Object Detection

Ujjwal

Post-Doc, STARS Team

INRIA Sophia Antipolis

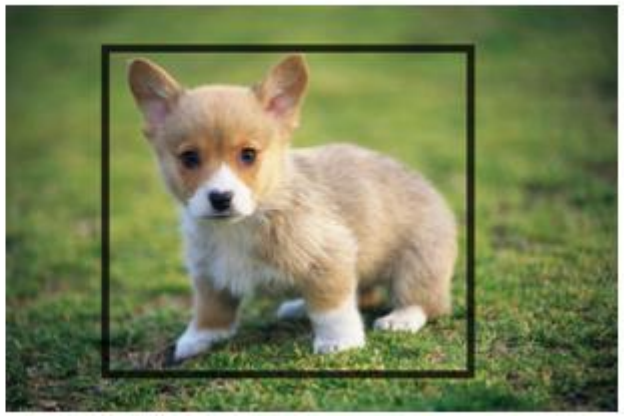
Outline

- What is Object Detection ?
  - Qualitative Definition.
  - Machine Learning Definition.
- Ingredients of Object Detection.
- Components of a typical deep learning object detector.
- Faster-RCNN.

Object Detection: Qualitative Discussion

Classification

There is a dog.



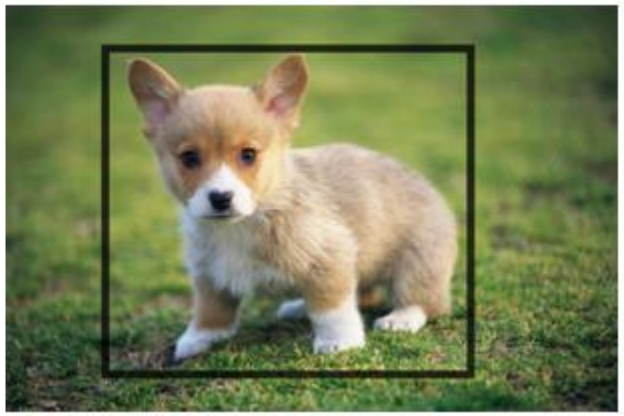
Detection

There is a dog with a bounding box around it.

Object Detection: Qualitative Discussion

Classification

There is a dog.



Detection

There is a dog with a bounding box around it.

Object Detection = Classification + Localization

Object Detection: Machine Learning Terms

Classification (N classes)

$$f: X \rightarrow Y$$

- $\boldsymbol{f}$: Mapping.
- $\boldsymbol{X}$: Set of training images.
- $\boldsymbol{Y}$: Set of labels $\mathbb{R}^N$.

Detection (N classes)

$$f: X \rightarrow Y$$

- $\boldsymbol{f}$: Mapping.
- $\boldsymbol{X}$: Set of training images.
- $\boldsymbol{Y}$: Cartesian product ($\mathbb{R}^N$, $\mathbb{R}^4$).
  - First element is object label.
  - Second element is object bounding box.

Ingredients of Object Detection

- Data.
- Base Network/Backbone.
- Detection Components.
- Loss functions.
- Pre-Processing.
- Post-Processing.

Data

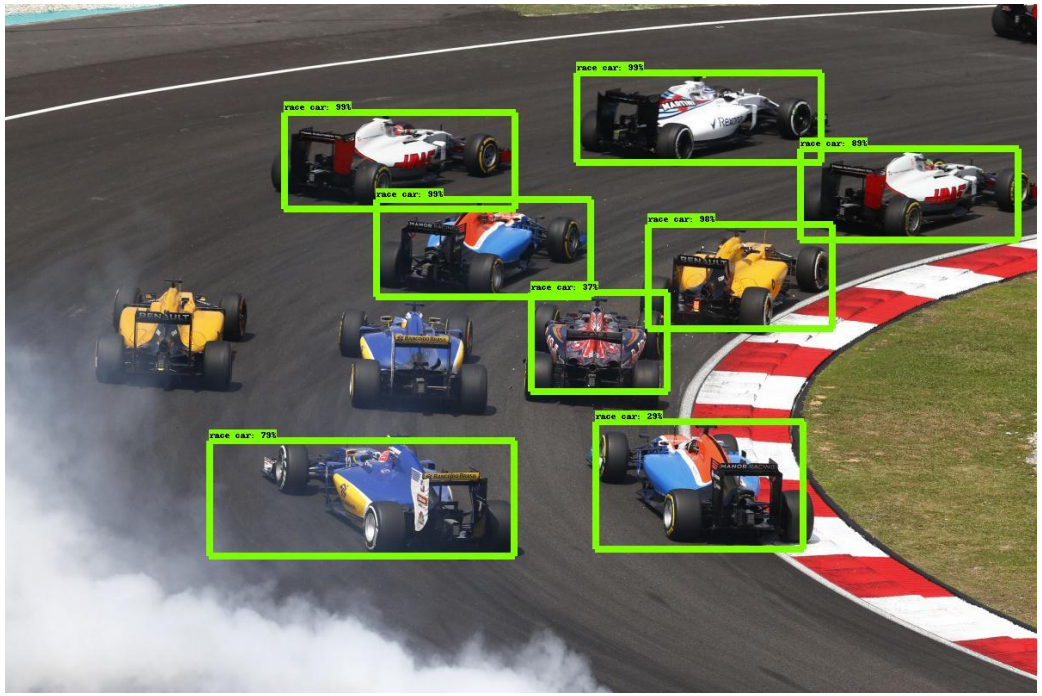
- Data could be:
  - Fully labeled (Fully-Supervised Detection)
  - Partially labeled (Semi-Supervised Detection)
  - Indirectly labeled (Weakly-Supervised Detection)

Data: Fully labeled

- All instances of all object classes are labeled in all images, if present.
- Good amount of supervision with a lot of information.
- Most popular public datasets are fully-labeled
  - Pascal VOC
  - INRIA Pedestrians.
  - Caltech Pedestrians.
  - MSCOCO.
  - Objects-365

Data: Partially labeled

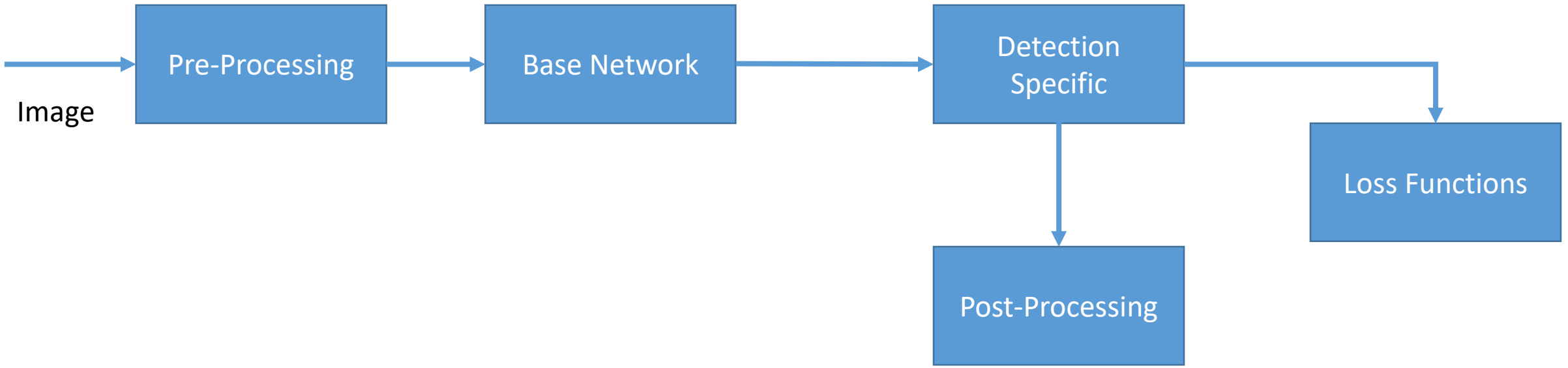
- Only some instances of objects of interest are labeled.
- Supervision is present but only partial.
- OpenImages is a major partially labeled dataset.



Data: Indirectly labeled

- Labelling is in some other form. Example is below:
  - Given an image, it is told that there are people and cars in there.
  - It is not told as to where they are.
- Thus a very weak form of supervision is provided.
- There is no dataset for weakly supervised detection.
- This is an advanced subject and will not be considered here.

How an Object Detector looks like ?



Pre-Processing

- Pre-Processing is needed for:
  - Rescaling the pixel range from [0,255] to [0,1] or [-1,1].
  - Perform data augmentation.

Data Augmentation for Object Detection

- Randomly change contrast, brightness, colors.
- Randomly horizontal or vertical flipping of the image.
- Random rotation of the image.
- Randomly distort bounding boxes.
- Randomly translate an image.
- Randomly add black patches.

Base Network/BackBone

- A base network is essentially any CNN architecture without its fully connected layers.
- It is responsible to perform initial feature extraction from images.
  - A better feature extraction leads to better detection.
- “The CNN backbone is the most important part of a detection framework.”

  - Ross Girschik (Author of Faster-RCNN and Mask-RCNN)

Base Network/BackBone

- Common base networks:
  - VGG16 (Not used anymore)
  - ResNet Family of networks
    - ResNet-50
    - ResNet-101
    - ResNet-152
  - Inception Family of networks
    - InceptionV1
    - InceptionV2
    - InceptionV3
    - InceptionV4
  - InceptionResNet
  - ResNeXt-50,101,152

BackBone: What is important ?

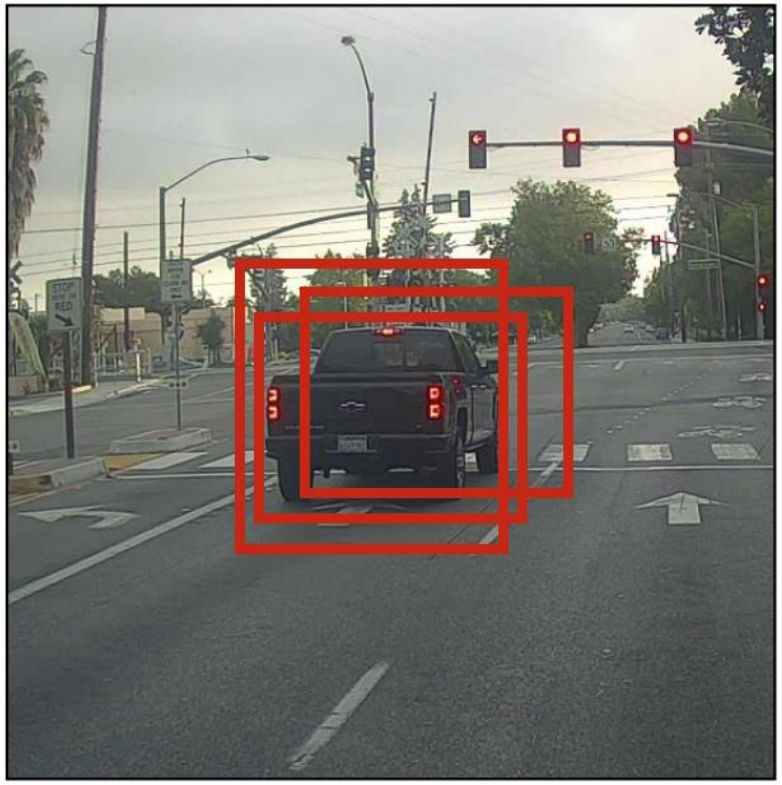
- How big is the backbone?
  - Too small means not suitable for feature extraction.
  - Too big means it might not fit in a limited GPU memory.
- What are its salient characteristics?
  - Is it good for multi-scale ?
- What is it trained on ?
  - Usually for images, we prefer using a pre-trained network.
  - Pre-training is usually preferred on imagenet dataset.

Detection Specific Components

- These components vary with techniques (eg: SSD, Faster-RCNN)
- Some components are omnipresent
  - Bounding box classifier.
  - Bounding box regressor.

Post-Processing

Before non-max suppression



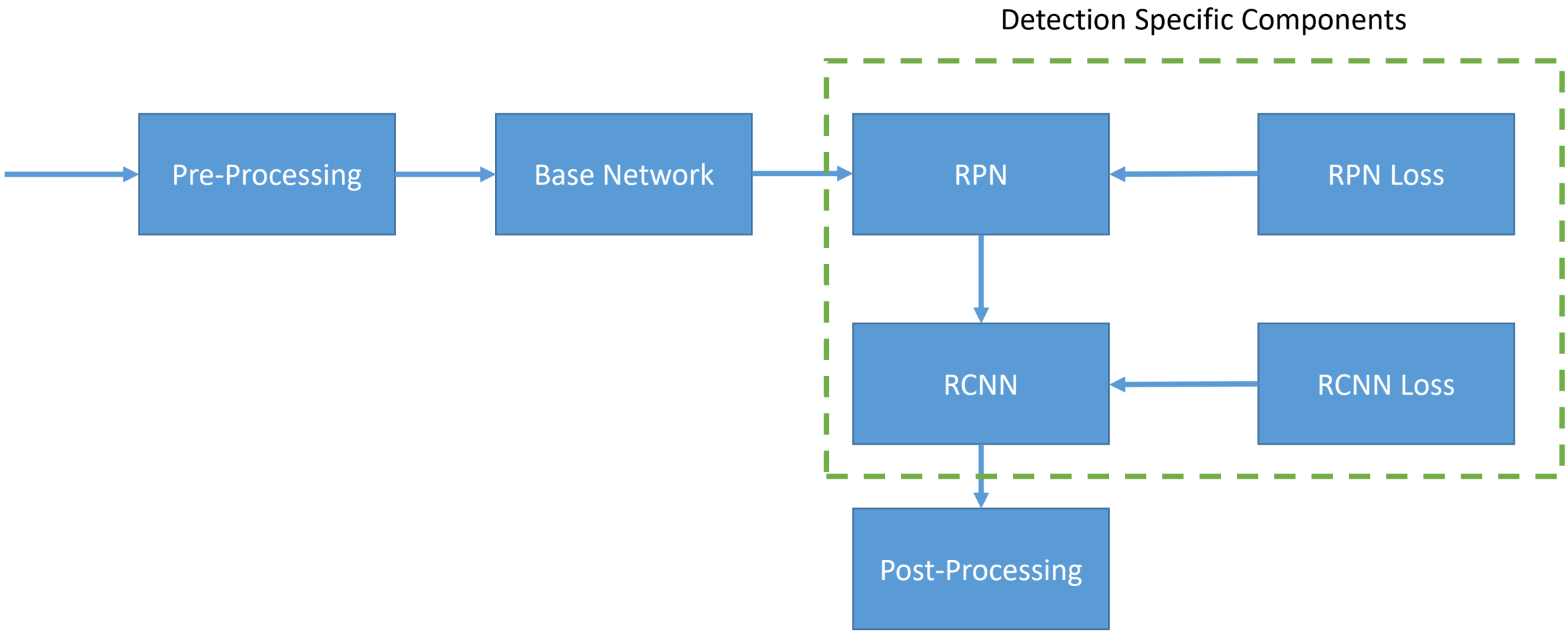
Non-Max Suppression

After non-max suppression

Loss Function

- Two loss functions are primarily used in object detection
  - Classification Loss : Which object is present in a given bounding box.
  - Localization Loss: How good is that bounding box.

Faster-RCNN


Detection Specific Components
Pre-Processing
Base Network
RPN
RPN Loss
RCNN
RCNN Loss
Post-Processing

Before RPN: The basic challenge of detection

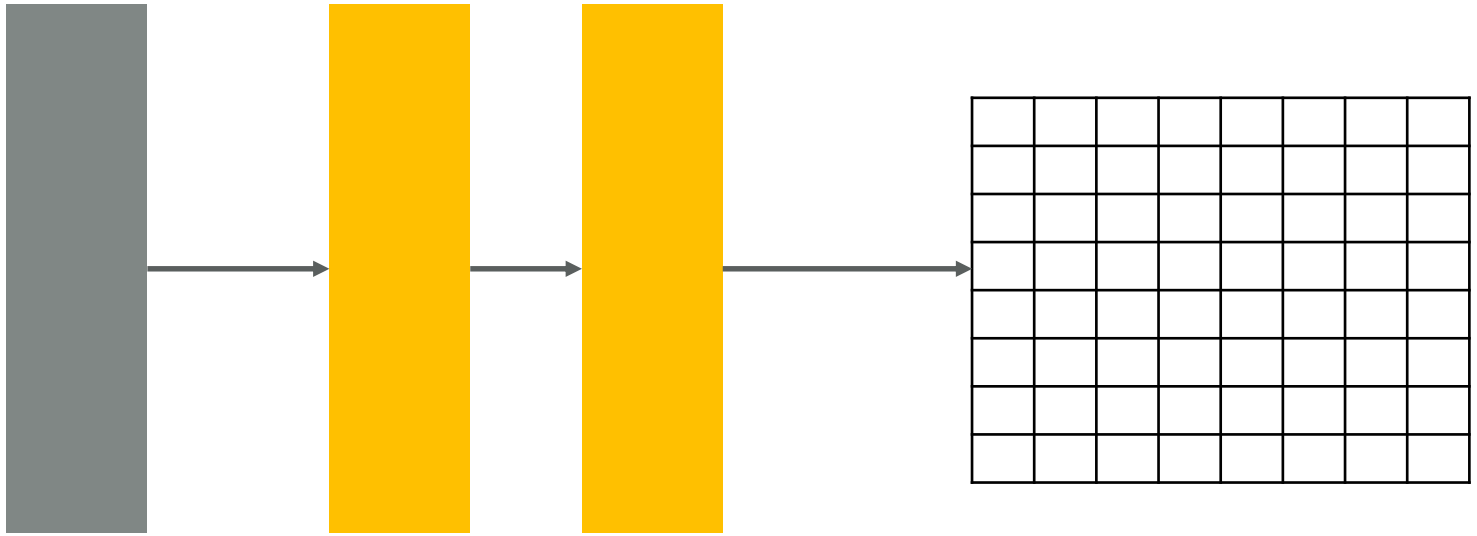
- Processing all possible regions of an image is computationally intractable.
- Therefore, RPN is a tool to reduce the number of regions in an image which need be processed.

RPN: Region Proposal Network

Original Image

RPN Output: Proposals

Region Proposal Network: Step 1

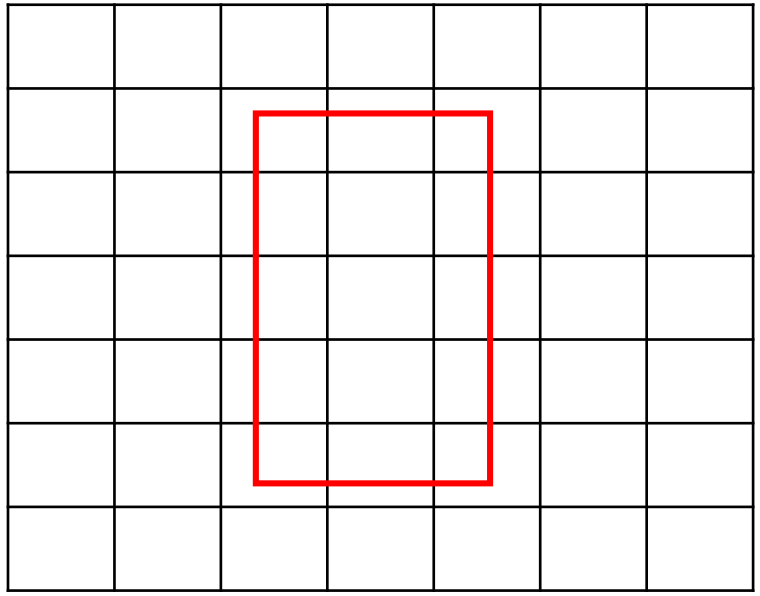


Base Network

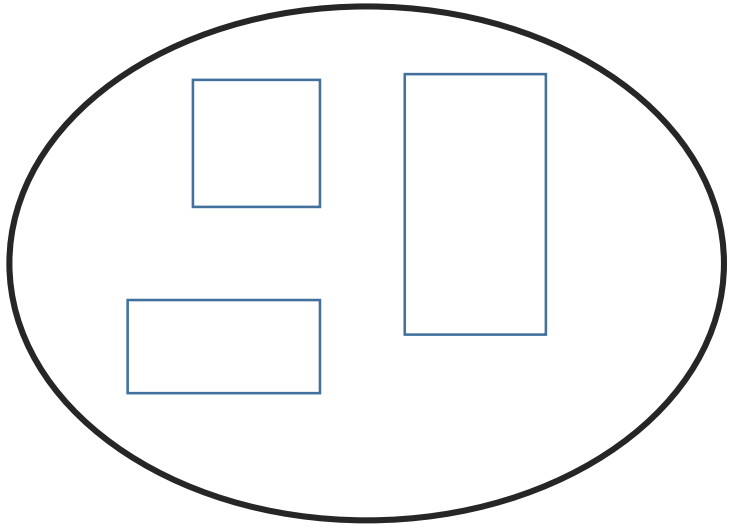
Extra Convolutional Layers (Optional and must be decided by experimentation)

Feature Map

Region Proposal Network: Step 2



Feature Map with object bounding box $\boldsymbol{p_i^*}$

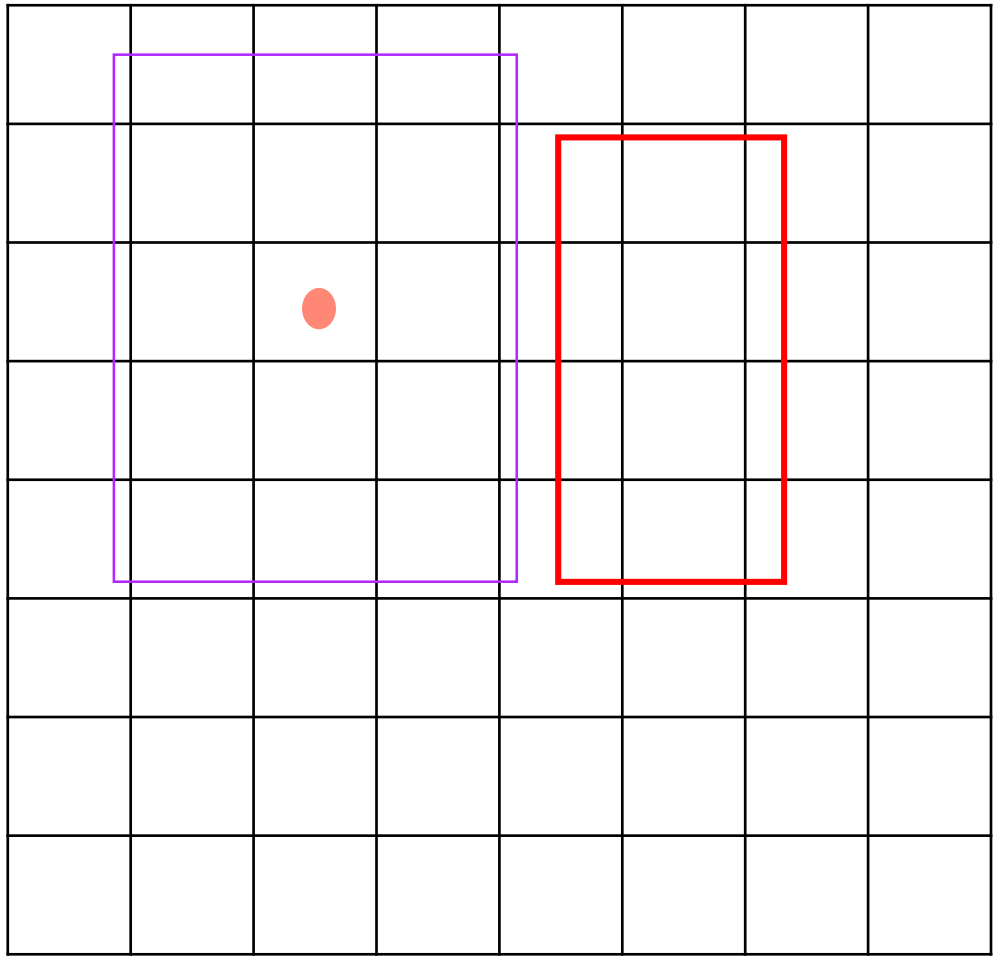


Pool of predefined anchors

$\boldsymbol{p_i}$

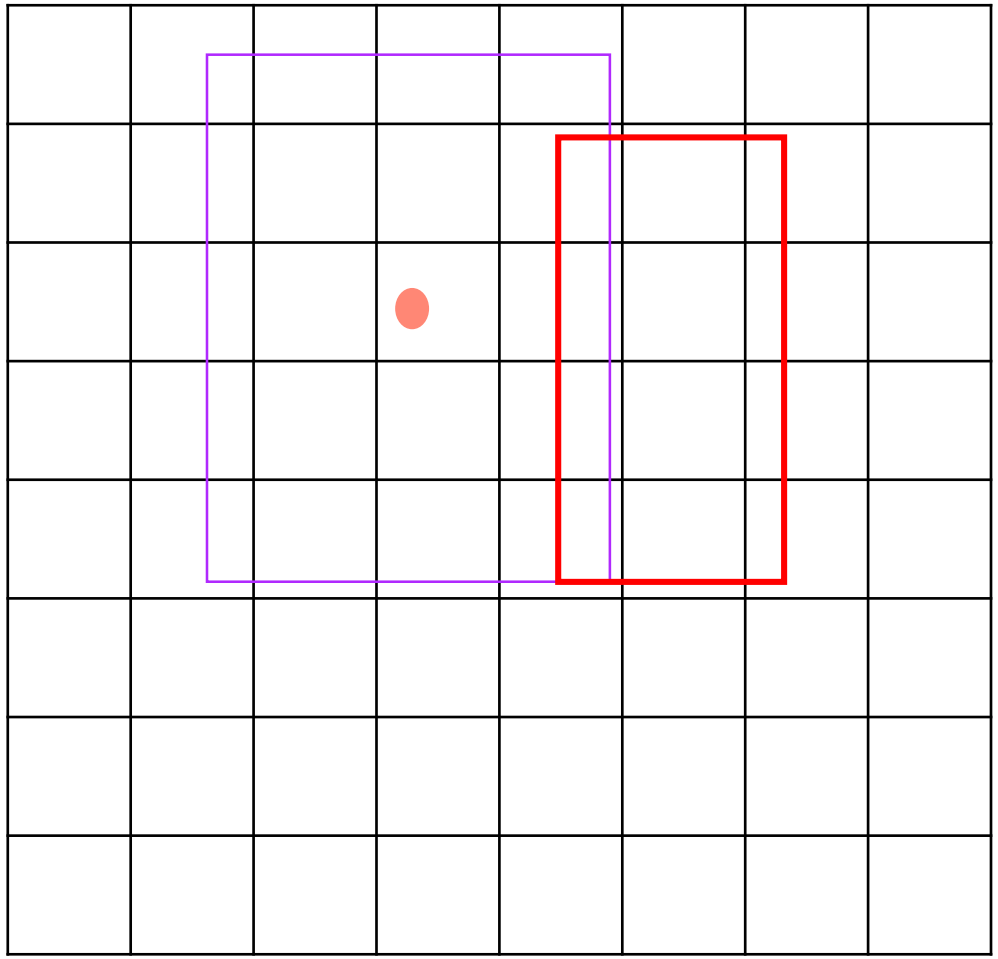
- Map GT to feature map.
- Define a pool of hypothet bounding boxes (called *anchors*) with different scales/aspect-ratios.

Region Proposal Network: Step 3



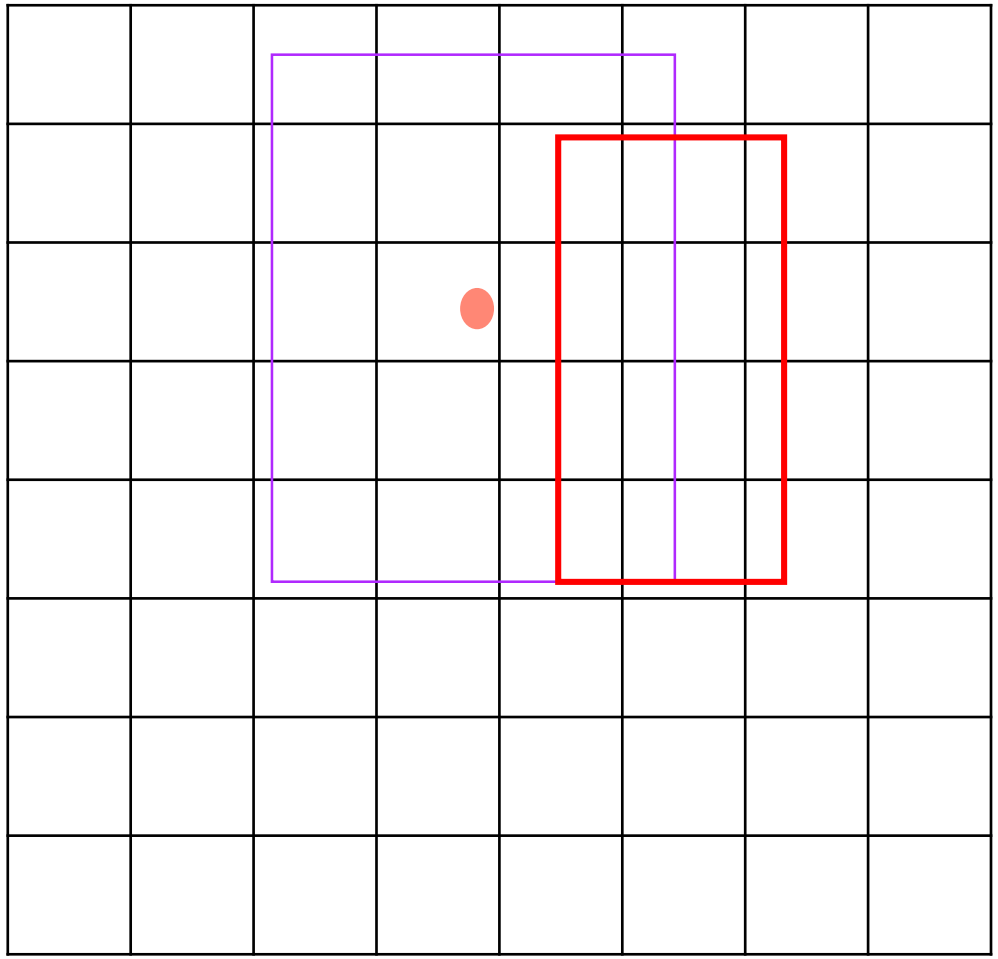
- Slide every anchor over the feature map and measure the intersection-over-union with every GT box.
- For IoU>UT we call it a positive anchor.
- For IoU<LT, we call it a negative anchor.
- For LT<IoU<UT, we simply ignore the anchor i.e don’t do any computations.

Region Proposal Network: Step 3



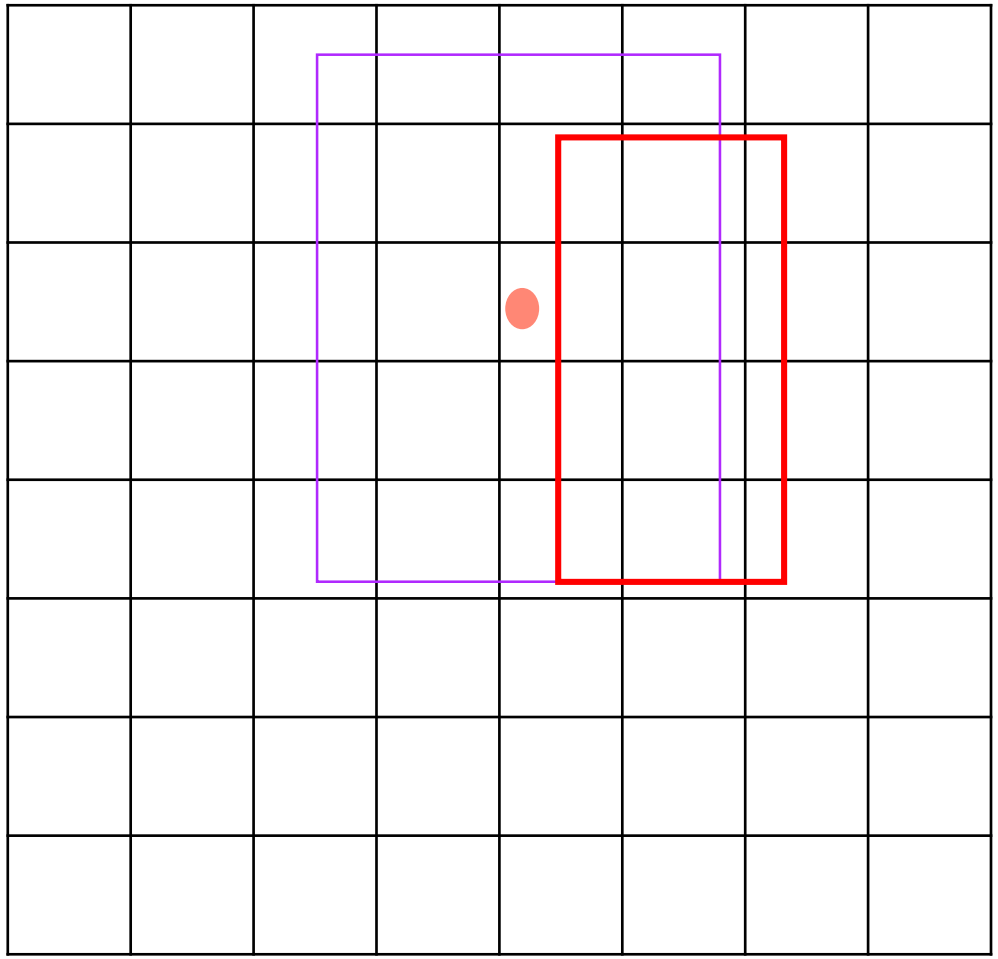
- Slide every anchor over the feature map and measure the intersection-over-union with every GT box.
- For IoU>UT we call it a positive anchor.
- For IoU<LT, we call it a negative anchor.
- For LT<IoU<UT, we simply ignore the anchor i.e don’t do any computations.

Region Proposal Network: Step 3



- Slide every anchor over the feature map and measure the intersection-over-union with every GT box.
- For IoU>UT we call it a positive anchor.
- For IoU<LT, we call it a negative anchor.
- For LT<IoU<UT, we simply ignore the anchor i.e don’t do any computations.

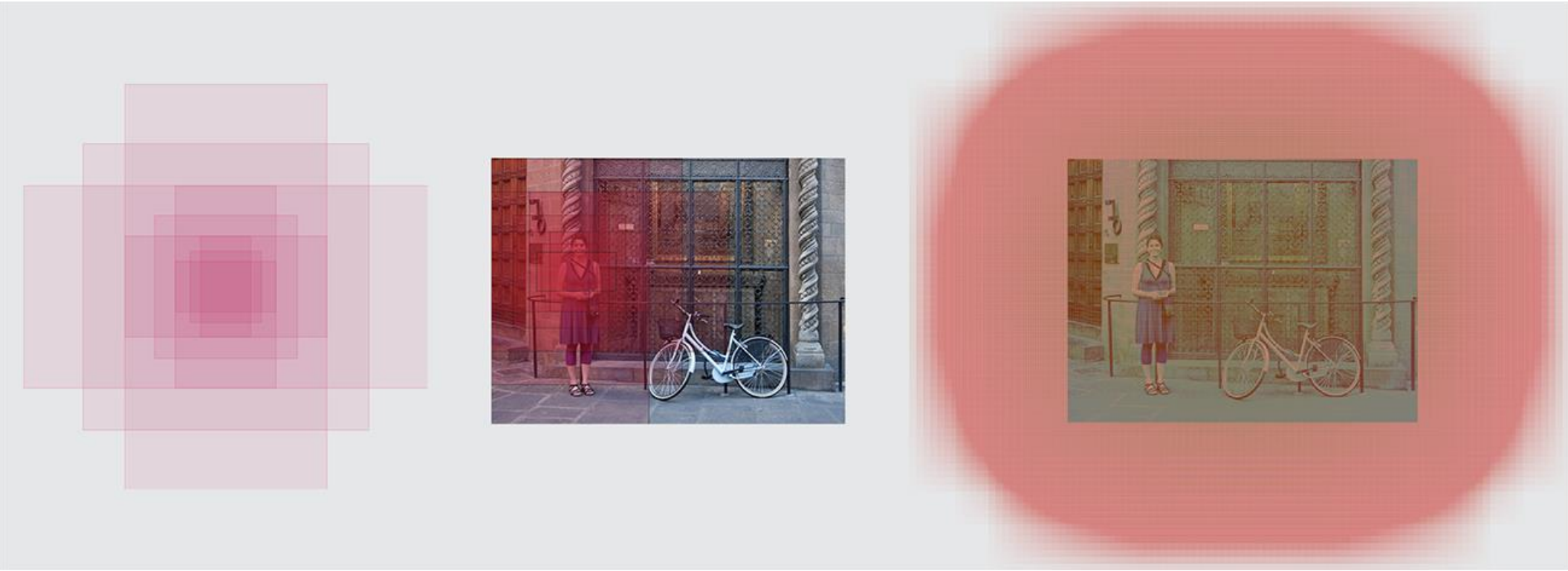
Region Proposal Network: Step 3



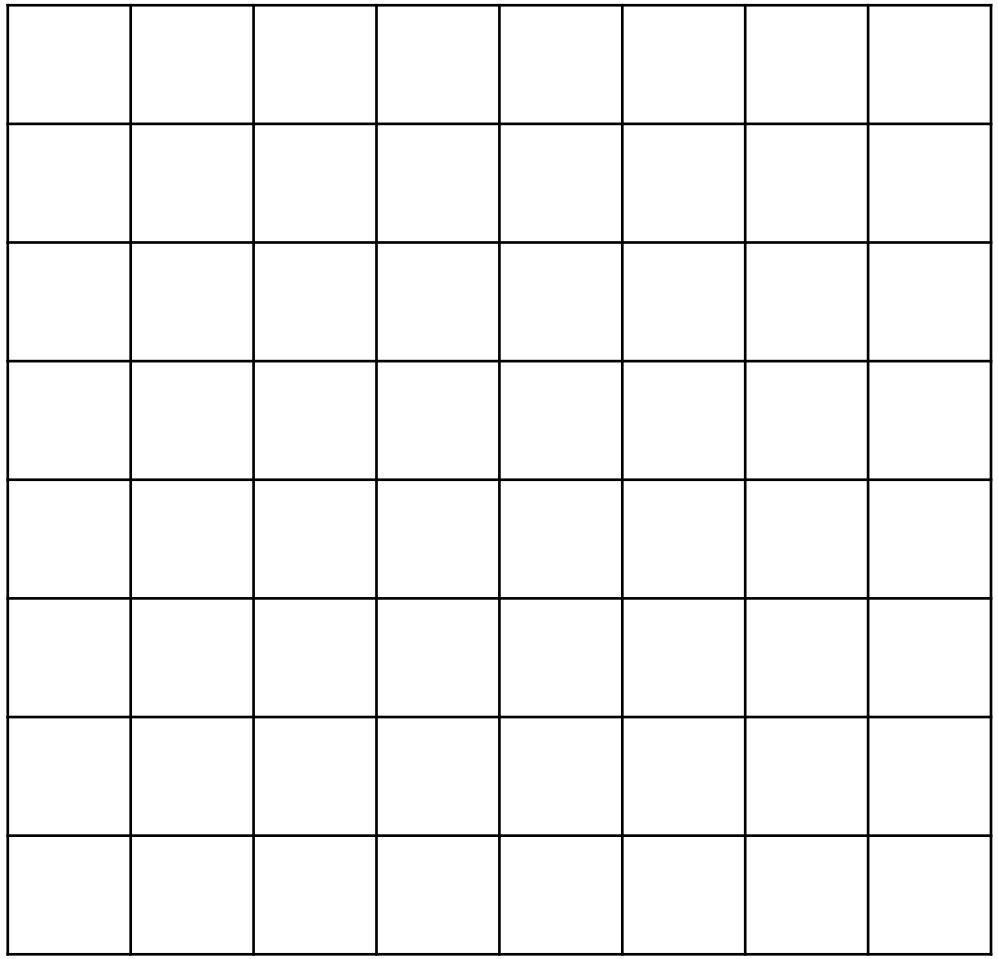
- Slide every anchor over the feature map and measure the intersection-over-union with every GT box.
- For IoU>UT we call it a positive anchor.
- For IoU<LT, we call it a negative anchor.
- For LT<IoU<UT, we simply ignore the anchor i.e don’t do any computations.

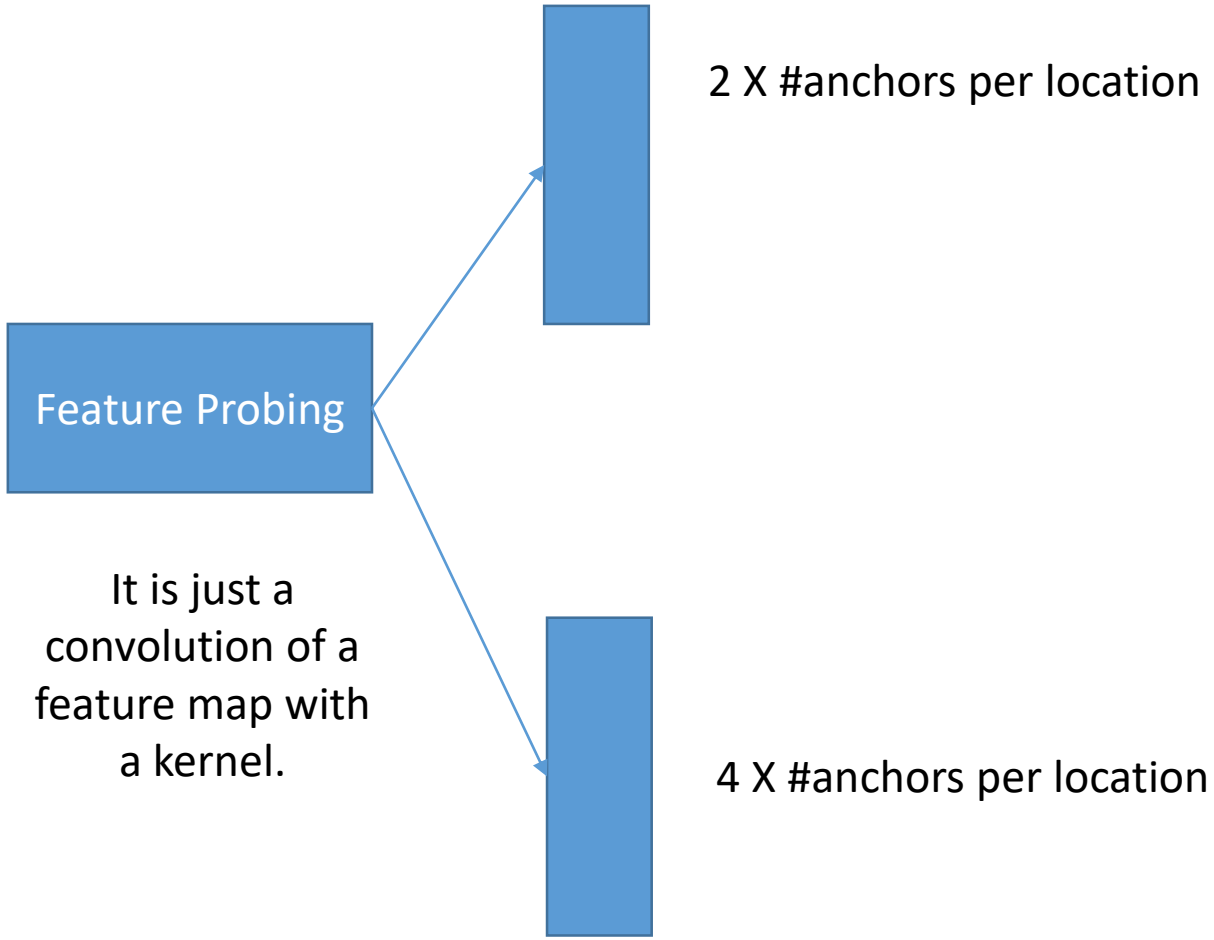
Region Proposal Network: Step 3

- In reality, this sliding is never done.
- Instead, it is assumed that anchors are tiled all over the feature map.



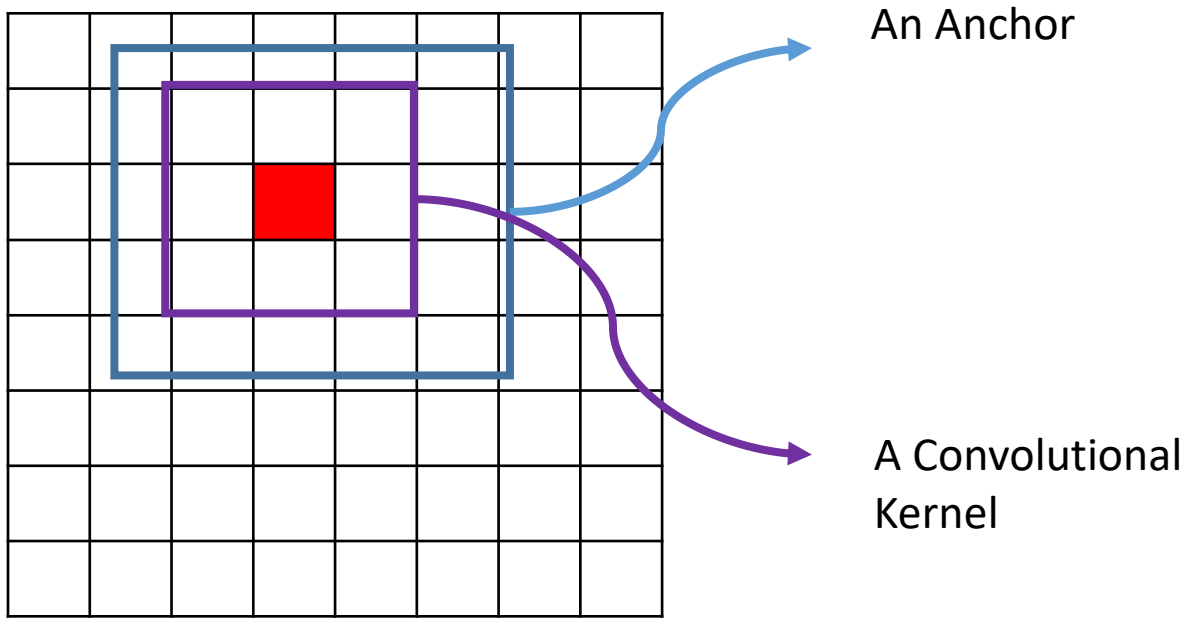
Region Proposal Network: Step 4



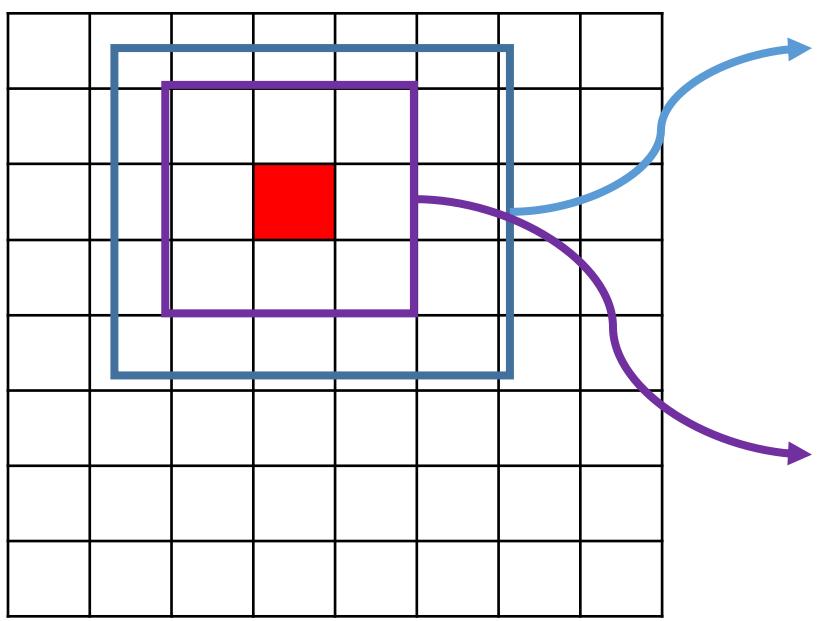


It is just a convolution of a feature map with a kernel.

A little deeper into Feature Probing



A little deeper into Feature Probing



- The convolutional kernel may not look completely inside an anchor.
- Thus the information it gathers through convolution is relatively incomplete.
- Multiple anchors are centered at each location.
- Therefore the convolutional kernel output is representative of all the confocal anchors.
- Being convolution, it is very fast.

RPN Output

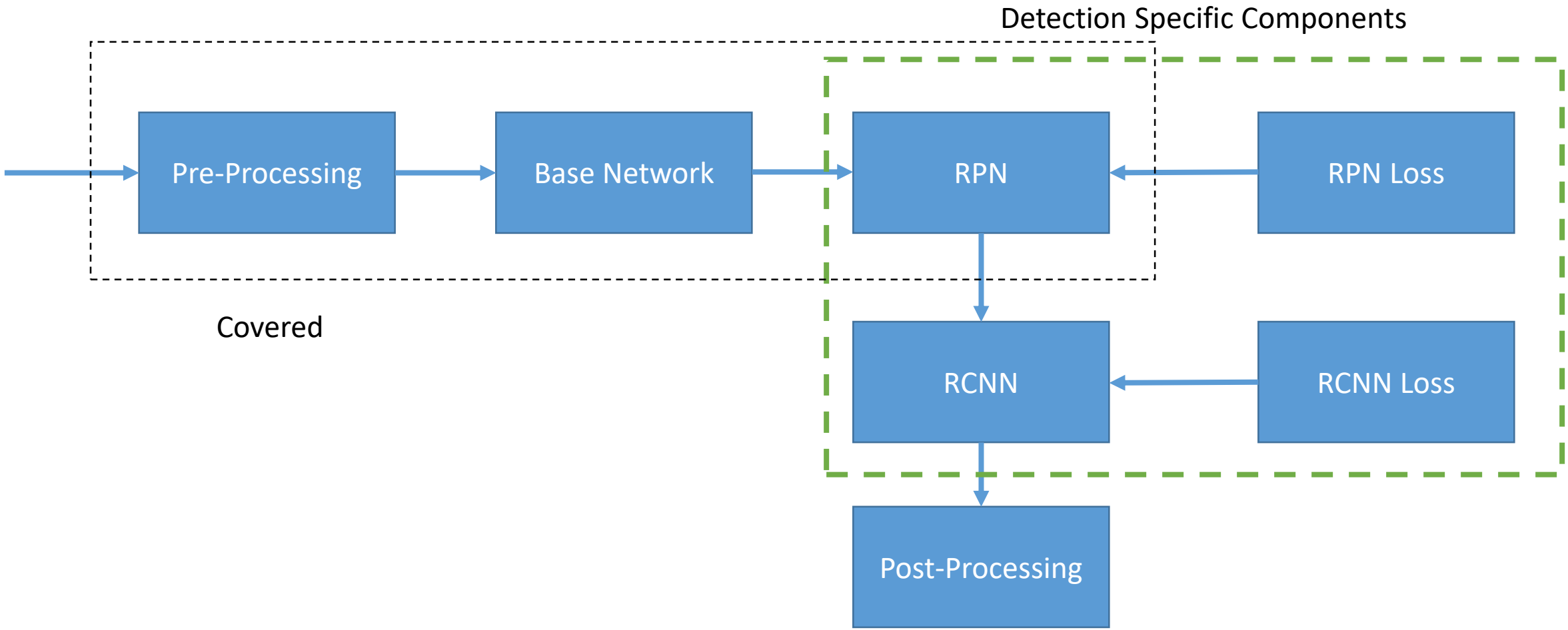
- The classifier of RPN describes if an object is of interest or not.
  - If an anchor is positive, during training it is labeled as an object of interest.
  - If an anchor is negative, during training is is labeled as no object.
  - If an anchor is in the don’t care range, we do not process it during training at all.
- The regressor of RPN simply, regresses the coordinates in order to better fit it to the bounding box of the object in the training set.

RPN Output

Original Image

RPN Output: Proposals

Faster-RCNN


Detection Specific Components
Pre-Processing
Base Network
RPN
RPN Loss
Covered
RCNN
RCNN Loss
Post-Processing

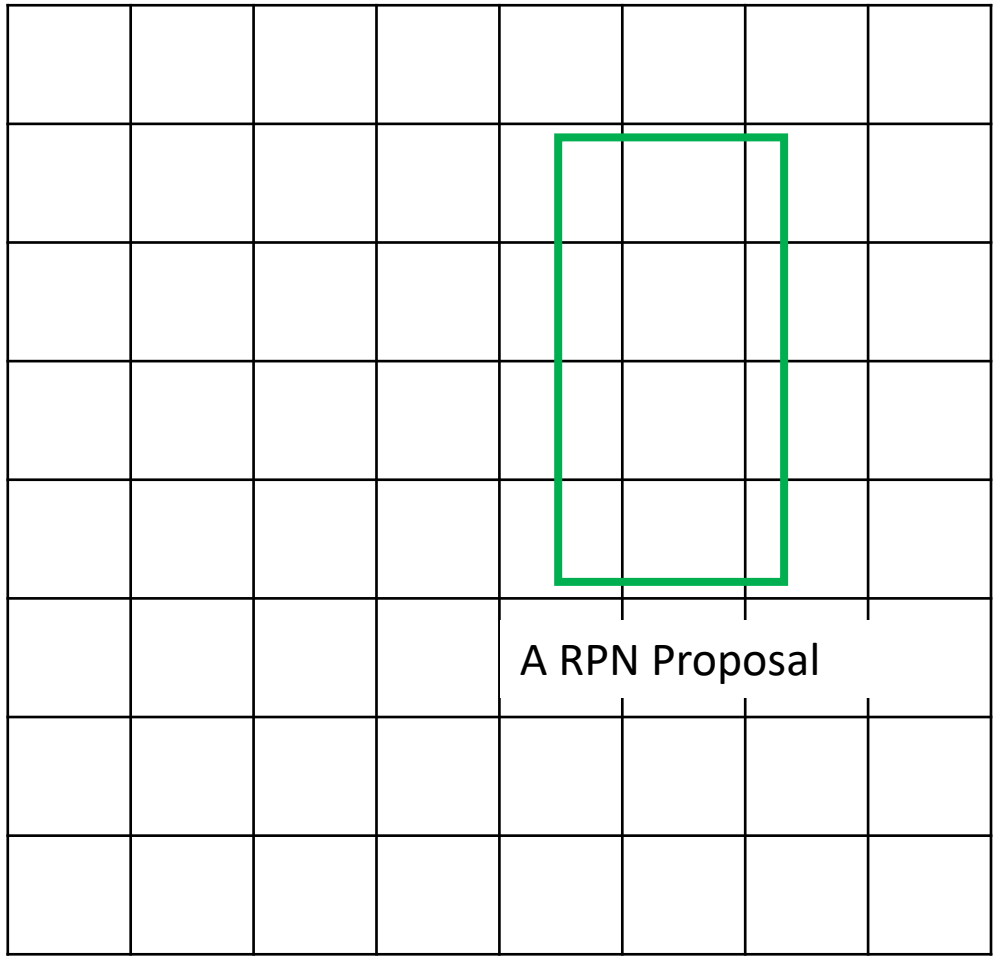
After RPN

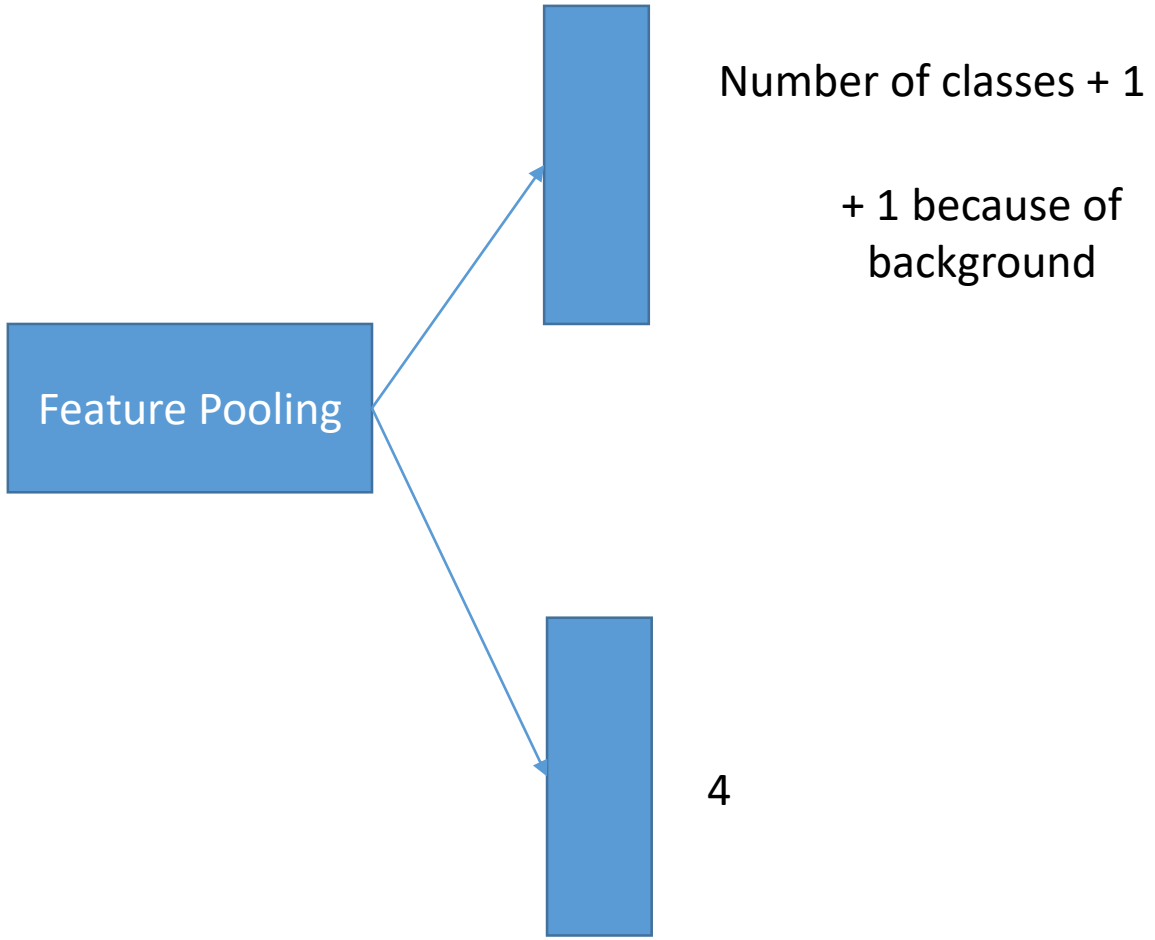
- RPN classification results in proposals with classification scores.
- Usually all proposals are not used for further processing.
- Proposals are ranked according to their classification scores.
- Top K of such proposals are selected and are further processed by RCNN during test time.
- What happens during training time ?

After RPN: Training time

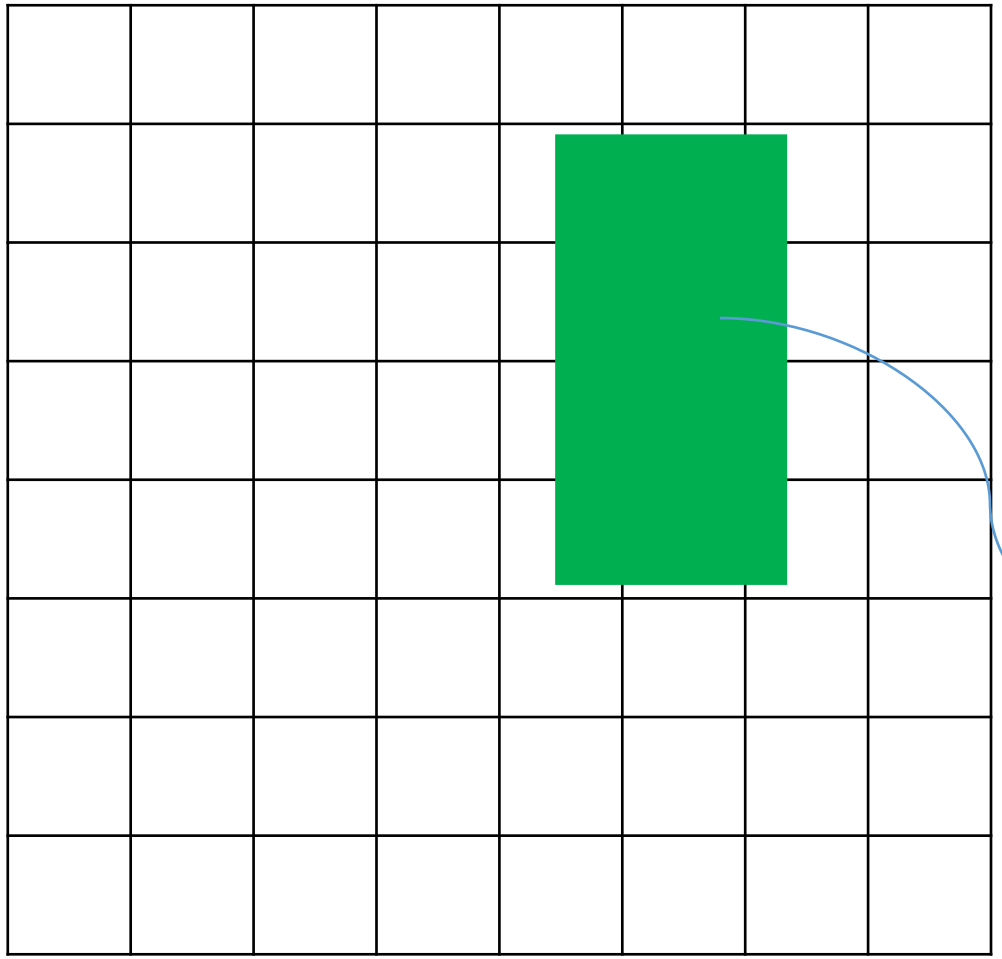
- Training in deep learning involves computing and optimizing a loss function.
- A good training regimen needs positive as well as negative examples.
- During training time a ratio of positive and negative examples is maintained during RPN training.
  - A ratio of 1:3 is found to be good. Here 1 refers to positive examples and 3 refers to negative examples.
  - This is a very critical fact which must be observed during the training of a deep learning system.

RCNN: Regional CNN





Feature Pooling



- Feature pooling means to extract features inside a subregion of an image or feature map.

Features inside the shaded area are extracted.

Why Feature Pooling ?

- For fully-connected layers we need a fixed length of a feature vector.
- Different anchors cover different spatial areas.
- Hence, feature pooling is needed in order to extract a fixed length feature vector from a region.

Challenges in Feature Pooling

- Anchor coordinates could be in non-integer locations.
- Higher computational complexity.

Methods of Feature Pooling

- ROI-Pooling.
- Crop and Resize Operation.
- ROI-Align : Will be covered with Mask-RCNN

ROI-Pooling

- There are two hyper-parameters in ROI-Pooling.
  - Pool height.
  - Pool width.

ROI-Pooling Operation

- Imagine a feature map with 256 channels.
- Let us suppose that,
  - Pool Height = 7
  - Pool Width = 7
- ROI-Pooling works as follows:
  - For a given ROI, divide it into 7x7 blocks.
  - Within each block do a max-pooling operation.
  - At the end you will end up with a 7x7x256 feature map.
  - Flatten it to get a fixed length feature vector.

ROI-Pooling: Cautions

- Some ROIs could be very small.
  - They need to be rejected.

Crop and Resize Operation

- This operation was proposed by a master's student in Stanford.
- This was never published but is widely used due to its simplicity and speed.
- The idea is as follows:
  - Crop the ROI.
  - Resize the ROI to a fixed size i.e Pool height x Pool Width x Number of channel
  - Flatten it to get a fixed-length feature vector.
- The resizing must be done using nearest neighbor or bilinear interpolation.
- Why can't you use bicubic interpolation ?

Loss Functions in Faster-RCNN

- Classification Loss
  - Cross Entropy : Same as used in classification module.
  - Focal Loss: Will be covered in detail during Feature Pyramid Networks.
- Regression Loss
  - Smooth L1 Loss
  - Repulsion Loss
- Remember in Faster-RCNN these losses are used for RPN as well as RCNN.

Feature Pooling vs. Feature Probing

- Feature pooling is significantly slower than feature probing.
  - A speed difference of around 2-18 times can be observed depending upon:
    - Pooling size.
    - Size of feature map.
    - Hardware specification.

Loss Function for RPN

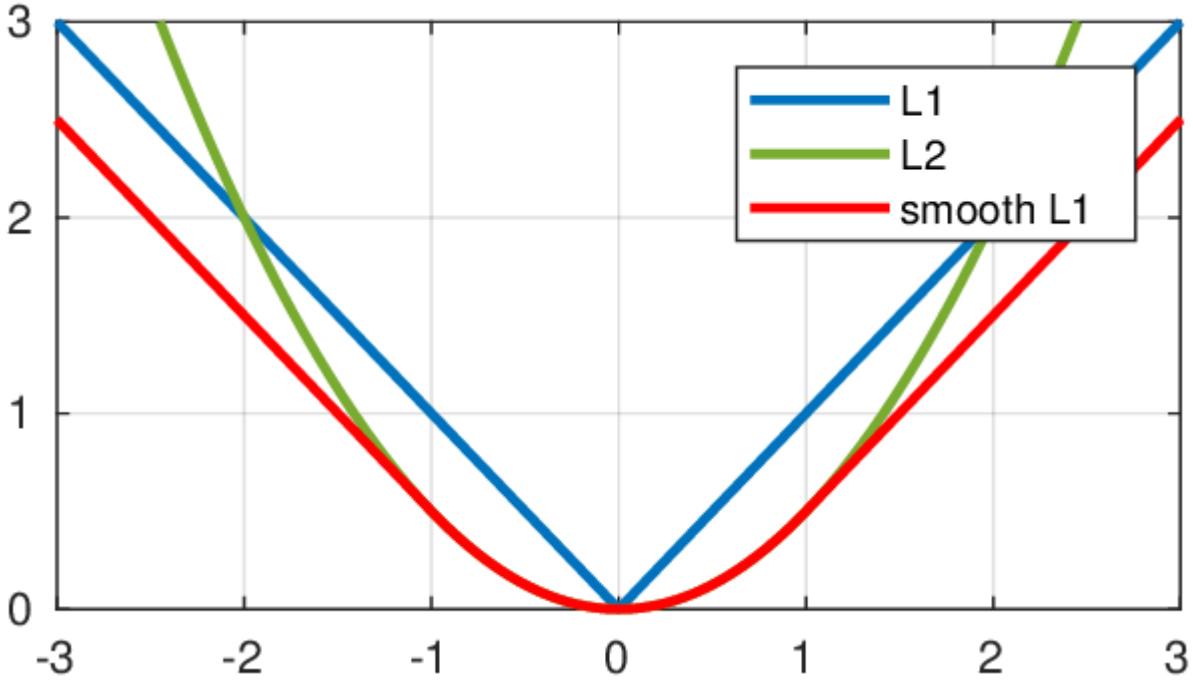
$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \frac{\lambda}{N_{reg}}\sum_i L_{reg}(t_i, t_i^*)\, p_i^*$$

- $\{p_i\}$: Labels of anchors (+ve Vs. –ve)
- $\{t_i\}$: Bounding box coordinates of anchors.
- L_{cls} : Cross-entropy loss
- L_{reg} : Smoothed-L1 loss.
- λ : Scalar constant.
- p_i^* : Groundtruth label of the anchor.
- t_i^* : Groundtruth bounding box coordinates
- N_{cls} : Minibatch size
- N_{reg} : Total number of anchor locations

Loss Function for RCNN

- Same as RPN except:
  - Now classification is across N classes.
  - All bounding boxes are regressed except the background ones.

Smooth L1 Loss



- Can you think which one of these 3 is suitable for bounding box regression ?
- Most importantly why ?

Smooth L1-Loss

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

Regression in Faster-RCNN

Ground Truth Bounding Box

w^*: Box width
h^*: Box height
x^*,y^*: Box center

Anchor's properties

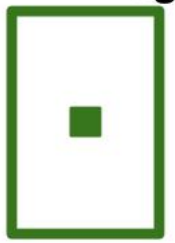
w_a: width
h_a: height
x_a,y_a: center

$p^* \in \{0,1,-1\}$
based on IoU between
GT-Box and Anchor

Classification ***cls***

p

Regressor ***reg***



w: width
h: height
x,y: center

$$t = [(x-x_a)/w_a,\ (y-y_a)/h_a,\ \log w/w_a,\ \log h/h_a]$$

$$t^* = [(x^*-x_a)/w_a,\ (y^*-y_a)/h_a,\ \log w^*/w_a,\ \log h^*/h_a]$$

@vmirly

Overall Training

- There are several ways to train Faster-RCNN
  - Alternating Training : Train RPN first and then train RCNN.
  - Approximate Joint Training: ROI Pooling layer gradients with respect to bounding box coordinates are ignored.
  - Non-approximate Joint Training: ROI Pooling layer gradients with respect to bounding box coordinates are not ignored.

What you need to know about Object Detectors ?

- Understanding comes from both reading (40%) and implementing (60%).
- Understand your data.
  - Pay attention to number and type of classes.
  - What are salient characteristics of the data.
  - Is the data properly labeled ?
- Good object detector is built from a good backbone.
- Experiment exhaustively with all parameters and develop your intuition.