Convolutional Neural Networks for Computer Vision

Example of the use of CNNs for various computer vision tasks

High-Dimensional-Deep-Learning – juliette.chevallier@insa-toulouse.fr

INSA Toulouse, Applied Mathematics, 5th year

Some Classic Problems in Image Processing

Some Classic Problems in Image Processing

Classification

or: Annotation, *Labelling*

Main class: **Person**

Secondary classes:
Building, trees, grass, sky

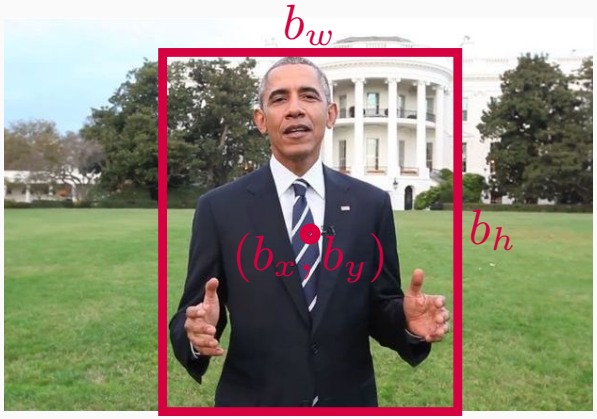
Some Classic Problems in Image Processing

Classification

or: Annotation, *Labelling*

Main class: **Person**

Secondary classes:
Building, trees, grass, sky



Localization

Goal: Delineate the position of the object, using a *bounding box*.

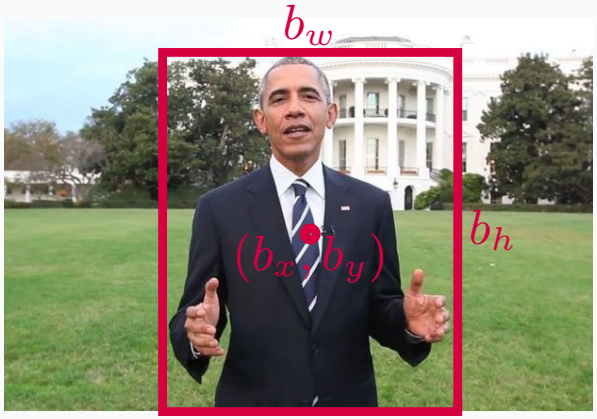
Some Classic Problems in Image Processing

Classification

or: Annotation, *Labelling*

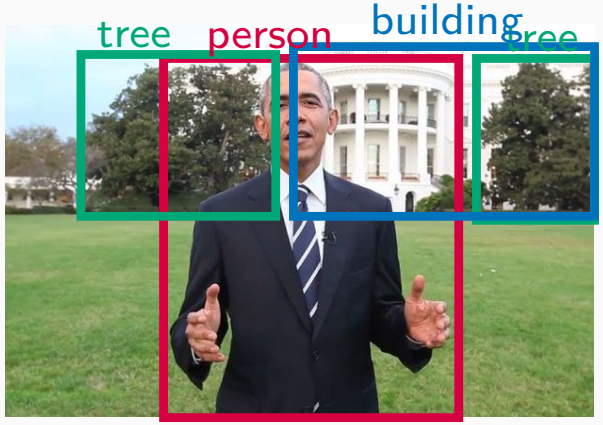
Main class: **Person**

Secondary classes:
Building, trees, grass, sky



Localization

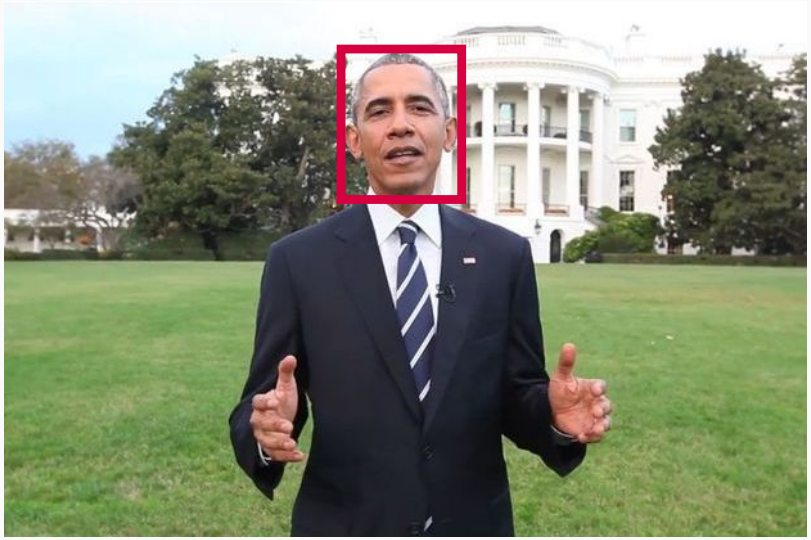
Goal: Delineate the position of the object, using a *bounding box*.



Object detection

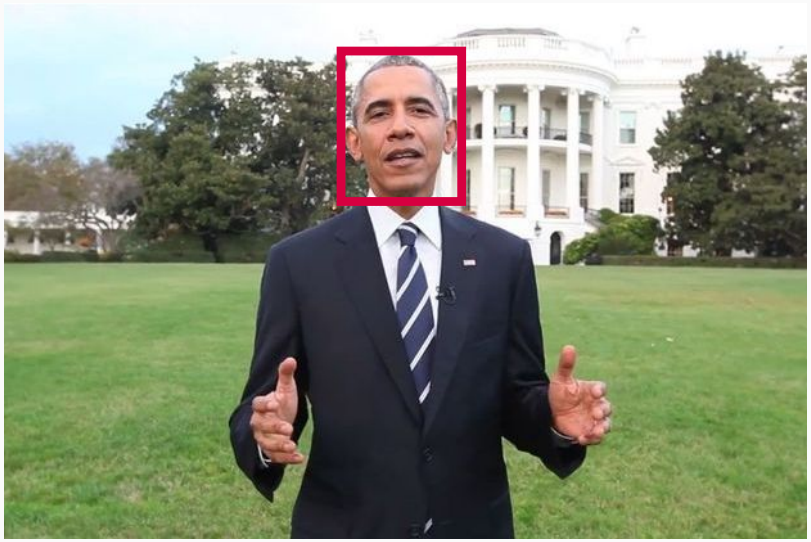
Goal: Delineate the position of multiple objects, possibly with multiple instances, always using bounding boxes.

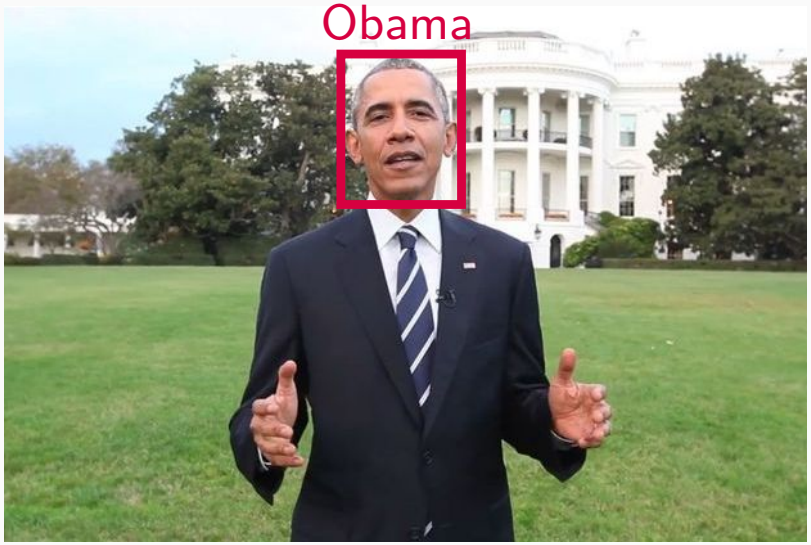
A Famous Sub-Problem of Object Detection: Face Detection



A Famous Sub-Problem of Object Detection: Face Detection

. . . and the recognition that goes with it!





Goal: Given a detected face and a face database, recognize the person.

Remark: *This problem* ***cannot*** *be formulated as a classification problem: Too many classes, and not enough examples.*

This problem is a typical example of ***few-shot learning****.*

Segmentation

Object Segmentation

Goal: Binary classification of image pixels as part of the object or background.

Segmentation

Object Segmentation

Goal: Binary classification of image pixels as part of the object or background.

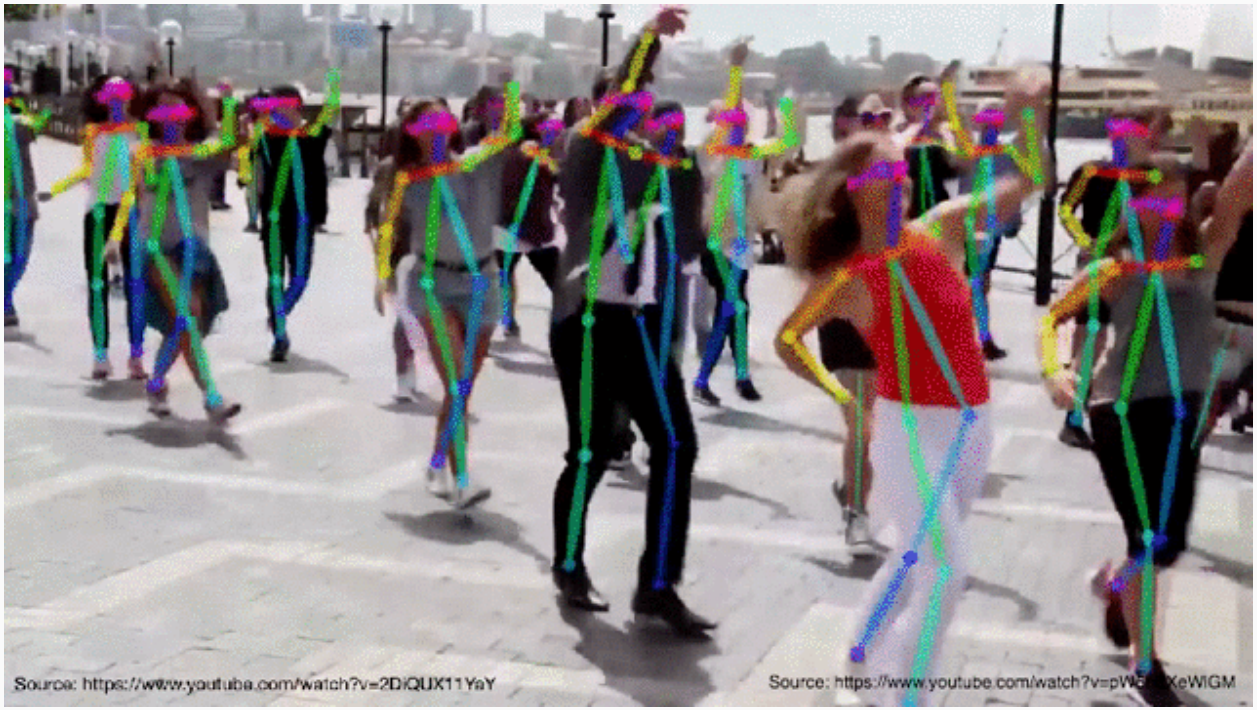
Image Segmentation

ou *Image parsing*

Goal: Classification in n-area of the pixels of the image.

But Also

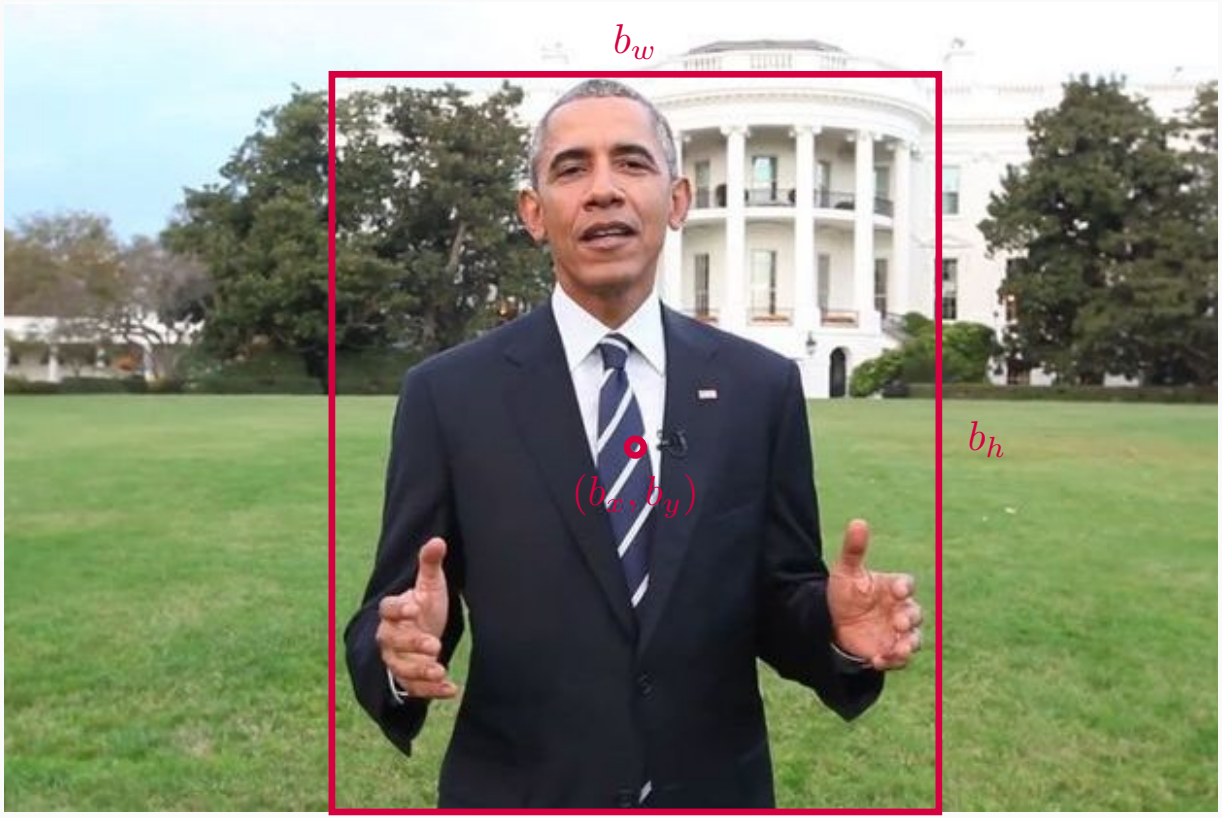
Pose estimation: Locating the "joints" or articulations of persons in images.



And many others, but rather using *generative models* like GAN:
Style transfer, B&W image colorization, Image reconstruction, Super-resolution image, Image synthesis, *etc.*

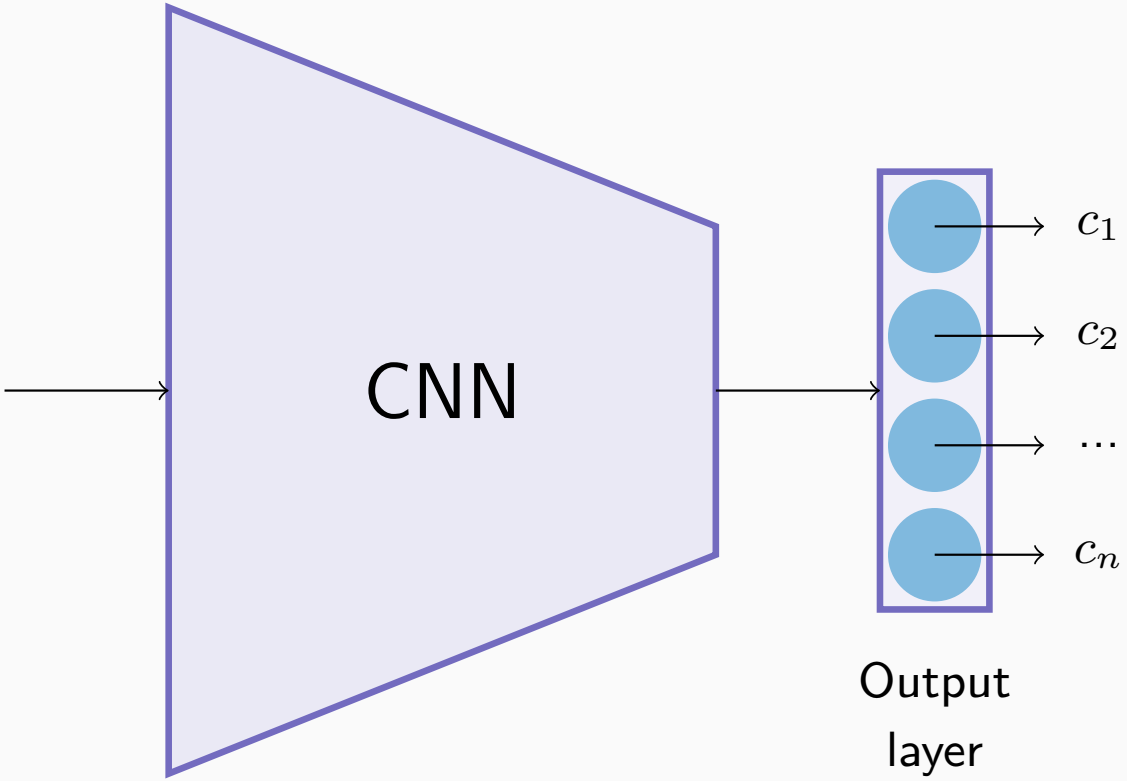
Introduction to Object Detection

Joint Classification and Localization



In this problem, we seek to classify and locate **an** object by image.

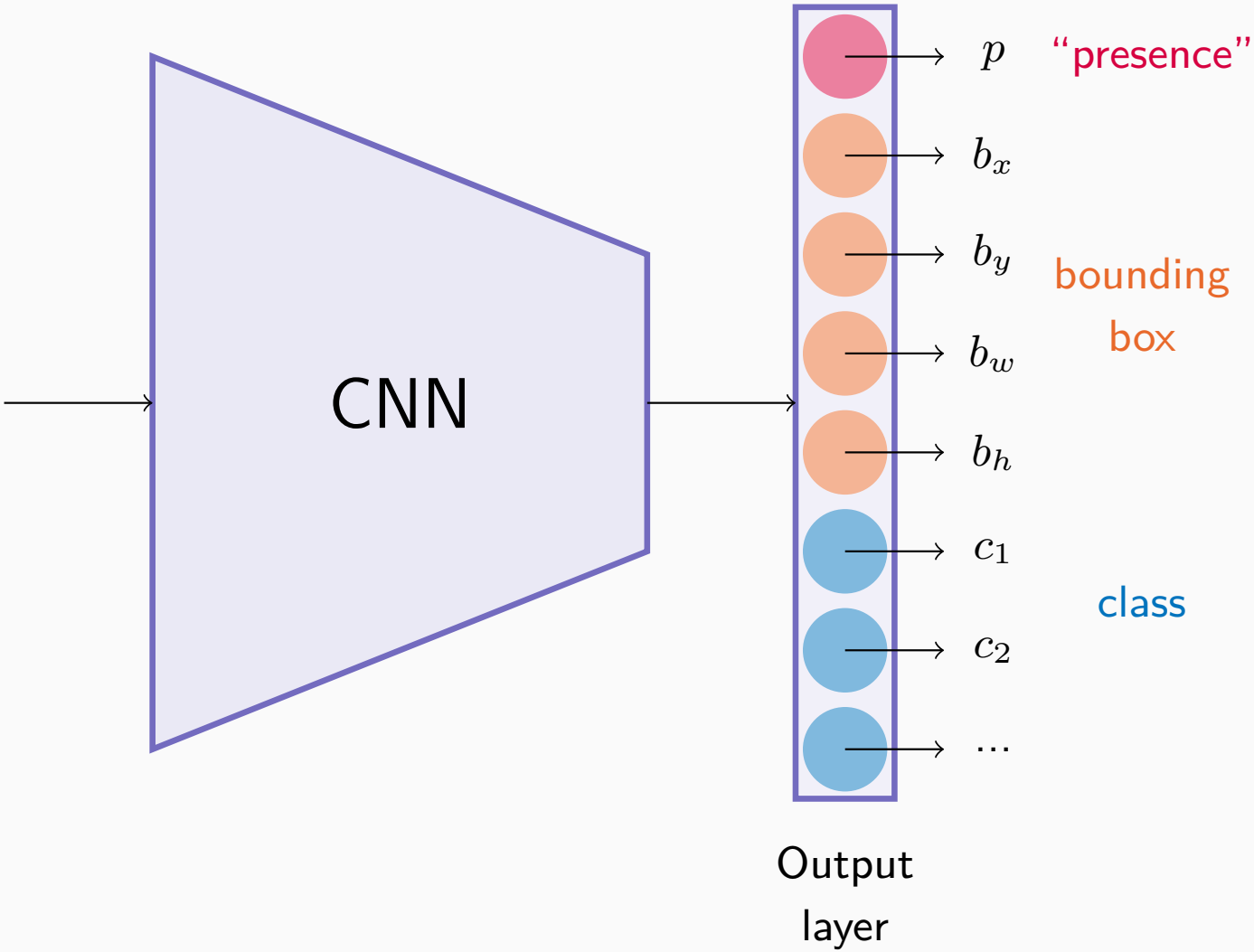
Joint Classification and Localization



We need to add *spatial* information about the classified object.

$\longrightarrow$ Add the coordinates of the bounding box around the object.

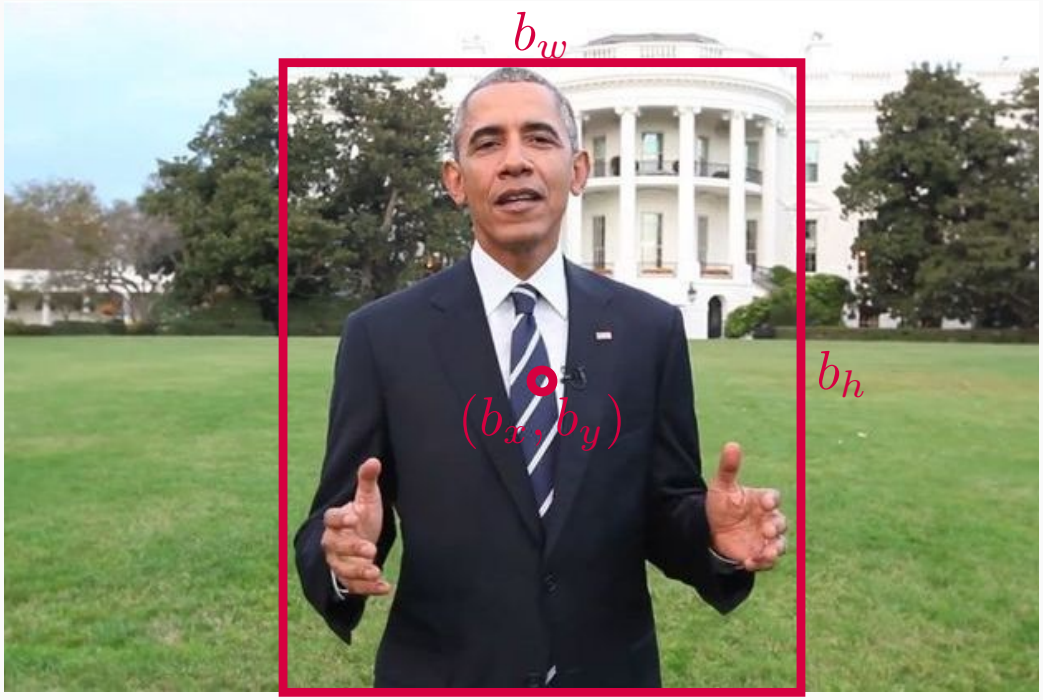
Joint Classification and Localization



We need to add *spatial* information about the classified object.

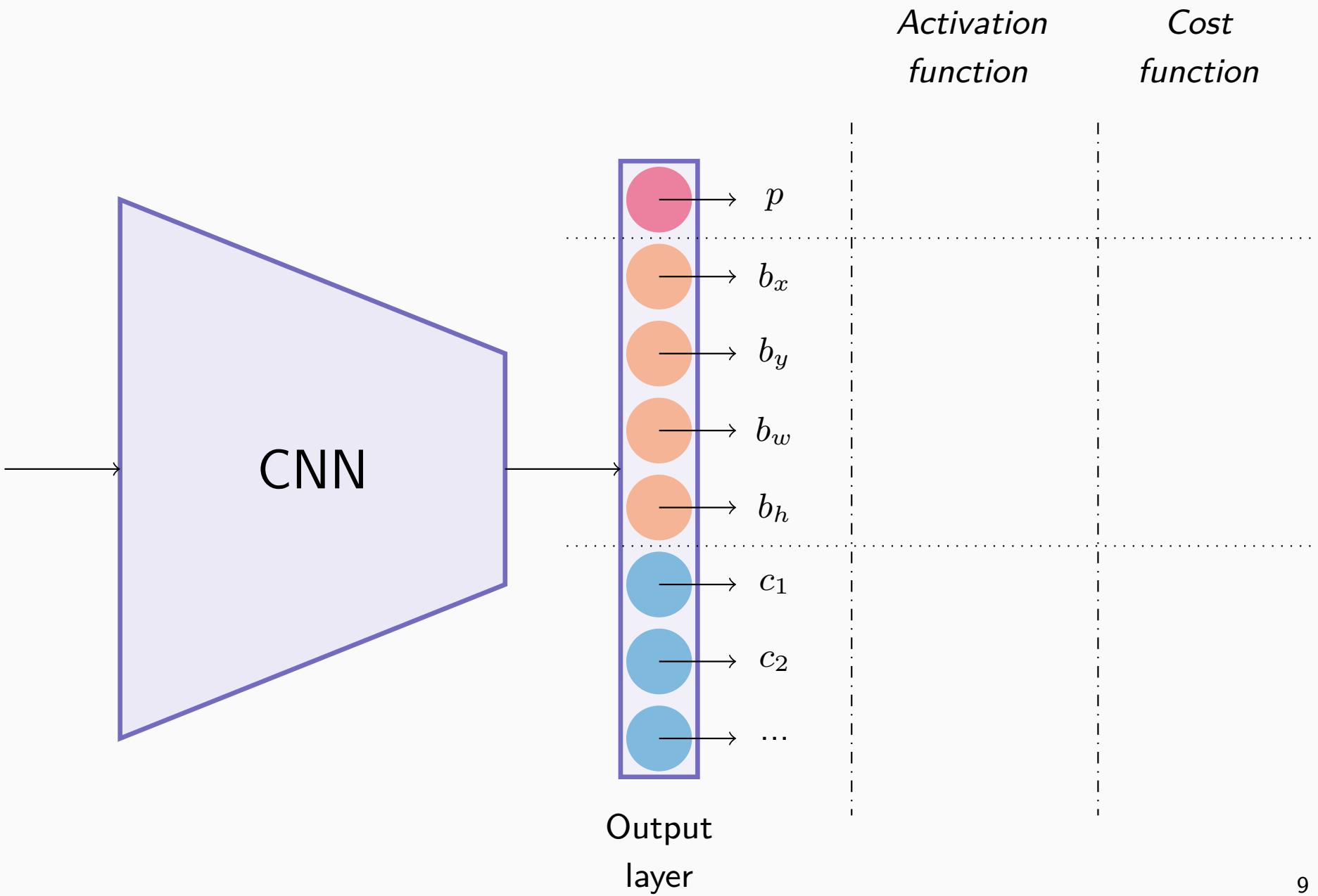
$\longrightarrow$ Add the coordinates of the bounding box around the object.

Joint Classification and Localization: An Example

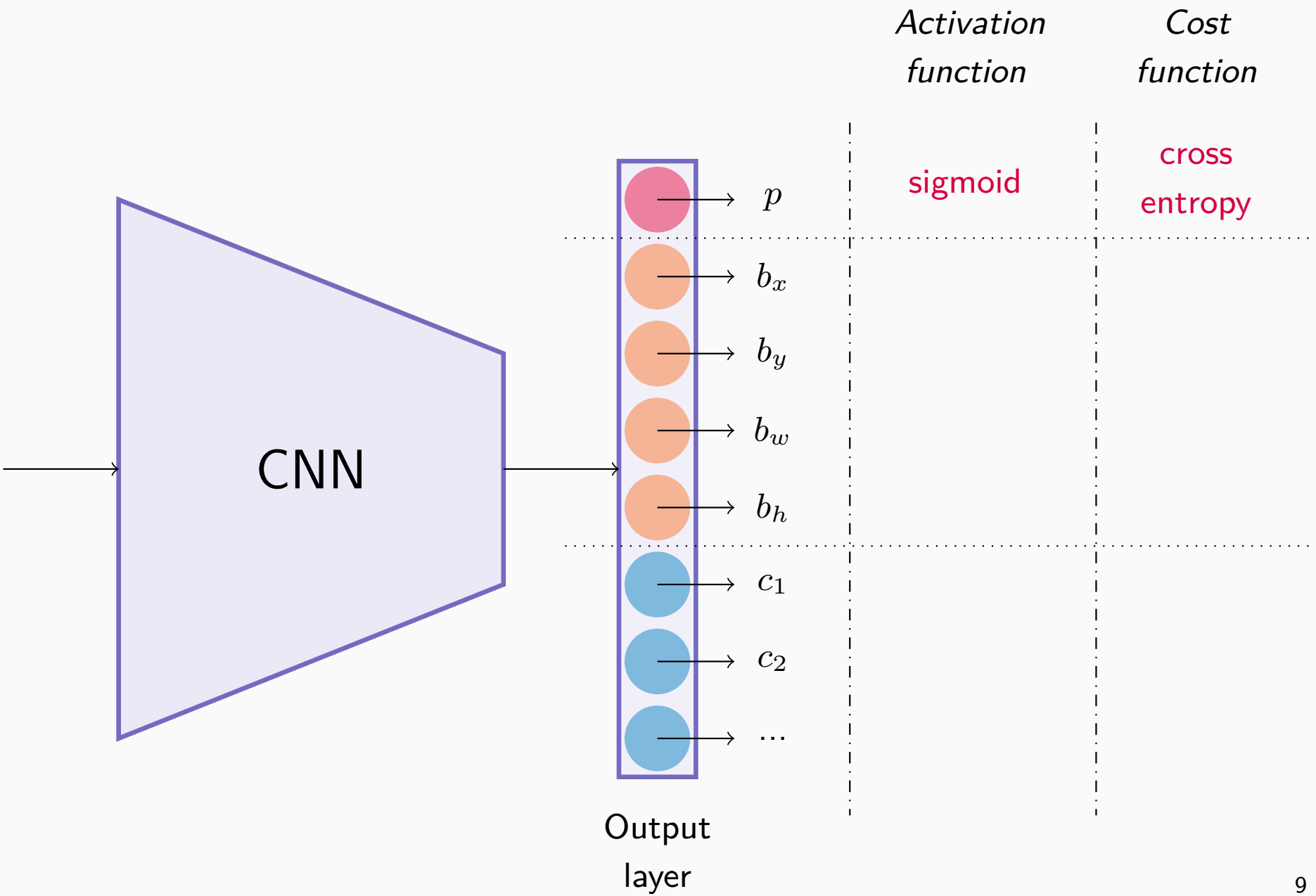


$$\hat{y} = \begin{pmatrix} 1 \\ 0.52 \\ 0.46 \\ 0.5 \\ 0.91 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix}$$

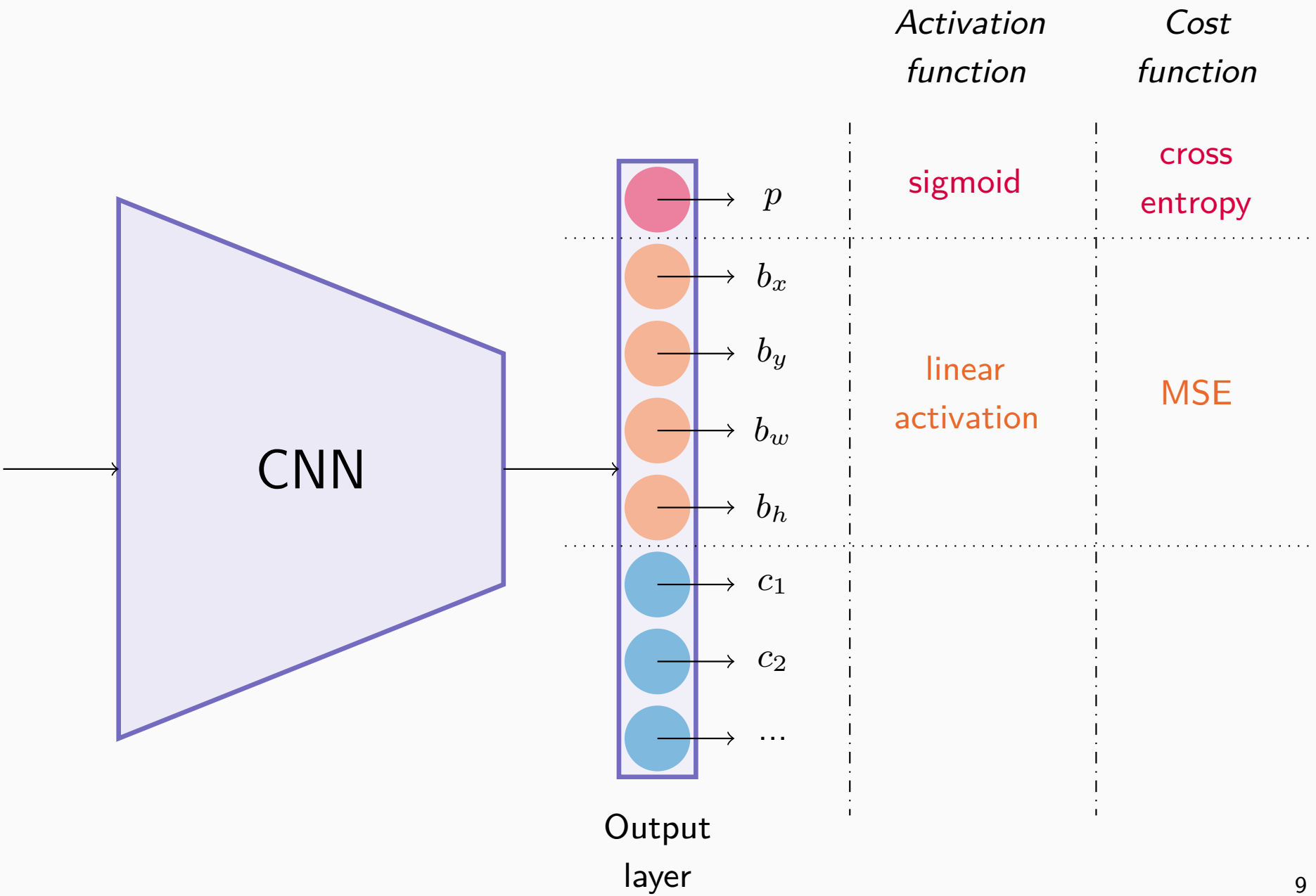
Joint Classification and Localization: Activation and Loss Functions



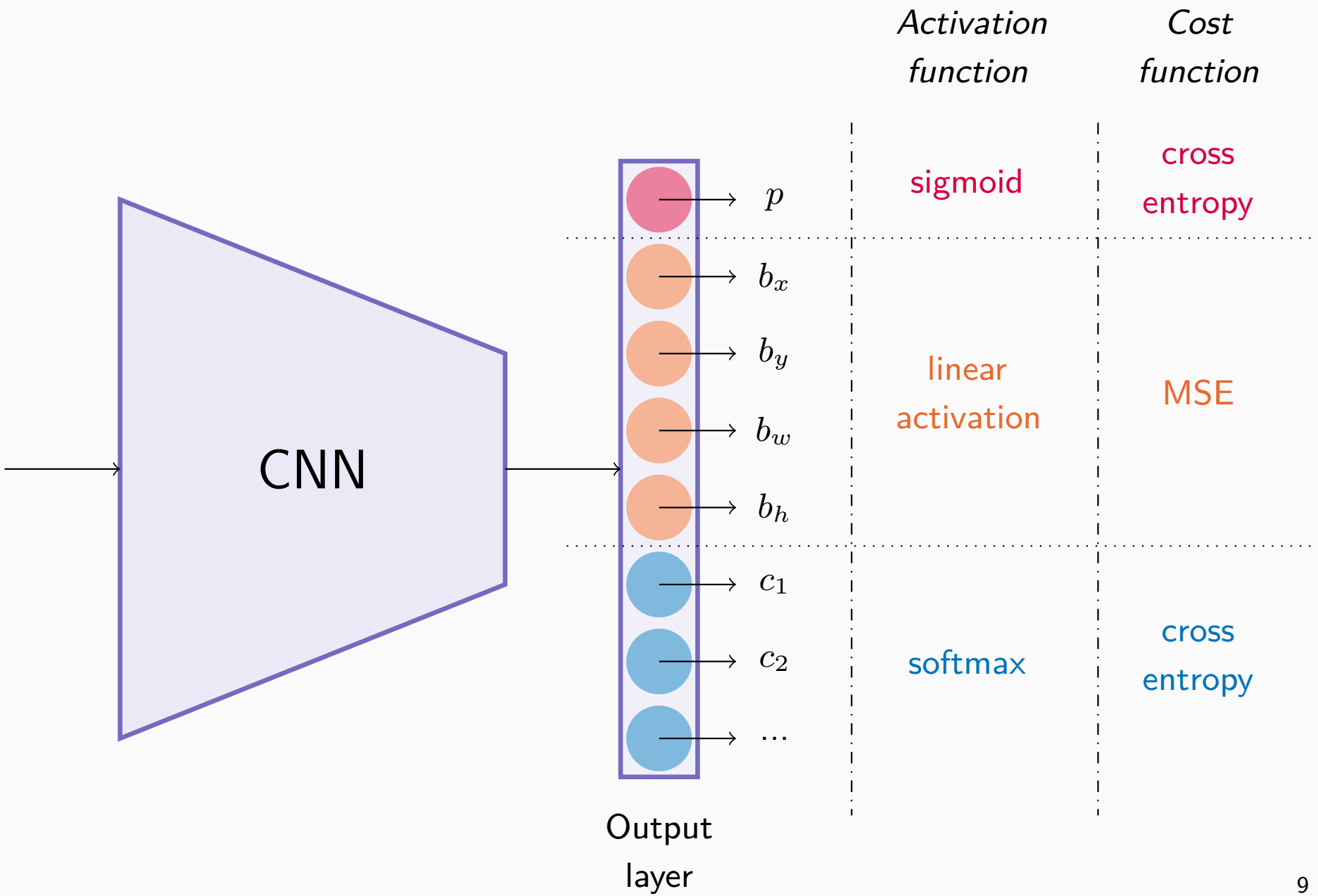
Joint Classification and Localization: Activation and Loss Functions



Joint Classification and Localization: Activation and Loss Functions



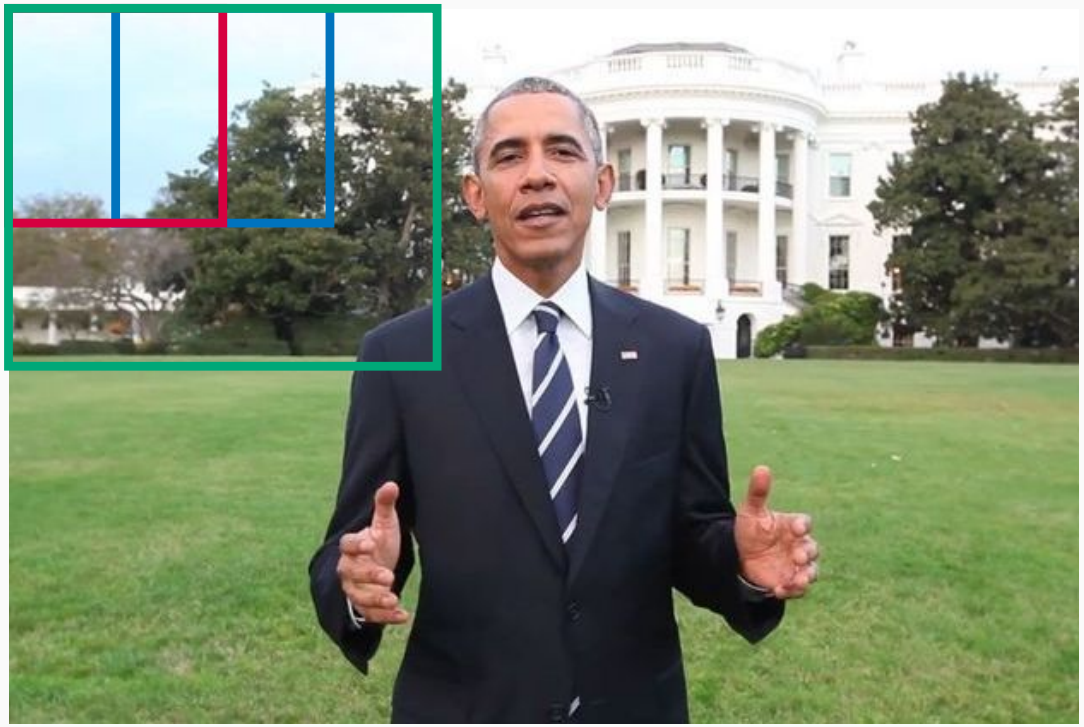
Joint Classification and Localization: Activation and Loss Functions



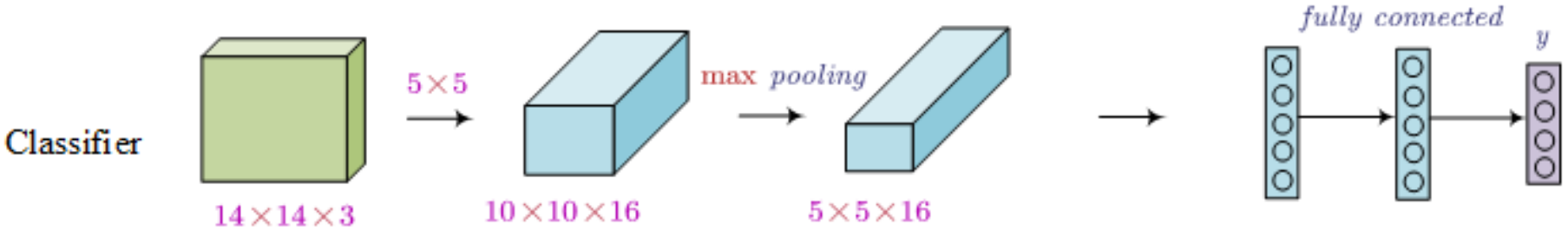
Introduction to Object Detection

From Localization to Object Detection

First idea: use a *multi-scale* sliding window to locate the objects present on each sub-part of the image.

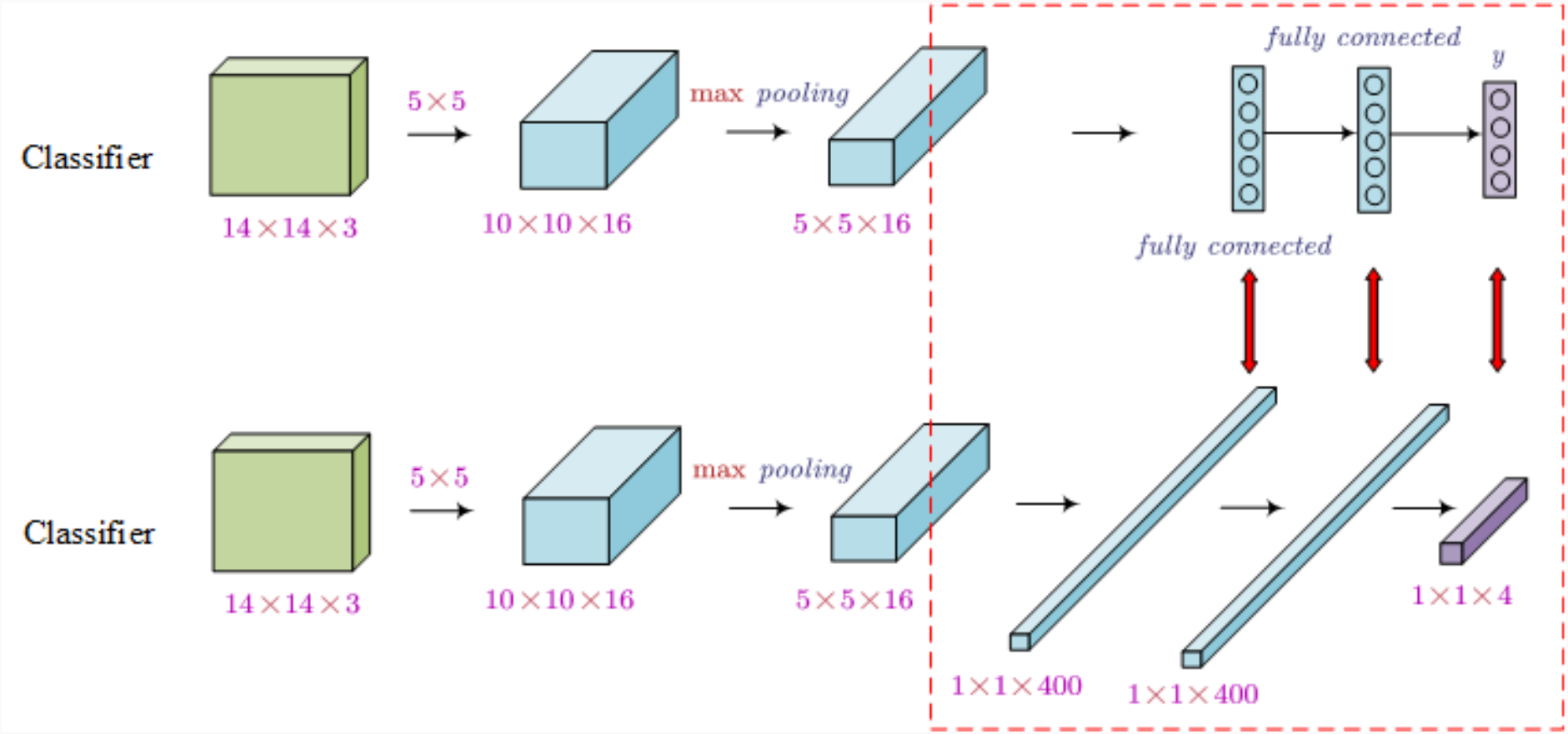


Convolutional Sliding Windows (Sermanet *et al.*, 2013)



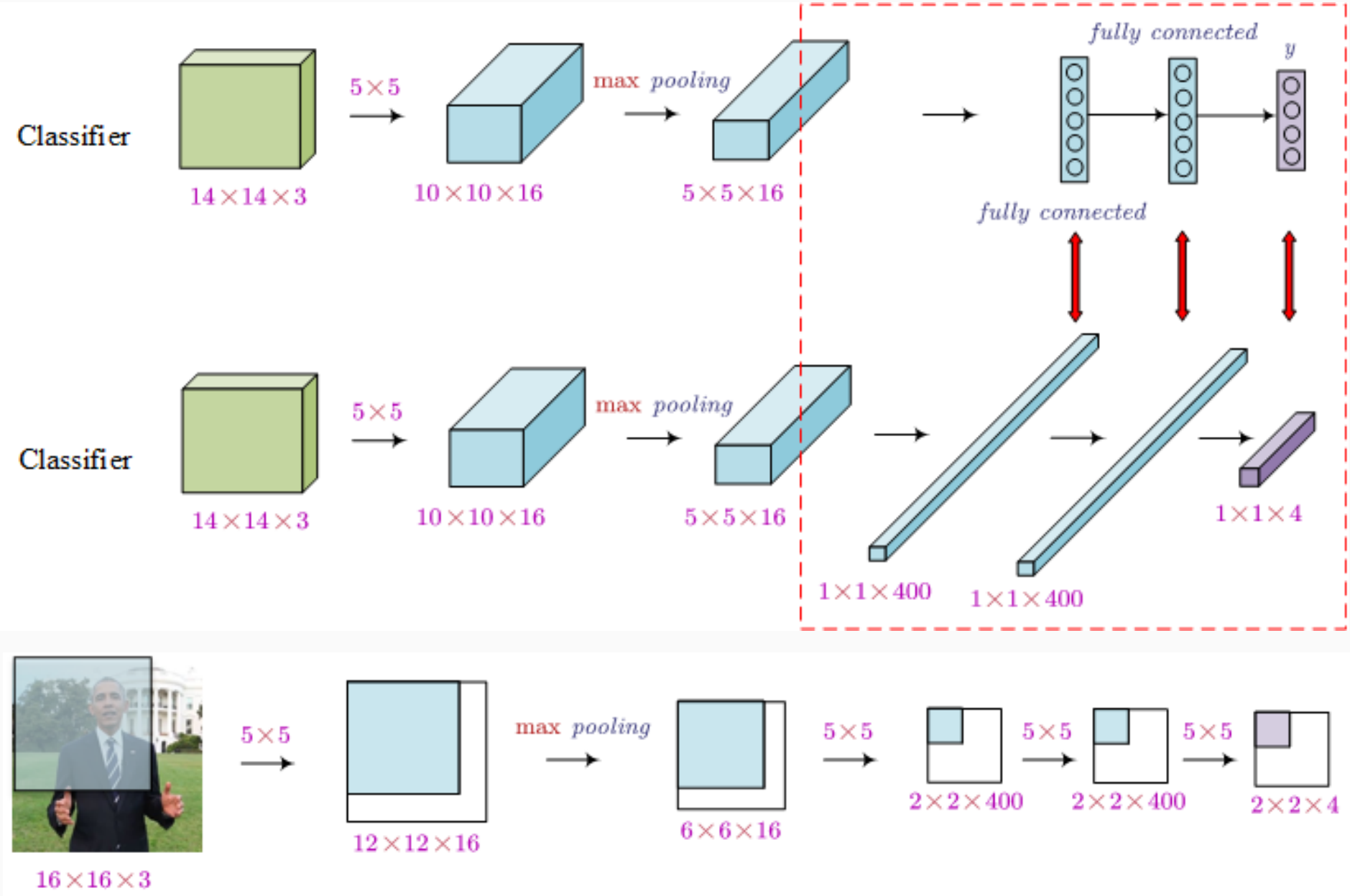
[Sermanet et al.] OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks

Convolutional Sliding Windows (Sermanet *et al.*, 2013)



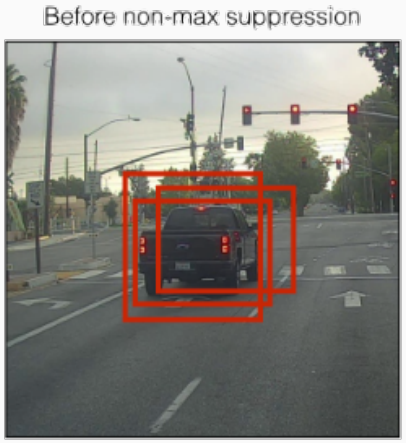
[Sermanet et al.] OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks

Convolutional Sliding Windows (Sermanet *et al.*, 2013)



[Sermanet et al.] OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks

Duplicates During Detection

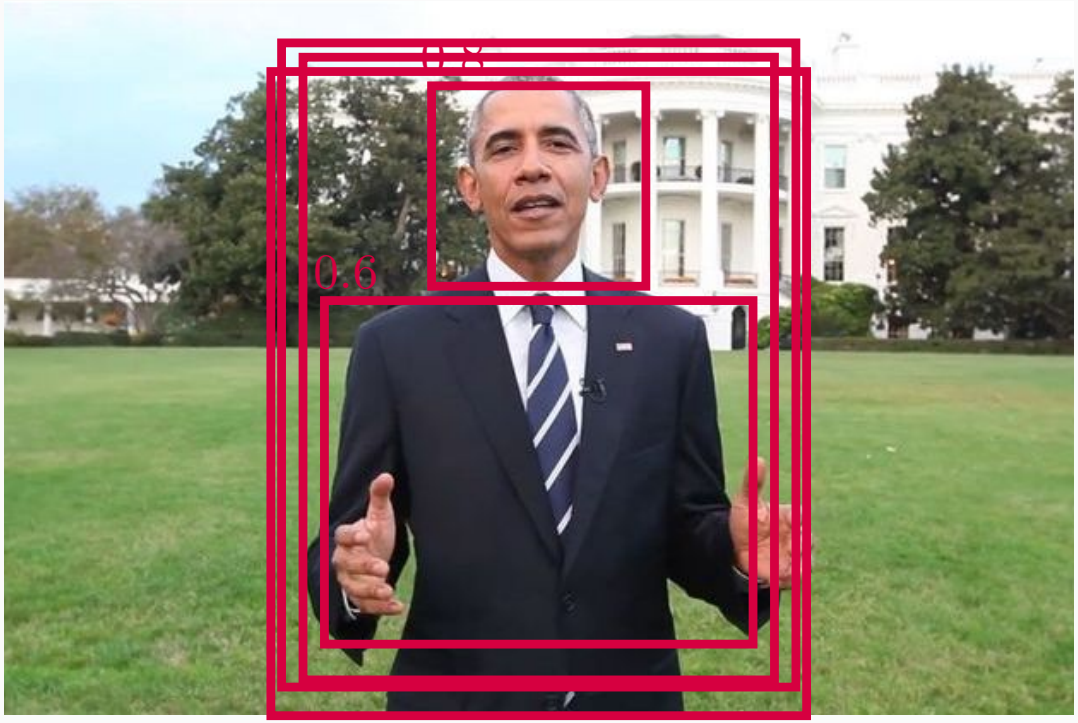


[Sermanet et al.] OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks

Duplicates During Detection $\rightsquigarrow$ Suppress Unnecessary Anchor Boxes

[Sermanet et al.] OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks

Non-Max Suppression Algorithm

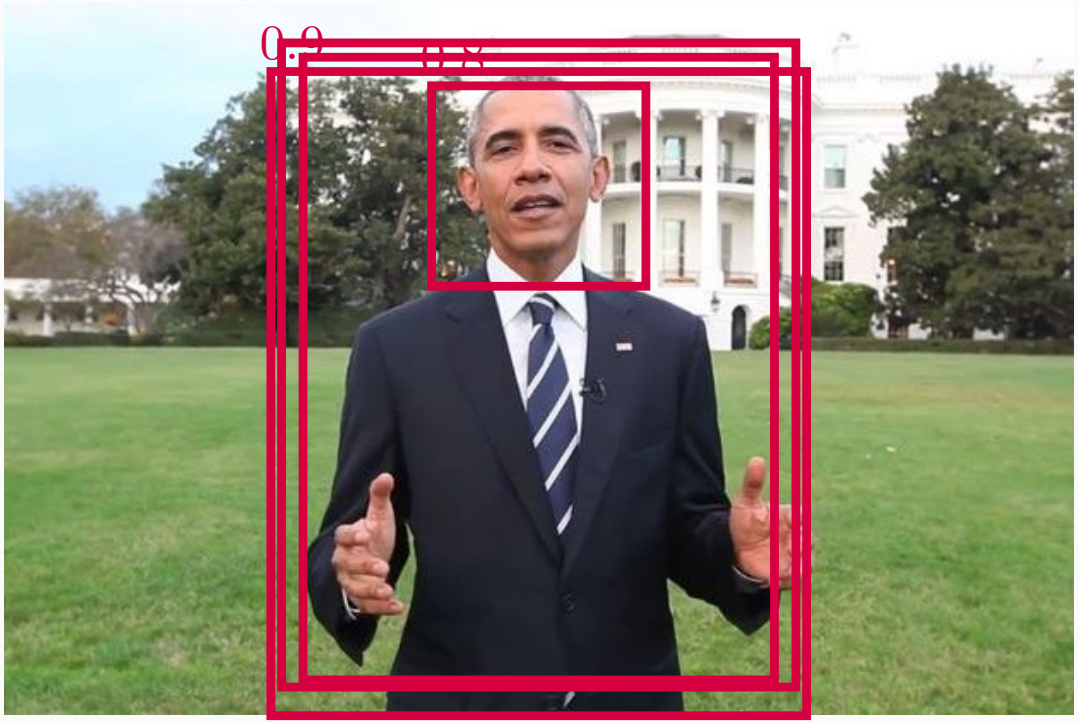


IoU: Intersection over Union

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2}$$

1. Suppression of boxes with a confidence index $< \alpha = 0.6$;
2. Selection among the remaining boxes of the b_{max} box with the highest confidence index;
3. Suppression of all the boxes having an IoU with $b_{max} > \beta = 0.5$.

Non-Max Suppression Algorithm

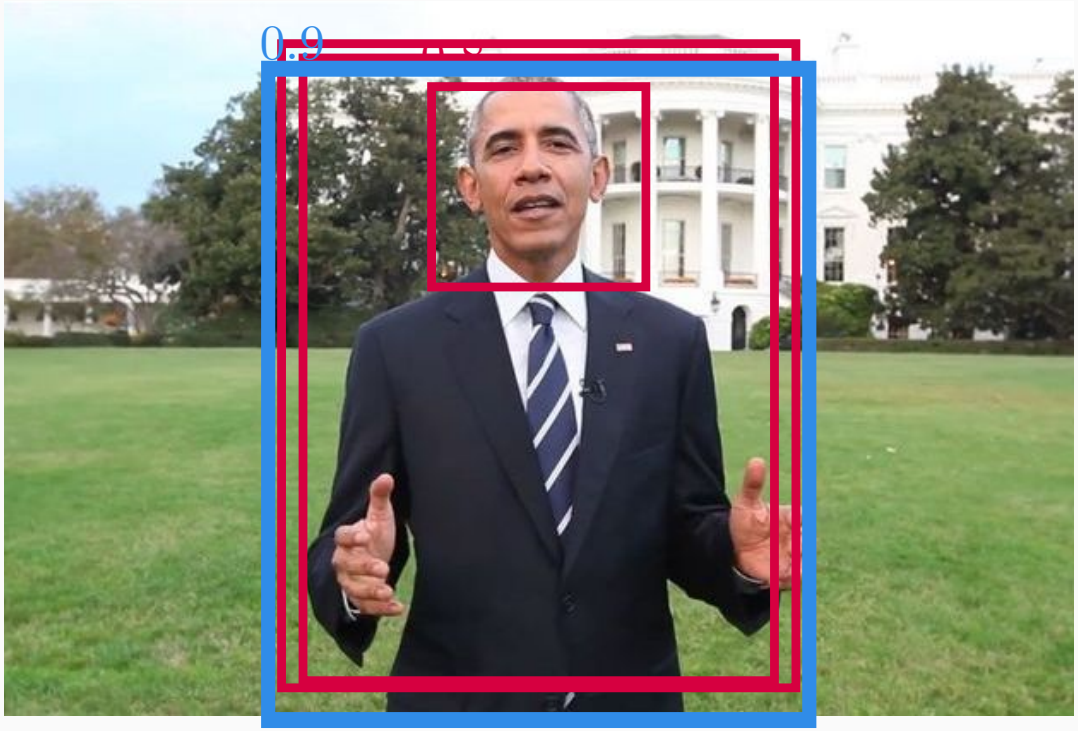


IoU: Intersection over Union

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2}$$

1. Suppression of boxes with a confidence index $< \alpha = 0.6$;
2. Selection among the remaining boxes of the b_{max} box with the highest confidence index;
3. Suppression of all the boxes having an IoU with $b_{max} > \beta = 0.5$.

Non-Max Suppression Algorithm

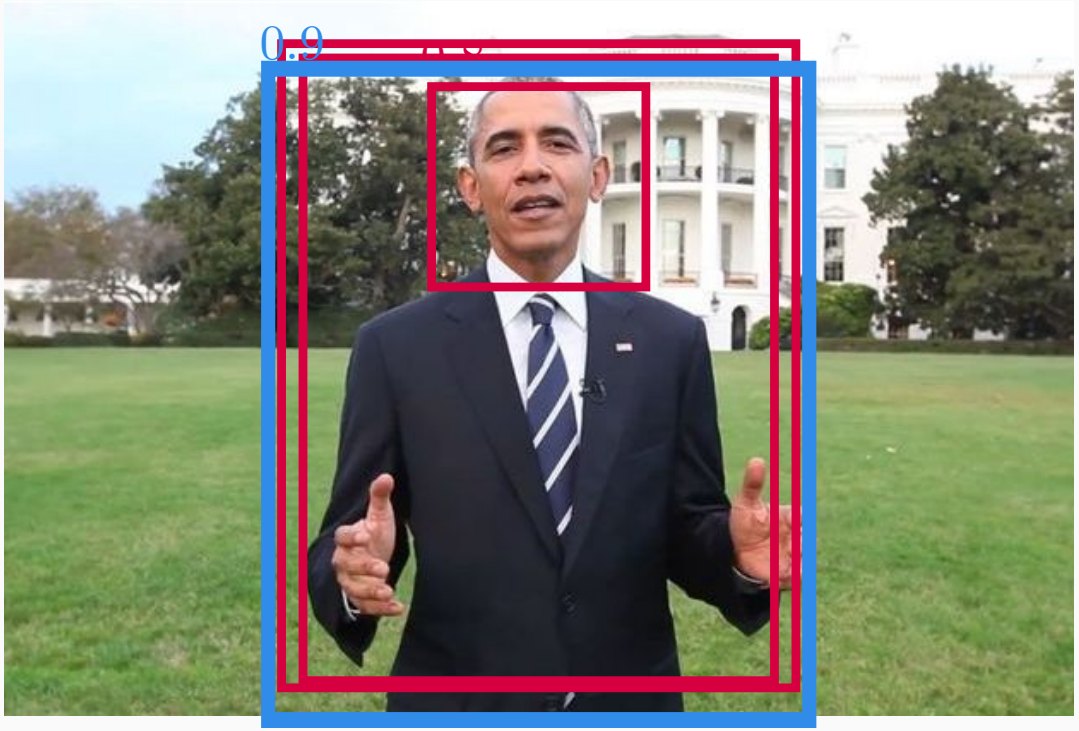


IoU: Intersection over Union

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2}$$

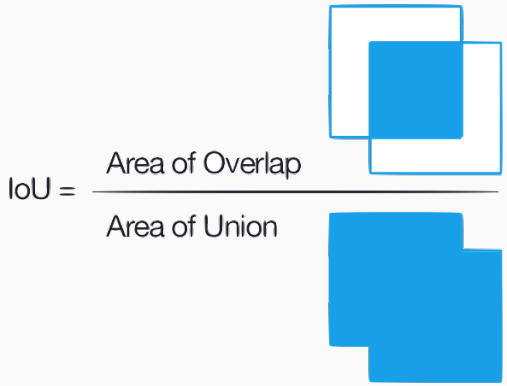
1. Suppression of boxes with a confidence index $< \alpha = 0.6$;
2. Selection among the remaining boxes of the b_{max} box with the highest confidence index;
3. Suppression of all the boxes having an IoU with $b_{max} > \beta = 0.5$.

Non-Max Suppression Algorithm



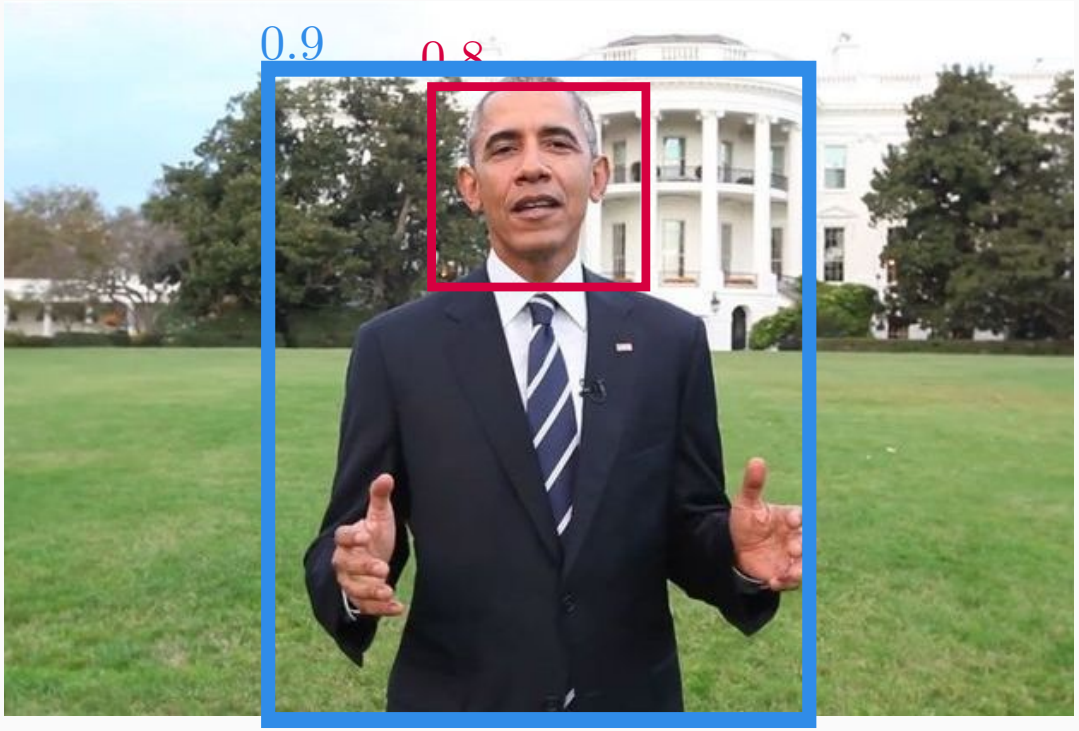
IoU: Intersection over Union

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2}$$



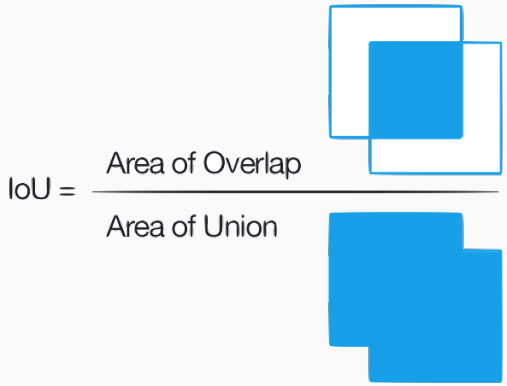
1. Suppression of boxes with a confidence index $< \alpha = 0.6$;
2. Selection among the remaining boxes of the b_{max} box with the highest confidence index;
3. Suppression of all the boxes having an IoU with $b_{max} > \beta = 0.5$.

Non-Max Suppression Algorithm



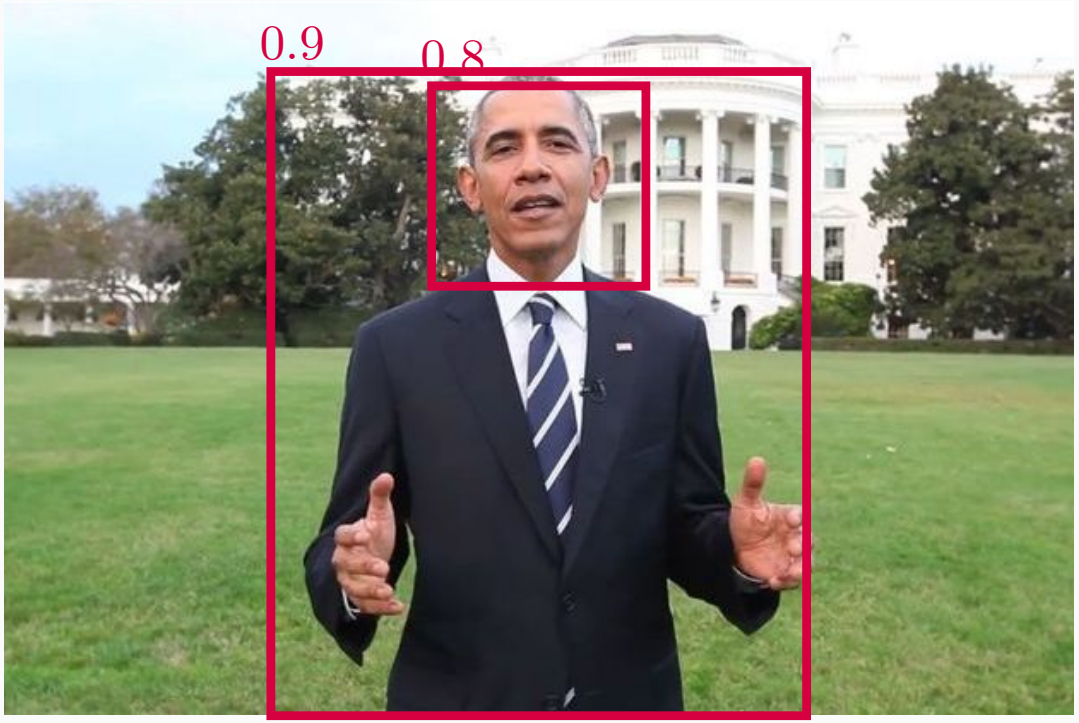
IoU: Intersection over Union

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2}$$



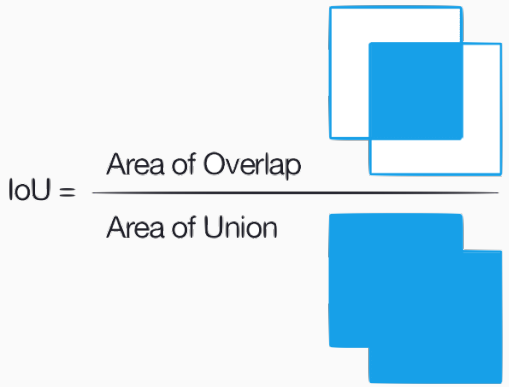
1. Suppression of boxes with a confidence index $< \alpha = 0.6$;
2. Selection among the remaining boxes of the b_{max} box with the highest confidence index;
3. Suppression of all the boxes having an IoU with $b_{max} > \beta = 0.5$.

Non-Max Suppression Algorithm



IoU: Intersection over Union

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2}$$



1. Suppression of boxes with a confidence index $< \alpha = 0.6$;
2. Selection among the remaining boxes of the b_{max} box with the highest confidence index;
3. Suppression of all the boxes having an IoU with $b_{max} > \beta = 0.5$.

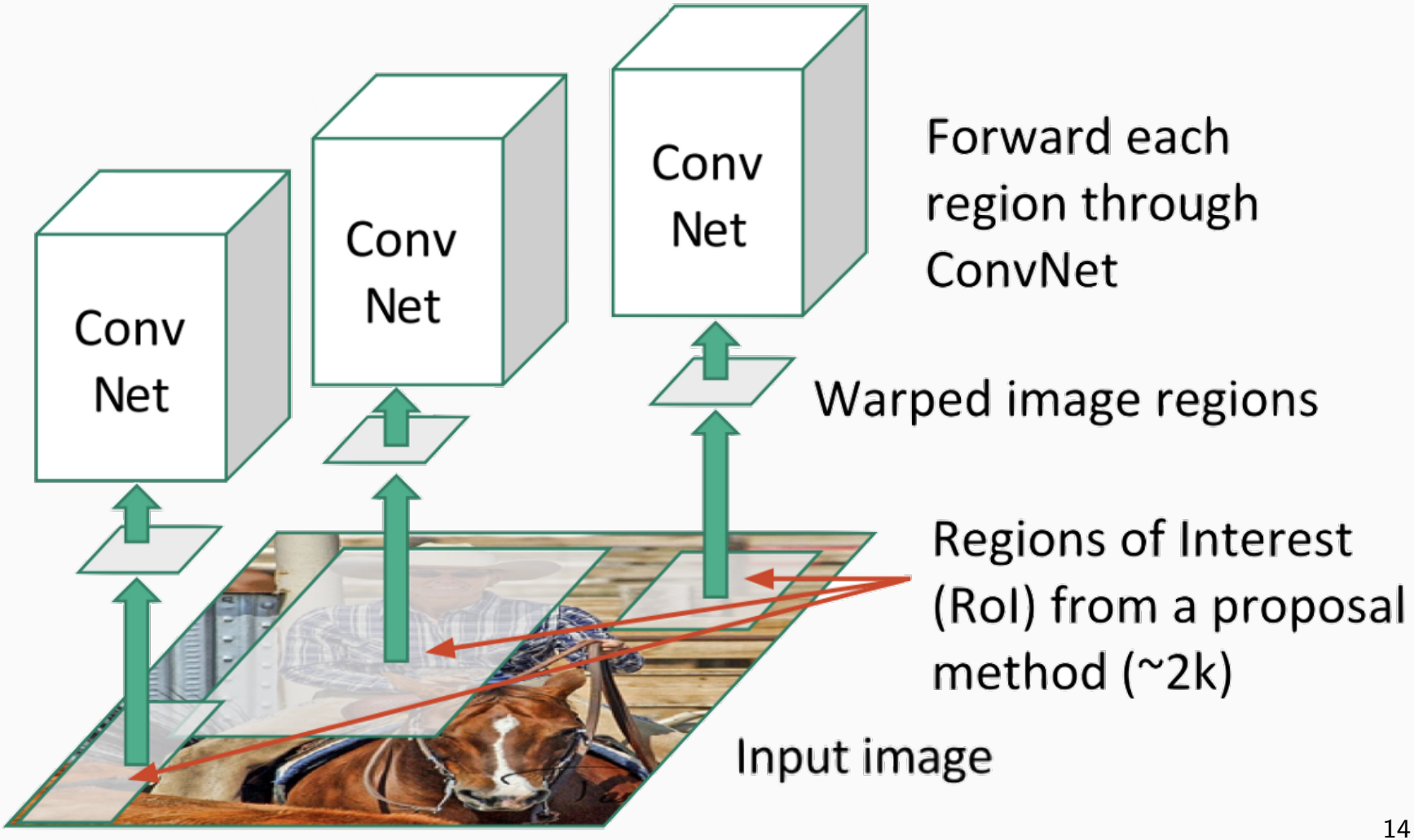
Region-Based Convolutional Neural Networks

Region-based Convolutional Neural Networks (Girshick *et al.*, 2014)

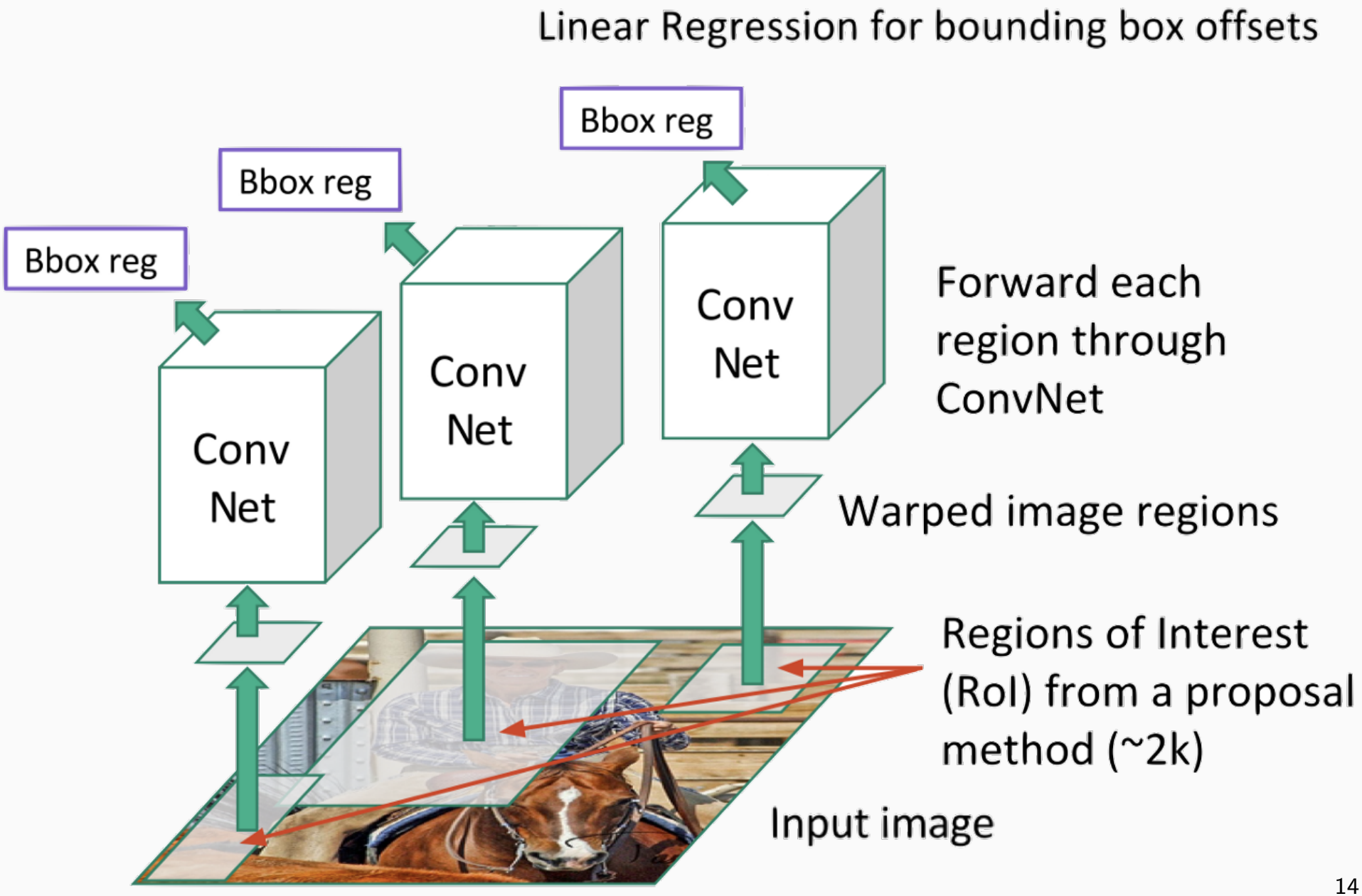
Regions of Interest (RoI) from a proposal method (~2k)

Input image

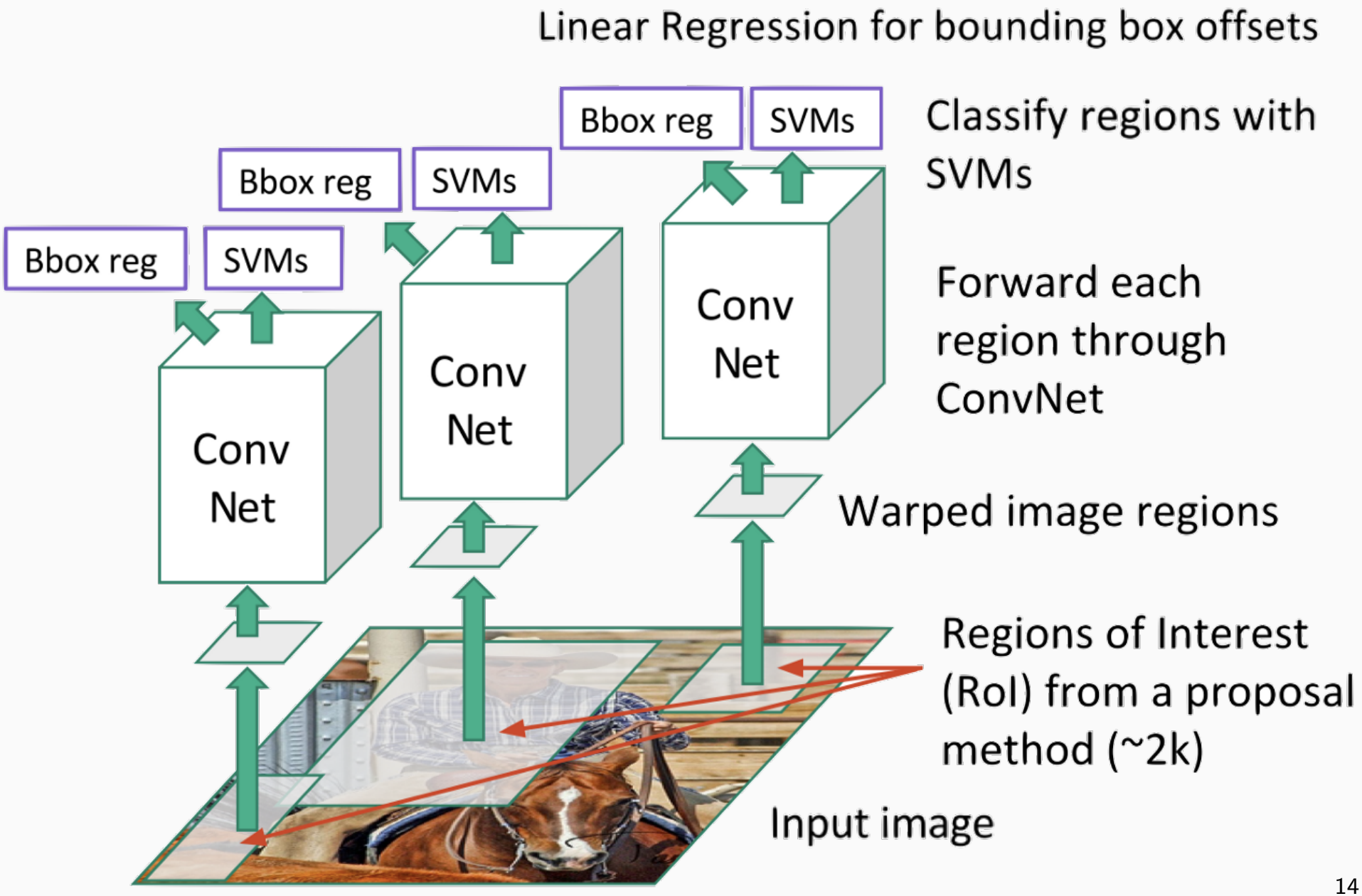
Region-based Convolutional Neural Networks (Girshick *et al.*, 2014)



Region-based Convolutional Neural Networks (Girshick *et al.*, 2014)

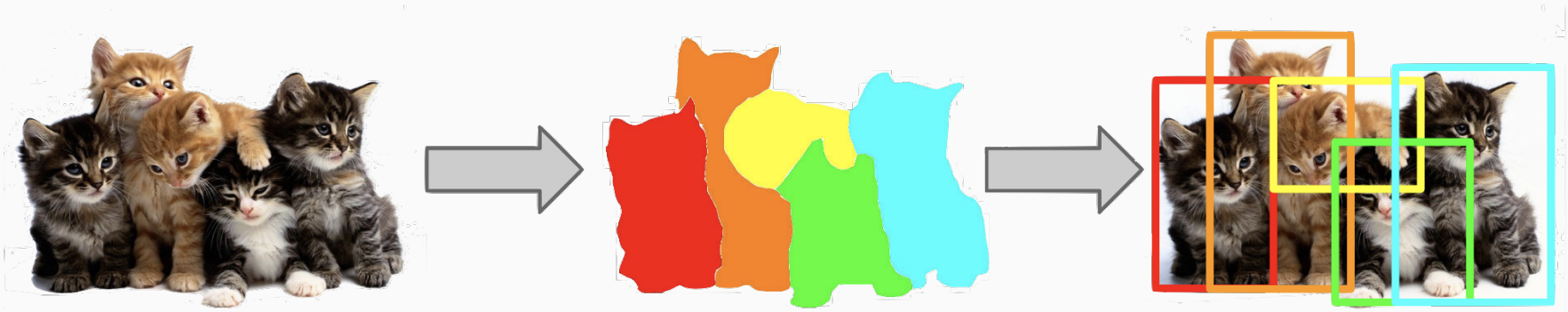


Region-based Convolutional Neural Networks (Girshick *et al.*, 2014)



Selective Search: An Example of RoI Proposal Method

Determine the Regions of Interest (RoI) $\rightsquigarrow$ Segmentation



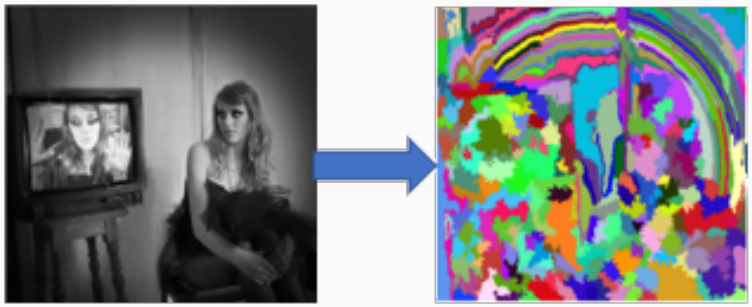
Idea: If we correctly segment the image before running object recognition, we can use our segmentations as candidate objects.

Selective Search: An Example of RoI Proposal Method

1. Generate initial sub-segmentation of input image.

2. Recursively combine small similar regions into larger ones, using **greedy** algo:
   - 2.1 From set of regions, choose two that are most similar.
   - 2.2 Combine them into a single, larger region.
   - 2.3 Repeat the above steps for multiple iterations.

3. Use the segmented region proposals to generate candidate object locations.

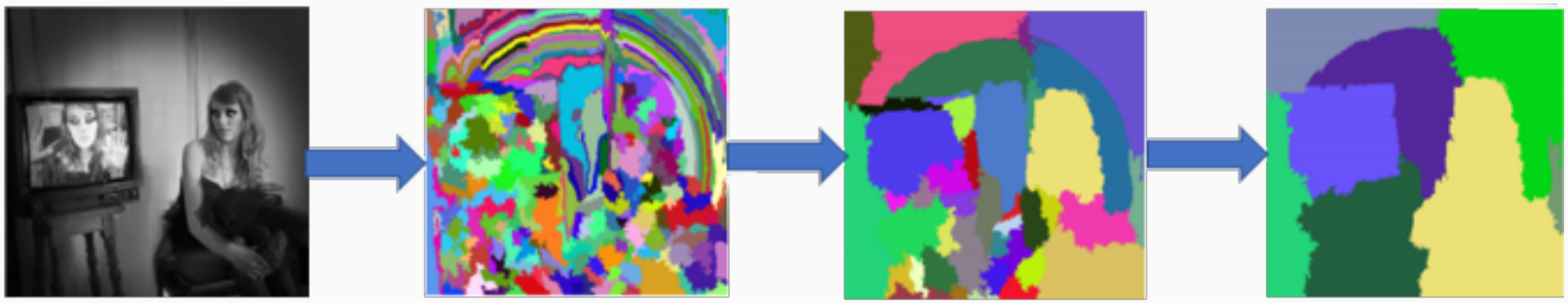


Input Image

After Initial Segmentation

Selective Search: An Example of RoI Proposal Method

1. Generate initial sub-segmentation of input image.
2. Recursively combine small similar regions into larger ones, using **greedy** algo:
   - 2.1 From set of regions, choose two that are most similar.
   - 2.2 Combine them into a single, larger region.
   - 2.3 Repeat the above steps for multiple iterations.
3. Use the segmented region proposals to generate candidate object locations.

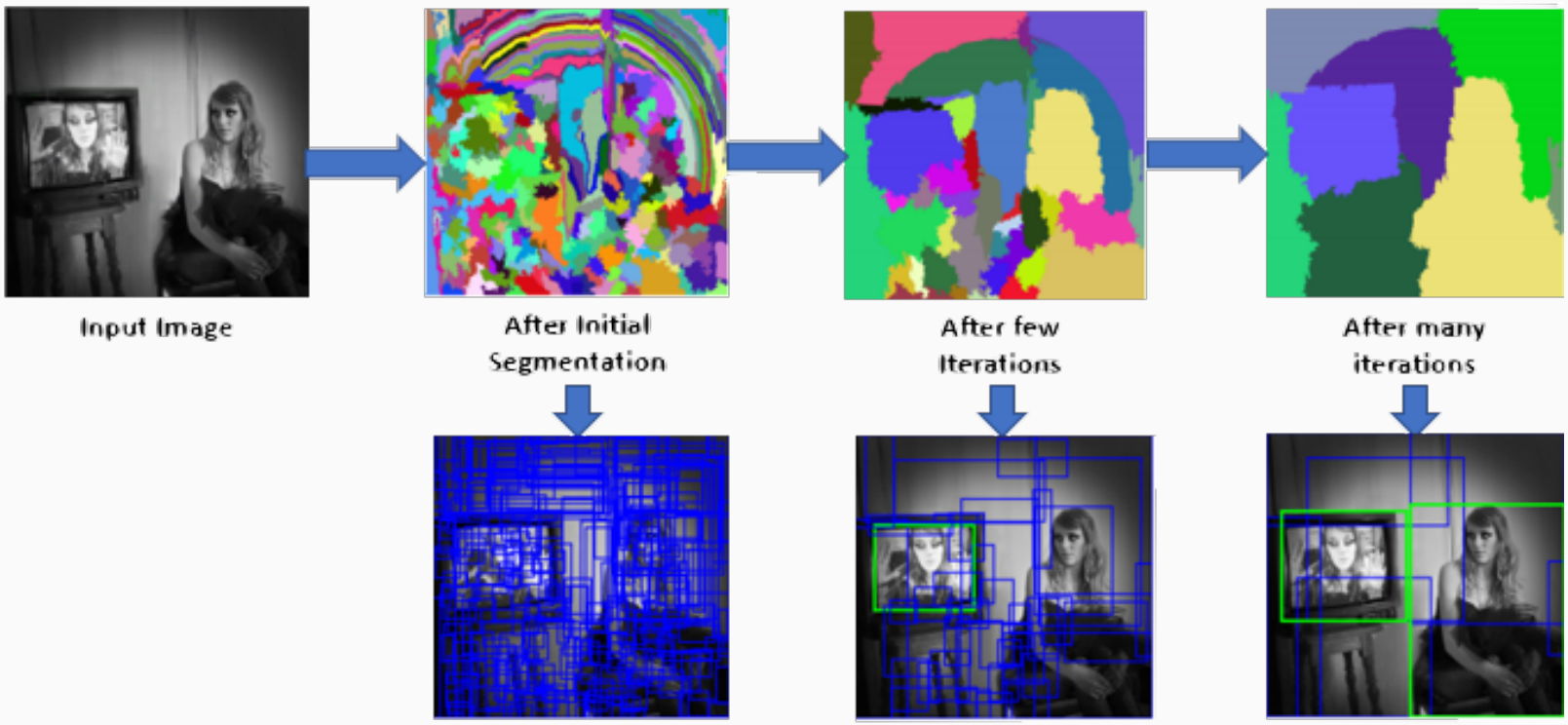


Selective Search: An Example of RoI Proposal Method

1. Generate initial sub-segmentation of input image.

2. Recursively combine small similar regions into larger ones, using **greedy** algo:
   - 2.1 From set of regions, choose two that are most similar.
   - 2.2 Combine them into a single, larger region.
   - 2.3 Repeat the above steps for multiple iterations.

3. Use the segmented region proposals to generate candidate object locations.



Limitations of R-CNN

1. Extracting 2 000 regions for each image based on selective search.

2. Extracting features using CNN for every image region.
   $\rightsquigarrow N$ images $\longleftrightarrow 2\,000N$ CNN features.

3. Entire process of R-CNN has three models:
   - CNN for feature extraction,
   - Linear SVM classifier for identifying objects,
   - Regression model for tightening the bounding boxes.

4. Slow training (84h), Takes a lot of disk space.

5. Slow inference/detection: 47s per image with VGG16.

Main performance **bottleneck** of R-CNN:
Need to *independently extract features for each proposed region.*

Limitations of R-CNN

1. Extracting 2 000 regions for each image based on selective search.

2. Extracting features using CNN for every image region.
   $\rightsquigarrow N$ images $\longleftrightarrow 2\,000N$ CNN features.

3. Entire process of R-CNN has three models:
   - CNN for feature extraction,
   - Linear SVM classifier for identifying objects,
   - Regression model for tightening the bounding boxes.

4. Slow training (84h), Takes a lot of disk space.

5. Slow inference/detection: 47s per image with VGG16.

Main performance **bottleneck** of R-CNN:
Need to *independently extract features for each proposed region.*

Region-Based Convolutional Neural Networks

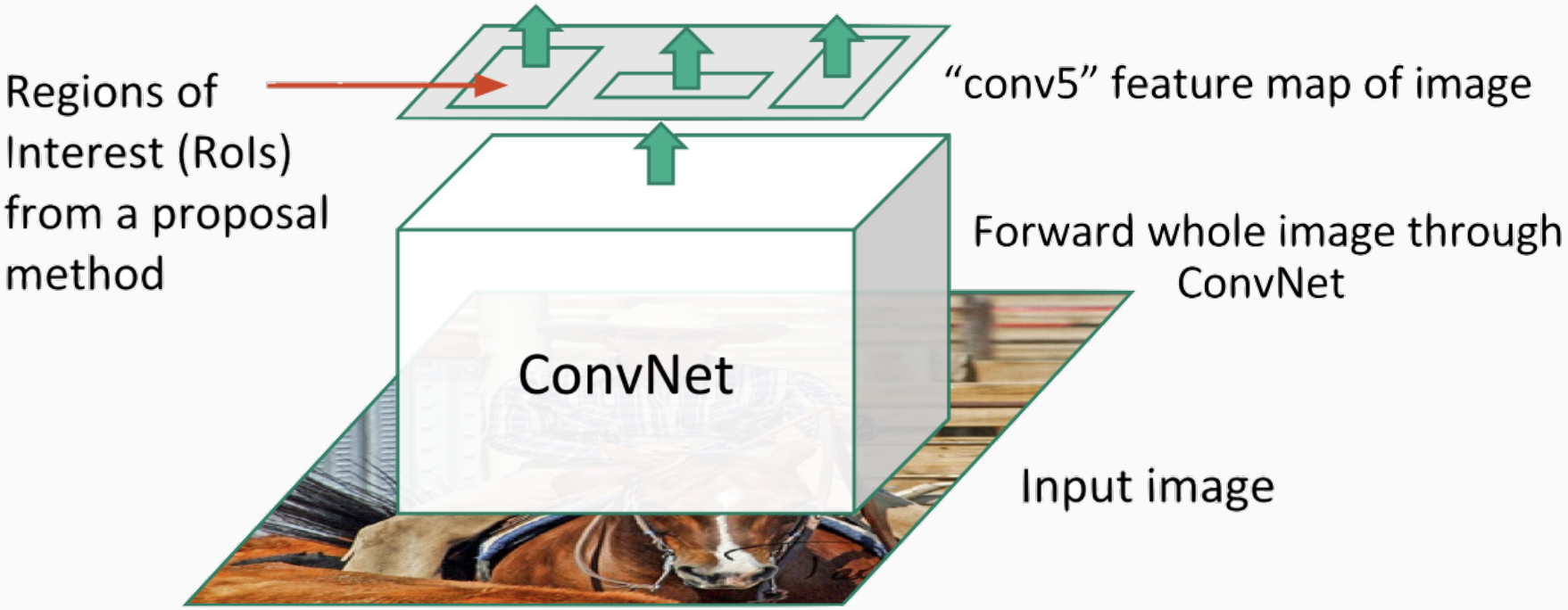
Fast R-CNN

Instead of running a CNN 2000 times per image,
we can run it just once per image and get all the regions of interest

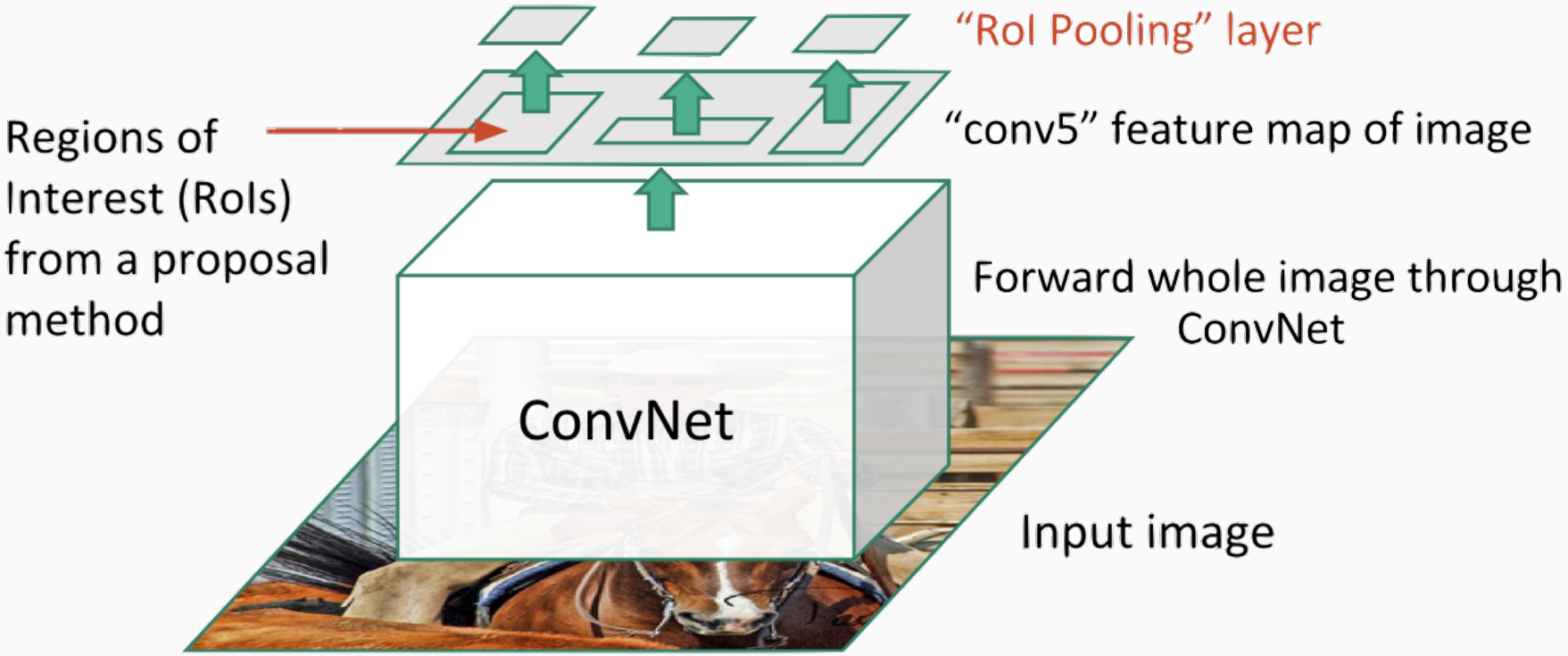
1. Performing feature extraction over the image before proposing regions,
   $\rightsquigarrow$ Run only **one** CNN over the entire image instead of 2000 CNN's over 2000 overlapping regions.

2. Replacing the SVM with a softmax layer,
   $\rightsquigarrow$ Extend the neural network for predictions instead of creating a new model.

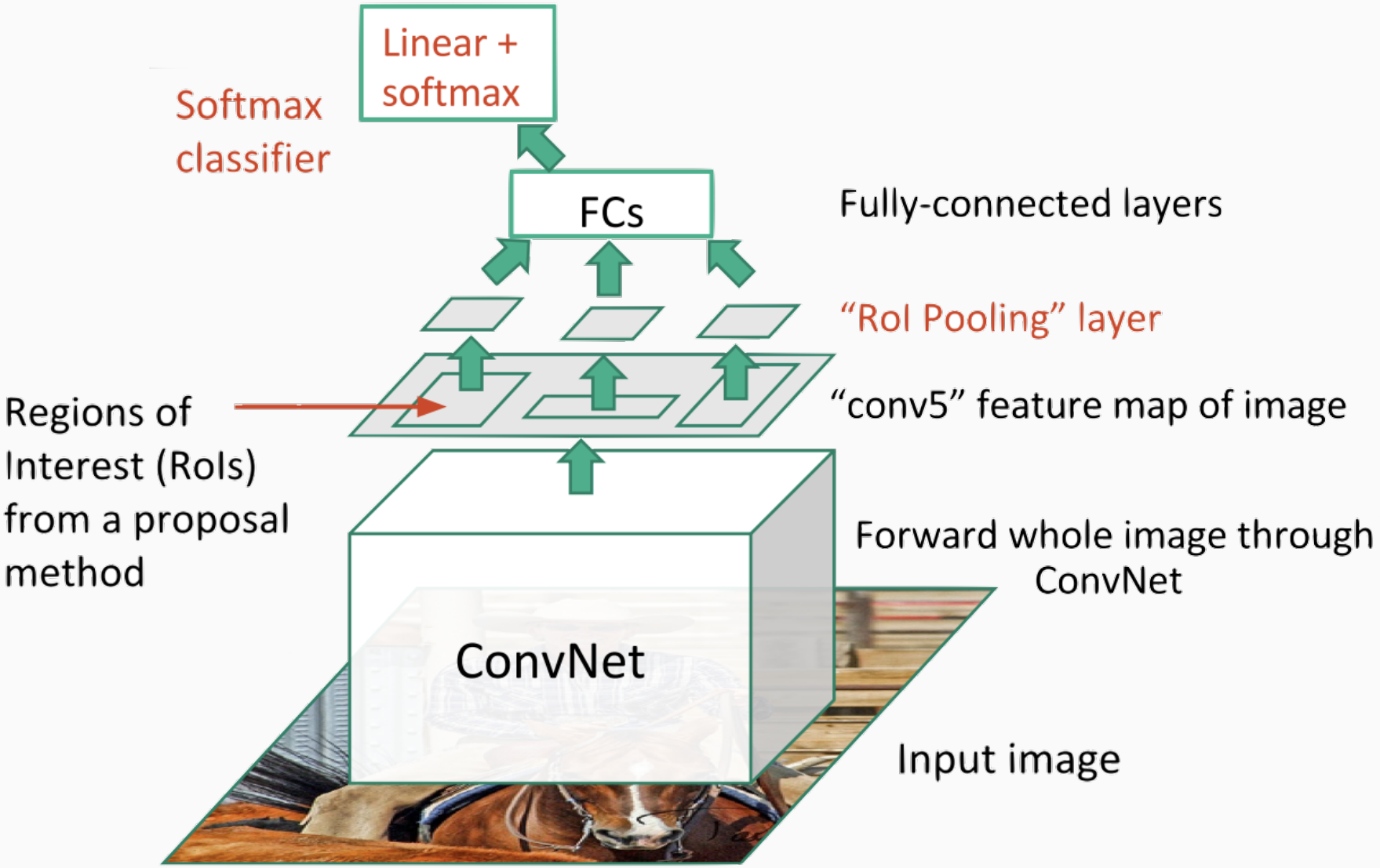
Fast R-CNN (Girshick, 2015)



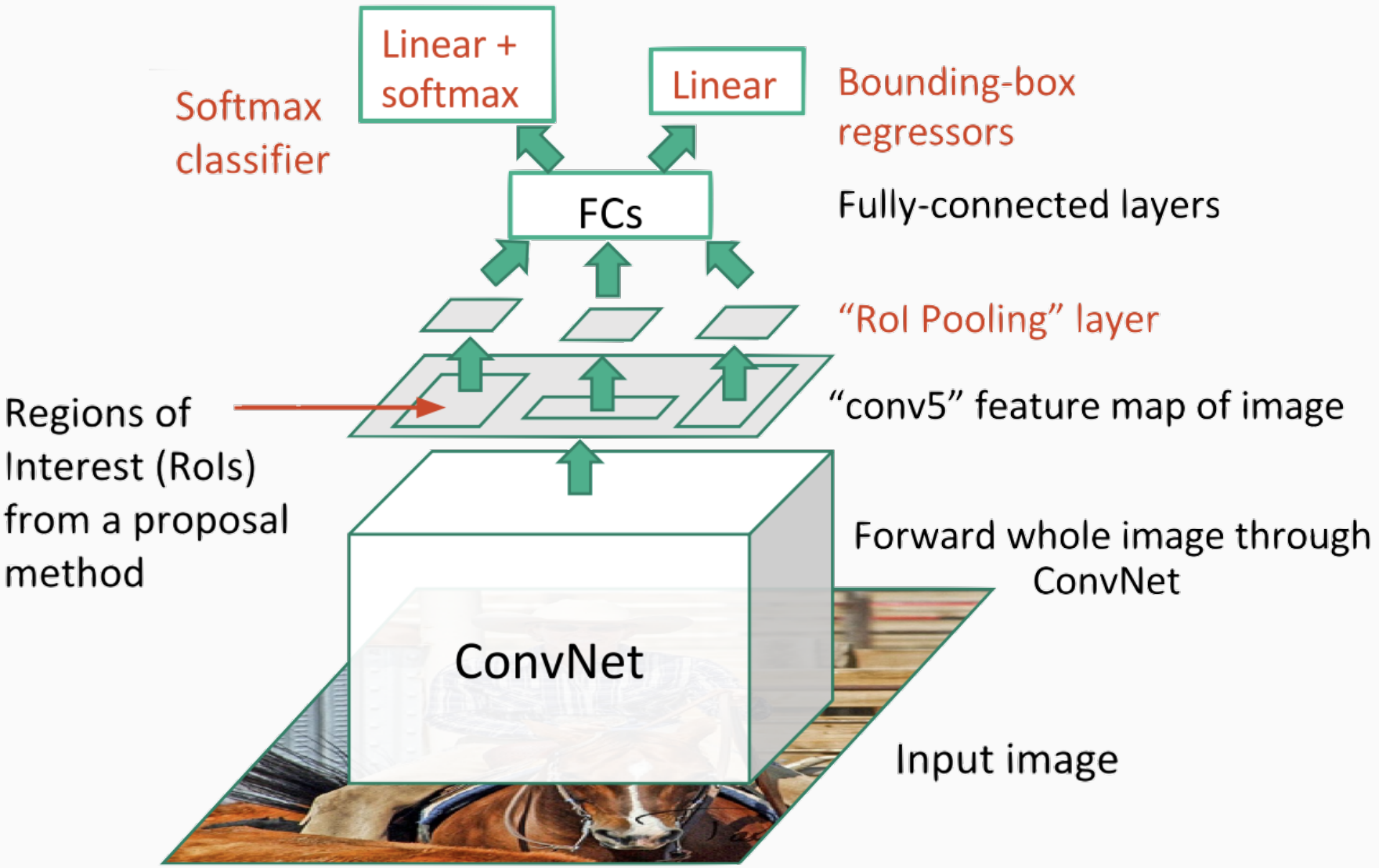
Fast R-CNN (Girshick, 2015)



Fast R-CNN (Girshick, 2015)



Fast R-CNN (Girshick, 2015)



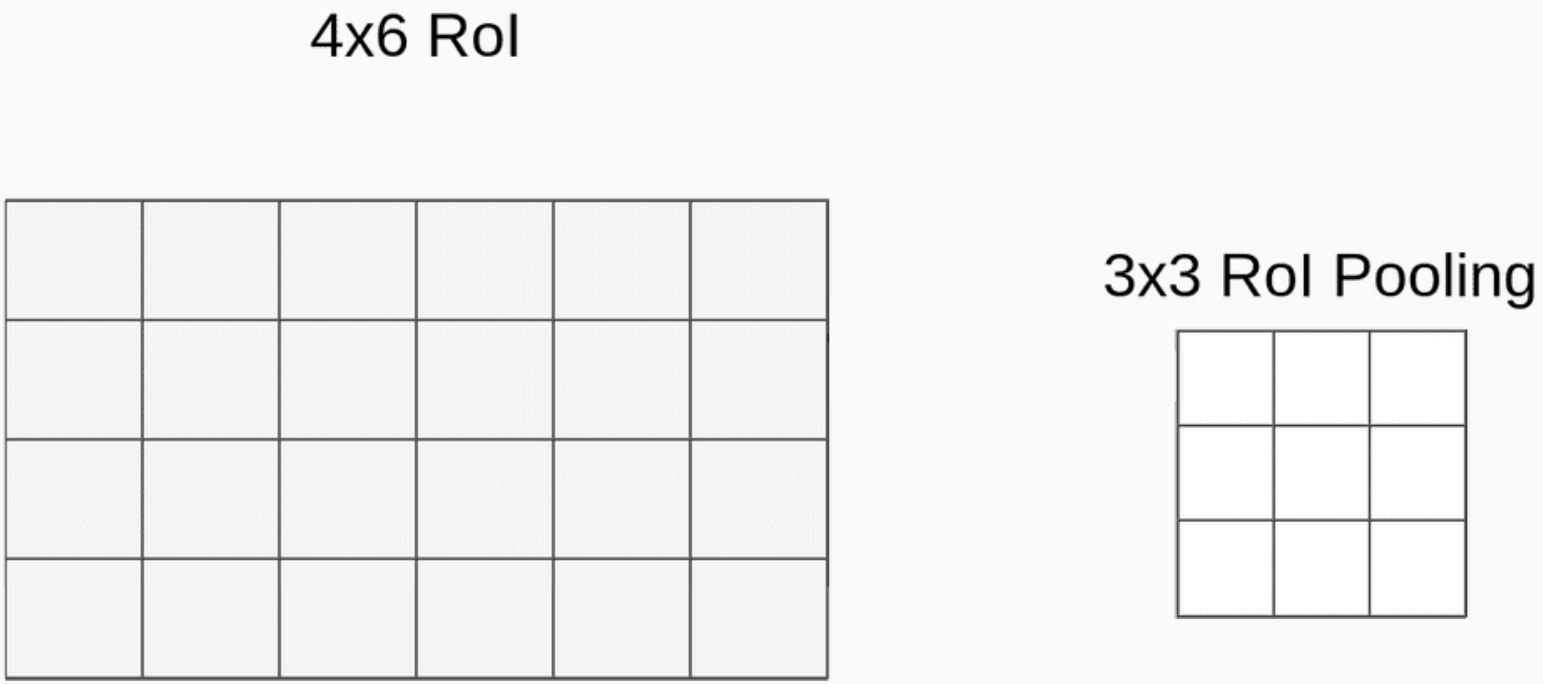
Fast R-CNN

1. A single model which:
   - extracts features from the regions,
   - divides them into different classes,
   - and returns the boundary boxes for the identified classes simultaneously.

2. Faster than R-CNN. But, still use Selective Search algorithm, which is a slow and time consuming process (2s per image).

RoI Pooling

- After RoI pooling layer: Fully connected layer with a fixed size.
  RoIs have different sizes $\rightsquigarrow$ Pool them into the same size

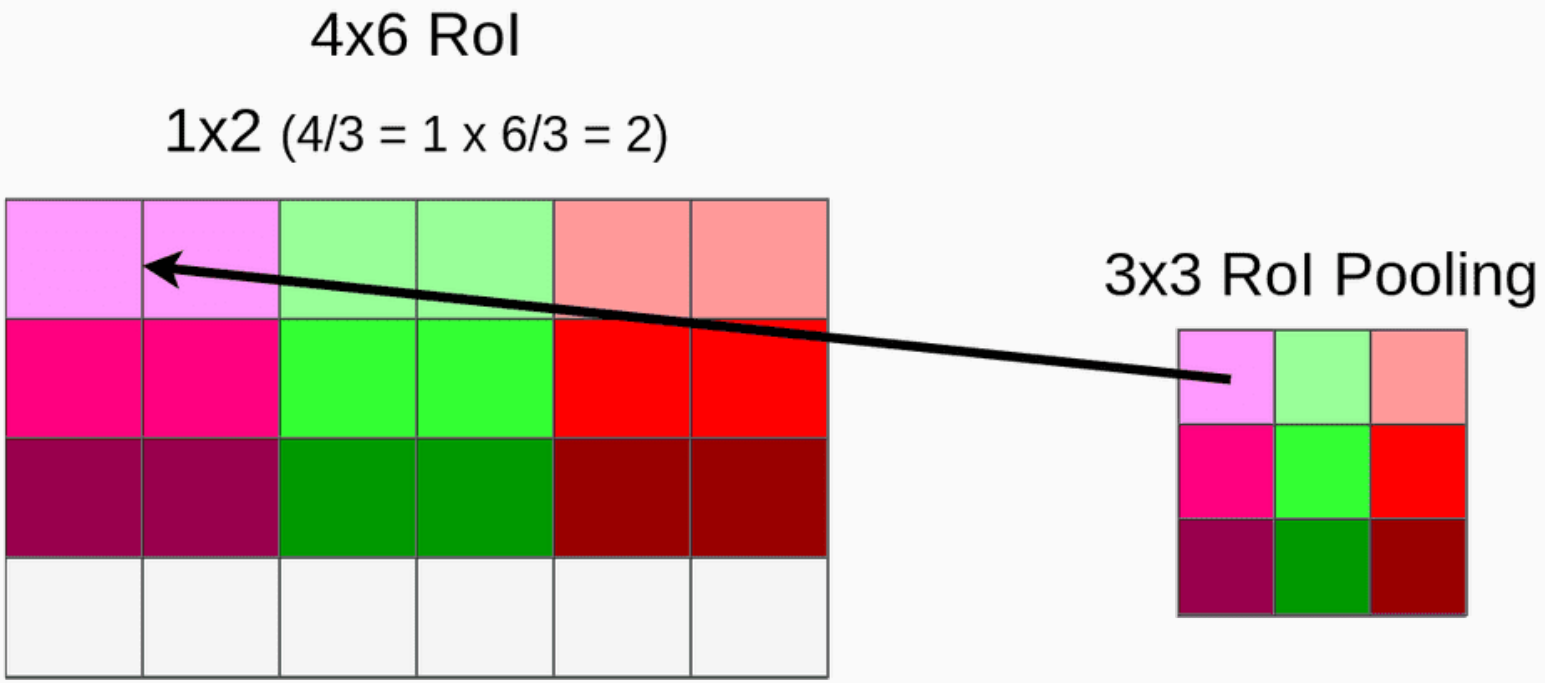


- Same process applied to each RoI of the original image
  $\rightsquigarrow$ Hundreds or even thousands of "small" matrices of the same size.

- Each one is sent to the rest of the network.
  For each, the model generates bbox and class separately.

RoI Pooling

- After RoI pooling layer: Fully connected layer with a fixed size.
  RoIs have different sizes $\rightsquigarrow$ Pool them into the same size



- Same process applied to each RoI of the original image
  $\rightsquigarrow$ Hundreds or even thousands of "small" matrices of the same size.

- Each one is sent to the rest of the network.
  For each, the model generates bbox and class separately.

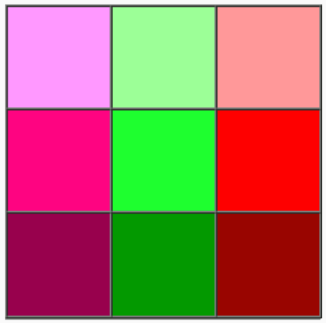
RoI Pooling

- After RoI pooling layer: Fully connected layer with a fixed size.
  RoIs have different sizes $\rightsquigarrow$ Pool them into the same size

4x6 RoI

0.1	0.2	0.3	0.4	0.5	0.6
1	0.7	0.2	0.6	0.1	0.9
0.9	0.8	0.7	0.3	0.5	0.2

3x3 RoI Pooling



- Same process applied to each RoI of the original image
  $\rightsquigarrow$ Hundreds or even thousands of "small" matrices of the same size.
- Each one is sent to the rest of the network.
  For each, the model generates bbox and class separately.

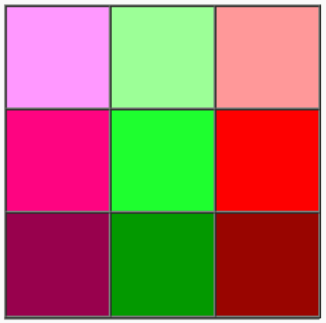
RoI Pooling

- After RoI pooling layer: Fully connected layer with a fixed size.
  RoIs have different sizes $\rightsquigarrow$ Pool them into the same size

4x6 RoI

0.1	0.2	0.3	0.4	0.5	0.6
1	0.7	0.2	0.6	0.1	0.9
0.9	0.8	0.7	0.3	0.5	0.2

3x3 RoI Pooling



- Same process applied to each RoI of the original image
  $\rightsquigarrow$ Hundreds or even thousands of "small" matrices of the same size.
- Each one is sent to the rest of the network.
  For each, the model generates bbox and class separately.

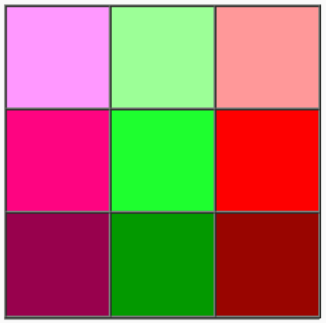
RoI Pooling

- After RoI pooling layer: Fully connected layer with a fixed size.
  RoIs have different sizes $\rightsquigarrow$ Pool them into the same size

4x6 RoI

0.1	0.2	0.3	0.4	0.5	0.6
1	0.7	0.2	0.6	0.1	0.9
0.9	0.8	0.7	0.3	0.5	0.2

3x3 RoI Pooling



- Same process applied to each RoI of the original image
  $\rightsquigarrow$ Hundreds or even thousands of "small" matrices of the same size.
- Each one is sent to the rest of the network.
  For each, the model generates bbox and class separately.

Region-Based Convolutional Neural Networks

Region Proposal Network

Idea: Replace *slow* selective search with *fast* net: **Region proposal network** (**RPN**)

(i) Let k fixed-ratio anchor boxes (default bounding boxes).

(ii) At the last layer of an initial CNN: 3×3 sliding window

(iii) For each sliding-window, generate several possible regions based on anchor boxes.

(iv) Each region proposal consists of:

a) An "objectness" score for that region,

b) 4 coordinates representing the bounding box of the region.

[Ren et al.] Faster R-CNN : Towards Real-Time Object Detection with Region Proposal Networks

Region Proposal Network

Idea: Replace *slow* selective search with *fast* net: **Region proposal network** (**RPN**)

(i) Let k fixed-ratio anchor boxes (default bounding boxes).

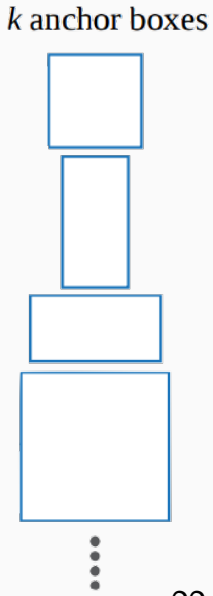
(ii) At the last layer of an initial CNN: 3×3 sliding window

(iii) For each sliding-window, generate several possible regions based on anchor boxes.

(iv) Each region proposal consists of:

a) An "objectness" score for that region,

b) 4 coordinates representing the bounding box of the region.



[Ren et al.] Faster R-CNN : Towards Real-Time Object Detection with Region Proposal Networks

Region Proposal Network

Idea: Replace *slow* selective search with *fast* net: **Region proposal network** (**RPN**)

(i) Let k fixed-ratio anchor boxes (default bounding boxes).

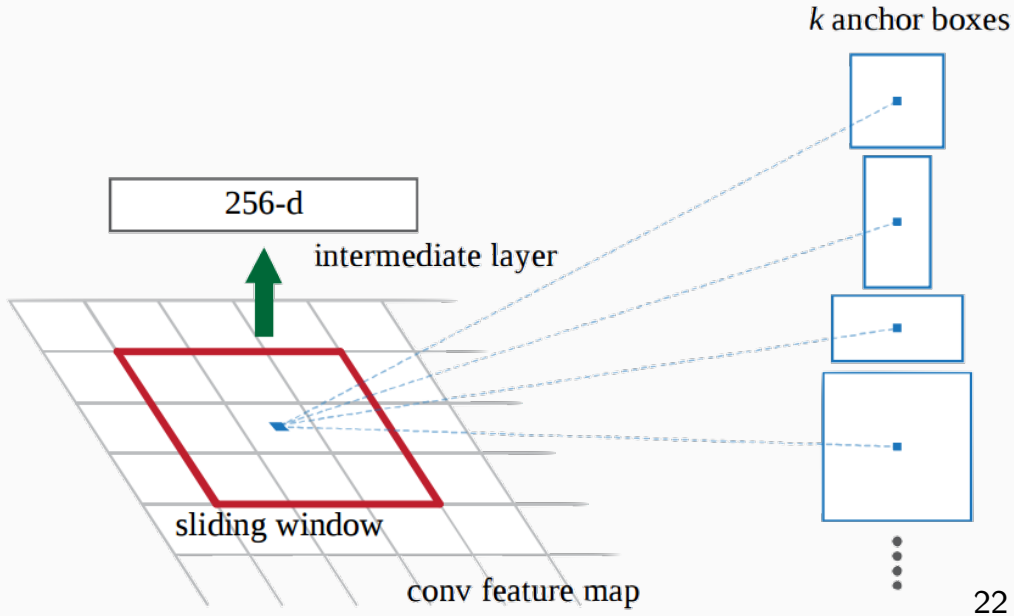
(ii) At the last layer of an initial CNN: 3×3 sliding window

(iii) For each sliding-window, generate several possible regions based on anchor boxes.

(iv) Each region proposal consists of:

a) An "objectness" score for that region,

b) 4 coordinates representing the bounding box of the region.



[Ren et al.] Faster R-CNN : Towards Real-Time Object Detection with Region Proposal Networks

Region Proposal Network

Idea: Replace *slow* selective search with *fast* net: **Region proposal network** (**RPN**)

(i) Let k fixed-ratio anchor boxes (default bounding boxes).

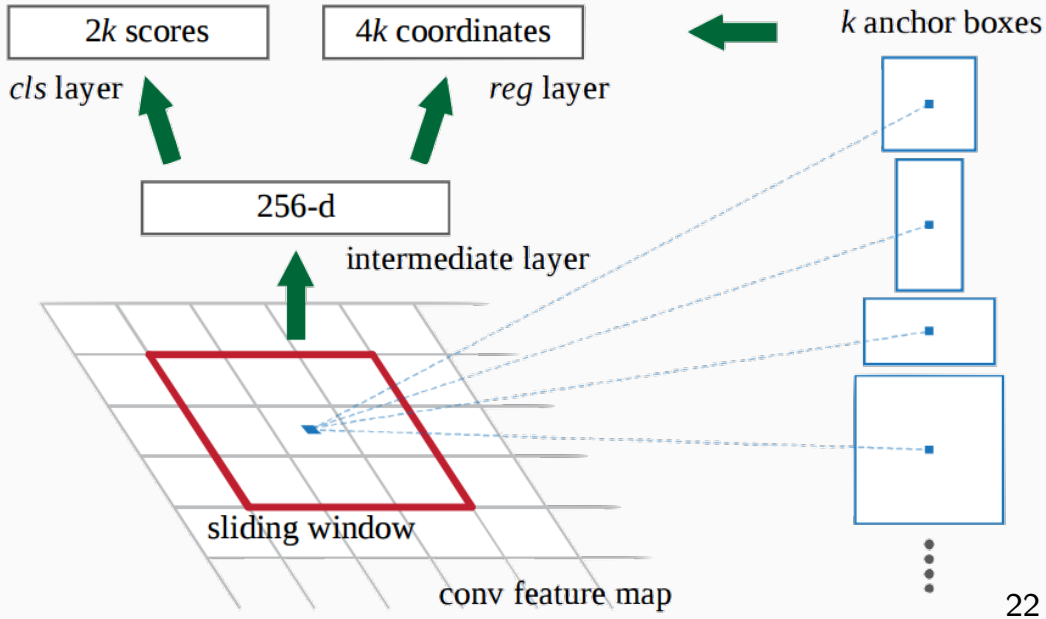
(ii) At the last layer of an initial CNN: 3×3 sliding window

(iii) For each sliding-window, generate several possible regions based on anchor boxes.

(iv) Each region proposal consists of:

a) An "objectness" score for that region,

b) 4 coordinates representing the bounding box of the region.



[Ren et al.] Faster R-CNN : Towards Real-Time Object Detection with Region Proposal Networks

Faster R-CNN (Ren, *et al.*, 2015)

Remarks on RPN:

- $2k$ scores $\equiv$ Softmax proba of each of the k bounding boxes being an "object".

- RPN outputs bounding box coordinates, **but** it does not try to classify any potential objects!
  Its job: Proposing object regions.

- If an anchor box has a good "objectness", its coordinates get passed forward as a region proposal.

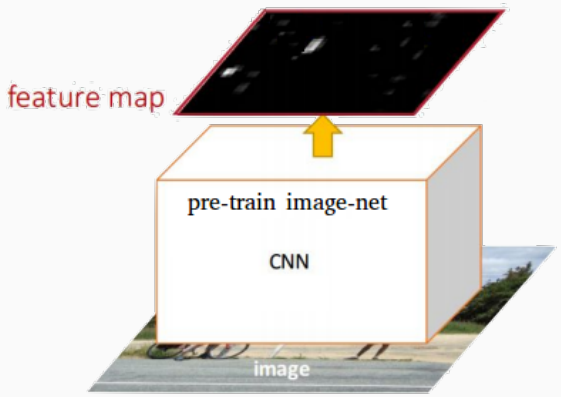
Faster R-CNN = RPN + Fast R-CNN (Ren, *et al.*, 2015)

Remarks on RPN:

- $2k$ scores $\equiv$ Softmax proba of each of the k bounding boxes being an "object".

- RPN outputs bounding box coordinates, **but** it does not try to classify any potential objects!
  Its job: Proposing object regions.

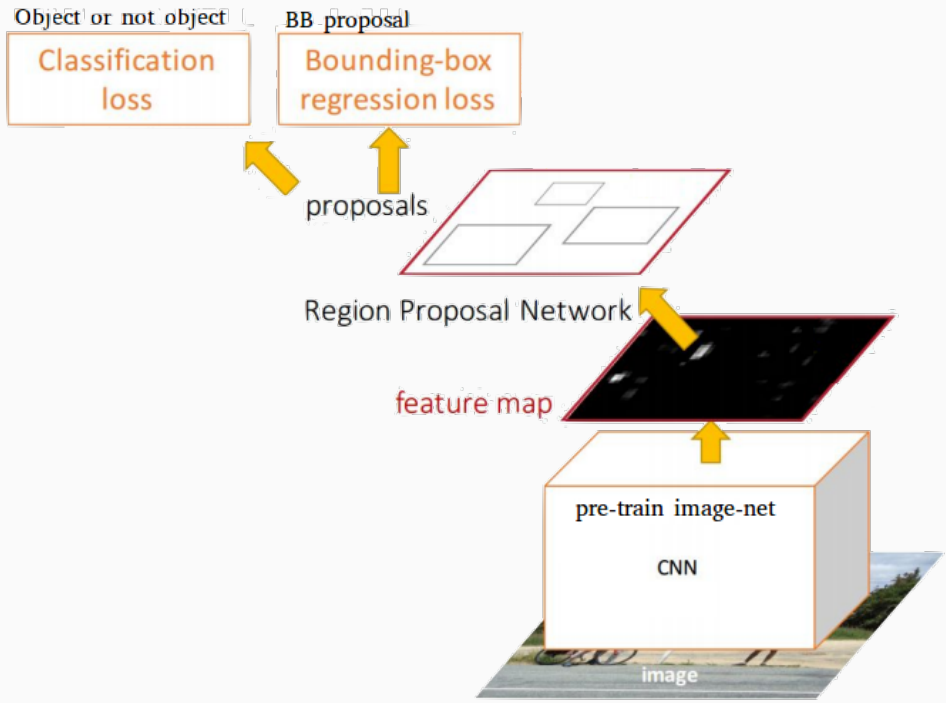
- If an anchor box has a good "objectness", its coordinates get passed forward as a region proposal.



Faster R-CNN = RPN + Fast R-CNN (Ren, *et al.*, 2015)

Remarks on RPN:

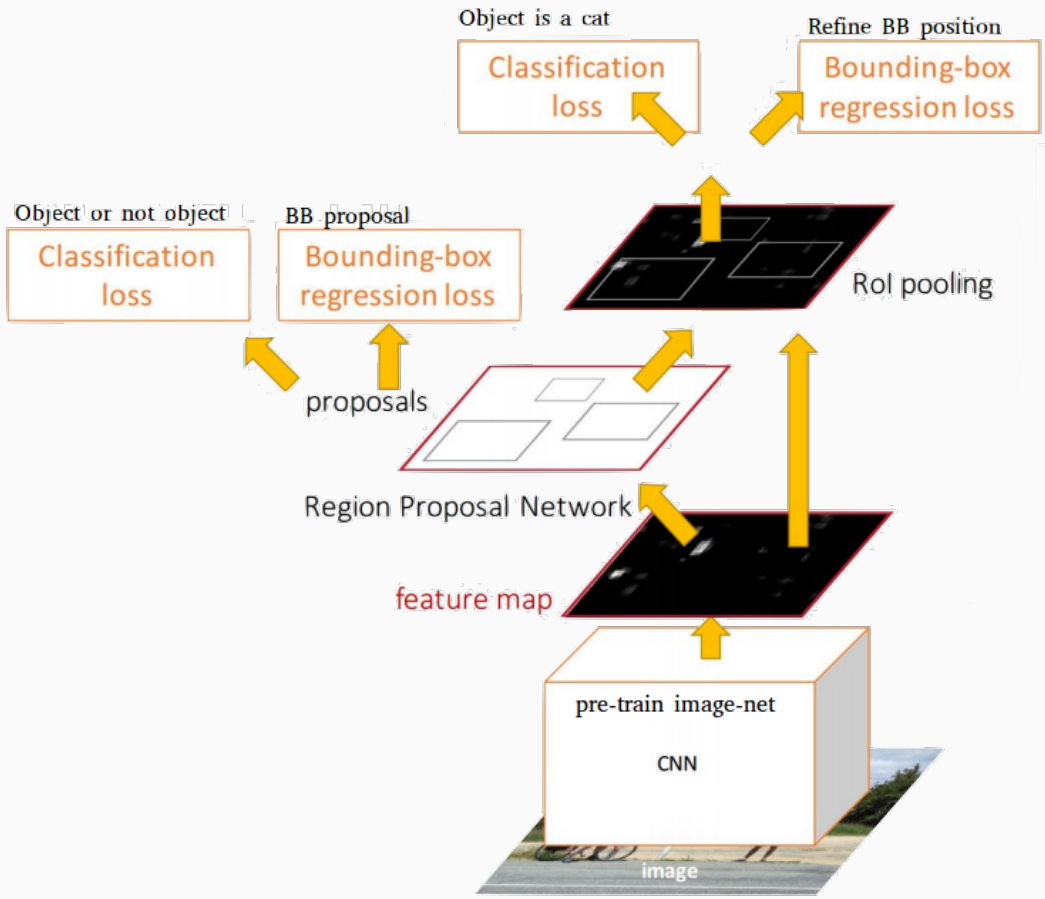
- $2k$ scores $\equiv$ Softmax proba of each of the k bounding boxes being an "object".
- RPN outputs bounding box coordinates, **but** it does not try to classify any potential objects!
  Its job: Proposing object regions.
- If an anchor box has a good "objectness", its coordinates get passed forward as a region proposal.



Faster R-CNN = RPN + Fast R-CNN (Ren, *et al.*, 2015)

Remarks on RPN:

- $2k$ scores $\equiv$ Softmax proba of each of the k bounding boxes being an "object".
- RPN outputs bounding box coordinates, **but** it does not try to classify any potential objects!
  Its job: Proposing object regions.
- If an anchor box has a good "objectness", its coordinates get passed forward as a region proposal.



A few words about Faster R-CNN:

- Faster R-CNN achieved much better speeds and is state-of-the-art accuracy
- Future models greatly increased detection speeds,
  But only few were able to outperform Faster R-CNN by a significant margin.

$\rightsquigarrow$ *Faster R-CNN may not be the simplest or fastest method for object detection, but it is still one of the most successful.*

Example: Tensorflow's Faster R-CNN with Inception ResNet is their slowest but most accurate model.

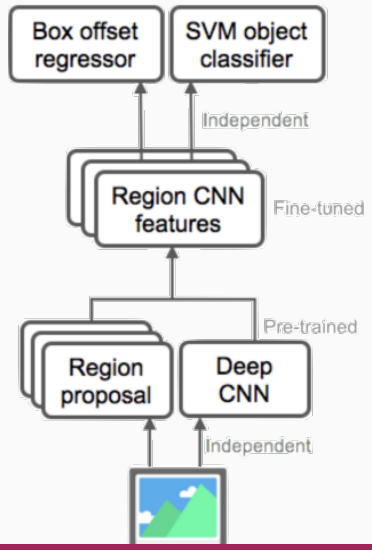
Region-Based Convolutional Neural Networks

Summary of Region-Based Convolutional Neural Networks

Algo	Features	Pred time/img	Limitations
R-CNN	• Selective search to generate regions, • Extract $2k$ regions per image.	40 s	• Each region passed to the CNN separately $\rightsquigarrow$ High comput. time, • Three different models to predict.
Fast R-CNN	• Each image forwarded once to the CNN, Feature maps extracted, • Selective search used on these maps to generate predictions, • Combine the 3 models used in RCNN together.	2 s	• Selective search slow $\rightsquigarrow$ Computation time still high.
Faster R-CNN	• Replaces selective search method with region proposal network.	0.2 s	• Several systems work one after the other: Performance of the following ones depends on the previous ones.

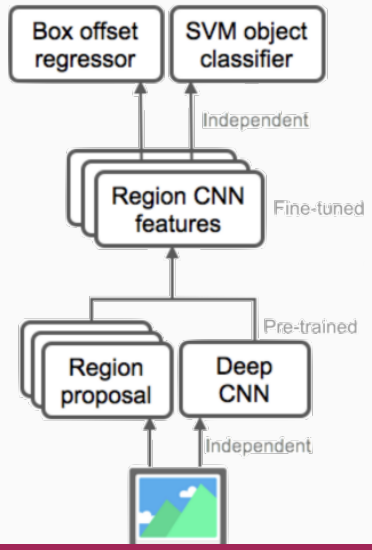
Summary of Region-Based Convolutional Neural Networks

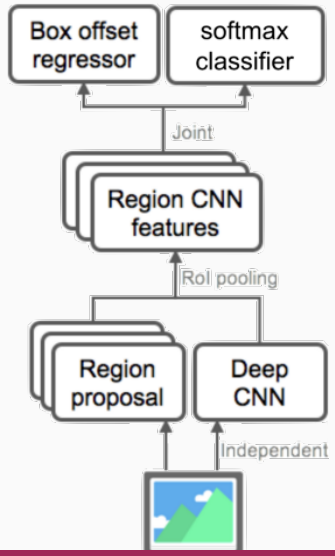
Algo	Features	Pred time/img	Limitations
R-CNN	• Selective search to generate regions, • Extract $2k$ regions per image.	40 s	• Each region passed to the CNN separately $\rightsquigarrow$ High comput. time, • Three different models to predict.
Fast R-CNN	• Each image forwarded once to the CNN, Feature maps extracted, • Selective search used on these maps to generate predictions, • Combine the 3 models used in RCNN together.	2 s	• Selective search slow $\rightsquigarrow$ Computation time still high.
Faster R-CNN	• Replaces selective search method with region proposal network.	0.2 s	• Several systems work one after the other: Performance of the following ones depends on the previous ones.



Summary of Region-Based Convolutional Neural Networks

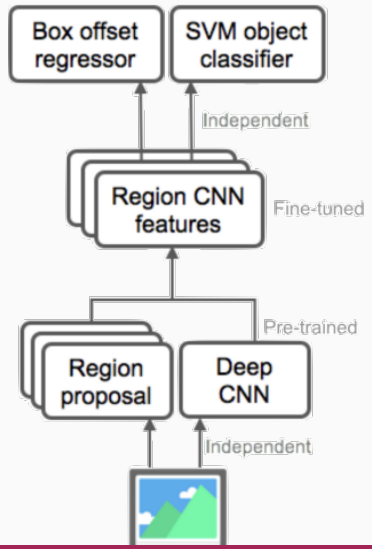
Algo	Features	Pred time/img	Limitations
R-CNN	• Selective search to generate regions, • Extract $2k$ regions per image.	40 s	• Each region passed to the CNN separately $\rightsquigarrow$ High comput. time, • Three different models to predict.
Fast R-CNN	• Each image forwarded once to the CNN, Feature maps extracted, • Selective search used on these maps to generate predictions, • Combine the 3 models used in RCNN together.	2 s	• Selective search slow $\rightsquigarrow$ Computation time still high.
Faster R-CNN	• Replaces selective search method with region proposal network.	0.2 s	• Several systems work one after the other: Performance of the following ones depends on the previous ones.

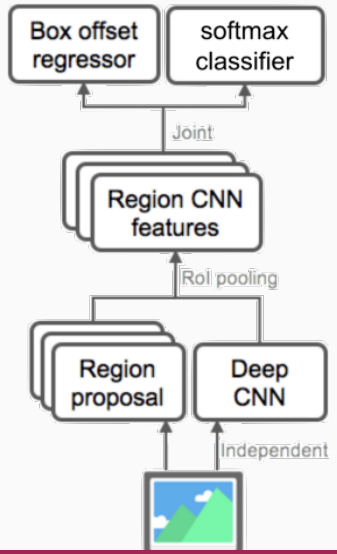


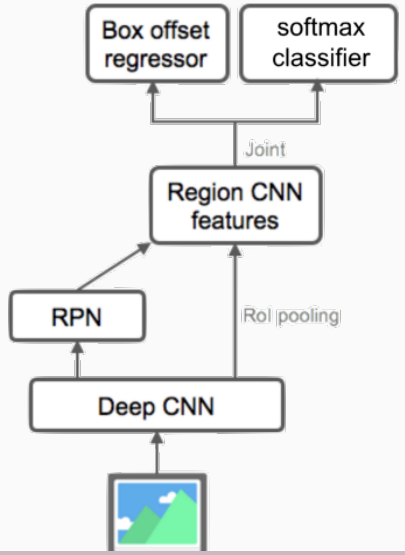


Summary of Region-Based Convolutional Neural Networks

Algo	Features	Pred time/img	Limitations
R-CNN	• Selective search to generate regions, • Extract $2k$ regions per image.	40 s	• Each region passed to the CNN separately $\rightsquigarrow$ High comput. time, • Three different models to predict.
Fast R-CNN	• Each image forwarded once to the CNN, Feature maps extracted, • Selective search used on these maps to generate predictions, • Combine the 3 models used in RCNN together.	2 s	• Selective search slow $\rightsquigarrow$ Computation time still high.
Faster R-CNN	• Replaces selective search method with region proposal network.	0.2 s	• Several systems work one after the other: Performance of the following ones depends on the previous ones.







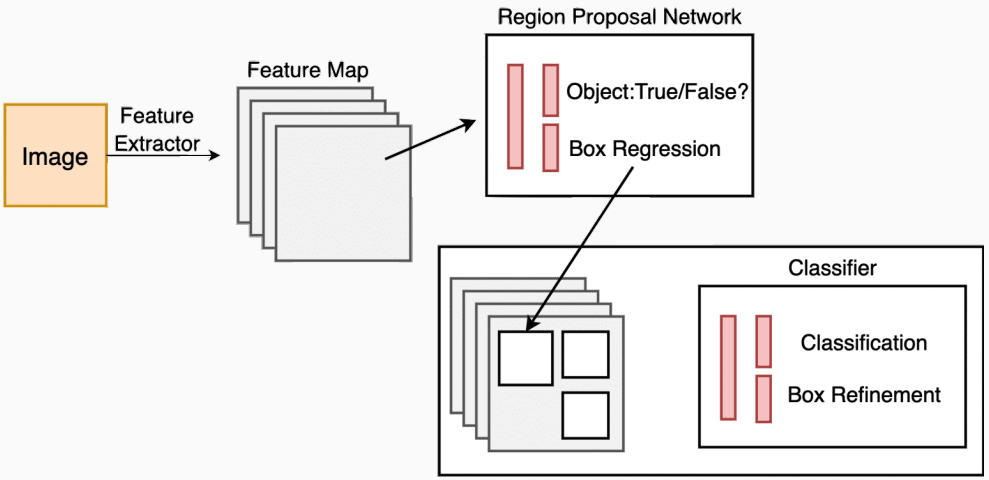
The YOLO (You Only Look Once) Framework

One vs. Two-Stage Object Detection

Two-stage object detection:

1. Detecting possible object regions,
2. Classifying the image in those regions into object classes

- Better accuracy,
- *But:*
  - Multiple iterations takes place,
  - Slows down detection speed
  - Prevents real-time detection.



Two-stage algorithm:

CNN, R-CNN, Fast R-CNN, Faster R-CNN Mask R-CNN, . . .

One-stage algorithm:

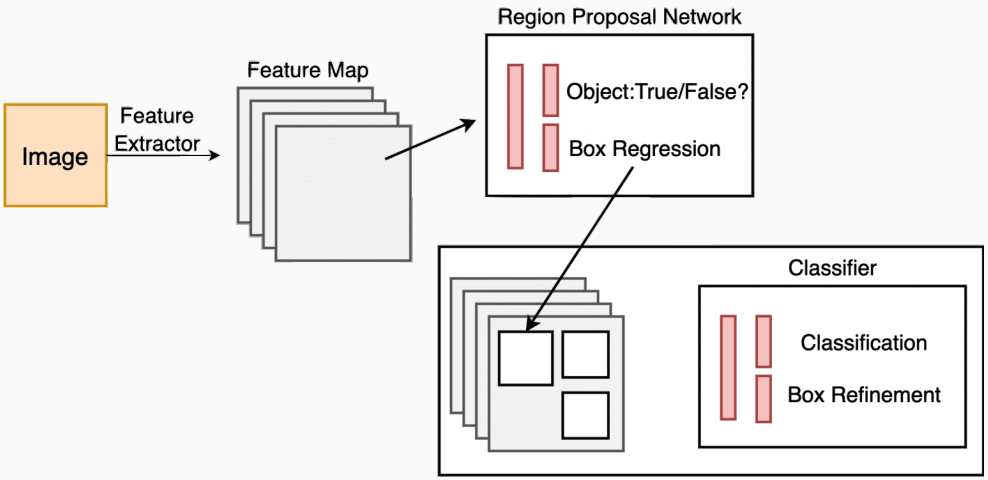
Yolo, SSD, . . .

One vs. Two-Stage Object Detection

Two-stage object detection:

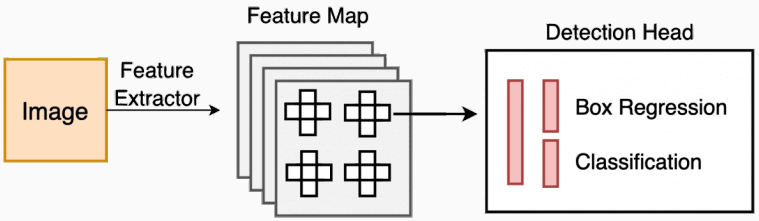
1. Detecting possible object regions,
2. Classifying the image in those regions into object classes

- Better accuracy,
- *But:*
  - Multiple iterations takes place,
  - Slows down detection speed
  - Prevents real-time detection.



Two-stage algorithm:

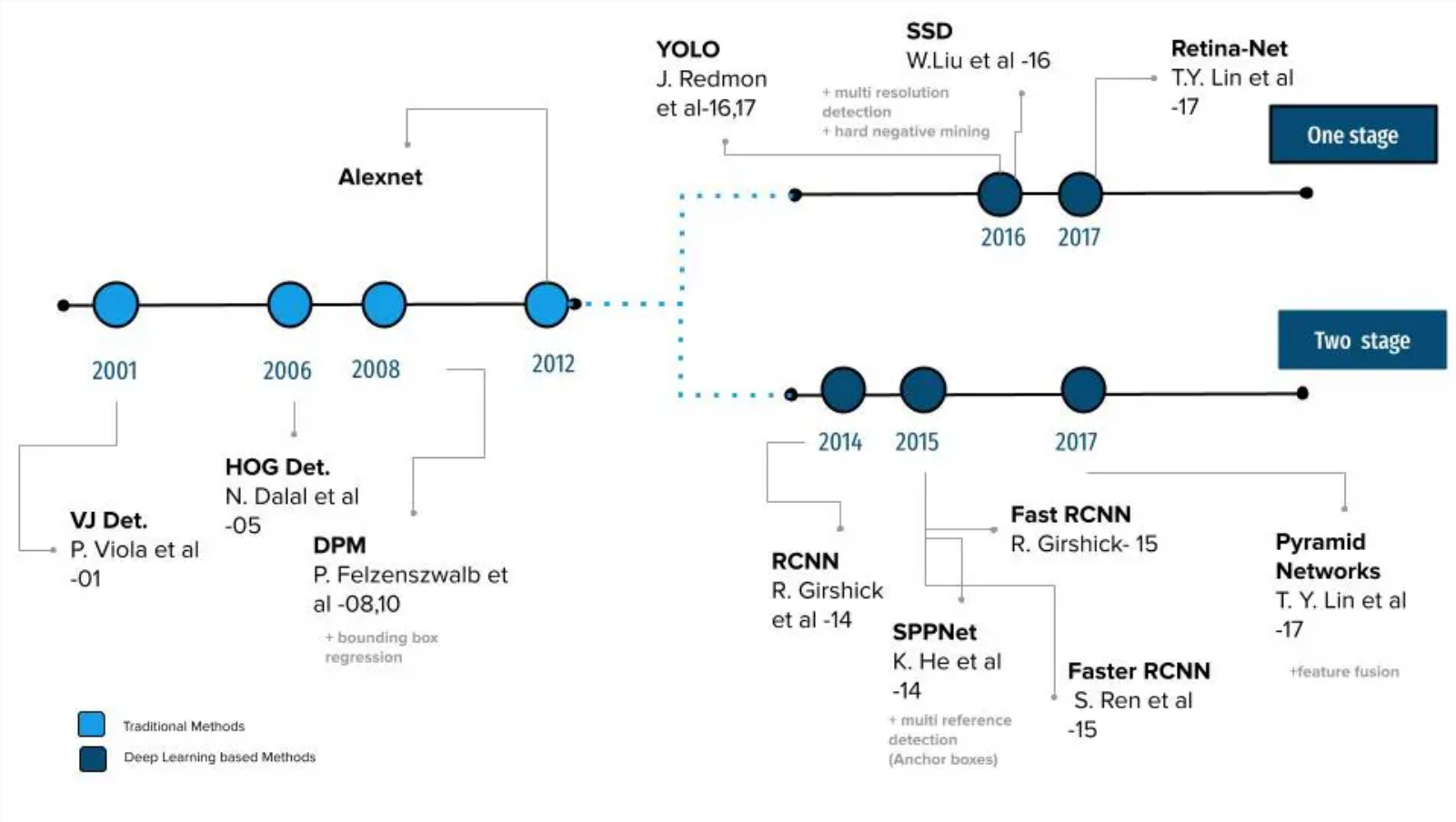
CNN, R-CNN, Fast R-CNN, Faster R-CNN Mask R-CNN, . . .



One-stage algorithm:

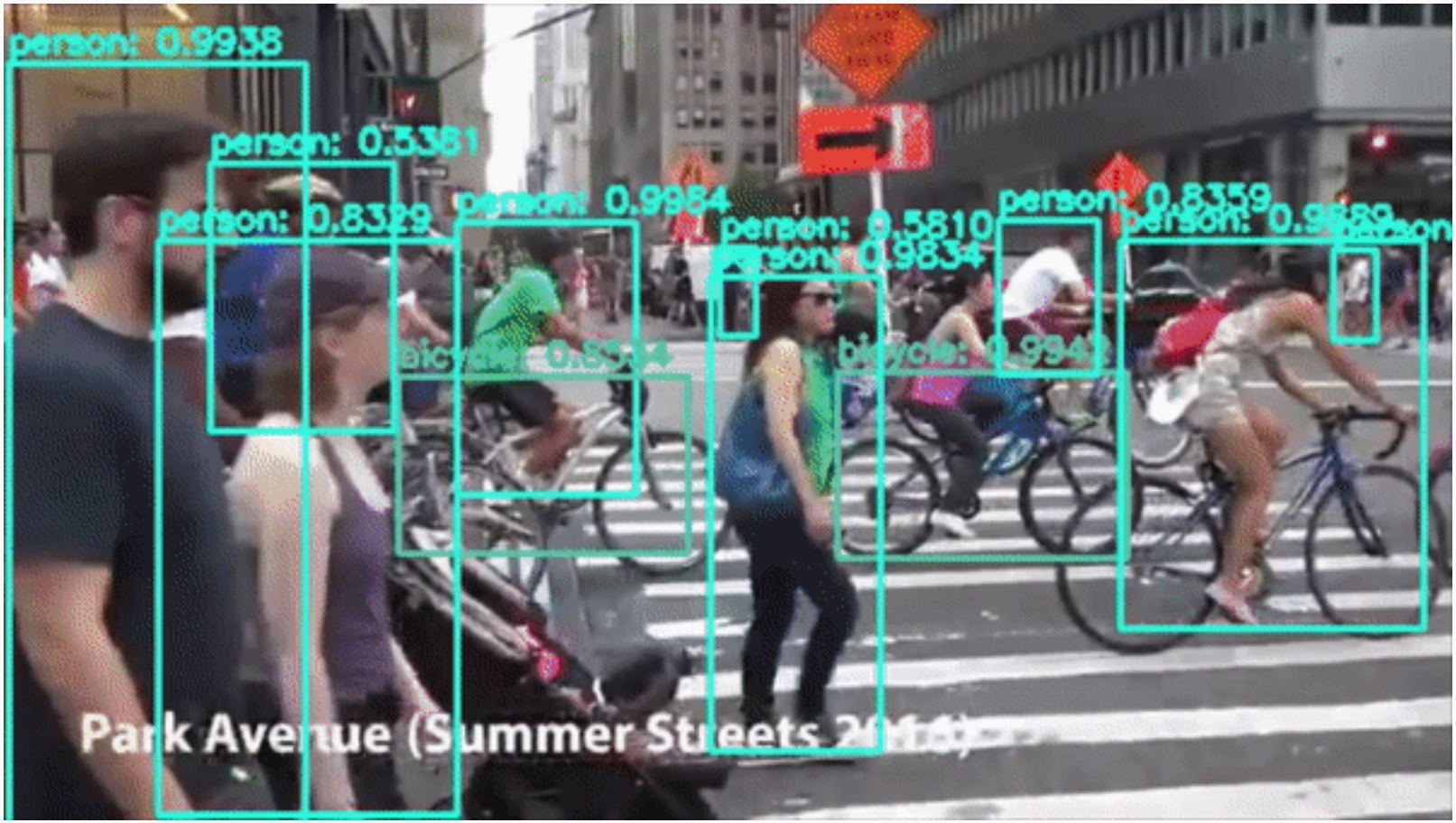
Yolo, SSD, . . .

One vs. Two-Stage Object Detection



[Zou et al.] Object Detection in 20 Years: A Survey

Real-Time Detection



See `https://youtu.be/MPU2HistivI`

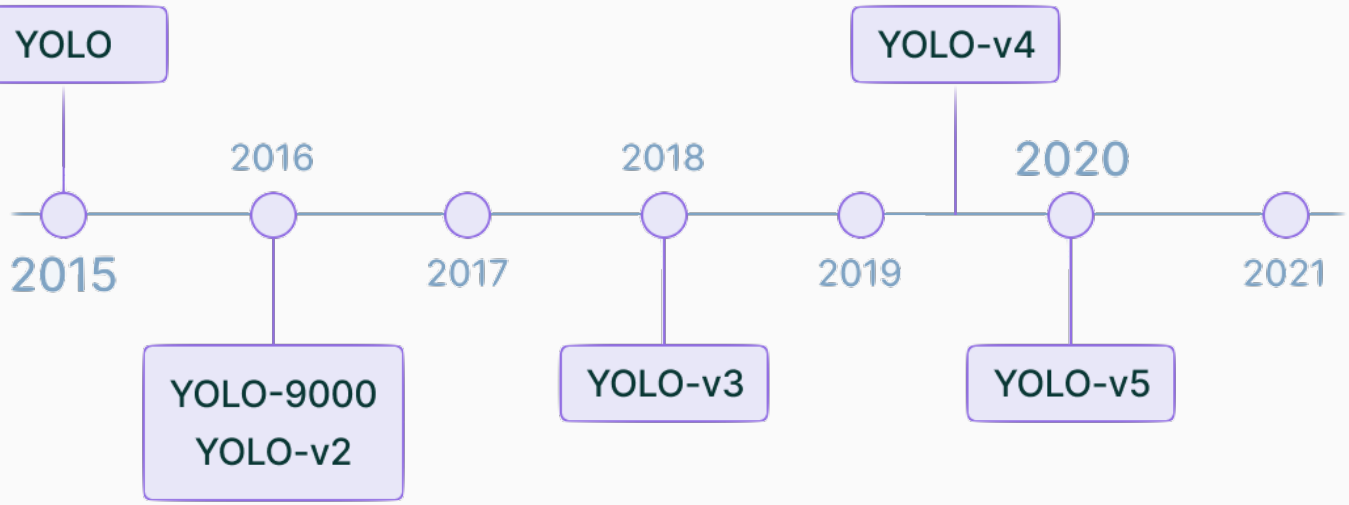
	Faster R-CNN		Yolo	
Real-time detection:	7 FPS	140 ms/img	45 FPS	22 ms/img

The YOLO (You Only Look Once) Framework

YOLO Framework (Redmon et al, 2015)

YOLO $\equiv$ You Only Look Once

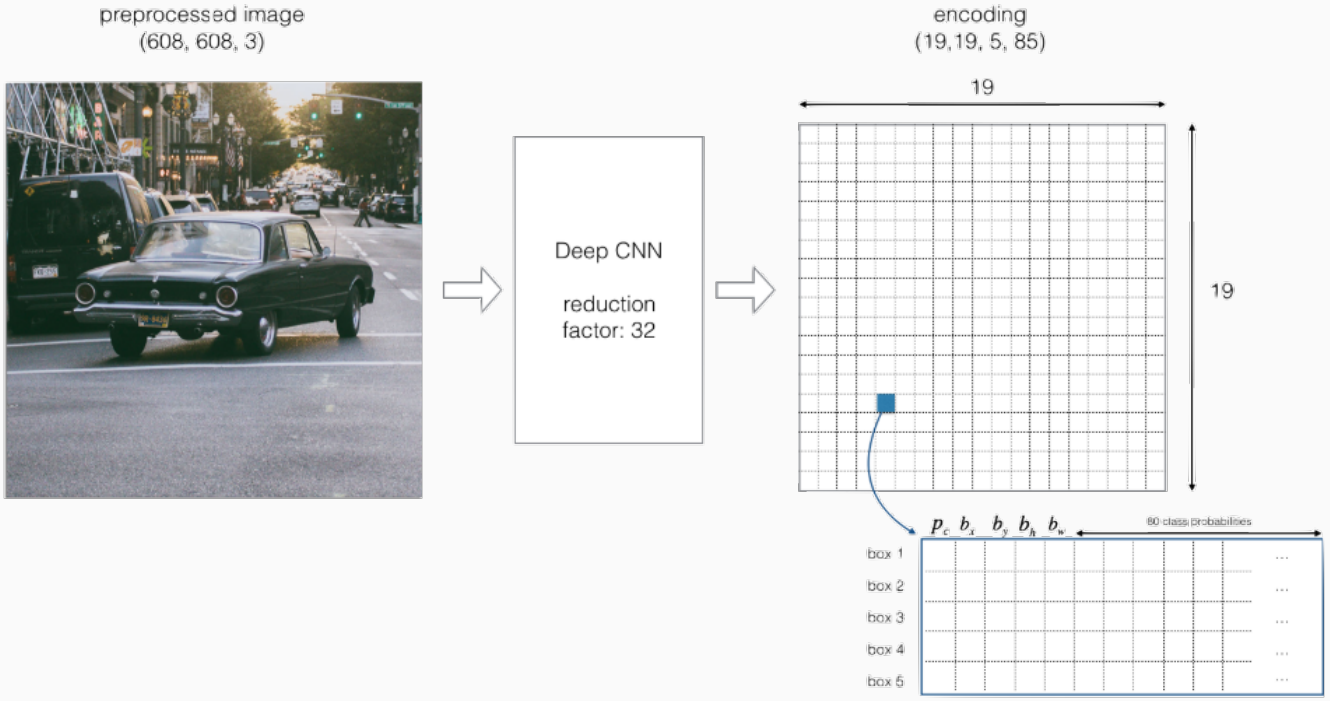
- Algorithm based on regression instead of selecting RoI in images,
- Predicts classes and bounding boxes for the whole image in one run.



[Redmon et al.] You Only Look Once : Unified, Real-Time Object Detection

YOLO Algorithm

1. Let k fixed-ratio bounding boxes,
2. Divides the input image into grids,
3. Image classification and localization are applied on each grid: 5 descriptors:
   - p_c: Proba there is an object in the bounding box
   - (b_x, b_y): Center of the box
   - b_h: Height ratio of the anchor box vs. corresponding grid cell
   - b_w: Width ration. . .
   - Class of the (possible) object c



YOLO Algorithm

Remarks:

- Object considered to lie in a specific cell only if the center co-ordinates of the anchor box lie in that cell
- Probability there is an object of a certain class c_i: $\text{score}_i = p_c \times c_i$
  $\rightsquigarrow$ Class with the max proba assigned to that cell.
- Similar process happens for all the grid cells.

S x S grid on input

YOLO Algorithm

Remarks:

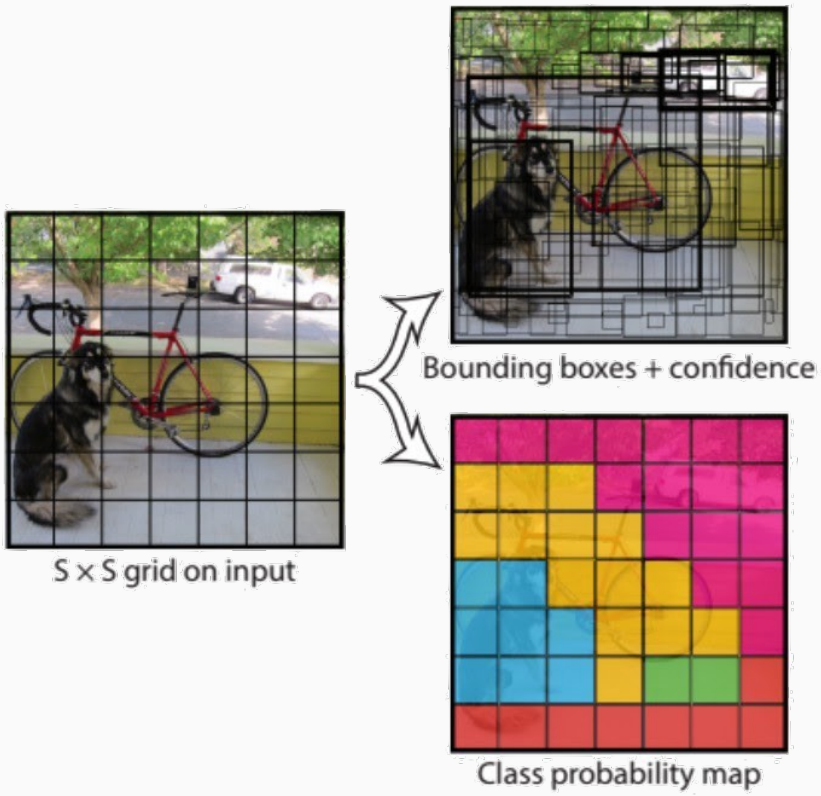
- Object considered to lie in a specific cell only if the center co-ordinates of the anchor box lie in that cell
- Probability there is an object of a certain class c_i: $\boxed{\text{score}_i = p_c \times c_i}$
  $\rightsquigarrow$ Class with the max proba assigned to that cell.
- Similar process happens for all the grid cells.

S x S grid on input

YOLO Algorithm

Remarks:

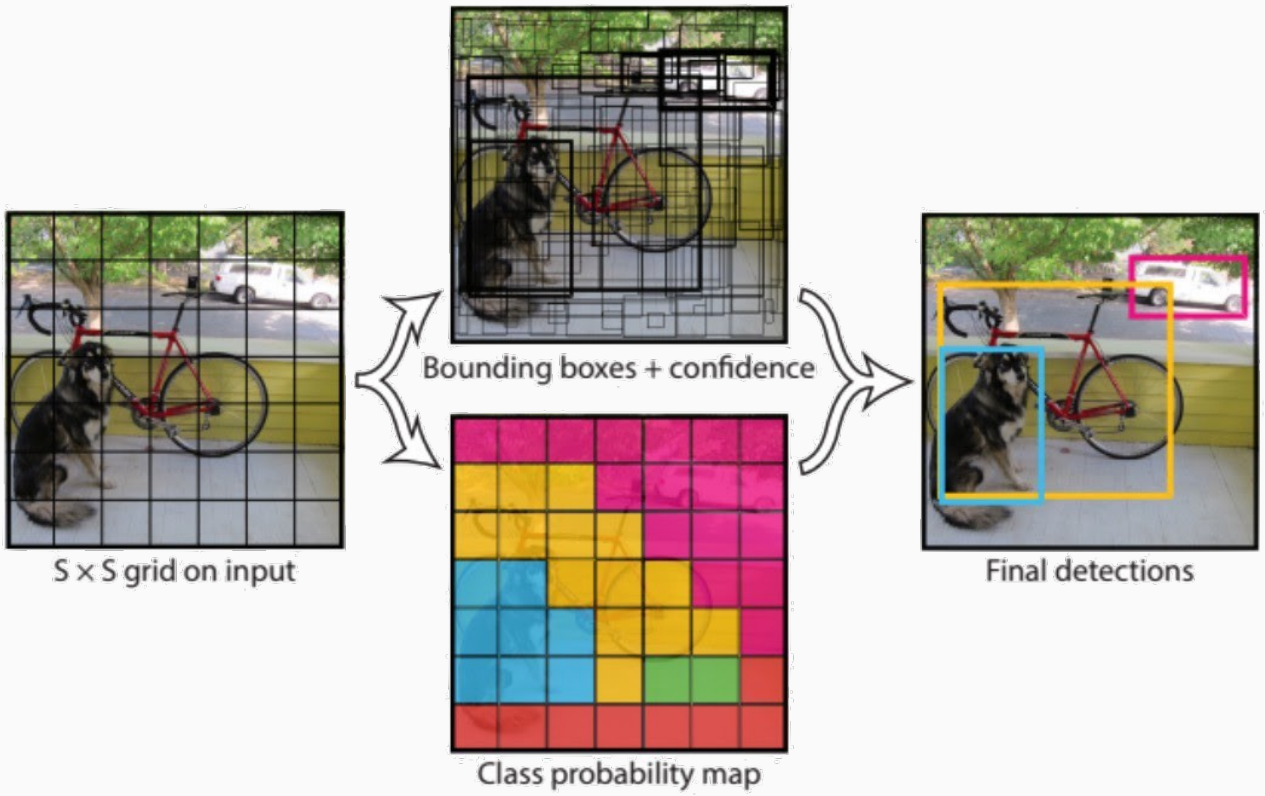
- Object considered to lie in a specific cell only if the center co-ordinates of the anchor box lie in that cell
- Probability there is an object of a certain class c_i: $\boxed{\text{score}_i = p_c \times c_i}$
  - $\rightsquigarrow$ Class with the max proba assigned to that cell.
- Similar process happens for all the grid cells.



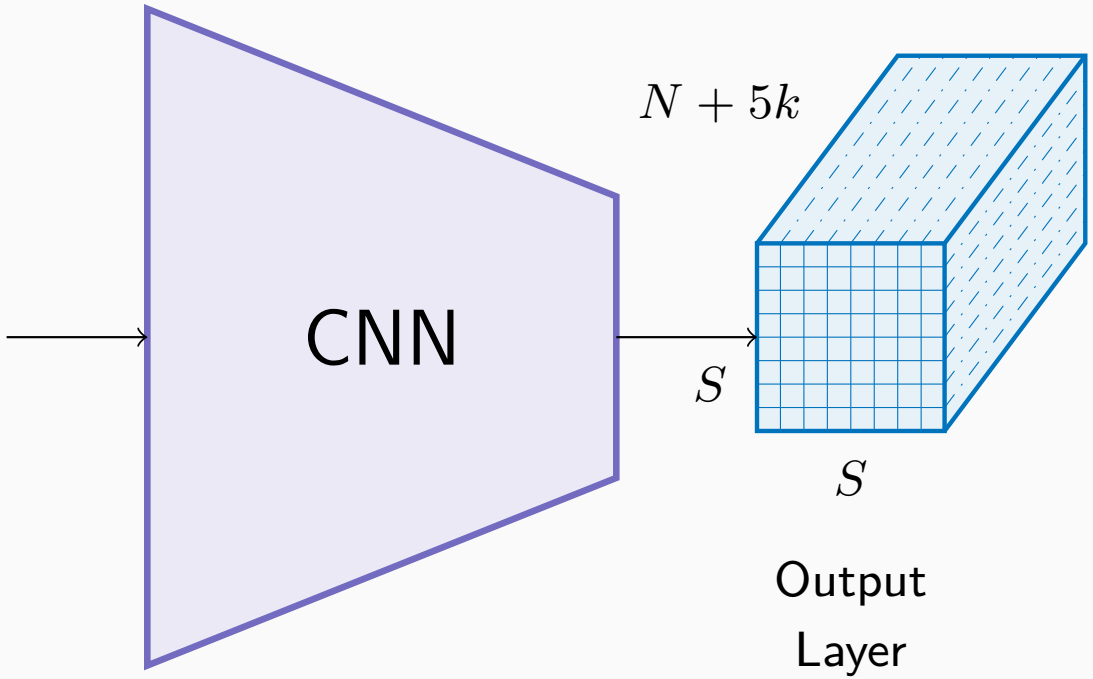
YOLO Algorithm

Remarks:

- Object considered to lie in a specific cell only if the center co-ordinates of the anchor box lie in that cell
- Probability there is an object of a certain class c_i: $\boxed{\text{score}_i = p_c \times c_i}$
  $\rightsquigarrow$ Class with the max proba assigned to that cell.
- Similar process happens for all the grid cells.



YOLO: Network Structure



where :

- S is the grid size (on PASCAL, $S = 7$)
- k is the number of objects detected per cell (PASCAL: $k = 2$)
- N is the number of classes (PASCAL: $N = 20$)

[Redmon et al.] You Only Look Once: Unified, Real-Time Object Detection

[PASCAL] The PASCAL Visual Object Classes: http://host.robots.ox.ac.uk/pascal/VOC/

From YOLO to YOLOv3

Numerous improvements achieved, such as:

- Use of bounding box "templates" $\rightsquigarrow$ **anchor boxes**,
- Multi-scale training and prediction,
- A more advanced CNN at the core,
- Update of the way bounding boxes are encoded,

and many others!

For some, these modifications are taken from other object detection algorithms (Faster-RCNN, SSD, *etc.*)

[Redmon et al.] YOLO9000: Better, Faster, Stronger

[Redmon et al.] YOLOv3: An Incremental Improvement

Image segmentation

5.1 Segmentation Challenges

5.2 First Examples of Segmentation Networks

Segmentation Challenges

Segmentation: Delineation of different semantic areas of the image

$\longleftrightarrow$ Associate a class to *each* pixel of the image

A lot of predictions!

MS COCO: 80 classes, img>200x200

$\rightsquigarrow$ several million simultaneous predictions

Pixels du ciel | Image originale | Pixels de la mer

Prediction for a given pixel requires

- analysis of local information,
- but also a more global understanding of the scene.

Segmentation Challenges

Segmentation: Delineation of different semantic areas of the image

$\longleftrightarrow$ Associate a class to *each* pixel of the image

A lot of predictions!

MS COCO: 80 classes, img>200x200

$\rightsquigarrow$ several million simultaneous predictions

Prediction for a given pixel requires

- analysis of local information,
- but also a more global understanding of the scene.

Segmentation Challenges

Segmentation: Delineation of different semantic areas of the image

$\longleftrightarrow$ Associate a class to *each* pixel of the image

A lot of predictions!

MS COCO: 80 classes, img>200x200

$\rightsquigarrow$ several million simultaneous predictions

Pixels du ciel | Image originale | Pixels de la mer

Prediction for a given pixel requires

- analysis of local information,
- but also a more global understanding of the scene.

Construction of Learning Database. Example of MS COCO

- More than 300K images, 80 classes,
- More than 2.5M instances of segmented objects
- More than 22 man-hours for 1000 segmentations

  $\rightsquigarrow 55k$ hours $\simeq 3k$ days $\simeq 18k$ working days!

[Lin et al.] Microsoft COCO: Common Objects in Context.

Image segmentation

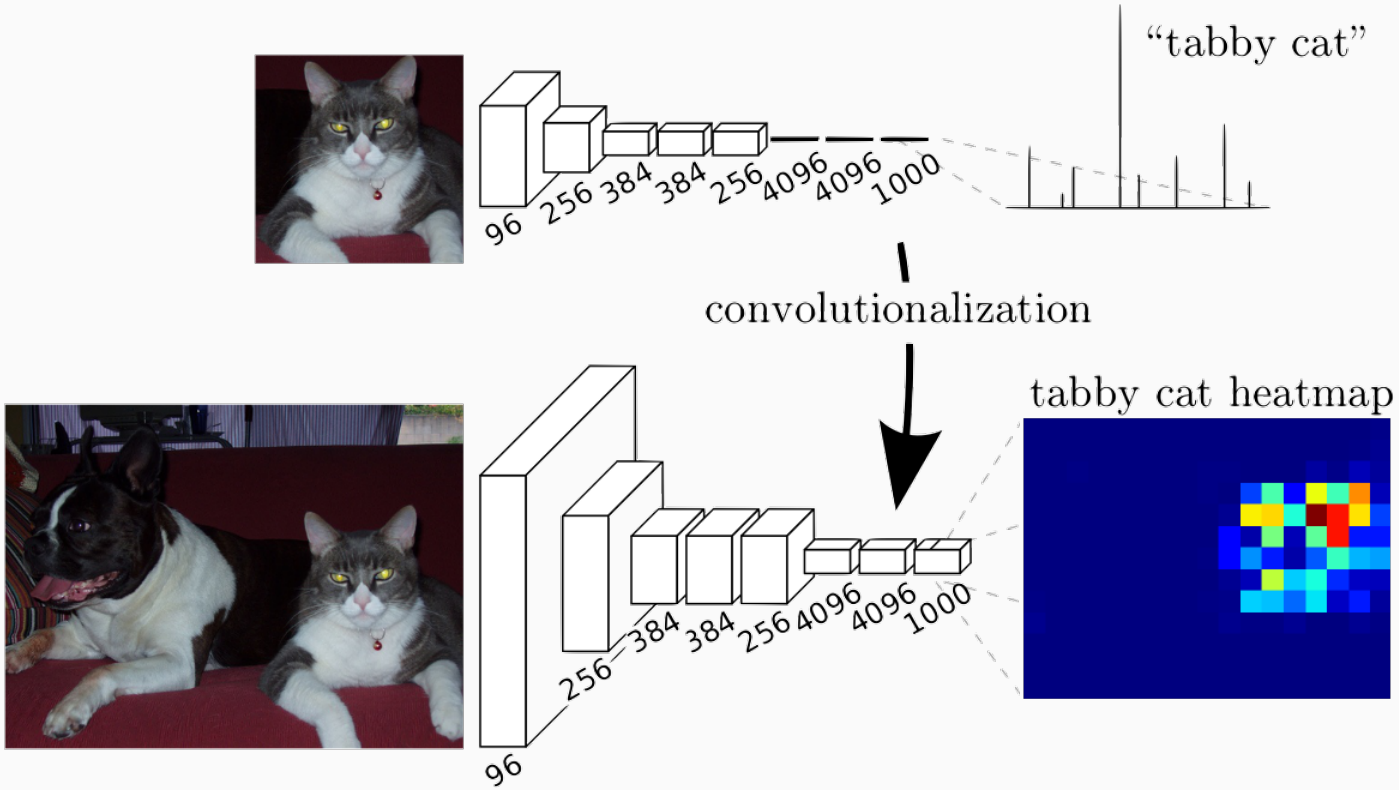
Naive Approach: Sliding Window Segmentation

Disadvantages:

- Very expensive,
- No consideration of the overall image.

Nevertheless, **FCM algorithm**

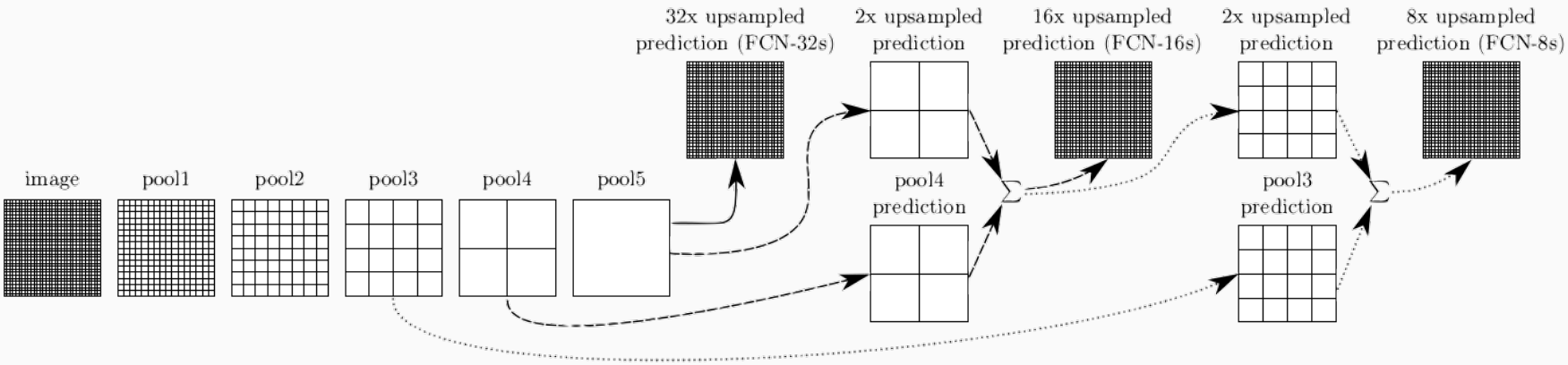
Fully Convolutional Networks (Long *et al.*, 2014)

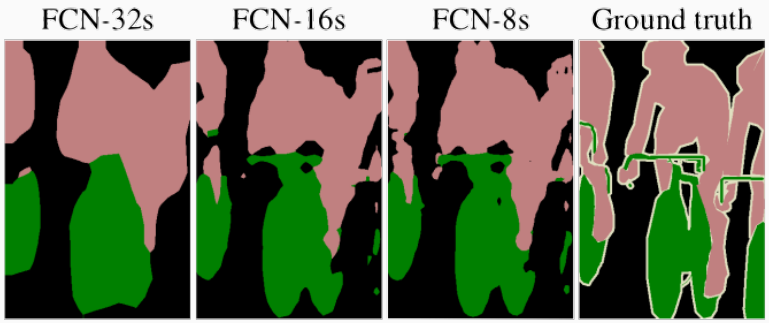


- First algorithm to implement a segmentation *end-to-end*,
- Based on a fully convolutional reformulation of the networks,
- Base VGG-16.

[Long et al.] Fully Convolutional Networks for Semantic Segmentation.

Fully Convolutional Networks (Long *et al.*, 2014)





Result refined by taking into account intermediate predictions!

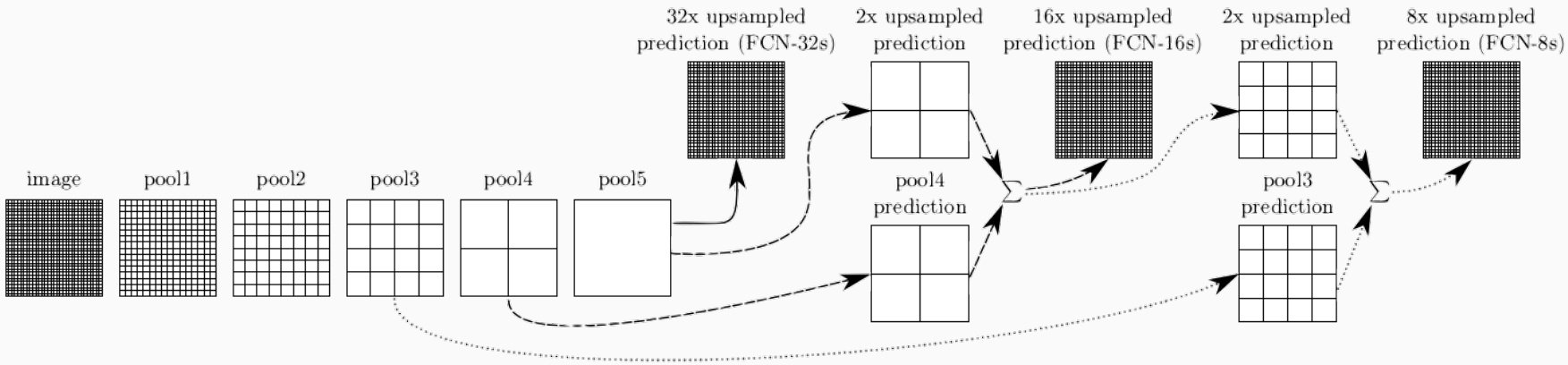
$\rightsquigarrow$ Merge (sum term to term) of several depth levels

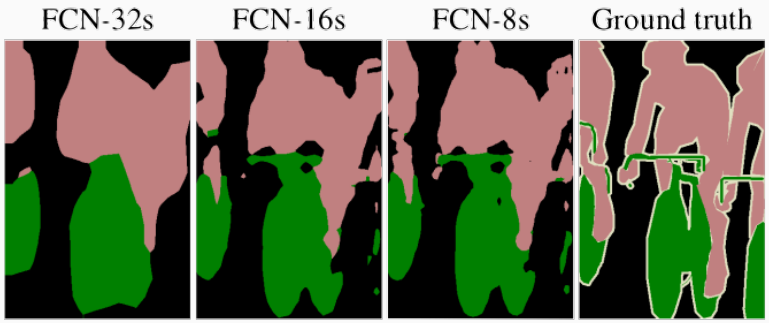
Cost functions:

- Each pixel carries a function `softmax` to determine which class is most likely.
- Cross-entropy to compare the predicted class of a pixel and the true one
- The Final cost function is the average of the cross entropies over all the pixels in the image.

[Long et al.] Fully Convolutional Networks for Semantic Segmentation.

Fully Convolutional Networks (Long *et al.*, 2014)





Result refined by taking into account intermediate predictions!

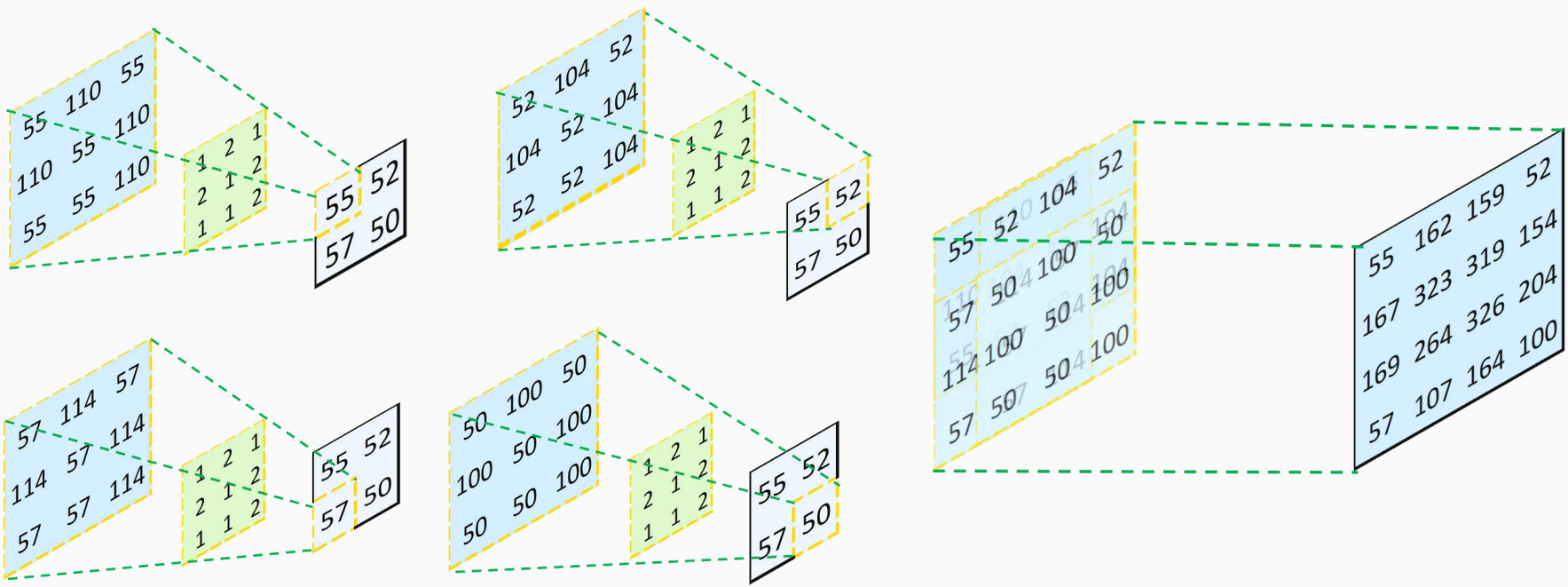
$\rightsquigarrow$ Merge (sum term to term) of several depth levels

Cost functions:

- Each pixel carries a function `softmax` to determine which class is most likely.
- Cross-entropy to compare the predicted class of a pixel and the true one
- The Final cost function is the average of the cross entropies over all the pixels in the image.

[Long et al.] Fully Convolutional Networks for Semantic Segmentation.

Deconvolution, or Transposed Convolution

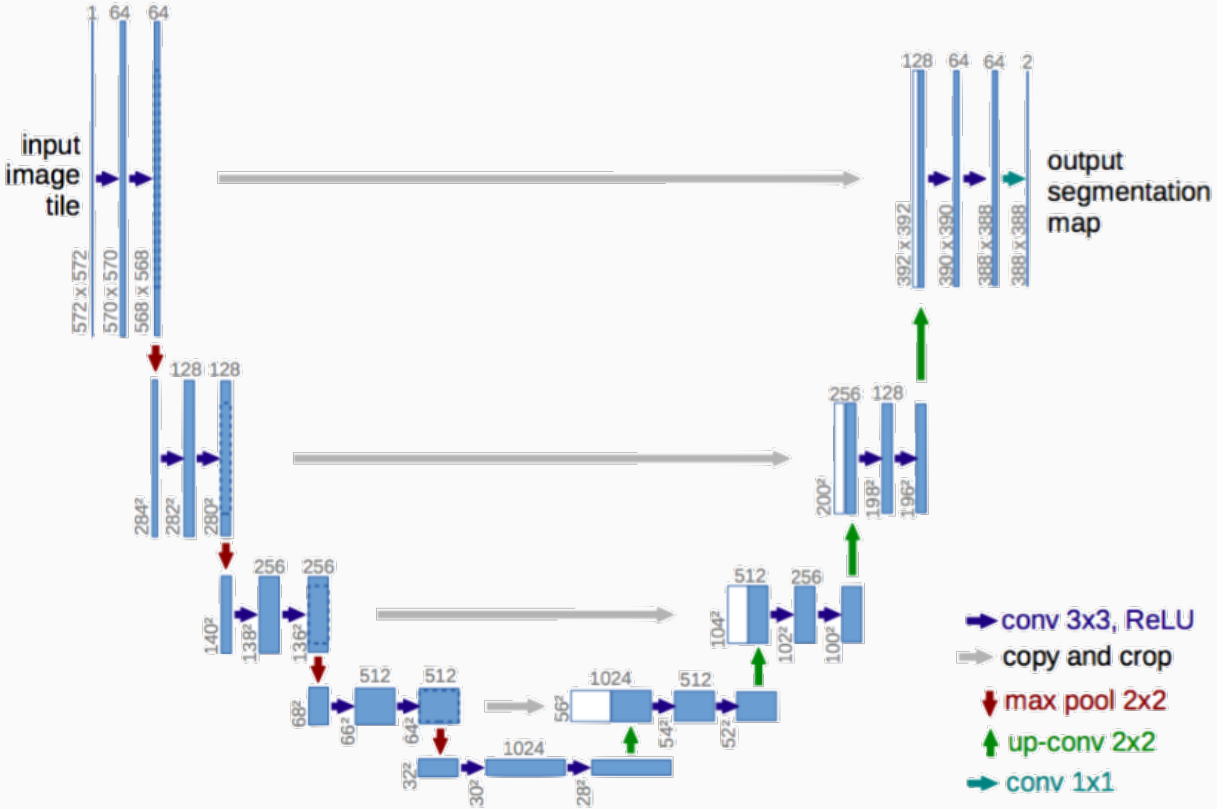


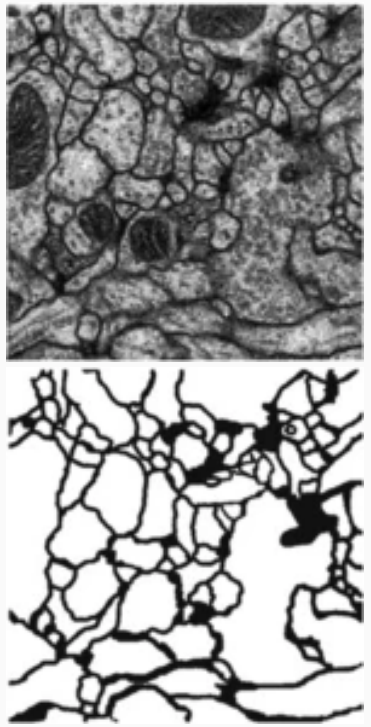
Here: Transposed 2D convolution with no *padding*, no *stride*, and *kernel* of 3.

`https://towardsdatascience.com/`

`understand-transposed-convolutions-and-build-your-own-transposed-convolution-layer-from-scratch-4f5d97b2967`

UNet (Ronneberger *et al.*, 2015)



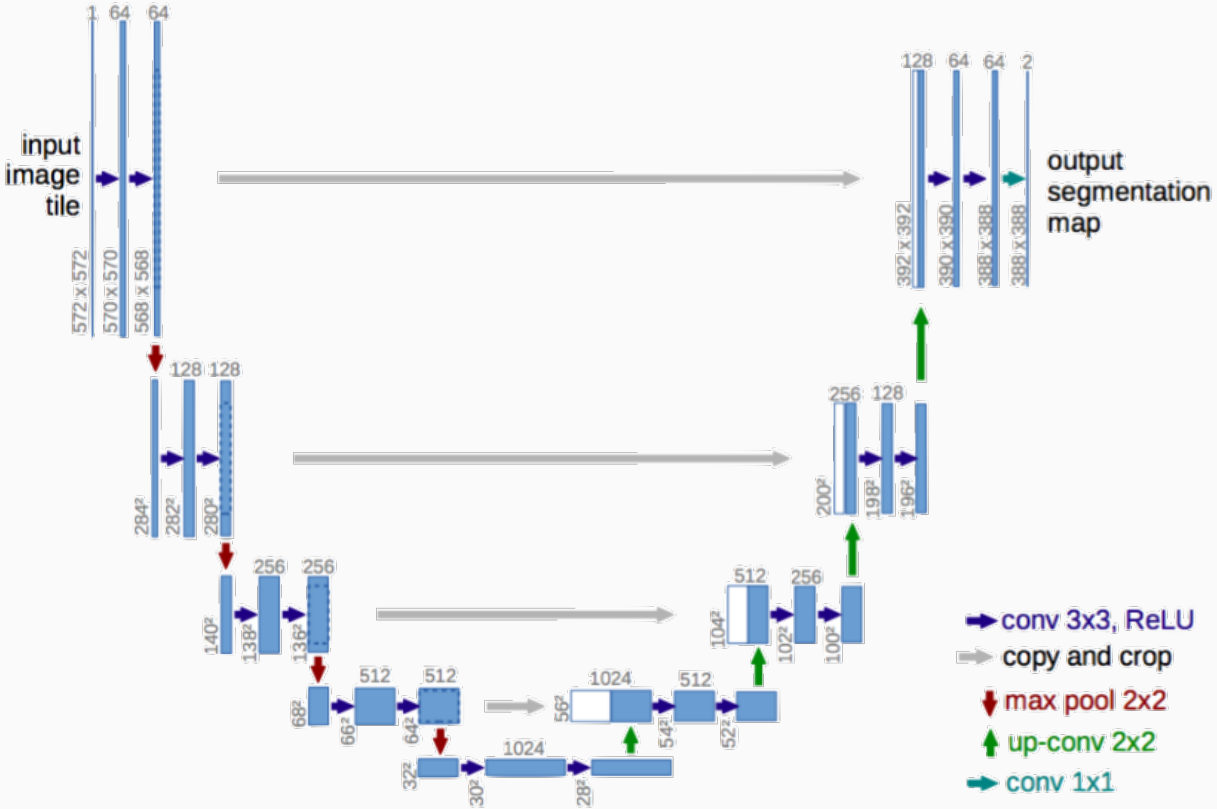


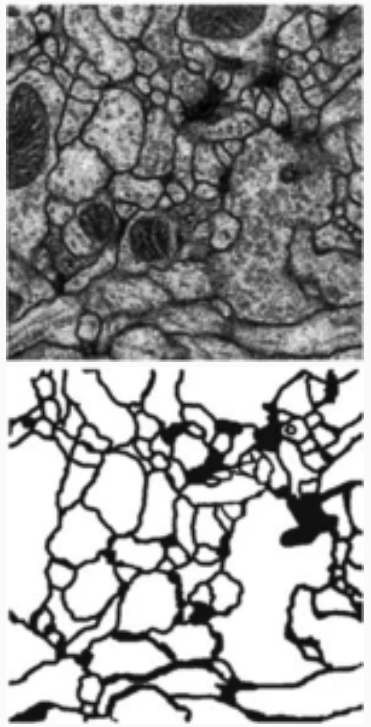
- FCN improvement,
- Probably one of the most used networks for segmentation.

+ *Adaptable architecture with any convolutional base!*

[Ronneberger et al.] U-Net: Convolutional Networks for Biomedical Image Segmentation.

UNet (Ronneberger *et al.*, 2015)





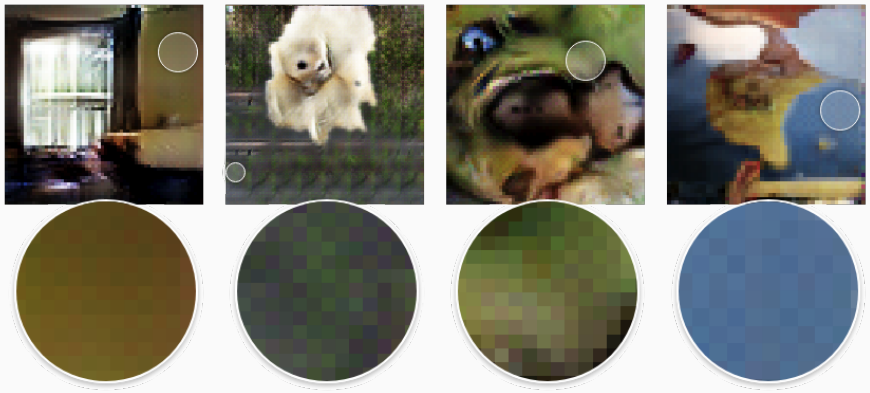
- FCN improvement,
- Probably one of the most used networks for segmentation.

\+ *Adaptable architecture with any convolutional base!*

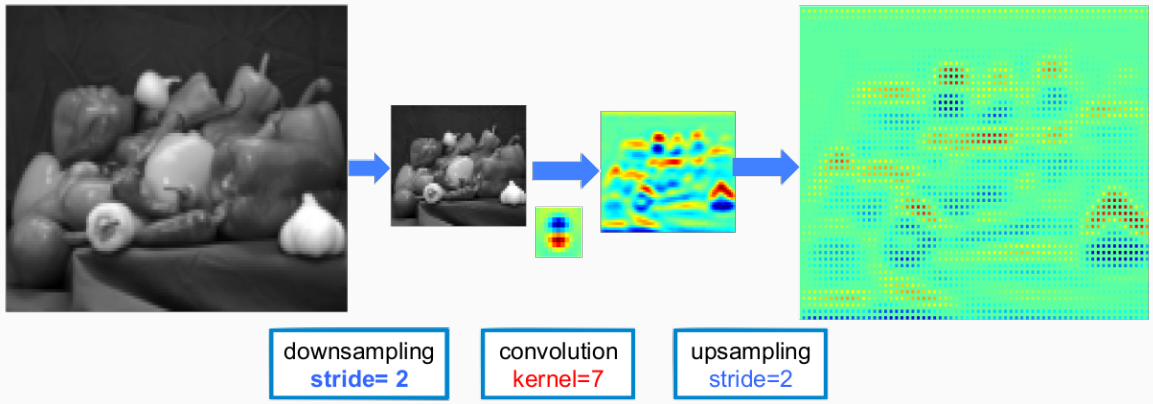
[Ronneberger et al.] U-Net: Convolutional Networks for Biomedical Image Segmentation.

Checkerboard Effect and "Atrous" Convolution

Checkerboard effect:



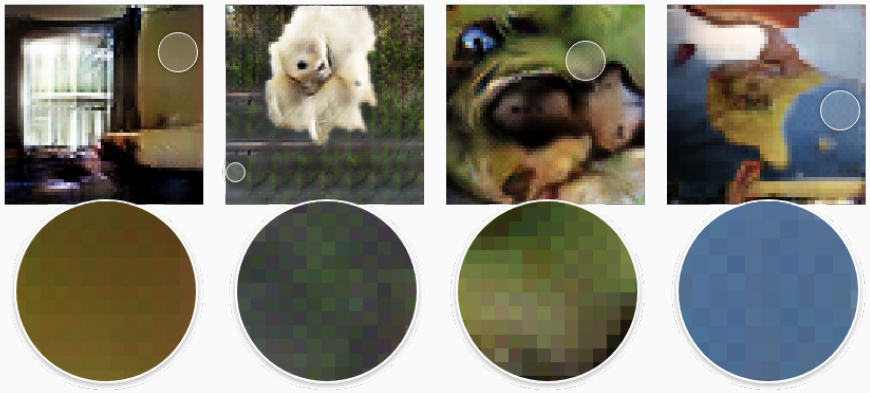
"Atrous" or dilated convolution:

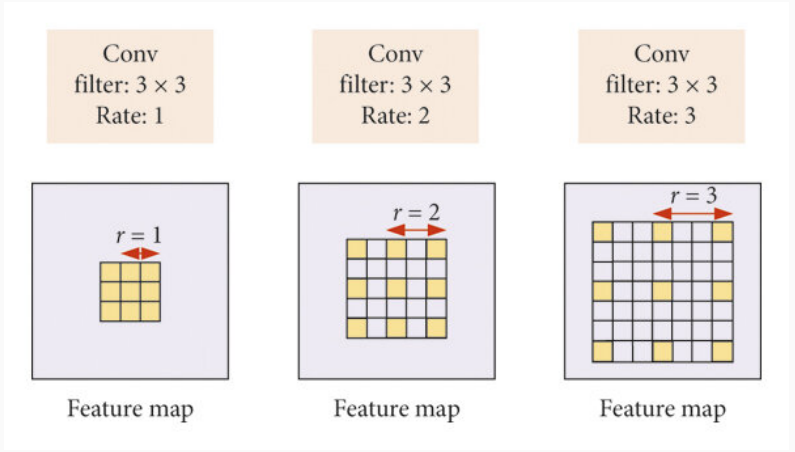


[Odena et al.] Deconvolution and Checkerboard Artifacts.

Checkerboard Effect and "Atrous" Convolution

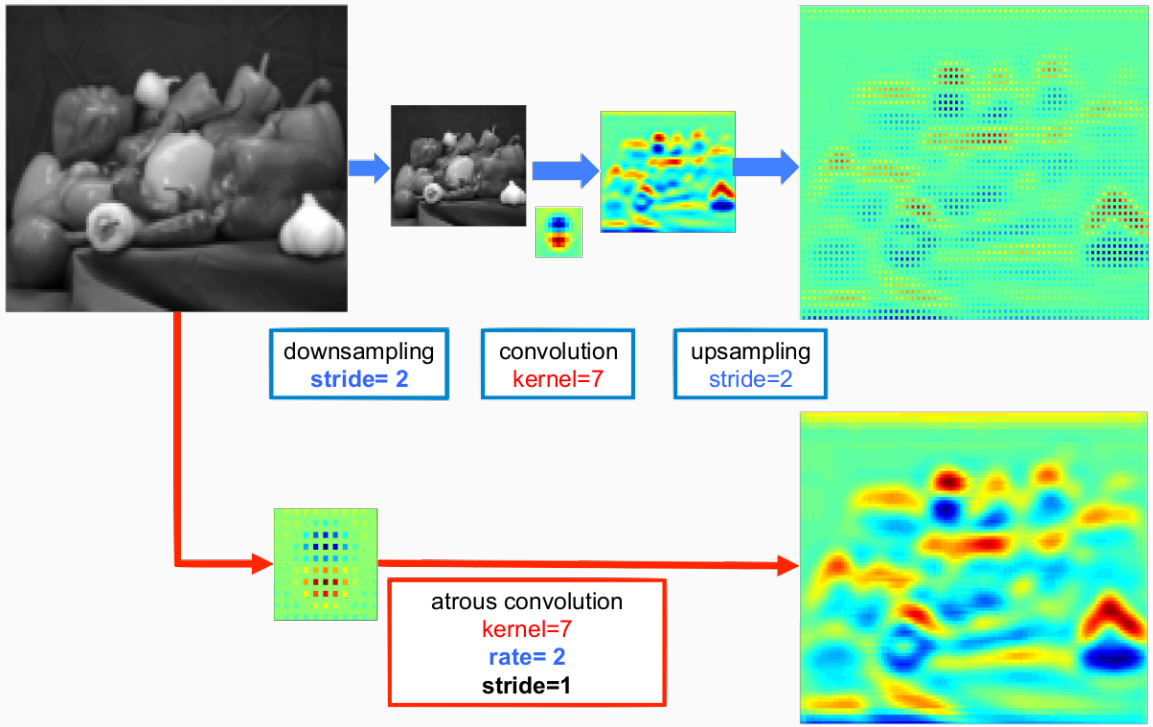
Checkerboard effect:





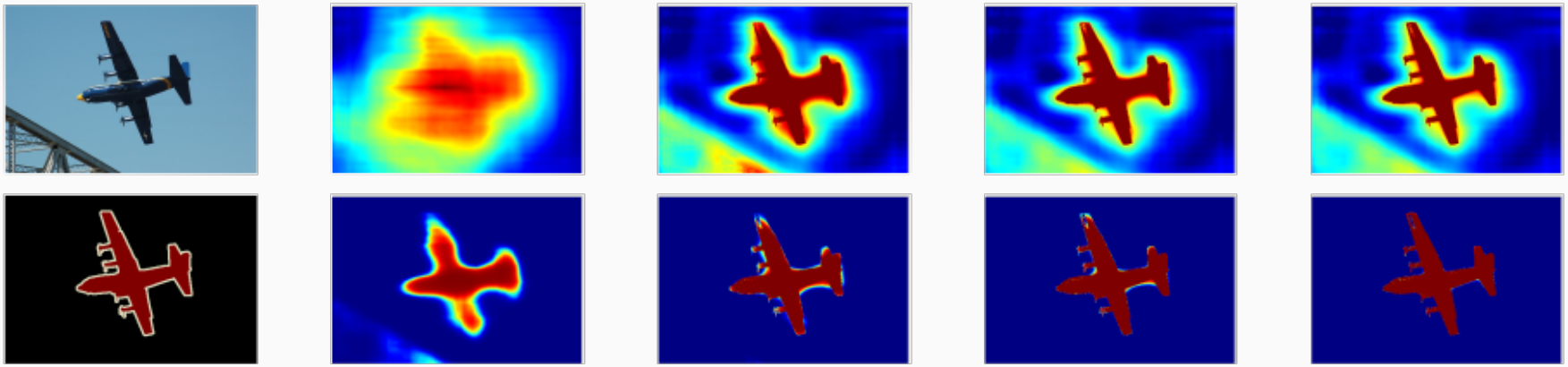
"Atrous" or dilated convolution:

[Odena et al.] Deconvolution and Checkerboard Artifacts.



Conditional Random Fields

And other techniques, such as the post-processing used by DeepLab



[Chen et al.] DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs.

Take-Home Message on Segmentation

Two main issues:

- **Density of the output layer**: increase of the dimension by deconvolution or "atrous" convolution,
- **Multi-scale** : U-shape (*auto-encoder*), spatial pyramid.

Challenges:

References i

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017a). **Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs.** IEEE transactions on pattern analysis and machine intelligence, 40(4):834–848.

Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. (2017b). **Rethinking atrous convolution for semantic image segmentation.** arXiv preprint arXiv:1706.05587.

Girshick, R. (2015). **Fast R-CNN.** In Proceedings of the IEEE international conference on computer vision, pages 1440–1448.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). **Rich feature hierarchies for accurate object detection and semantic segmentation.** In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 580–587.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). **Microsoft coco: Common objects in context.** In European conference on computer vision, pages 740–755. Springer.

Long, J., Shelhamer, E., and Darrell, T. (2015). **Fully convolutional networks for semantic segmentation.** In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3431–3440.

Odena, A., Dumoulin, V., and Olah, C. (2016). **Deconvolution and checkerboard artifacts.** Distill, 1(10):e3.

References ii

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). **You only look once: Unified, real-time object detection.** In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 779–788.

Redmon, J. and Farhadi, A. (2017). **Yolo9000: Better, faster, stronger.** In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7263–7271.

Redmon, J. and Farhadi, A. (2018). **Yolov3: An incremental improvement.** arXiv preprint arXiv:1804.02767.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). **Faster R-CNN: Towards real-time object detection with region proposal networks.** Advances in neural information processing systems, 28:91–99.

Ronneberger, O., Fischer, P., and Brox, T. (2015). **U-net: Convolutional networks for biomedical image segmentation.** In International Conference on Medical image computing and computer-assisted intervention, pages 234–241. Springer.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2013). **Overfeat: Integrated recognition, localization and detection using convolutional networks.** arXiv preprint arXiv:1312.6229.

Zou, Z., Shi, Z., Guo, Y., and Ye, J. (2019). **Object detection in 20 years: A survey.** arXiv preprint arXiv:1905.05055.