CS 1674: Intro to Computer Vision

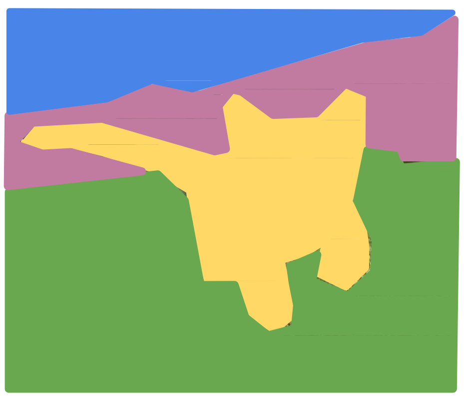
Object Recognition

Prof. Adriana Kovashka
University of Pittsburgh

April 5, 2022

Different Flavors of Object Recognition

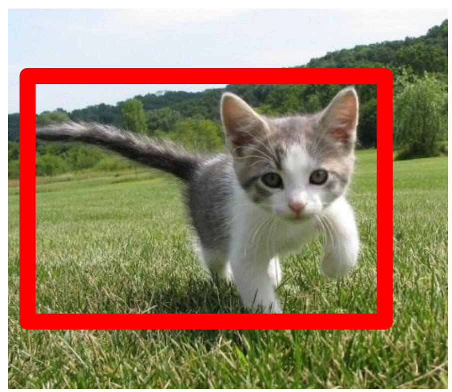
Semantic Segmentation



GRASS, CAT, TREE, SKY

No objects, just pixels

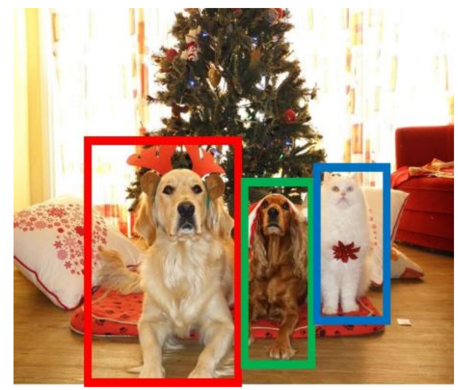
Classification + Localization



CAT

Single Object

Object Detection



DOG, DOG, CAT

Instance Segmentation

DOG, DOG, CAT

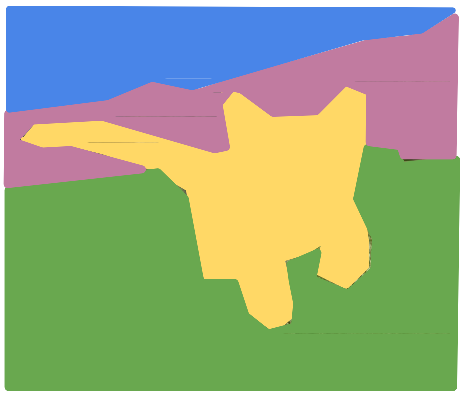
Multiple Object

Adapted from Justin Johnson

Plan for the next three lectures

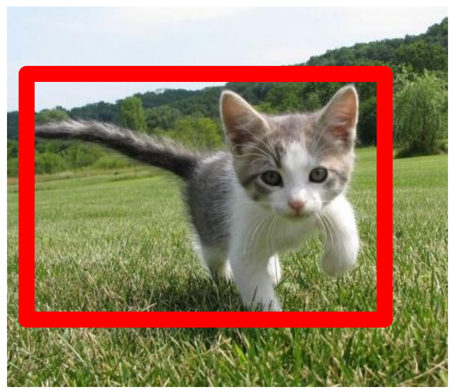
- Detection approaches
  - Pre-CNNs
    - Detection with whole windows: Pedestrian detection
    - Part-based detection: Deformable Part Models
  - Post-CNNs
    - Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    - Detection without region proposals: YOLO, SSD
- Segmentation approaches
  - Semantic segmentation: FCN
  - Instance segmentation: Mask R-CNN

Object Detection



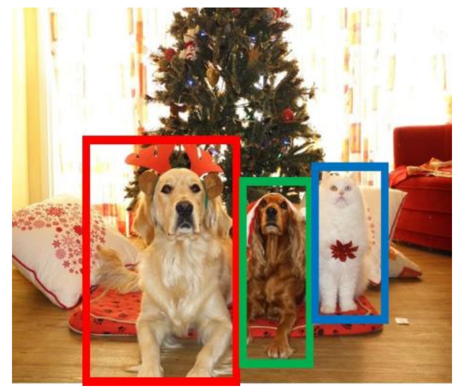
GRASS, CAT, TREE, SKY

No objects, just pixels



CAT

Single Object



DOG, DOG, CAT

DOG, DOG, CAT

Multiple Object

Slide by: Justin Johnson

Object detection: basic framework

- Build/train object model
- Generate candidate regions in new image
- Score the candidates

Adapted from Kristen Grauman

Window-template-based models
Building an object model

Given the representation, train a binary classifier

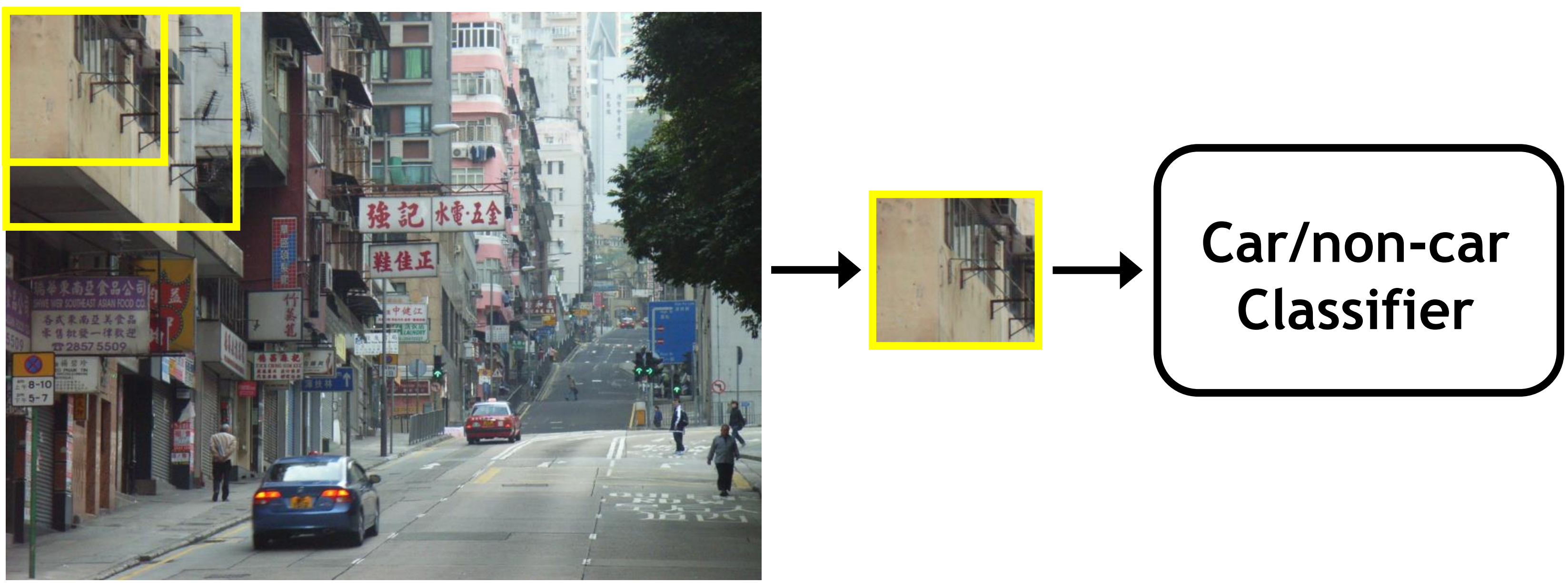
Car/non-car
Classifier

No, not a car.

Yes, car.

Kristen Grauman

Window-template-based models
Generating and scoring candidates



Kristen Grauman

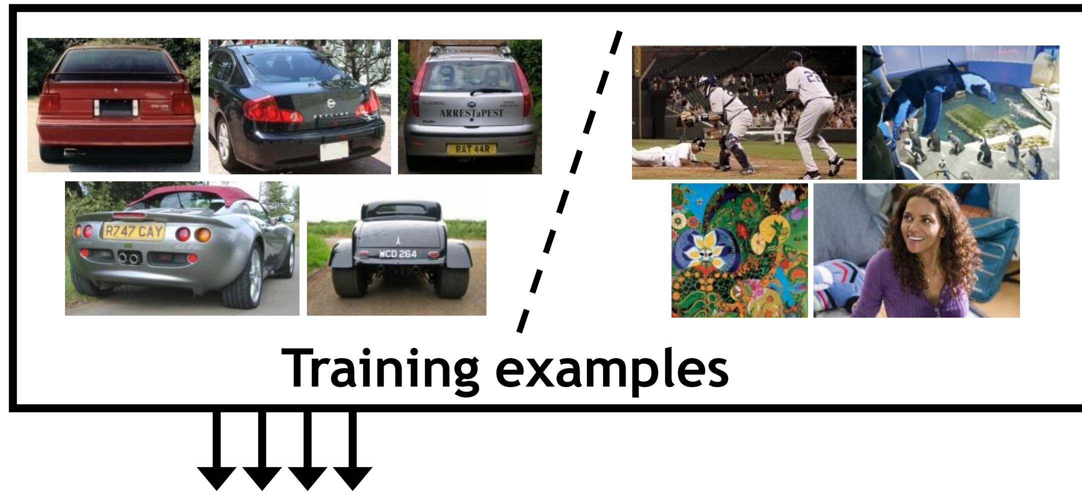
Window-template-based object detection: recap

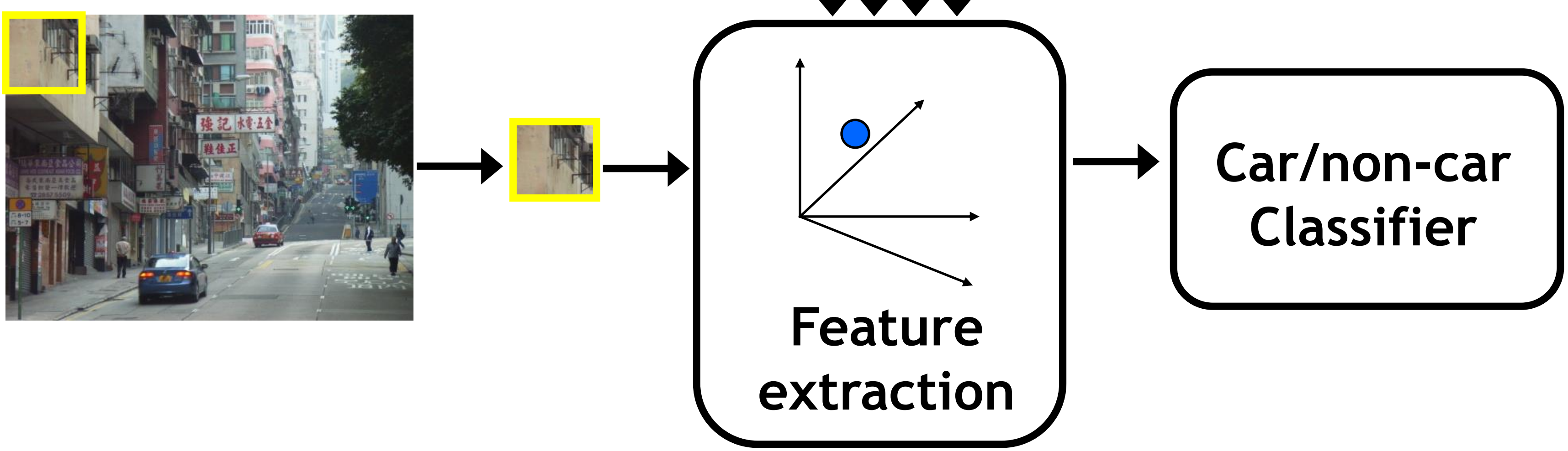
Training:

1. Obtain training data
2. Define features
3. Define classifier

Given new image:

1. Slide window
2. Score by classifier



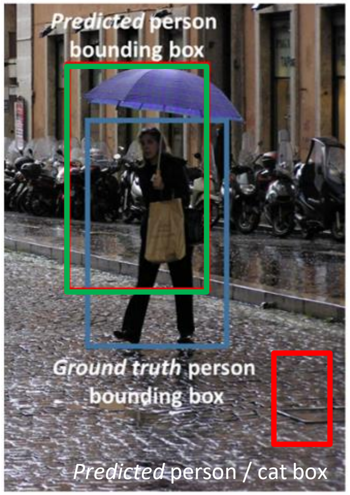


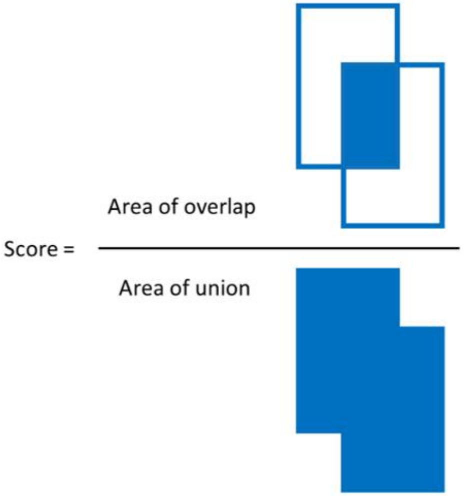
Kristen Grauman

Evaluating detection methods

$$mAP = \frac{1}{|classes|} \sum_{c \in classes} \frac{\#TP(c)}{\#TP(c) + \#FP(c)}$$

- True Positive - TP(c): a predicted bounding box (pred_bb) was made for class c, there is a ground truth bounding box (gt_bb) of class c, and IoU(pred_bb, gt_bb) >= 0.5.
- False Positive - FP(c): a pred_bb was made for class c, and there is no gt_bb of class c. Or there is a gt_bb of class c, but IoU(pred_bb, gt_bb) < 0.5.





Dalal-Triggs pedestrian detector

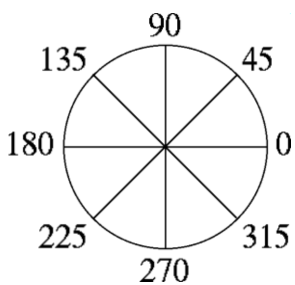
1. Extract fixed-sized (64x128 pixel) window at multiple positions and scales
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

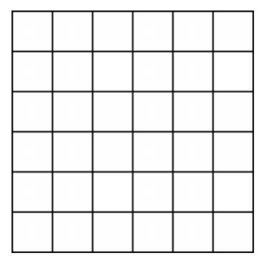
Histograms of oriented gradients (HOG)

Divide image into 8x8 regions

Orientation: 9 bins
(for unsigned angles)



Histograms in
8x8 pixel cells



Votes weighted by magnitude

Adapted from Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

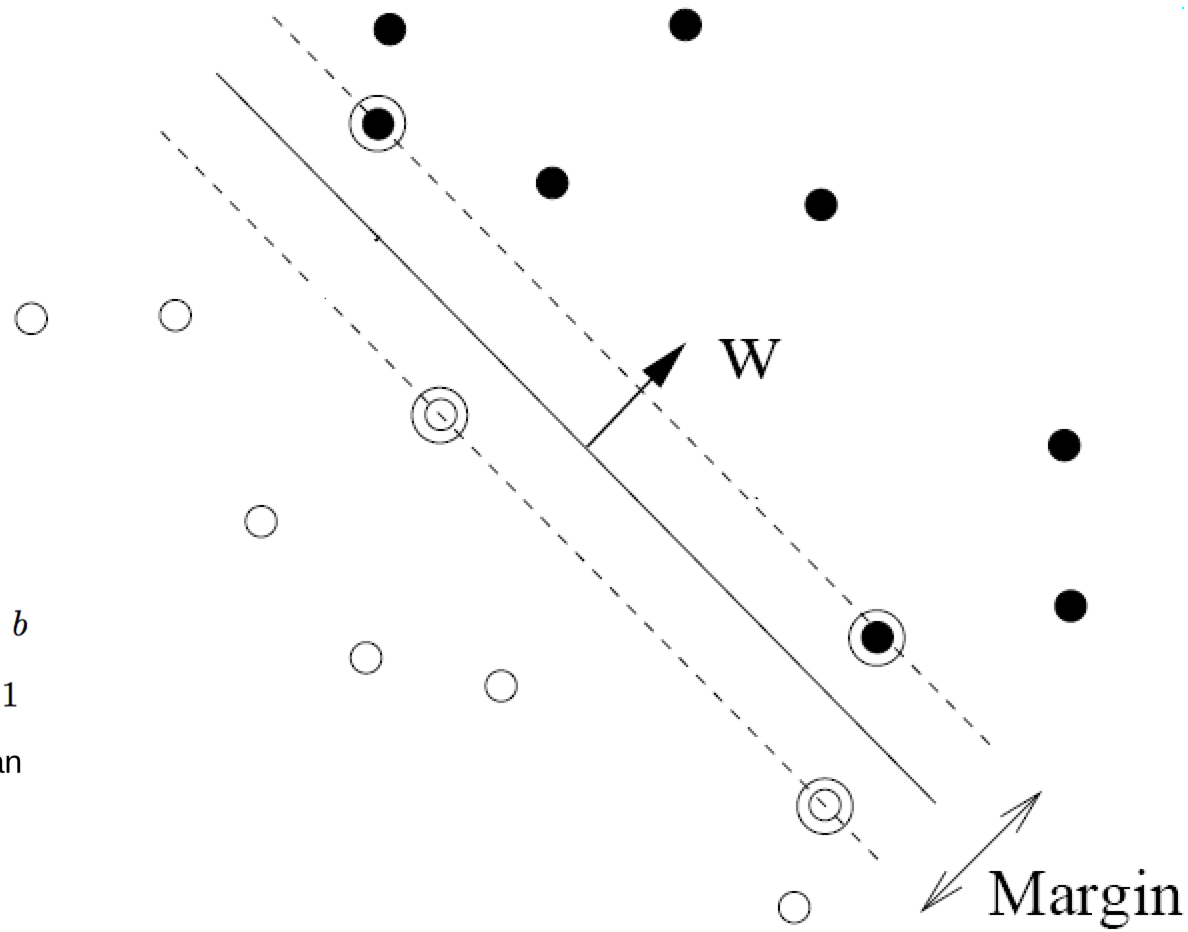
Train SVM for pedestrian detection using HoG

pos w

$$0.16 = w^T x + b$$

$$sign(0.16) = 1$$

=> pedestrian

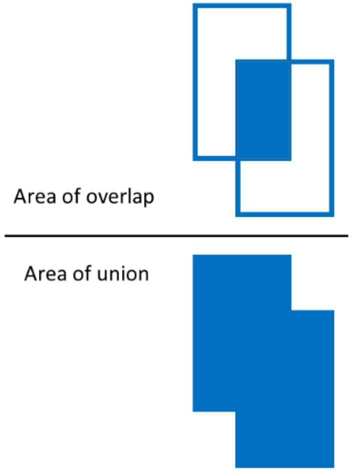


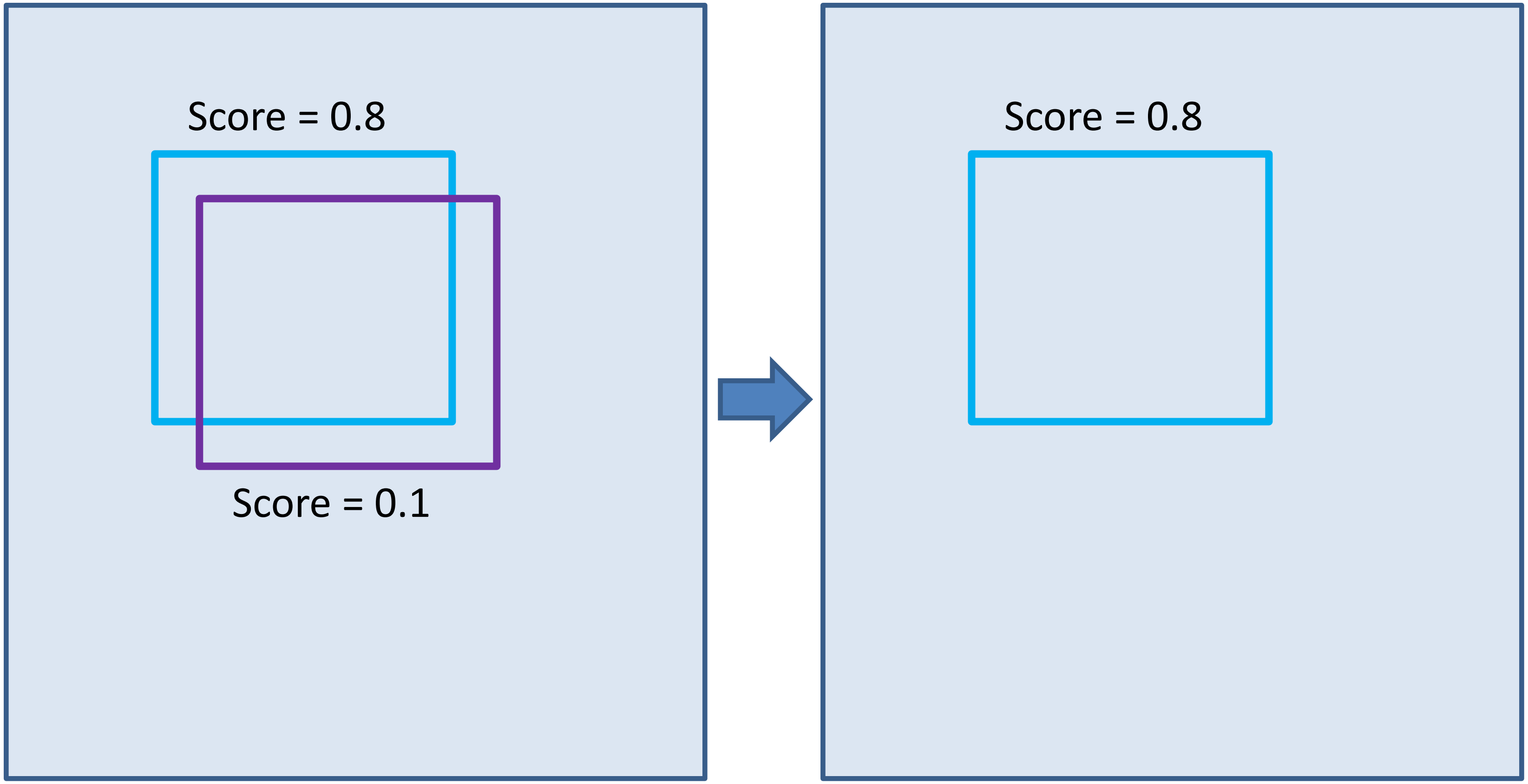
Adapted from Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Remove overlapping detections

Non-max suppression





Adapted from Derek Hoiem

Are window templates enough?

- Many objects are articulated, or have parts that can vary in configuration

Images from Caltech-256, D. Ramanan

- Many object categories look very different from different viewpoints, or from instance to instance

Adapted from N. Snavely, D. Tran

Parts-based Models

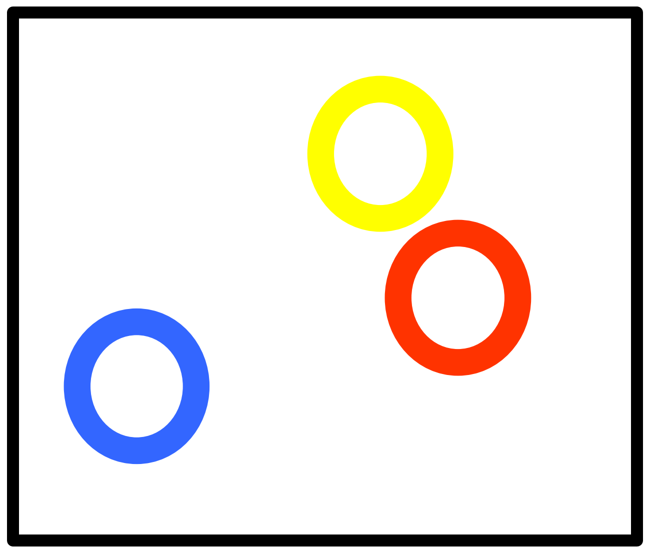
Define object by collection of parts modeled by

1. Appearance
2. Spatial configuration

Slide credit: Rob Fergus

How to model spatial relations?

- One extreme: fixed template



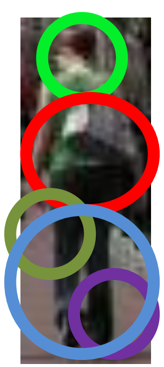
Derek Hoiem

Fixed part-based template

- Object model = sum of scores of features at fixed positions

$$+3 +2 -2 -1 -2.5 = -0.5 \overset{?}{>} 7.5$$

Non-object

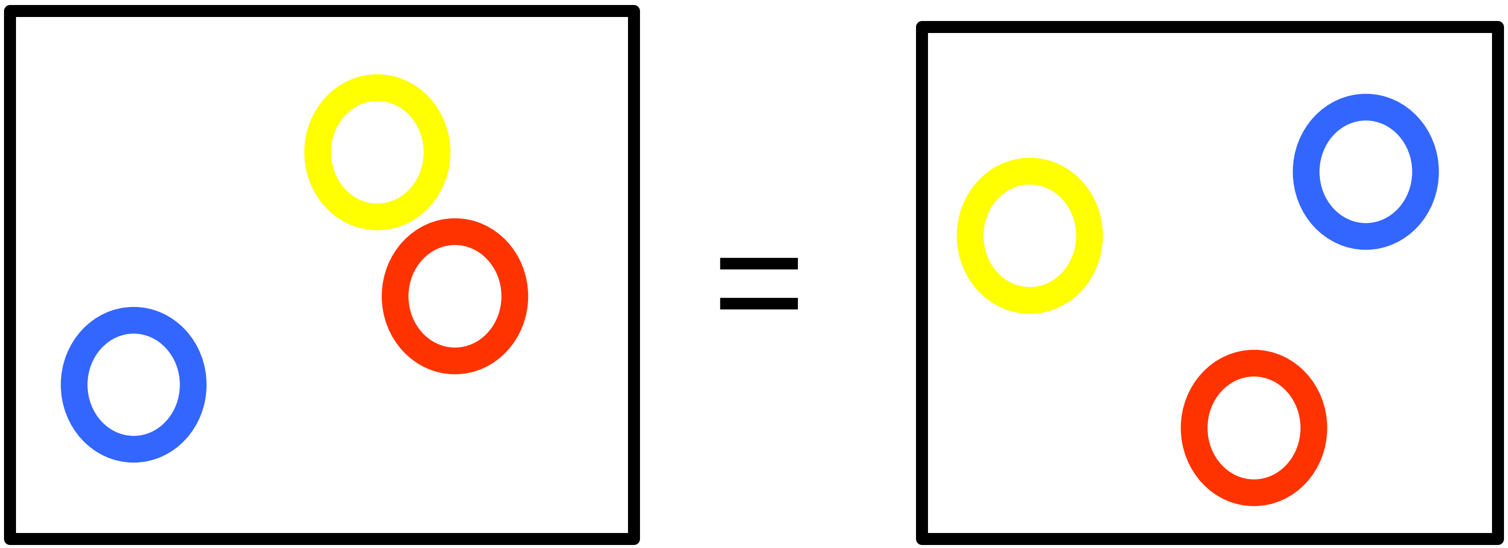


$$+4 +1 +0.5 +3 +0.5 = 10.5 \overset{?}{>} 7.5$$

Object

How to model spatial relations?

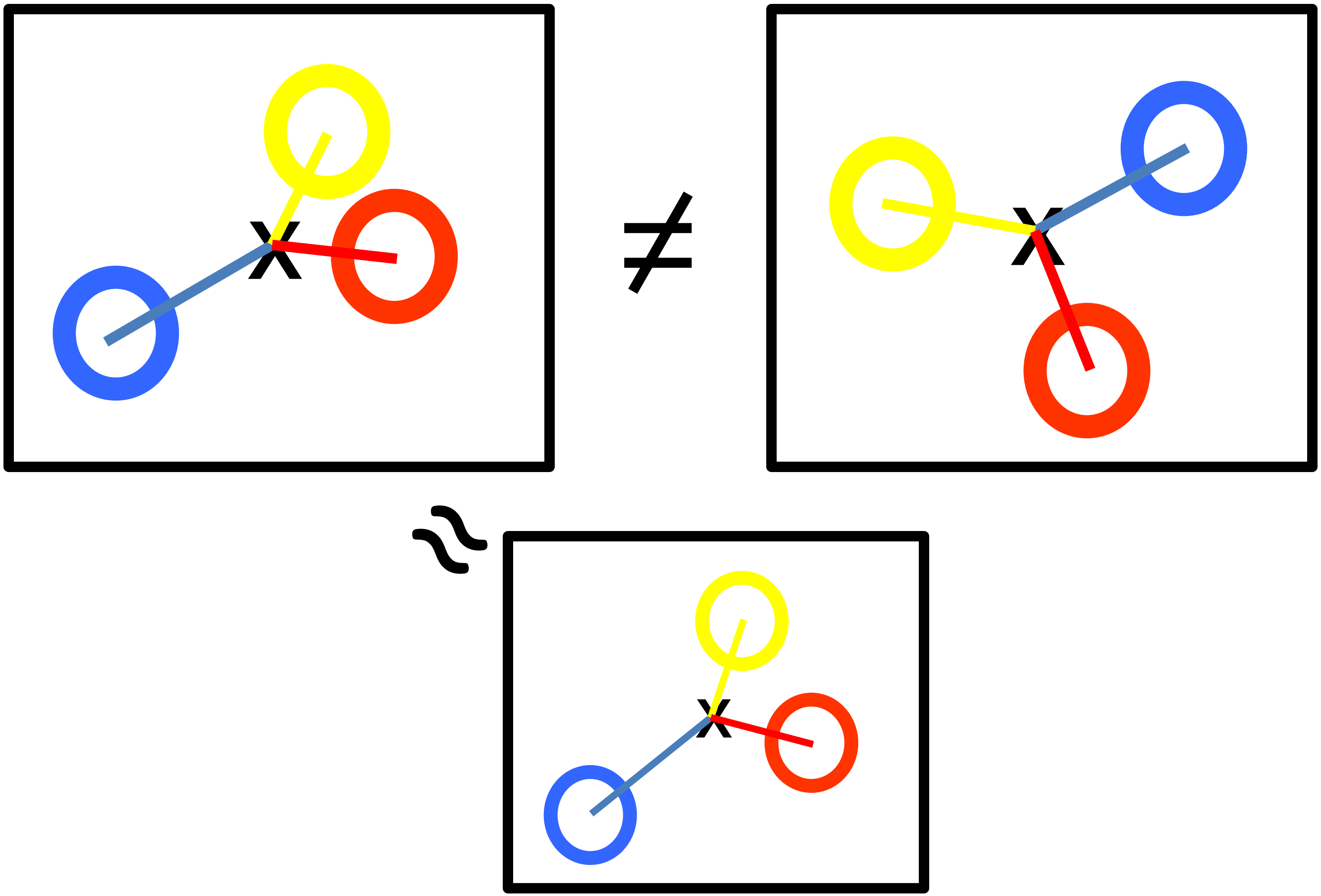
- Another extreme: bag of words



Derek Hoiem

How to model spatial relations?

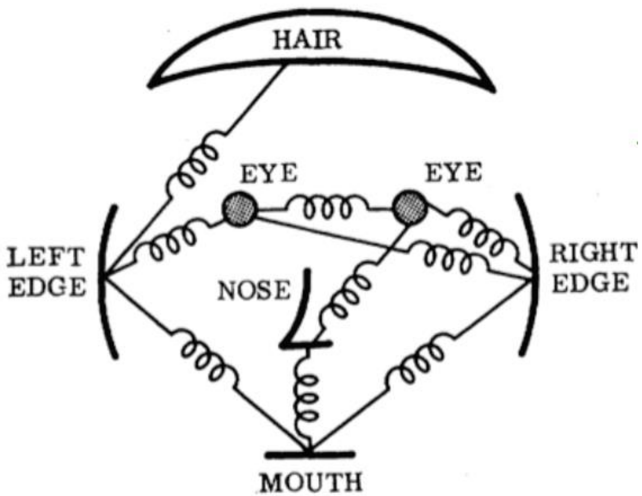
- Star-shaped model

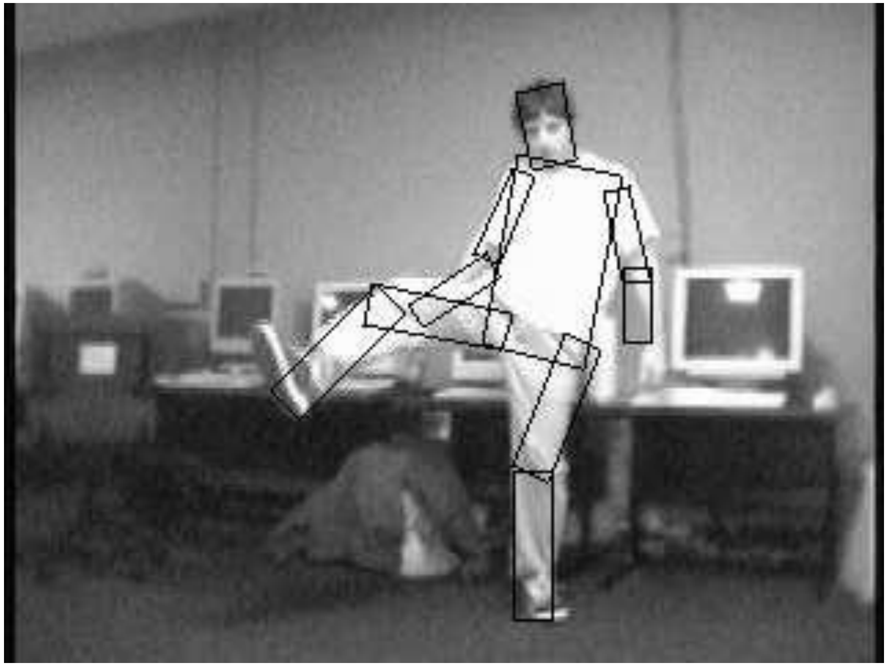


Derek Hoiem

Parts-based Models

- Articulated parts model
  - Object is configuration of parts
  - Each part is detectable and can move around





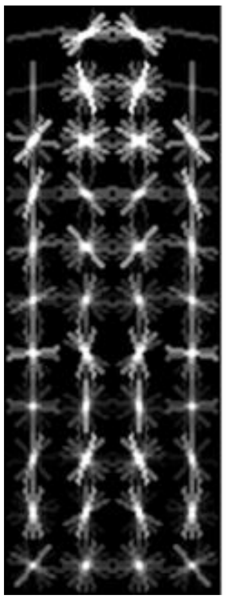
Adapted from Derek Hoiem, images from Felzenszwalb

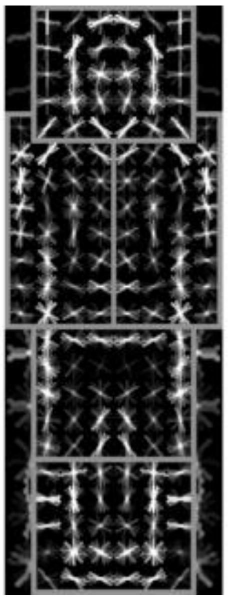
Deformable Part Models

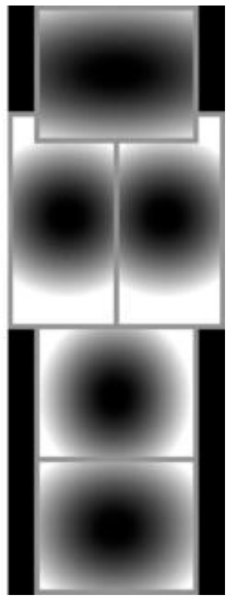
Root filter

Part filters

Deformation weights







P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, [Object Detection with Discriminatively Trained Part Based Models](), PAMI 32(9), 2010

Lana Lazebnik

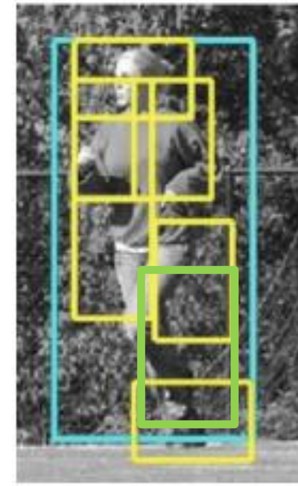
Scoring an object hypothesis

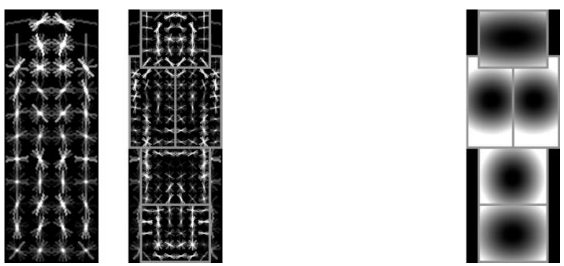
- The score of a hypothesis is the sum of appearance scores minus the sum of deformation costs

$z = (p_0, \ldots, p_n)$

p_0 : location of root

$p_1, \ldots, p_n$: location of parts





part loc (where we see part)

anchor loc (where we *expect* to see part)

$$(dx_i, dy_i) = (x_i, y_i) - (2(x_0, y_0) + v_i)$$

$$\text{score}(p_0, \ldots, p_n) = \sum_{i=0}^{n} \boxed{F_i'} \cdot \boxed{\phi(H, p_i)} - \sum_{i=1}^{n} \boxed{d_i} \cdot \boxed{\phi_d(dx_i, dy_i)} + b$$

Displacements

i.e. how much the part p_i moved from its expected anchor location in the x, y directions

Appearance weights

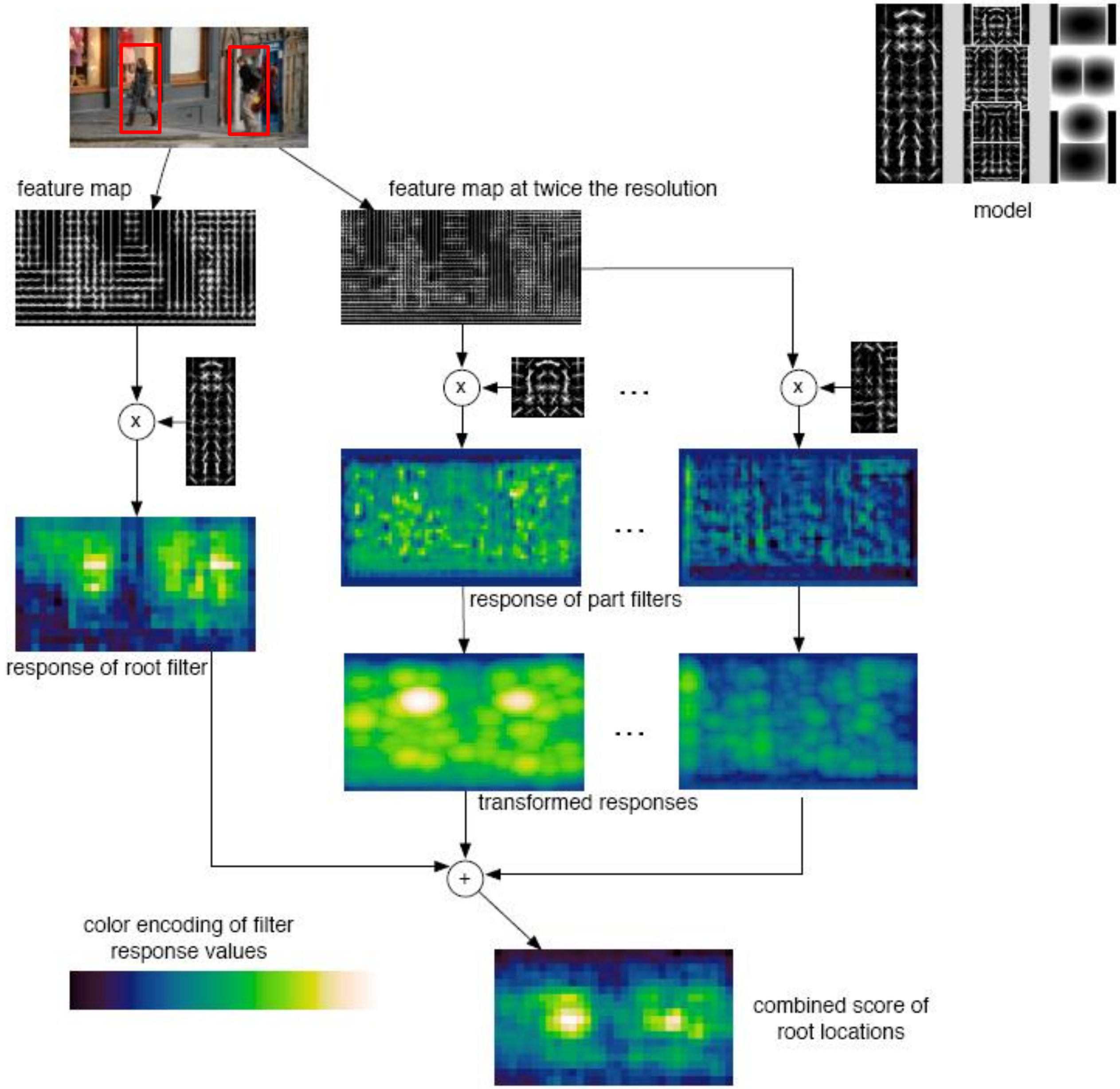
Part features

Deformation weights

i.e. how much we'll penalize the part p_i for moving from its expected location

Felzenszwalb et al.

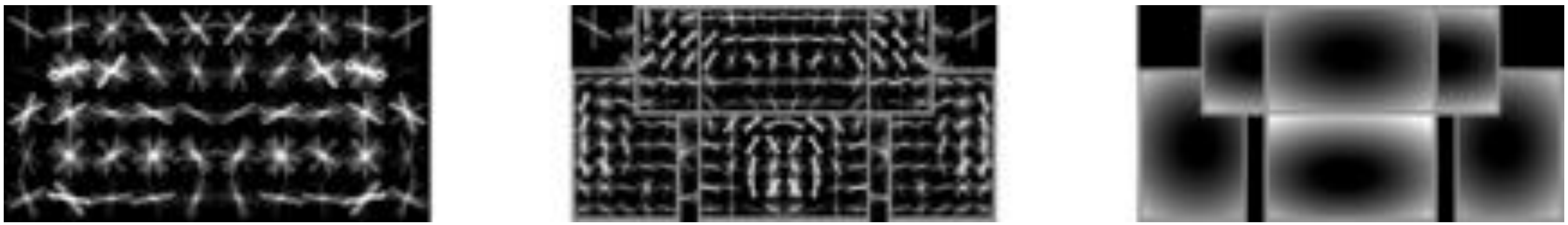
Detection



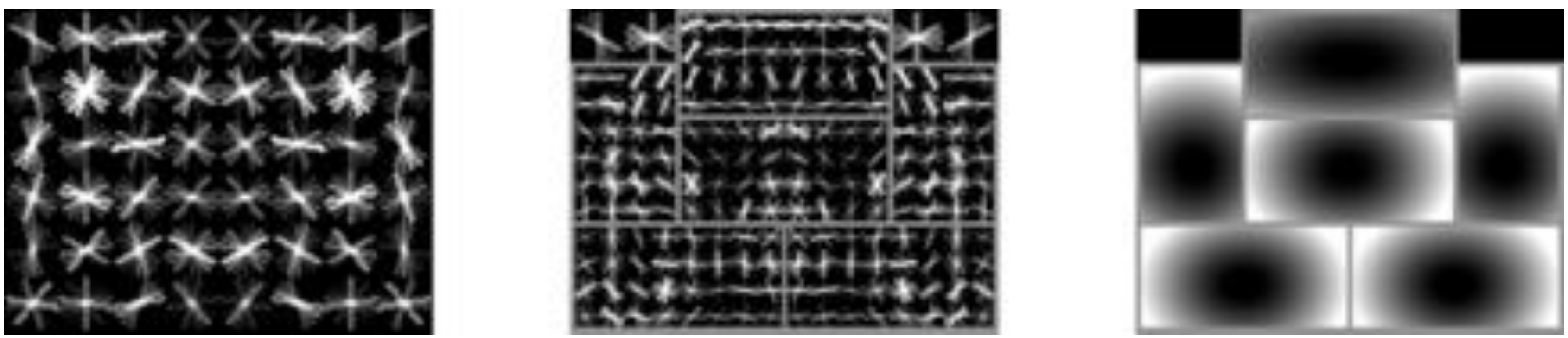
Felzenszwalb et al.

Car model

Component 1



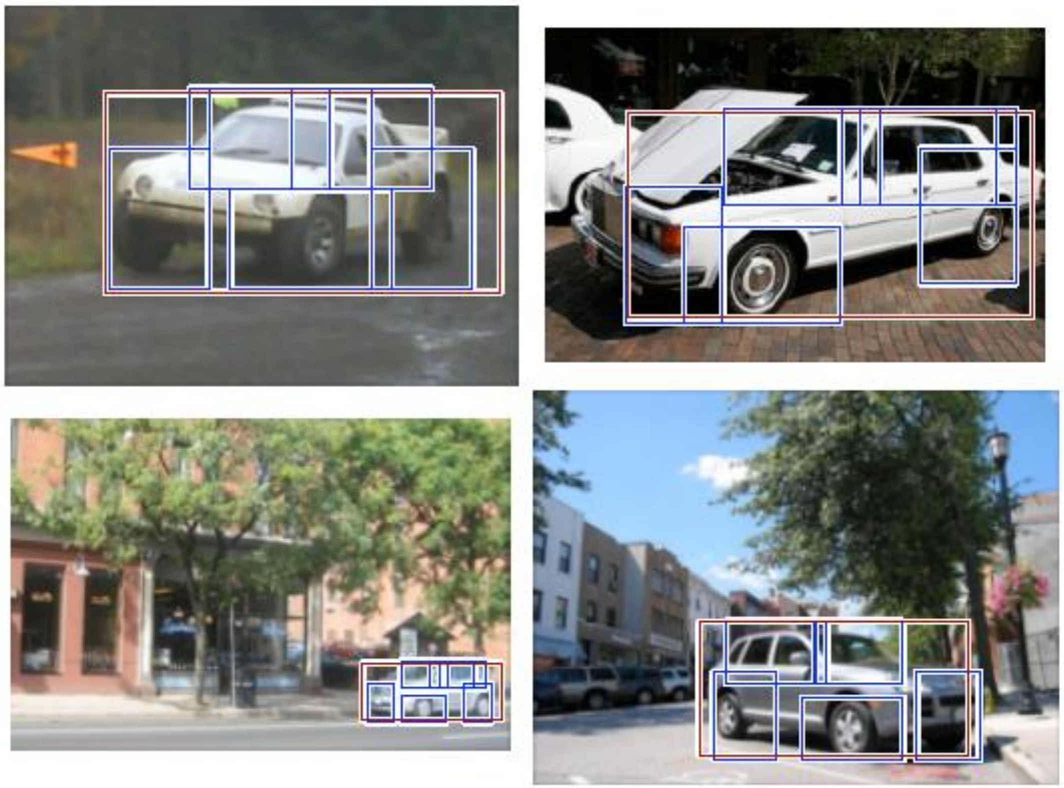
Component 2



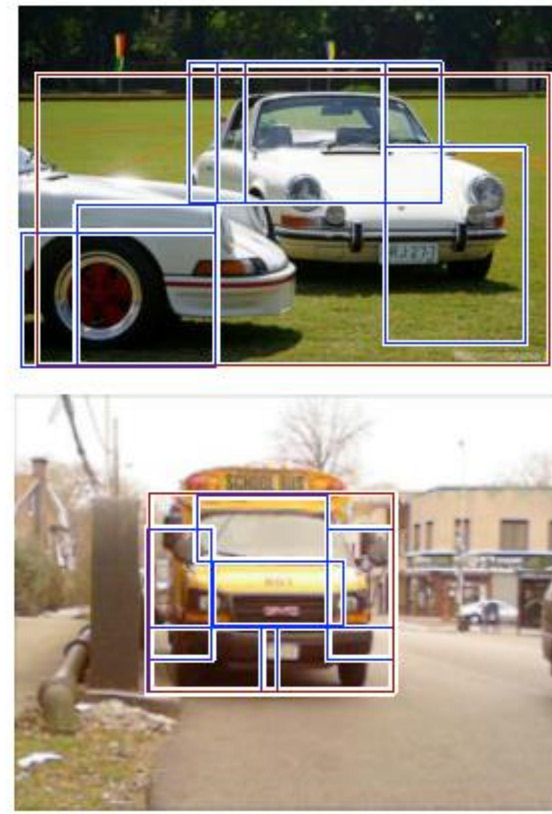
Lana Lazebnik

Car detections

high scoring true positives

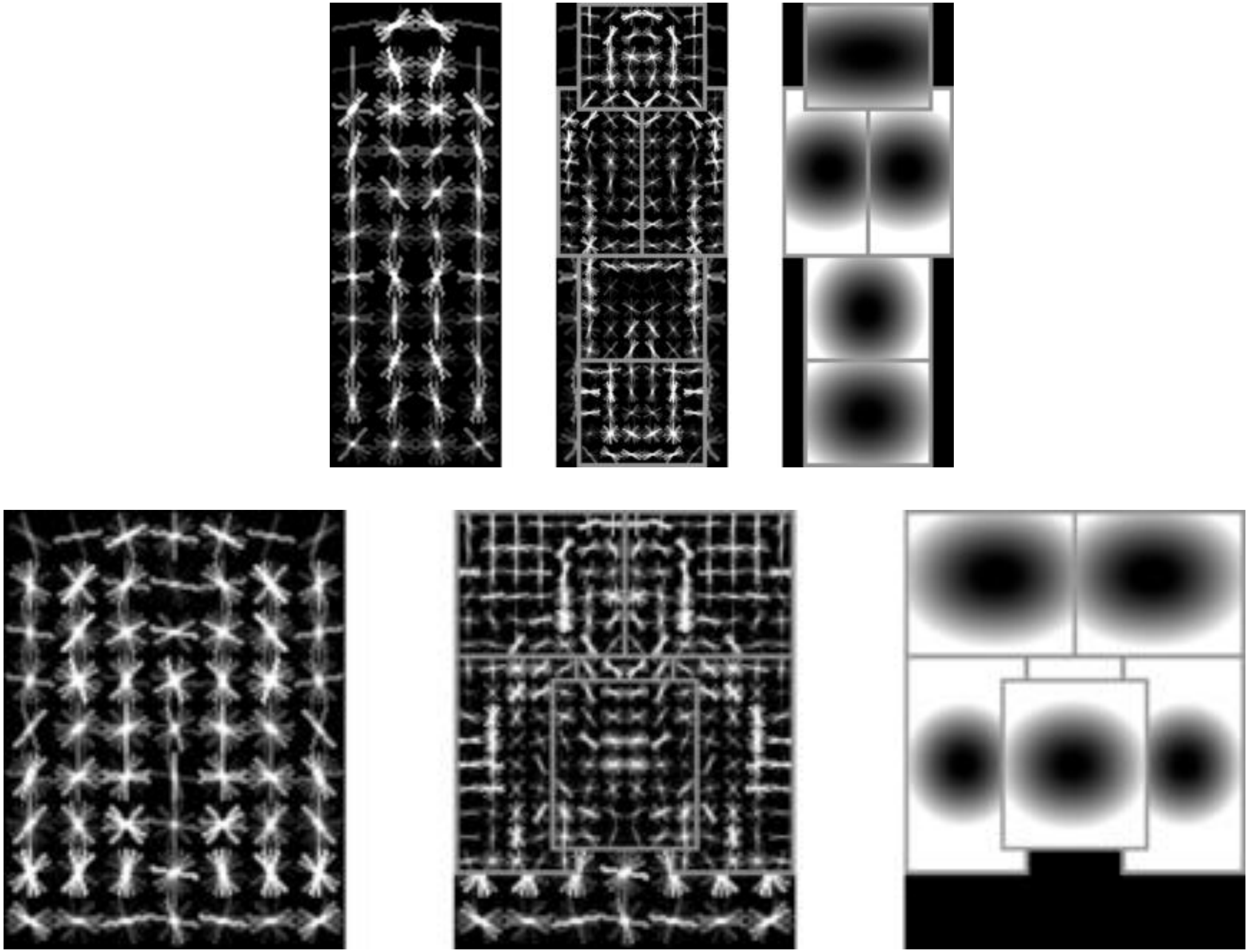


high scoring false positives



Lana Lazebnik

Person model



Lana Lazebnik

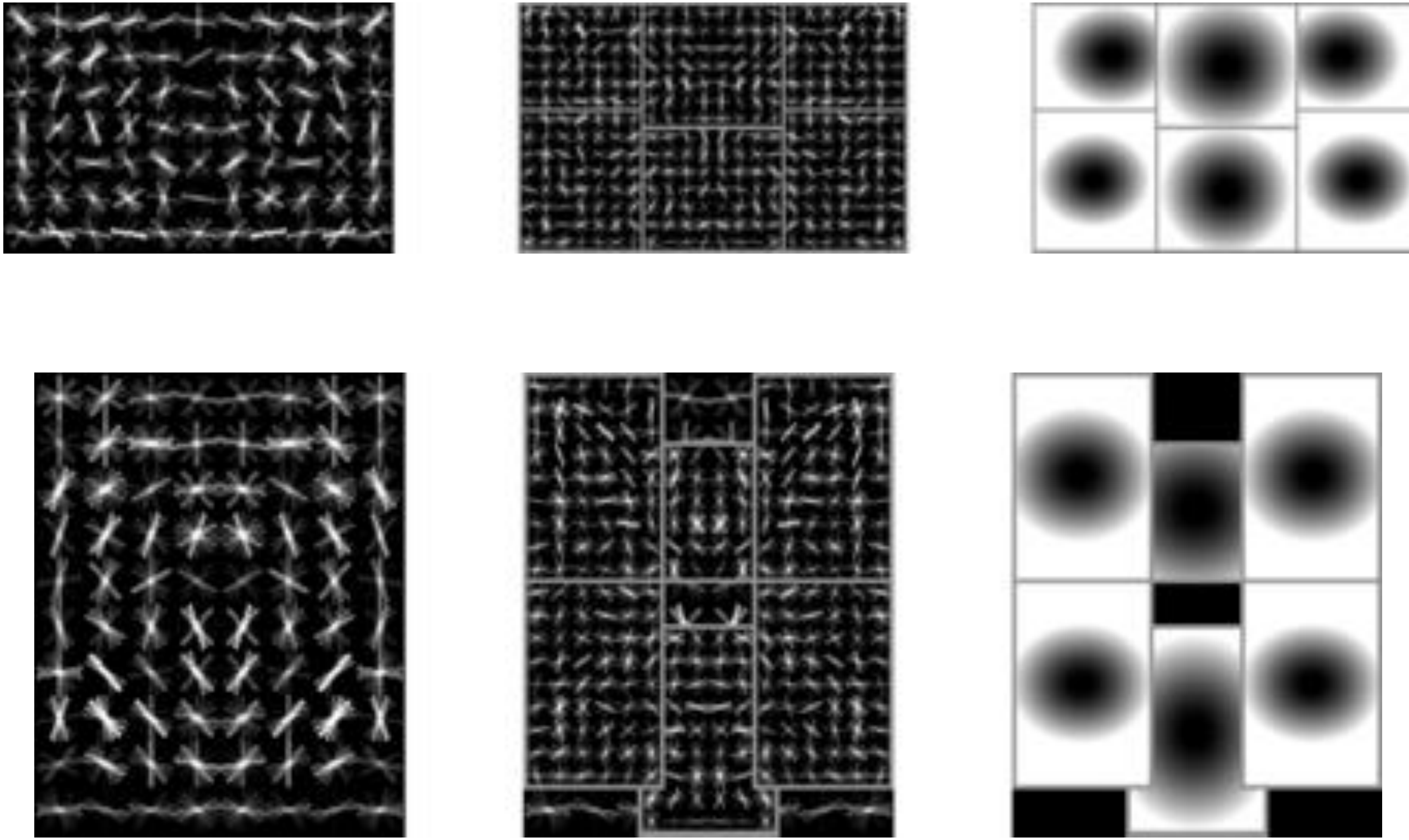
Person detections

high scoring true positives

high scoring false positives
(not enough overlap)

Lana Lazebnik

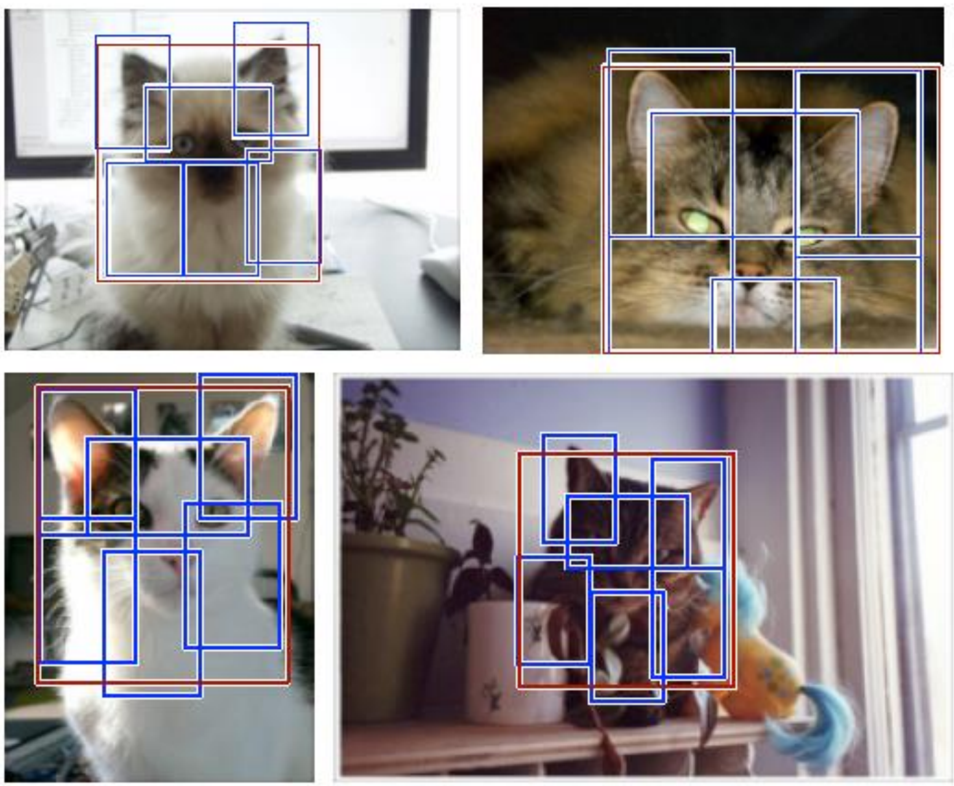
Cat model



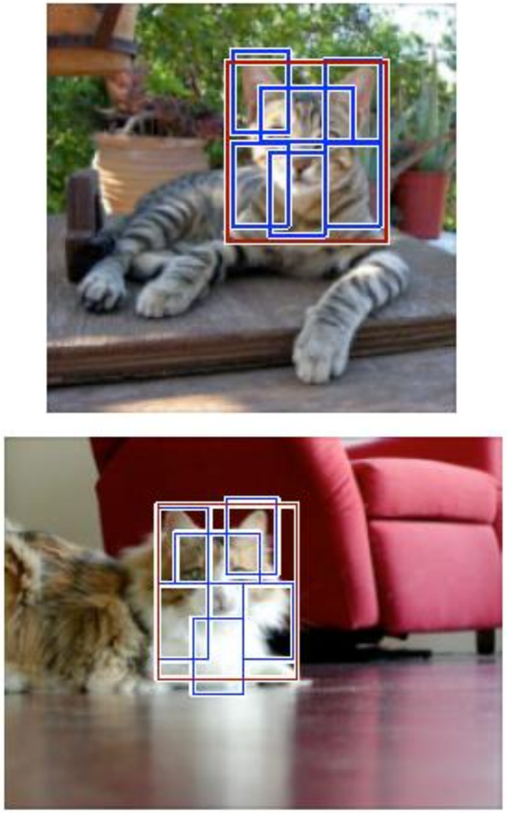
Lana Lazebnik

Cat detections

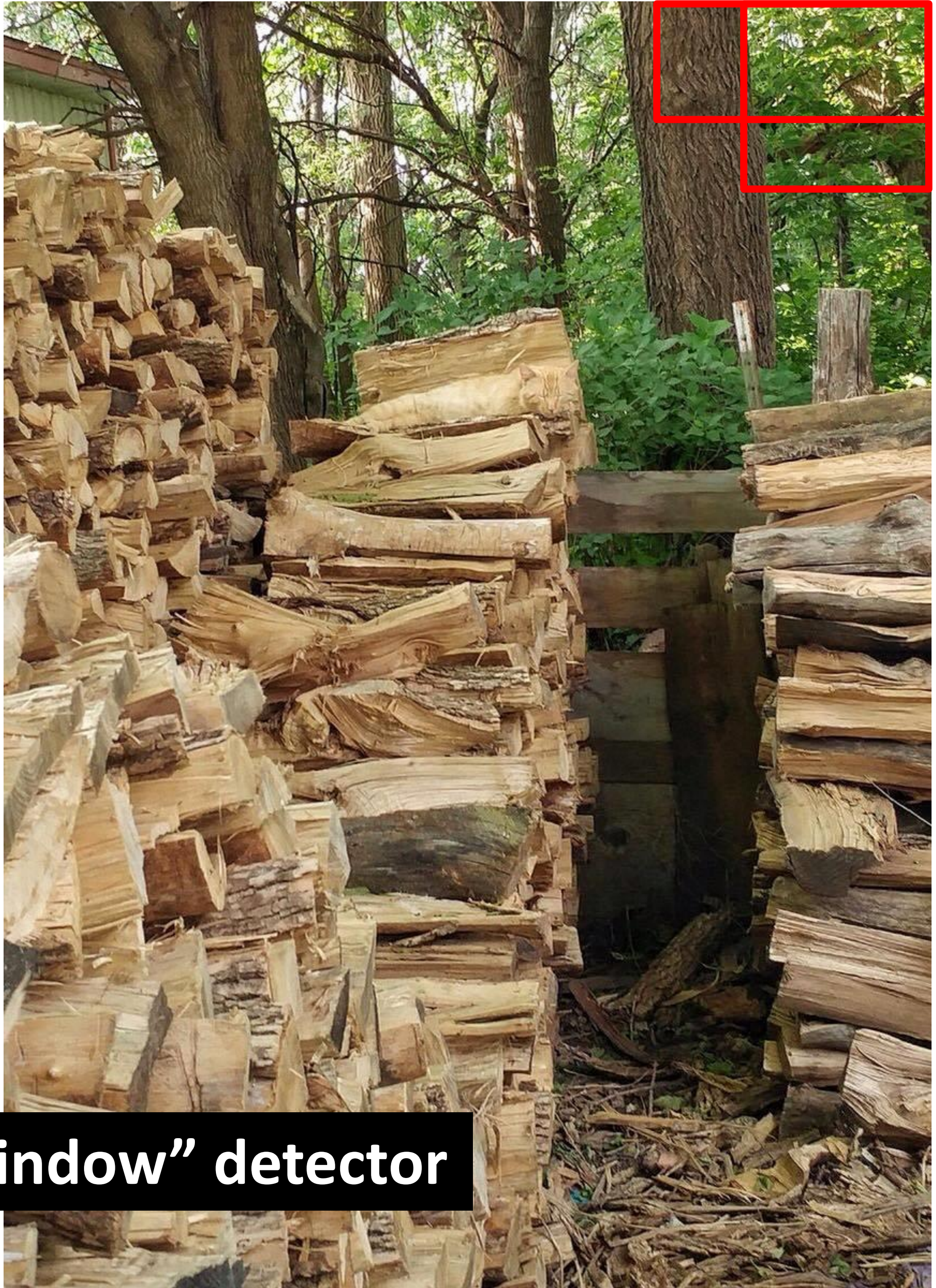
high scoring true positives



high scoring false positives
(not enough overlap)



Lana Lazebnik



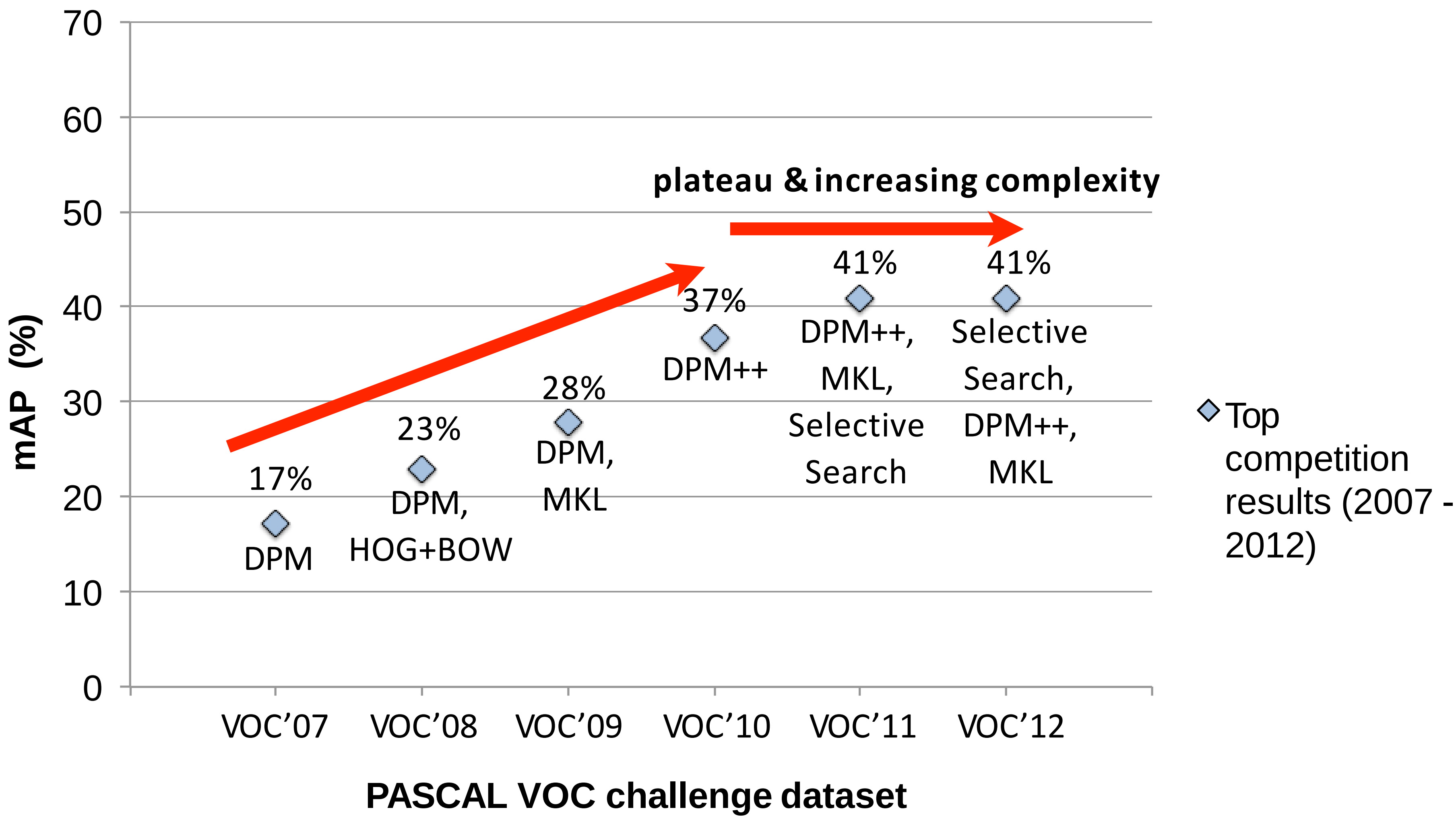
“Sliding window” detector

Plan for the next three lectures

- Detection approaches
  - Pre-CNNs
    - Detection with whole windows: Pedestrian detection
    - Part-based detection: Deformable Part Models
  - Post-CNNs
    - Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    - Detection without region proposals: YOLO, SSD
- Segmentation approaches
  - Semantic segmentation: FCN
  - Instance segmentation: Mask R-CNN

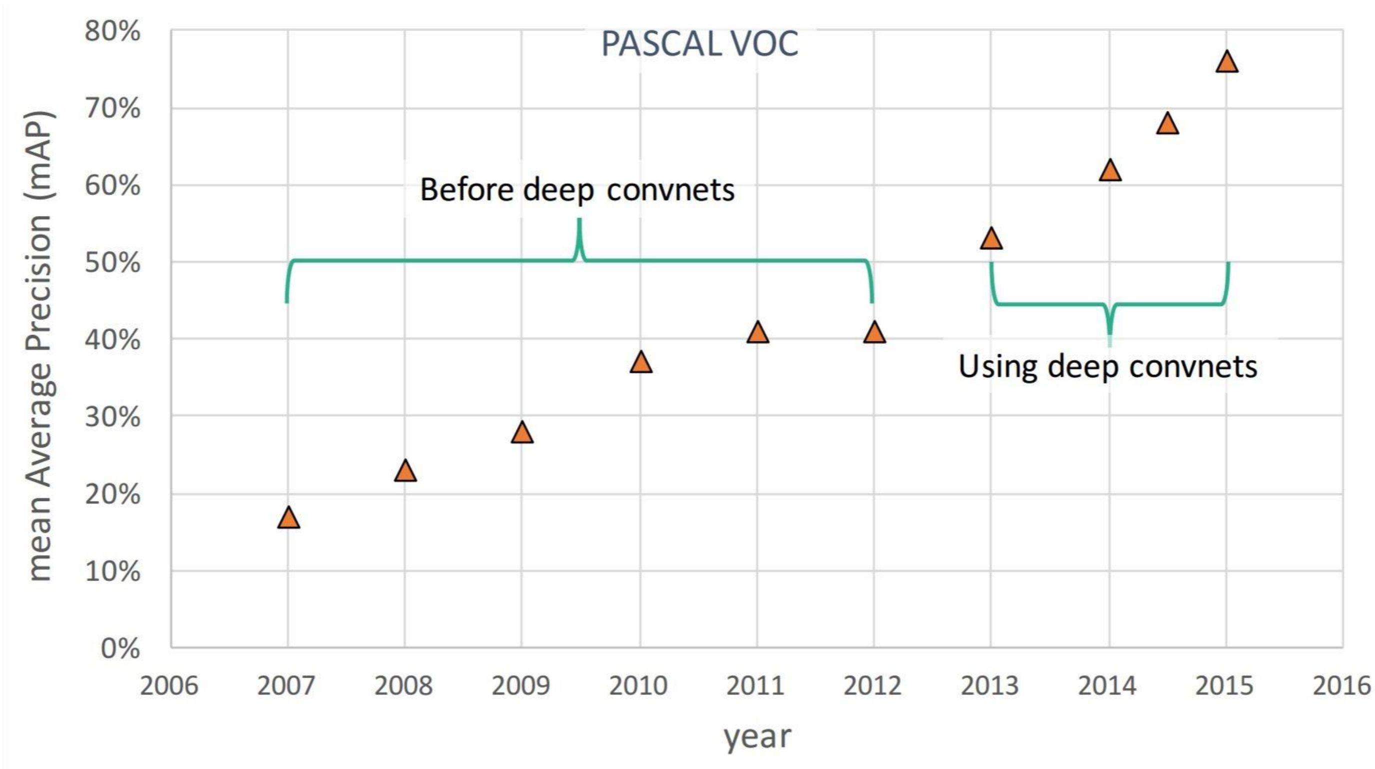
Complexity and the plateau

[Source: http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc20{07,08,09,10,11,12}/results/index.html]



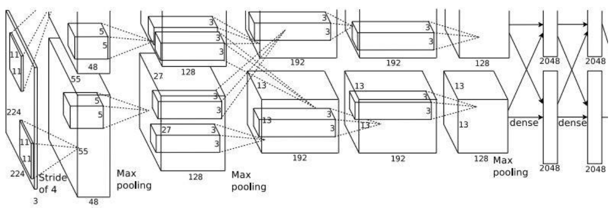
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

Impact of Deep Learning



Slide by: Justin Johnson

Before: Image Classification with CNNs



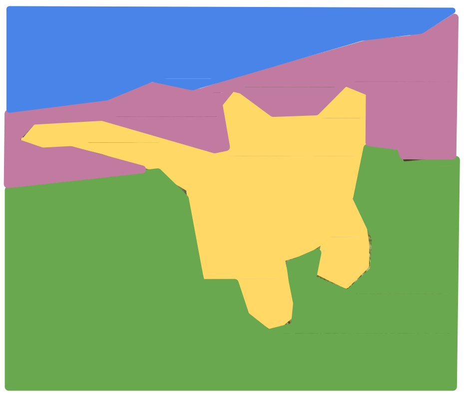
Vector:
4096

Fully-Connected:
4096 to 1000

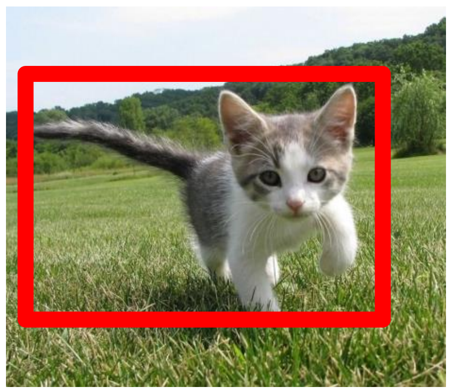
Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Slide by: Justin Johnson

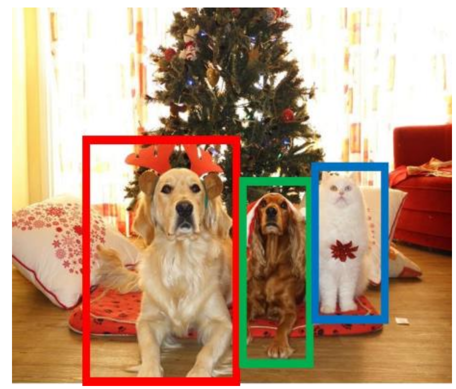
Classification + Localization



GRASS, CAT, TREE, SKY



CAT



DOG, DOG, CAT

DOG, DOG, CAT

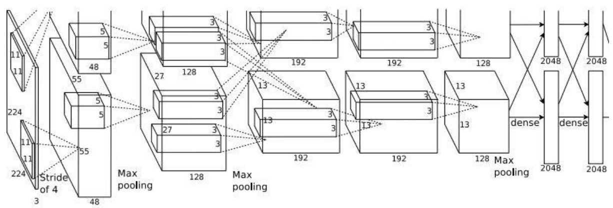
No objects, just pixels

Single Object

Multiple Object

Slide by: Justin Johnson

Classification + Localization



Vector: 4096

Fully Connected: 4096 to 1000

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

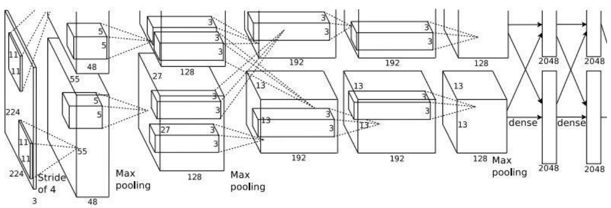
Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

Treat localization as a regression problem!

Slide by: Justin Johnson

Classification + Localization



Vector: 4096

Fully Connected: 4096 to 1000

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Correct label: Cat

Softmax Loss

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

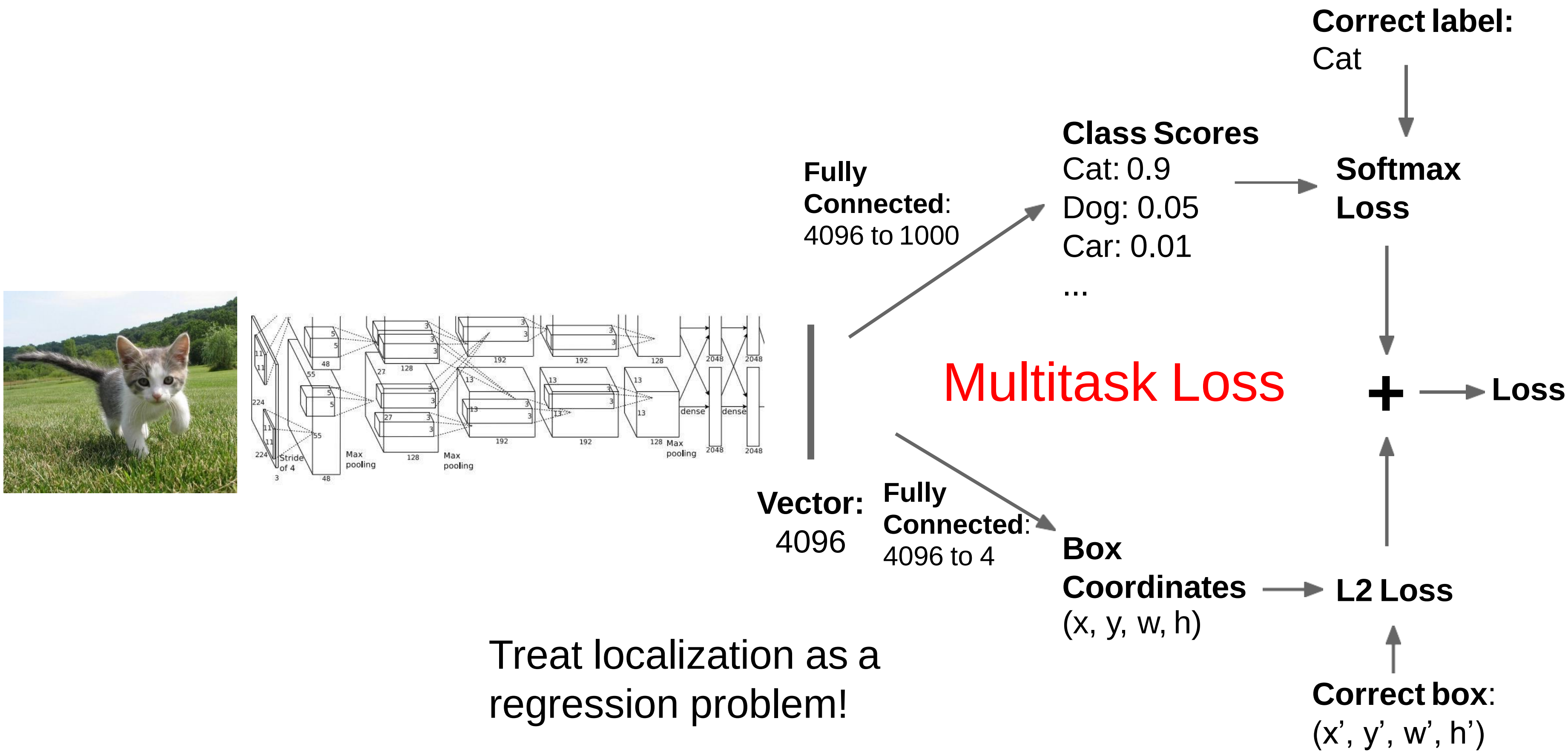
L2 Loss

Correct box: (x', y', w', h')

Treat localization as a regression problem!

Slide by: Justin Johnson

Classification + Localization



Treat localization as a regression problem!

Slide by: Justin Johnson

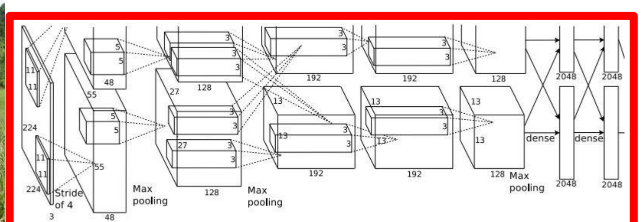
Classification + Localization

Correct label:
Cat

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Softmax Loss

Fully Connected:
4096 to 1000



Vector:
4096

Fully Connected:
4096 to 4

+ **Loss**

Often pretrained on ImageNet
(Transfer learning)

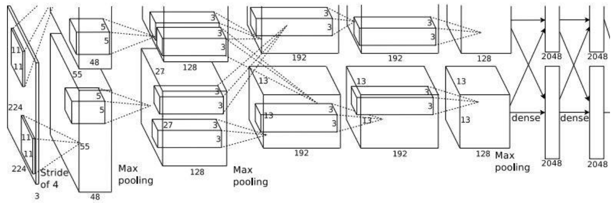
Box Coordinates
(x, y, w, h)

L2 Loss

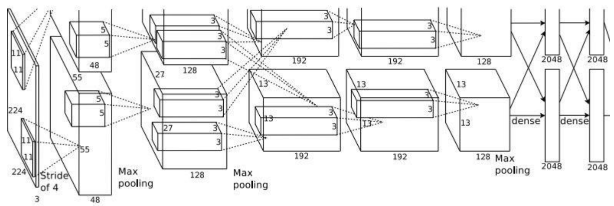
Correct box:
(x’, y’, w’, h’)

Treat localization as a regression problem!

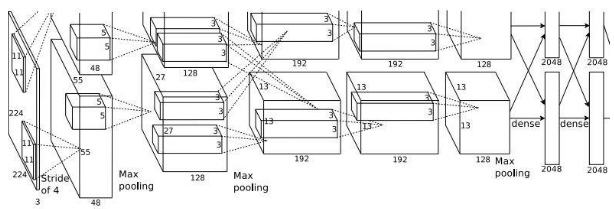
Object Detection as Regression?

CAT: (x, y, w, h)

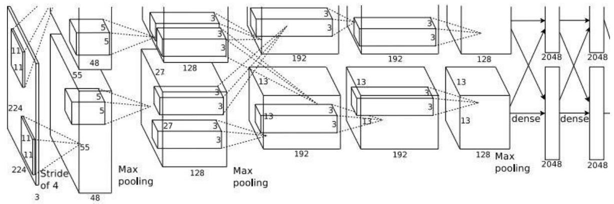
DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
….

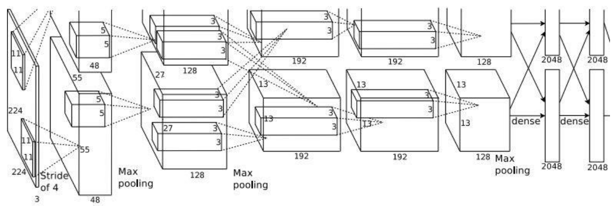
Slide by: Justin Johnson

Object Detection as Regression?



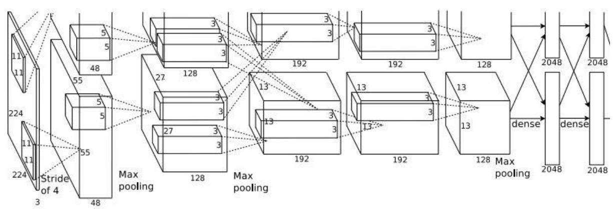
CAT: (x, y, w, h)

4 numbers



DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

16 numbers



DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
….

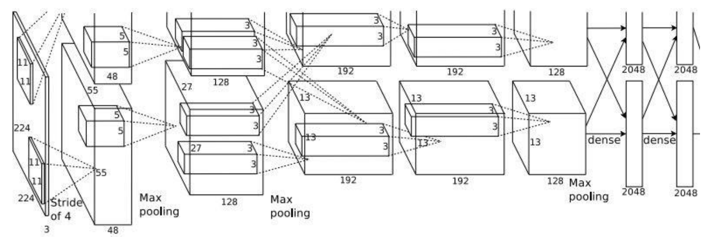
Many numbers!

Each image needs a different number of outputs!

Slide by: Justin Johnson

Object Detection as Classification: Sliding Window

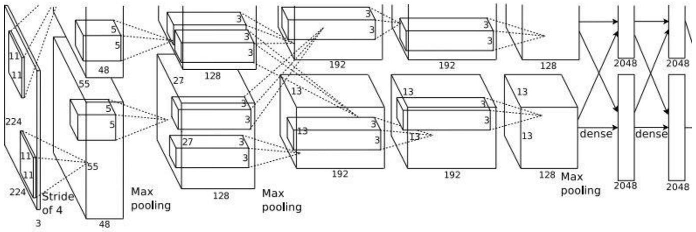
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? NO
Background? YES

Object Detection as Classification: Sliding Window

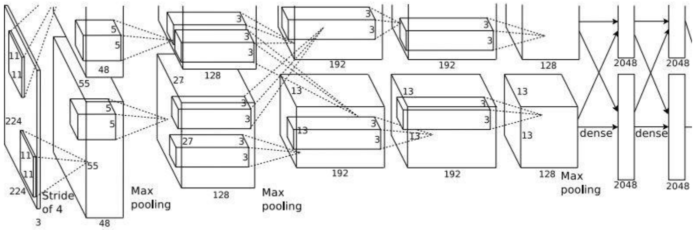
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

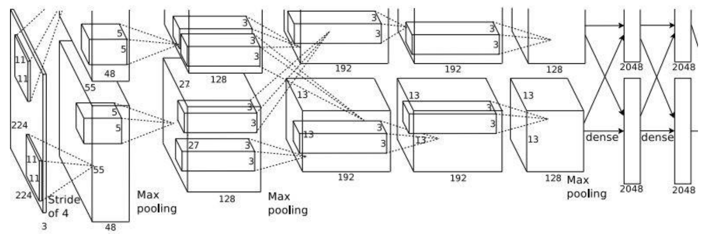


Dog? YES
Cat? NO
Background? NO

Slide by: Justin Johnson

Object Detection as Classification: Sliding Window

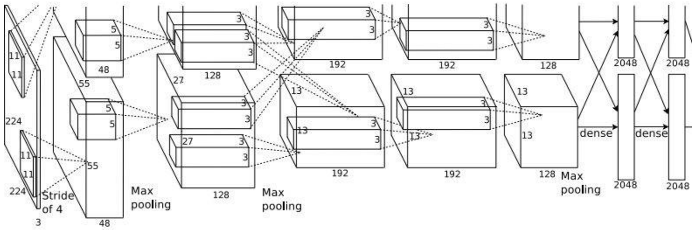
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? YES
Background? NO

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



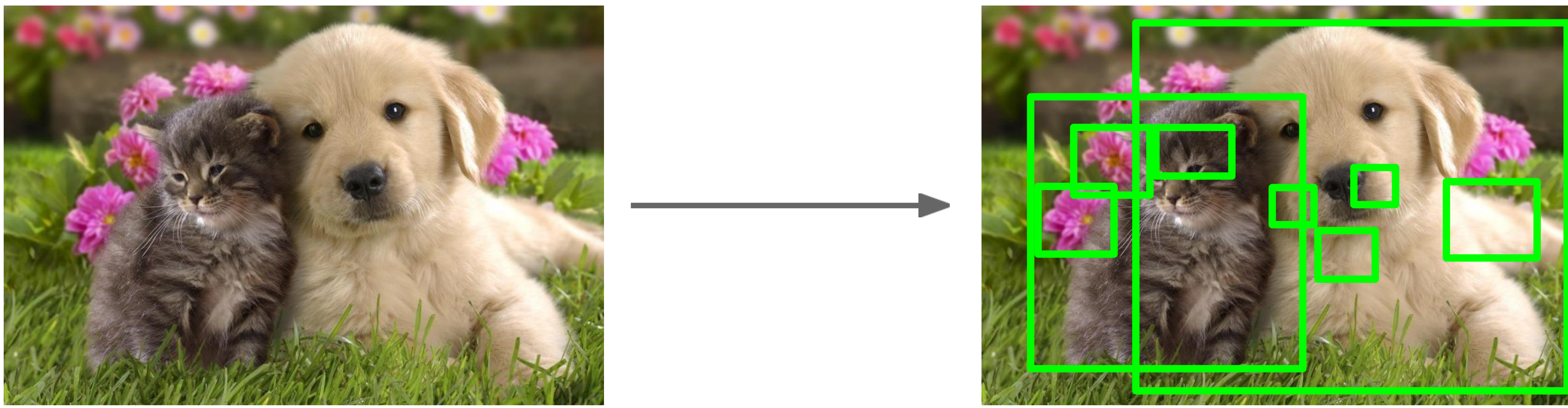
Dog? NO
Cat? YES
Background? NO

Problem: Need to apply CNN to huge number of locations and scales, very computationally expensive!

Slide by: Justin Johnson

Region Proposals

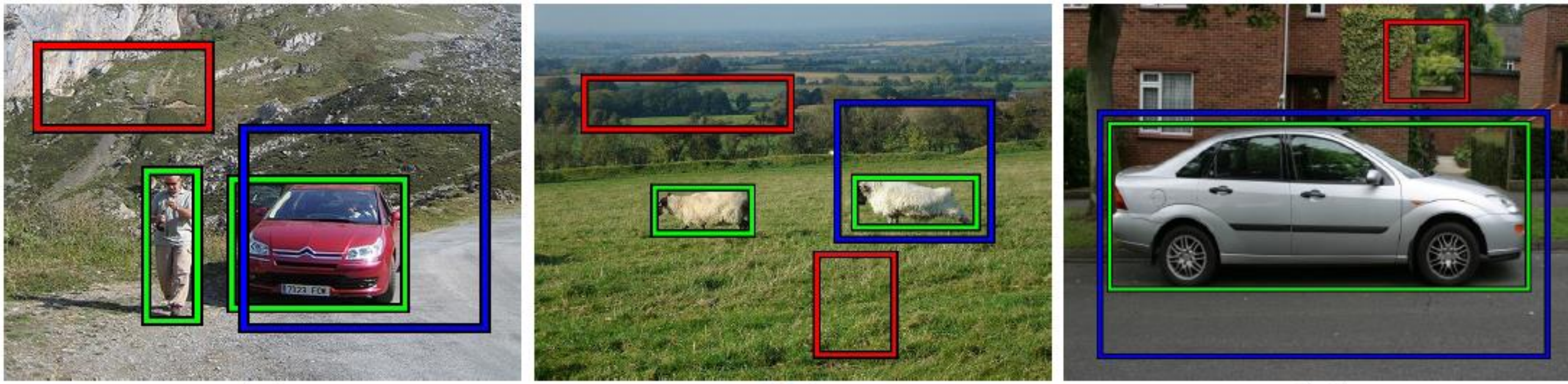
- Find “blobby” image regions that are likely to contain objects
- Relatively fast to run; e.g. Selective Search gives 1000 region proposals in a few seconds on CPU



Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012
Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013
Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014
Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”, ECCV 2014

Slide by: Justin Johnson

Speeding up detection: Restrict set of windows we pass through SVM to those w/ high "objectness"



(a) (b) (c)

Fig. 1: **Desired behavior of an objectness measure.** *The desired objectness measure should score the blue windows, partially covering the objects, lower than the ground truth windows (green), and score even lower the red windows containing only stuff or small parts of objects.*

Alexe et al., CVPR 2010

Objectness cue #1: Where people look

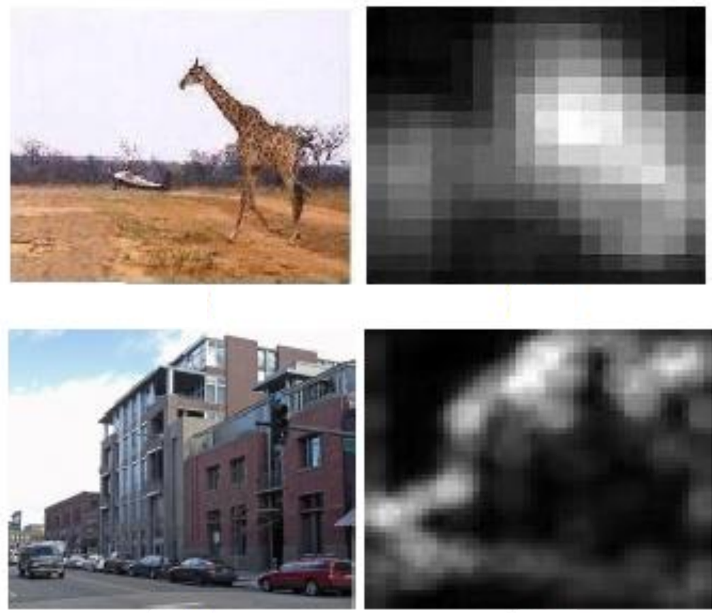


Fig. 2: **MS success and failure.**

Alexe et al., CVPR 2010

Objectness cue #2: color contrast at boundary

(a) (b) (c)

Fig. 3: **CC success and failure. Success:** *the windows containing the objects (cyan) have high color contrast with their surrounding ring (yellow) in images (a) and (b).* **Failure:** *the color contrast for windows in cyan in image (c) is much lower.*

Alexe et al., CVPR 2010

Objectness cue #3:
no segments “straddling” the object box

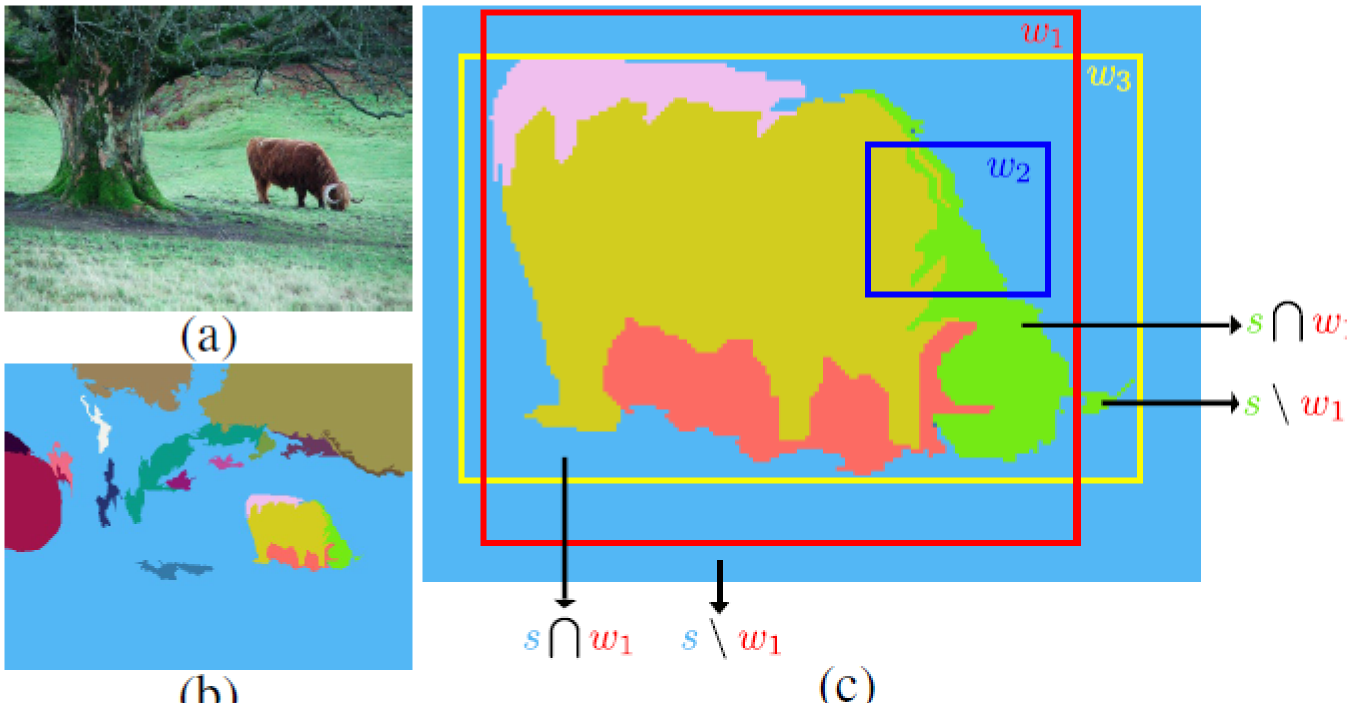


Fig. 5: **The SS cue.** *Given the segmentation (b) of image (a), for a window w we compute* $\mathrm{SS}(\mathrm{w}, \theta_{\mathrm{SS}})$ *(eq. 4). In (c), most of the surface of w_1 is covered by superpixels contained almost entirely inside it. Instead, all superpixels passing by w_2 continue largely outside it. Therefore, w_1 has a higher* SS *score than w_2. The window w_3 has an even higher score as it fits the object tightly.*

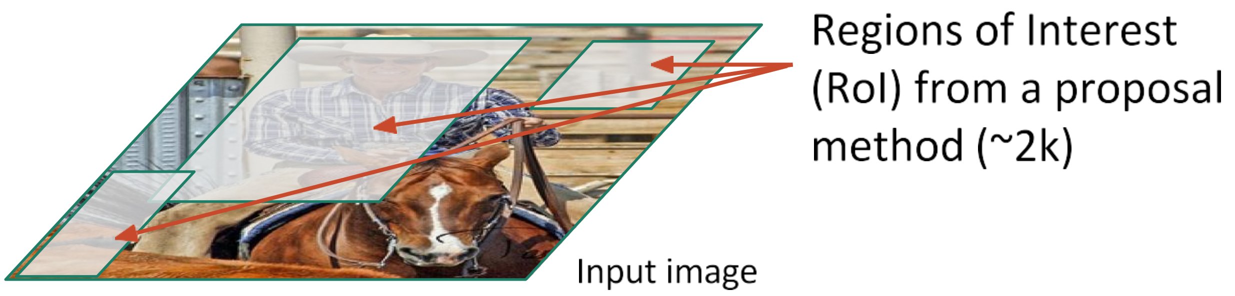
Alexe et al., CVPR 2010

R-CNN

Input image

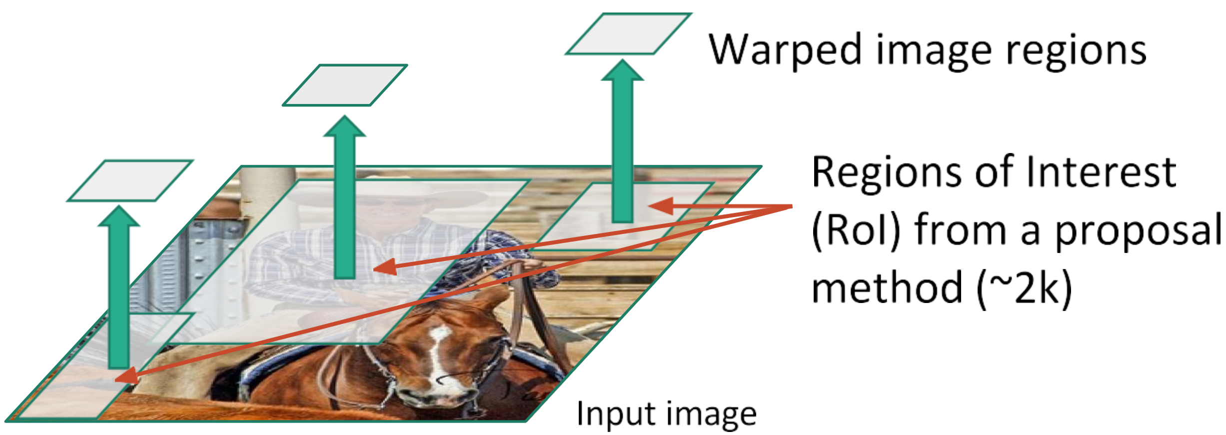
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN



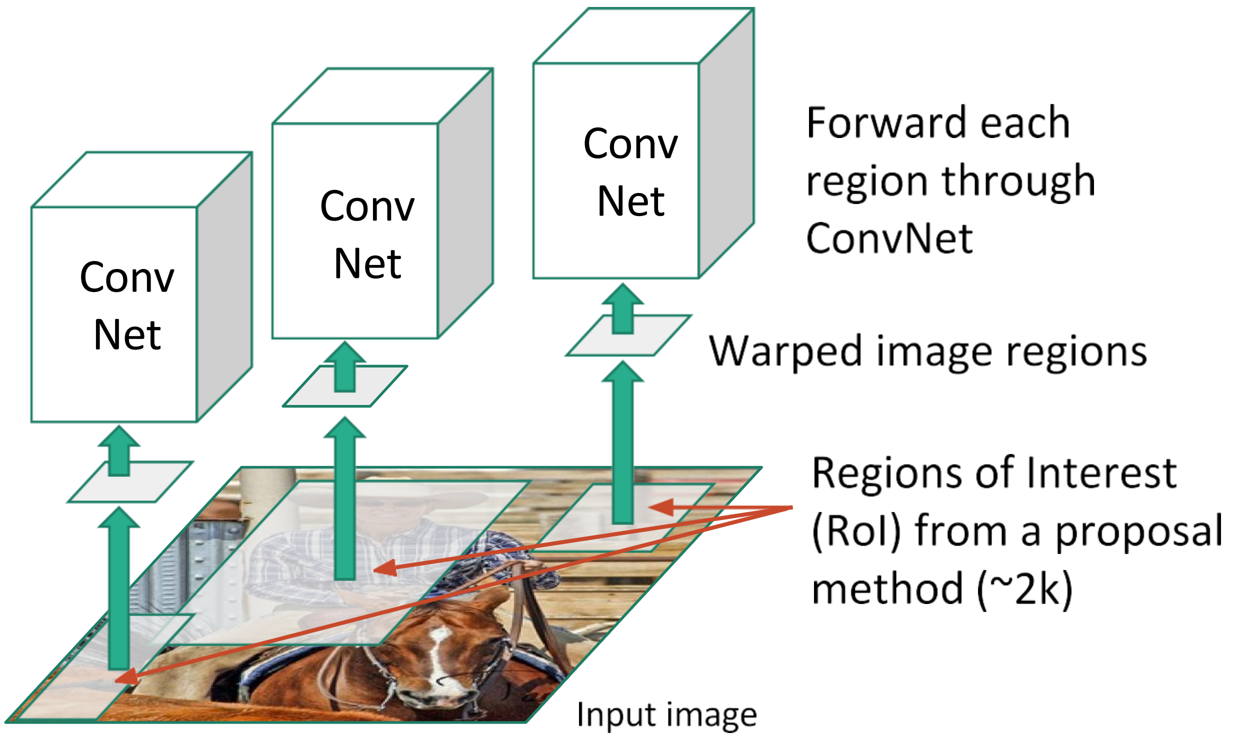
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN



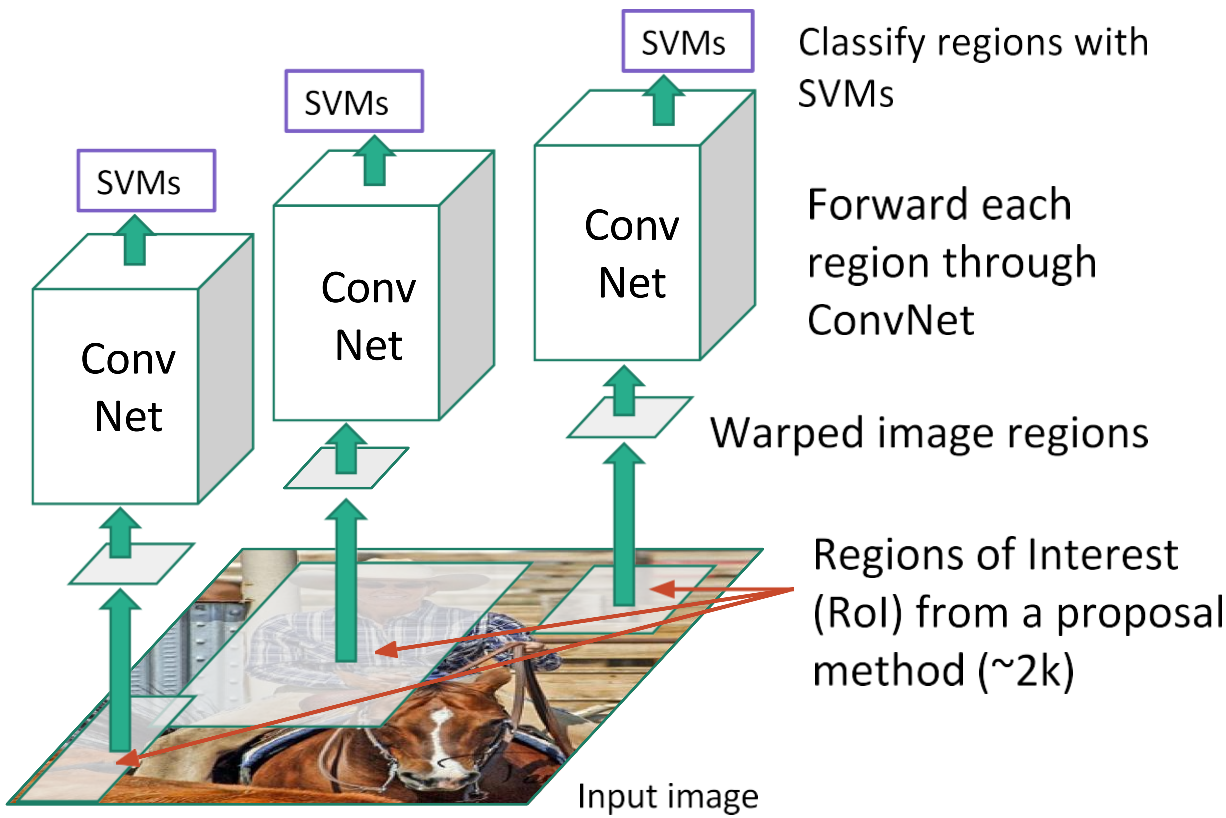
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN



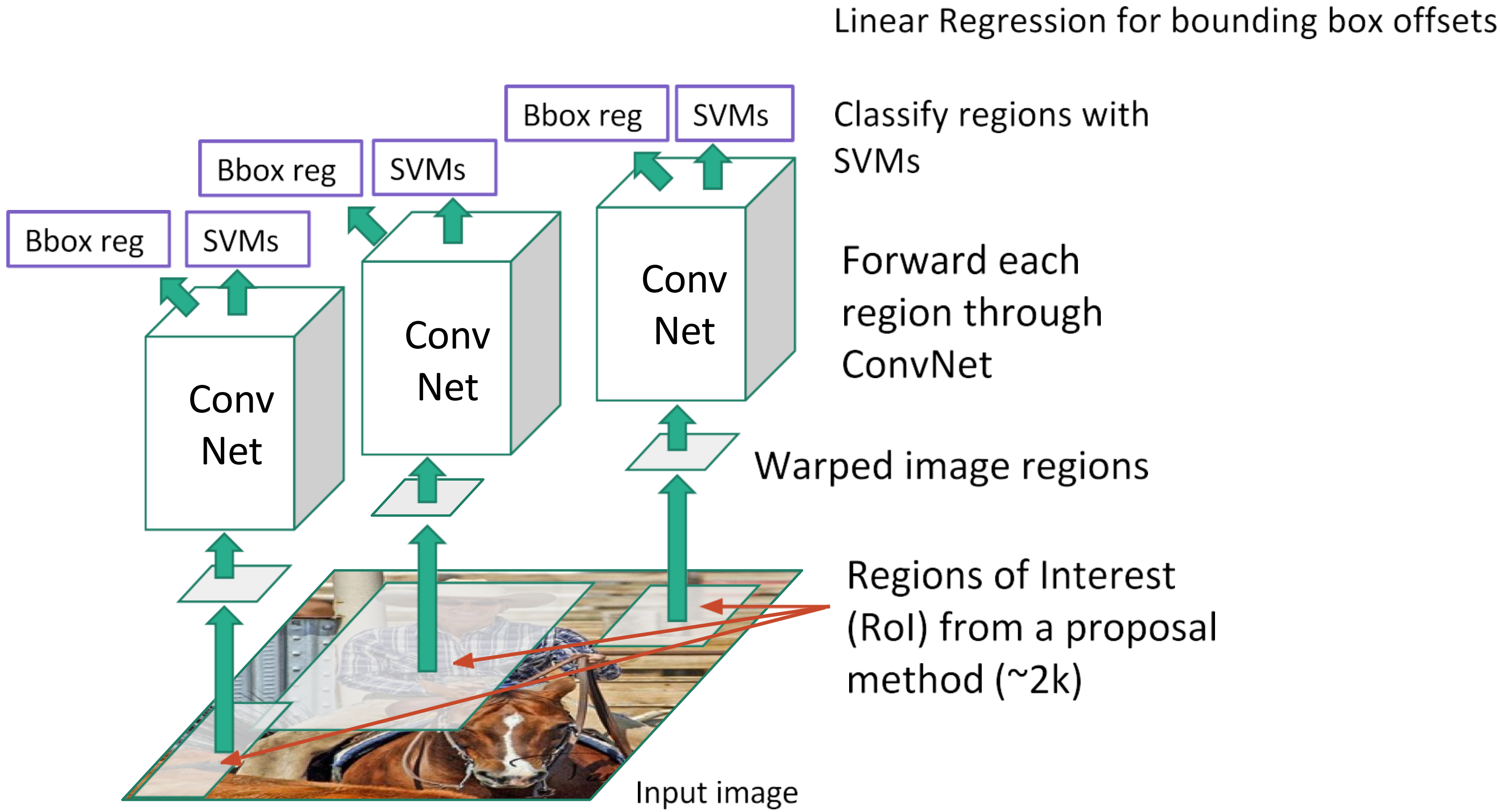
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN



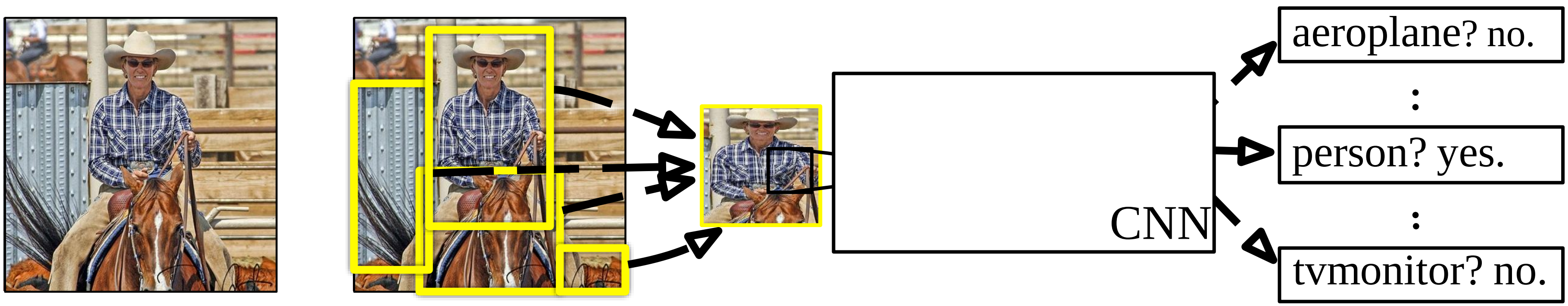
Girshick et al., “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”, CVPR 2014

R-CNN



Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN: Regions with CNN features



Input image

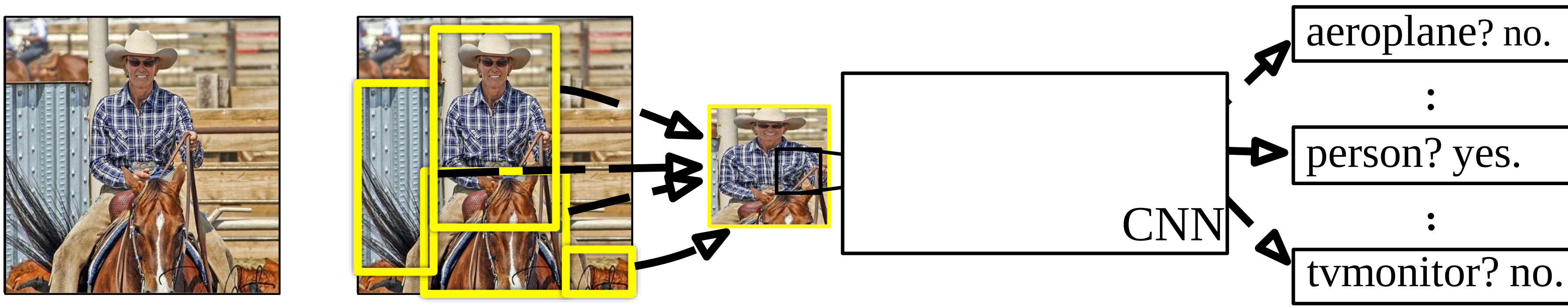
Extract region proposals (~2k / image)

Compute CNN features

Classify regions (linear SVM)

Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN at test time: Step 1

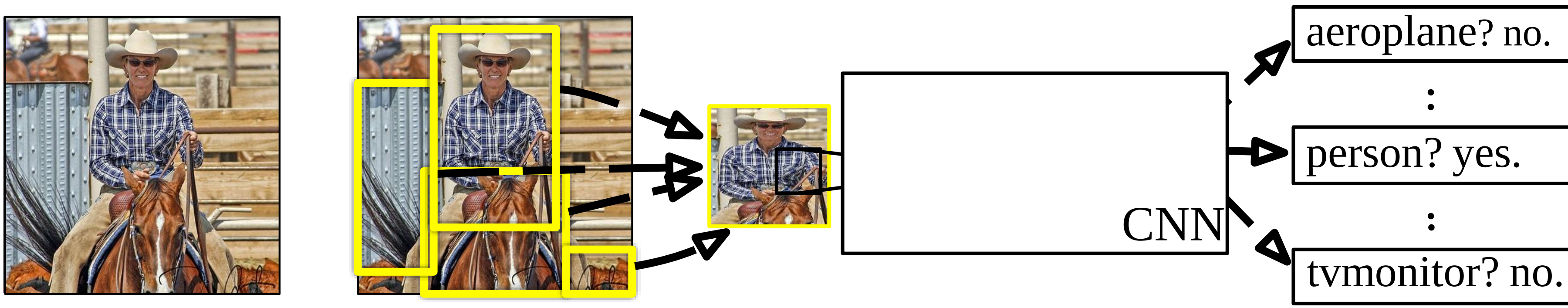


Input image → Extract region proposals (~2k / image)

Proposal-method agnostic, many choices
- Selective Search [van de Sande, Uijlings et al.] (Used in this work)
- Objectness [Alexe etal.]
- Category independent object proposals [Endres & Hoiem]
- CPMC [Carreira & Sminchisescu]

Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

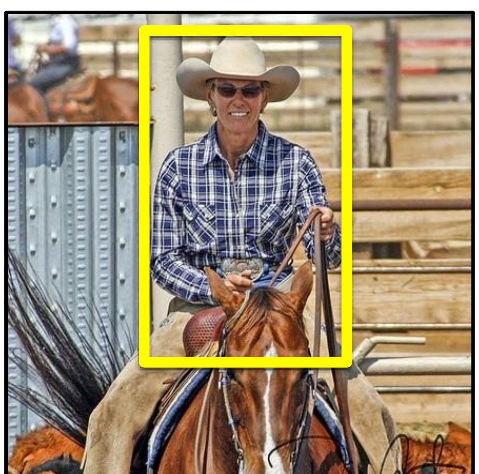
R-CNN at test time: Step 2



Input image

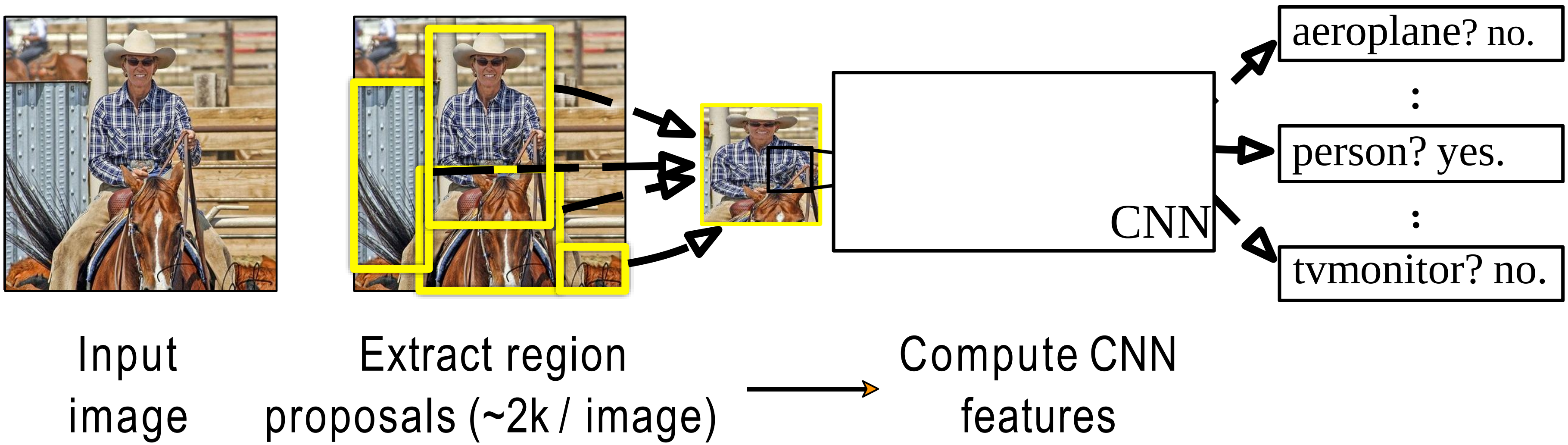
Extract region proposals (~2k / image)

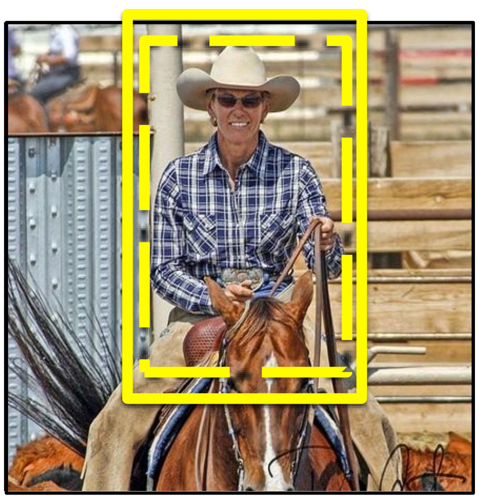
Compute CNN features



Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN at test time: Step 2

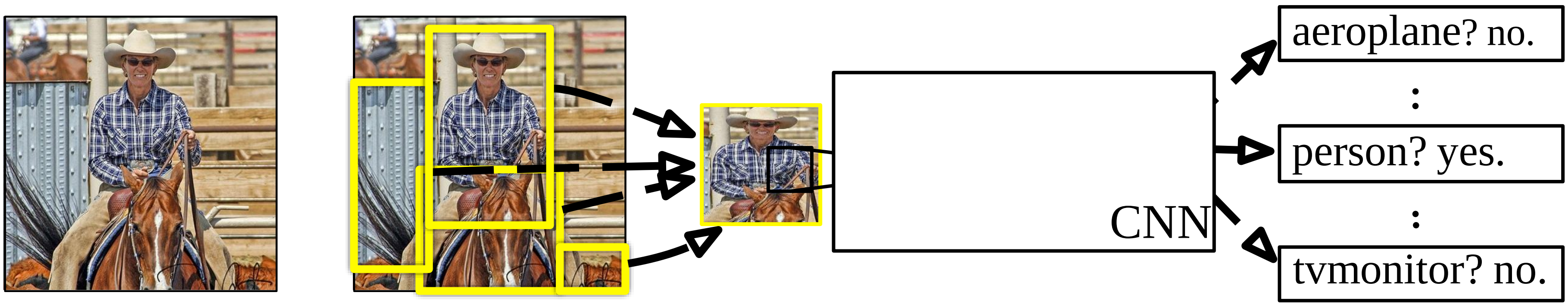




Dilate proposal

Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

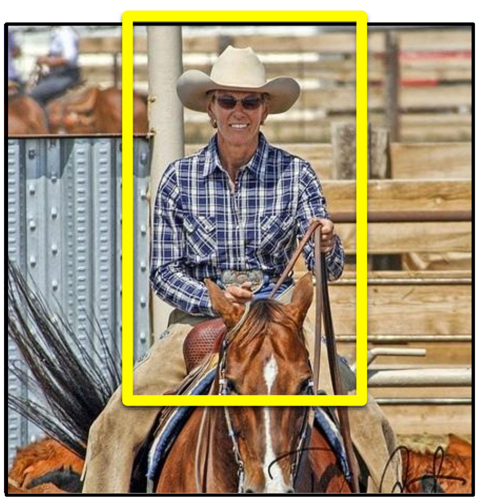
R-CNN at test time: Step 2



Input image

Extract region proposals (~2k / image)

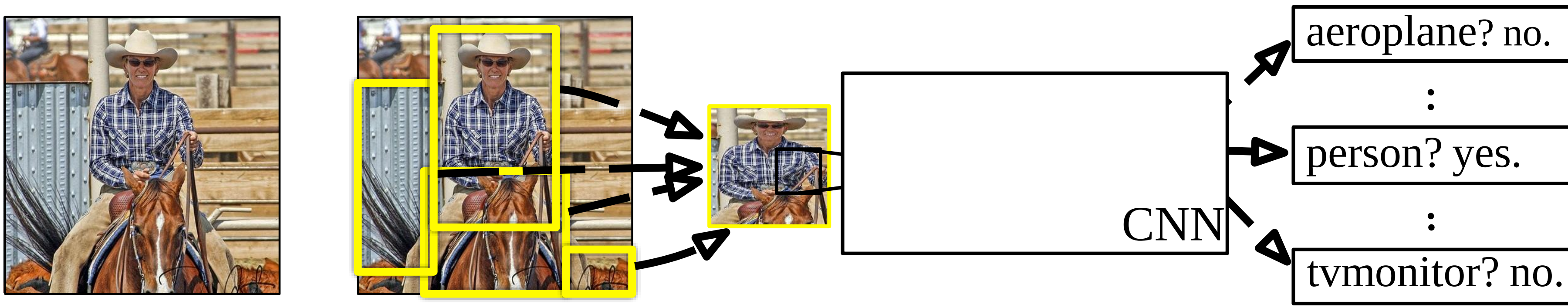
Compute CNN features



a. Crop

Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

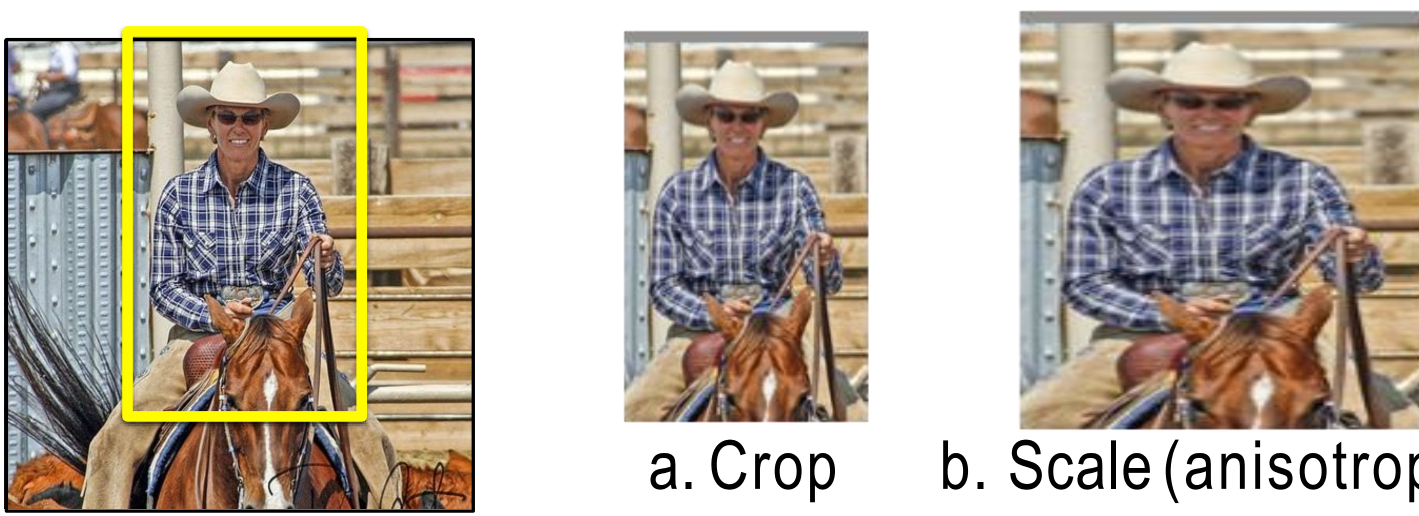
R-CNN at test time: Step 2



Input image

Extract region proposals (~2k / image)

Compute CNN features



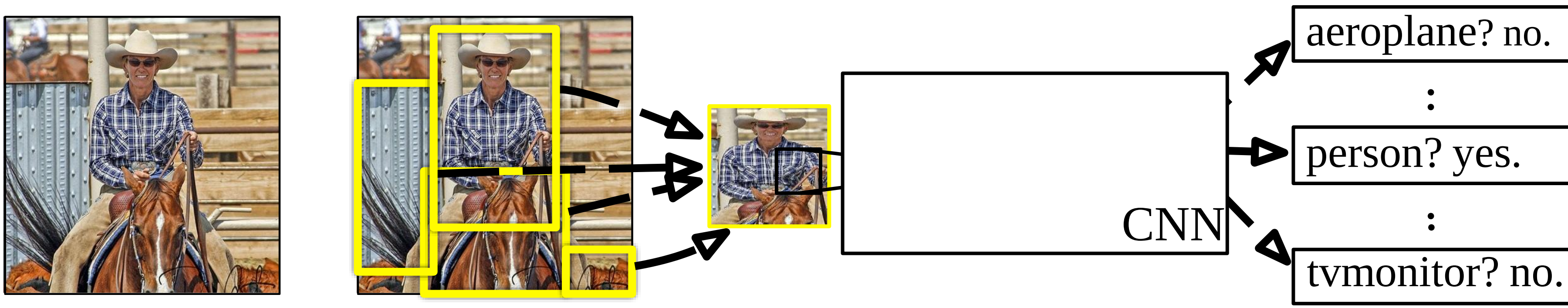
a. Crop

b. Scale (anisotropic)

227 x 227

Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

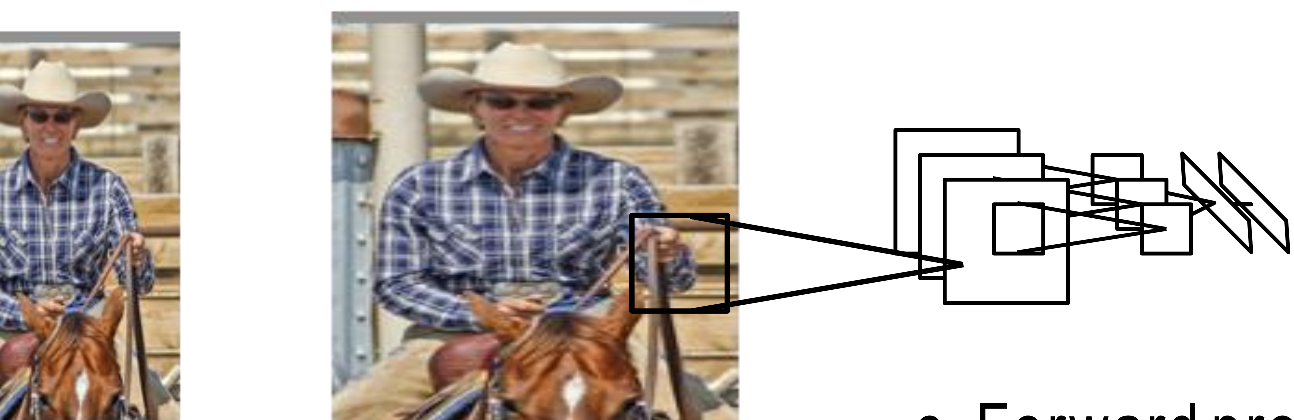
R-CNN at test time: Step 2



Input image

Extract region proposals (~2k / image)

Compute CNN features



. Crop

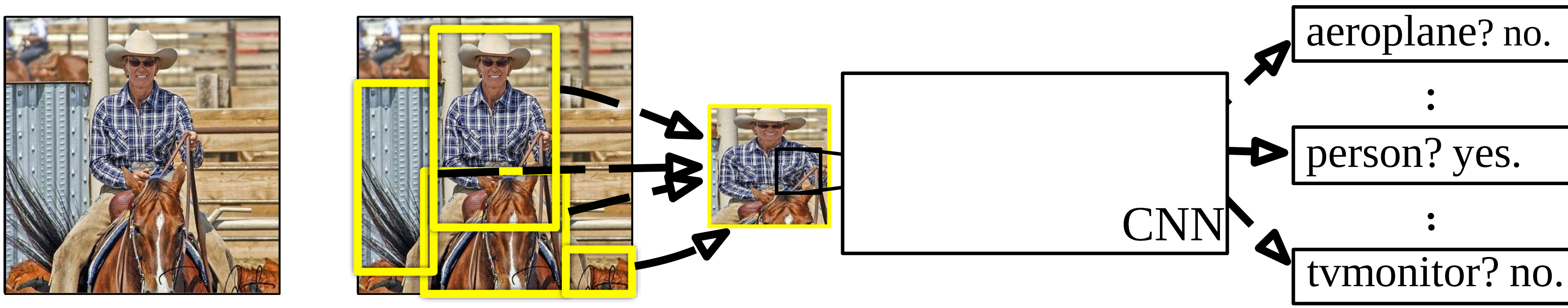
b. Scale (anisotropic)

c. Forward propagate

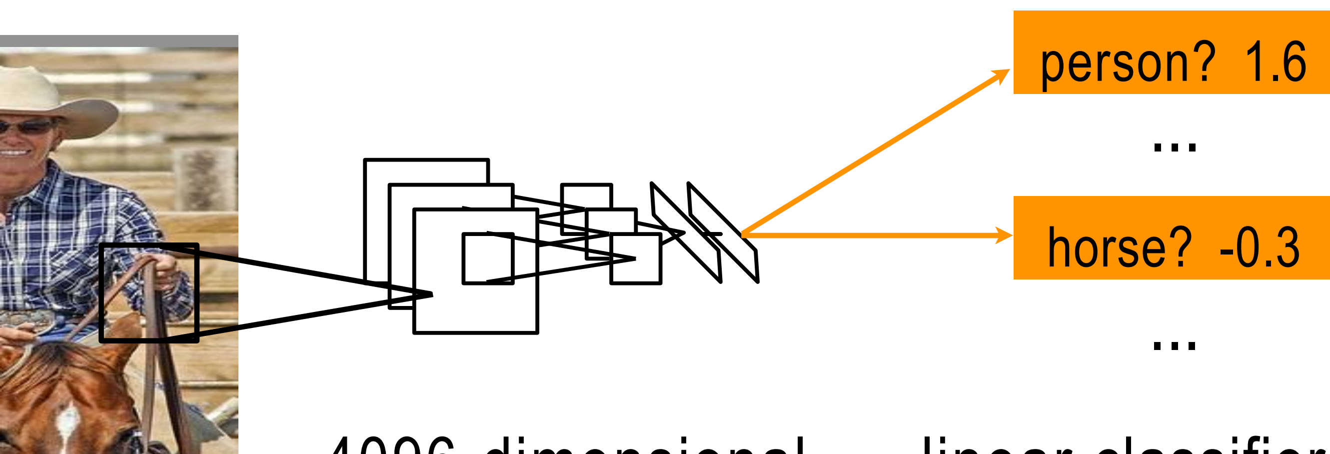
Output: “fc_7” features

Girshick et al., “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”, CVPR 2014

R-CNN at test time: Step 3



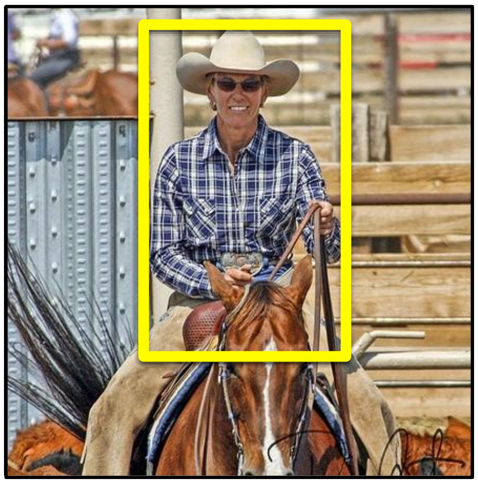
Input image | Extract region proposals (~2k / image) | Compute CNN features → Classify regions



proposal | 4096-dimensional fc_7 feature vector | linear classifiers (SVM or softmax)

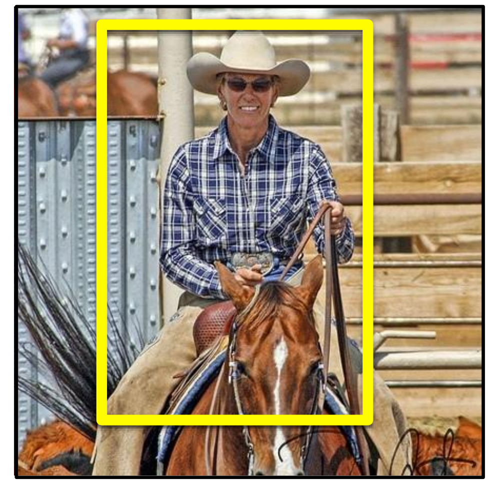
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

Step 4: Object proposal refinement



Original
proposal

Linear regression
on CNN features

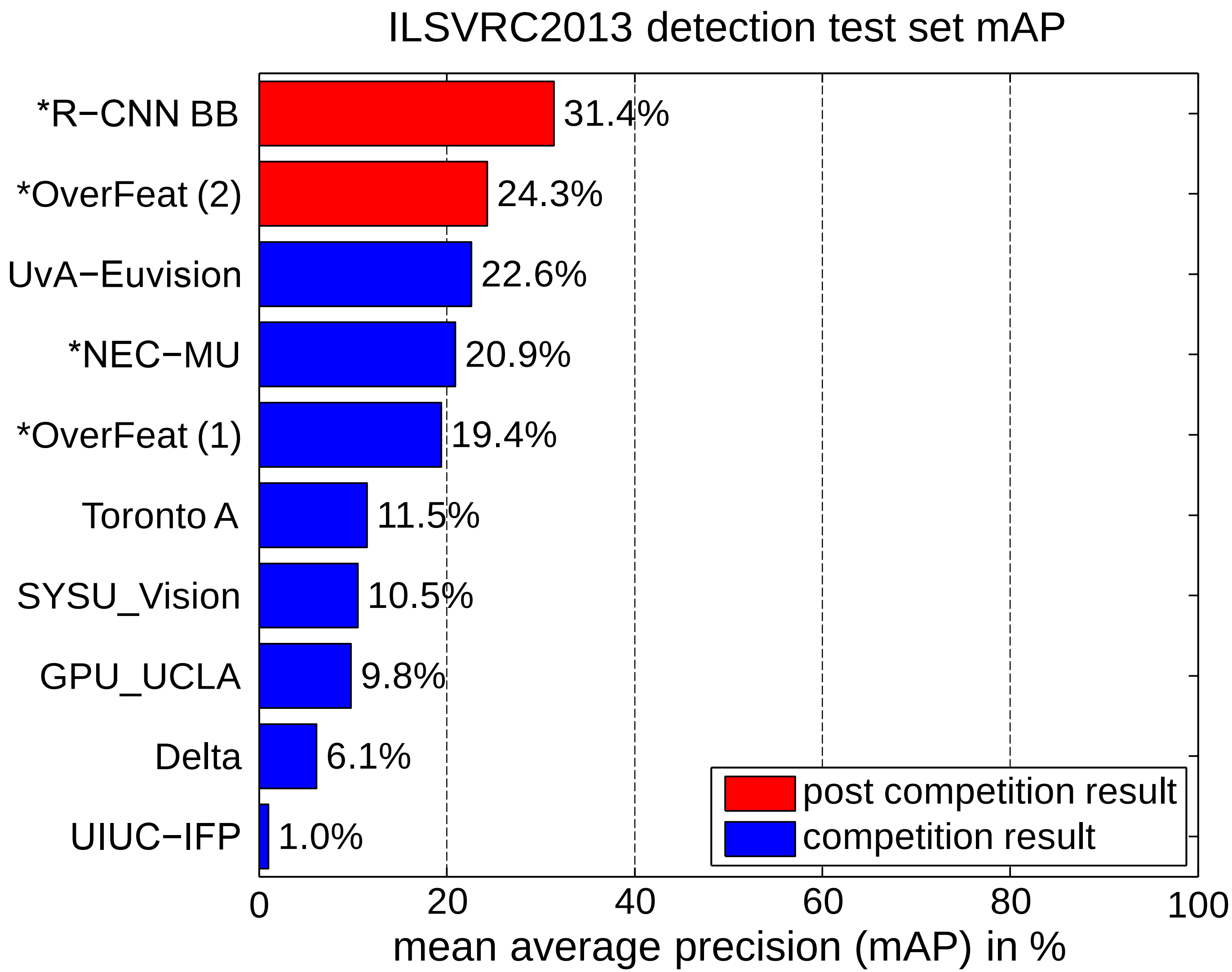


Predicted
object bounding box

Bounding-box regression

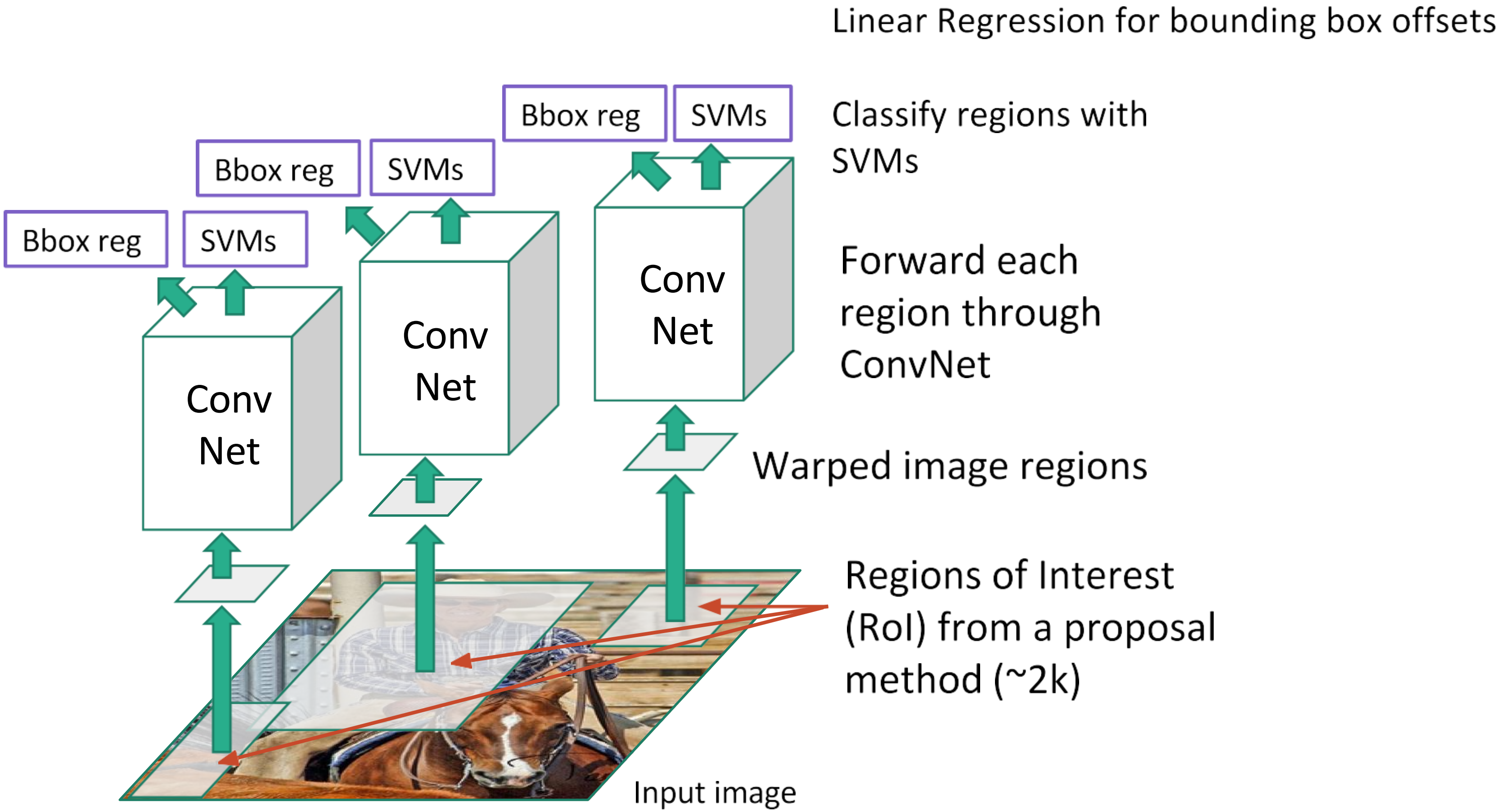
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN on ImageNet detection



Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN

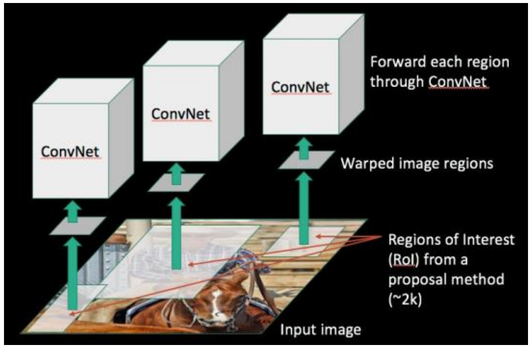


Post hoc component

Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

What's wrong with slow R-CNN?

- Ad-hoc training objectives
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressions (L2 loss)
- Training is slow (84h), takes a lot of disk space
  - Need to store all region crops
- Inference (detection) is slow
  - 47s / image with VGG16 [Simonyan & Zisserman, ICLR15]



~2000 ConvNet forward passes per image

Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN

- One network, applied one time, not 2000 times
- Trained end-to-end (in one stage)
- Fast test time
- Higher mean average precision

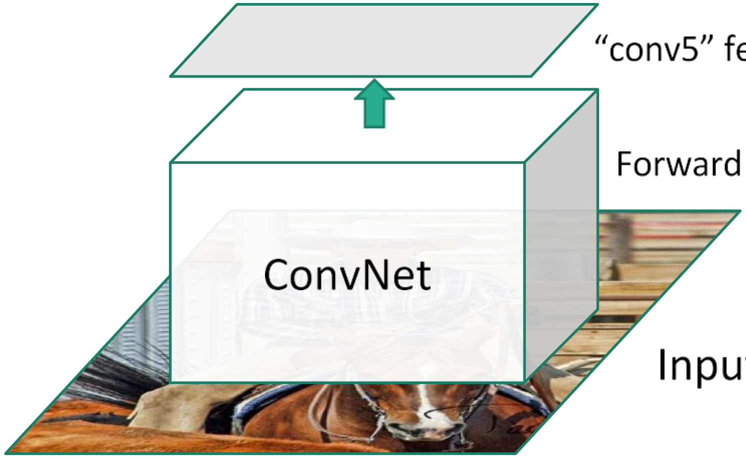
Adapted from Girshick, “Fast R-CNN”, ICCV 2015

Fast R-CNN

Input image

Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



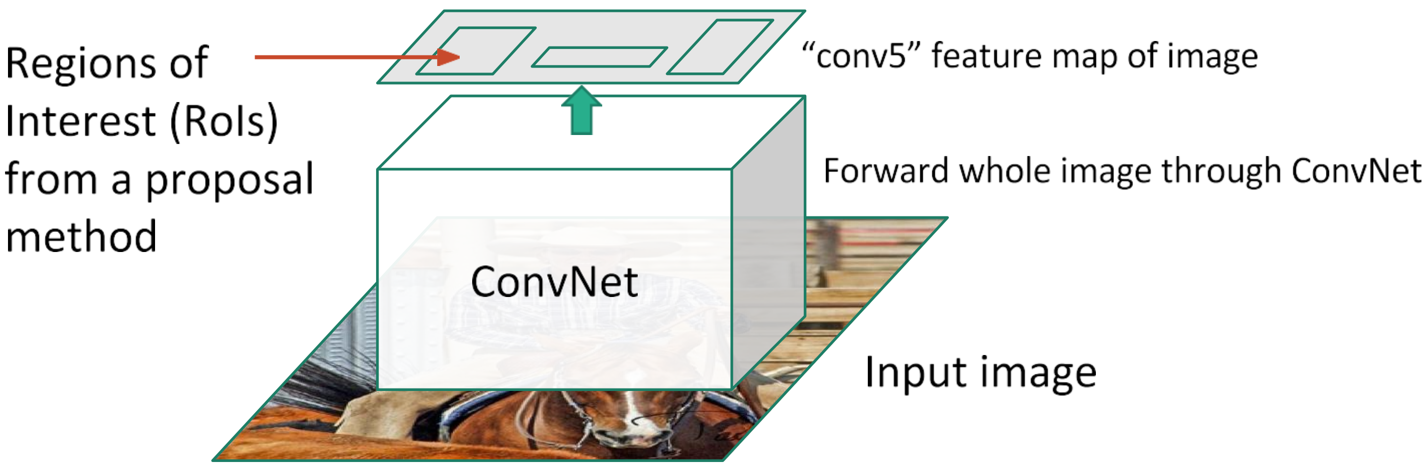
"conv5" feature map of image

Forward whole image through ConvNet

Input image

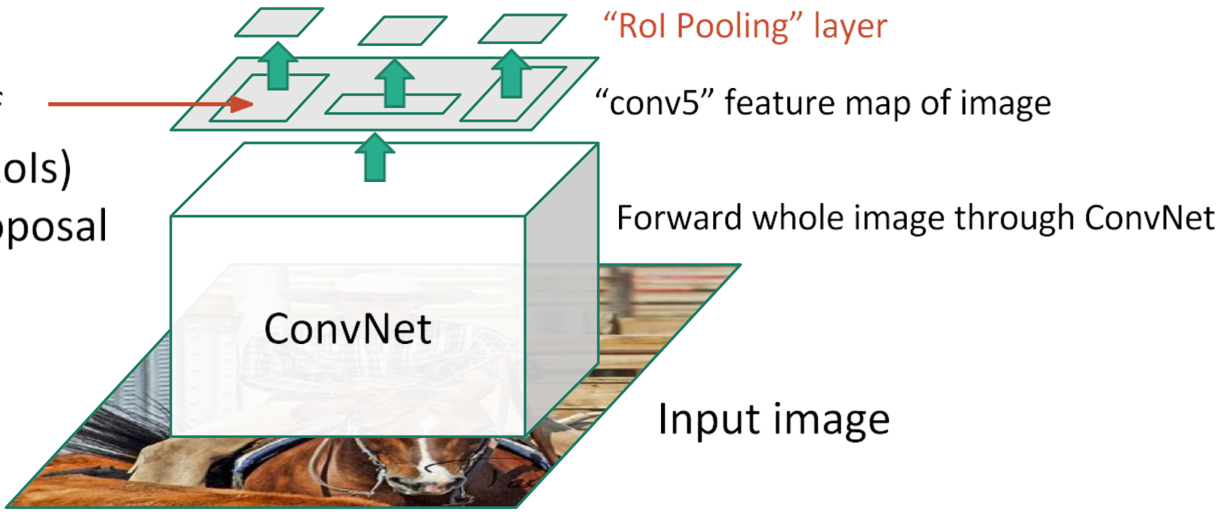
Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



Girshick, "Fast R-CNN", ICCV 2015

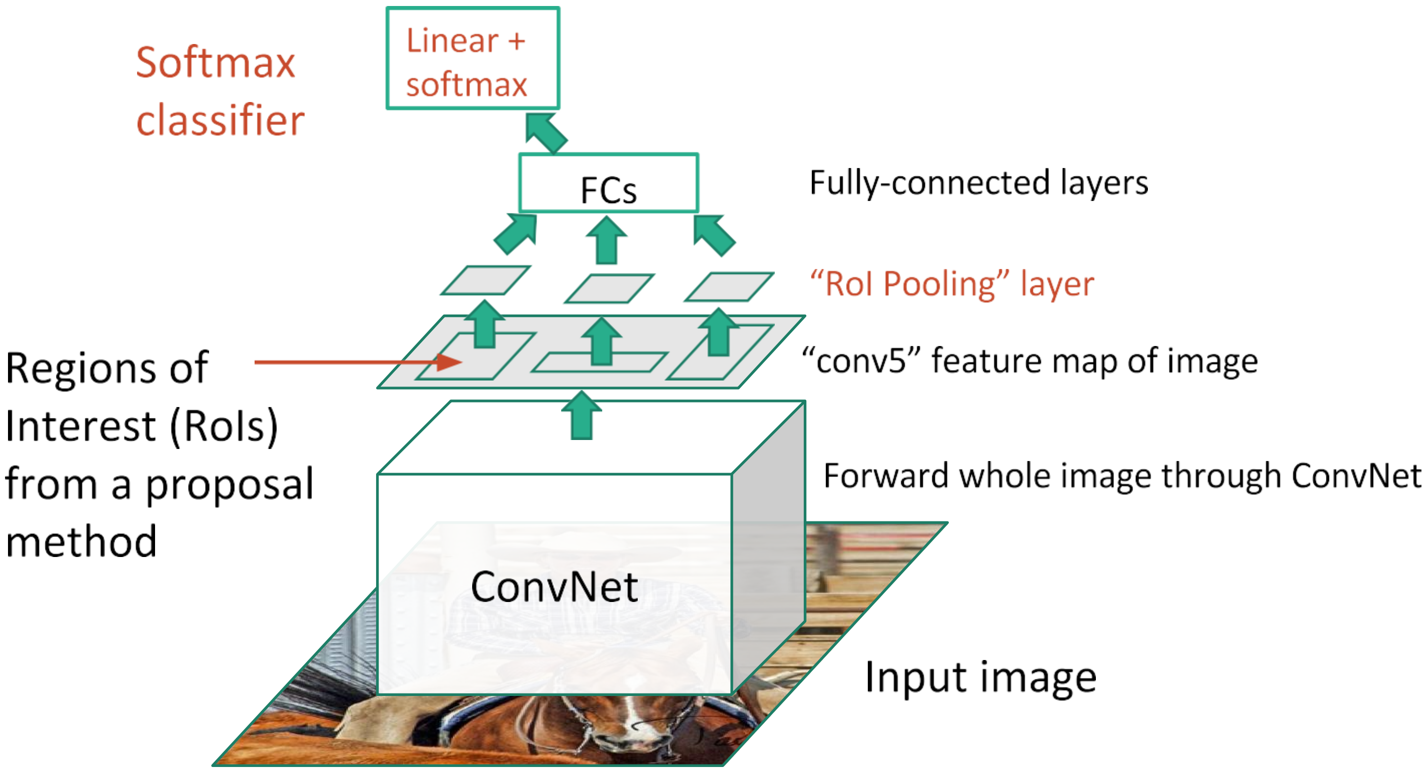
Fast R-CNN



Regions of Interest (RoIs) from a proposal method

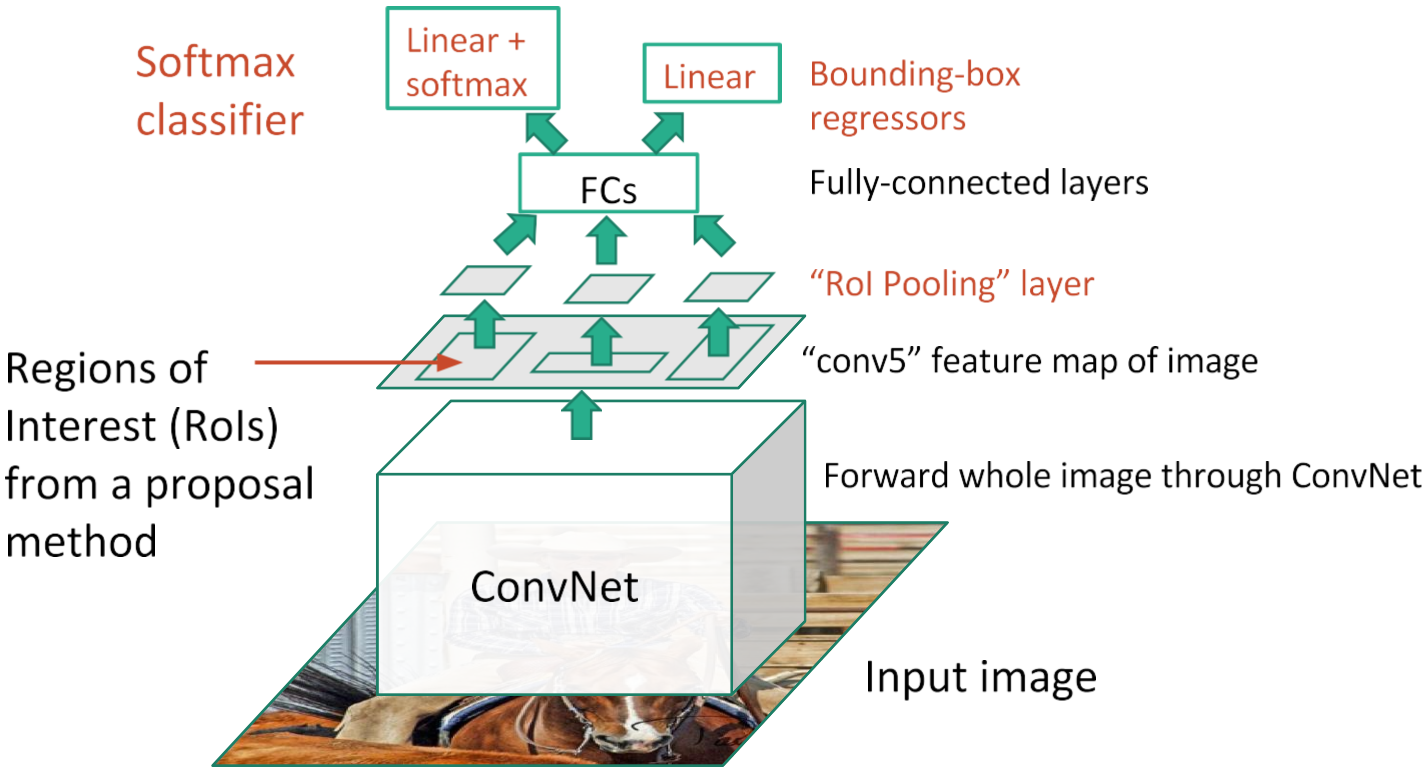
Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



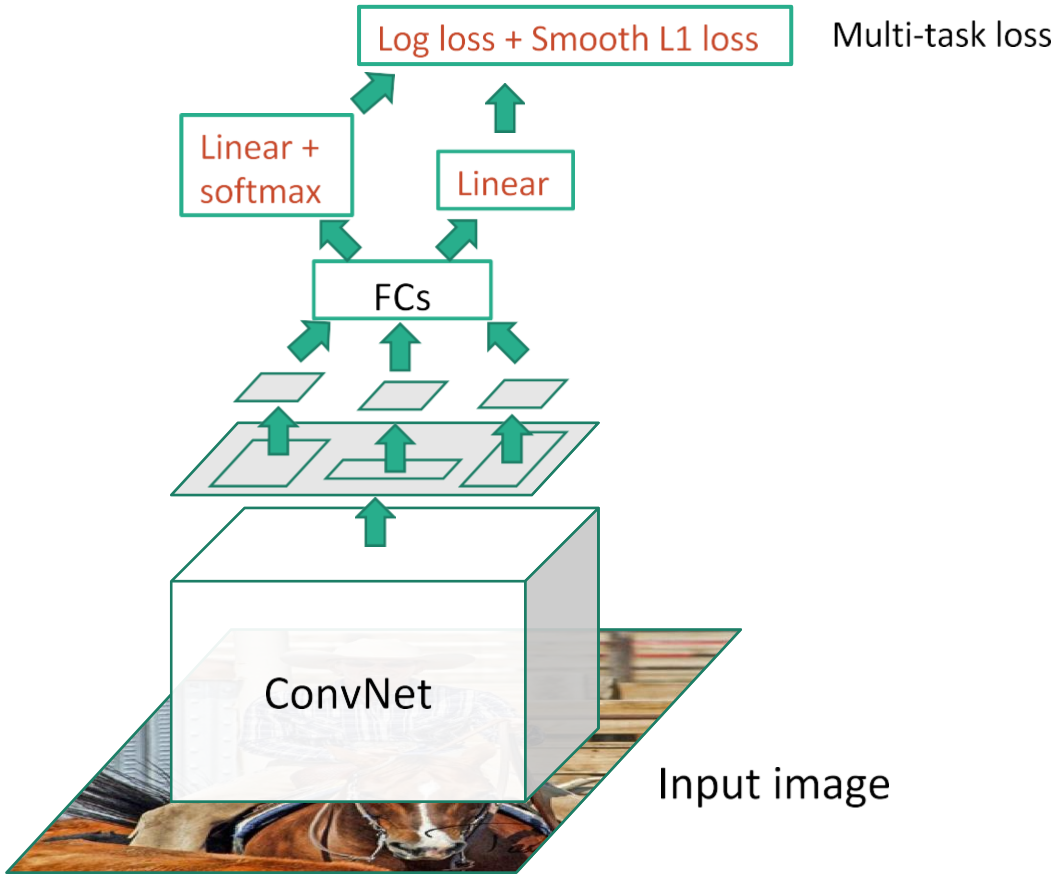
Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



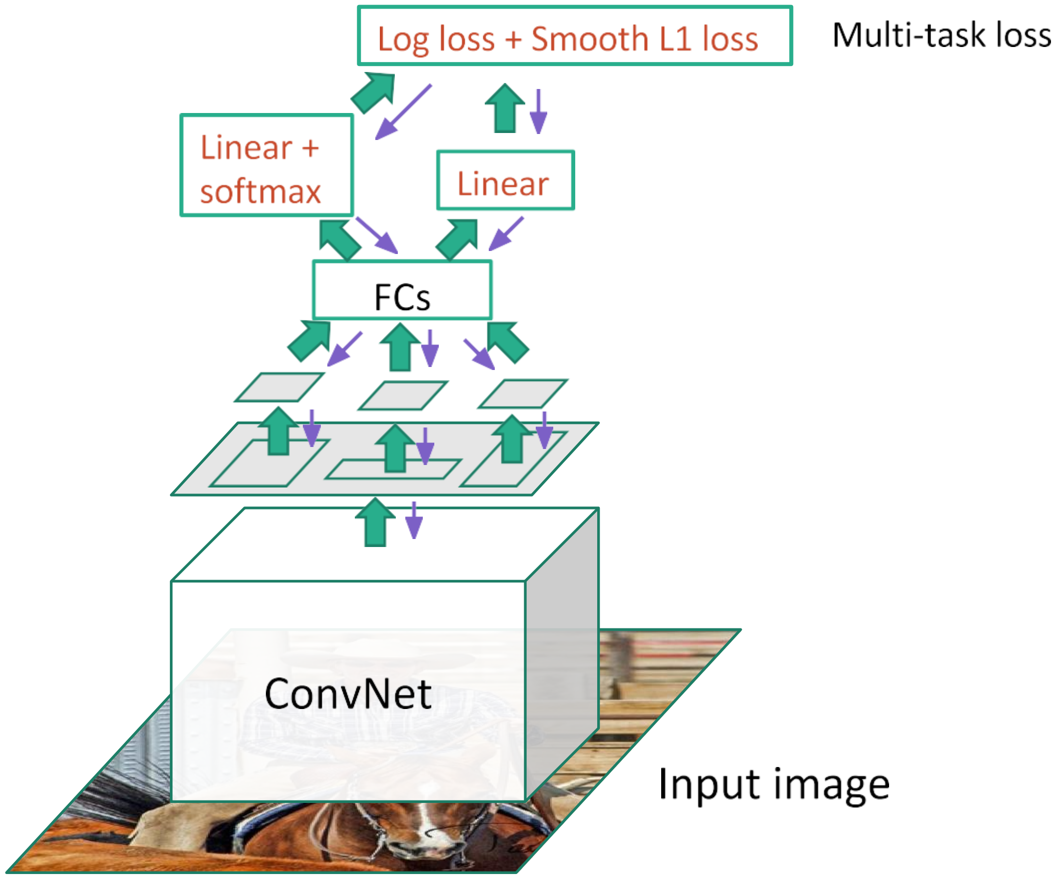
Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN (Training)



Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN (Training)



Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN vs R-CNN

	Fast R-CNN	R-CNN
Train time (h)	9.5	84
Speedup	8.8x	1x
Test time / image	0.32s	47.0s
Test speedup	146x	1x
mAP	66.9%	66.0%

Timings exclude object proposal time, which is equal for all methods.
All methods use VGG16 from Simonyan and Zisserman.

Girshick, "Fast R-CNN", ICCV 2015

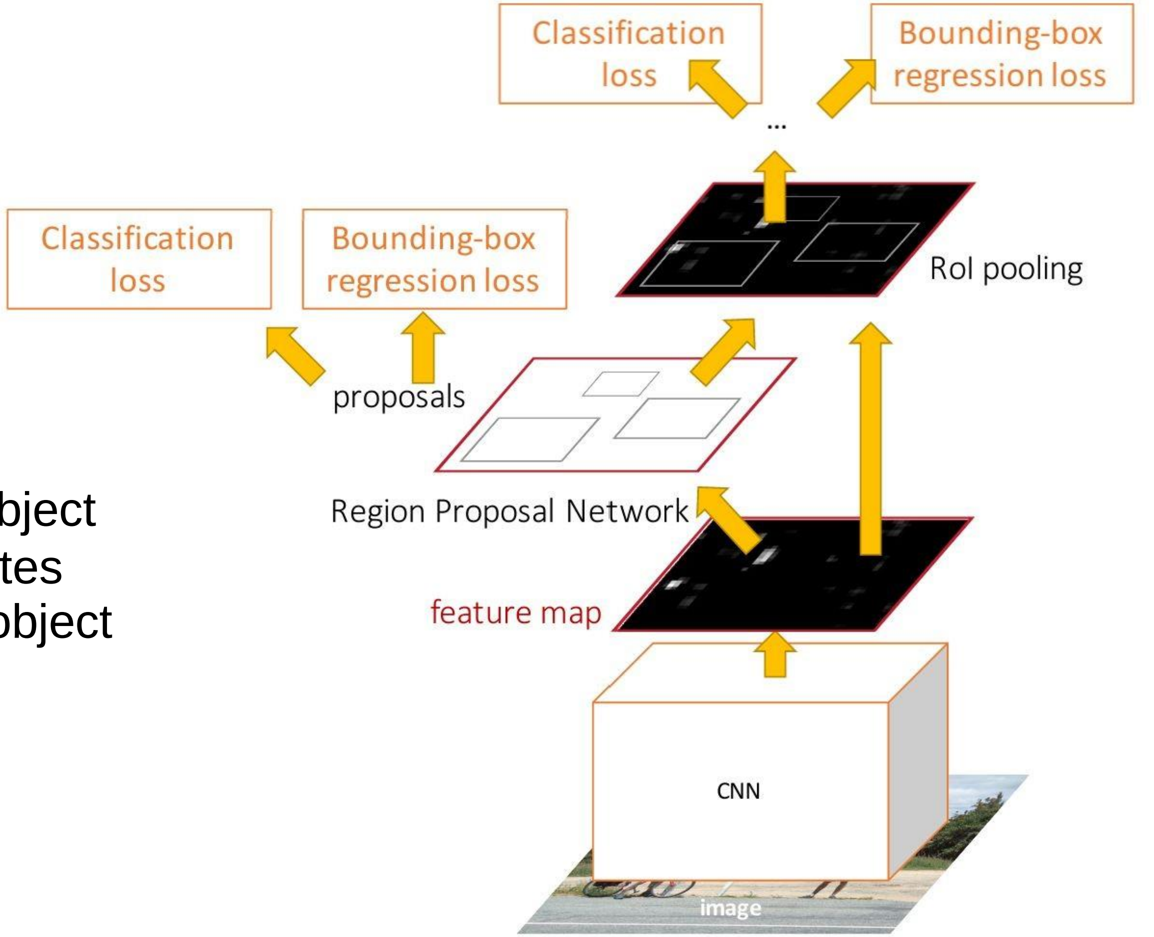
Faster R-CNN

Make CNN do proposals!

Insert **Region Proposal Network (RPN)** to predict proposals from features

Jointly train with 4 losses:

1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates



Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Accurate object detection is slow!

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img

⅓ Mile, 1760 feet

Redmon et al., “You Only Look Once: Unified, Real-Time Object Detection”, CVPR 2016

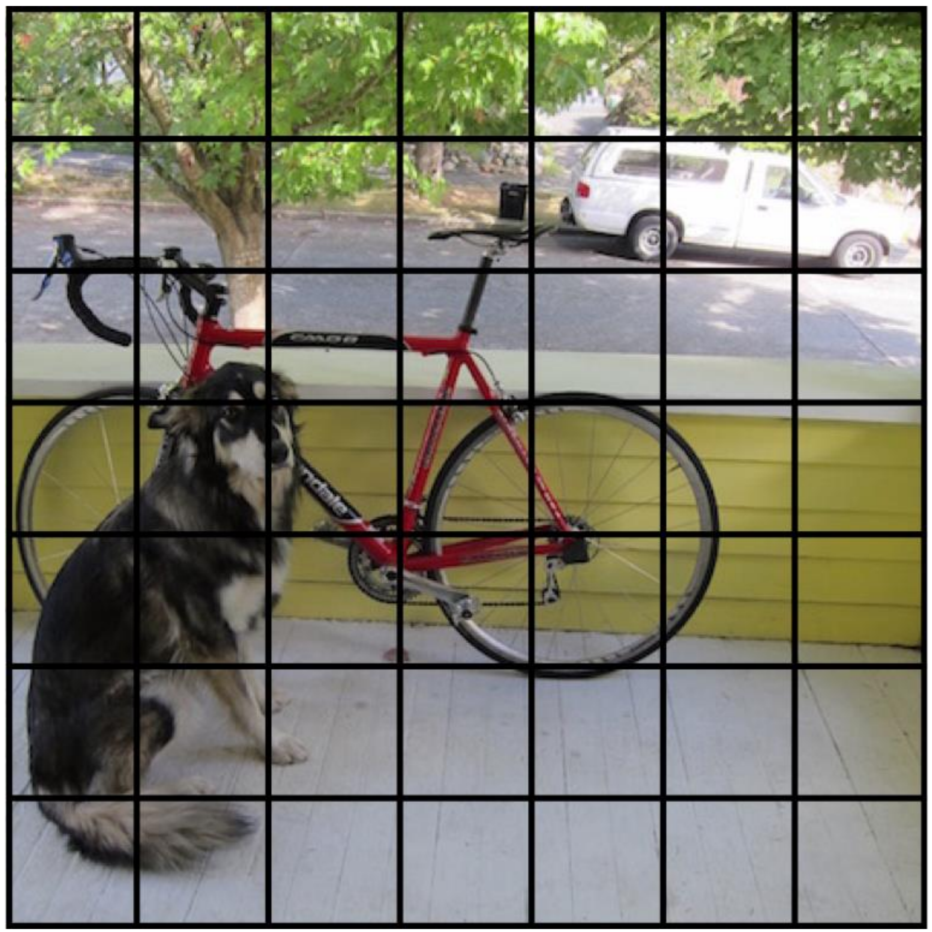
Accurate object detection is slow!

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	69.0	45 FPS	22 ms/img

2 feet

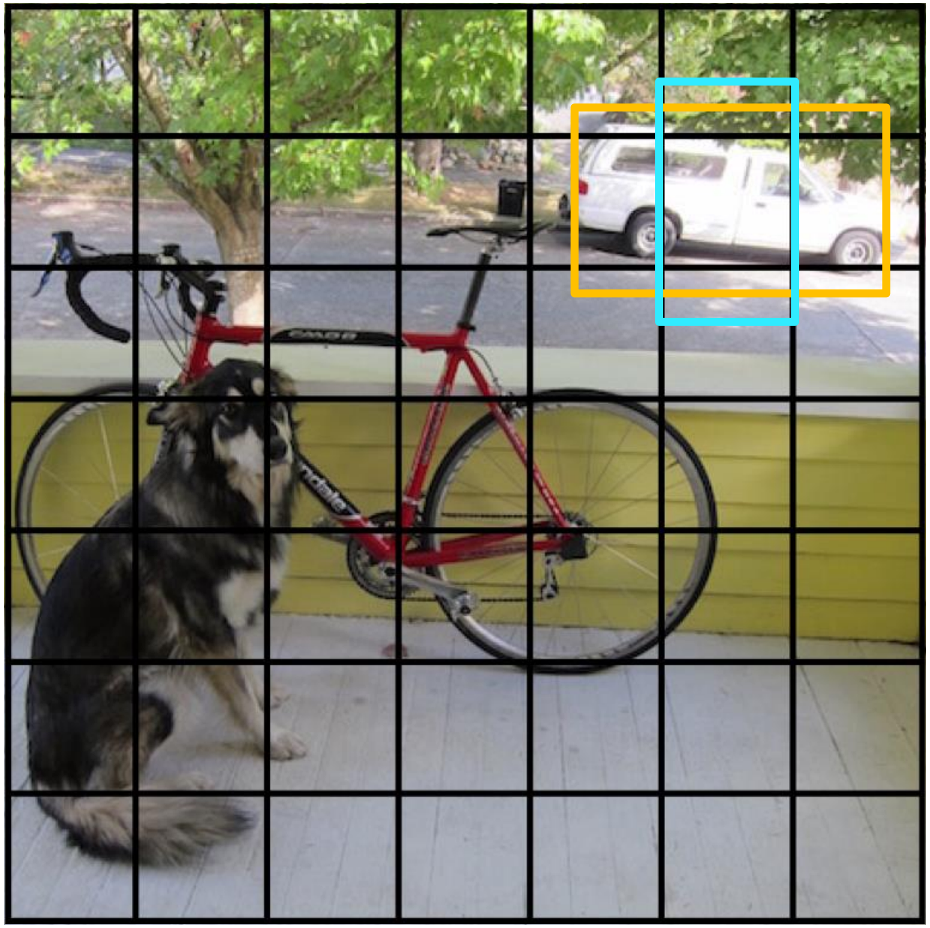
Redmon et al., *“You Only Look Once: Unified, Real-Time Object Detection”*, CVPR 2016

Detection without Proposals: YOLO



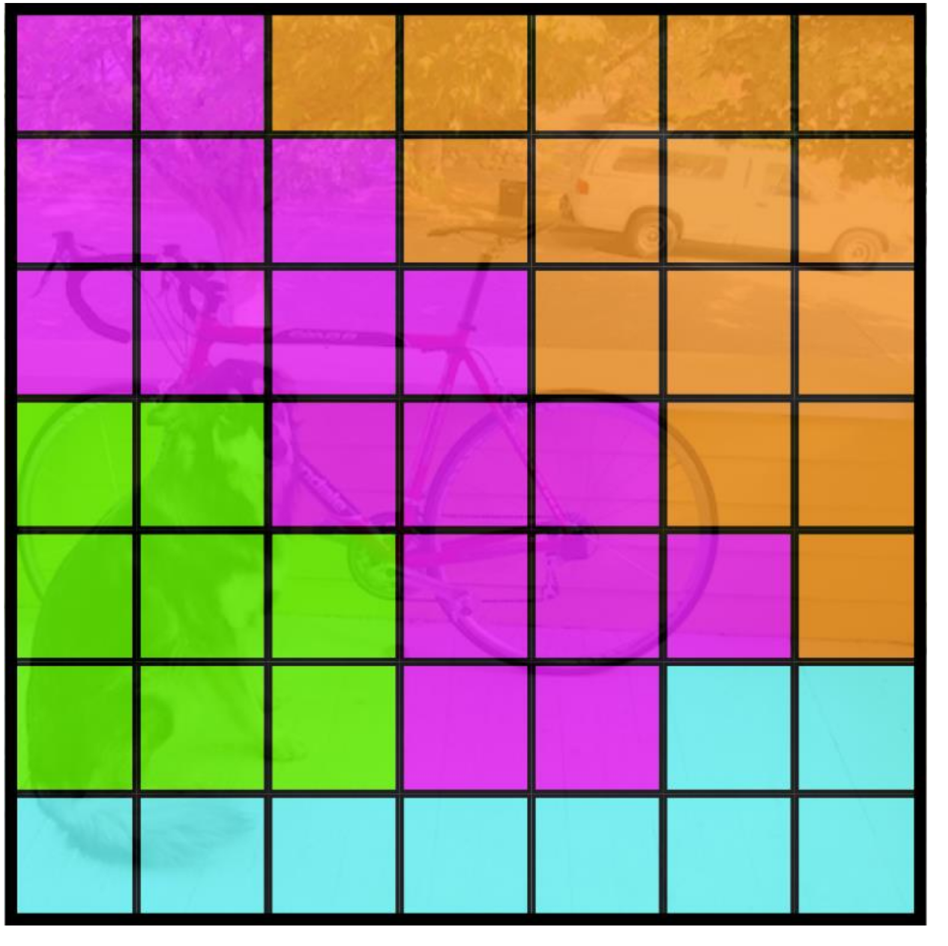
Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

Each cell predicts boxes and confidences: P(Object)



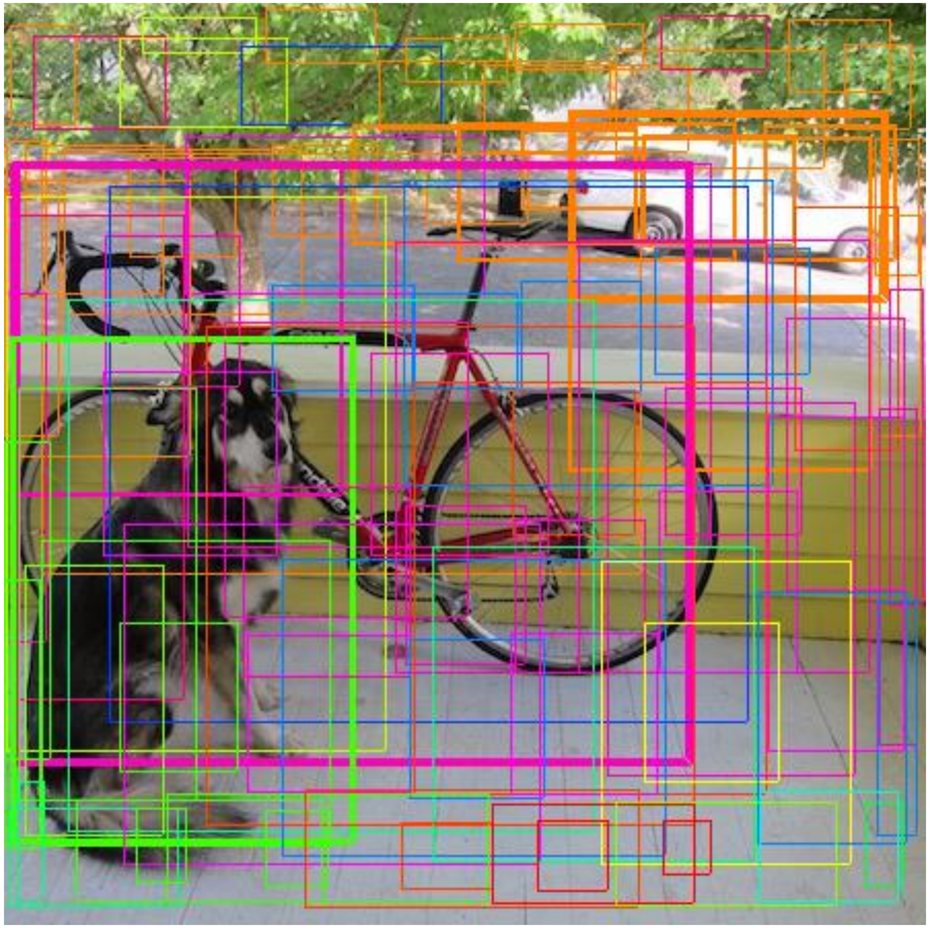
Redmon et al., “You Only Look Once: Unified, Real-Time Object Detection”, CVPR 2016

Each cell also predicts a probability P(Class | Object)



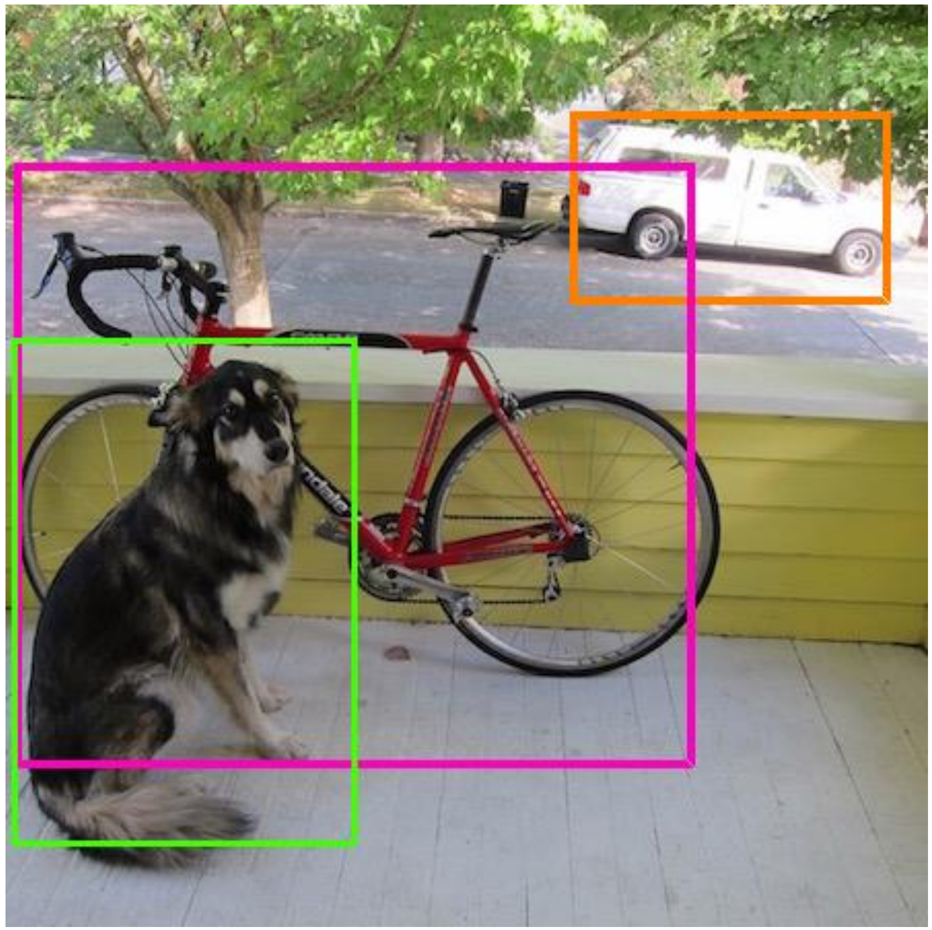
Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

Combine the box and class predictions



Redmon et al., *"You Only Look Once: Unified, Real-Time Object Detection"*, CVPR 2016

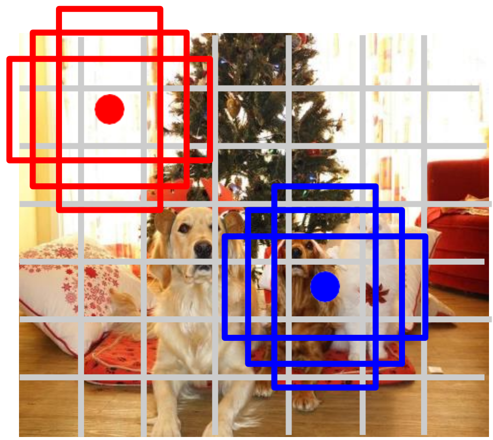
Finally do NMS and threshold detections



Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

Detection without Proposals: YOLO

Input image
3 x H x W



Divide image into grid
7 x 7

Image a set of **base boxes** centered at each grid cell
Here B = 3

Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers: (dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)

Output:
7 x 7 x (5 * B + C)

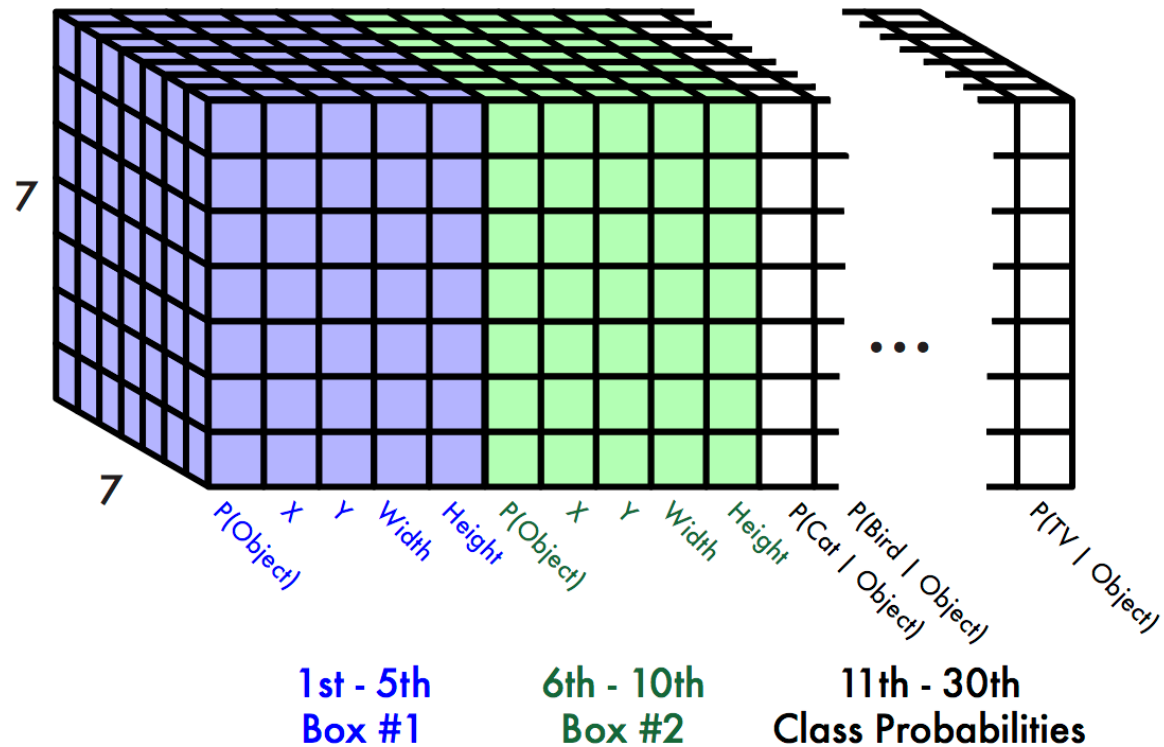
Redmon et al, "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016
Liu et al, "SSD: Single-Shot MultiBox Detector", ECCV 2016

Slide by: Justin Johnson

This parameterization fixes the output size

Each cell predicts:

- For each bounding box:
  - 4 coordinates (x, y, w, h)
  - 1 confidence value
- Some number of class probabilities



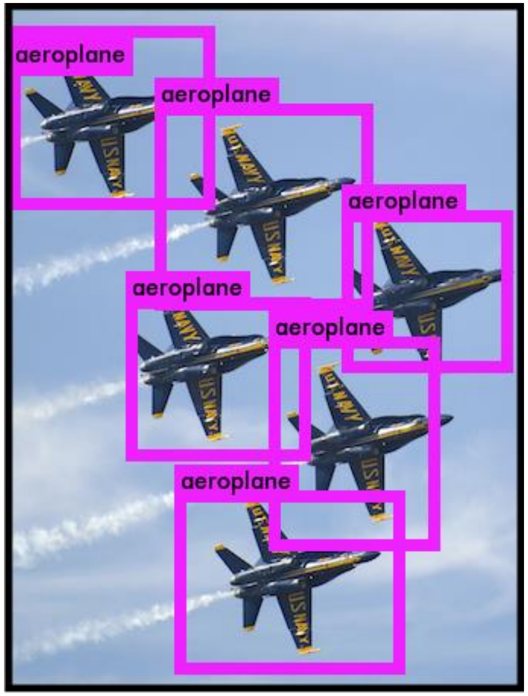
For Pascal VOC:

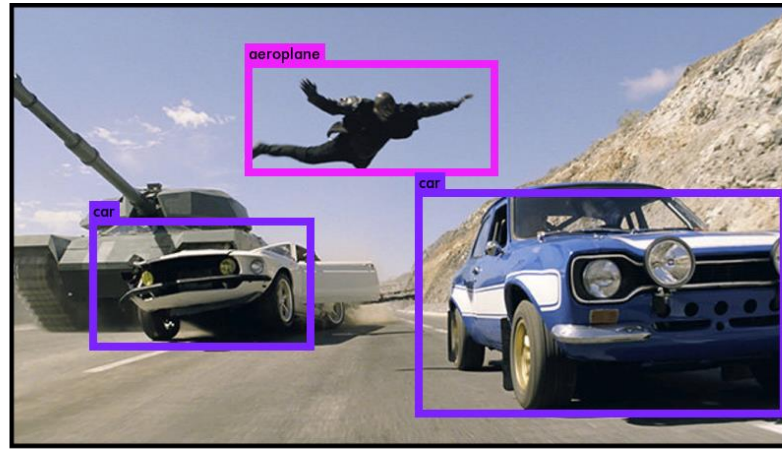
- 7x7 grid
- 2 bounding boxes / cell
- 20 classes

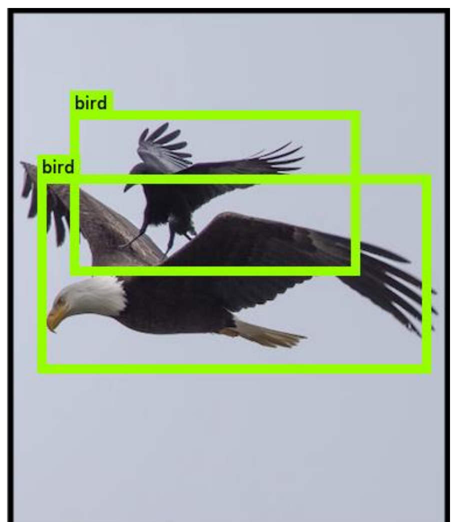
7 x 7 x (2 x 5 + 20) = 7 x 7 x 30 tensor = **1470 outputs**

Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

YOLO works across many natural images

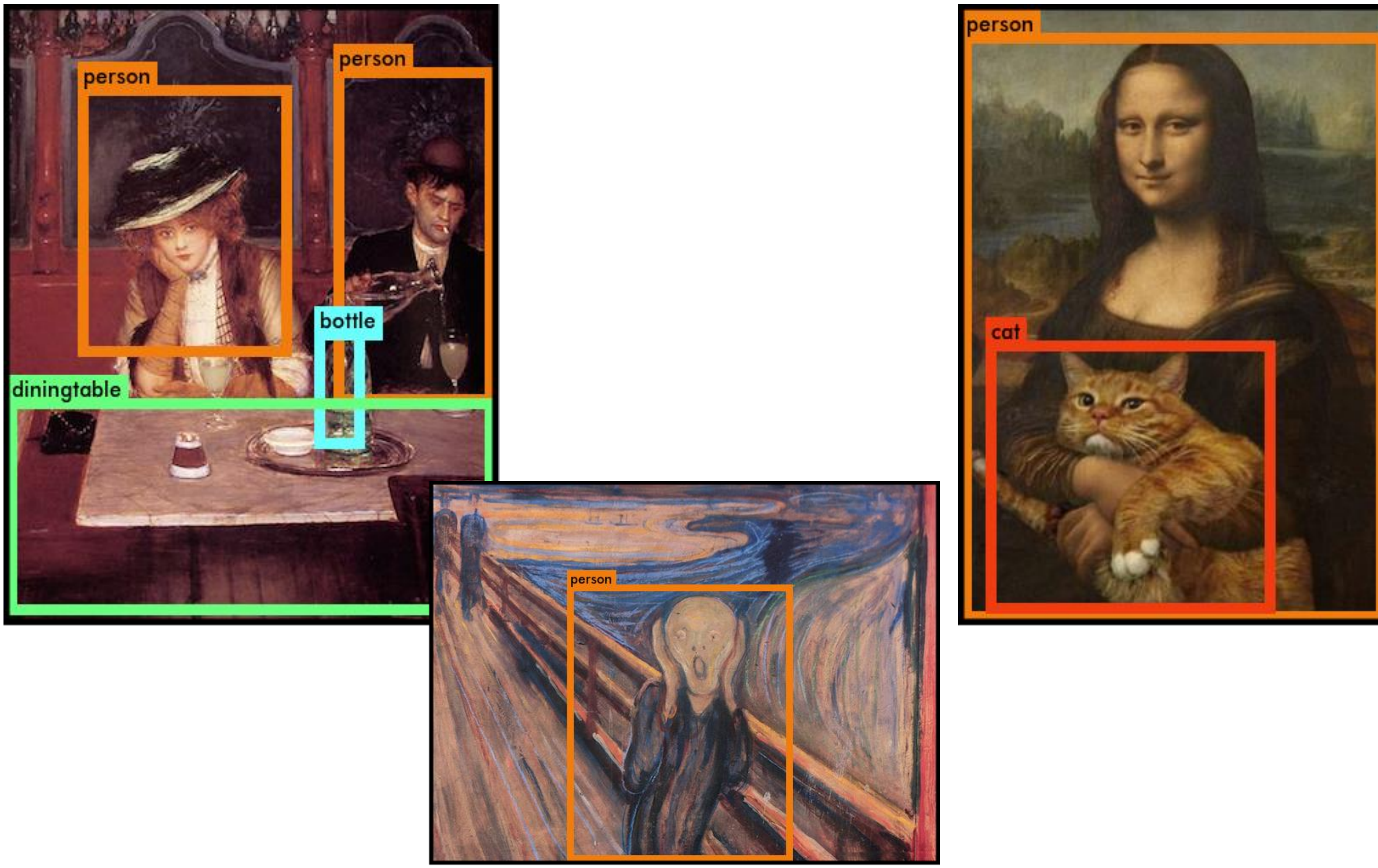






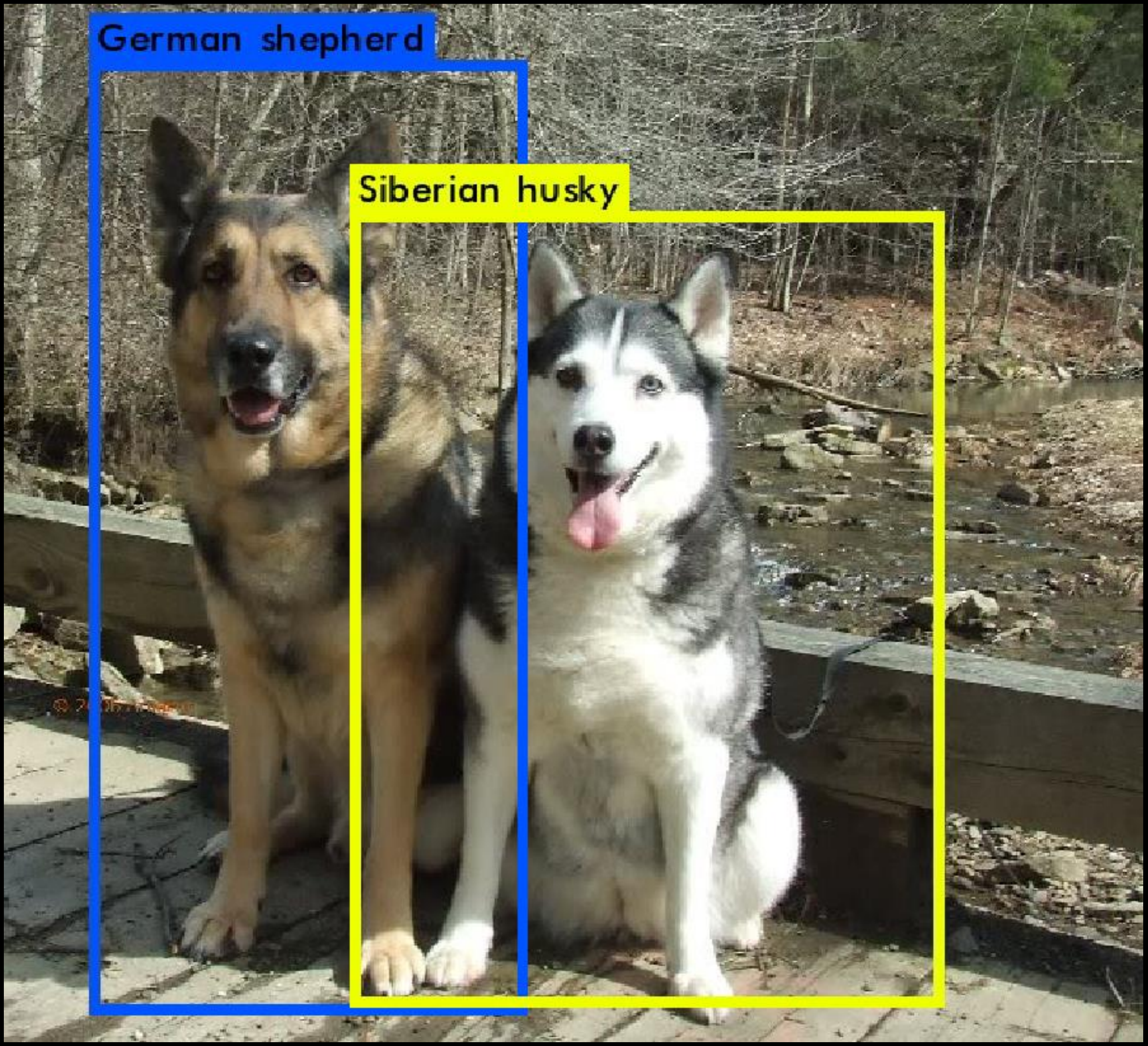
Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

It also generalizes well to new domains

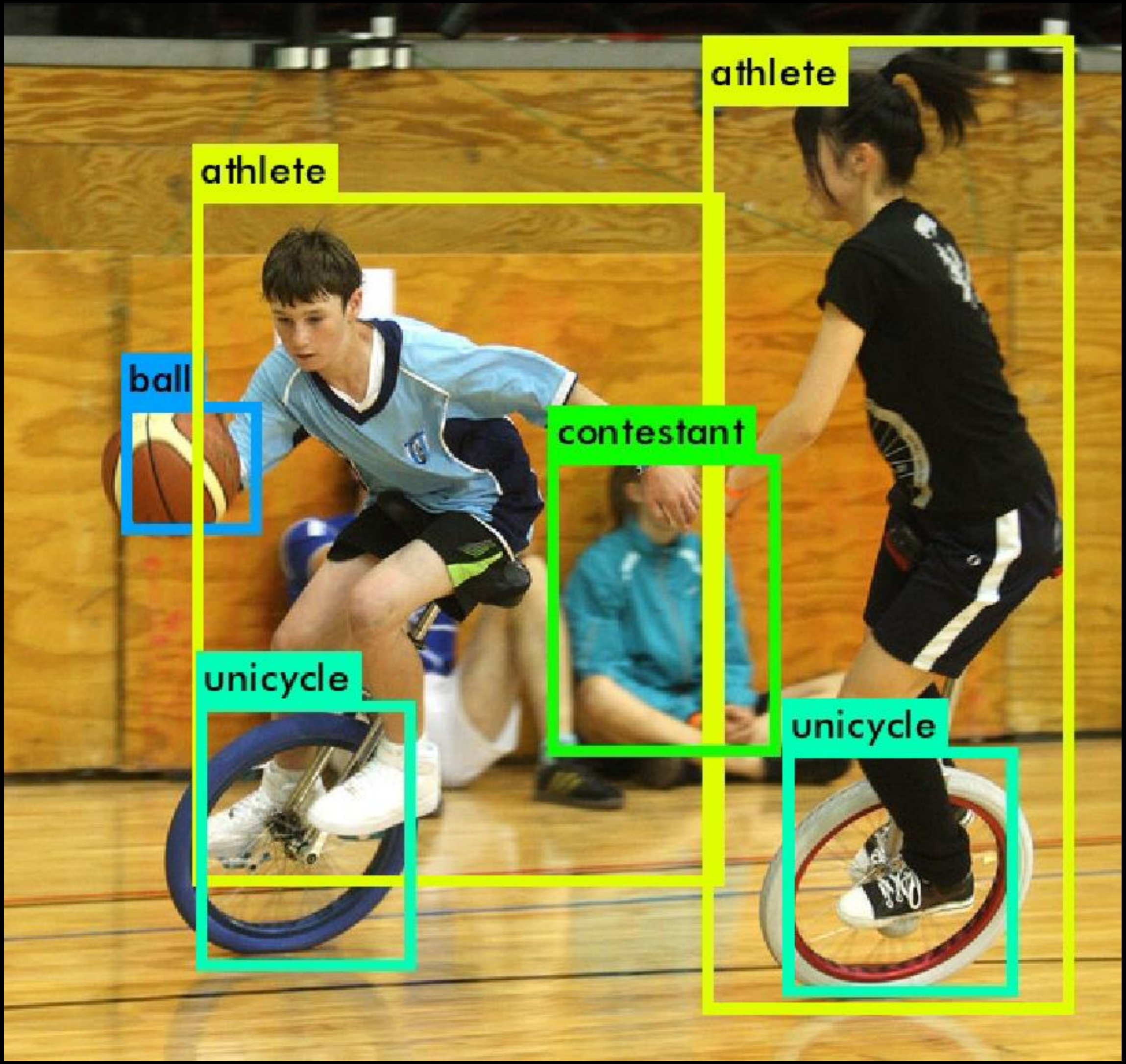


Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

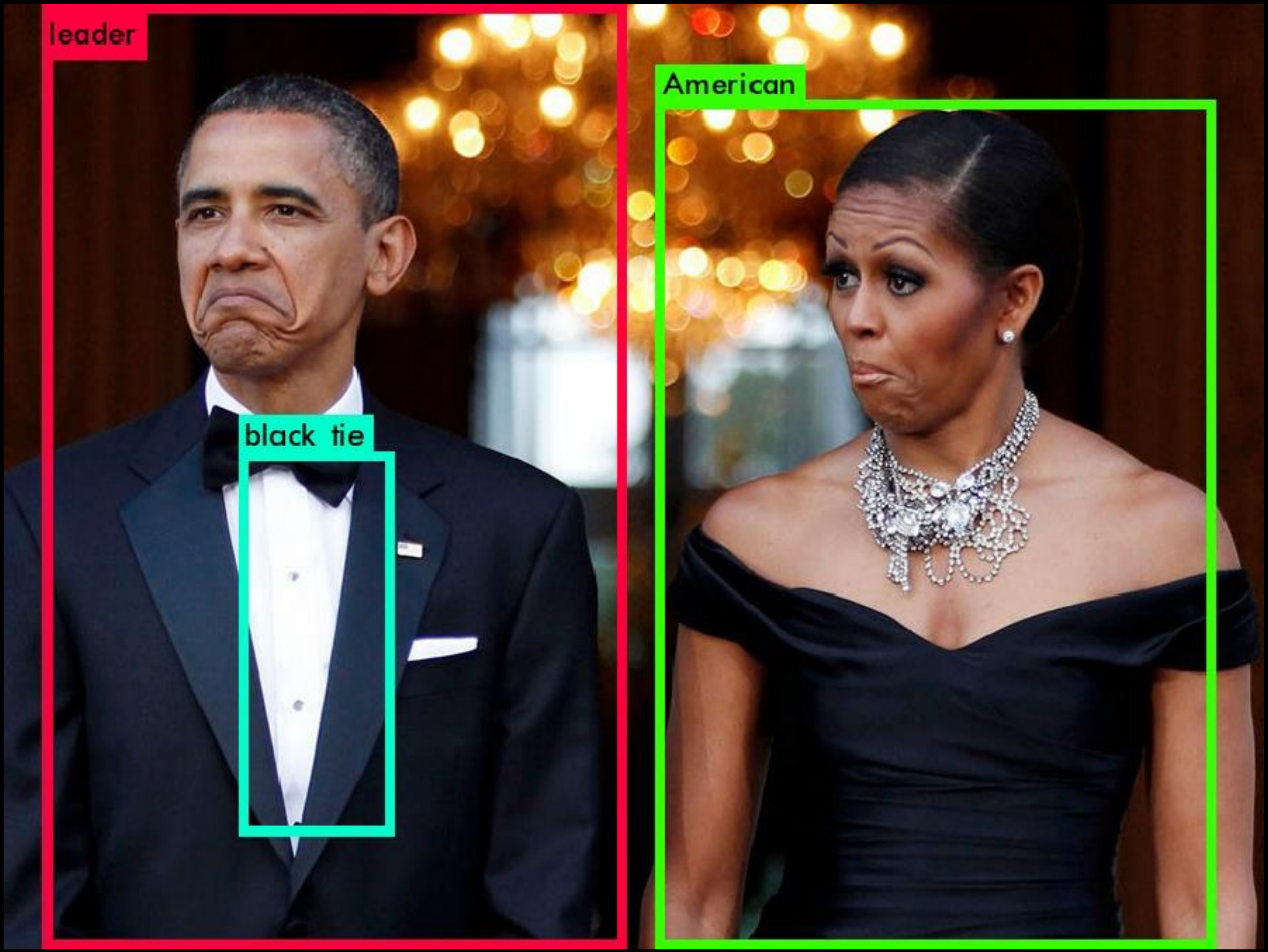
Redmon and Farhadi, “YOLO9000: Better, Faster, Stronger”, CVPR 2017



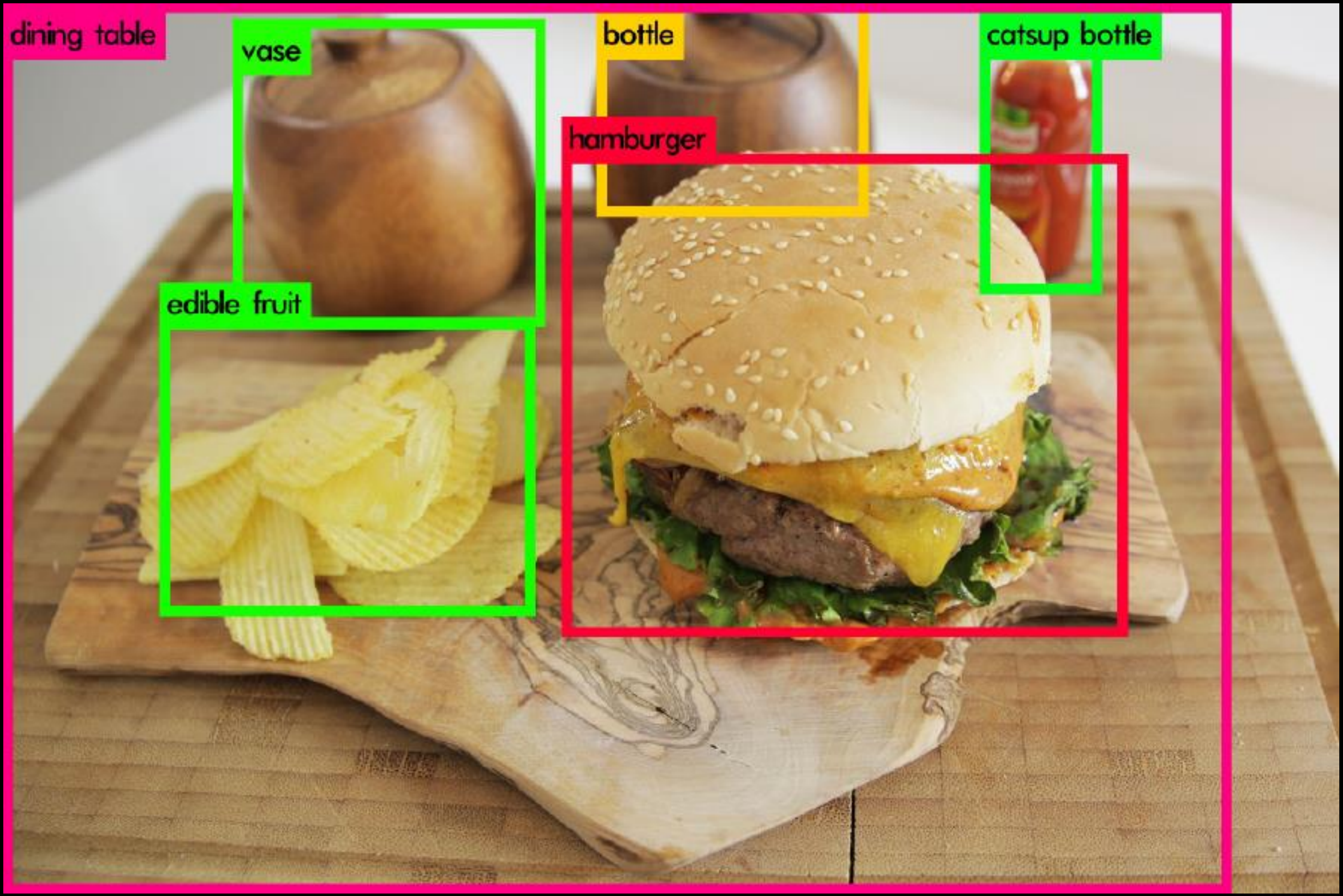
Redmon and Farhadi, “YOLO9000: Better, Faster, Stronger”, CVPR 2017



Redmon and Farhadi, "YOLO9000: Better, Faster, Stronger", CVPR 2017



Redmon and Farhadi, "YOLO9000: Better, Faster, Stronger", CVPR 2017



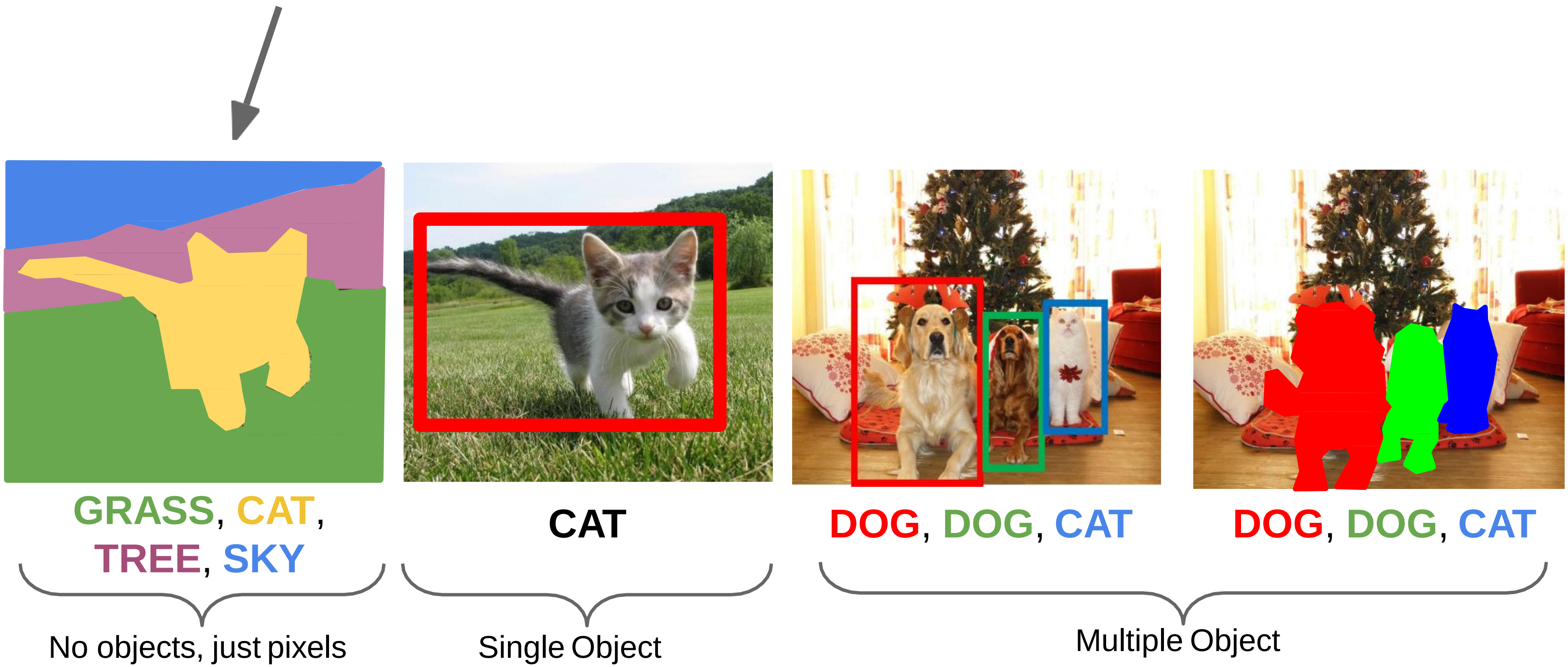
Redmon and Farhadi, "YOLO9000: Better, Faster, Stronger", CVPR 2017

Redmon and Farhadi, “YOLO9000: Better, Faster, Stronger”, CVPR 2017

Plan for the next two lectures

- Detection approaches
  - Pre-CNNs
    - Detection with whole windows: Pedestrian detection
    - Part-based detection: Deformable Part Models
  - Post-CNNs
    - Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    - Detection without region proposals: YOLO, SSD
- Segmentation approaches
  - Semantic segmentation: FCN
  - Instance segmentation: Mask R-CNN

Semantic Segmentation

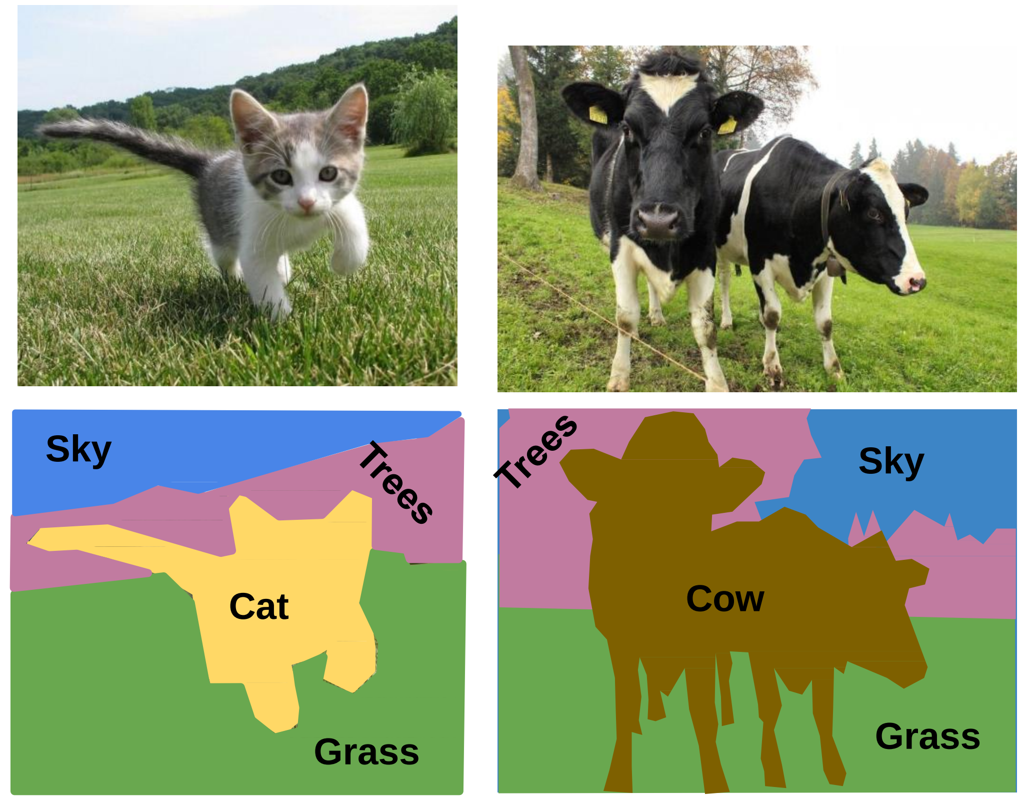


Slide by: Justin Johnson

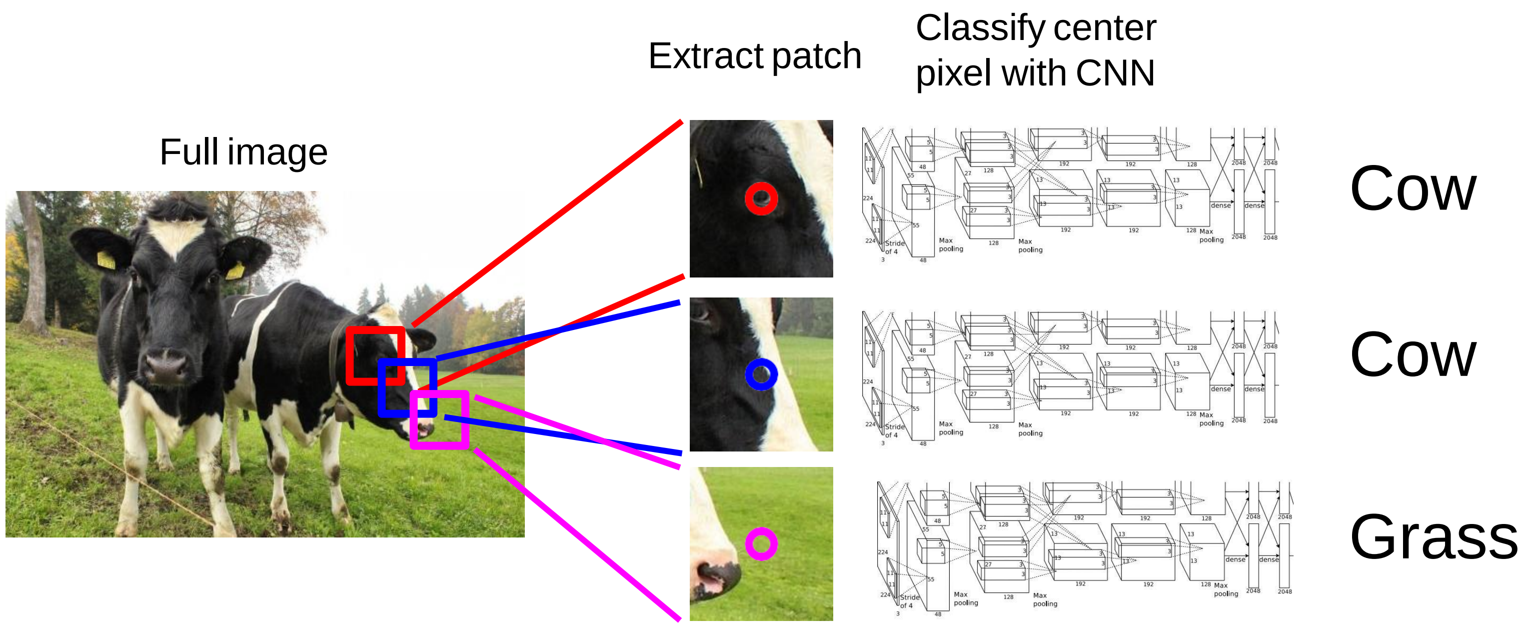
Semantic Segmentation

Label each pixel in the image with a category label

Don't differentiate instances, only care about pixels



Semantic Segmentation Idea: Sliding Window



Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013
Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014

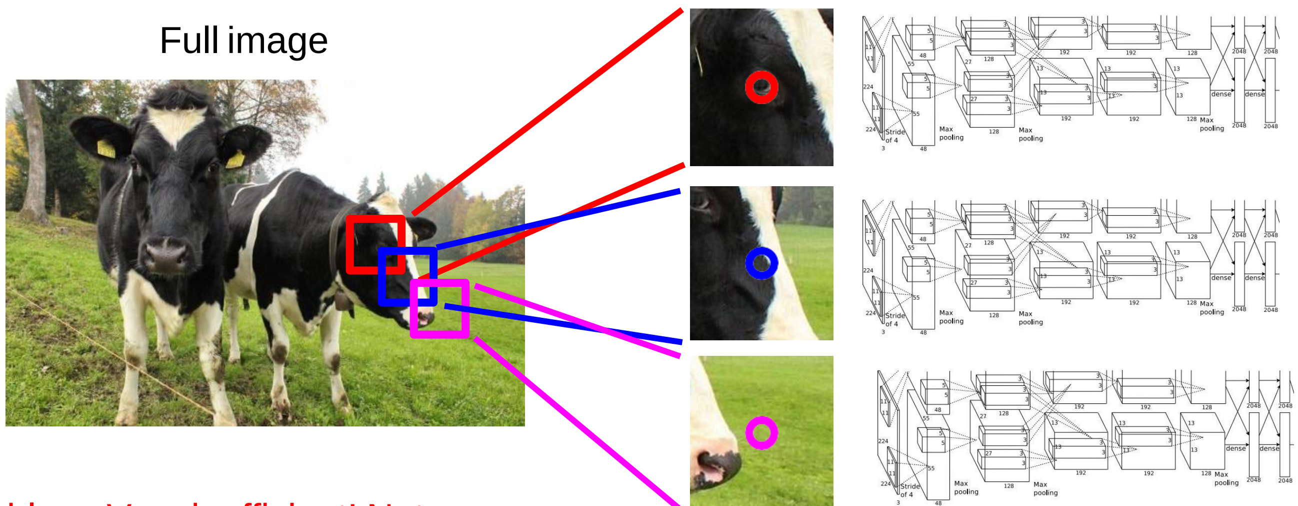
Slide by: Justin Johnson

Semantic Segmentation Idea: Sliding Window

Extract patch

Classify center pixel with CNN

Full image



Cow

Cow

Grass

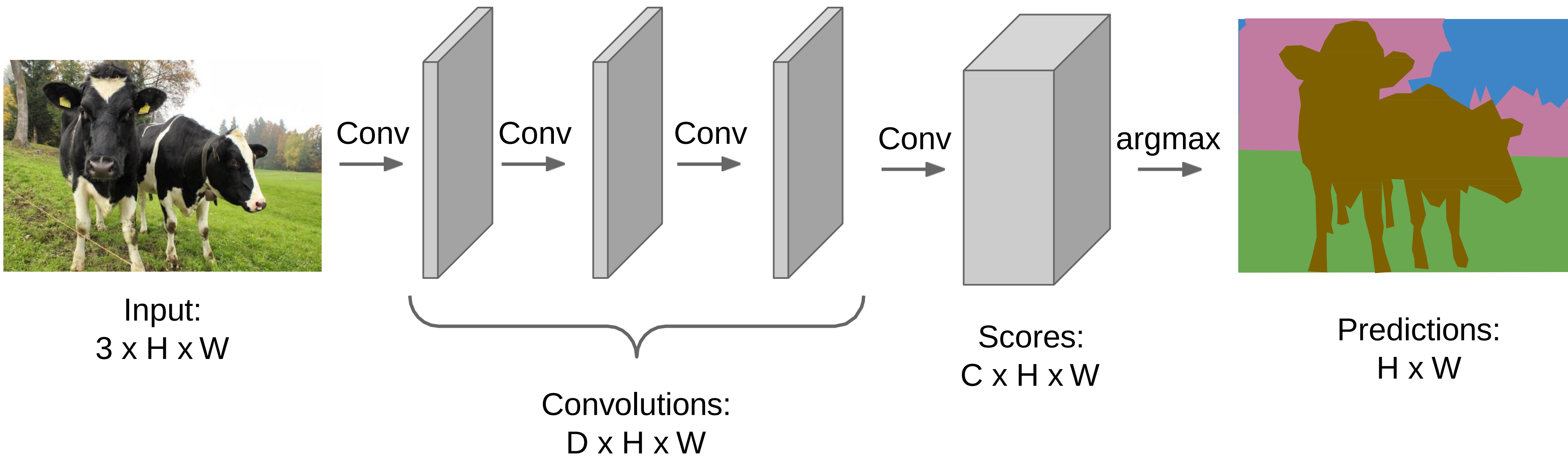
Problem: Very inefficient! Not reusing shared features between overlapping patches

Farabet et al, “Learning Hierarchical Features for Scene Labeling,” TPAMI 2013
Pinheiro and Collobert, “Recurrent Convolutional Neural Networks for Scene Labeling”, ICML 2014

Slide by: Justin Johnson

Semantic Segmentation Idea: Fully Convolutional

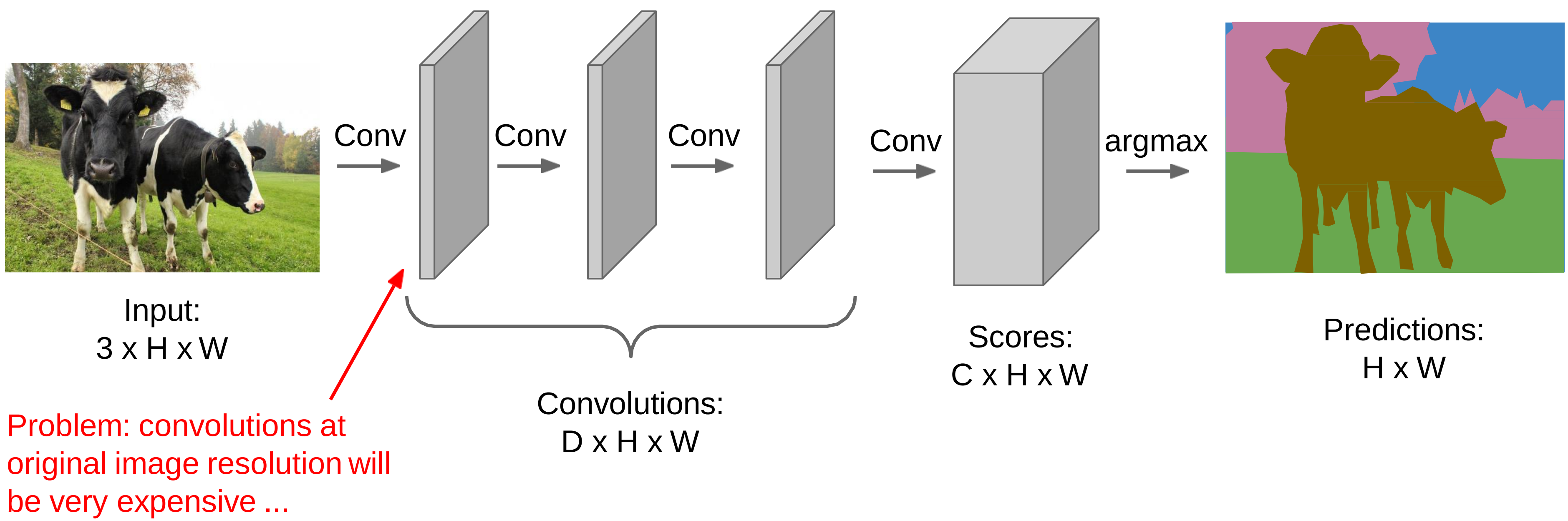
Design a network as a bunch of convolutional layers
to make predictions for pixels all at once!



Slide by: Justin Johnson

Semantic Segmentation Idea: Fully Convolutional

Design a network as a bunch of convolutional layers
to make predictions for pixels all at once!

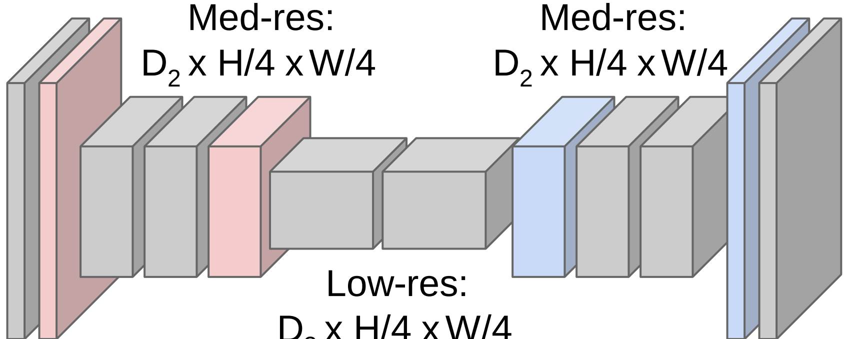


Slide by: Justin Johnson

Semantic Segmentation Idea: Fully Convolutional

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

Input:
3 x H x W



Predictions:
H x W

Long, Shelhamer, and Darrell, "Fully Convolutional Networks for Semantic Segmentation", CVPR 2015
Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV 2015

Slide by: Justin Johnson

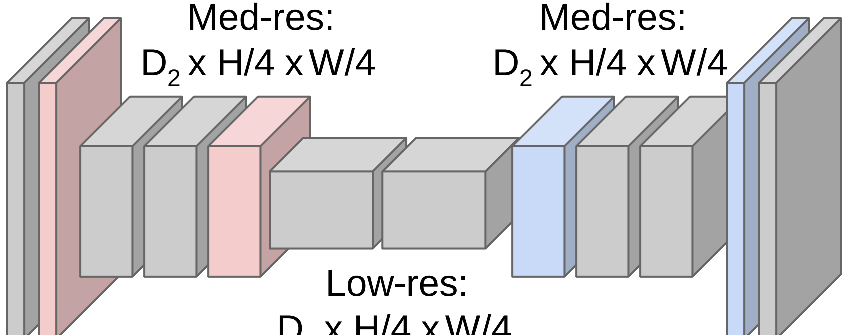
Semantic Segmentation Idea: Fully Convolutional

Downsampling: Pooling, strided convolution

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

Upsampling: ???

Input:
$3 \times H \times W$



High-res:
$D_1 \times H/2 \times W/2$

High-res:
$D_1 \times H/2 \times W/2$

Predictions:
$H \times W$

Long, Shelhamer, and Darrell, “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015
Noh et al, “Learning Deconvolution Network for Semantic Segmentation”, ICCV 2015

Slide by: Justin Johnson

In-Network upsampling: “Unpooling”

Nearest Neighbor

1	2
3	4

Input: 2 x 2

1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Output: 4 x 4

“Bed of Nails”

1	2
3	4

Input: 2 x 2

1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Output: 4 x 4

Slide by: Justin Johnson

In-Network upsampling: “Max Unpooling”

Max Pooling
Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4

5	6
7	8

Output: 2 x 2

…

Rest of the network

Max Unpooling
Use positions from pooling layer

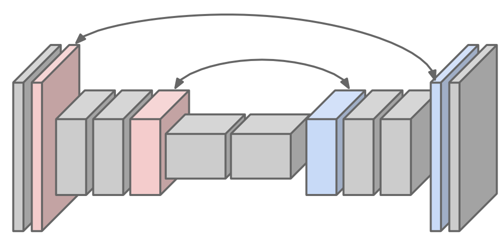
1	2
3	4

Input: 2 x 2

0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

Output: 4 x 4

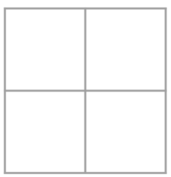
Corresponding pairs of downsampling and upsampling layers



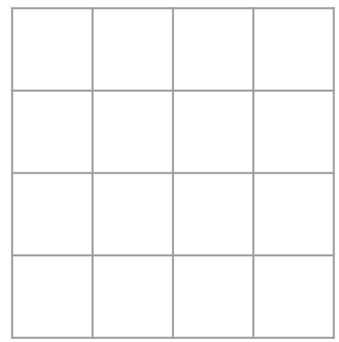
Slide by: Justin Johnson

Learnable Upsampling: Transpose Convolution

3 x 3 **transpose** convolution, stride 2 pad 1



Input: 2 x 2

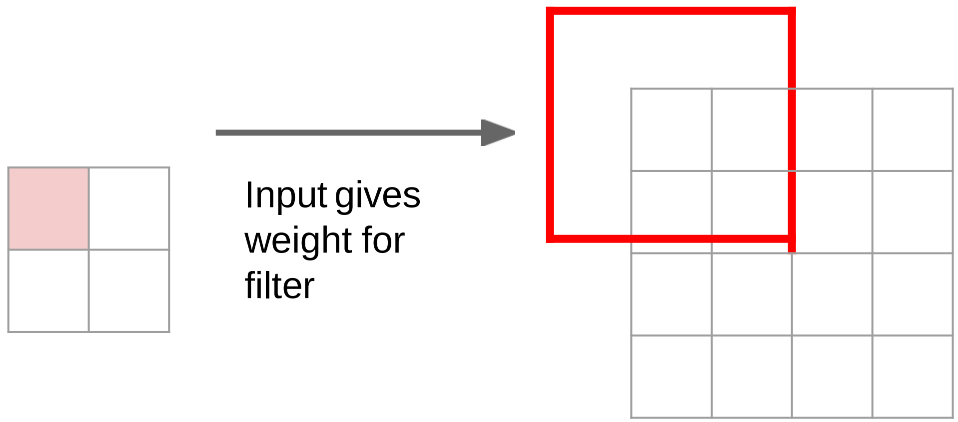


Output: 4 x 4

Slide by: Justin Johnson

Learnable Upsampling: Transpose Convolution

3 x 3 **transpose** convolution, stride 2 pad 1



Input: 2 x 2

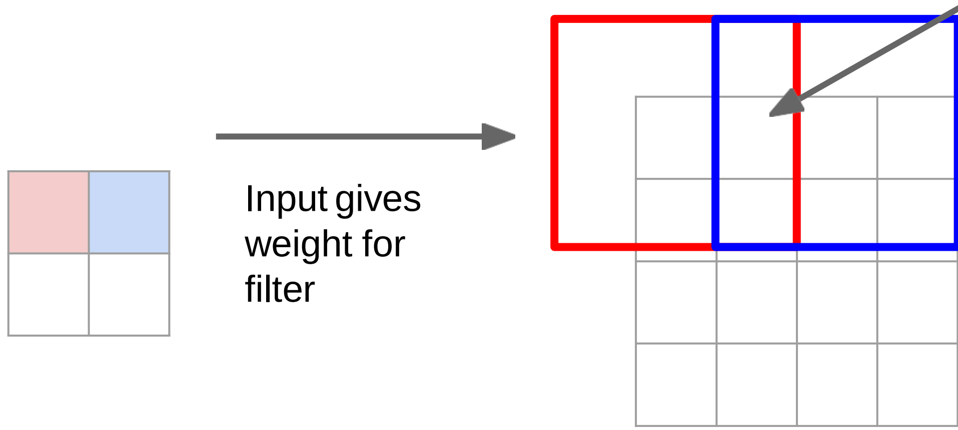
Output: 4 x 4

Slide by: Justin Johnson

Learnable Upsampling: Transpose Convolution

3 x 3 **transpose** convolution, stride 2 pad 1

Sum where output overlaps



Input: 2 x 2

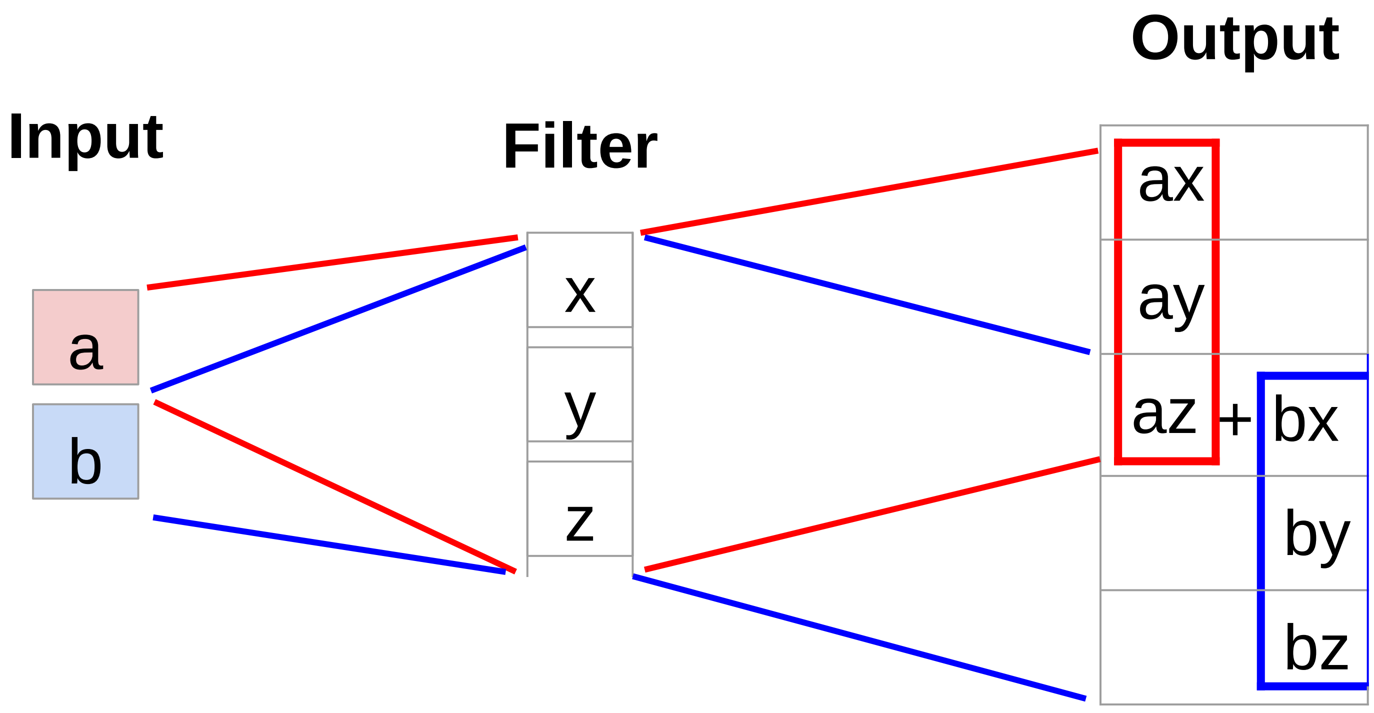
Output: 4 x 4

Filter moves 2 pixels in the output for every one pixel in the input

Stride gives ratio between movement in output and input

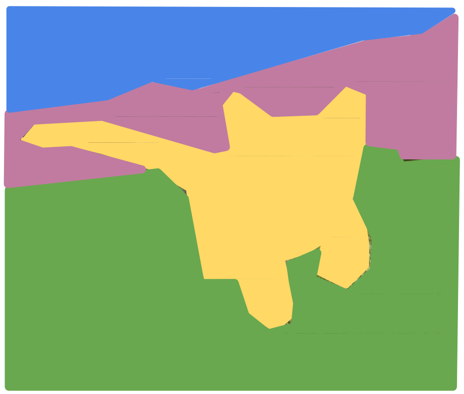
Slide by: Justin Johnson

Transpose Convolution: 1D Example



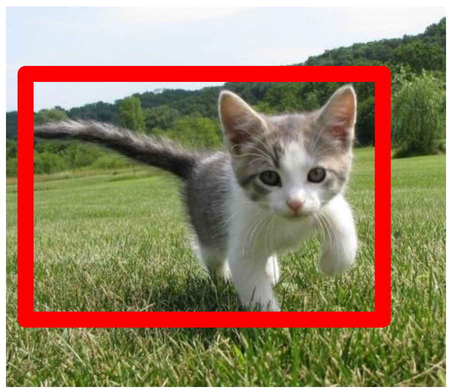
Output contains copies of the filter weighted by the input, summing at where at overlaps in the output

Instance Segmentation



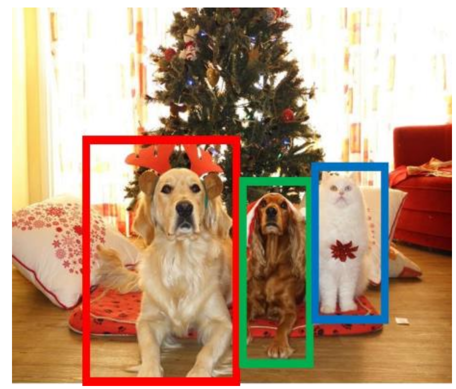
GRASS, CAT, TREE, SKY

No objects, just pixels



CAT

Single Object



DOG, DOG, CAT

DOG, DOG, CAT

Multiple Object

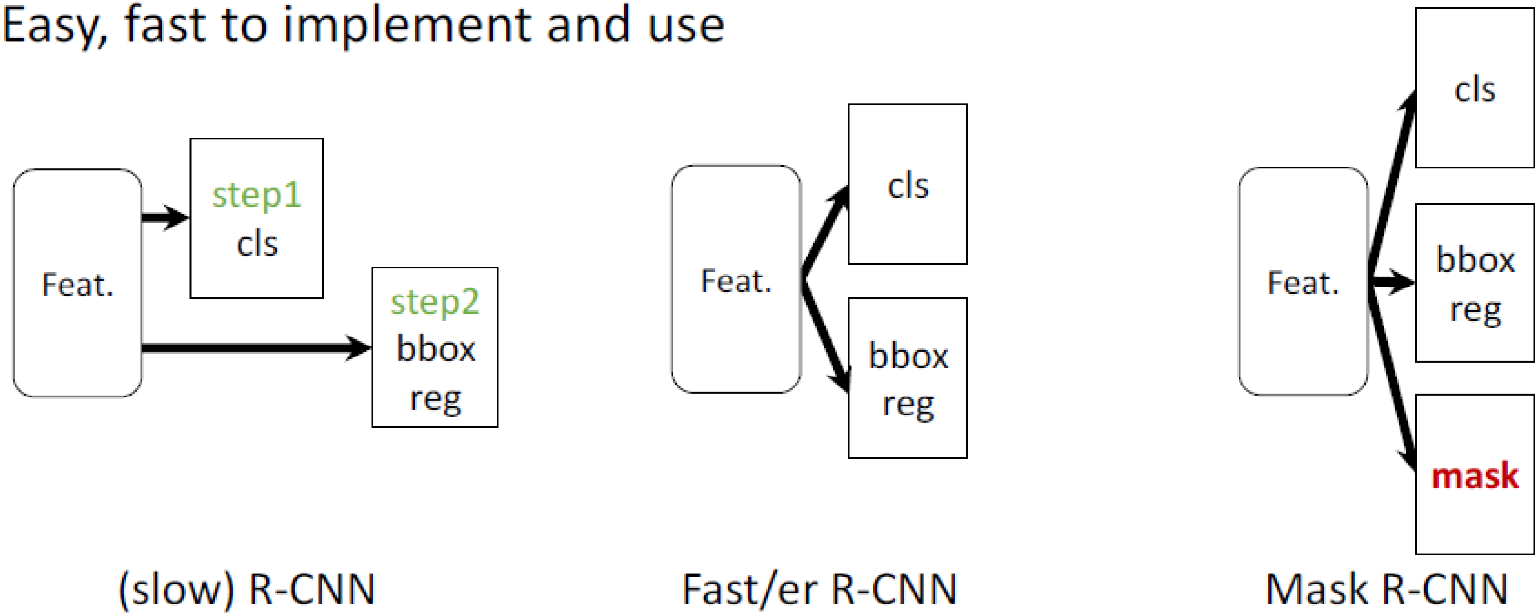
Slide by: Justin Johnson

Mask R-CNN

He et al, “Mask R-CNN”, ICCV 2017

What is Mask R-CNN: Parallel Heads

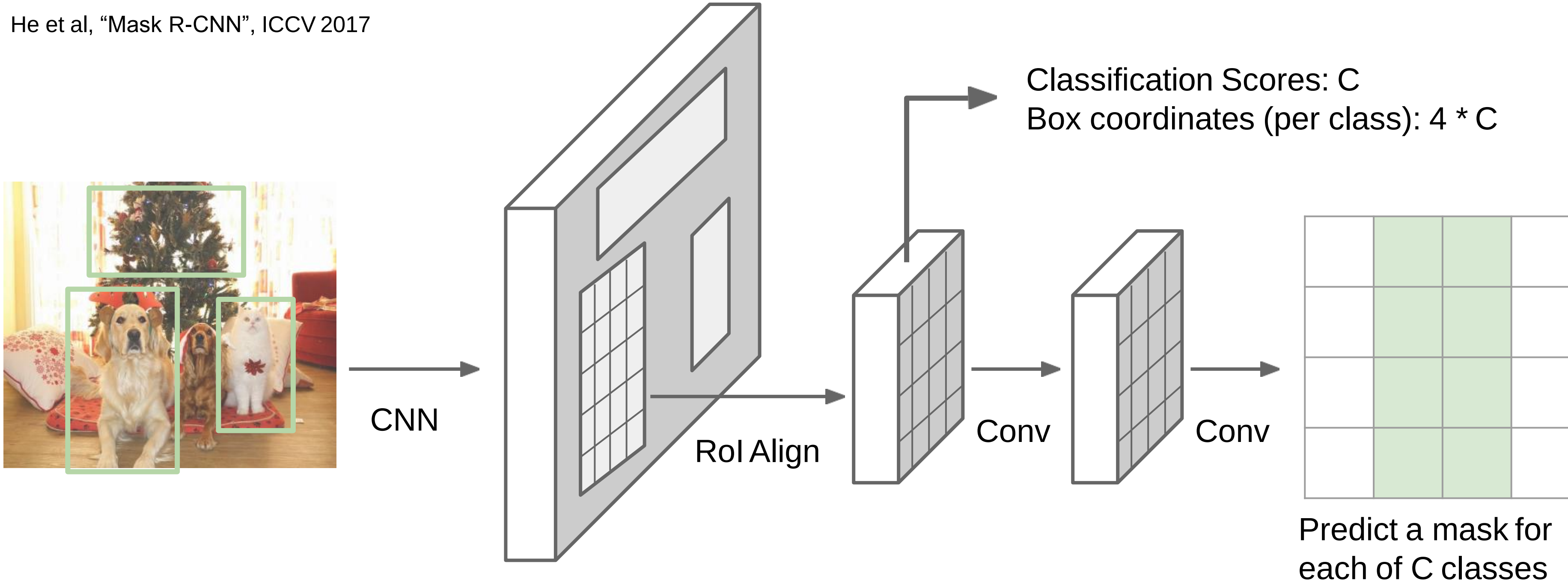
- Easy, fast to implement and use



Slide by: Kaiming He

Mask R-CNN

He et al, “Mask R-CNN”, ICCV 2017



Adapted from Justin Johnson