Imperial College London

Object Detection

Dr Wenjia Bai

Department of Computing & Brain Sciences

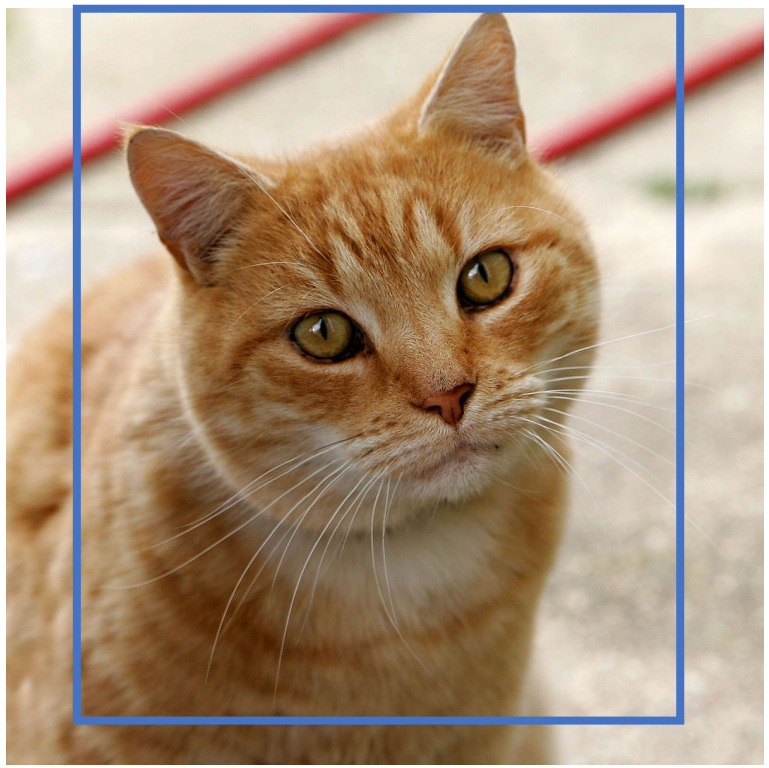
Image classification

Classifier

Cat

Probability
Cat: 85%
Tiger: 10%
Dog: 1%
…

Object detection



Detector

Cat

Bounding box
(x, y, w, h) denoting the top left (or centre) and width, height of the box

Object detection

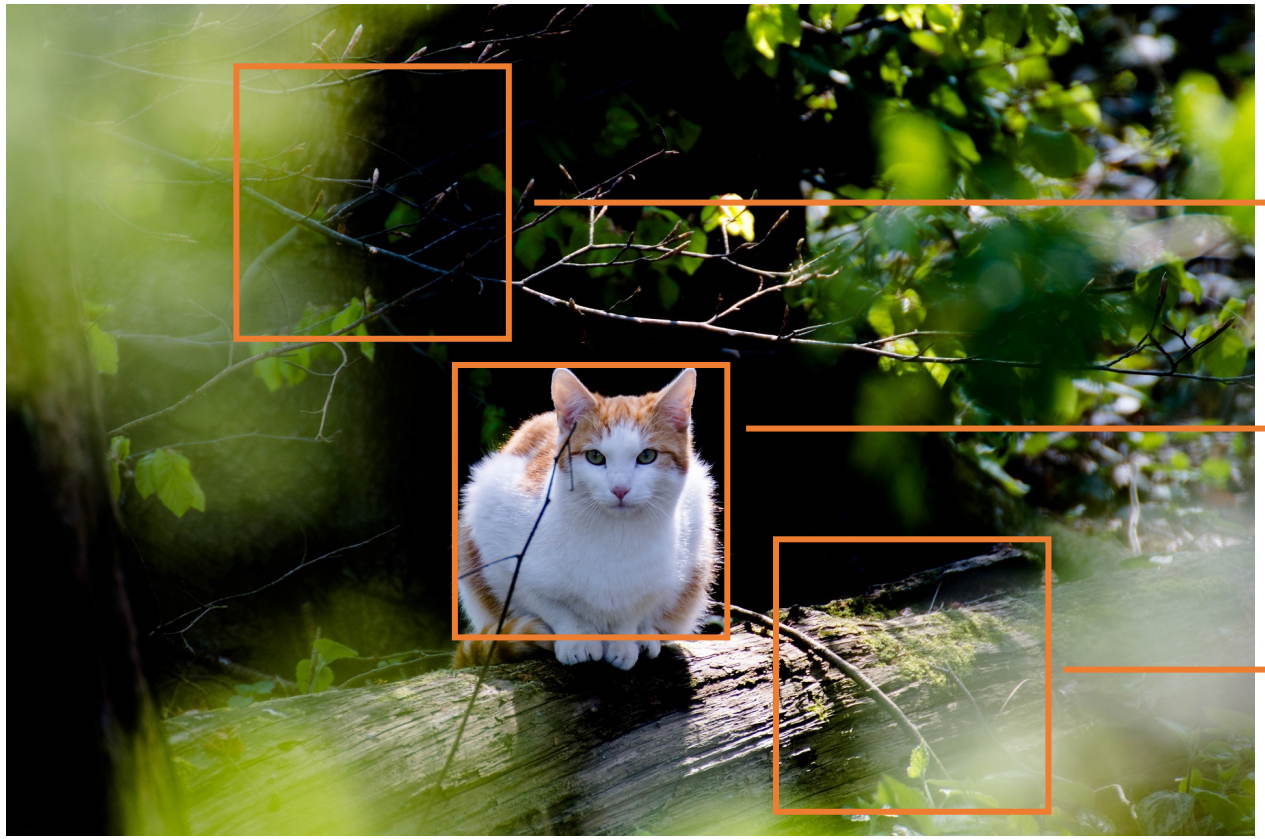
- The goal is to predict a set of bounding boxes and labels for each object of interest.
- But how?
  - We have already built a neural network that can perform image classification.
  - How about moving a sliding window and applying the classification network to different regions of an image for detection?

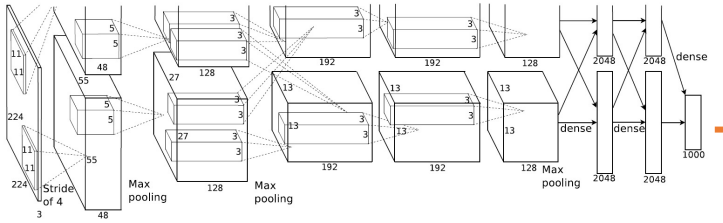
Linear SVM + HOG feature for human detection

Credit: https://medium.com/@madhawavidanapathirana

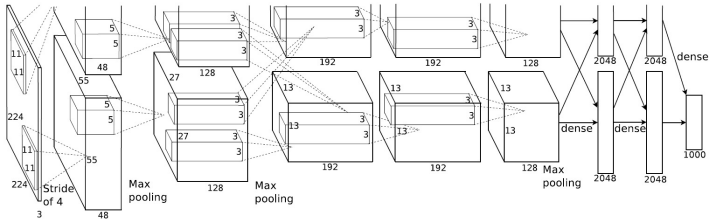
Object detection

- Previously, we used a similar idea, i.e. sliding window + linear SVM, for human detection.
- Now, let us replace the SVM with a neural network.

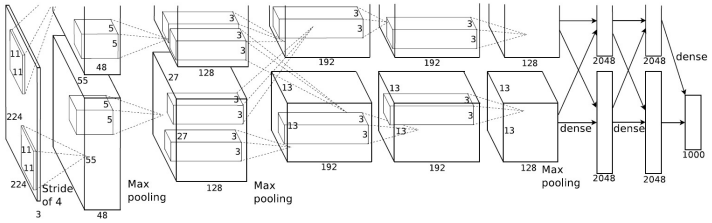




Background



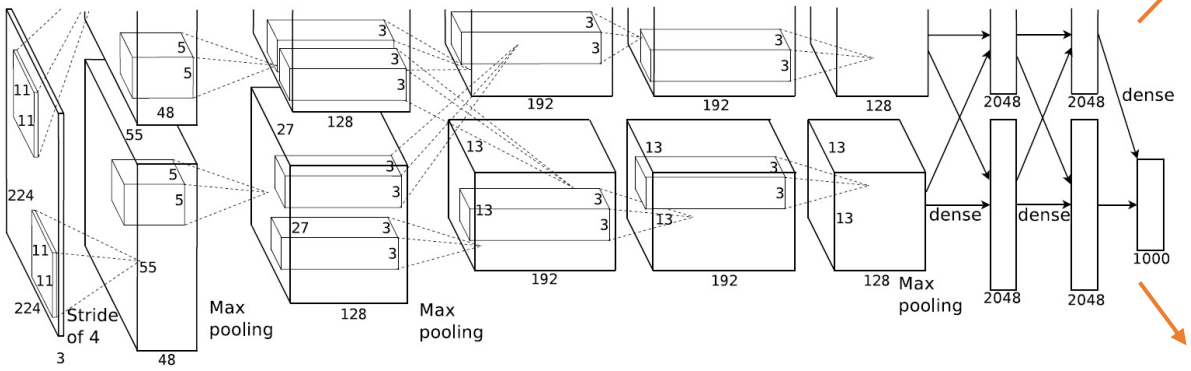
Cat



Background

Ideas for object detection

- At each sliding window, we perform two tasks:
  - Task 1: classification (whether this is a cat or not)
  - Task 2: localisation (bounding box coordinates)
  - The sliding window already provides rough localisation. Using CNN features, we can refine the localisation.



Fully connected to K classes → Class score → $L_{classification}$

Ground truth class label

Fully connected to 4 variables → Bounding box (X, Y, W, H) → $L_{localisation}$

Ground truth bounding box

Ideas for object detection

- It may be too expensive to apply a convolutional network to each pixel location on the image.
- What about having some initial guesses?

Object detection

- Two-stage detection
  - Stage 1: initial guess, propose some possible regions of interest
  - Stage 2: classify what these regions are and predict the bounding boxes

- One-stage detection
  - Do not guess. But simply divide the image into grid cells.
  - Classify these grid cells and predict the bounding boxes.

Two-stage object detection

	Stage 1: Region proposal	Stage 2: Detector
R-CNN, 2014 [1]	Selective search	SVM for classification, linear regression for localization
Fast R-CNN, 2015 [2]	Selective search	CNN
Faster R-CNN, 2015 [3]	CNN	CNN
...		

Comparison of R-CNN-based methods.
The name R-CNN stands for Regions with CNN features.

[1] Ross Girshick et al. Rich feature hierarchies for accurate object detection and semantic segmentation. CVPR 2014.
[2] Ross Girshick. Fast R-CNN. ICCV 2015.
[3] Shaoqing Ren et al. Faster R-CNN: Towards real-time object detection with region proposal networks. NIPS 2015.

Two-stage object detection

- R-CNN and Fast R-CNN rely on an conventional approach, selective search, for the region proposal stage.

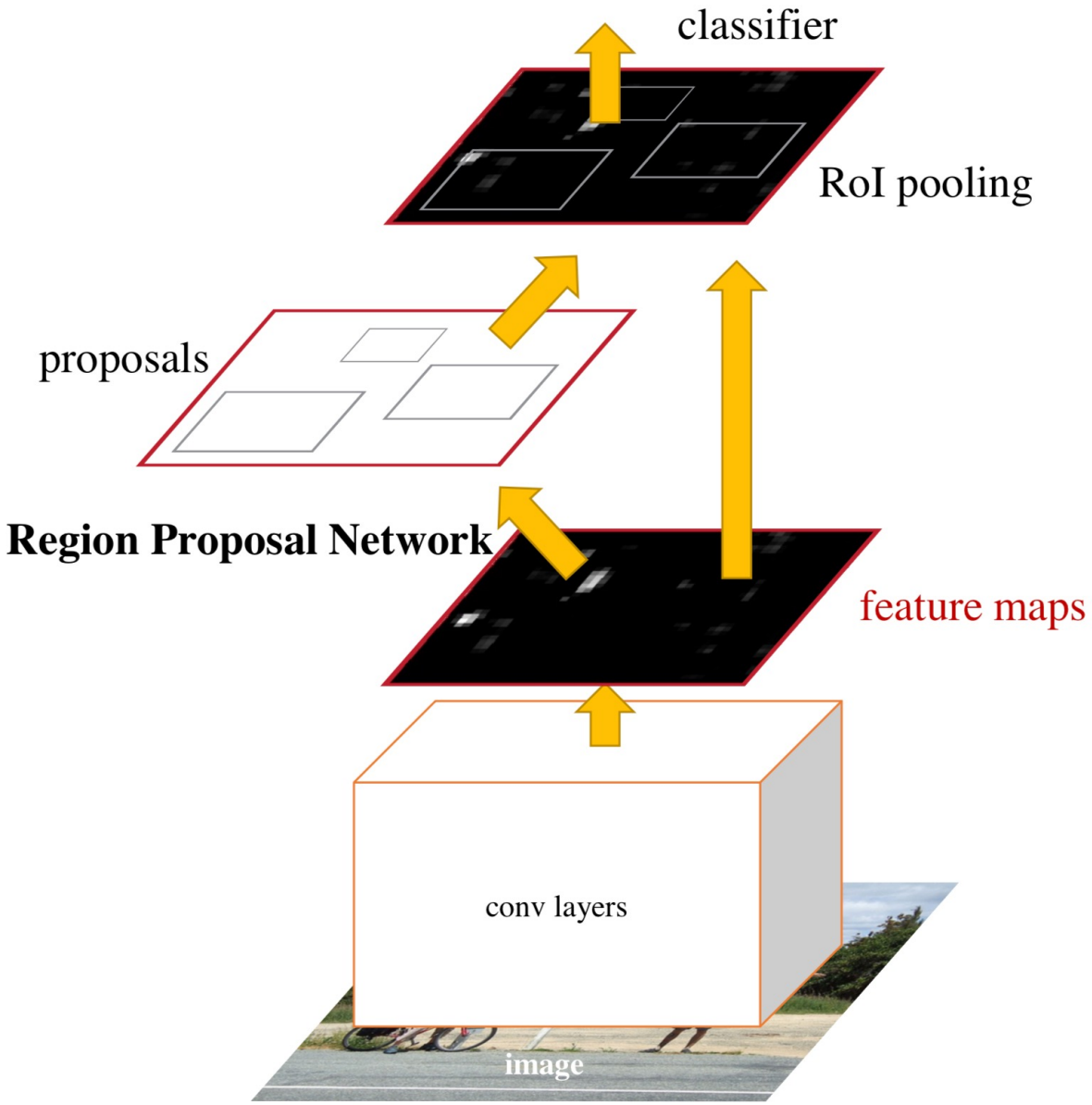
Selective search for region proposals.

The image is segmented (top). Regions with similar features (colour, texture etc.) are grouped together, which provide the proposals (blue boxes at the bottom figures).

J.R.R. Uijlings et al. Selective search for object recognition, IJCV 2013.

Two-stage object detection

- We will focus on explaining Faster R-CNN, which uses CNN for both stages.
  - Stage 1: a region proposal network (RPN)
  - Stage 2: a detection network
- It is faster compared to R-CNN and Fast R-CNN, due to the efficiency of the region proposal network, compared to selective search.



S. Ren et al. Faster R-CNN: Towards real-time object detection with region proposal networks. NIPS 2015.

Faster R-CNN

- Faster R-CNN consists of two stages.
  - Stage 1: a region proposal network (RPN)
  - Stage 2: a detection network
- Both stages rely on the convolutional feature map.
  - What is this feature map?

AlexNet

[224x224x3] Input
[55x55x96] Conv1, 11x11, 96, s=4
[55x55x96] Norm1
[27x27x96] Pool1
[27x27x256] Conv2, 5x5, 256
[27x27x256] Norm2
[13x13x256] Pool2
[13x13x384] Conv3, 3x3, 384
[13x13x384] Conv4, 3x3, 384
[13x13x256] Conv5, 3x3, 256
[13x13x256] Norm3
[6x6x256] Pool3
[4096] FC1
[4096] FC2
[1000] FC3 (class score)

convolutional feature map

VGG-16

[224x224x3] Input
[224x224x64] 3x3 conv1, 64
[224x224x64] 3x3 conv1, 64
[112x112x64] Pool
[112x112x128] 3x3 conv2, 128
[112x112x128] 3x3 conv2, 128
[56x56x128] Pool
[56x56x256] 3x3 conv3, 256
[56x56x256] 3x3 conv3, 256
[56x56x256] 3x3 conv3, 256
[28x28x256] Pool
[28x28x512] 3x3 conv4, 512
[28x28x512] 3x3 conv4, 512
[28x28x512] 3x3 conv4, 512
[14x14x512] Pool
[14x14x512] 3x3 conv5, 512
[14x14x512] 3x3 conv5, 512
[14x14x512] 3x3 conv5, 512
[7x7x512] Pool
[4096] FC, 4096
[4096] FC, 4096
[1000] FC, 1000

convolutional feature map

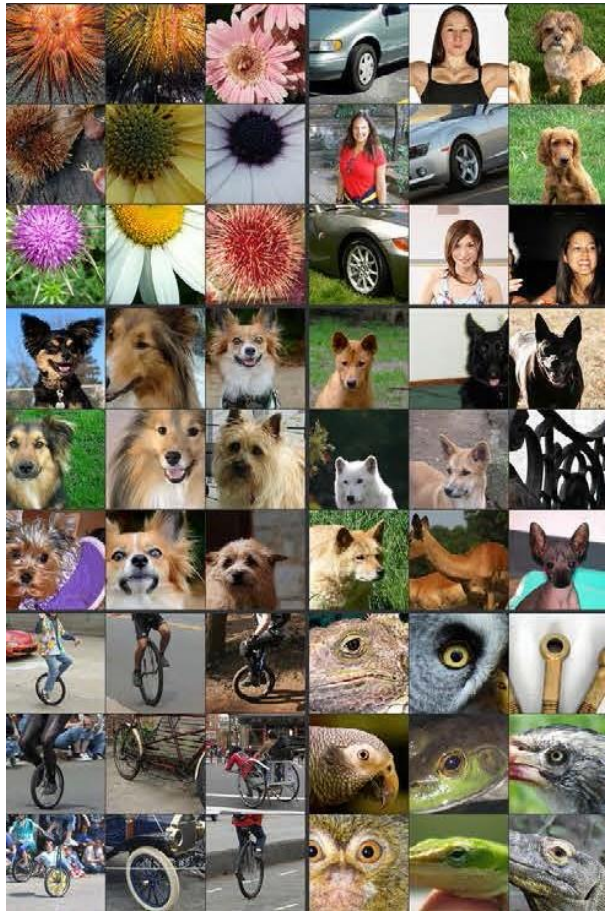
These are called backbone networks, which provide the convolutional feature maps.
They are often pre-trained on ImageNet or other datasets.

Convolutional feature map

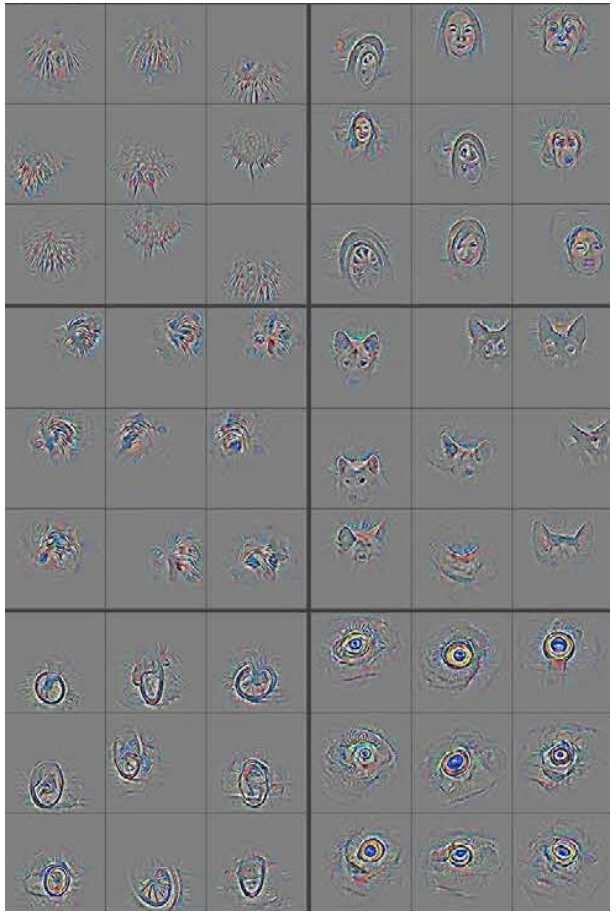
- For VGG net, there are 4 max pooling layers before conv5.
- Therefore, the first two dimensions of the feature map is X/16 x Y/16.
  - If the input image is 224 x 224, then conv5 feature map is 14 x 14 x D.
  - For larger input images, e.g. 640 x 480, conv5 feature map is larger, e.g. 40 x 30 x D.
  - D is the depth dimension or the number of filters. For conv5 in VGG net, D = 512.

Convolutional feature map

- Each pixel of the feature map is a high-dimensional feature vector, which describes the content of a small region in the input image.
- This feature vector may tell us whether this is an interesting region or not.



Input images

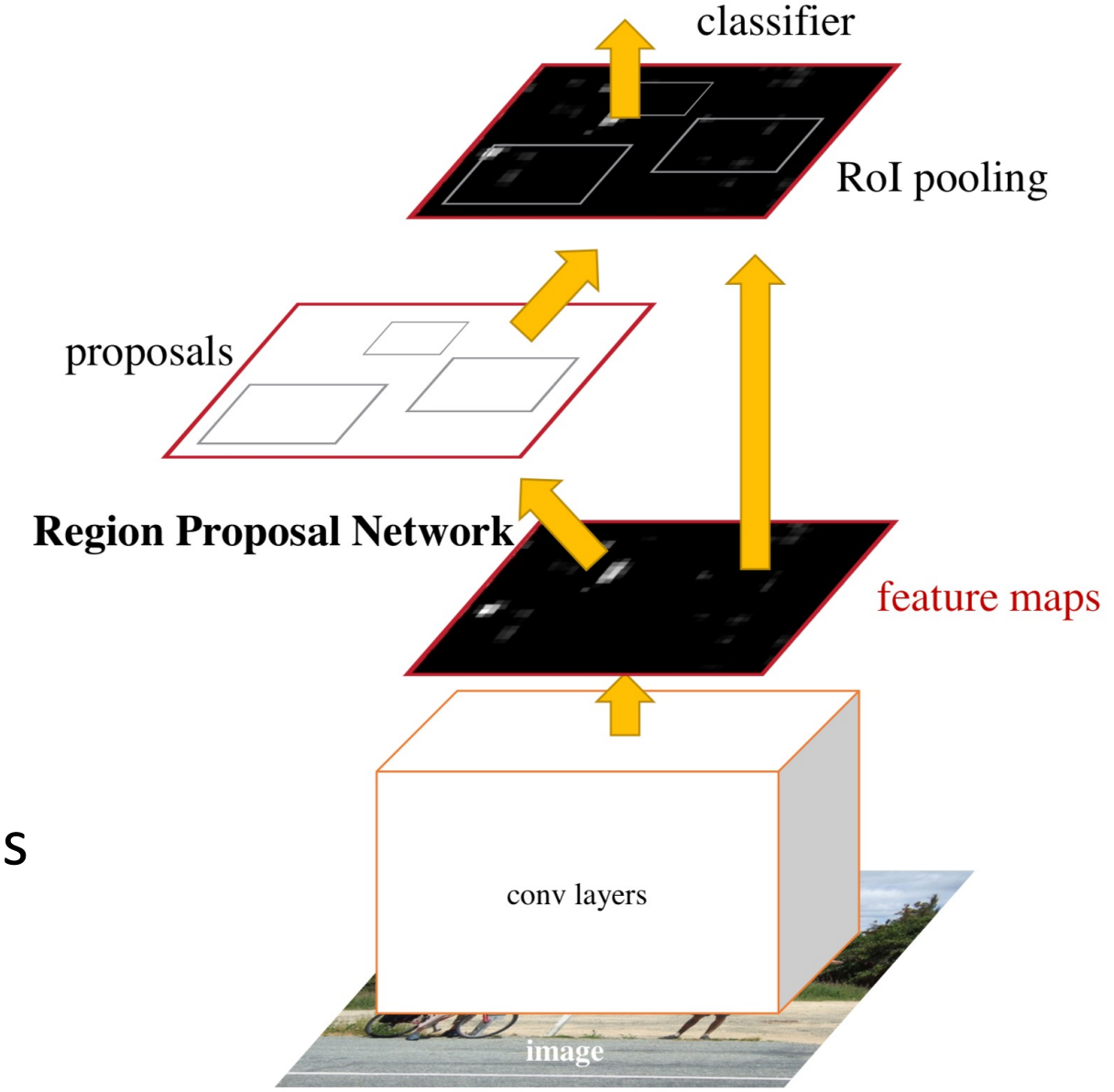


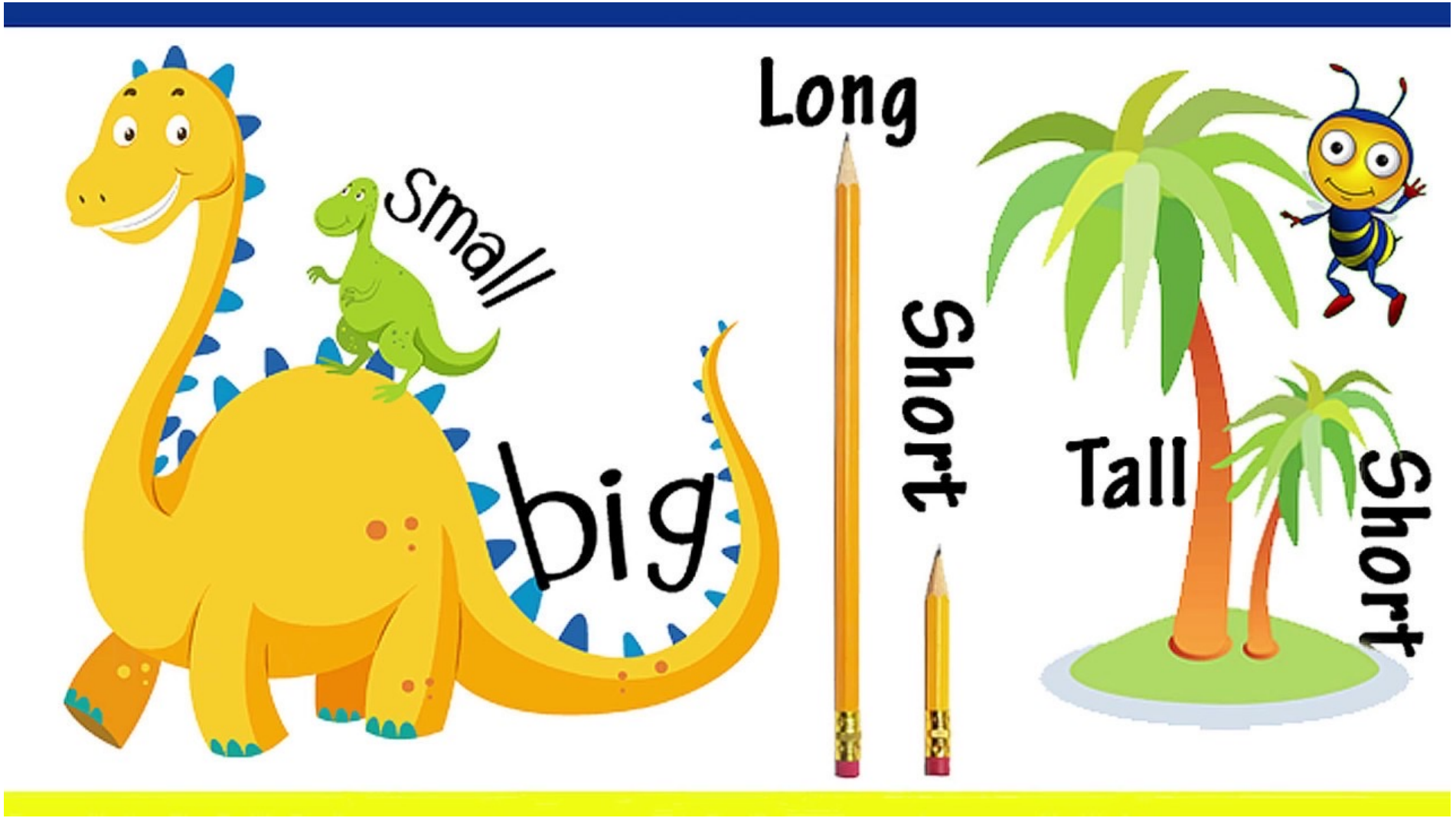
Visualisation of Conv5 features

M.D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. ECCV 2014.

Region proposal network

- To proposing regions, we go across the convolutional feature map using a 3x3 sliding window.
- At each location, we perform a binary classification.
  - 0: not interesting
  - 1: interesting
- However, how does this fixed size window handle objects of different sizes and different aspect ratios?



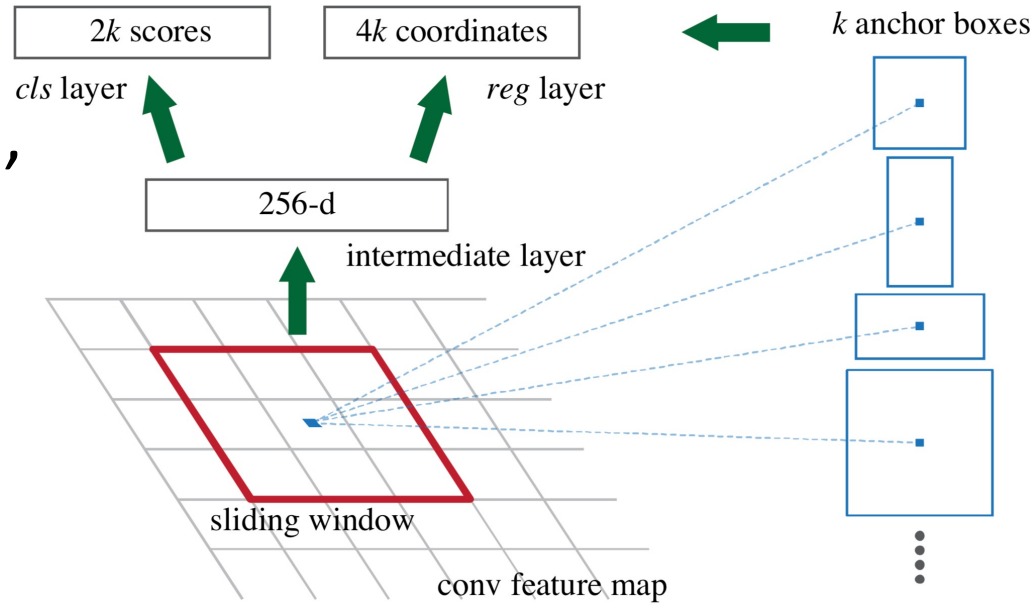


We want to detect objects of different sizes and aspect ratios.

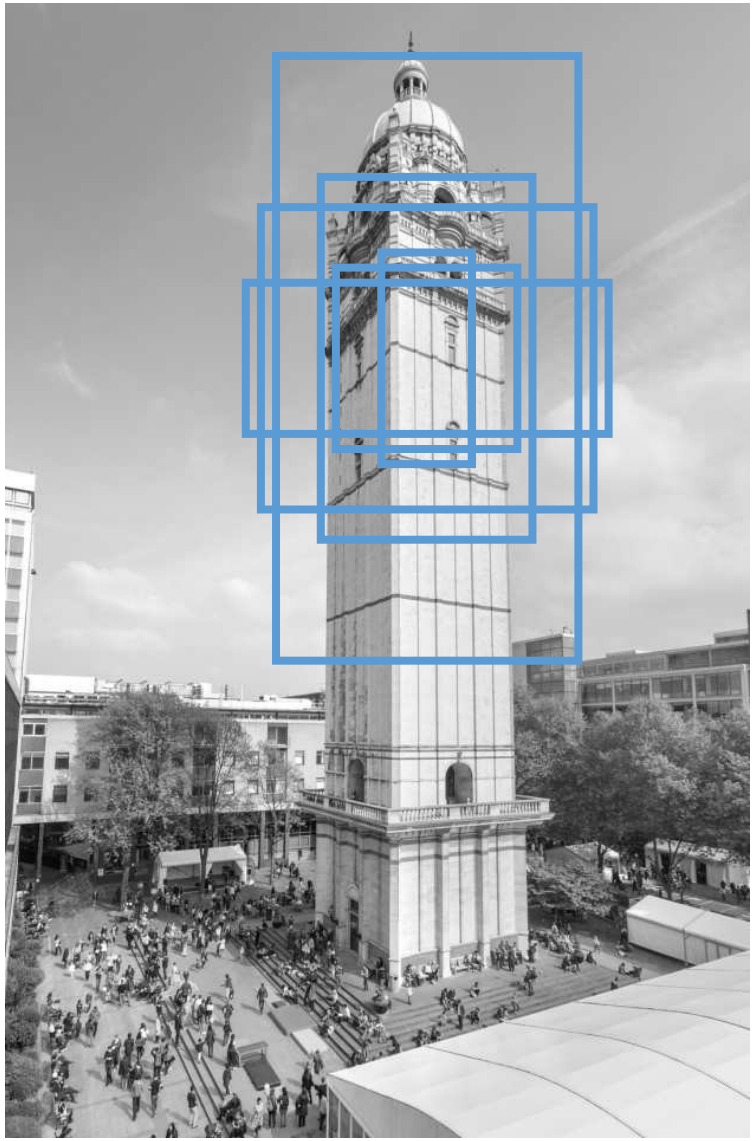
Region proposal network

- At each sliding window, the network makes k predictions (k bounding boxes), for objects of different scales and aspect ratios.
  - 3 scales: 128^2, 256^2, 512^2
  - 3 aspect ratios: 1:1, 1:2, 2:1
  - In total, k = 3 x 3 = 9
  - These bounding boxes are called “anchors”.
- For a convolutional feature map of size W x H,

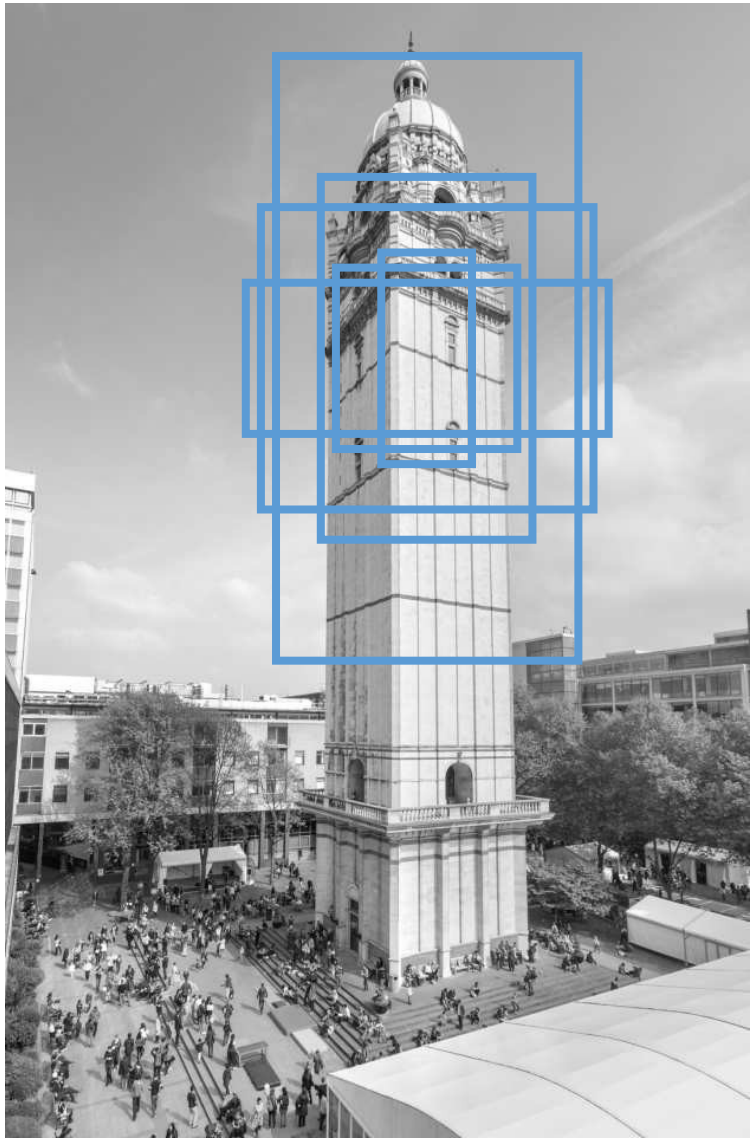
  W x H x k anchors are predicted.



S. Ren et al. Faster R-CNN: Towards real-time object detection with region proposal networks. NIPS 2015.



At each location, the network aims to classify whether each anchor (e.g. 128x128 anchor, 128x256 anchor, 128x64 anchor, 256x256 anchor …) is an object or not.



At test time, the network will generate an objectness score for each anchor at each location on the feature map. Only anchors with the highest scores will be kept.

Region proposal network

- The loss is defined as

$$L(p,t) = \sum_{i=1}^{n_{anchor}} L_{cls}(p_i, p_i^*) + \lambda \cdot \sum_{i=1}^{n_{anchor}} 1_{y=1} L_{loc}(t_i, t_i^*)$$

p_i: predicted objectness score

p_i^*: ground truth, binary value

Classification loss

t_i: predicted bounding box

t_i^*: ground truth bounding box

Localisation loss

How do we describe a bounding box?

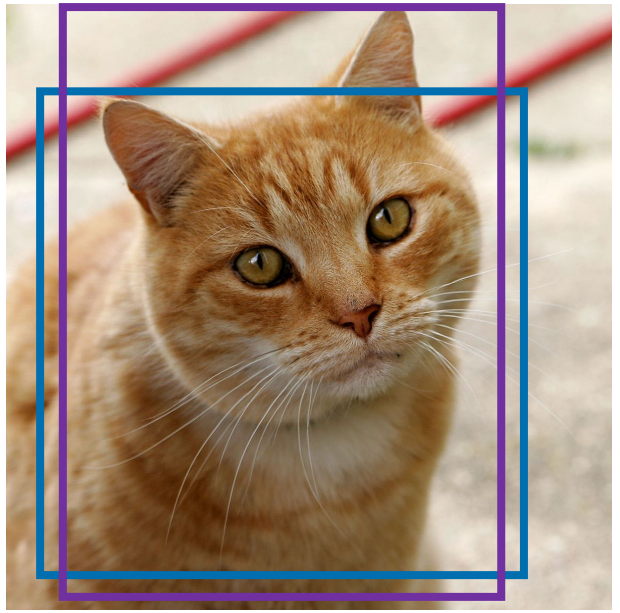
- A bounding box can be described by its centre coordinates (x, y) and size (w, h).
- We can either directly predict these 4 coordinates or transformation parameters t.
- If this prediction is for an anchor of size 128x128 and of 1:1 aspect ratio, we predict how to transform this anchor into the ground truth bounding box.
  - Anchor: (x_a, y_a, w_a, h_a)
  - Ground truth bounding box: (x^*, y^*, w^*, h^*)
  - Ground truth transformation t^*: $(t_x^*, t_y^*, t_w^*, t_h^*)$

$$t_x^* = \frac{x^* - x_a}{w_a}$$

$$t_y^* = \frac{y^* - y_a}{h_a}$$

$$t_w^* = \log\left(\frac{w^*}{w_a}\right)$$

$$t_h^* = \log\left(\frac{w^*}{h_a}\right)$$



Blue box: anchor
Purple box: ground truth

How do we describe a bounding box?

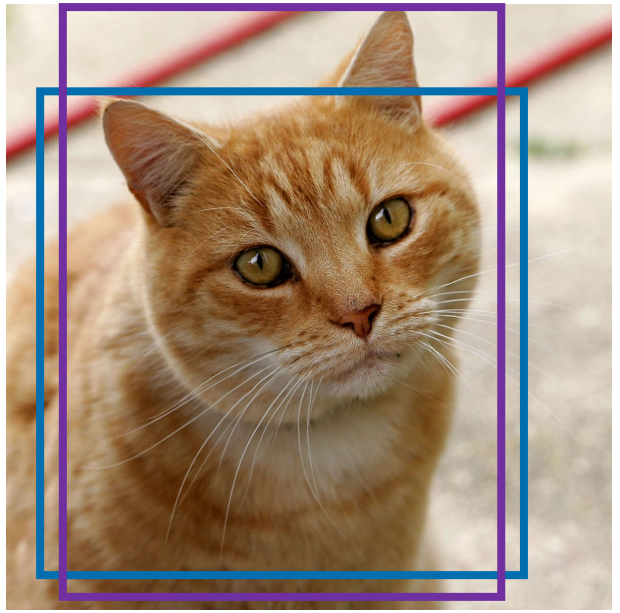
- A bounding box can be described by its centre coordinates (x, y) and size (w, h).
- We can either directly predict these 4 coordinates or transformation parameters t.
- If this prediction is for an anchor of size 128x128 and of 1:1 aspect ratio, we predict how to transform this anchor into the ground truth bounding box.
  - Anchor: (x_a, y_a, w_a, h_a)
  - Predicted bounding box: (x, y, w, h)
  - Predicted transformation t: (t_x, t_y, t_w, t_h)

$$t_x = \frac{x - x_a}{w_a}$$

$$t_y = \frac{y - y_a}{h_a}$$

$$t_w = \log\left(\frac{w}{w_a}\right)$$

$$t_h = \log\left(\frac{h}{h_a}\right)$$



Blue box: anchor
Purple box: ground truth

Region proposal network

- The loss is defined as

$$L(p,t) = \sum_{i=1}^{n_{anchor}} L_{cls}(p_i, p_i^*) + \lambda \cdot \sum_{i=1}^{n_{anchor}} 1_{y=1} L_{loc}(t_i, t_i^*)$$

p_i: predicted objectness score

p_i^*: ground truth, binary value

Classification loss

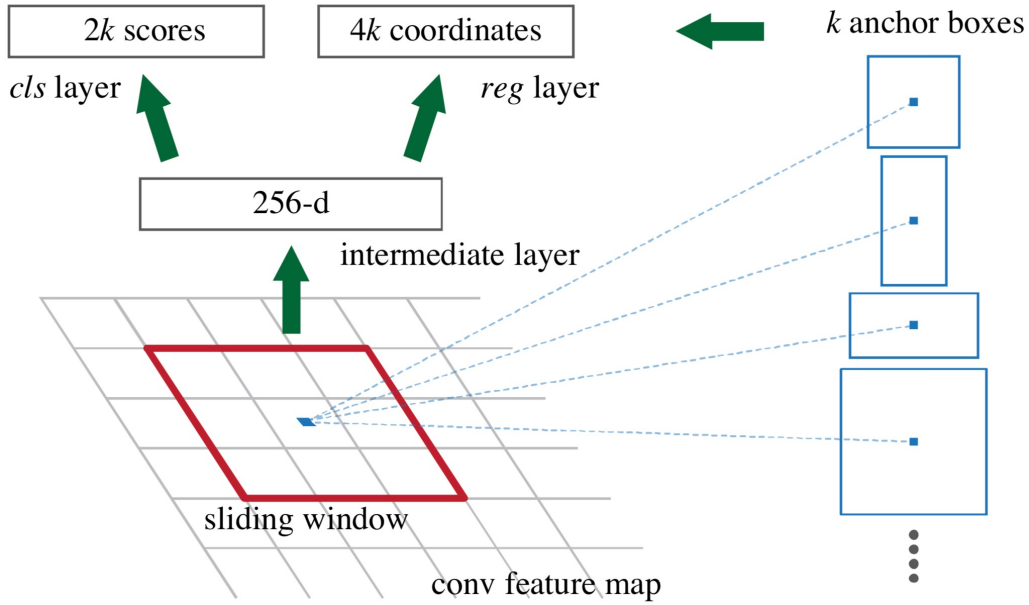
t_i: predicted bounding box

t_i^*: ground truth bounding box

Localisation loss

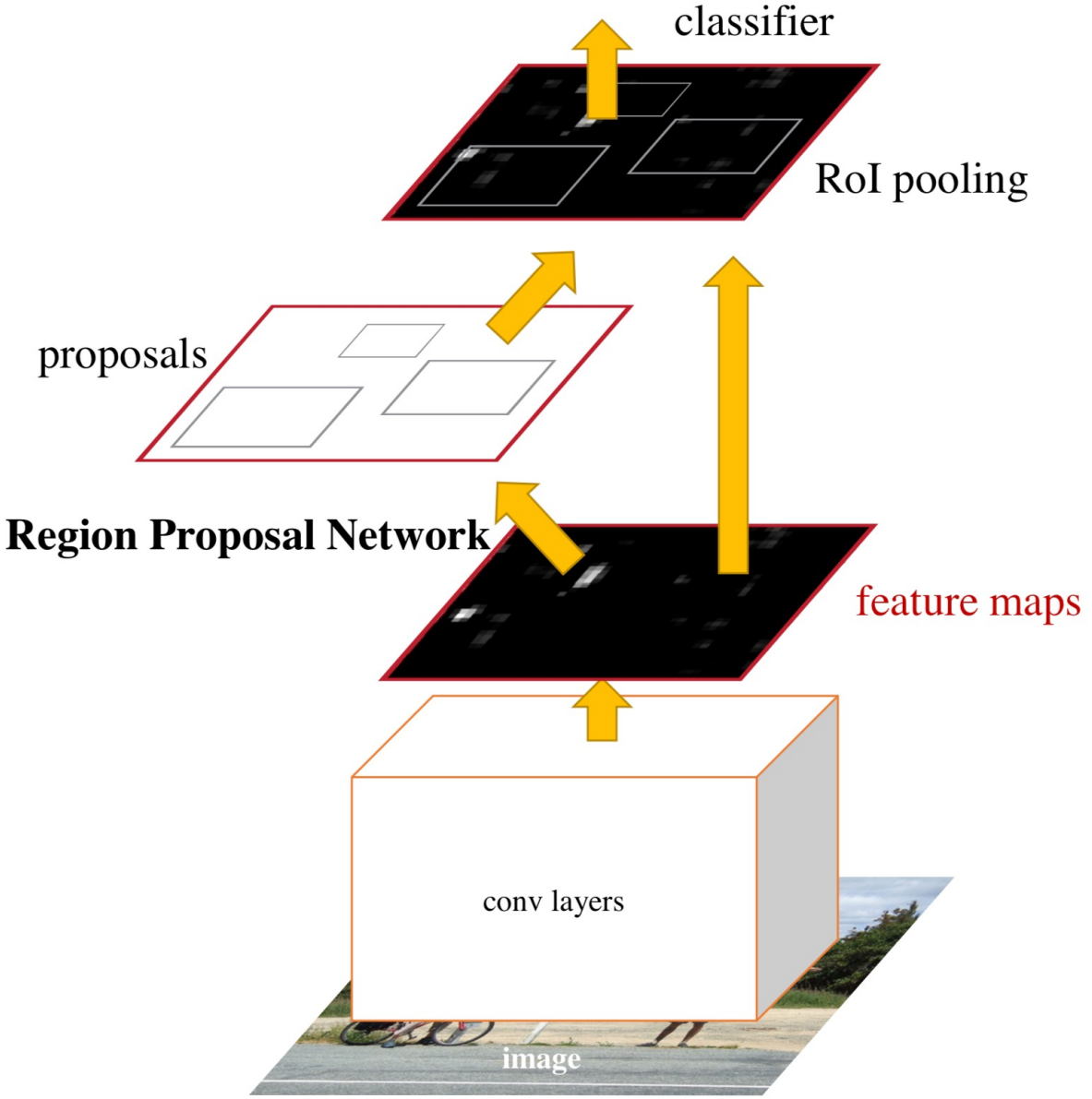
Region proposal network

- In this way, regions of interest are proposed.
- At each location of the convolution feature map, k predictions are made.
  - Classification: whether this anchor is of interest. However, we only know that this region contain interesting objects, we do not know what they are.
  - Regression: how to transform this anchor to be closer to the object size.
- In the next stage, we refine both the classification and localisation.



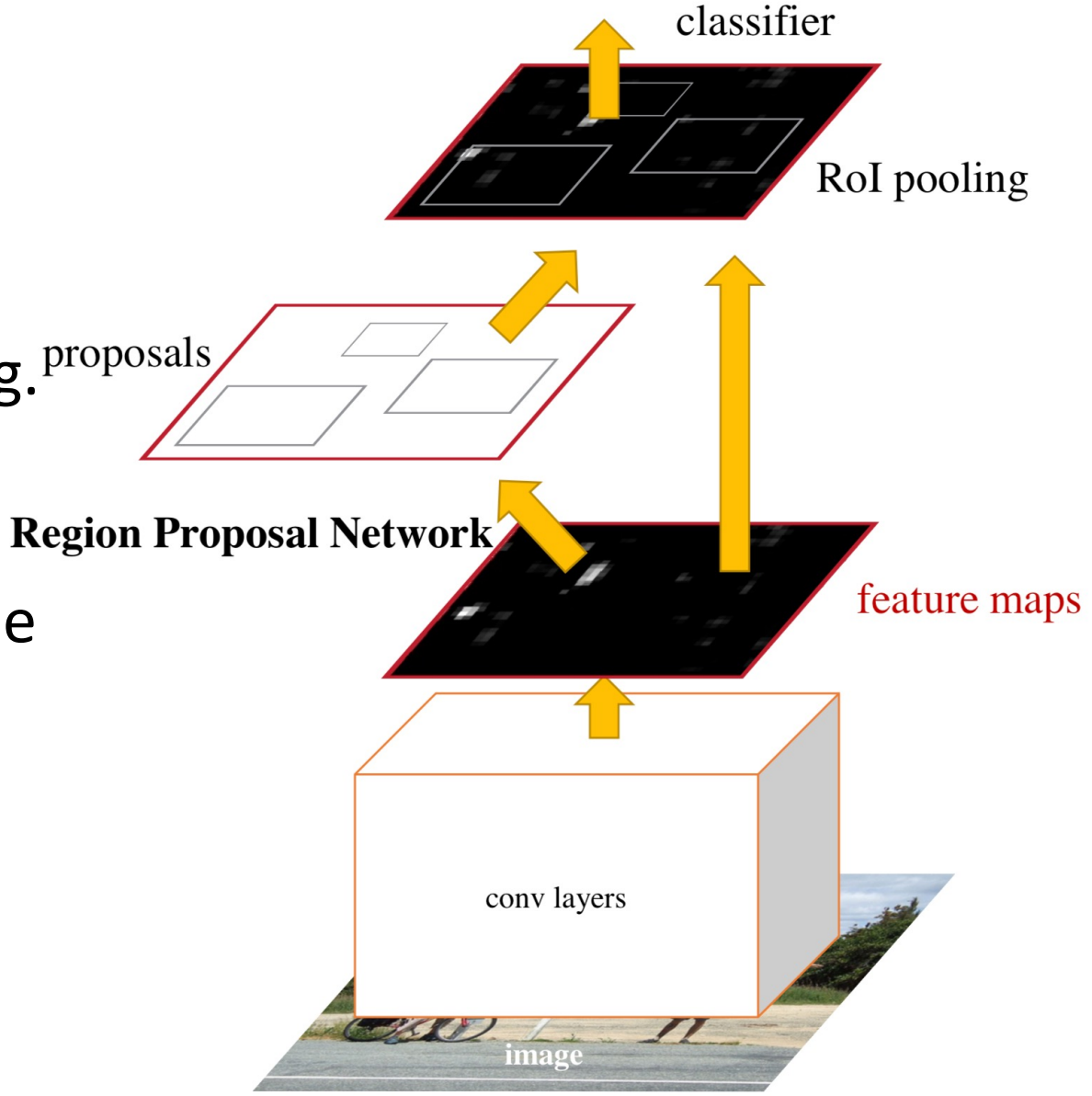
Detection network

- Previously, we look at 3x3 windows on the feature map, searching for objects.
- Now, since we know where they are and their size, why don't we look closer?
  - For each region, we use features from this region (no longer a fixed 3x3 window), known as ROI pooling.
  - We perform classification for this region.



Detection network

- For example, a ROI is 120 x 80 pixels.
- In the ROI pooling layer, we calculate its location and size on the feature map (e.g. 120/16 x 80/16).
- Then we convert the features of this region into a fixed size and provide to the classifier.



Multi-class classifier

- For each ROI, the classifier predicts its label class and refine the bounding box estimate.

$$L(p,t) = L_{cls}(p,y) + \lambda \cdot 1_{y \geq 1} L_{loc}(t,t^*)$$

ground truth, multi-class

Classification loss

Localisation loss

RPN vs Detection network

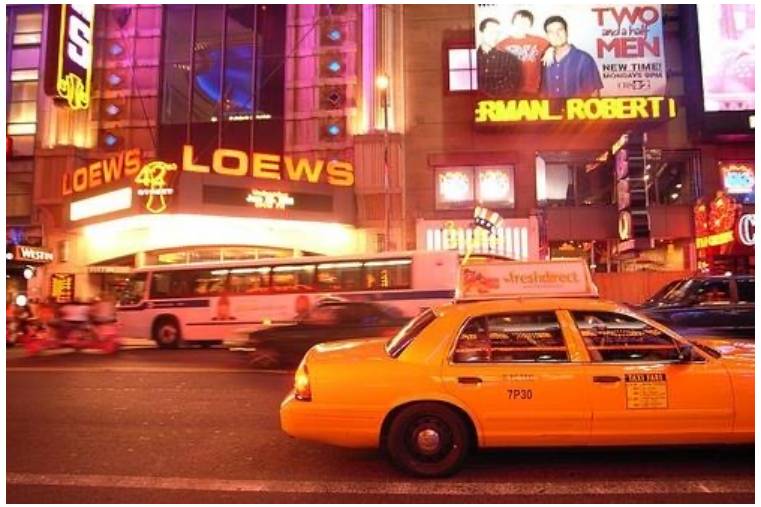
- Looks quite similar. What is the difference between
  - Region proposal network in Stage 1
  - Detection network in Stage 2

- The input to RPN is a 3 x 3 window on conv5 feature map.
- RPN is class-agnostic. It only checks whether this is a ROI or not (binary), providing an objectness score.
- RPN needs to use a lot of anchors, because we do not know what the object looks like yet.

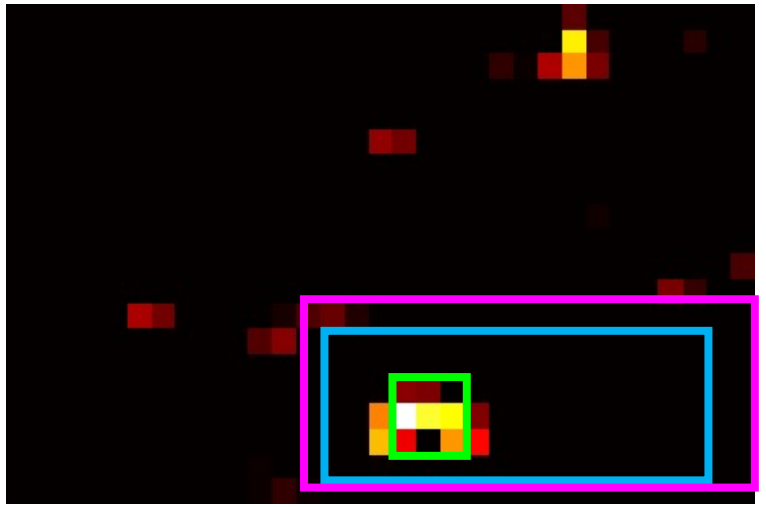
- The input to detection network is a proposed region, thus contains more accurate features.
- The detection network classifies the region into a number of classes.
- The detection network does not need to use anchors. We already know a rough size from the proposal.

RPN vs Detection network

- RPN moves the 3 x 3 sliding window (green), classifies it to be something interesting (binary classification) and proposes a region (blue).
- The detection network takes the features inside the blue region, pools it into a 7 x 7 window, classifies it to be a car (multi-class classification) and refines the bounding box (purple).



Input image



Feature map

Image credit: K. He.

One-stage object detection

- Faster R-CNN is a two-stage object detection method.
  - Stage 1: region proposal.
  - Stage 2: classification and refined localisation.
- Can we do everything in one go?
  - Yes, these are called one-stage object detection methods.
  - Examples: YOLO [1], SSD [2].

[1] J. Redmon et al. You only look once: Unified, real-time object detection. CVPR 2016.
[2] W. Liu et al. SSD: Single shot multibox detector. ECCV 2016.

Region proposal network

- In a two-stage detection method, the loss for RPN is defined as

$$L(p,t) = \sum_{i=1}^{n_{anchor}} \boldsymbol{L_{cls}}(p_i, p_i^*) + \lambda \cdot \sum_{i=1}^{n_{anchor}} 1_{y=1} L_{loc}(t_i, t_i^*)$$

predicted objectness score

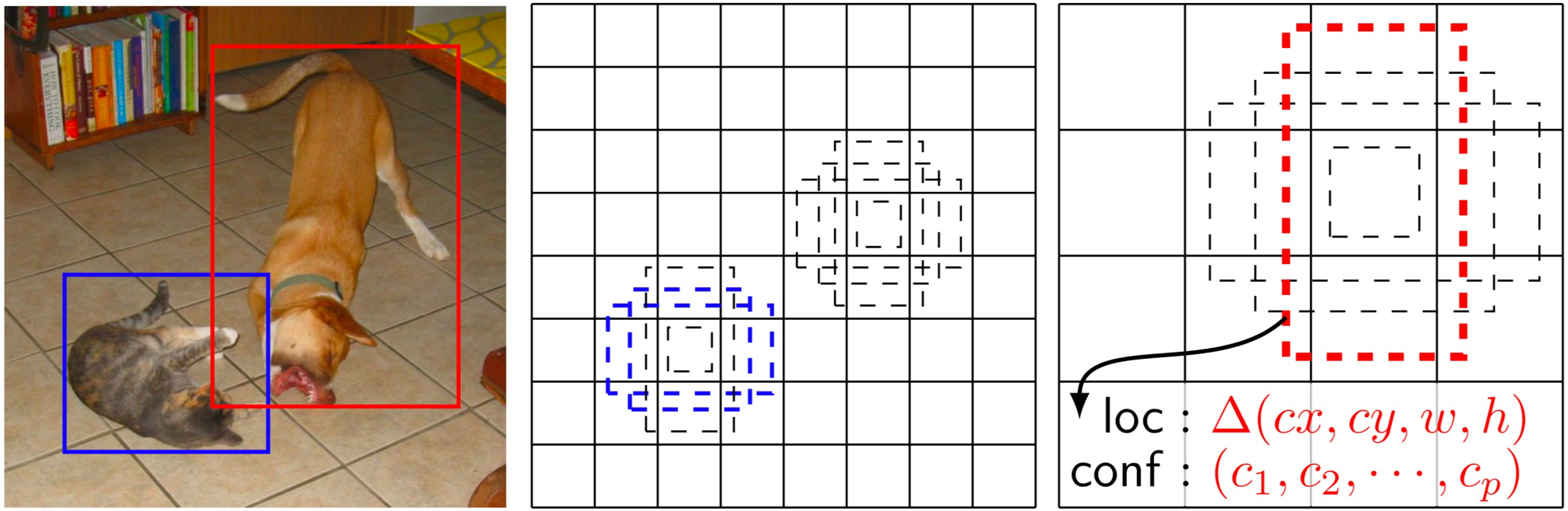
ground truth, **binary** value

predicted bounding box

ground truth bounding box

- Why don't we change this binary classifier into a multi-class classifier?
- For each anchor, we directly predict what object it is.
- SSD is one such example.

Single shot multibox detector (SSD)



(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

(b) At each location on the feature map, SSD evaluates a set of boxes of different scales and aspect ratios (similar to anchors in RPN).

(c) For each box, SSD performs multi-class classification (different from RPN) and localisation.

W. Liu et al. SSD: Single shot multibox detector. ECCV 2016.

One-stage vs Two-stage detection

- Which one performs better?
  - Faster R-CNN is more accurate but slower.
  - SSD is much faster but less accurate.

- Why is Faster R-CNN more accurate?
  - Stage 1: Ah, there is something there. Roughly estimate the region size.
  - Stage 2: Look closely at features in this region to classify and refine.

Object detection performance

- An important component in the detection method is the feature it uses.
- Therefore, the performance of object detection also depends on the backbone network, which provides the convolutional feature map. There are multiple choices for the backbone network.
  - AlexNet
  - VGG
  - ResNet
  - …

J. Huang et al. Speed/accuracy trade-offs for modern convolutional object detectors. CVPR 2017.

AlexNet

[224x224x3] Input
[55x55x96] Conv1, 11x11, 96, s=4
[55x55x96] Norm1
[27x27x96] Pool1
[27x27x256] Conv2, 5x5, 256
[27x27x256] Norm2
[13x13x256] Pool2
[13x13x384] Conv3, 3x3, 384
[13x13x384] Conv4, 3x3, 384
[13x13x256] Conv5, 3x3, 256
[13x13x256] Norm3
[6x6x256] Pool3
[4096] FC1
[4096] FC2
[1000] FC3 (class score)

convolutional feature map

VGG-16

[224x224x3] Input
[224x224x64] 3x3 conv1, 64
[224x224x64] 3x3 conv1, 64
[112x112x64] Pool
[112x112x128] 3x3 conv2, 128
[112x112x128] 3x3 conv2, 128
[56x56x128] Pool
[56x56x256] 3x3 conv3, 256
[56x56x256] 3x3 conv3, 256
[56x56x256] 3x3 conv3, 256
[28x28x256] Pool
[28x28x512] 3x3 conv4, 512
[28x28x512] 3x3 conv4, 512
[28x28x512] 3x3 conv4, 512
[14x14x512] Pool
[14x14x512] 3x3 conv5, 512
[14x14x512] 3x3 conv5, 512
[14x14x512] 3x3 conv5, 512
[7x7x512] Pool
[4096] FC, 4096
[4096] FC, 4096
[1000] FC, 1000

convolutional feature map

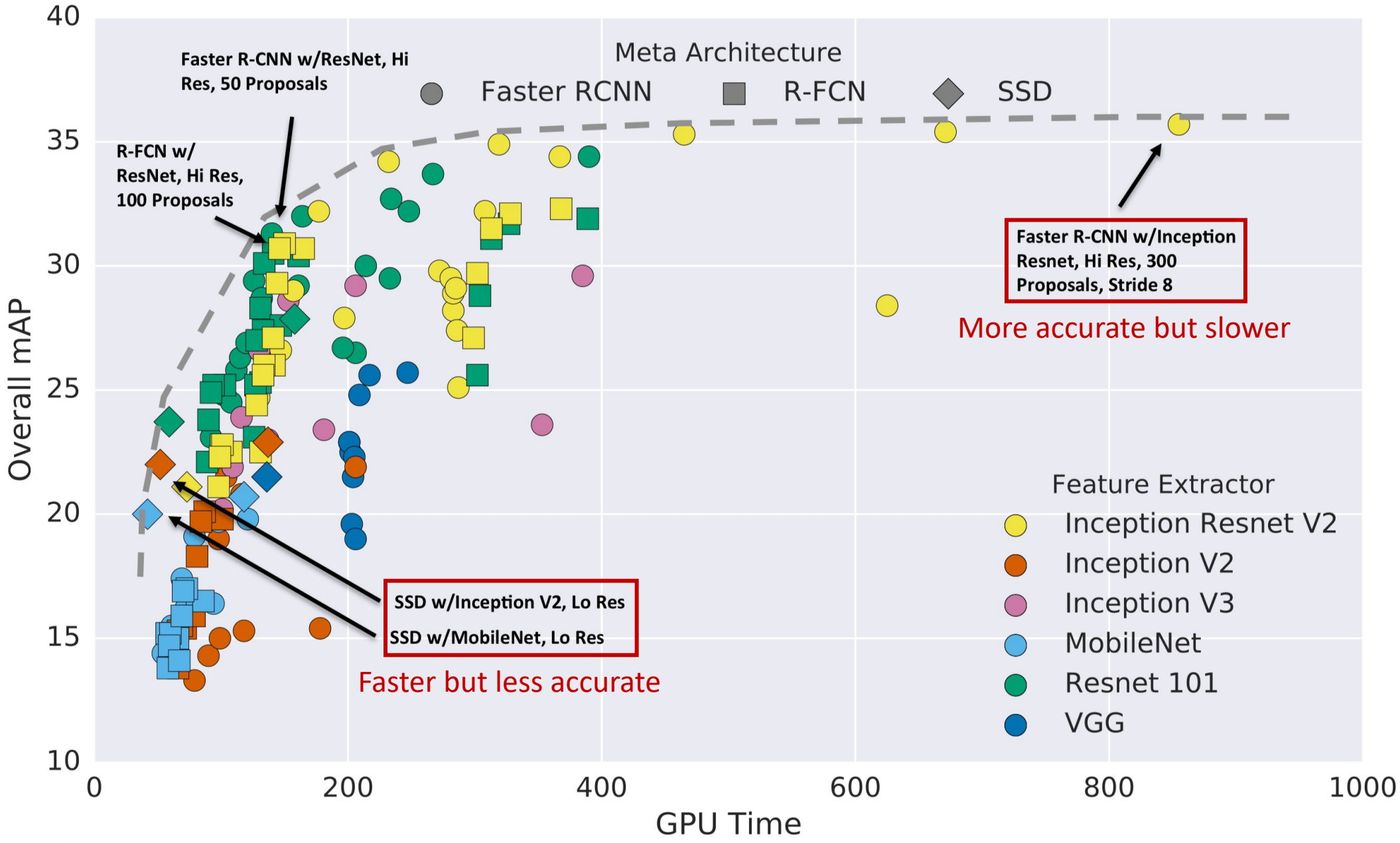
These are called backbone networks, which provide the convolutional feature maps.
They are often pre-trained on ImageNet or other datasets.

Backbone networks

Model	Top-1 accuracy	Num. Params.
VGG-16	71.0	14,714,688
MobileNet	71.1	3,191,072
Inception V2	73.9	10,173,112
ResNet-101	76.4	42,605,504
Inception V3	78.0	21,802,784
Inception Resnet V2	80.4	54,336,736

Table: Properties of 6 backbone networks.
Top-1 accuracy refers to the classification accuracy on ImageNet.

J. Huang et al. Speed/accuracy trade-offs for modern convolutional object detectors. CVPR 2017.



J. Huang et al. Speed/accuracy trade-offs for modern convolutional object detectors. CVPR 2017.

Summary

- Basic object detection ideas
  - Slide a window across the feature map.
  - Utilise the features for classification and localisation.
- One-stage vs two-stage detection methods
- Object detection is an important step for the computer to understand the world. It proposes a number of boxes, each associated with a label class.
- In the next class, we will talk about more detailed understanding of an image, which is semantic segmentation.

References

- Sec. 14.1 Object detection. Richard Szeliski, Computer Vision: Algorithms and Applications (http://szeliski.org/Book).