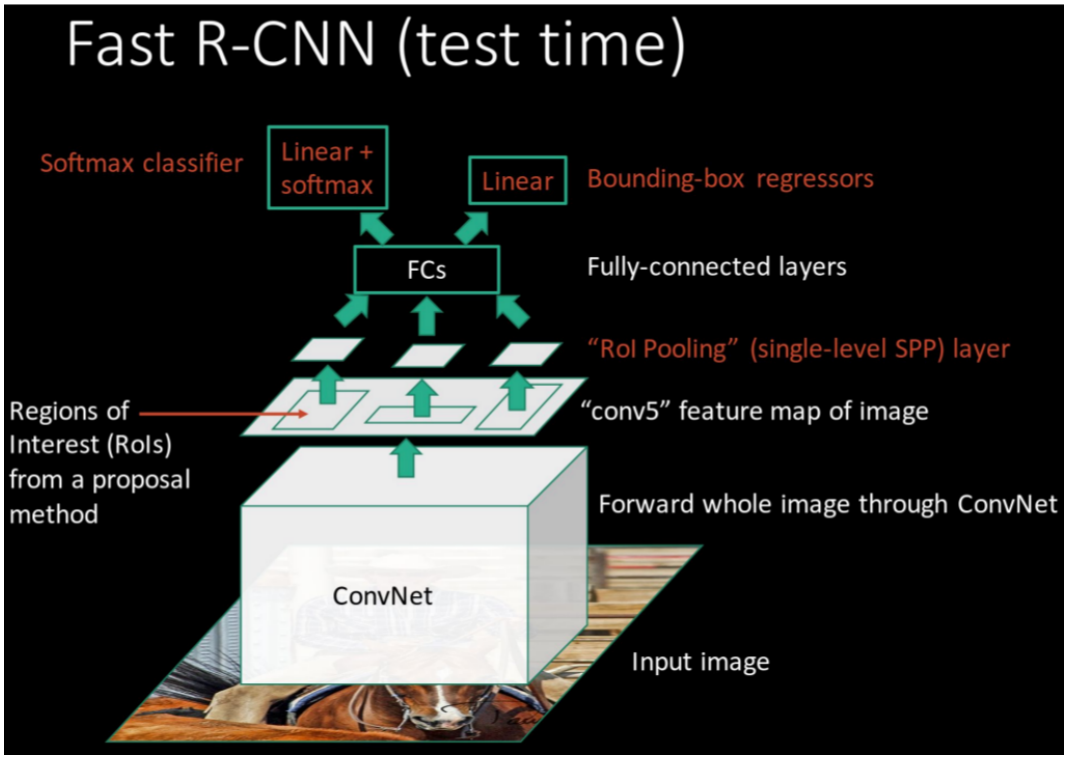
1. Extraction of region proposals using selective search.
2. Using a ConvNet to compute features for each region by forward propagating a mean-subtracted 227 × 227 RGB image through it.
3. Using a SVM classifier to classify each region.

Drawbacks:

1. Multi-stage training pipeline.
    a. Fine-tuning the CNN on object proposal using a log loss.
    b. Fitting SVMs to Convnet features.
    c. Learning bounding-box regressors.
2. Expensive training w.r.t. required time and space (2.5 days and hundreds of gigabytes).
3. Slow object detection.(47s/image)

Fast R-CNN (using deeper CNNs and RoI pooling layer mechanism to classify objects in proposed regions by SS)





It this architecture, an *R-CNN network* (actually its convolution and pooling layers before its FC layers except the last pooling layer) is used to generate region proposals. Then, for each region proposal an *RoI max pooling layer* divides h x w RoI window (region) into an H x W grid of sub-windows of approximate size of h/H x w/W and then max pools the values in each sub-window into the corresponding output *grid cell* which is a *fixed-length feature vector.* The result is then fed into a sequence of FC layers that finally branch into two siblings output layers:

I. One that produces softmax possibility estimates over *K* classes plus a catch-all “background” class
II. Another layer that outputs four real-valued numbers (bounding-box offsets) for each of the *K* object classes.

Advantages:

One fine-tuning stage jointly optimizes a softmax classifier and bounding-box regressors because:

a. It has a Multi-task loss that is a combination of :
    i. Log loss (L_{cls})
    ii. Smooth *L1* loss (L_{loc})
b. It uses mini-batch sampling.
    i. Mini-batch size: 128
    ii. Number of RoIs in each image: 64
    iii. The only data augmentation is horizontally flipping with probability 0.5.
c. It can propagate backward through RoI pooling Layer by computing partial derivatives of the loss function w.r.t. Each input variable.
d. It uses SGD hyper parameters:
    i. softmax classification and bounding-box regression are initialized from zero-mean Gaussian distributions with standard deviations 0.01 and 0.001, respectively.
    ii. Biases are initialized to 0.
    iii. All layers use a per-layer learning rate of 1 for weights and 2 for biases and a global learning rate of 0.001.
    iv. Momentum: 0.9
    v. Weight decay: 0.0005
    vi. When training on VOC07 or VOC12 trainval we run SGD for 30k mini-batch iterations, and then lower the learning rate to 0.0001 and train for another 10k iterations.

Authors used multi-scale training (using image pyramids) to make the system *scale invariant.*

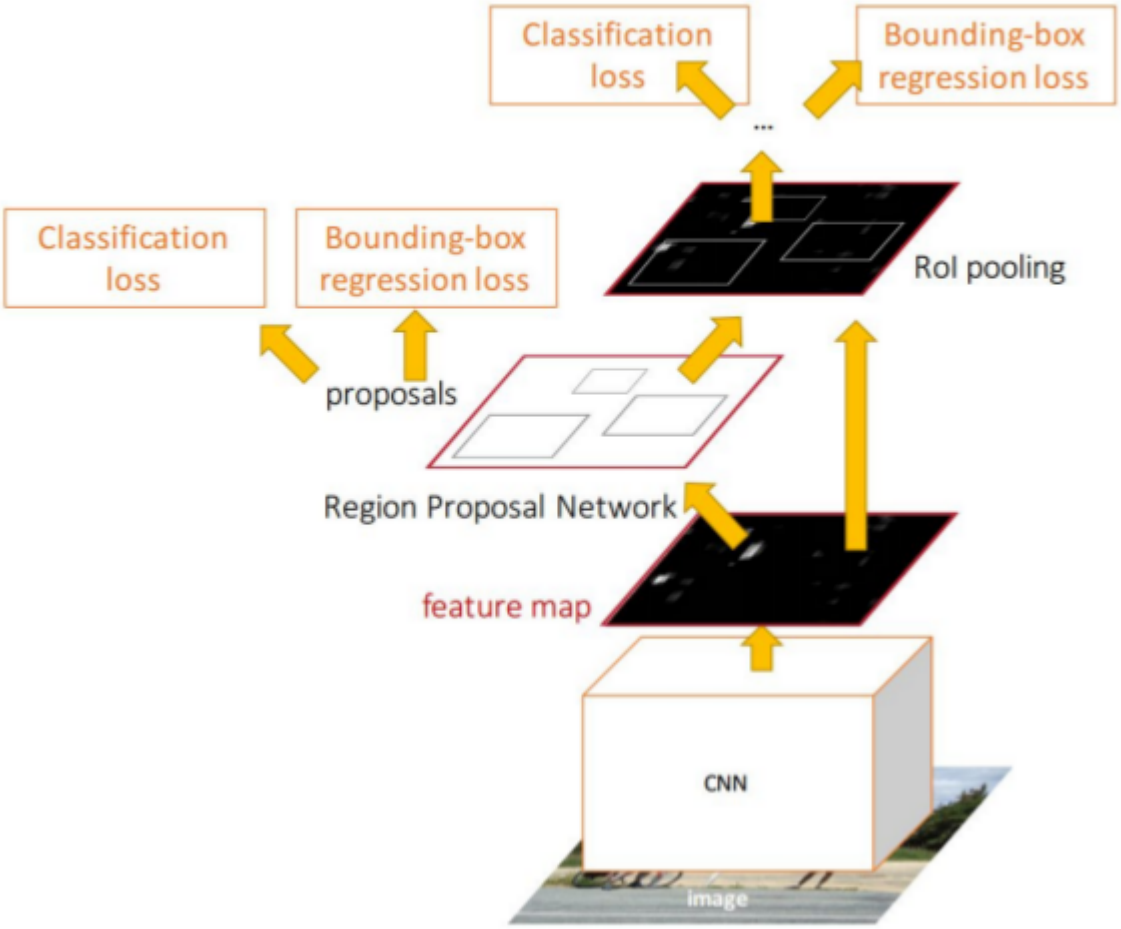
Experimental Setup:

Three different Caffe CNN architectures are used in this paper:

1. CaffeNet (AlexNet) (small=S)

2. VGG_CNN_M_1024 (Medium=M)
3. VGG16 (Large=L)

Faster R-CNN (proposing an RPN instead of other methods (e.g. SS) to extract region proposal using conv features shared with detection network that includes RoI pooling layers)



Region proposal computation was a bottleneck in detection networks, that was implemented using methods such as *Selective Search* (SS - greedy merge of superpixels based on engineered low-level features), so authors

have proposed a Region Proposal Network (RPN) that shares full-image convolution features with the detection networks using a *4-step training algorithm*:

Step 1:
Training the RPN. (Faster R-CNN paper, section 3)
This network is initialized with an ImageNet pre-trained model and fine-tuned end-to-end for the region proposal task.

Step 2:
Train a separate detection network by Fast R-CNN using the proposals generated by the step-1 RPN. This detection network is also initialized by the ImageNet-pre-trained model.
(At this point the two networks do not share conv layers.)

Step 3:
using the detector network to initialize RPN training, but fixing the shared conv layers and only fine-tune the layers unique to RPN.
(Now the two networks share conv layers.)

Step 4:
keeping the shared conv layers fixed and fine-tune the FC layers of the Fast R-CNN.

RPN is a fully-convolutional network that simultaneously predicts object bounds and objectness scores at each position of the input feature map. Note that each of these positions, has a large effective receptive field on the input image.

This network is fully connected to an n × n spatial window of the input conv feature map. Each sliding window is mapped to a lower-dimensional vector (256-d for ZF and 512-d for VGG). This vector is fed into two sibling fully-connected layers:

1. a box-regression layer (reg)
2. a box-classification layer (cls)

This architecture is naturally implemented with an n × n conv layer followed by two sibling 1 × 1 conv layers (for reg and cls, respectively) where ReLU nonlinearities are applied to the output of the n × n conv layer.

For training RPNs, authors assign a binary class label (of being an object or not) to each anchor.

Then they assign a positive label to two kinds of anchors:

1. The anchor/anchors with the highest Intersection over Union (IoU) overlap with a ground-truth box.
2. II. An anchor that has an IoU overlap higher than 0.7 with any ground-truth box.

They assign a negative label to a non-positive anchor if its IoU ratio is lower than 0.3 for all ground-truth boxes.

Anchors that are neither positive nor negative do not contribute to the training objective.

With these definitions, they minimize an objective function following the multi-task loss in Fast R-CNN:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*).$$

Sampling:
At training time, they sample a minibatch containing positive and negative regions.

Since, some RPN proposals highly overlap with each other. To reduce redundancy, authors adopt non-maximum suppression (NMS) on the proposal regions based on their *cls* scores. This sampling technique is also used in DenseCap at test time.