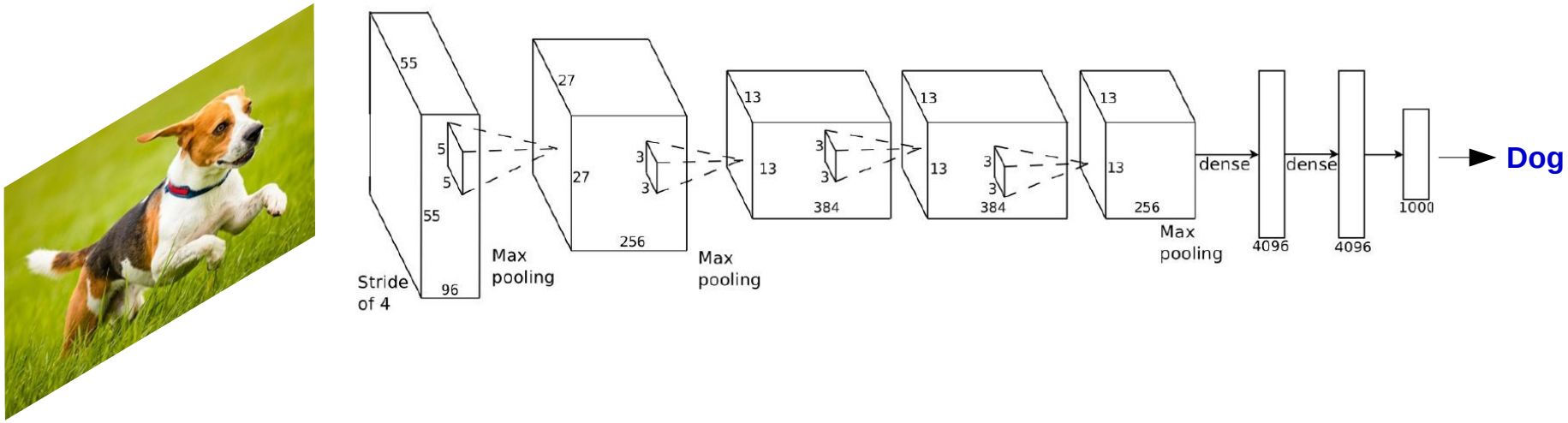
Deep Learning for Visual Recognition: Application to Object Recognition

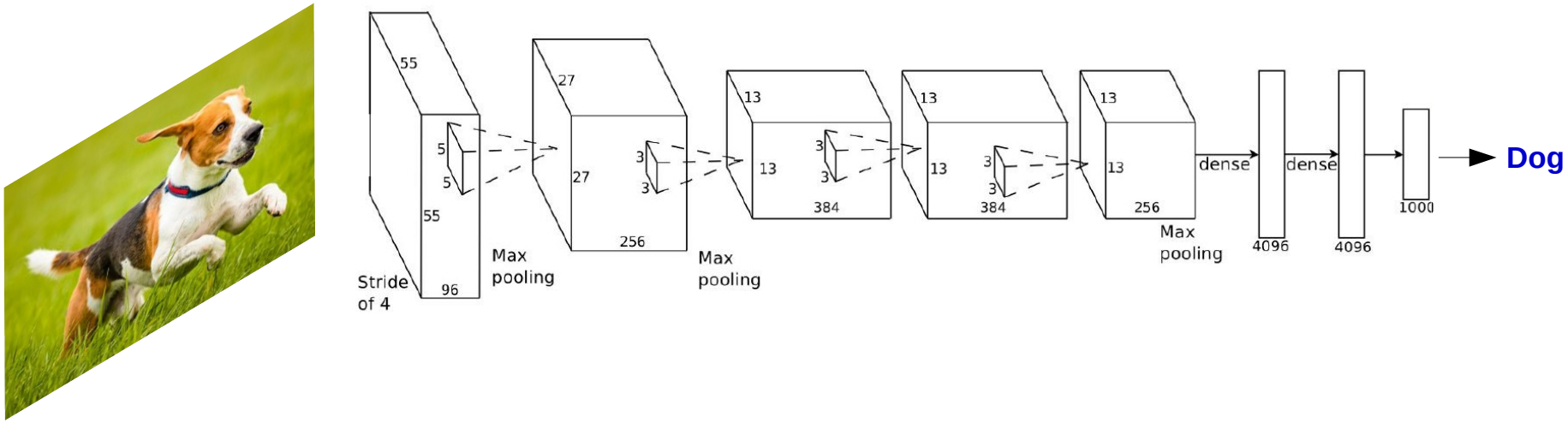
Alvaro Soto

Computer Science Department (DCC), PUC

Object Recognition with CNNs

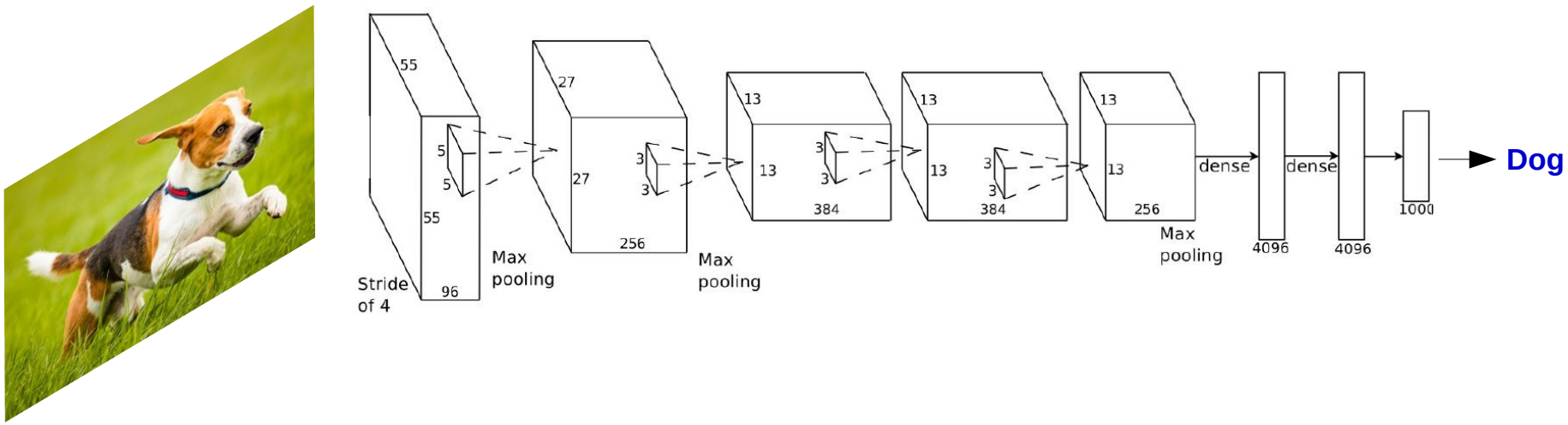


Object Recognition with CNNs



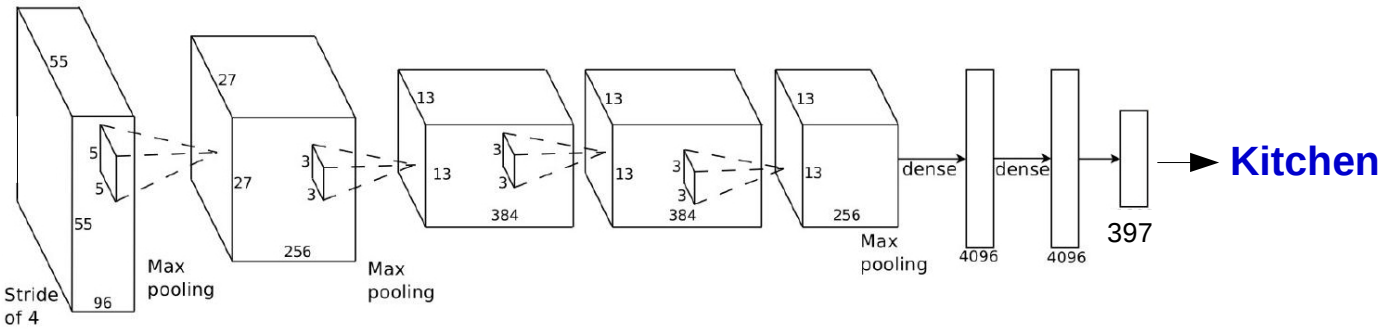
- In the figure, backbone is given by Alex's Net, but we can use other architectures: VGG, ResNet, Inception, etc.

Object Recognition with CNNs

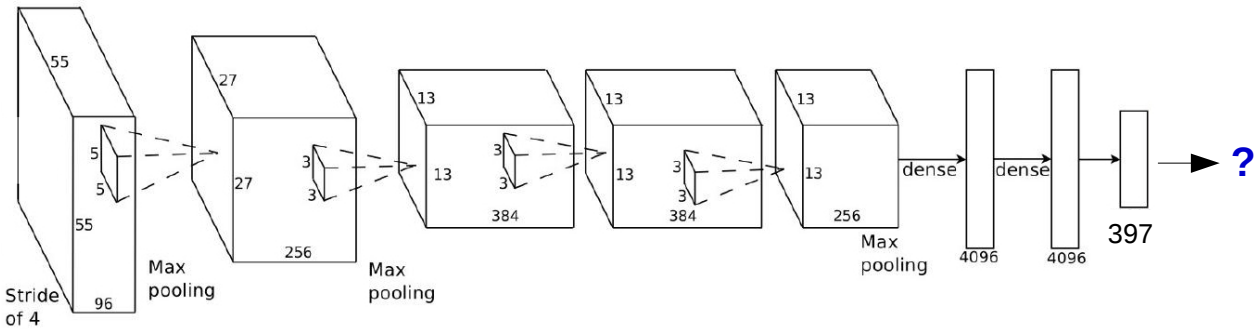


- In the figure, backbone is given by Alex's Net, but we can use other architectures: VGG, ResNet, Inception, etc.
- Output classifier (classification head) has 1000 outputs because it is trained using ImageNet.

What about now?

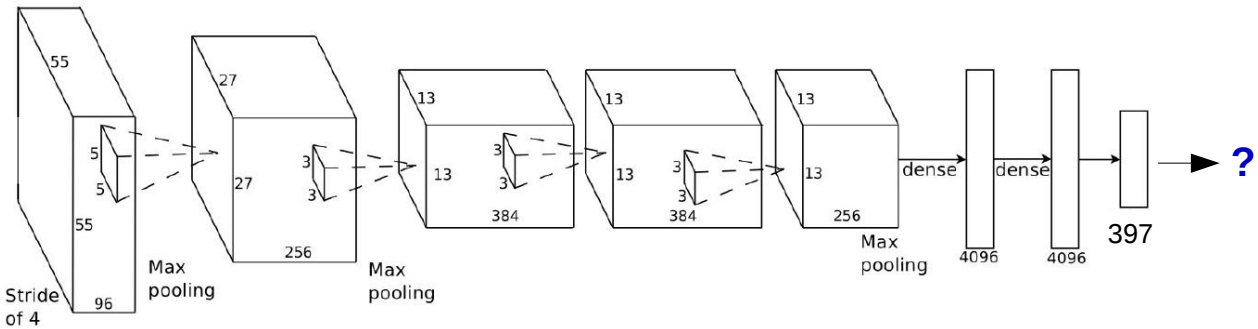


What about now?



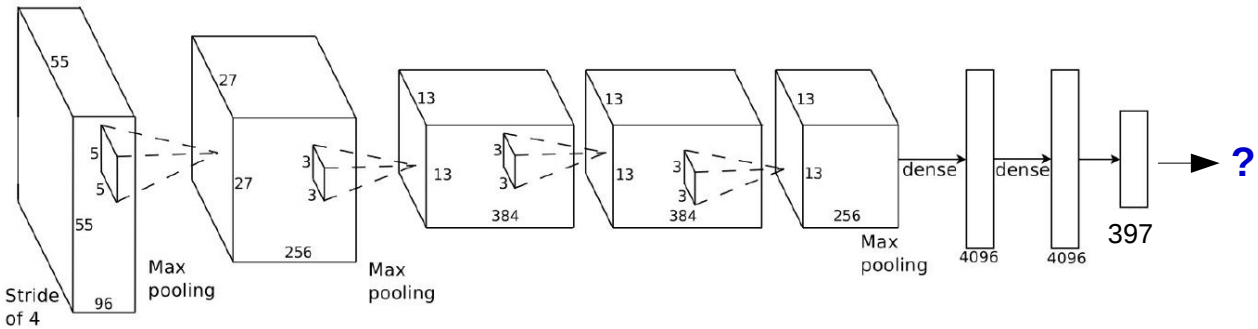
- This is a case of scene recognition with CNNs.

What about now?



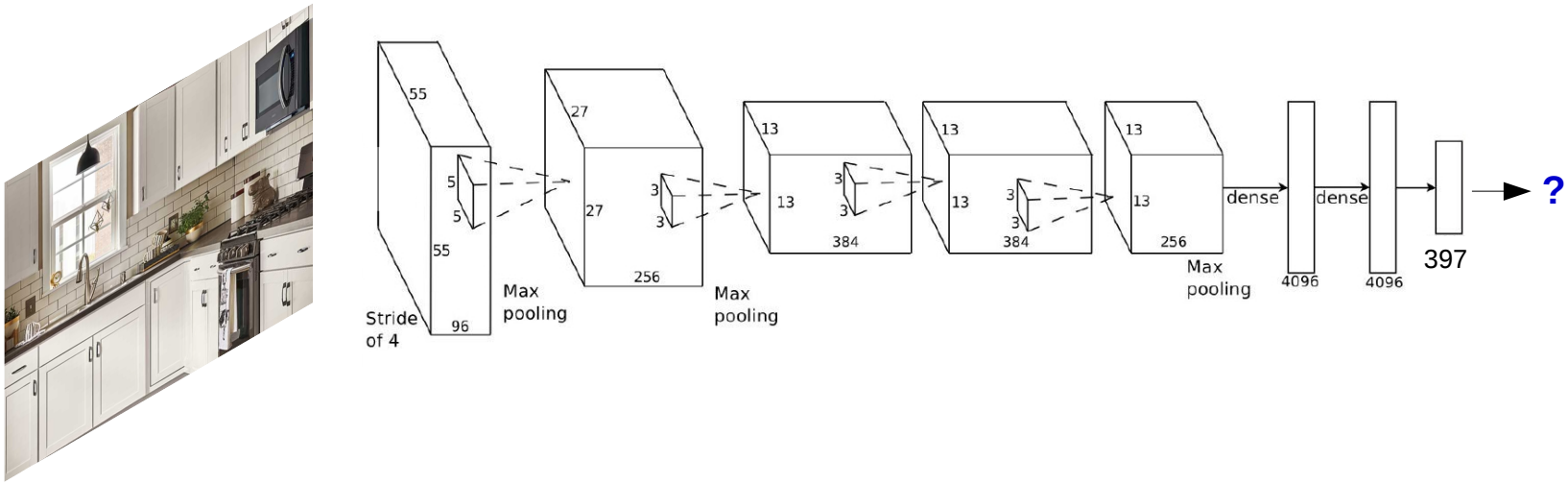
- This is a case of scene recognition with CNNs.
- Head has 397 outputs corresponding to the classes available in the dataset used during training.

What about now?



- This is a case of scene recognition with CNNs.
- Head has 397 outputs corresponding to the classes available in the dataset used during training.

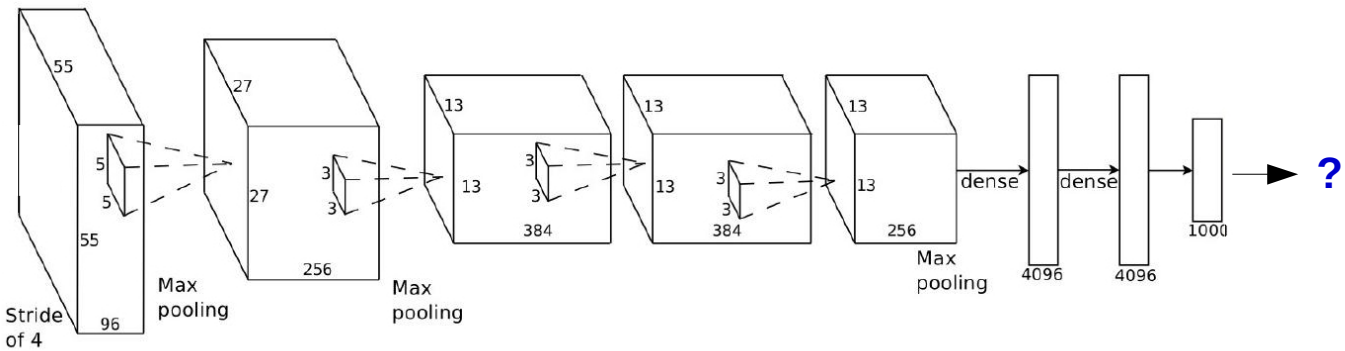
What about now?



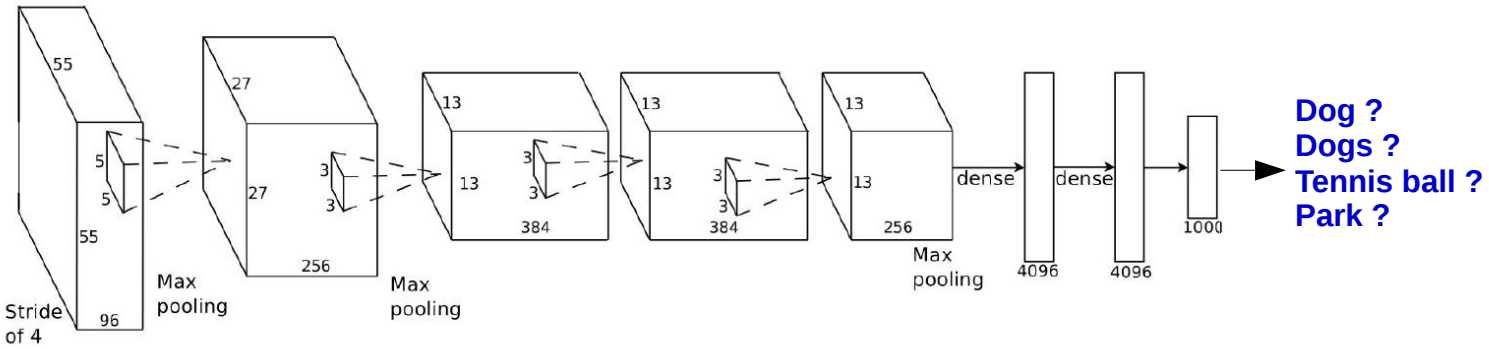
- This is a case of scene recognition with CNNs.
- Head has 397 outputs corresponding to the classes available in the dataset used during training.

Two popular datasets: Sun: `https://groups.csail.mit.edu/vision/SUN/`
Places: `http://places2.csail.mit.edu/`

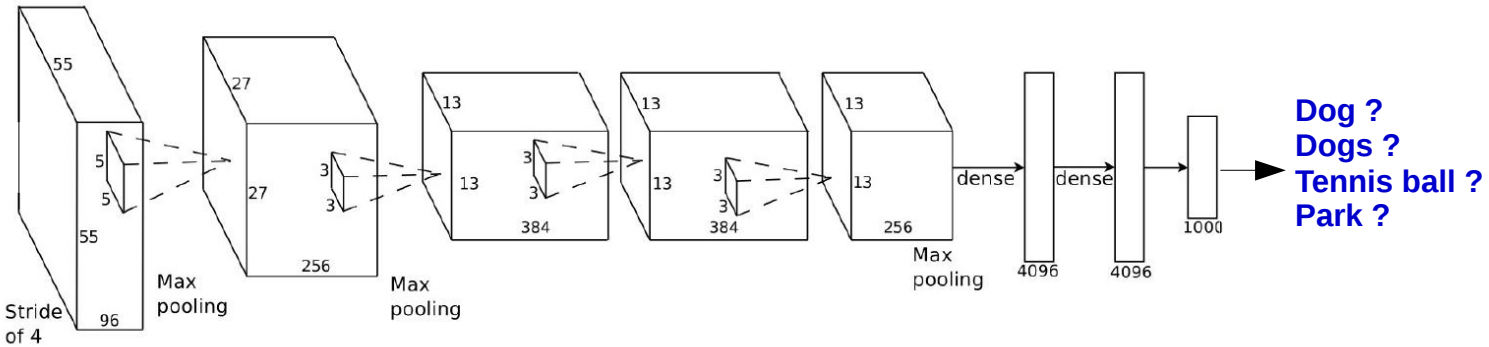
What about now?



What about now?



What about now?

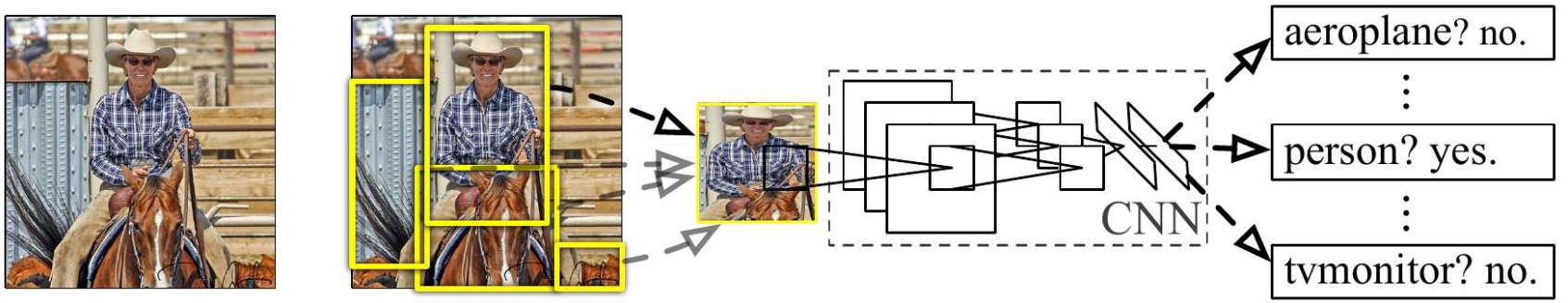


Here, we have a problem, this time we are not looking for a holistic recognition. Instead, we are looking for a region based or local recognition.

Idea:

- Select candidate regions before hand
- Then use CNN to classify each candidate region

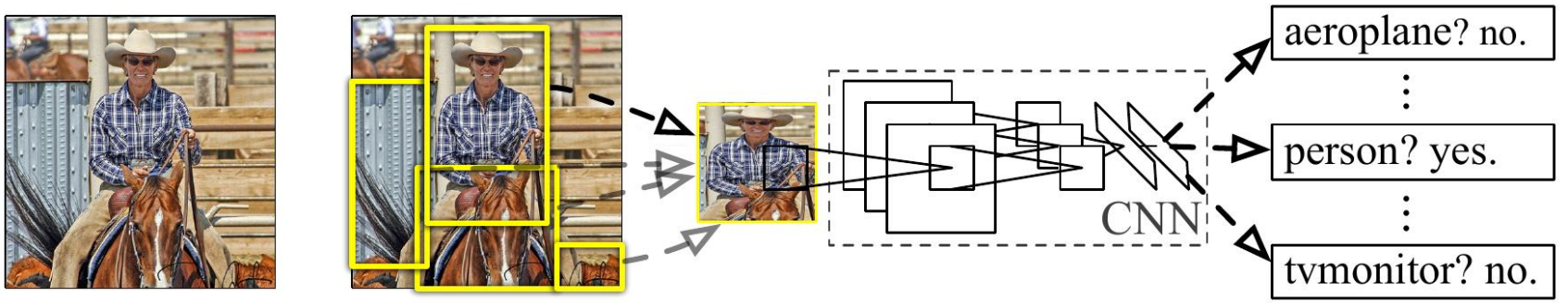
Region based CNNs or R-CNNs (R. Girshick et al., 2014)



Idea:

- Select candidate regions before hand
- Then use CNN to classify each candidate region

Region based CNNs or R-CNNs (R. Girshick et al., 2014)



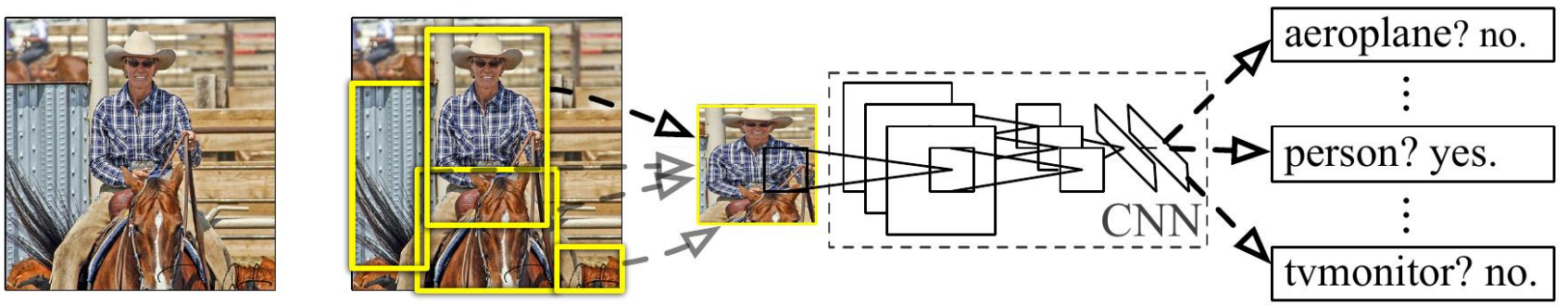
Input Image | Extract region proposals | Compute CNN features | Classify regions

1. How to extract the region proposal?

Idea:

- Select candidate regions before hand
- Then use CNN to classify each candidate region

Region based CNNs or R-CNNs (R. Girshick et al., 2014)



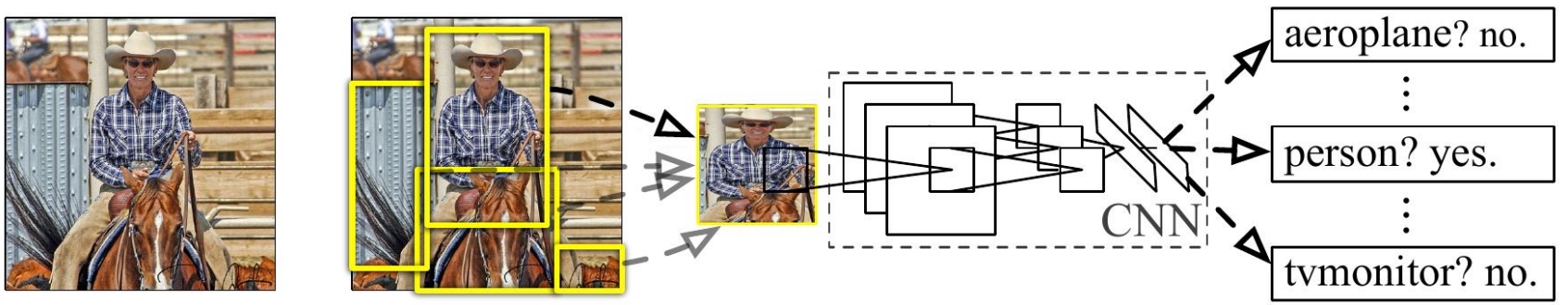
Input Image | Extract region proposals | Compute CNN features | Classify regions

1. How to extract the region proposal?
2. How to manage region size?

Idea:

- Select candidate regions before hand
- Then use CNN to classify each candidate region

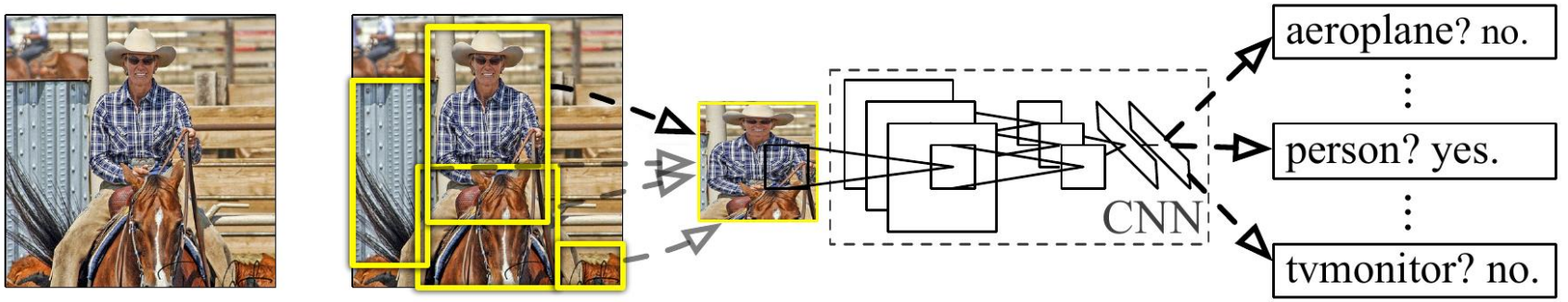
Region based CNNs or R-CNNs (R. Girshick et al., 2014)



Input Image | Extract region proposals | Compute CNN features | Classify regions

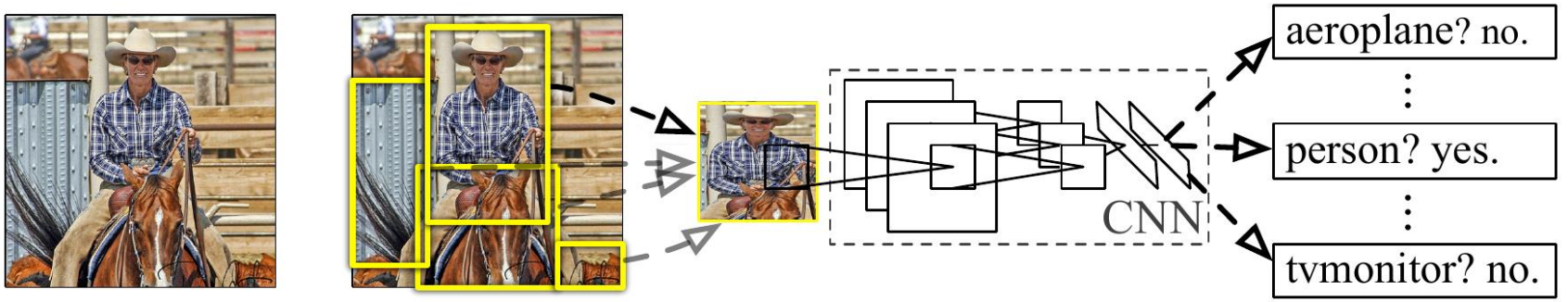
1. How to extract the region proposal?
2. How to manage region size?
3. How can I train a classifier?

Region based CNNs or R-CNNs (R. Girshick et al., 2014)



1. How to extract the region proposal?

Region based CNNs or R-CNNs (R. Girshick et al., 2014)



1. How to extract the region proposal?

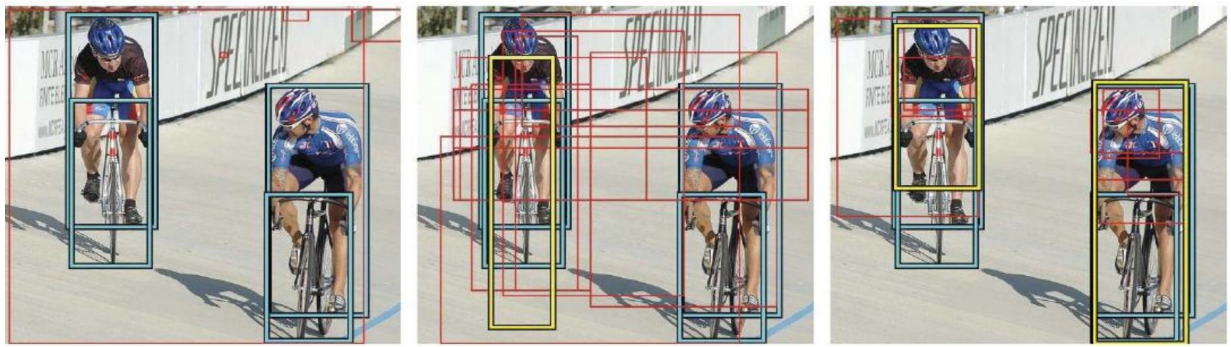
Use an external technique to identify bounding boxes with objects (objectness technique).

Objectness technique

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 34, NO. 11, NOVEMBER 2012 2189

Measuring the Objectness of Image Windows

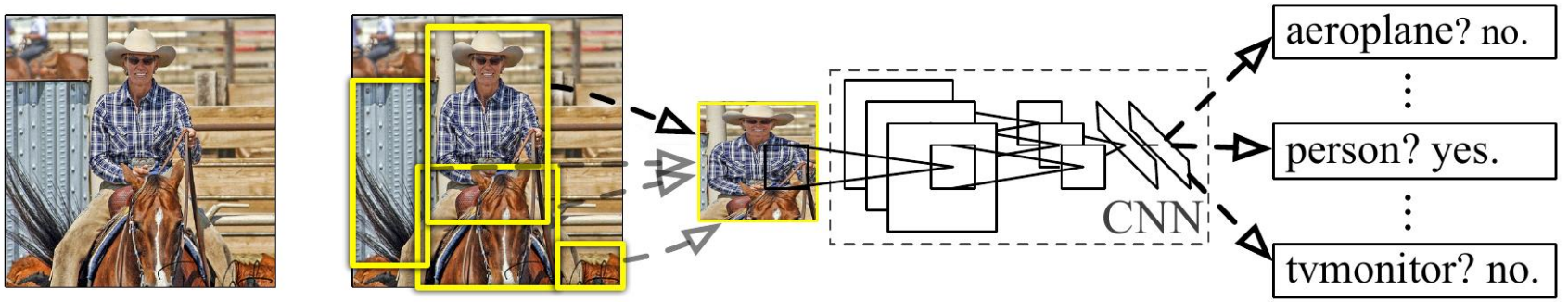
Bogdan Alexe, *Student Member*, *IEEE*, Thomas Deselaers, *Member*, *IEEE*, and Vittorio Ferrari, *Member*, *IEEE*



There are several alternative techniques:

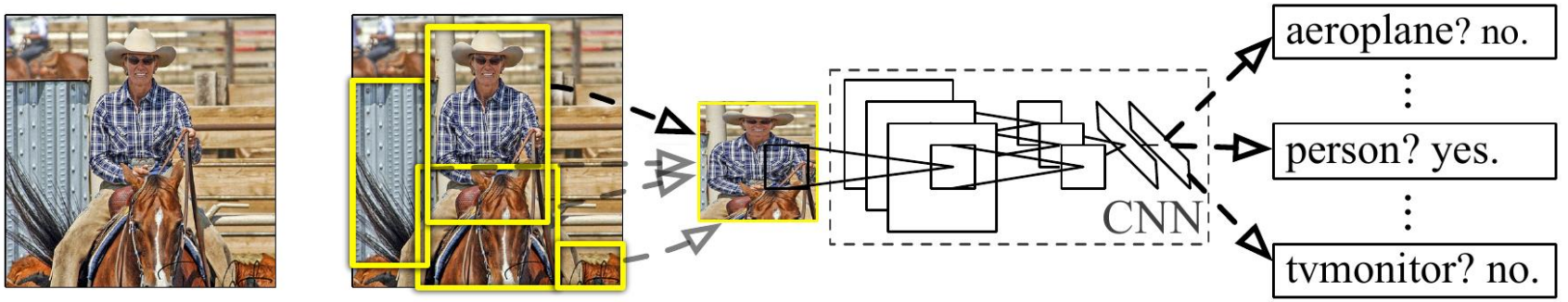
- Selective Search (van de Sande, Uijlings et al.)
- Objectness (Alexe et al.)
- Category independent object proposals (Endres & Hoiem)
- CPMC (Carreira & Sminchisescu)

Region based CNNs or R-CNNs (R. Girshick et al., 2014)



2. How to manage region size?

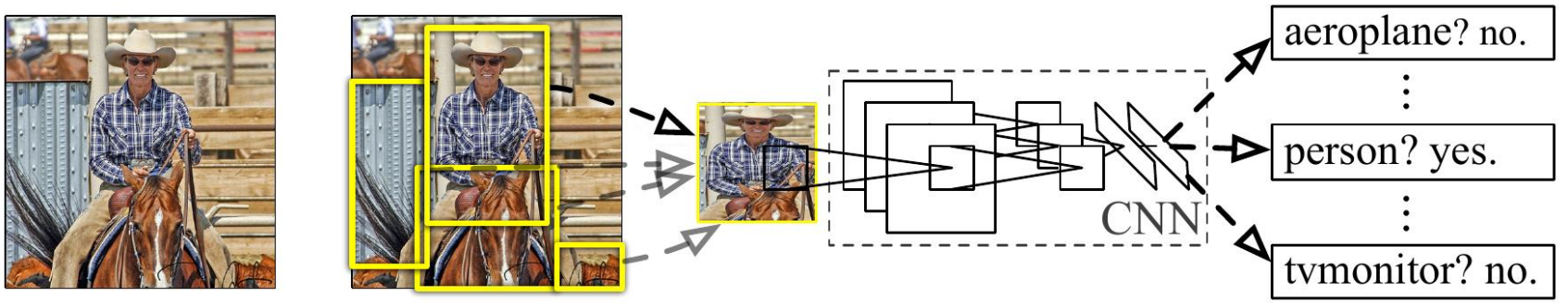
Region based CNNs or R-CNNs (R. Girshick et al., 2014)



2. How to manage region size?

Scale all regions to size used by target CNN (ex. 227x277 pixels).

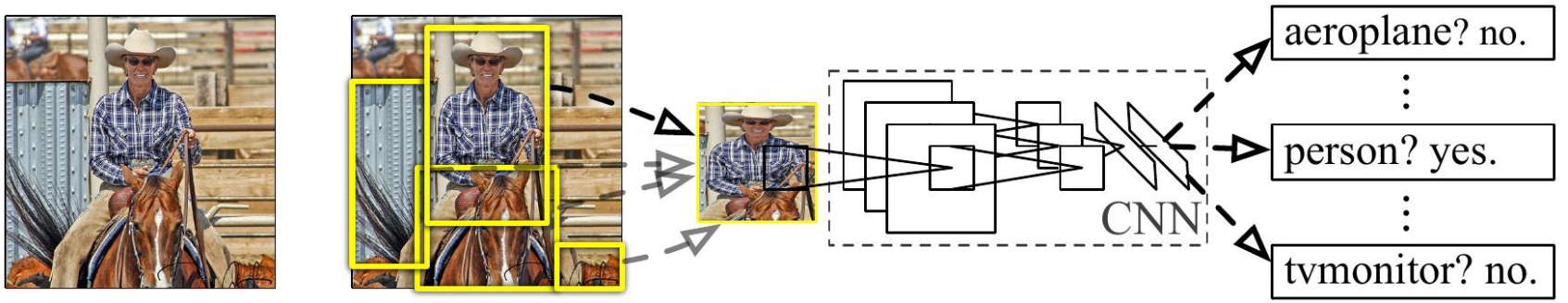
Region based CNNs or R-CNNs (R. Girshick et al., 2014)



Input Image | Extract region proposals | Compute CNN features | Classify regions

3. How can I train a classifier?

Region based CNNs or R-CNNs (R. Girshick et al., 2014)



3. How can I train a classifier?
   - Adapt the head of the CNN and train FC layers.
   - As an alternative, use pre-trained CNN to extract features. Then train an independent classifier such as a SVM.

What about object level datasets?

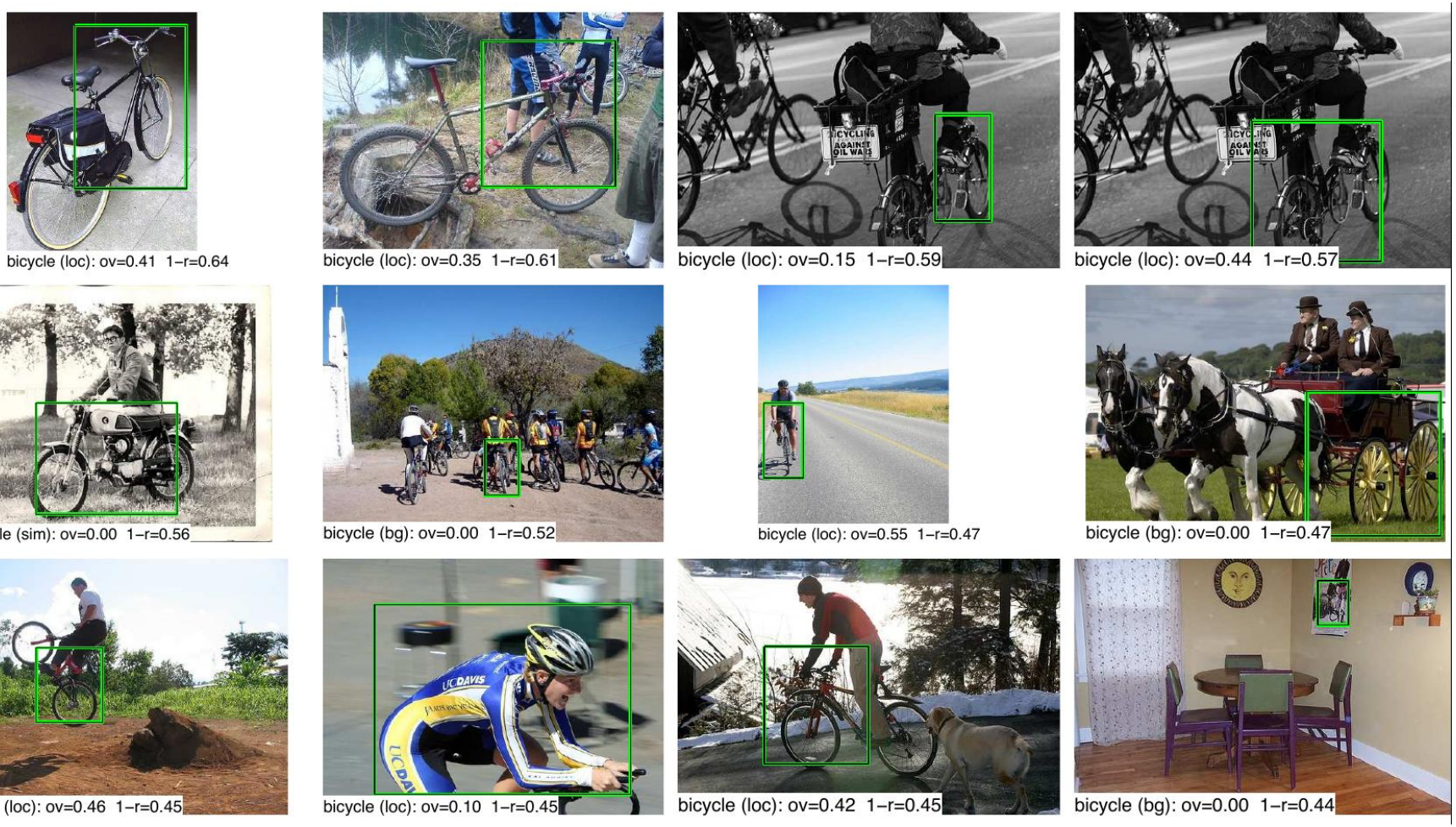
Pascal Dataset

Visual Object Classes Challenge 2012 (VOC2012)

PASCAL2
Pattern Analysis, Statistical Modelling and Computational Learning

`http://host.robots.ox.ac.uk/pascal/VOC/`

Results on Pascal dataset



Performance not so good, around 60% recognition accuracy.
Furthermore, Pascal dataset was very limited, just 20 object classes.

Can we develop an object recognition pipeline using only a CNN ?

Can we develop an object recognition pipeline using only a CNN ?

- Need a suitable dataset:

Can we develop an object recognition pipeline using only a CNN ?

- Need a suitable dataset:
  - Large size

Can we develop an object recognition pipeline using only a CNN ?

- Need a suitable dataset:
  - Large size
  - Several object categories

Can we develop an object recognition pipeline using only a CNN ?

- Need a suitable dataset:
  - Large size
  - Several object categories
  - Object level bounding boxes

Can we develop an object recognition pipeline using only a CNN ?

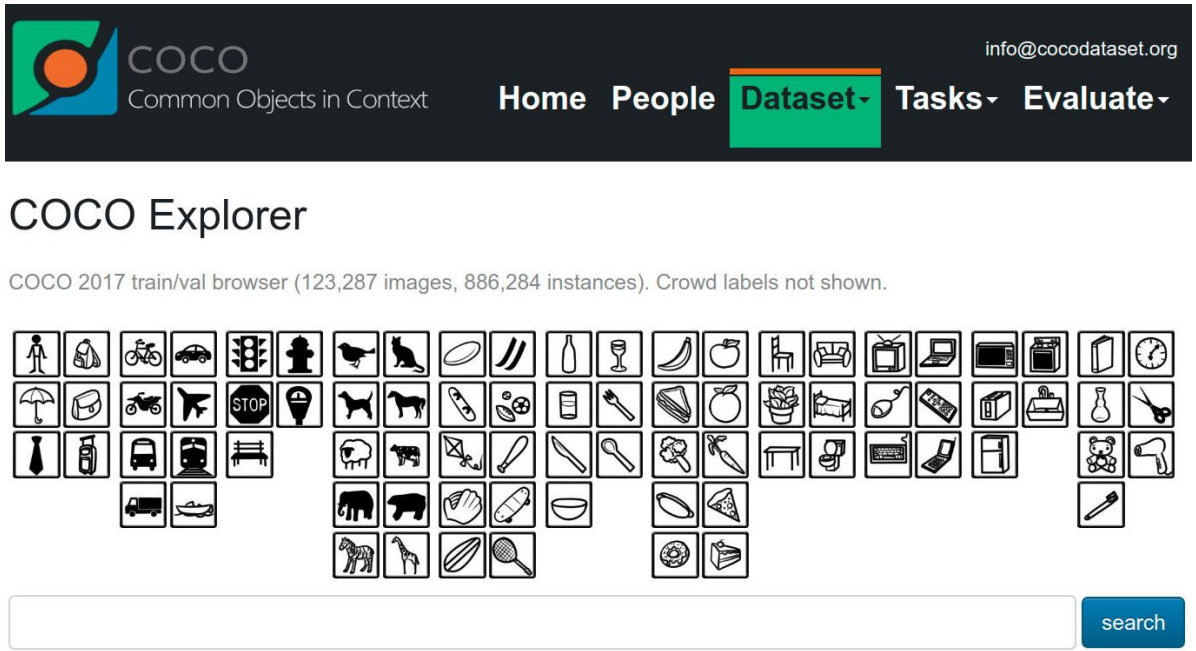
- Need a suitable dataset:
  - Large size
  - Several object categories
  - Object level bounding boxes
- Need a new CNN arquitecture:

Can we develop an object recognition pipeline using only a CNN ?

- Need a suitable dataset:
  - Large size
  - Several object categories
  - Object level bounding boxes
- Need a new CNN arquitecture:
  - Fully convolutional neural networks (FCNs)

Dataset with object level supervision

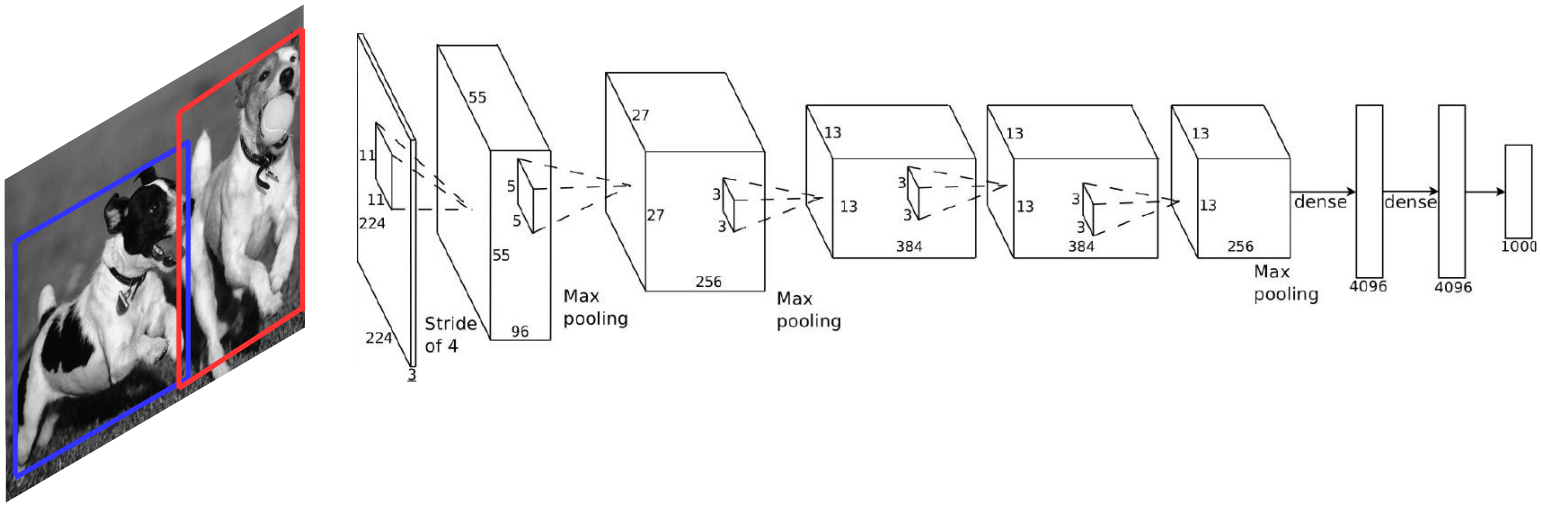
New Dataset Example: Microsoft-Coco



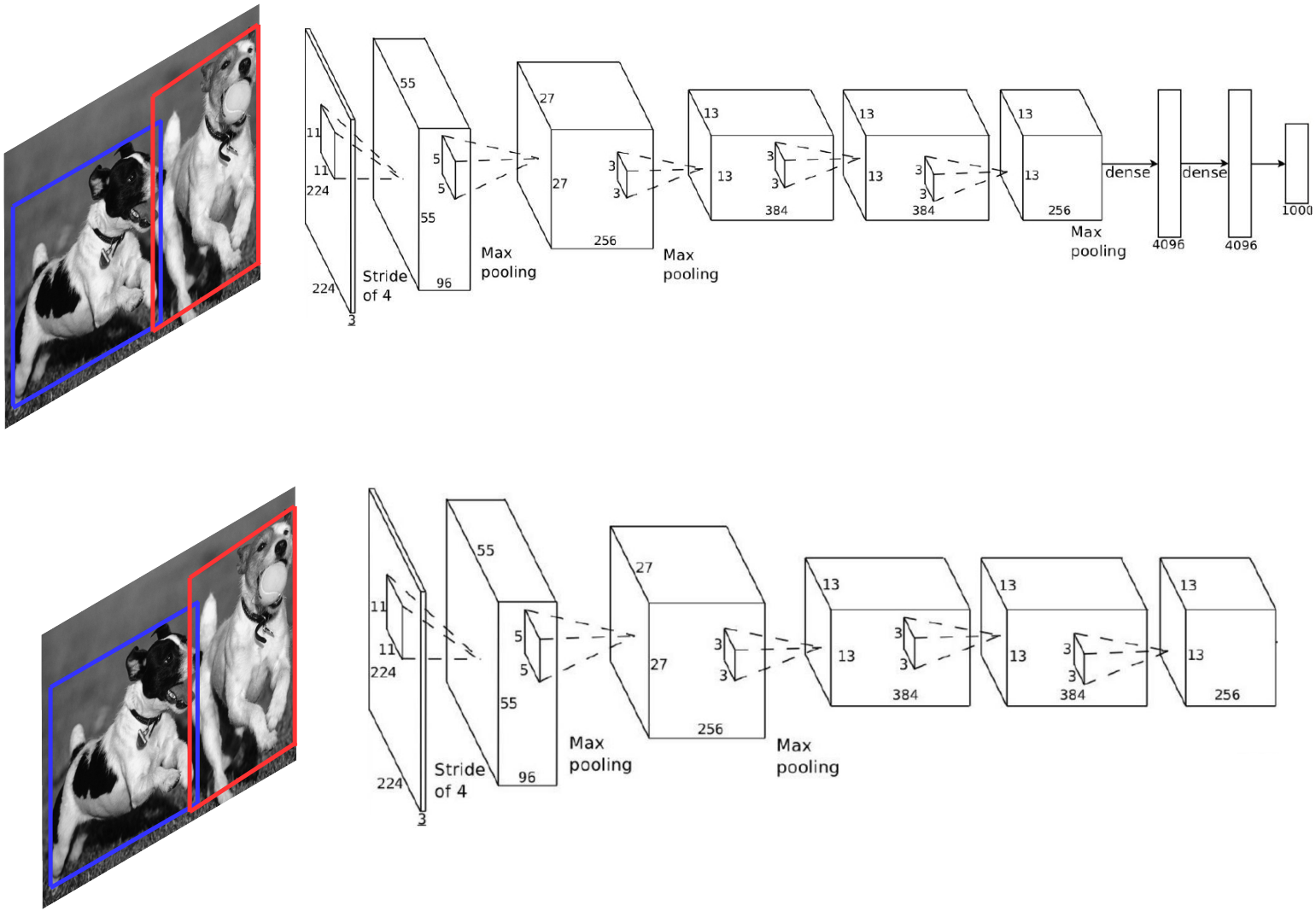
https://cocodataset.org

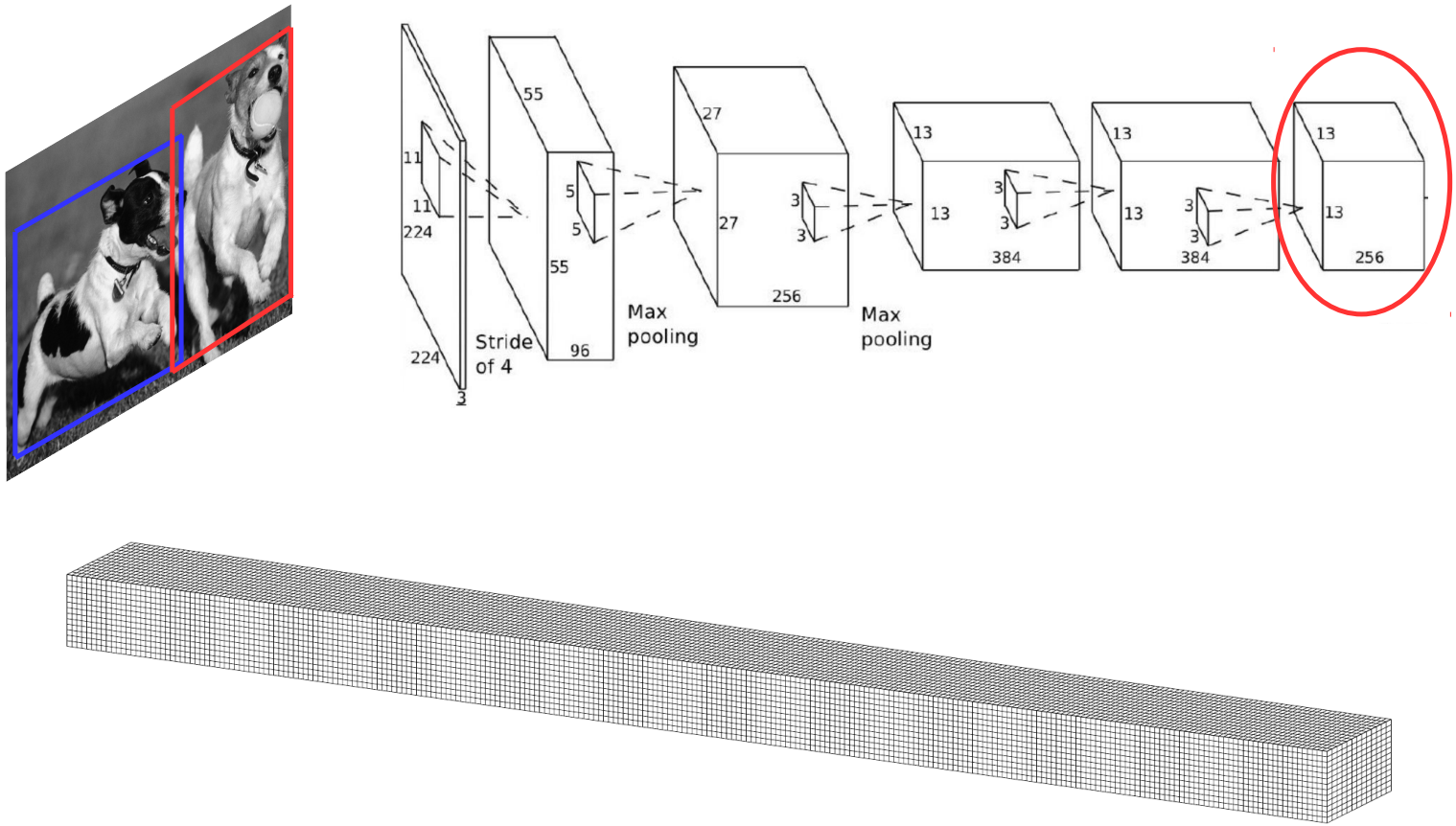
A fully convolutional CNN

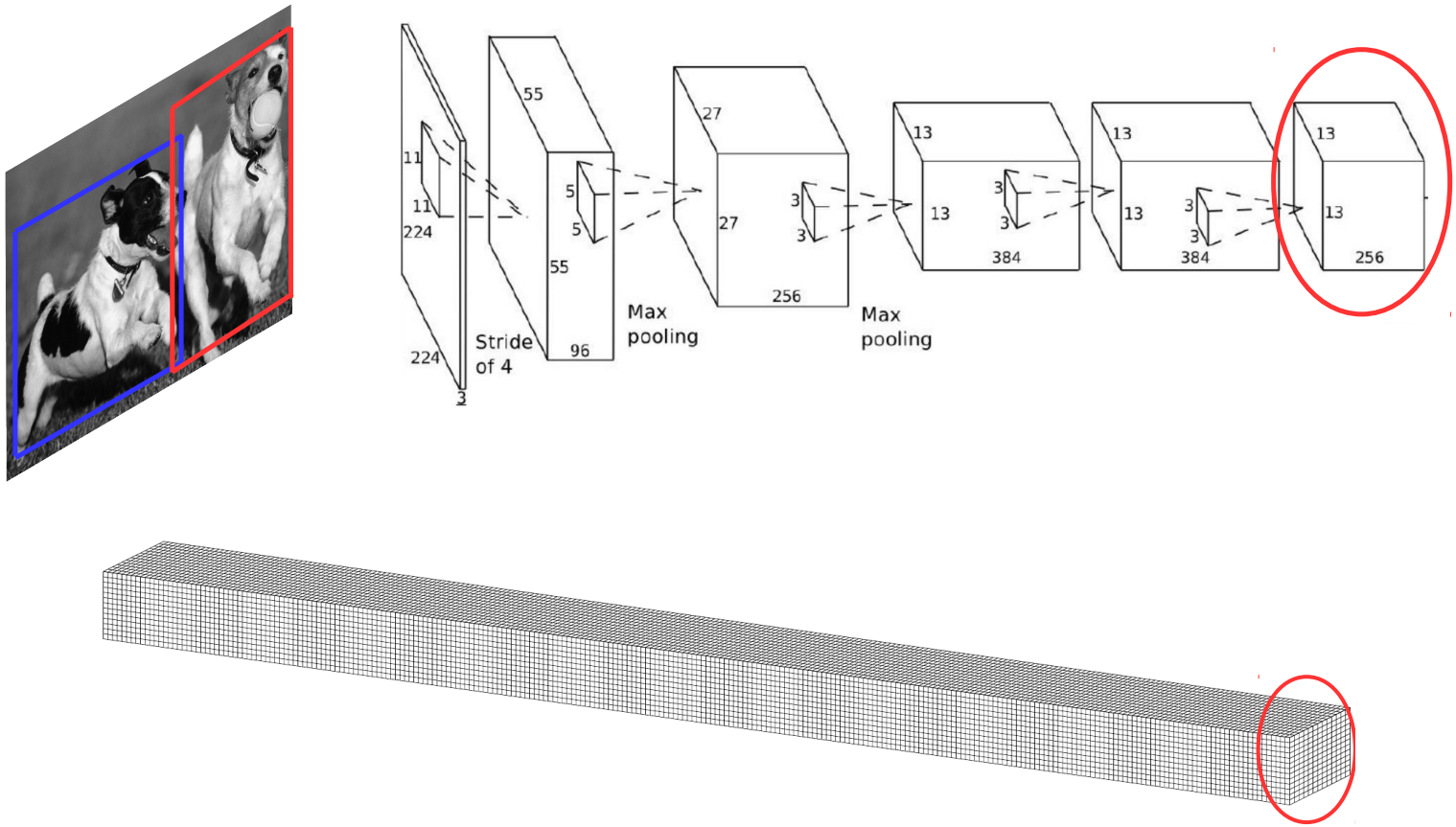
Fully Convolutional NNs (FCNs)

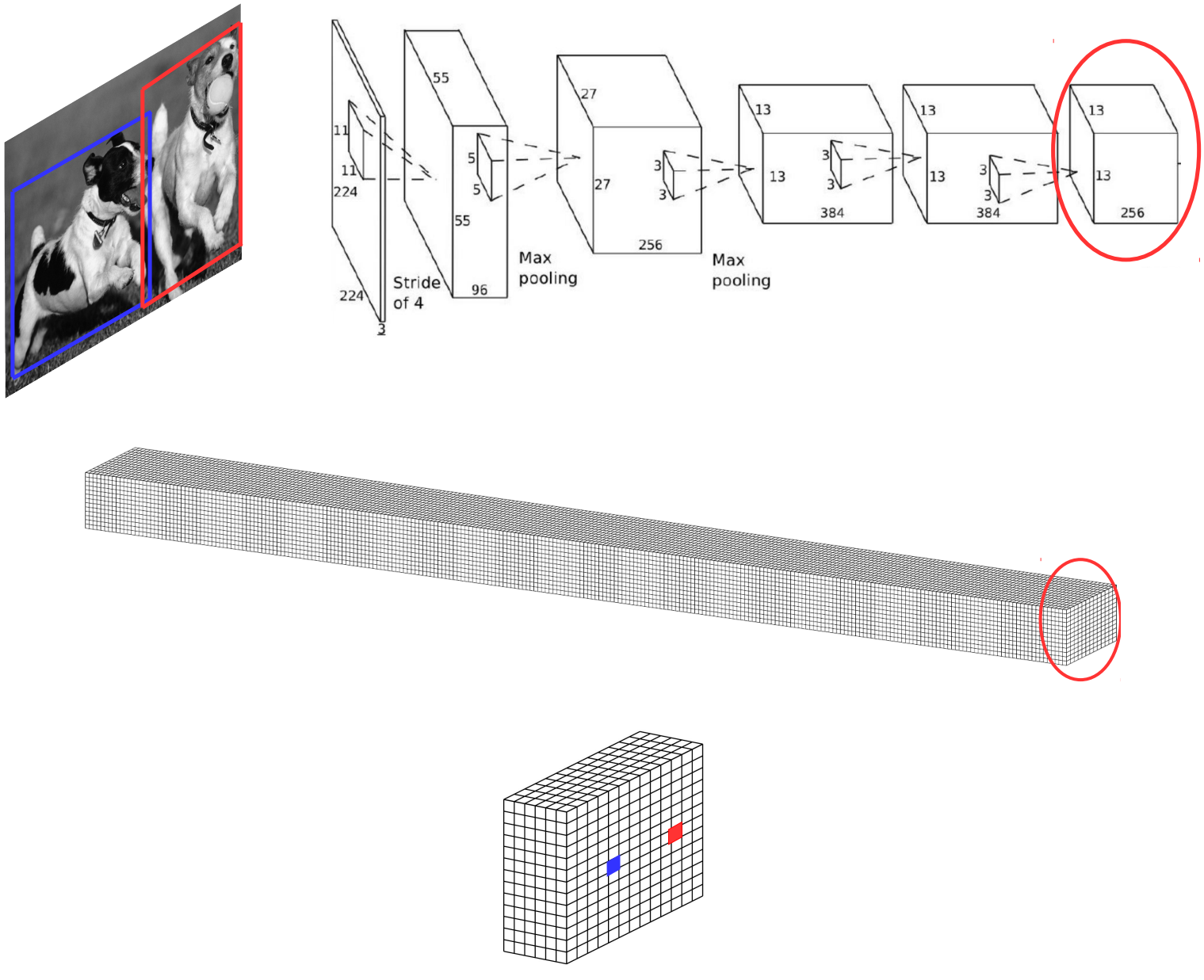


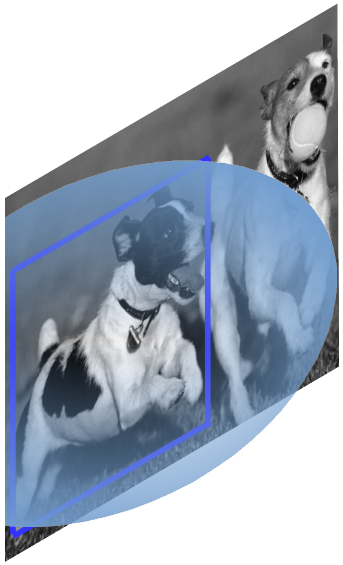
Fully Convolutional NNs (FCNs)

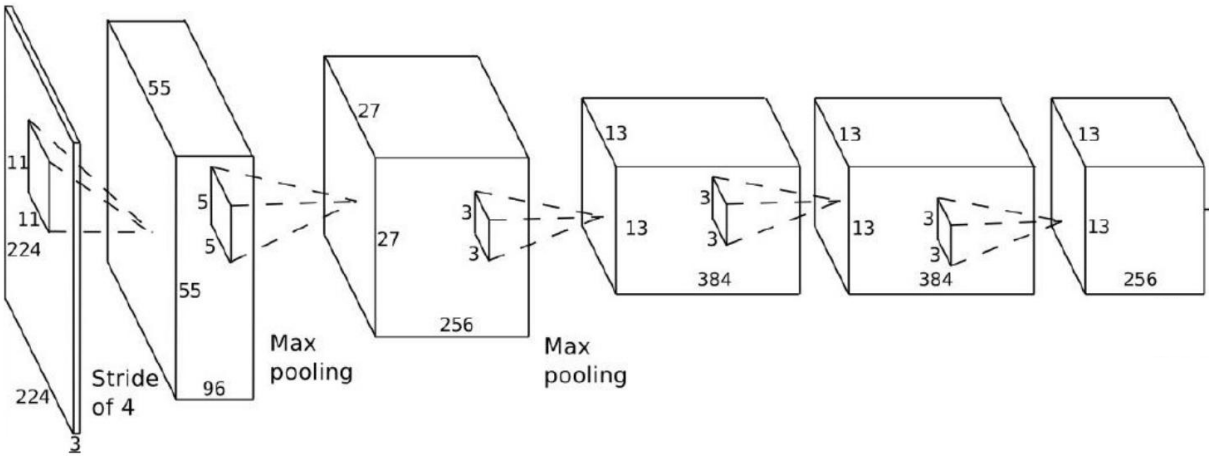


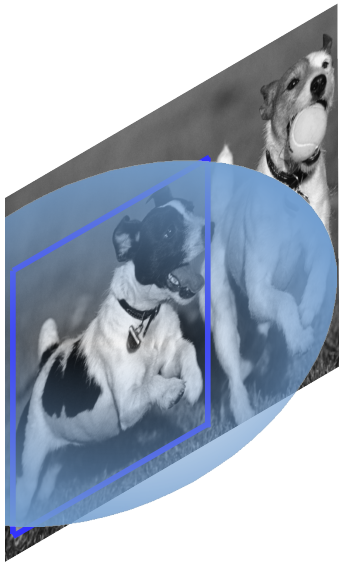

11
11
224
224
3
Stride
of 4
55
55
96
5
5
Max
pooling
27
27
256
3
3
Max
pooling
13
13
384
3
3
13
13
384
3
3
13
13
256

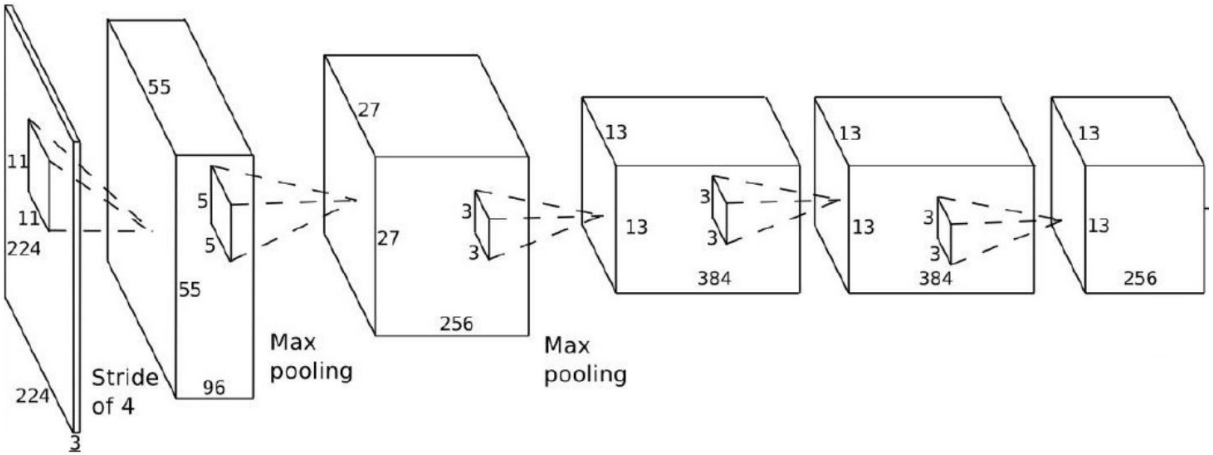

11
11
224
224
3
55
55
5
5
96
Stride
of 4
27
27
256
3
3
Max
pooling
13
13
384
3
3
Max
pooling
13
13
384
3
3
13
13
256

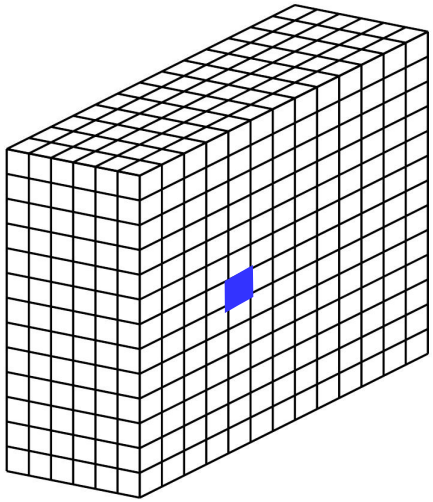

11
11
224
224
3
55
5
5
55
96
Stride
of 4
27
27
3
3
256
Max
pooling
13
13
3
3
384
Max
pooling
13
13
3
3
384
13
13
256

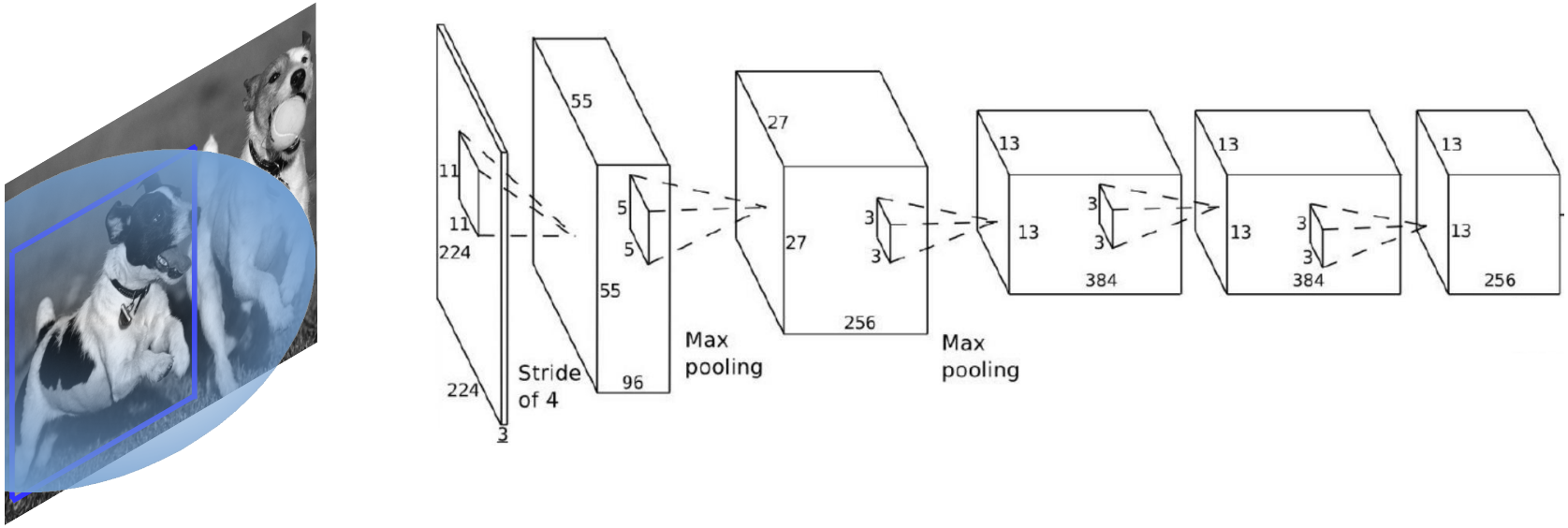



11
11
224
224
3
55
55
5
5
96
Stride
of 4
27
27
3
3
256
Max
pooling
13
13
3
3
384
Max
pooling
13
13
3
3
384
13
13
256

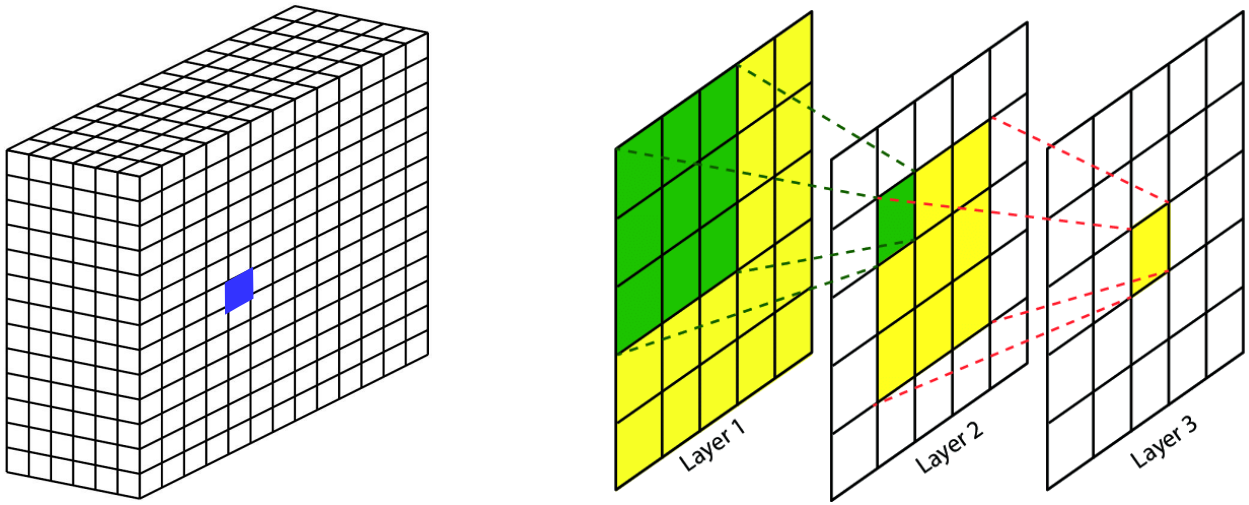


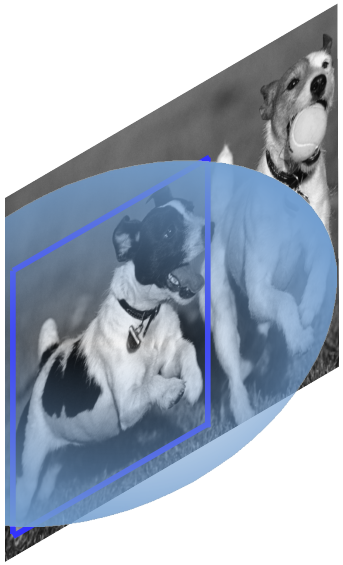

11
11
224
224
3
55
55
96
5
5
Stride
of 4
27
27
256
3
3
Max
pooling
13
13
384
3
3
Max
pooling
13
13
384
3
3
13
13
256

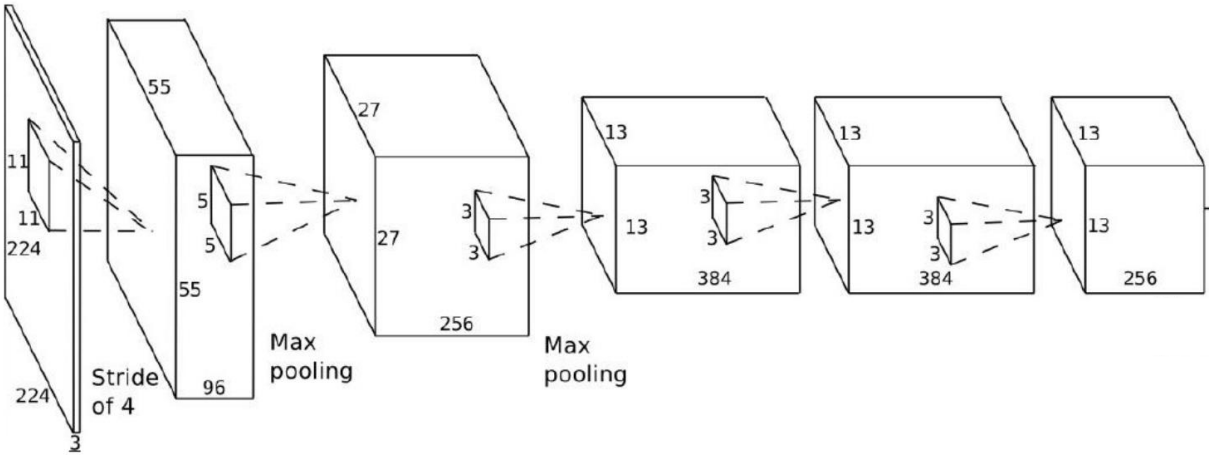


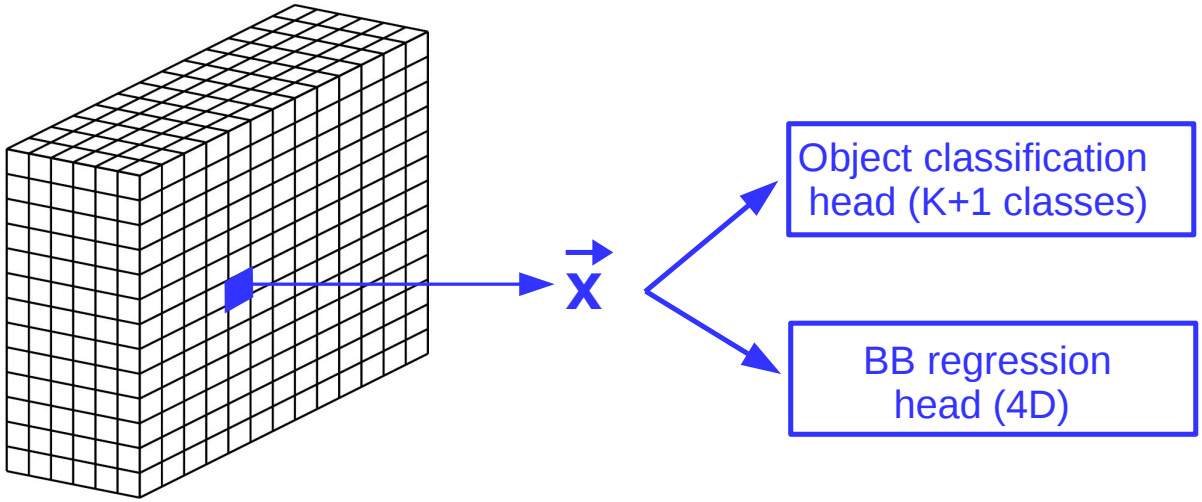


Recall the receptive field of a neuron

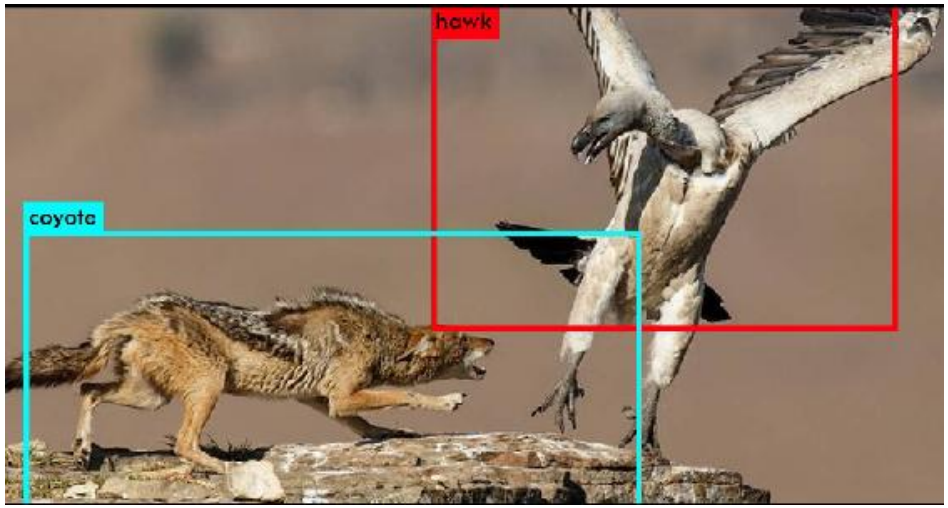





11
11
224
224
3
55
55
96
Stride
of 4
5
5
27
27
256
Max
pooling
3
3
13
13
384
Max
pooling
3
3
13
13
384
3
3
13
13
256


$\vec{x}$
Object classification
head (K+1 classes)
BB regression
head (4D)

Yolo: You only look once!



`https://pjreddie.com/darknet/yolo/`

Yolo training tricks: Anchor boxes

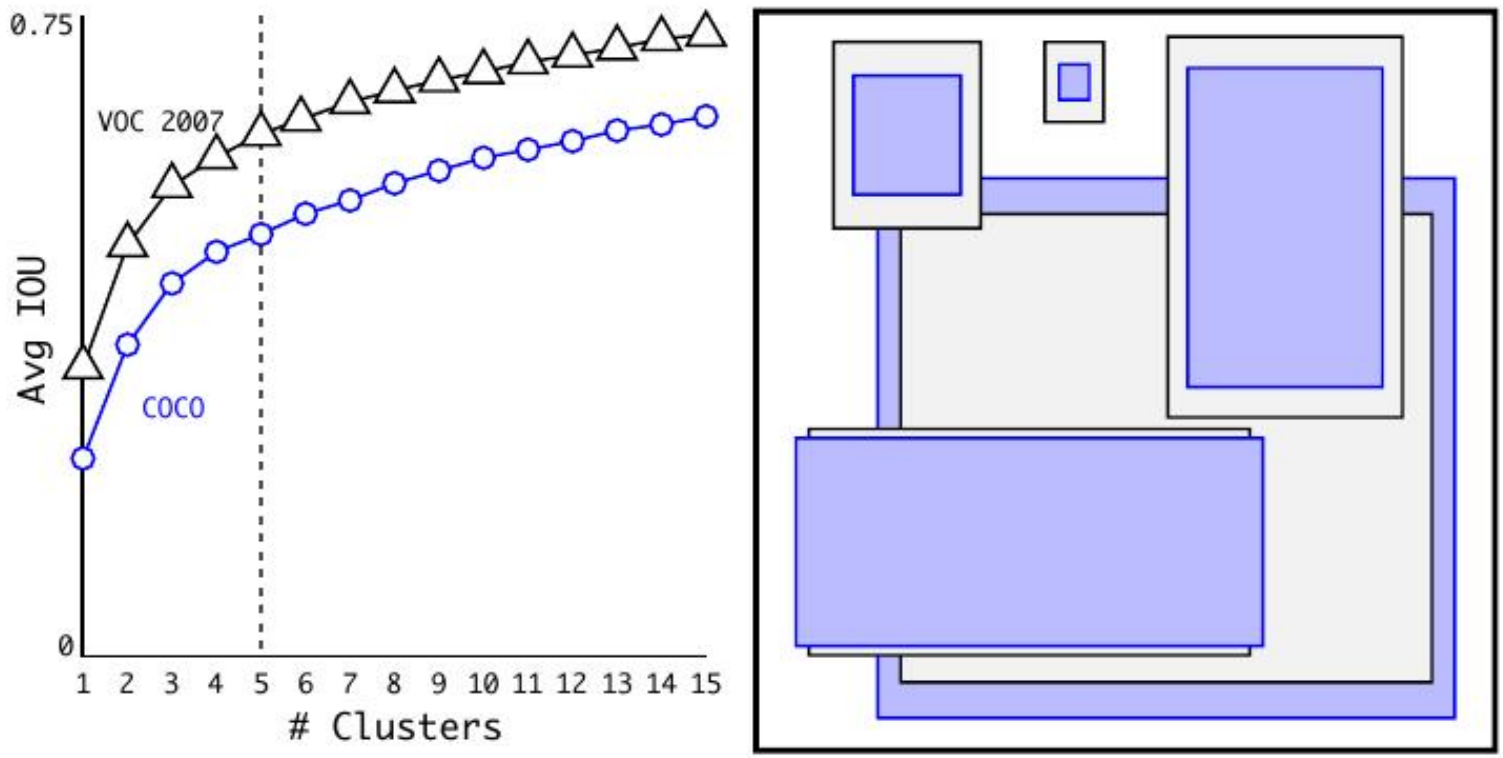


Image from original work.

- Yolo predicts offsets from anchor boxes instead of absolute BBs coordinates.
- 5 anchor boxes on each position

Yolo training tricks: Classifiers

- Any standard CNN can be used as a backbone, Yolo uses a 19 layers FCN (darknet)
- Last convolution map of Yolo has 13x13 cells
- For each cell, Yolo predicts BB offset for every anchor box and object class
- In total, Yolo predicts 13x13x5=845 BBs offsets
- Also, Yolo has a softmax to predict K class probabilities plus a background class
- Object label is assigned to center cell for each corresponding BB, and its propagate gradient. Each cell that has a relevant overlap with this center cell does not propagate error signal during training
- At test time, to avoid multiple detections for a single object, Yolo uses a non-maximum suppresion strategy (NMS)

Several further tricks, for details see: "YOLO9000: Better, Faster, Stronger" by J. Redmon and A. Farhadi.

Faster RCNNs

S. Ren, K. He, R. Girshick, and J. Sun

Yolo

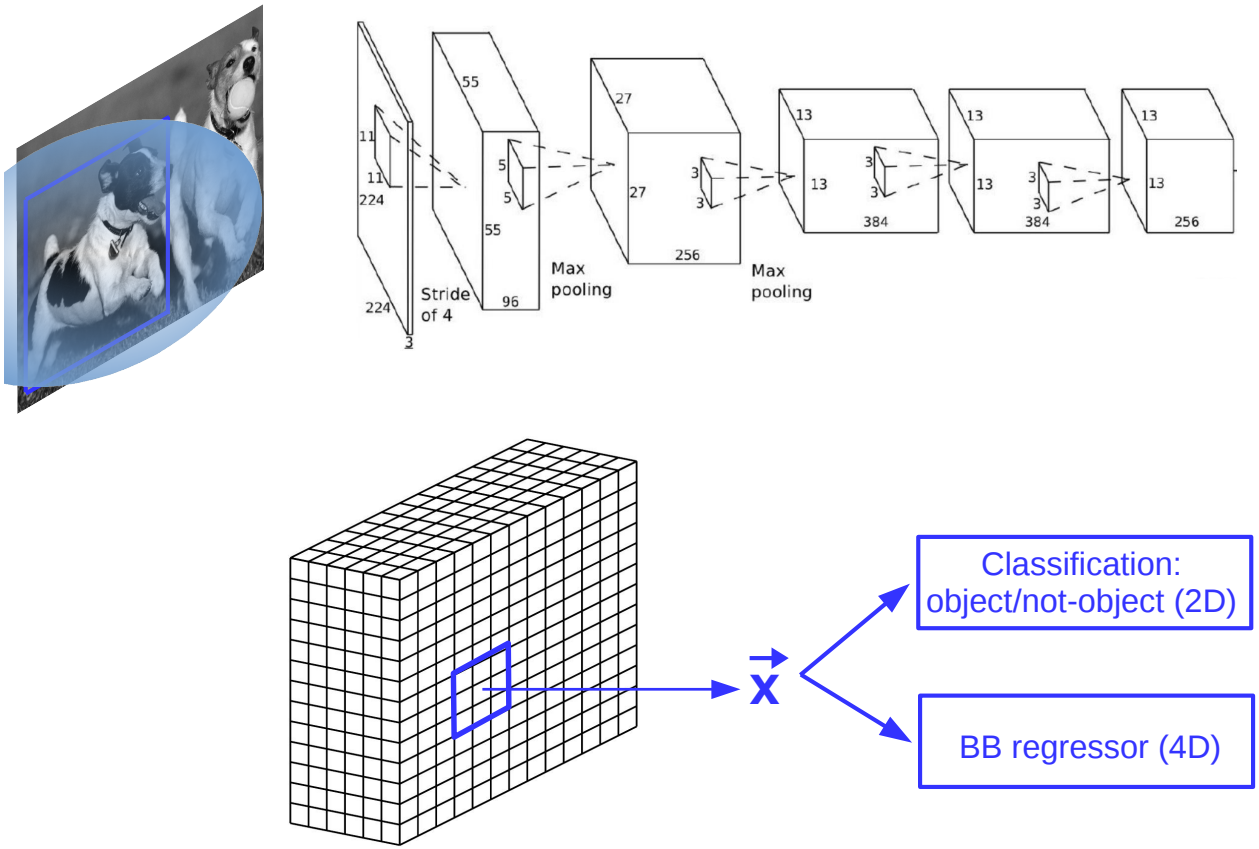
- Yolo has a single pipeline to perform: region label object detection and classification (you only look once)
- This process is fast and achieves good performance
- However, it demands a lot from the model, specially for the bounding box regressor

Faster-RCNN

- It proposes an alternative processing pipeline to Yolo
- Model performs detection and classification in two sequential steps: i) Initial BB detection, and then ii) BB refinement and object classification
- In general, this provides better results, in particular, in terms of BB localization accuracy

Step 1: Initial BB detection (objectness)

Fully convolutional pipeline, similar to the operation of Yolo, but it does not include information about object classes.



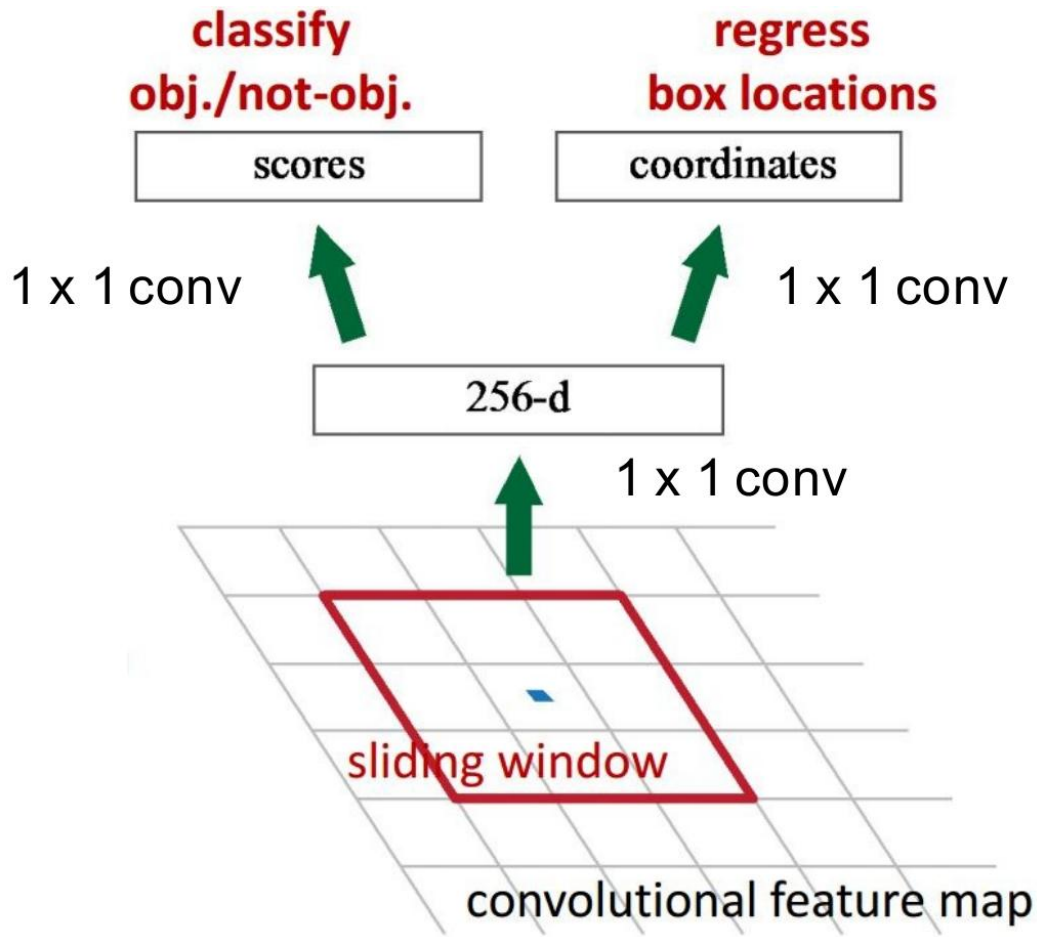


Image from R. Girshick.

- Classification head uses a fixed sliding window of 3x3
- Proposal label: object/no-object

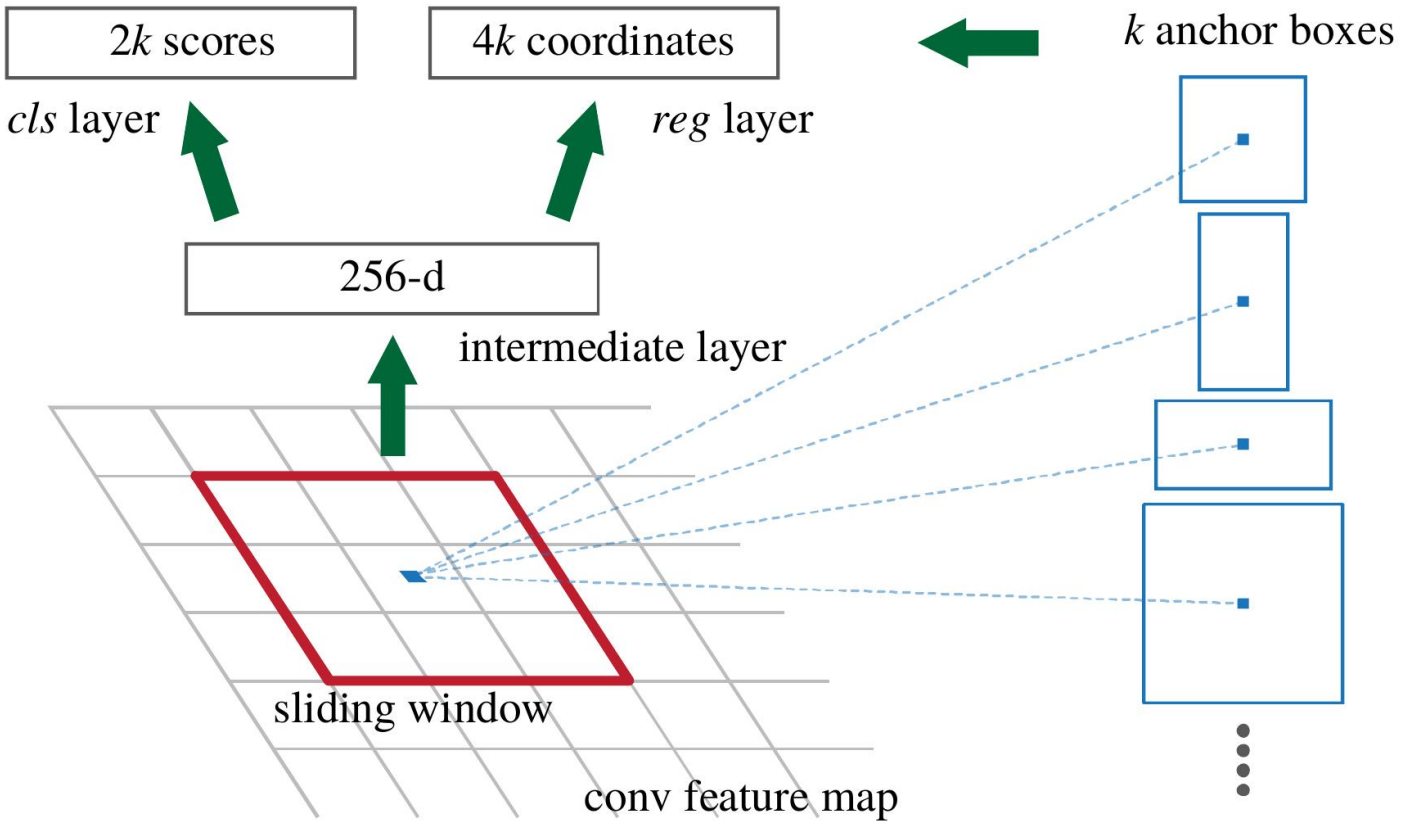


Image from R. Girshick.

- Regression head uses k=9 anchor boxes
- Each BB has 4 coordinates
- An anchor has a '+' label if it has the highest IoU for a given ground-truth box or an IoU over 0.7 with any ground-truth box.
- An anchor has a '-' label if it has IoU lower than 0.3 with respect to all ground-truth boxes

Step 2: BB refinement and object classification

Candidate object boxes go to the second step

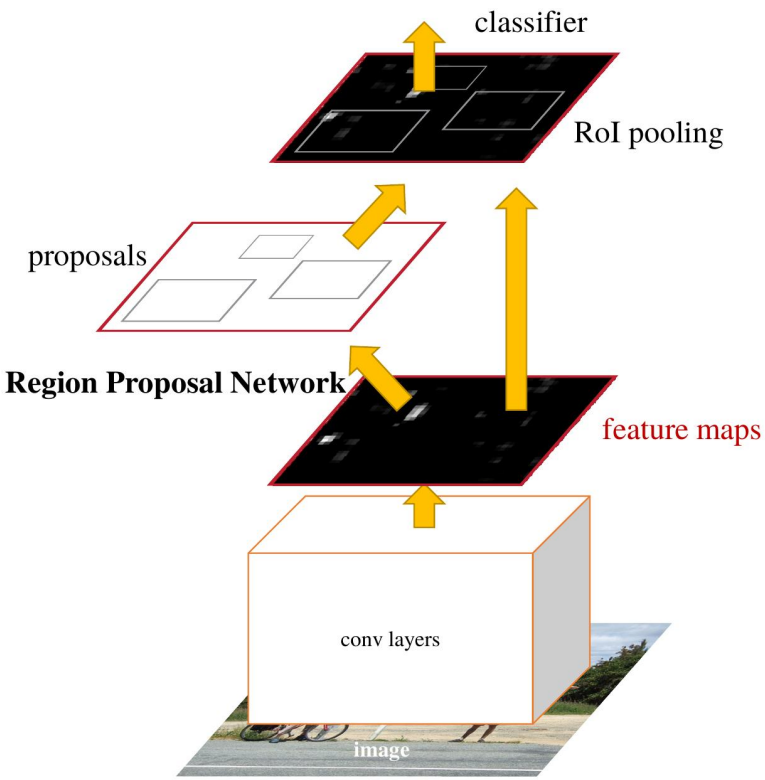
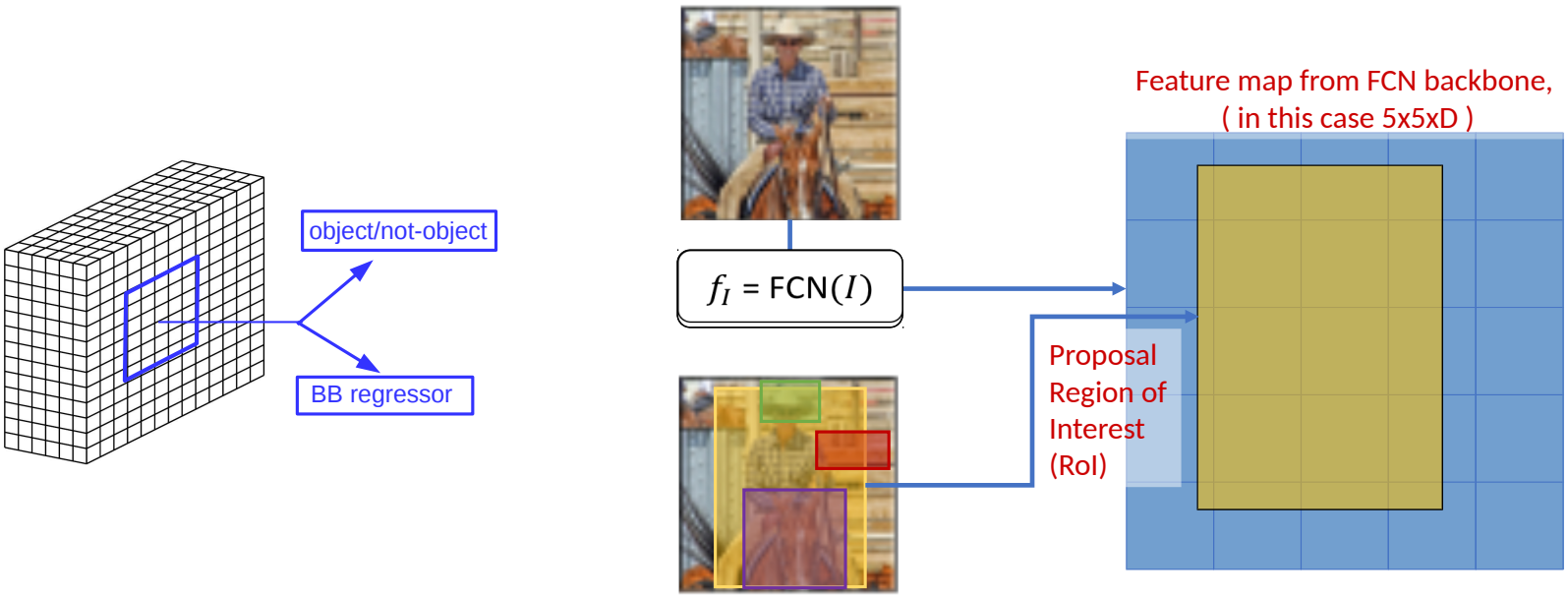


Image from original work.

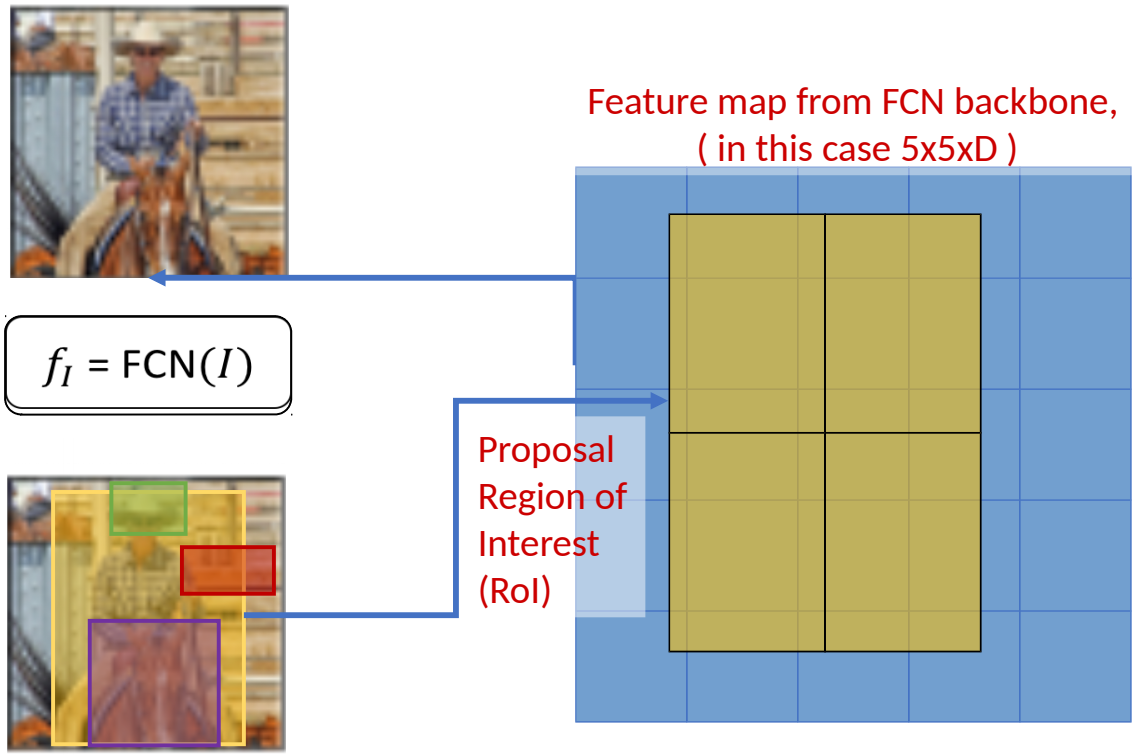
For each candidate box, a key task is to extract relevant features from the CNN backbone: Region of Interest Pooling (RoI pooling).

RoI Pooling



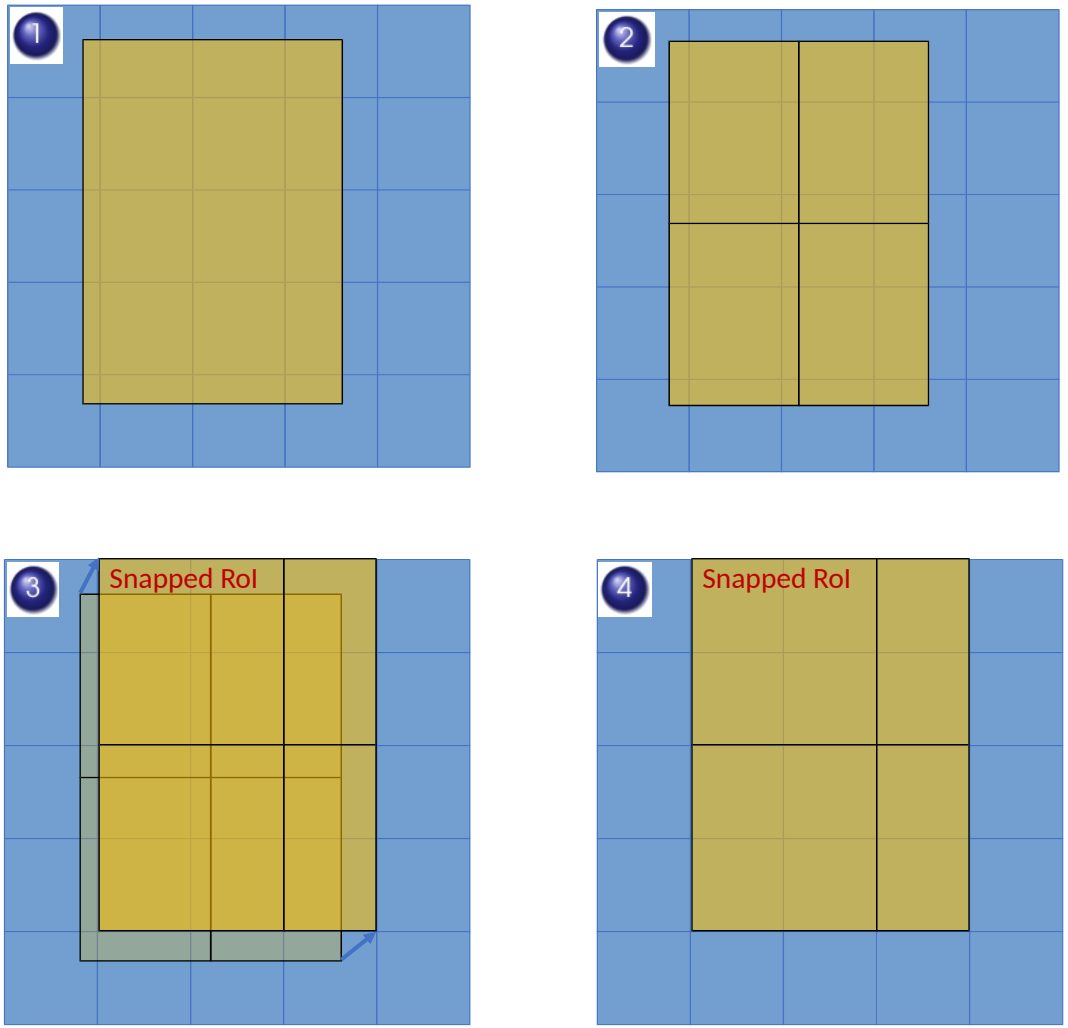
- We need to extract from FCN the features corresponding to the RoI
- This process is known as feature pooling

RoI Pooling



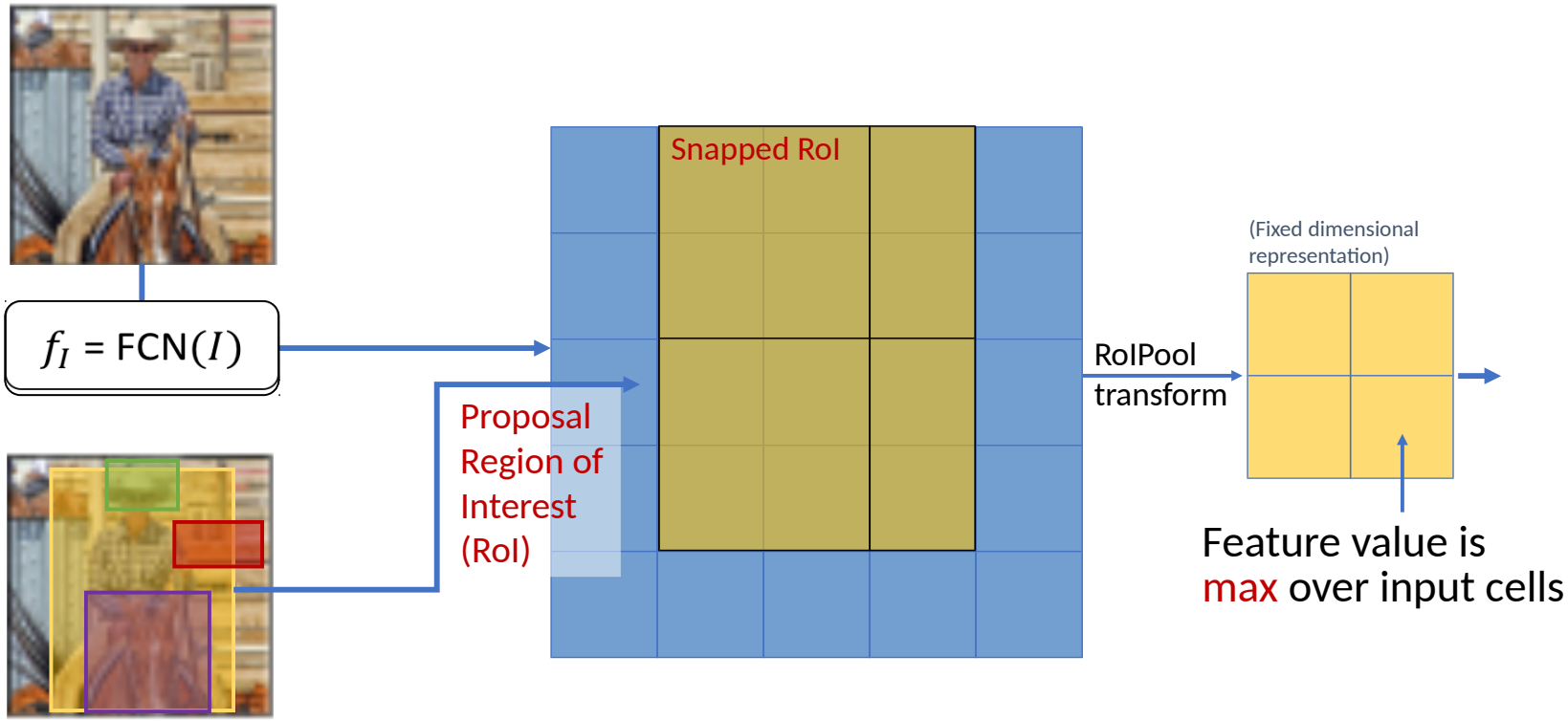
- Faster RCNN uses a fixed grid to move candidate object proposal to feature space (2x2 in the figure)
- We need to acomodate the different resolution between the candidate object proposal and the FCN backbone

RoI Pooling



- For each output cell and feature dimension, output feature value is max over input cells.

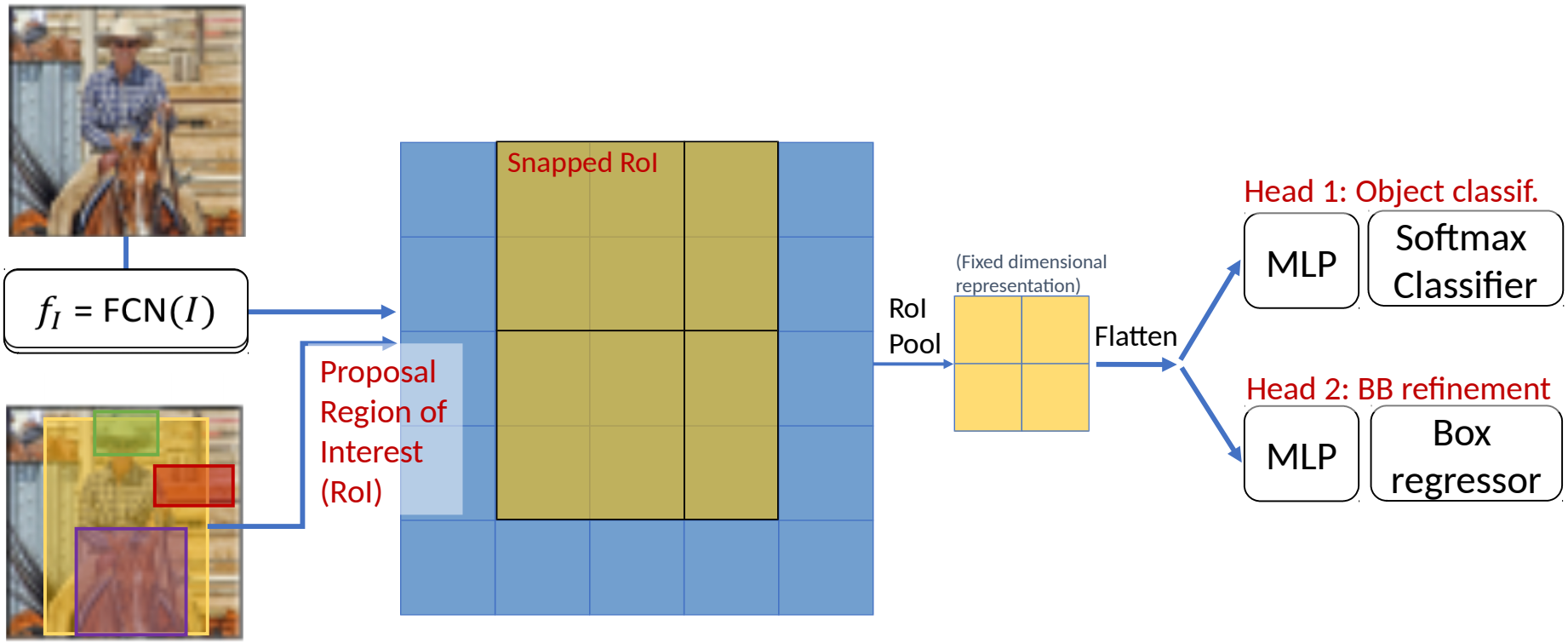
RoI Pooling



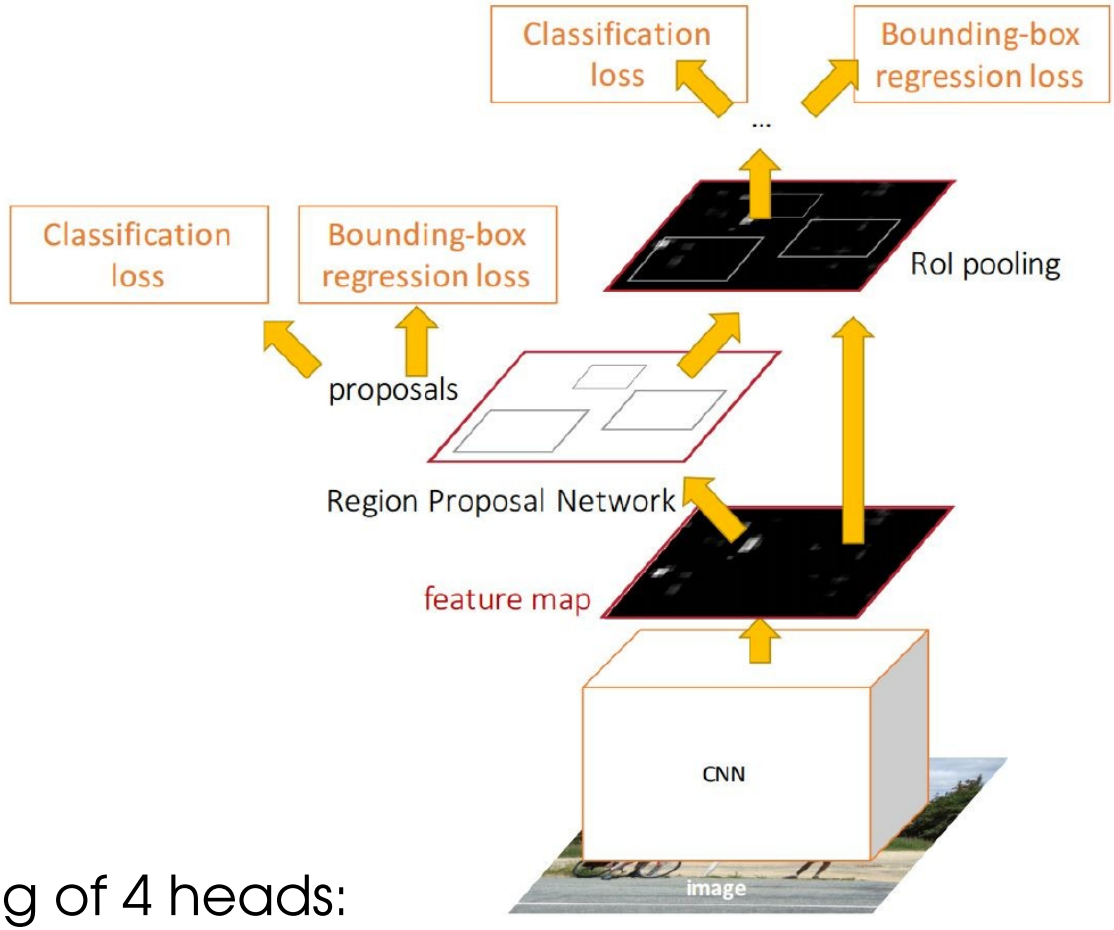
- RoI pool transforms proposals into a fixed-dimensional representation, in this case 2x2xd

Step 2: BB refinement and object classification

After RoI pooling, each candidate box goes to BB refinement and classification



Faster R-CNN: Final Model



Joint training of 4 heads:

- BB classification (3x3 window obj/no-obj)
- BB regression step 1(k=9 anchors)
- Classification (soft-max over obj. classes + background class)
- BB regression step 2 (no anchor now, just direct BB regression)

Almost done!, we still need to talk about image resolution

Faster-RCNN + Pyramidal Decomposition

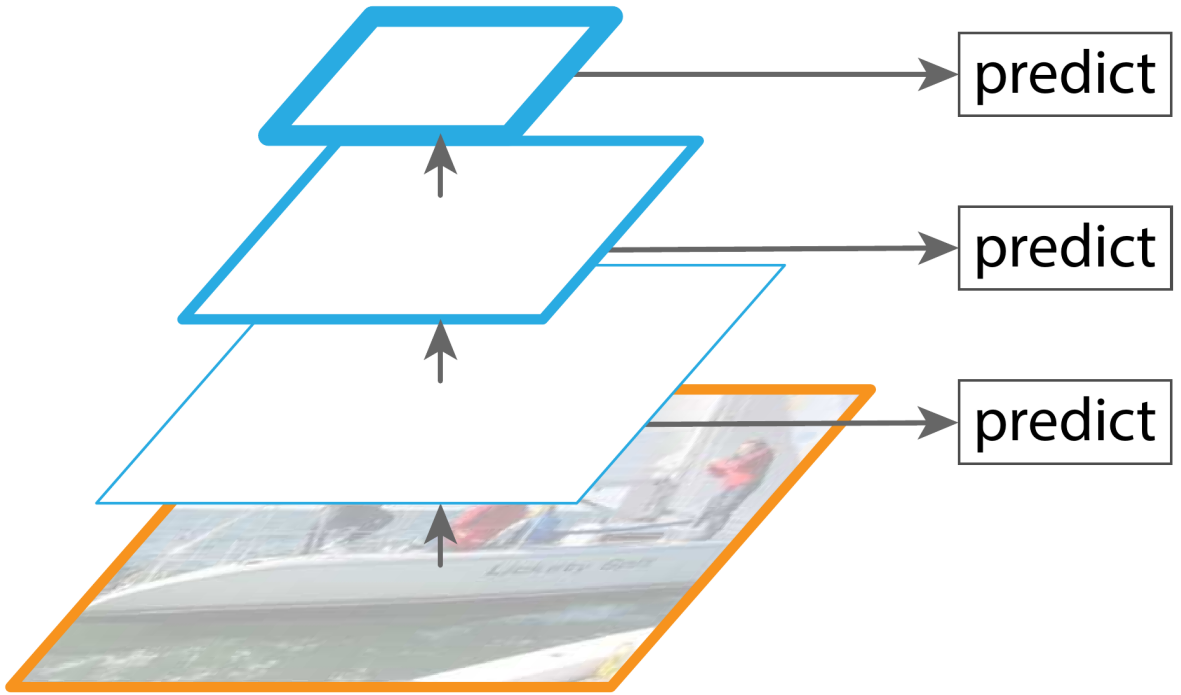
Object scale problem

- Model needs to detect and classify objects over a wide range of scales

Object scale problem

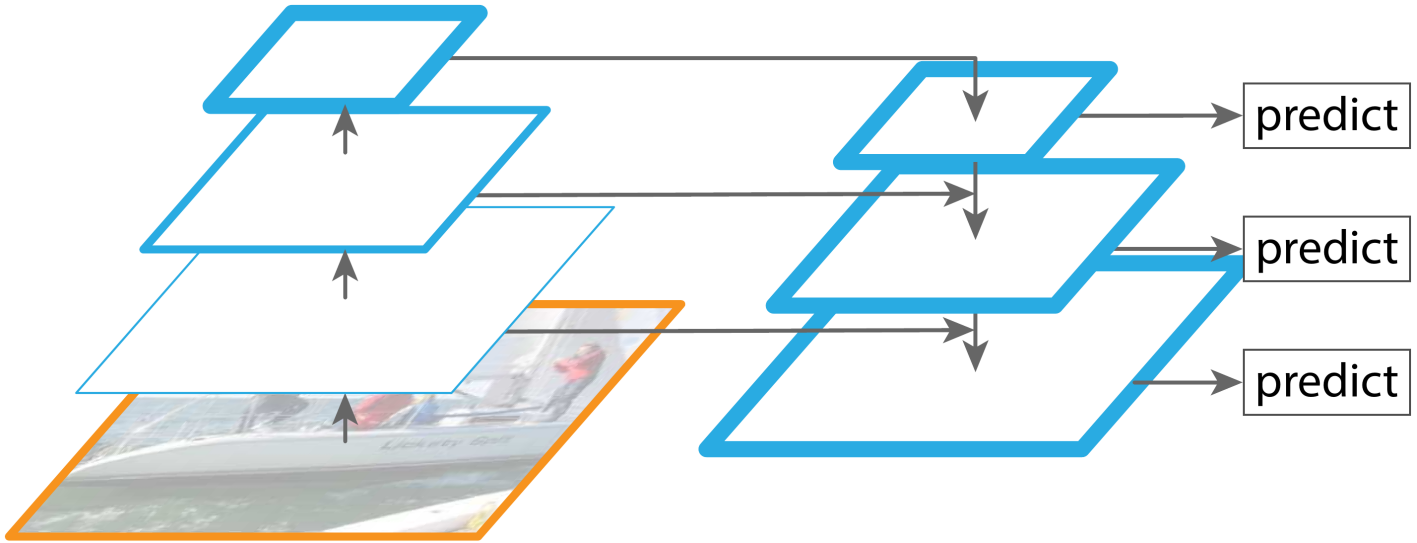
- Model needs to detect and classify objects over a wide range of scales
- It will be a good idea to adapt the object's features to a suitable scale

Hierarchical Feature Pyramidal Decomposition



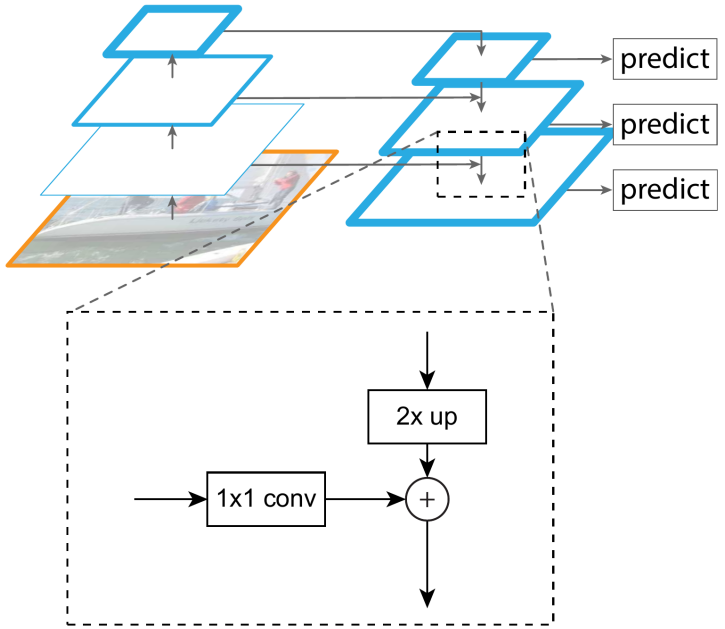
- A CNN builds a hierarchy of features, so it already includes a mechanism to build features at different scales
- A simple approach consists of applying a classifier to each level of the CNN's pipeline

Feature Pyramidal Networks (FPNs, Lin et al., 2017)



- Li et al. propose a clever idea. Combine information from bottom-up and top-down pathways
- For a given network layer:
  - Bottom-up pathway includes lower-level features (less semantic), but they have better spatial localization
  - Top-down pathway includes higher-level features (more semantic), but they have poorer spatial localization
- FPNs combine both pathways via lateral connections

Comibining bottom-up and top-down pathways



- For top-down pathway, FPNs upsample the spatial resolution by a factor of 2 using nearest neighbor upsampling for simplicity (Resnet adaptation).
- 1-D convolutions are used to adapt the dimensionality of both maps, so element-wise addition is used to combine them.
- Finally, they append a 3x3 convolution on each merged map to smooth the final feature map.

Finally, we got to the final model

Faster-RCNN + Pyramidal Decomposition

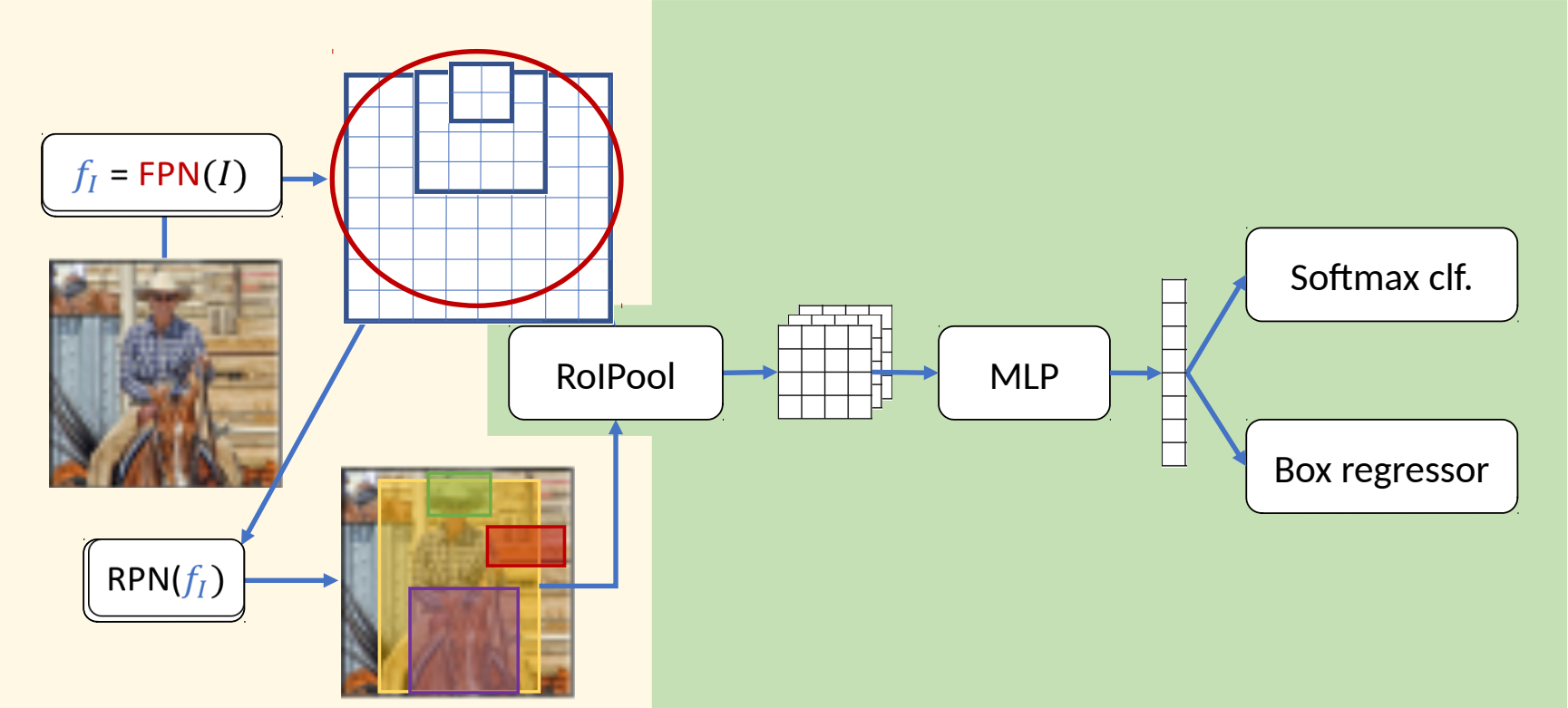


Image from R. Girshick.

- Original model was trained using Coco dataset

Faster-RCNN + Pyramidal Decomposition

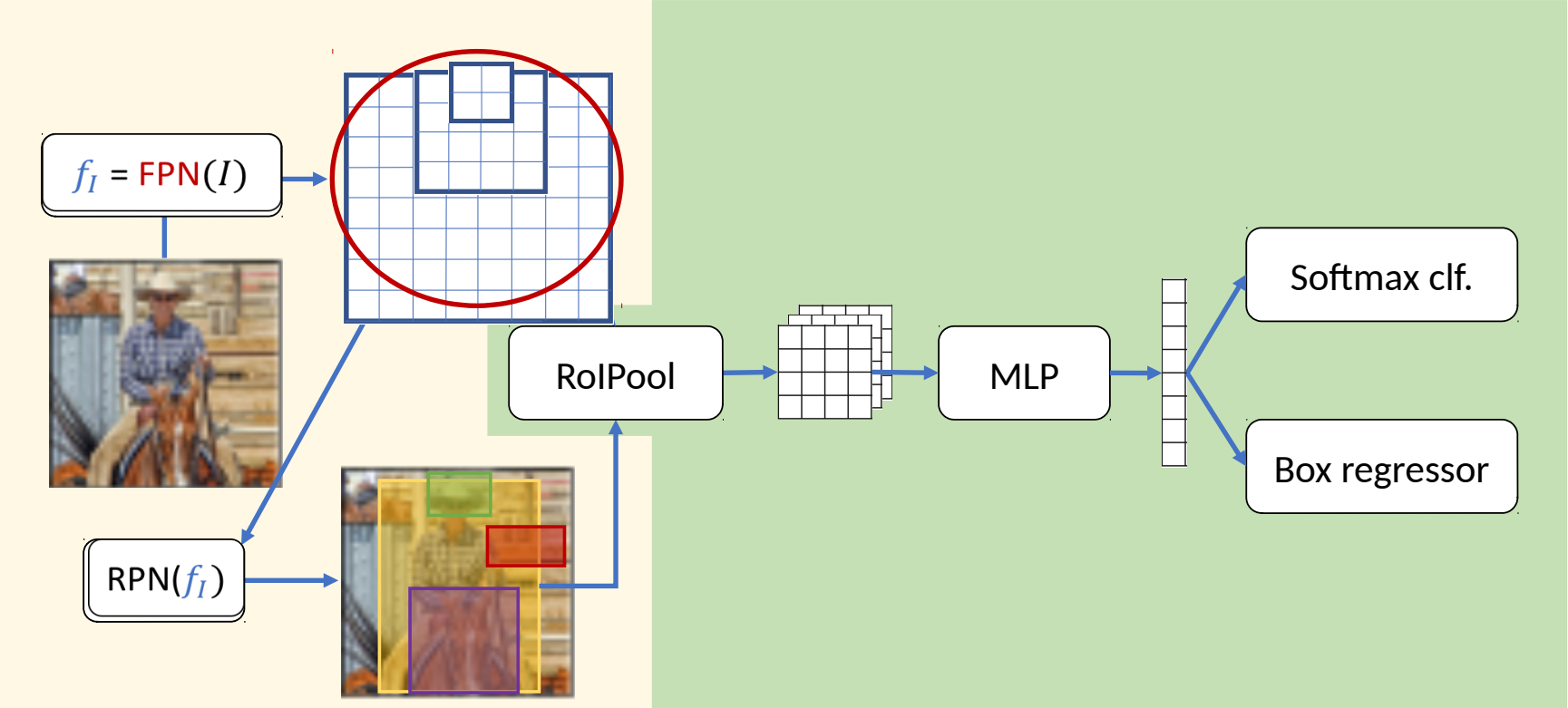


Image from R. Girshick.

- Original model was trained using Coco dataset
- By adapting the spatial range of the convolution operator, one can input images of different sizes

Faster-RCNN + Pyramidal Decomposition

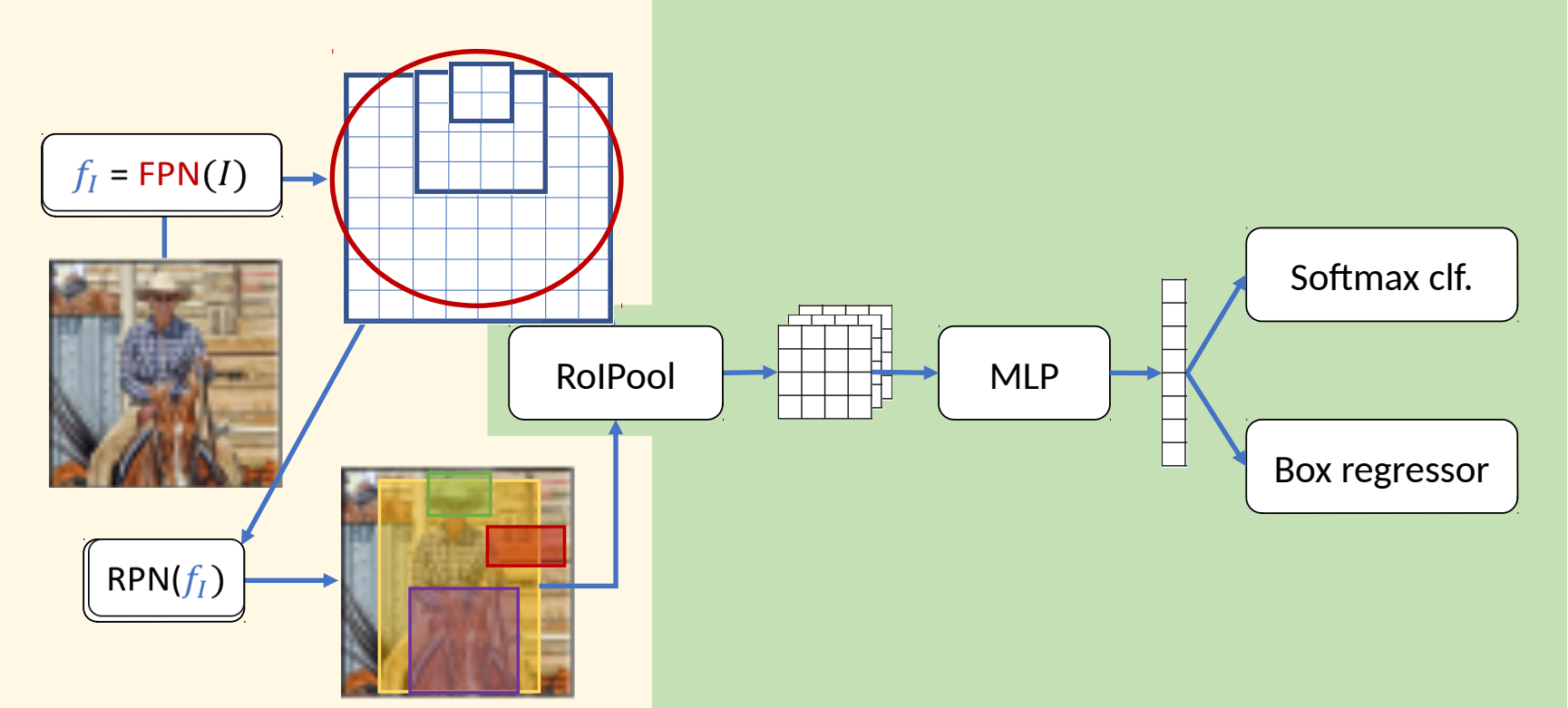


Image from R. Girshick.

- Original model was trained using Coco dataset
- By adapting the spatial range of the convolution operator, one can input images of different sizes
- By adapting the classification head, one can customize the model to specific applications (finetunning)

DEMO TIME