Object Detection
(Plus some bonuses)

EECS 442 – David Fouhey

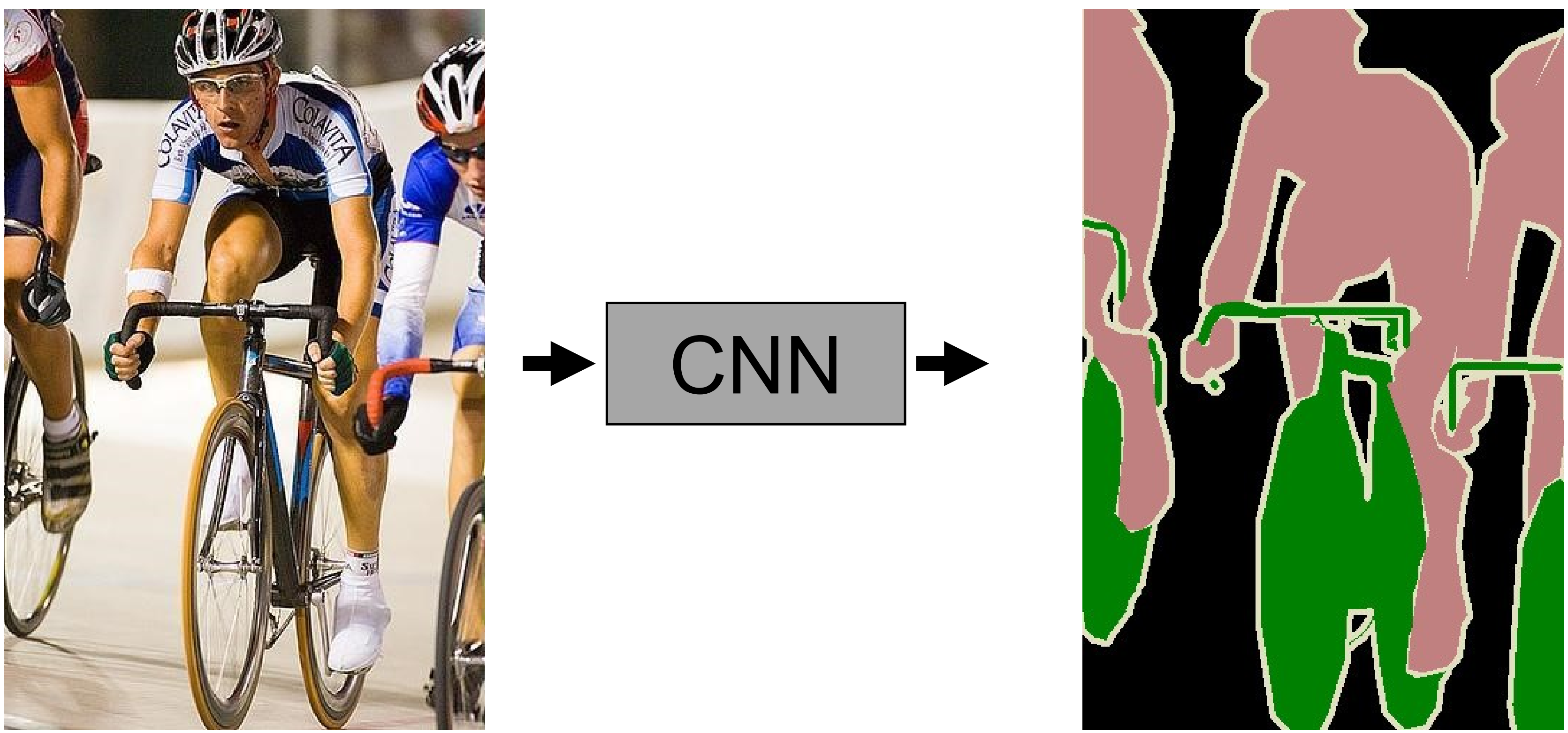
Fall 2019, University of Michigan

http://web.eecs.umich.edu/~fouhey/teaching/EECS442_F19/

Last Time

"Semantic Segmentation": Label each pixel with the object category it belongs to.

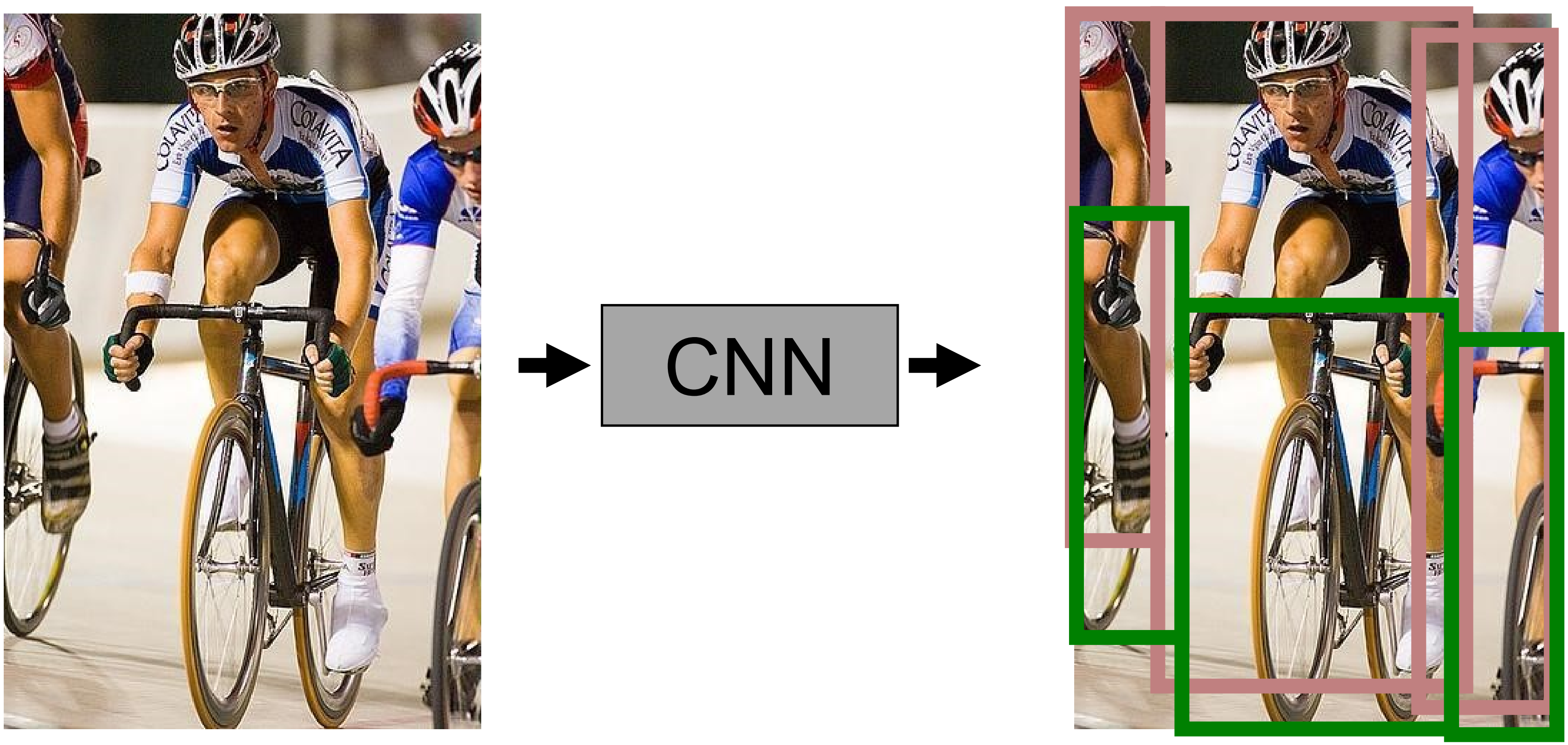
Input Target



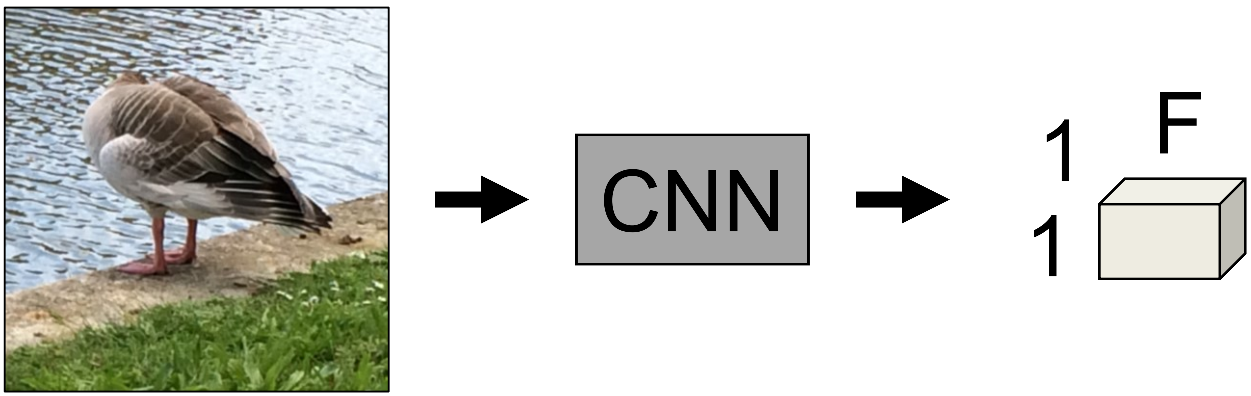
Today – Object Detection

"Object Detection": Draw a box around each instance of a list of categories

Input Target



The Wrong Way To Do It

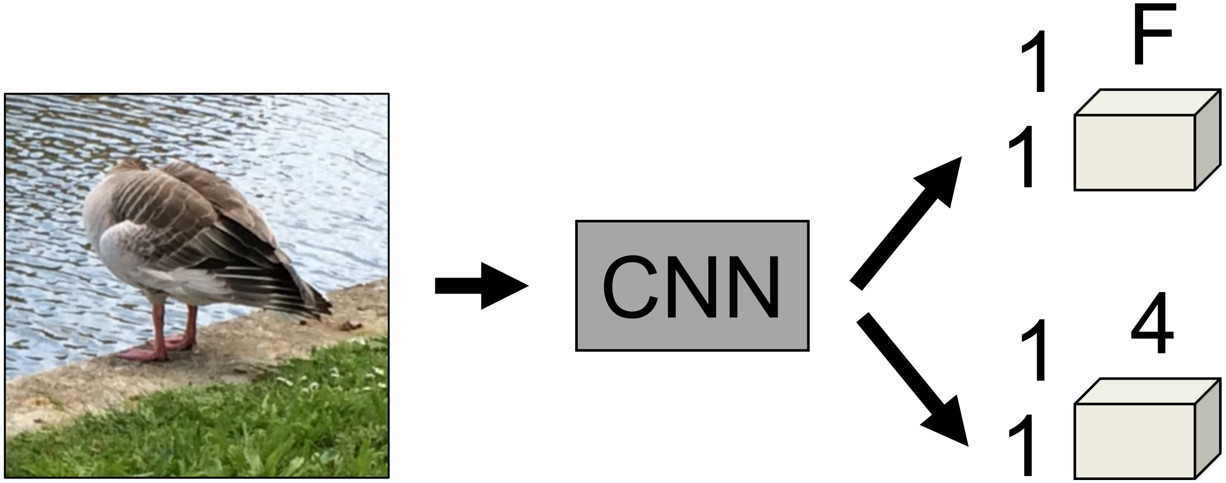


Starting point:

Can predict the probability of F classes

P(cat), P(goose), … P(tractor)

The Wrong Way To Do It

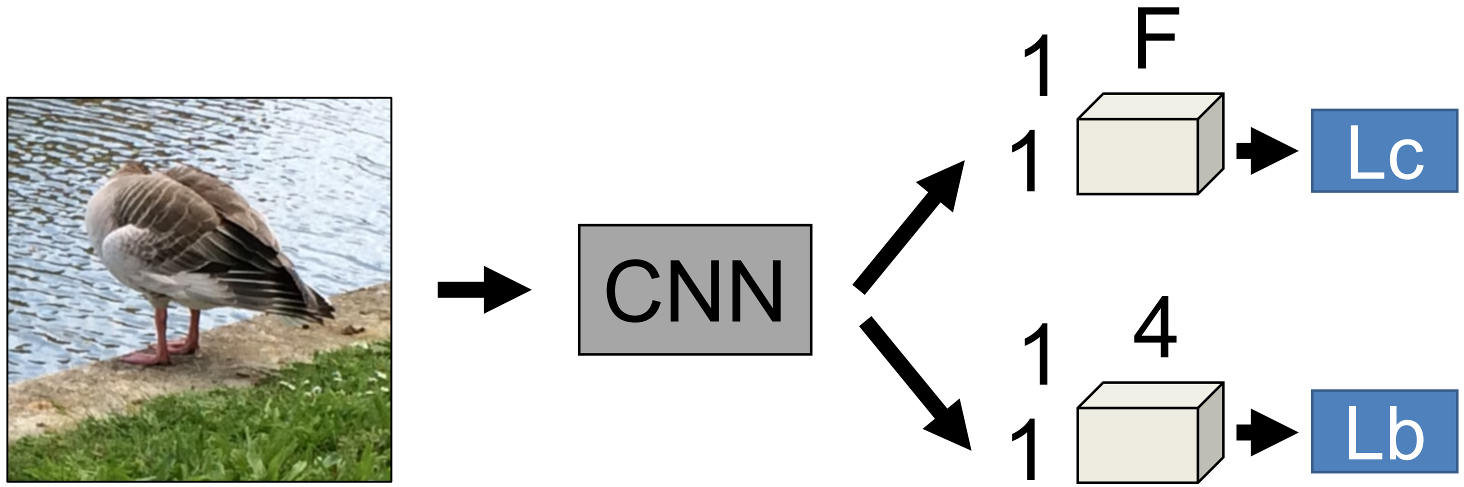


Add another output (why not):

Predict the *bounding box* of the object

[x,y,width,height] or [minX,minY,maxX,maxY]

The Wrong Way To Do It



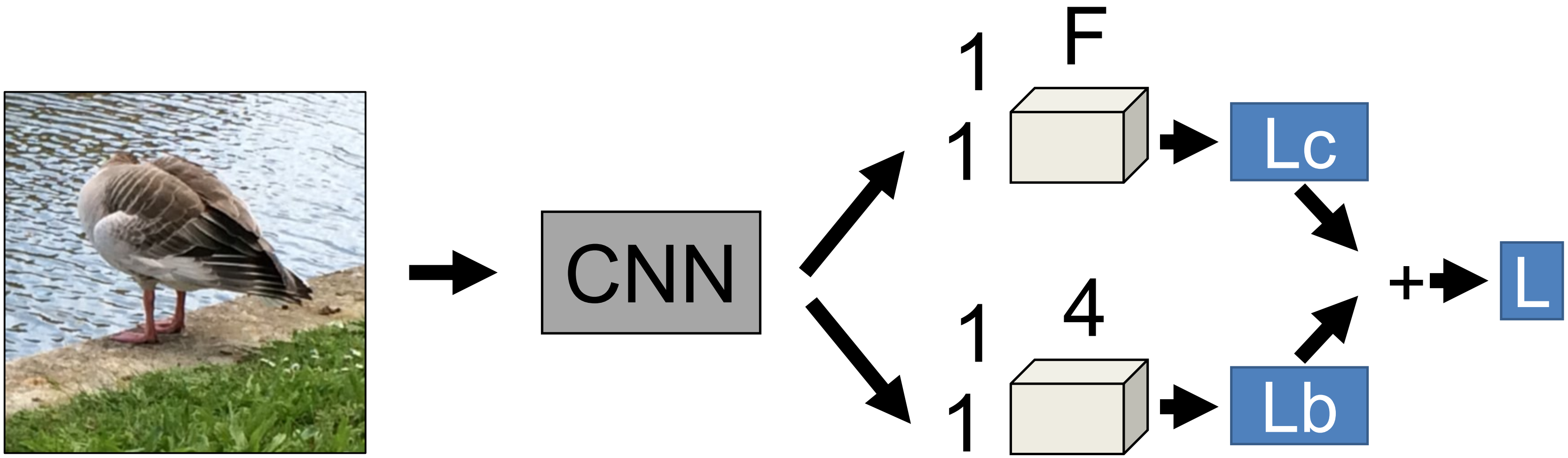
Put a loss on it:

Penalize mistakes on the classes with

Lc = negative log-likelihood

Lb = L2 loss

The Wrong Way To Do It

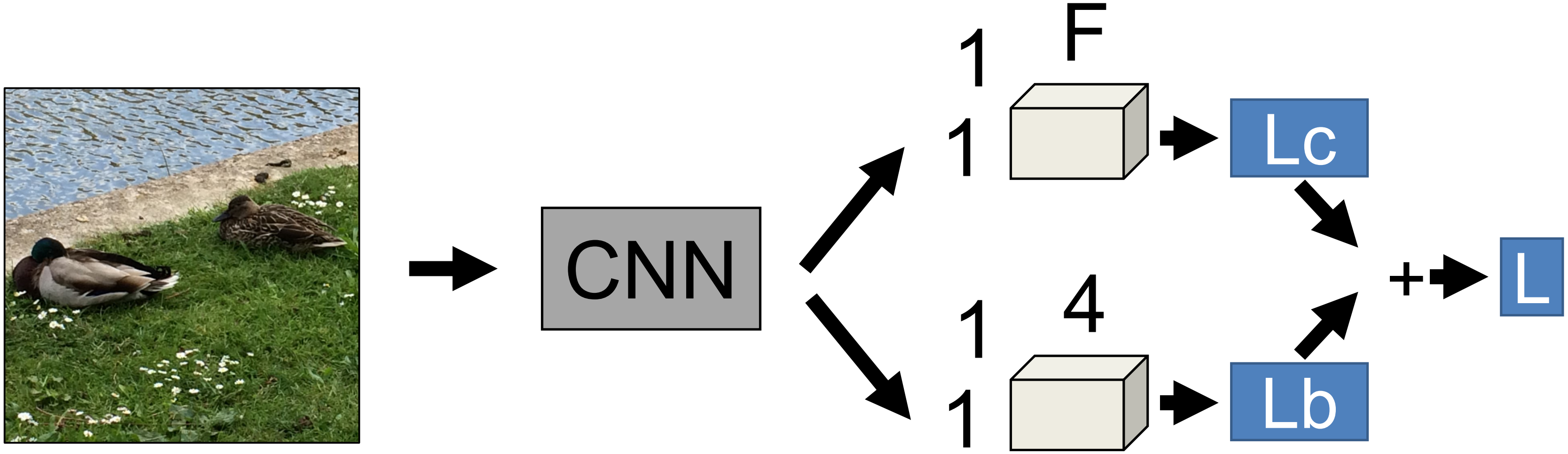


Add losses, backpropagate

Final loss: L = Lc + λLb

Why do we need the λ?

The Wrong Way To Do It

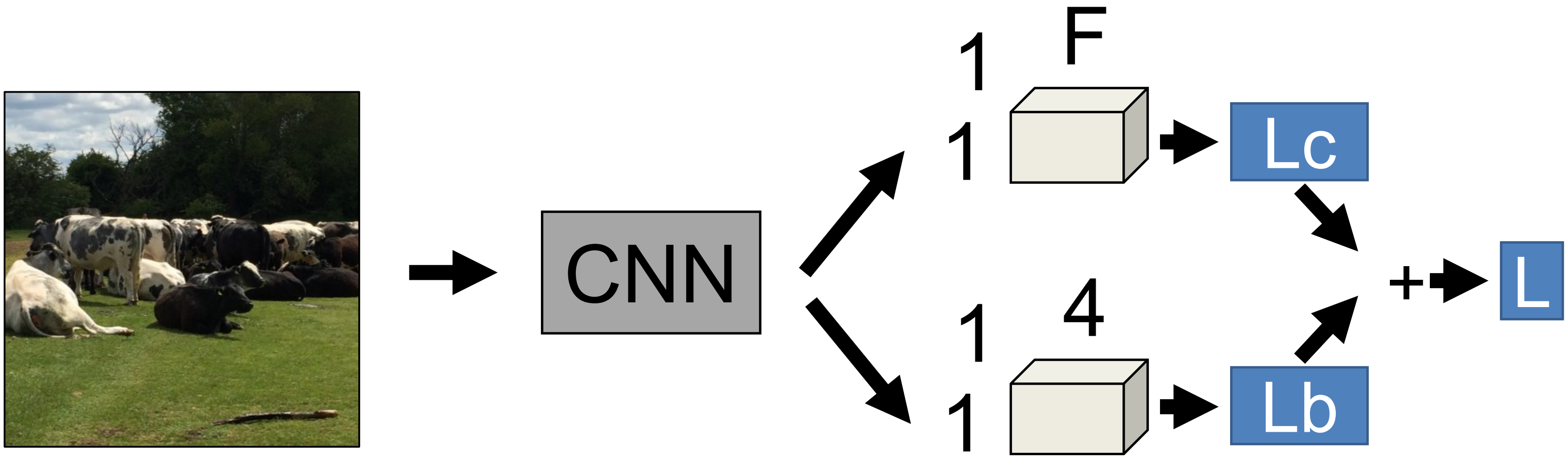


Now there are two ducks.

How many outputs do we need?

F, 4, F, 4 = 2*(F+4)

The Wrong Way To Do It

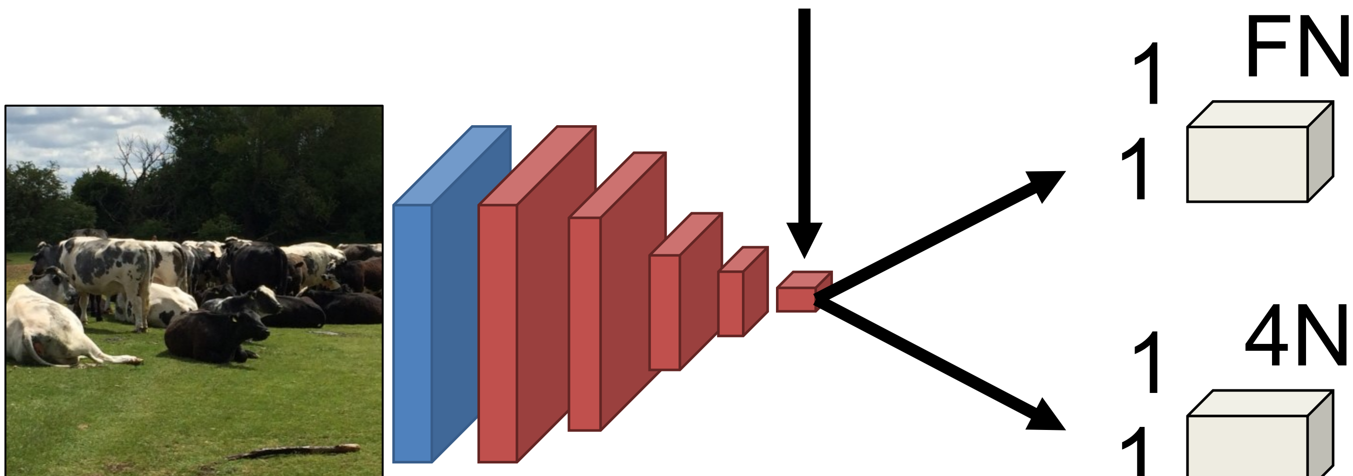


Now it's a herd of cows.
We need *lots* of outputs
(in fact the precise number of objects that are in the image, which is circular reasoning).

In General

- Usually can’t do varying-size outputs.
- Even if we could, think about how *you* would solve it if you were a network.

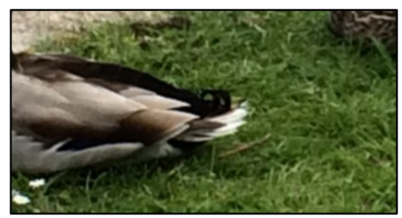
Bottleneck has to *encode* where the objects are for all objects and all N



An Alternate Approach

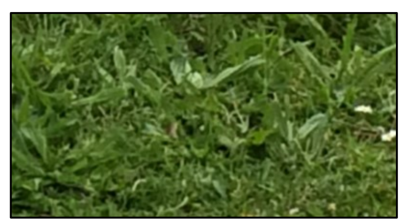
Examine every sub-window and determine if it is a tight box around an object

Yes



No?

Hold this thought



No

Sliding Window Classification

Let’s assume we’re looking for pedestrians in a box with a fixed aspect ratio.

Slide credit: J. Hays

Sliding Window

Key idea – just try all the subwindows in the image at all positions.

Slide credit: J. Hays

Generating hypotheses

Key idea – just try all the subwindows in the image at all positions **and scales.**

Note – Template did not change size

Slide credit: J. Hays

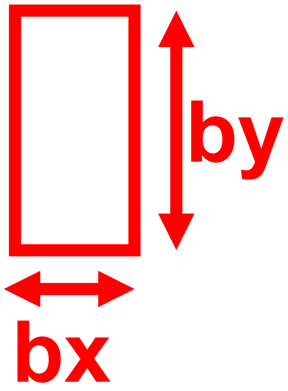
Each window classified separately

How Many Boxes Are There?

Given a HxW image and a "template" of size by, bx.

Q. How many sub-boxes are there of size (by,bx)?

A. (H-by)*(W-bx)

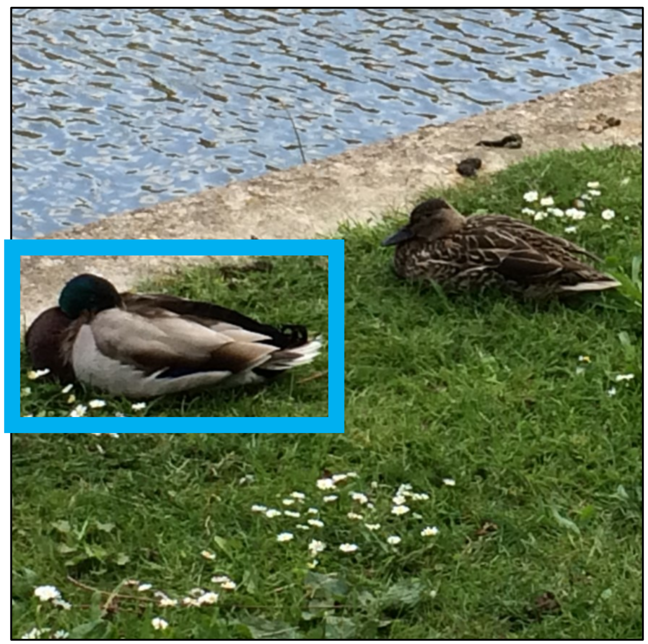


This is before considering adding:

- *scales* (by*s,bx*s)
- *aspect ratios* (by*sy,bx*sx)

Challenges of Object Detection

- Have to evaluate *tons* of boxes
- Positive instances of objects are *extremely* rare



How many ways can we get the box wrong?

1. Wrong left x
2. Wrong right x
3. Wrong top y
4. Wrong bottom y

Prime-time TV

Are You Smarter Than A 5th Grader?

Adults compete with 5th graders on elementary school facts.

Adults often not smarter.

Computer Vision TV

Are You Smarter Than A Random Number Generator?

Models trained on data compete with making random guesses.

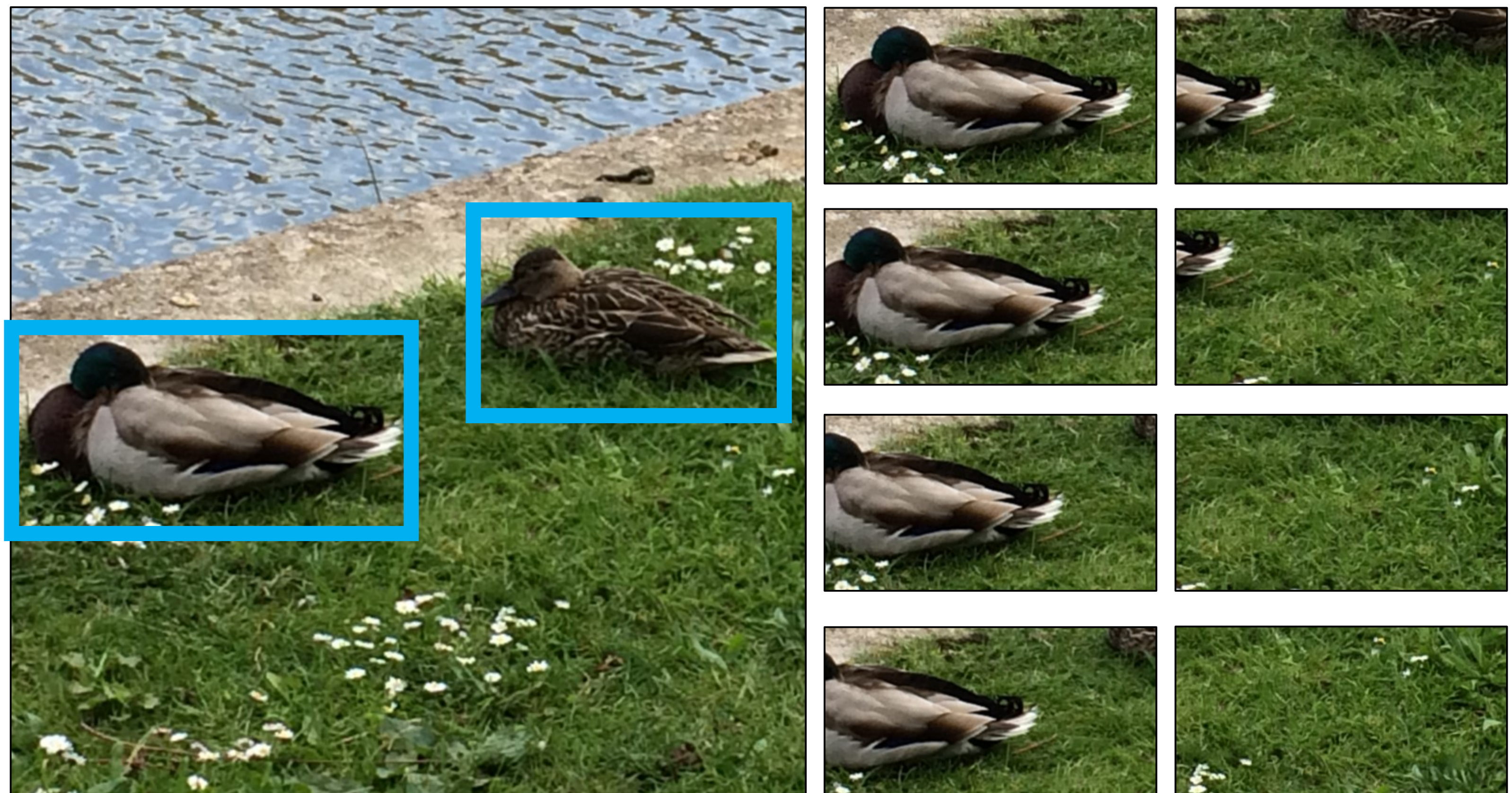
Models often not better.

Are You Smarter than a Random Number Generator?

- **Prob. of guessing 1k-way classification?**
  - 1/1,000
- **Prob. of guessing all 4 bounding box corners within 10% of image size**?
  - (1/10)*(1/10)*(1/10)*(1/10)=1/10,000
- Probability of guessing both: 1/10,000,000
- Detection is **hard** (via guessing and in general)
- Should *always compare* against guessing or picking most likely output label

Evaluating – Bounding Boxes

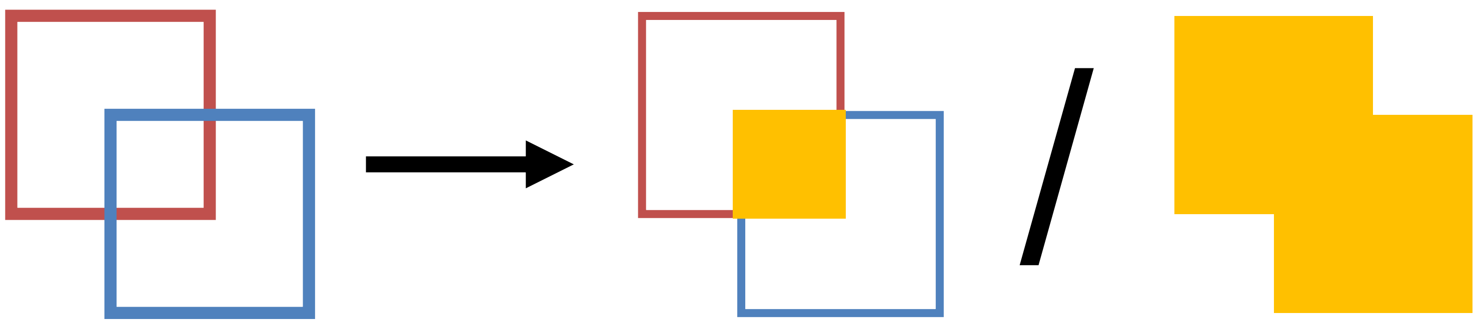
Raise your hand when you think the detection stops being correct.

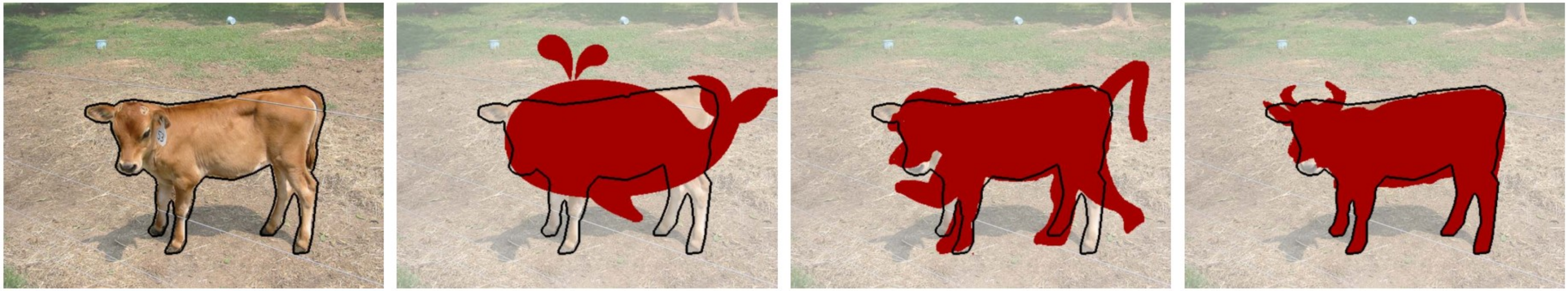


Evaluating – Bounding Boxes

Standard metric for two boxes:

Intersection over union/IoU/Jaccard coefficient





(a) Ground truth (b) $\mathcal{J} = 0.554$ (c) $\mathcal{J} = 0.703$ (d) $\mathcal{J} = 0.910$

Jaccard example credit: P. Kraehenbuehl et al. ECCV 2014

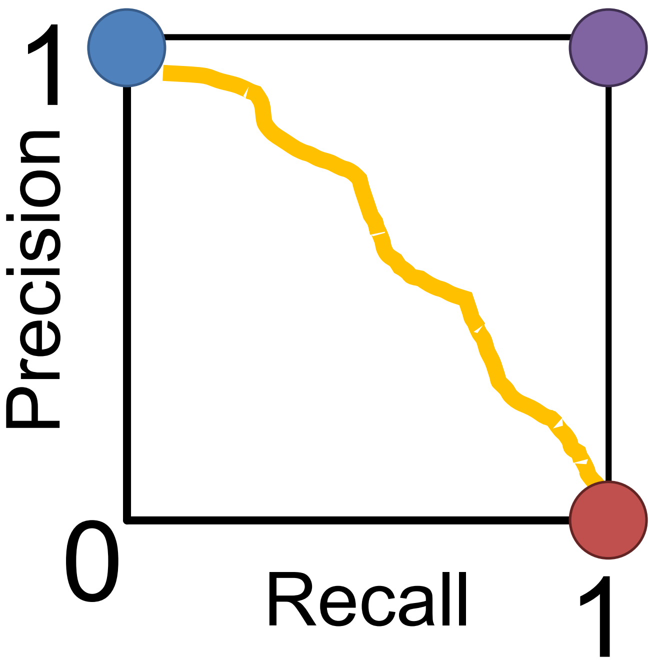
Evaluating Performance

- Remember: accuracy = average of whether prediction is correct
- Suppose I have a system that gets 99% accuracy in person detection.
- **What’s wrong?**
- I can get that by just saying no object everywhere!

Evaluating Performance

- True detection: high intersection over union
- Precision: #true detections / #detections
- Recall: #true detections / #true positives

Reject everything: no mistakes

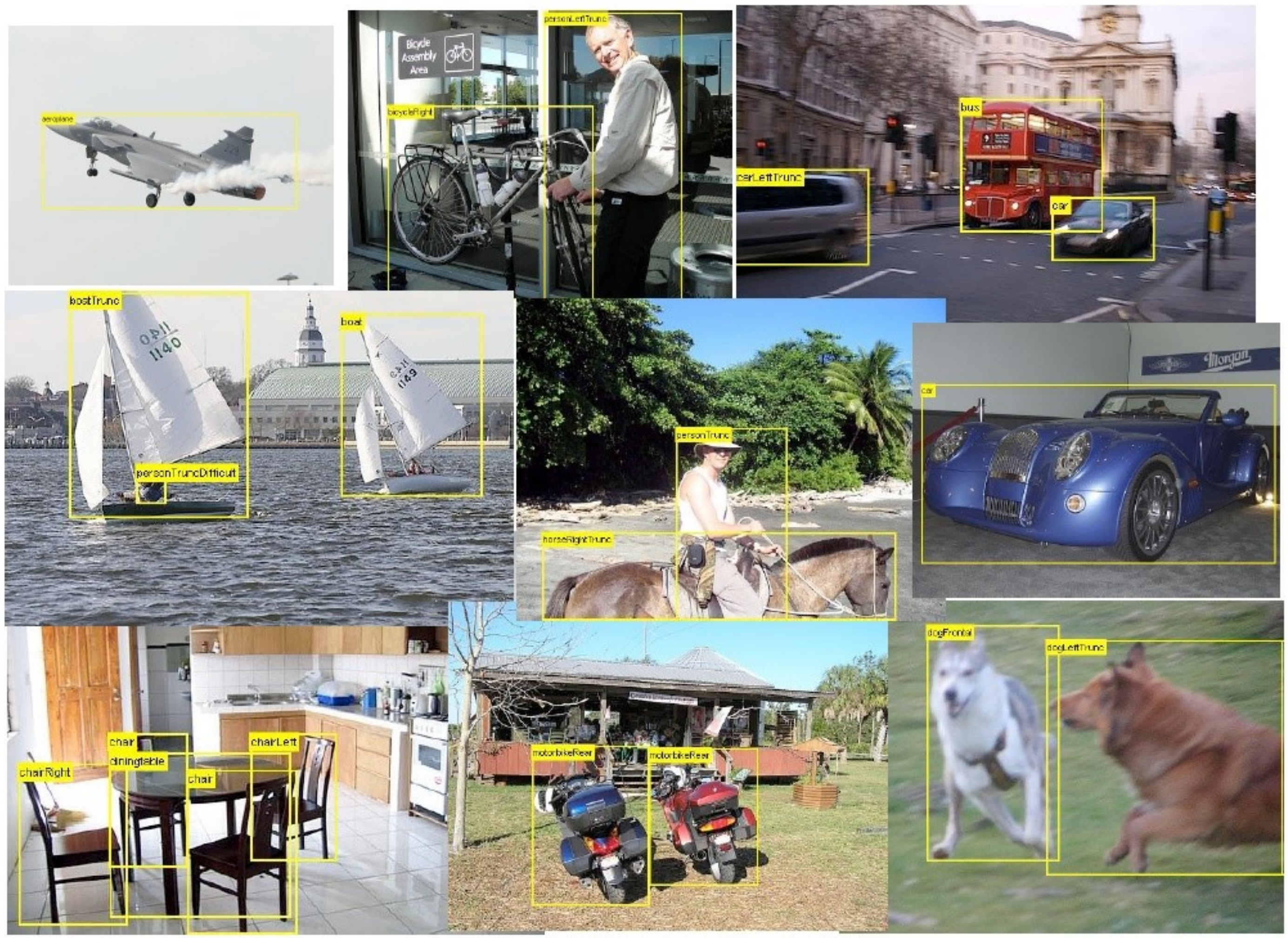


Ideal!

Summarize by area under curve (avg. precision)

Accept everything: Miss nothing

Generic object detection

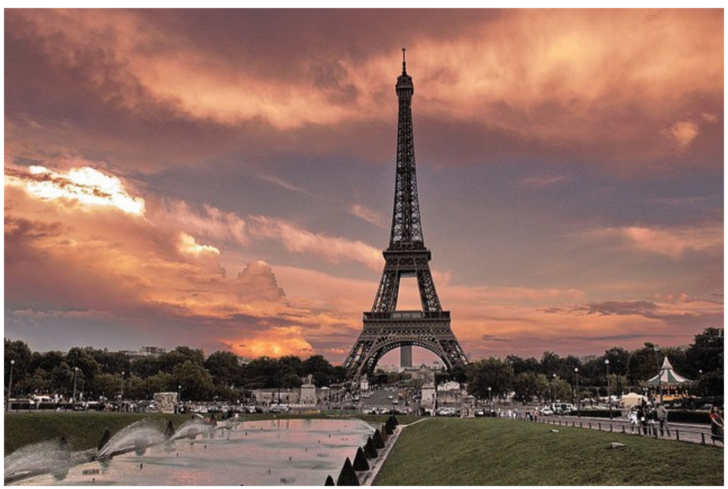


Slide Credit: S. Lazebnik

Histograms of oriented gradients (HOG)

Partition image into blocks and compute histogram of gradient orientations in each block

HxWx3 Image



H'xW'xC' Image

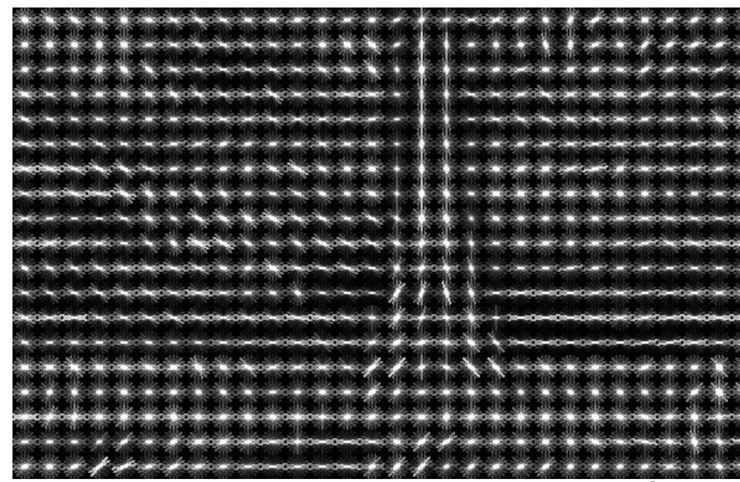


Image credit: N. Snavely

N. Dalal and B. Triggs, [Histograms of Oriented Gradients for Human Detection](), CVPR 2005

Slide Credit: S. Lazebnik

Pedestrian detection with HOG

- Train a pedestrian template using a linear support vector machine

positive training examples

negative training examples

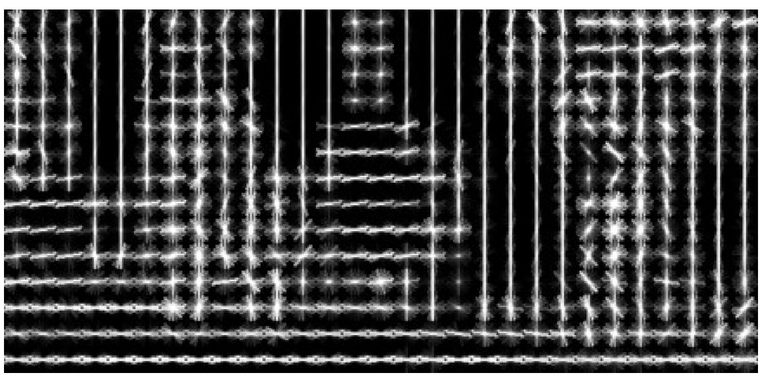
N. Dalal and B. Triggs, [Histograms of Oriented Gradients for Human Detection](), CVPR 2005

Slide Credit: S. Lazebnik

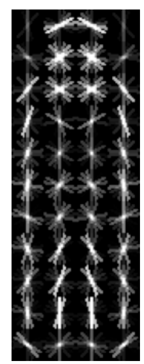
Pedestrian detection with HOG

- Train pedestrian "template" using a linear svm
- At test time, convolve feature map with template
- Find local maxima of response
- For multi-scale detection, repeat over multiple levels of a HOG *pyramid*

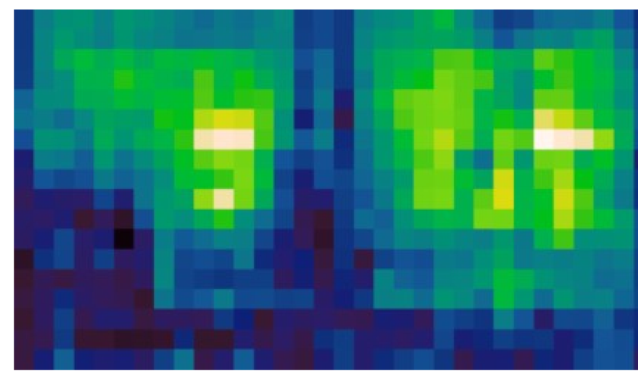
HOG feature map



Template



Detector response map



N. Dalal and B. Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005

Slide Credit: S. Lazebnik

Example detections

[Dalal and Triggs, CVPR 2005]

Slide Credit: S. Lazebnik

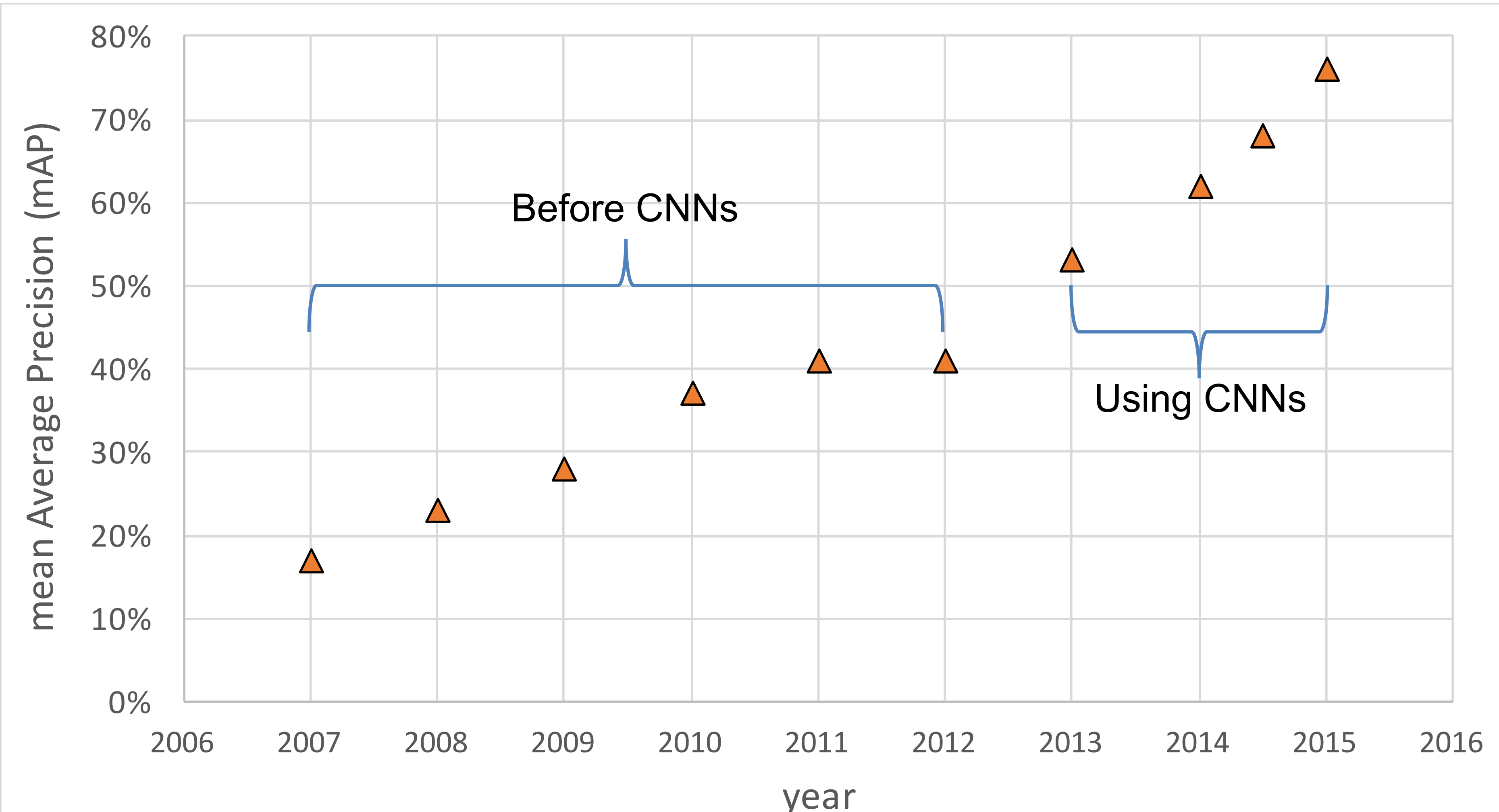
PASCAL VOC Challenge (2005-2012)

- 20 challenge classes:
- *Person*
- *Animals:* bird, cat, cow, dog, horse, sheep
- *Vehicles:* aeroplane, bicycle, boat, bus, car, motorbike, train
- *Indoor:* bottle, chair, dining table, potted plant, sofa, tv/monitor
- Dataset size (by 2012): 11.5K training/validation images, 27K bounding boxes, 7K segmentations

Slide Credit: S. Lazebnik http://host.robots.ox.ac.uk/pascal/VOC/

Object detection progress

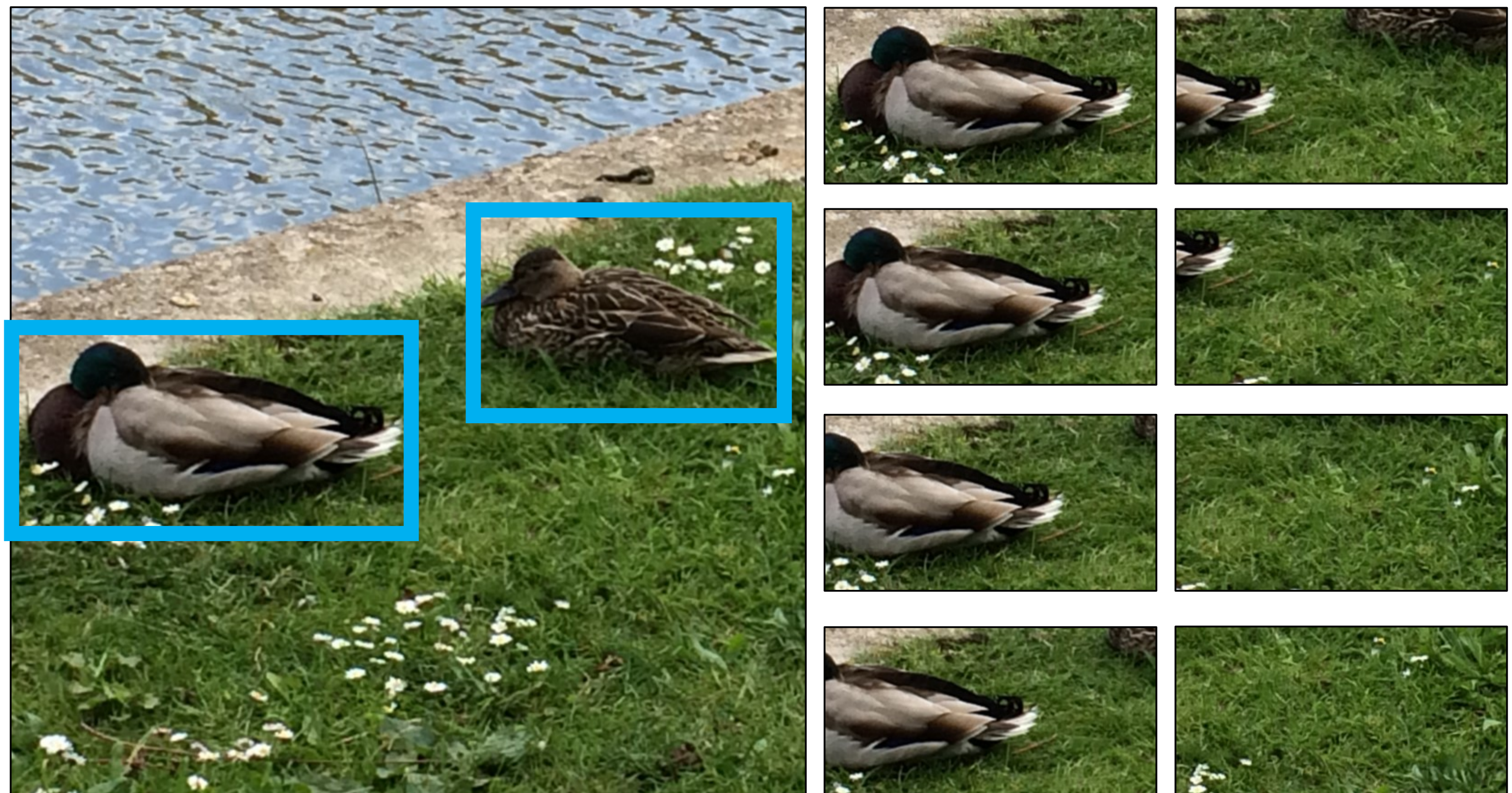
PASCAL VOC



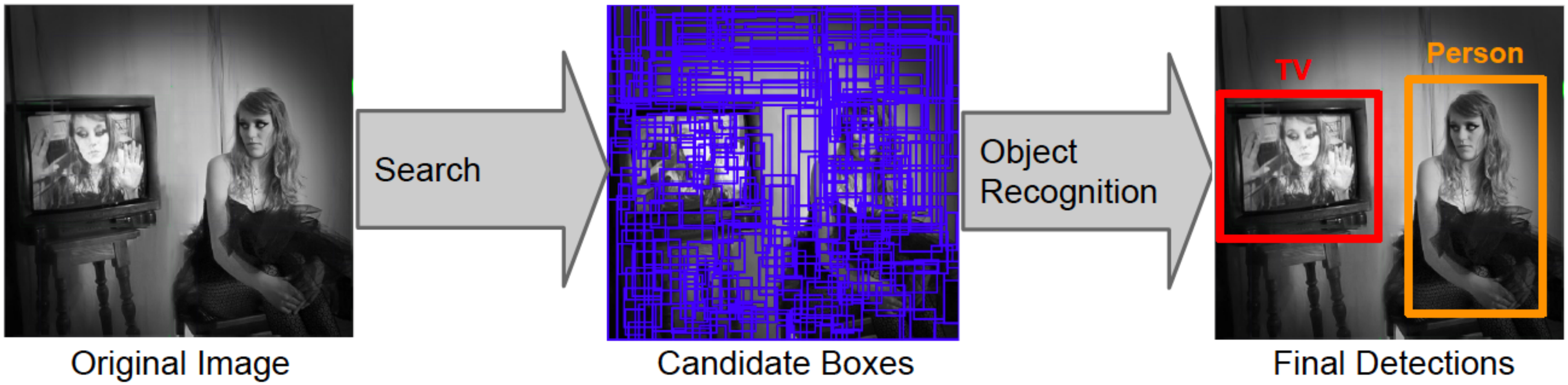
Source: R. Girshick

Region Proposals

Do I need to spend a lot of time filtering all the boxes covering grass?



Region proposals

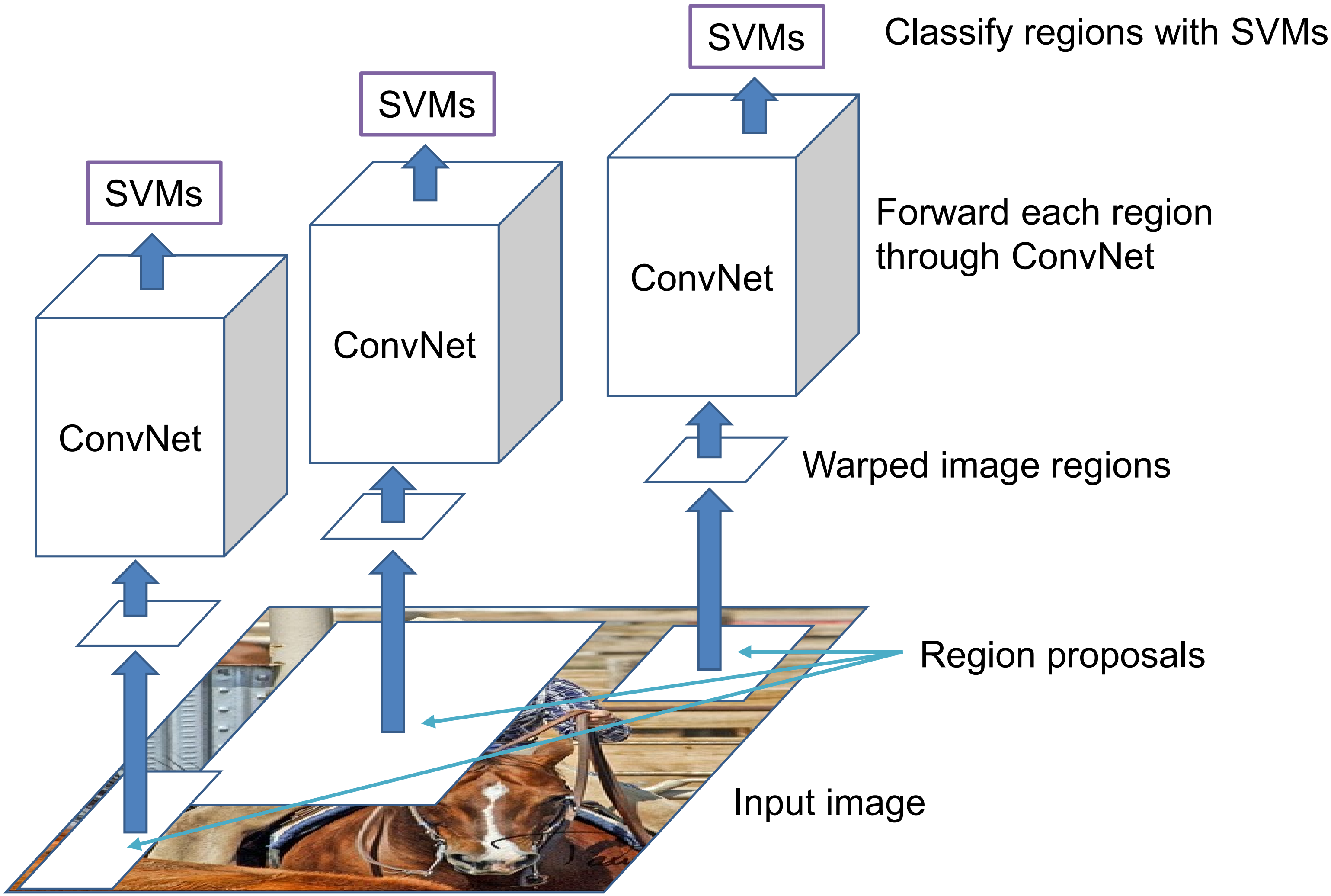


- As an alternative to sliding window search, evaluate a few hundred *region proposals*
  - Can use slower but more powerful features and classifiers
  - Proposal mechanism can be category-independent
  - Proposal mechanism can be trained

Slide Credit: S. Lazebnik

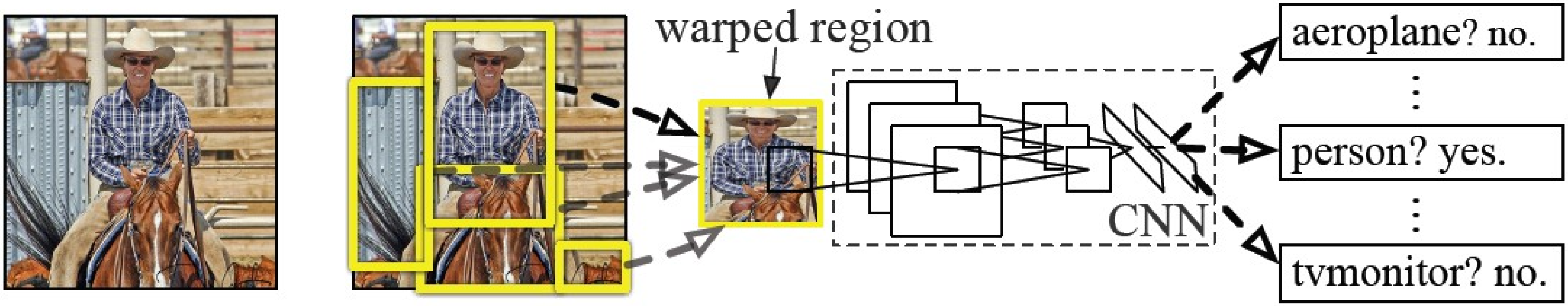
R-CNN: Region proposals + CNN features

Source: R. Girshick



R. Girshick, J. Donahue, T. Darrell, and J. Malik, [Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation], CVPR 2014.

R-CNN details



- **Regions**: ~2000 Selective Search proposals
- **Network**: AlexNet *pre-trained* on ImageNet (1000 classes), *fine-tuned* on PASCAL (21 classes)
- **Final detector**: warp proposal regions, extract fc7 network activations (4096 dimensions), classify with linear SVM
- **Bounding box regression** to refine box locations
- **Performance:** mAP of 53.7% on PASCAL 2010 (vs. 35.1% for Selective Search and 33.4% for DPM).

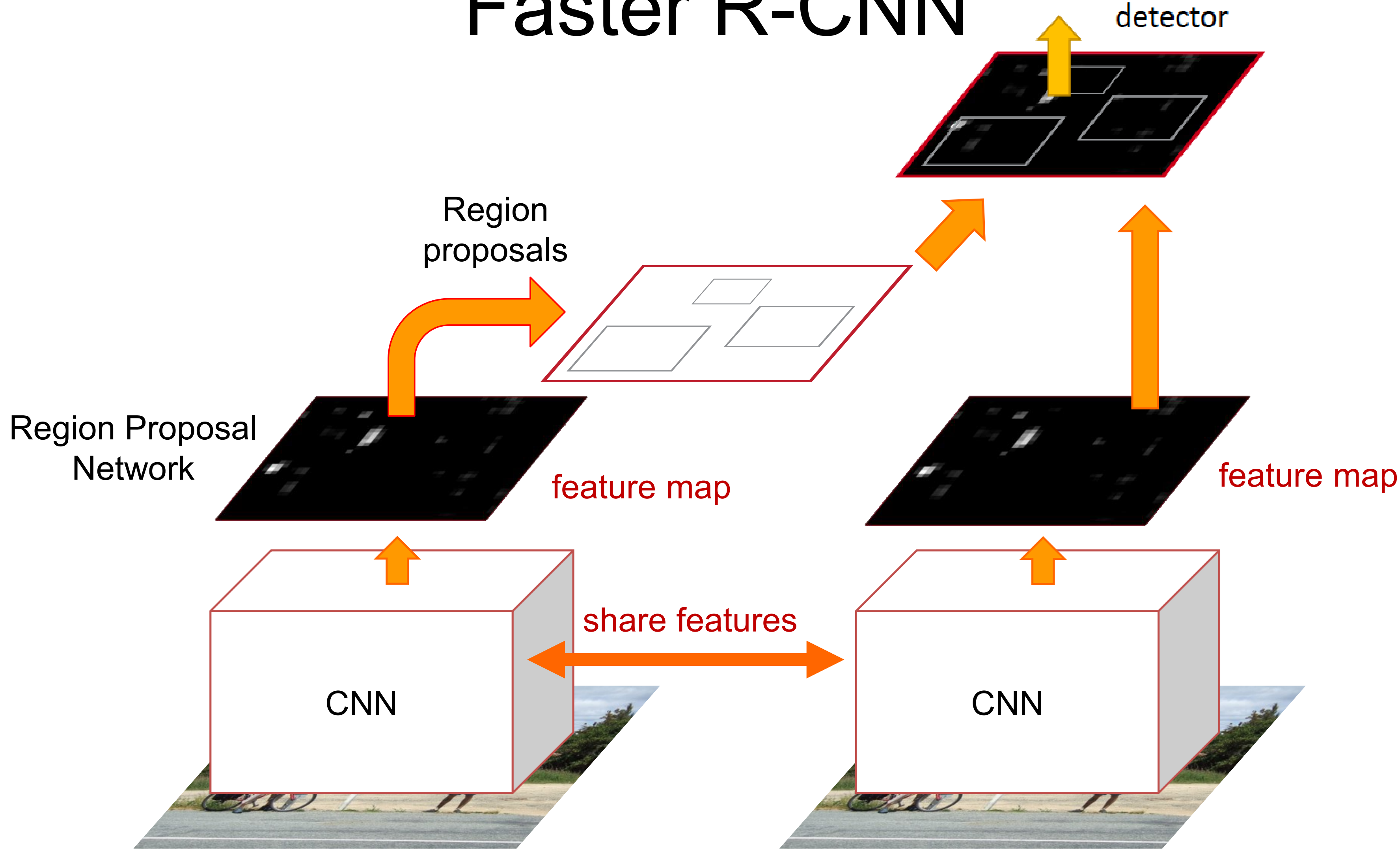
R. Girshick, J. Donahue, T. Darrell, and J. Malik, [Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation](), CVPR 2014.

R-CNN pros and cons

- Pros
  - Accurate!
  - Any deep architecture can immediately be “plugged in”
- Cons
  - Ad hoc training objectives
    - Fine-tune network with softmax classifier (log loss)
    - Train post-hoc linear SVMs (hinge loss)
    - Train post-hoc bounding-box regressions (least squares)
  - Training is slow (84h), takes a lot of disk space
    - 2000 CNN passes per image
  - Inference (detection) is slow (47s / image with VGG16)

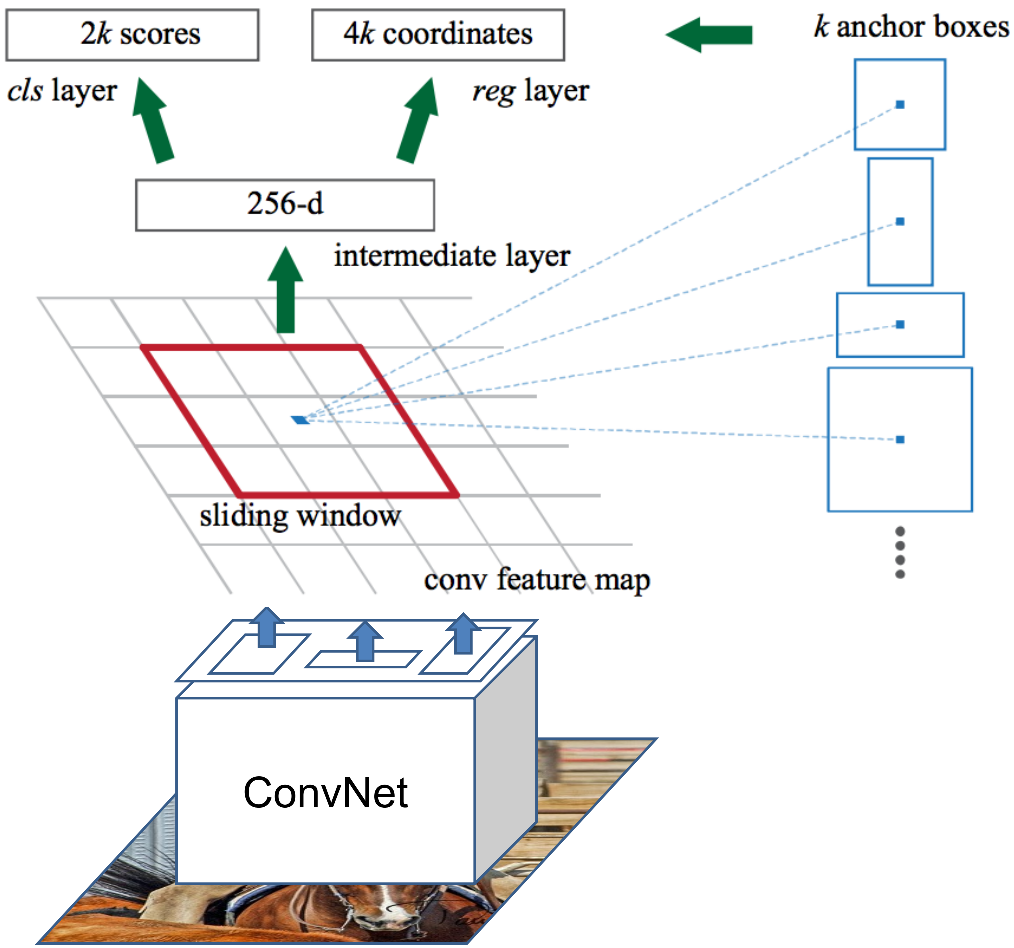
Slide Credit: S. Lazebnik

Faster R-CNN



S. Ren, K. He, R. Girshick, and J. Sun, [Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks], NIPS 2015

Region Proposal Network (RPN)



Small network applied to conv5 feature map.

Predicts:

- good box or not (classification),
- how to modify box (regression)

for k "anchors" or boxes relative to the position in feature map.

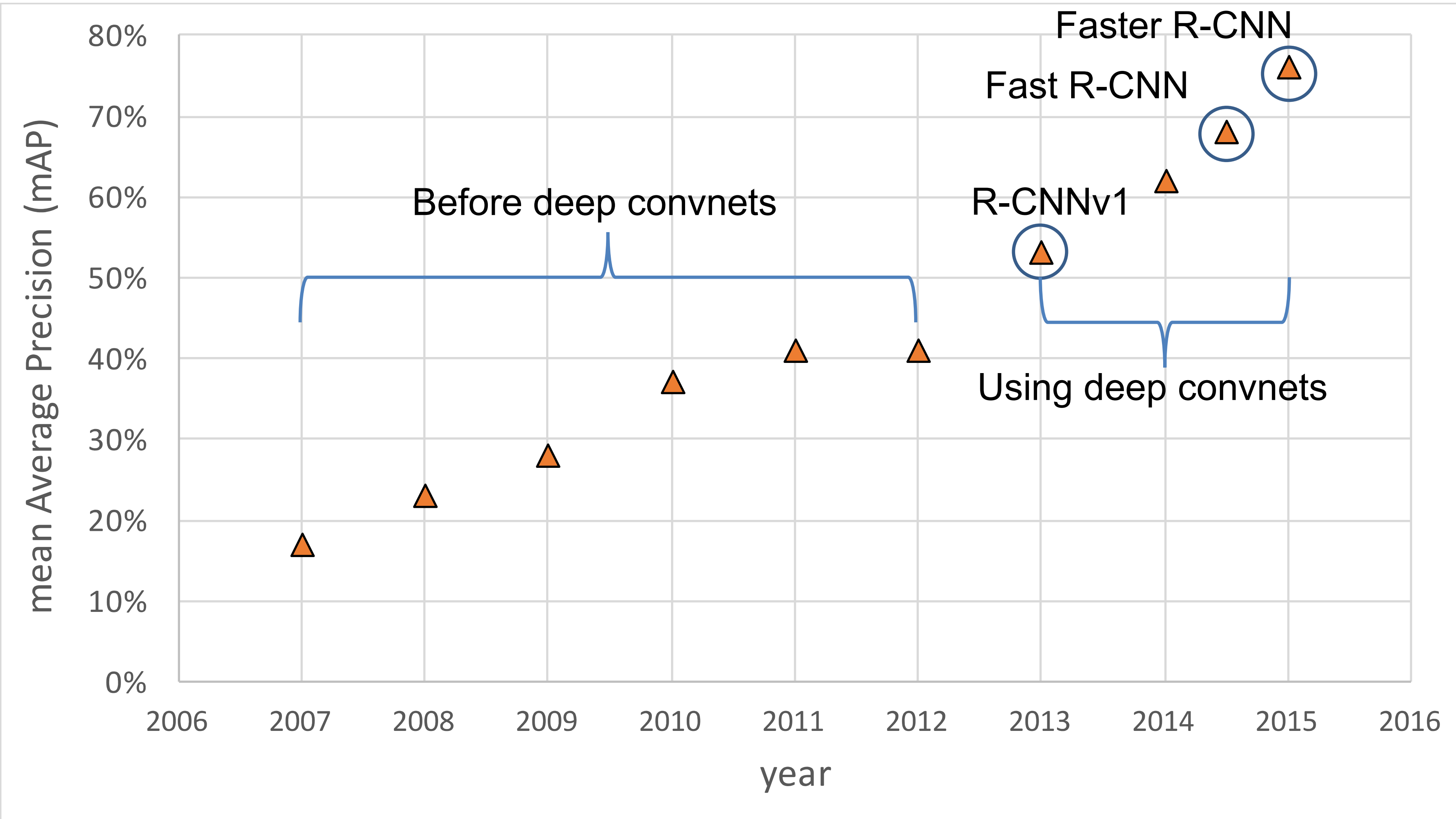
Source: R. Girshick

Faster R-CNN results

system	time	07 data	07+12 data
R-CNN	~50s	66.0	-
Fast R-CNN	~2s	66.9	70.0
Faster R-CNN	198ms	**69.9**	**73.2**

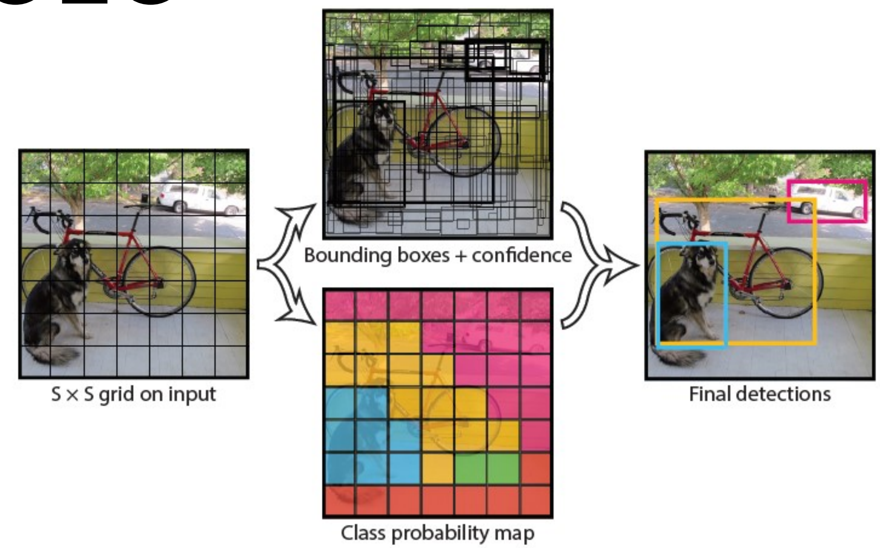
detection mAP on PASCAL VOC 2007, with VGG-16 pre-trained on ImageNet

Object detection progress

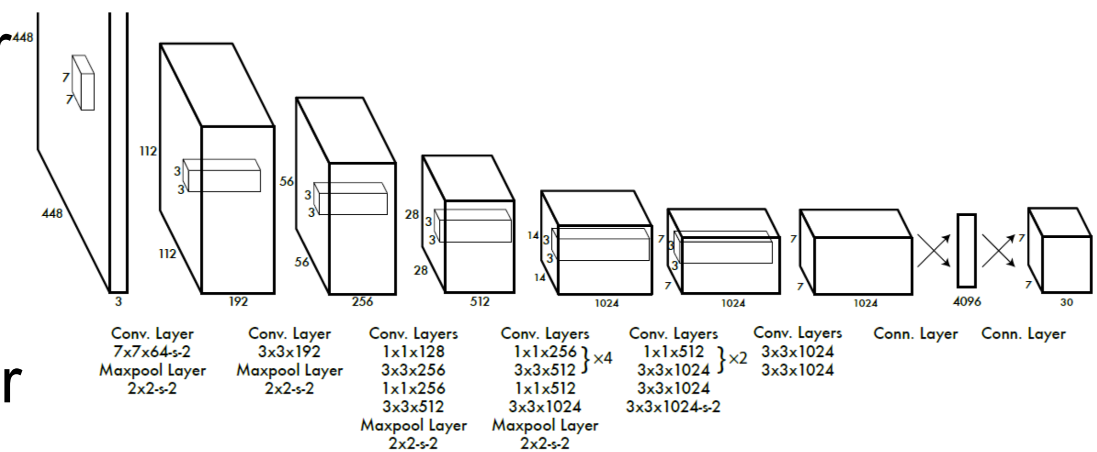


YOLO

1. Take conv feature maps at 7x7 resolution
2. Add two FC layers to predict, at each location, score for each class and 2 bboxes w/ confidences



- 7x speedup over Faster RCNN (45-155 FPS vs. 7-18 FPS)
- Some loss of accuracy due to lower recall, poor localization



J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, [You Only Look Once: Unified, Real-Time Object Detection](), CVPR 2016

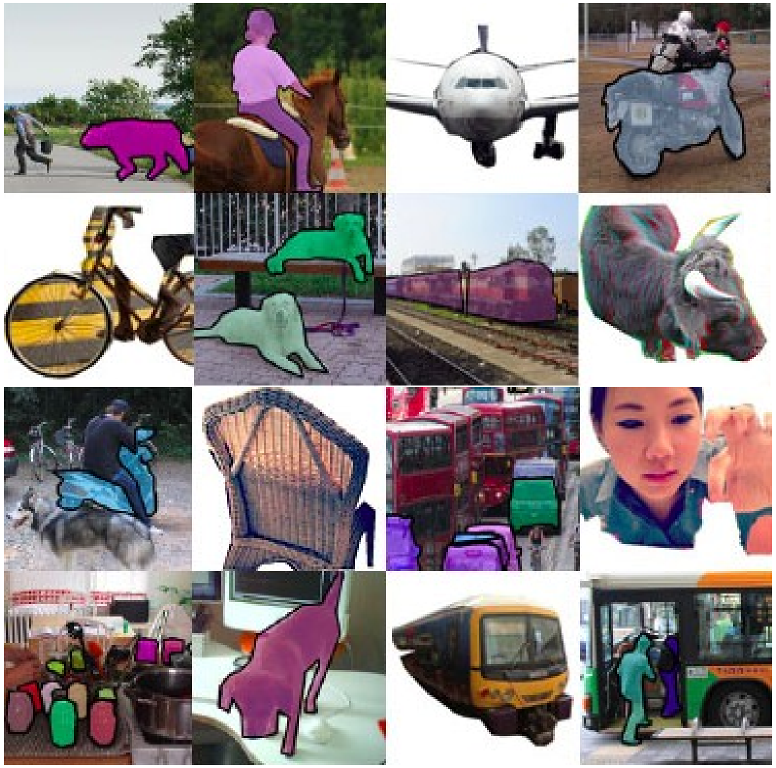
New detection benchmark: COCO (2014)

- 80 categories instead of PASCAL's 20
- Current best mAP: 52%

COCO
Common Objects in Context

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

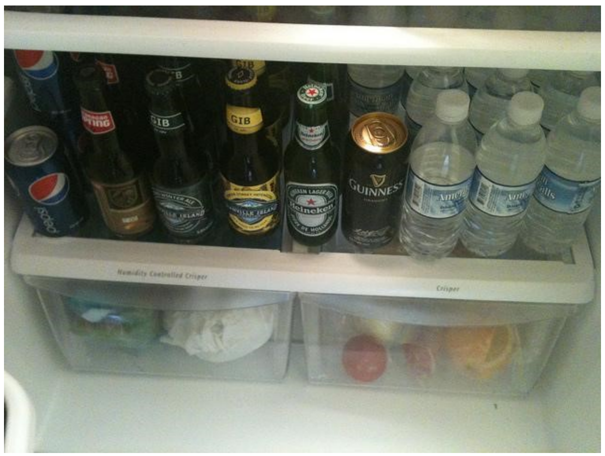
- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints



http://cocodataset.org/#home

A Few Caveats

- Flickr images come from a really weird process
- Step 1: user takes a picture
- Step 2: user decides to upload it
- Step 3: user decides to write something like “refrigerator” somewhere in the body
- Step 4: a vision person stumbles on it while searching Flickr for refrigerators for a dataset



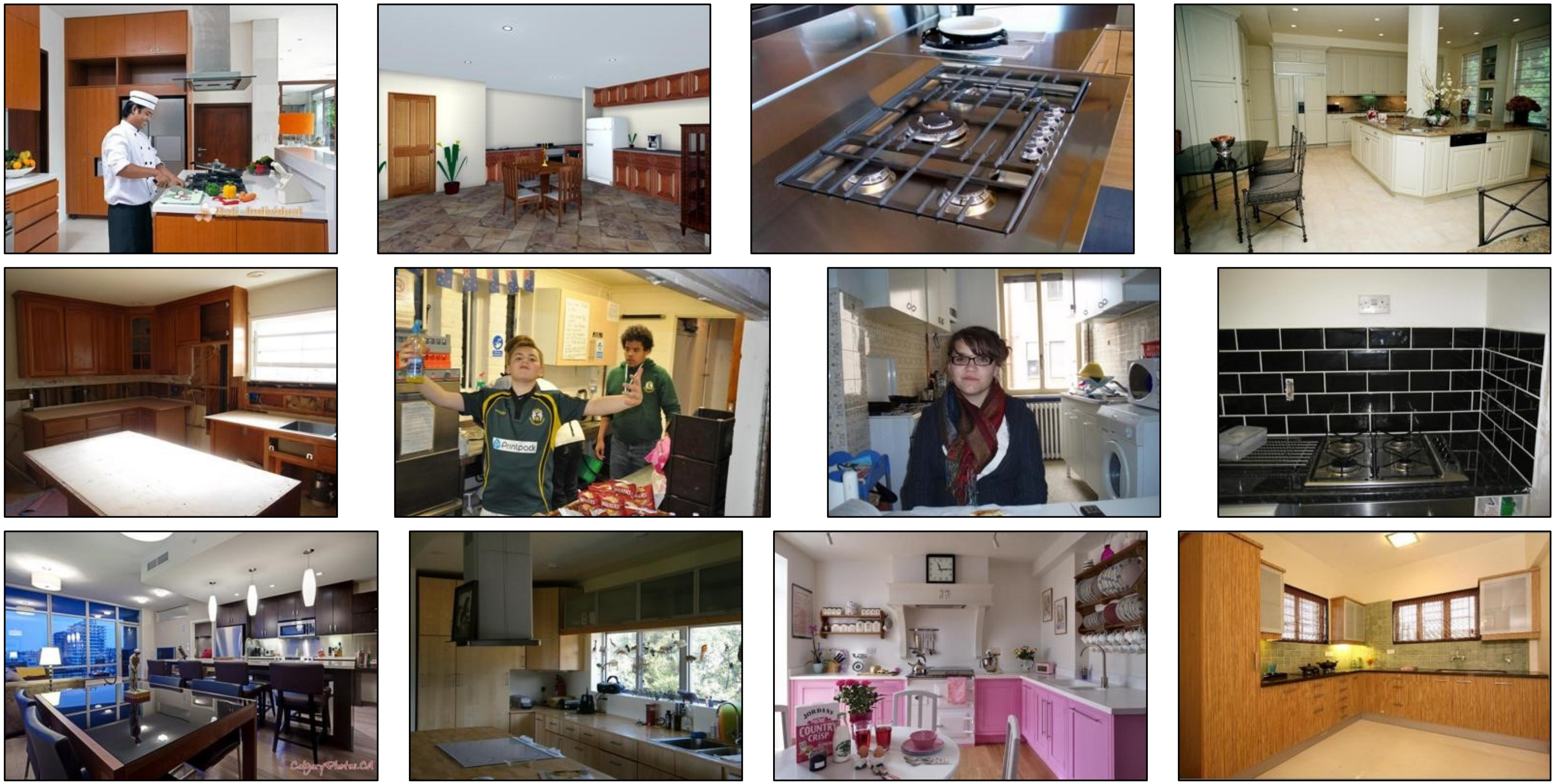
Who takes photos of open refrigerators ?????

Guess the category!

These were detected with >90% confidence, corresponding to 99% precision on original dataset

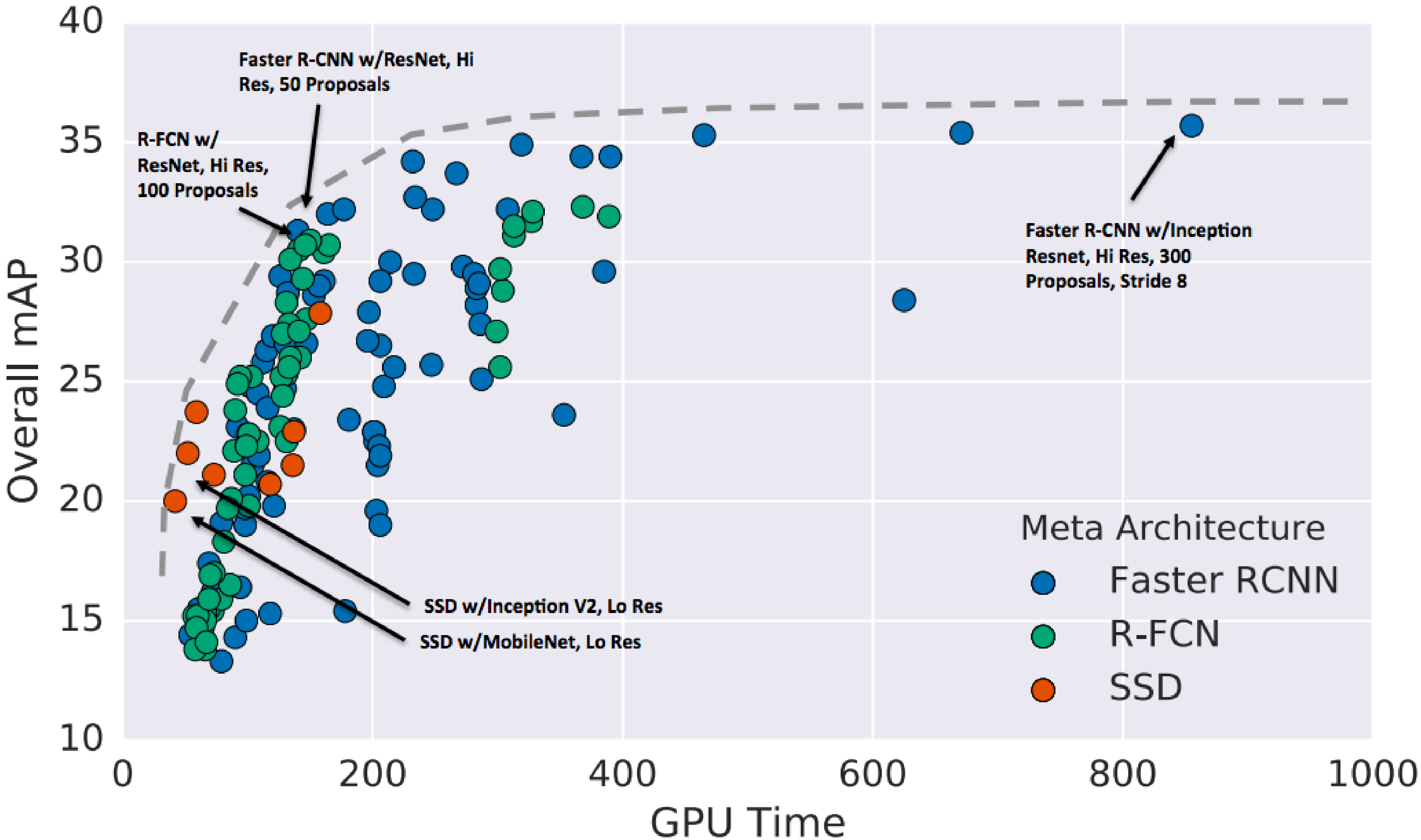
(a) Person (b) Bicycle (c) Giraffe

Kitchens from Googling



Places 365 Dataset, Zhou et al. ‘17

New detection benchmark: COCO (2014)



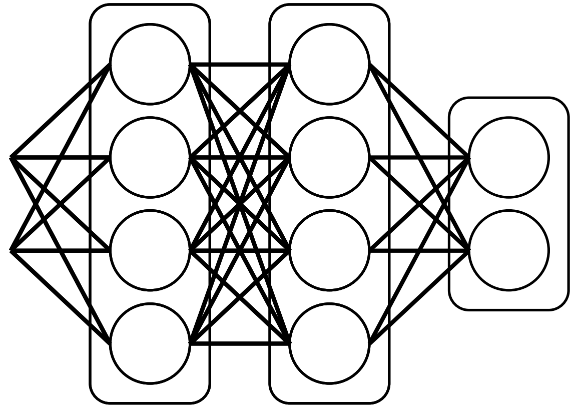
J. Huang et al., [Speed/accuracy trade-offs for modern convolutional object detectors](), CVPR 2017

Summary: Object detection with CNNs

- R-CNN: region proposals + CNN on cropped, resampled regions
- Fast R-CNN: region proposals + RoI pooling on top of a conv feature map
- Faster R-CNN: RPN + RoI pooling
- Next generation of detectors
    - Direct prediction of BB offsets, class scores on top of conv feature maps
    - Get better context by combining feature maps at multiple resolutions

And Now For Something Completely Different

ImageNet + Deep Learning

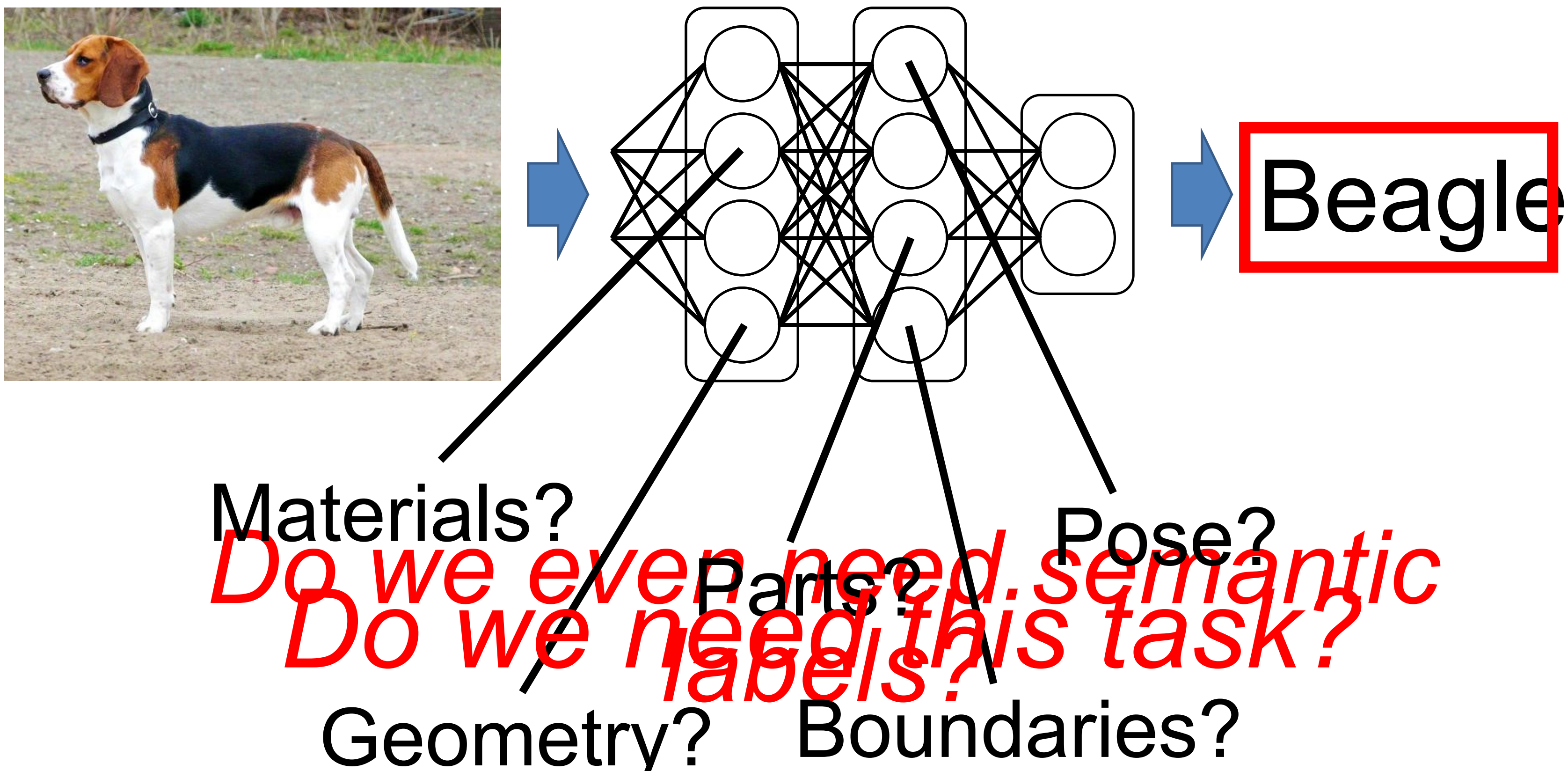


Beagle

- Image Retrieval
- Detection
- Segmentation
- Depth Estimation
- …

Slide Credit: C. Doersch

ImageNet + Deep Learning

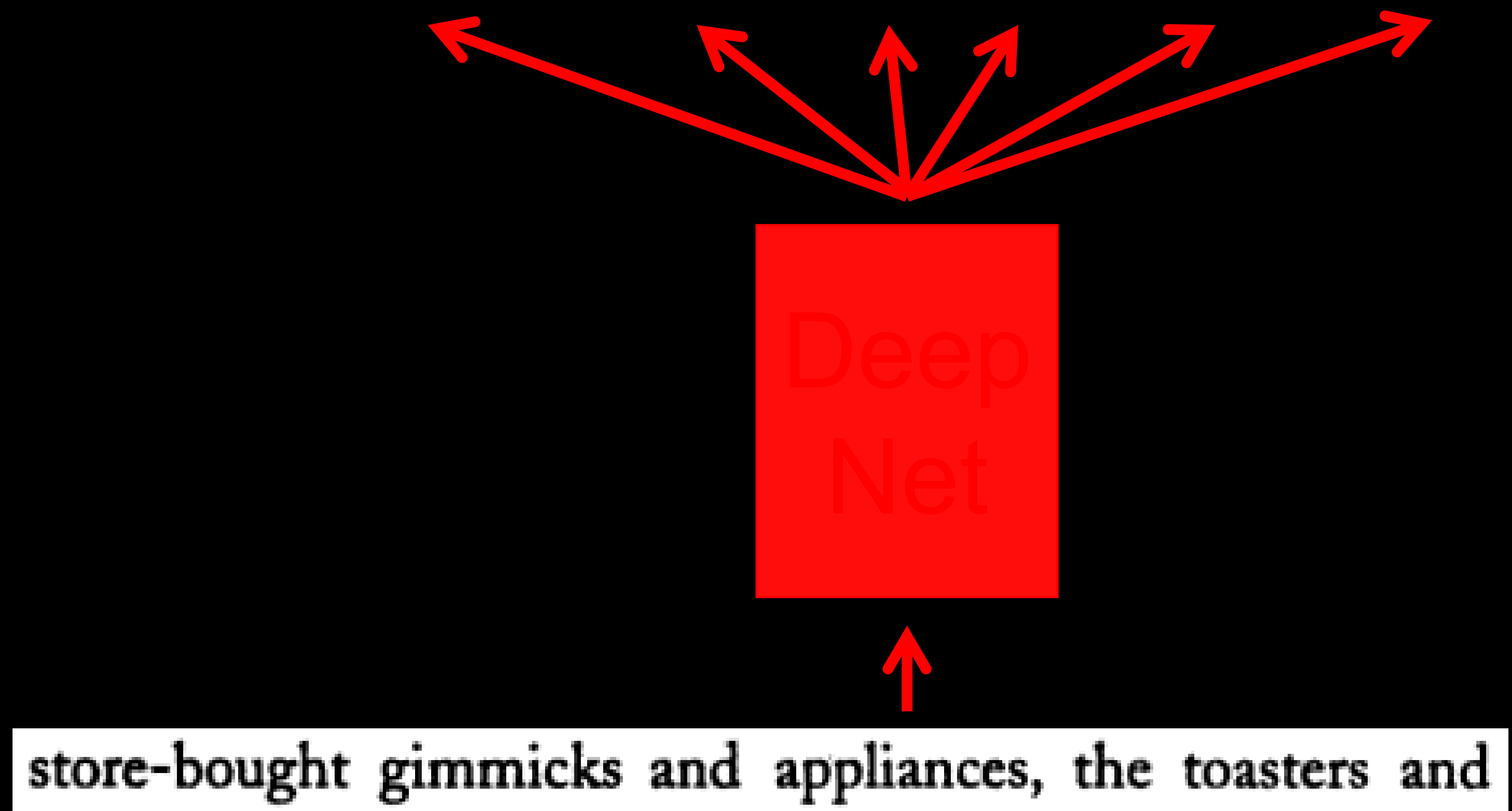


Do we even need semantic labels?

Do we need this task?

Slide Credit: C. Doersch

Context as Supervision
[Collobert & Weston 2008; Mikolov et al. 2013]

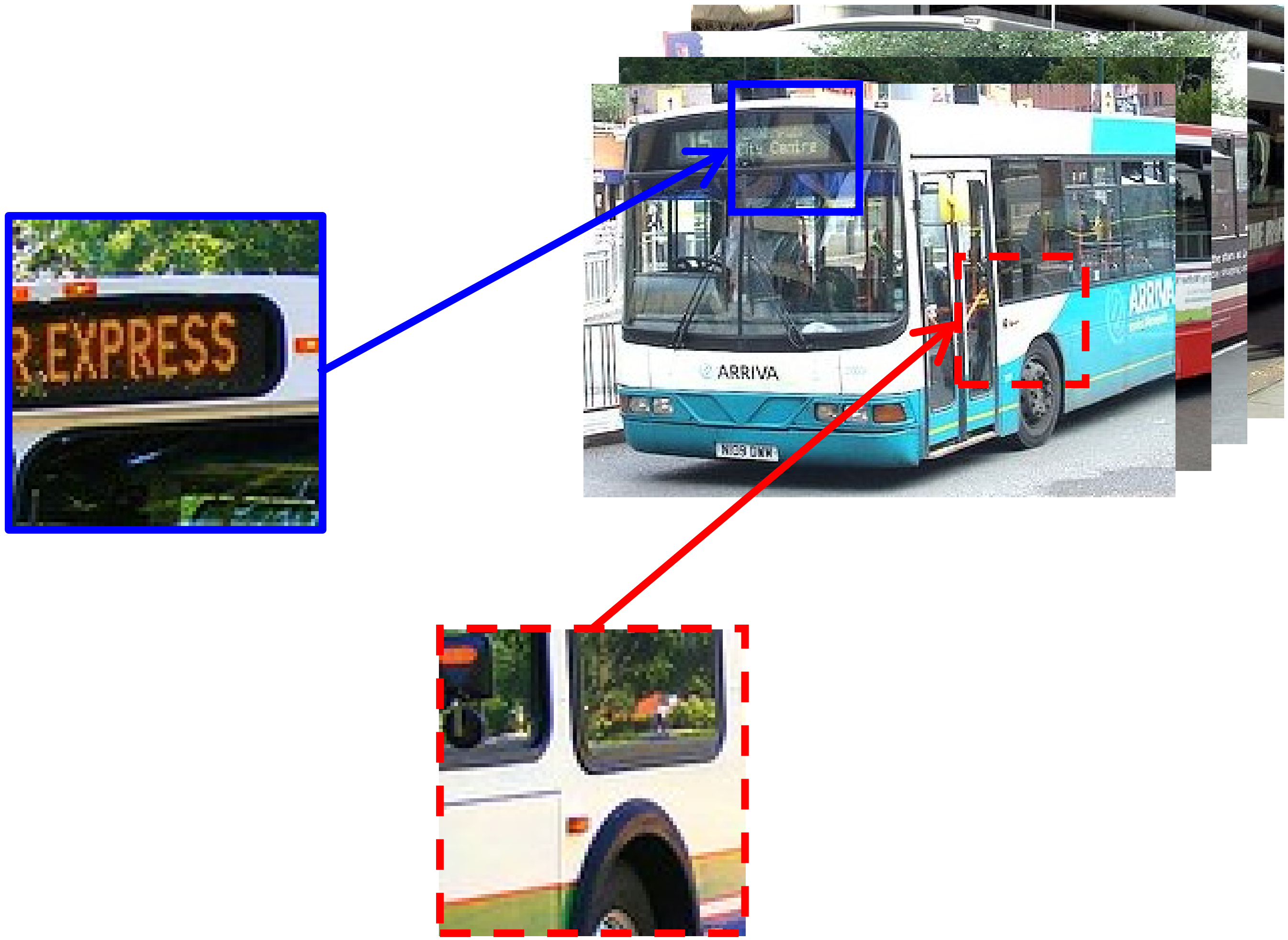


Slide Credit: C. Doersch

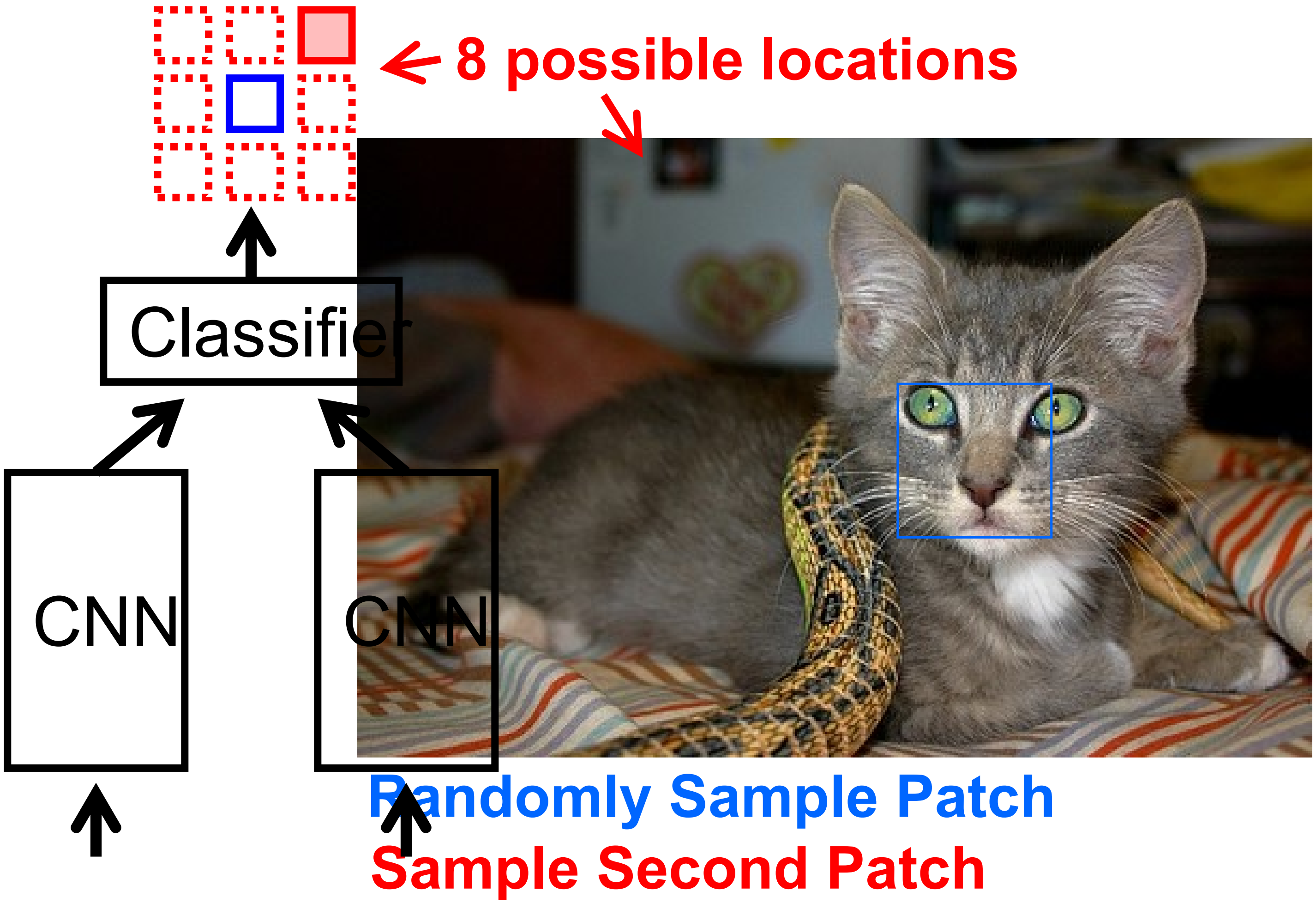
Context Prediction for Images

Slide Credit: C. Doersch

Semantics from a non-semantic task

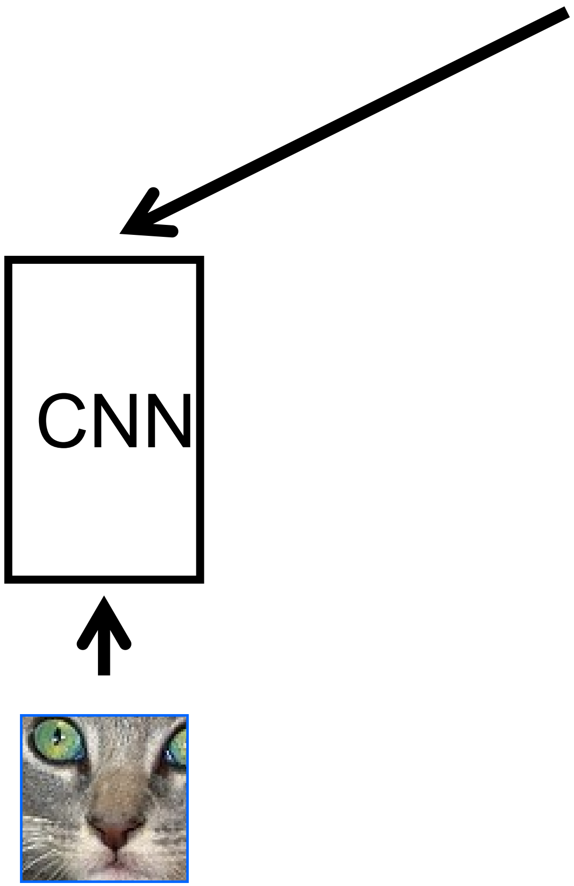


Relative Position Task



Slide Credit: C. Doersch

Patch Embedding

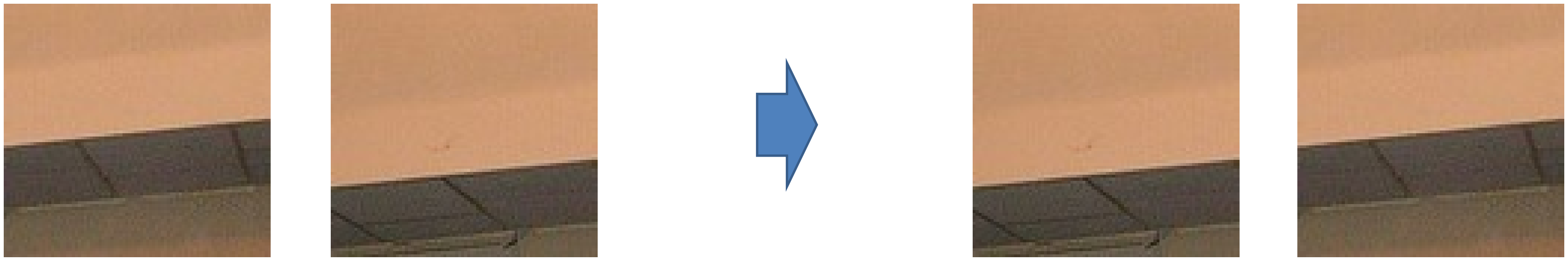


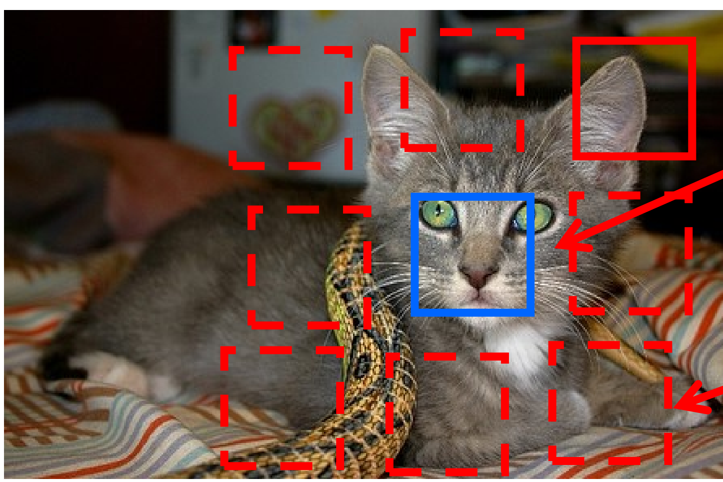
Input | Nearest Neighbors

Note: connects ***across*** instances

Slide Credit: C. Doersch

Avoiding Trivial Shortcuts



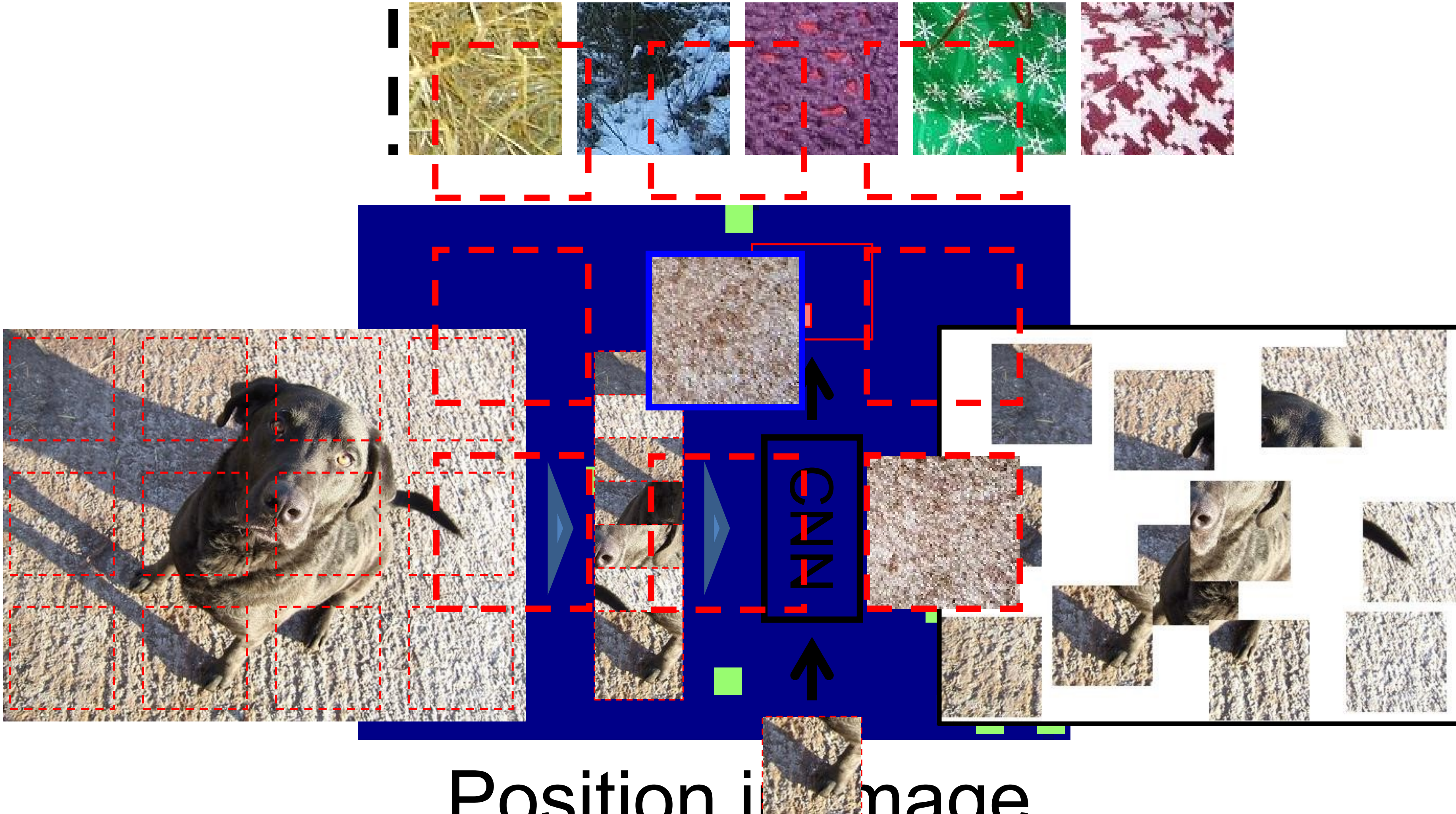


Include a gap

Jitter the patch locations

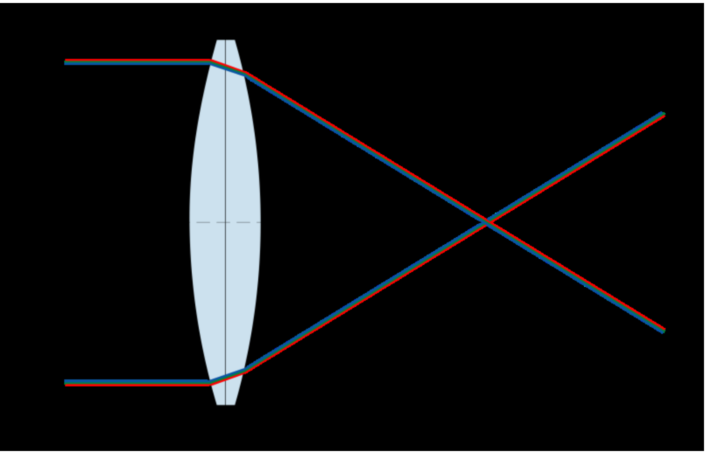
Slide Credit: C. Doersch

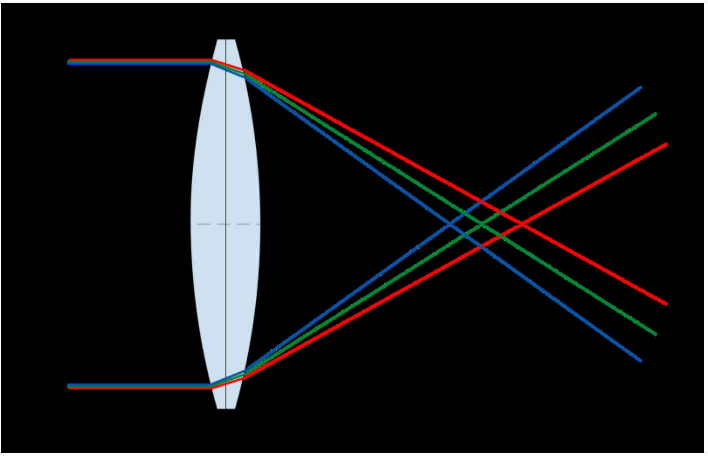
A Not-So "Trivial" Shortcut



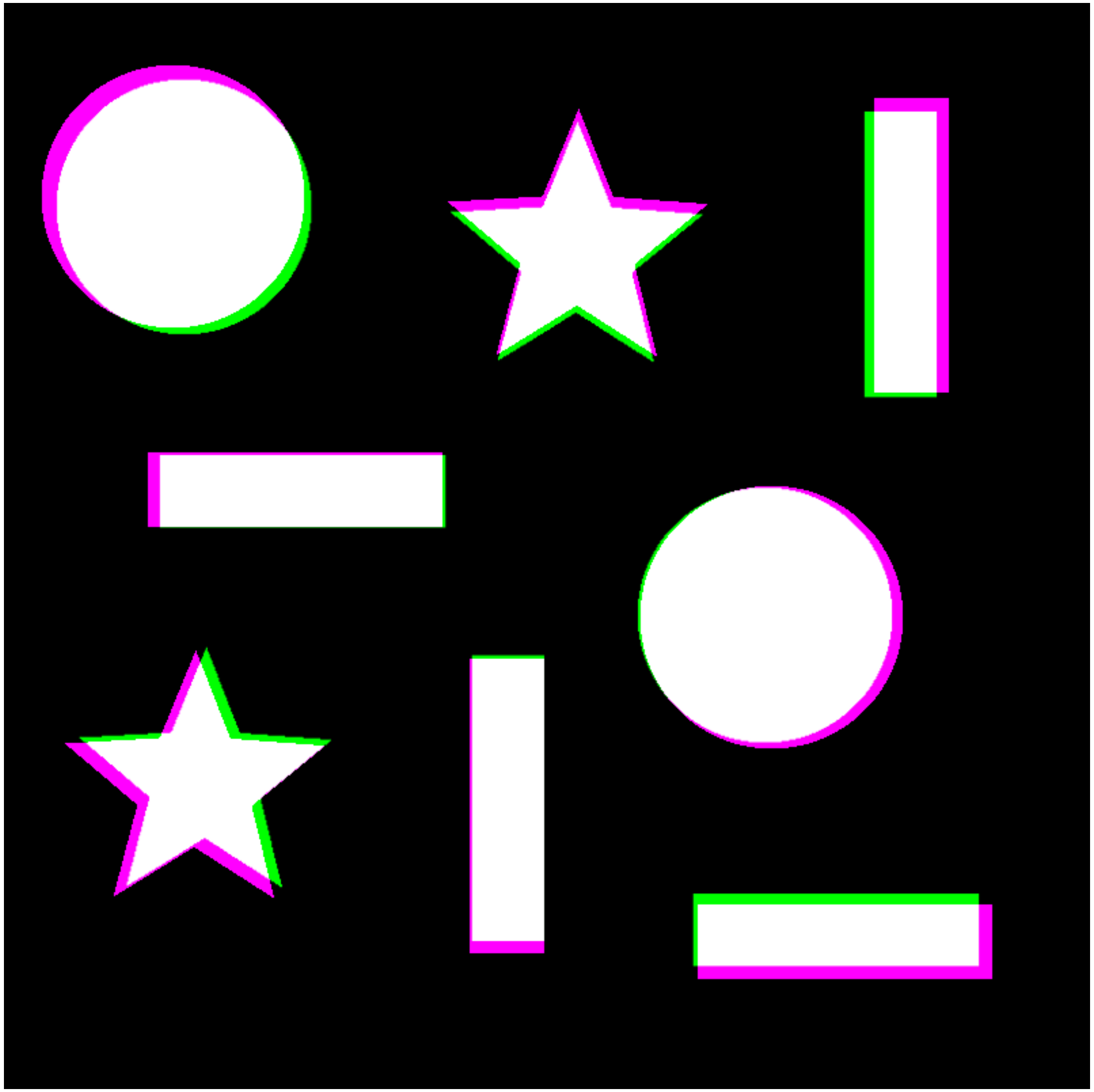
Slide Credit: C. Doersch

Chromatic Aberration

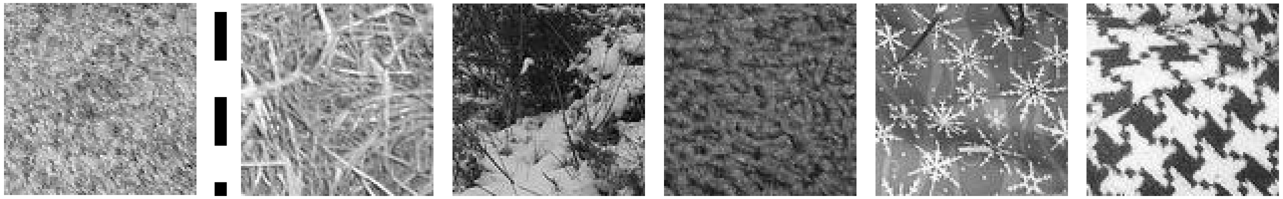


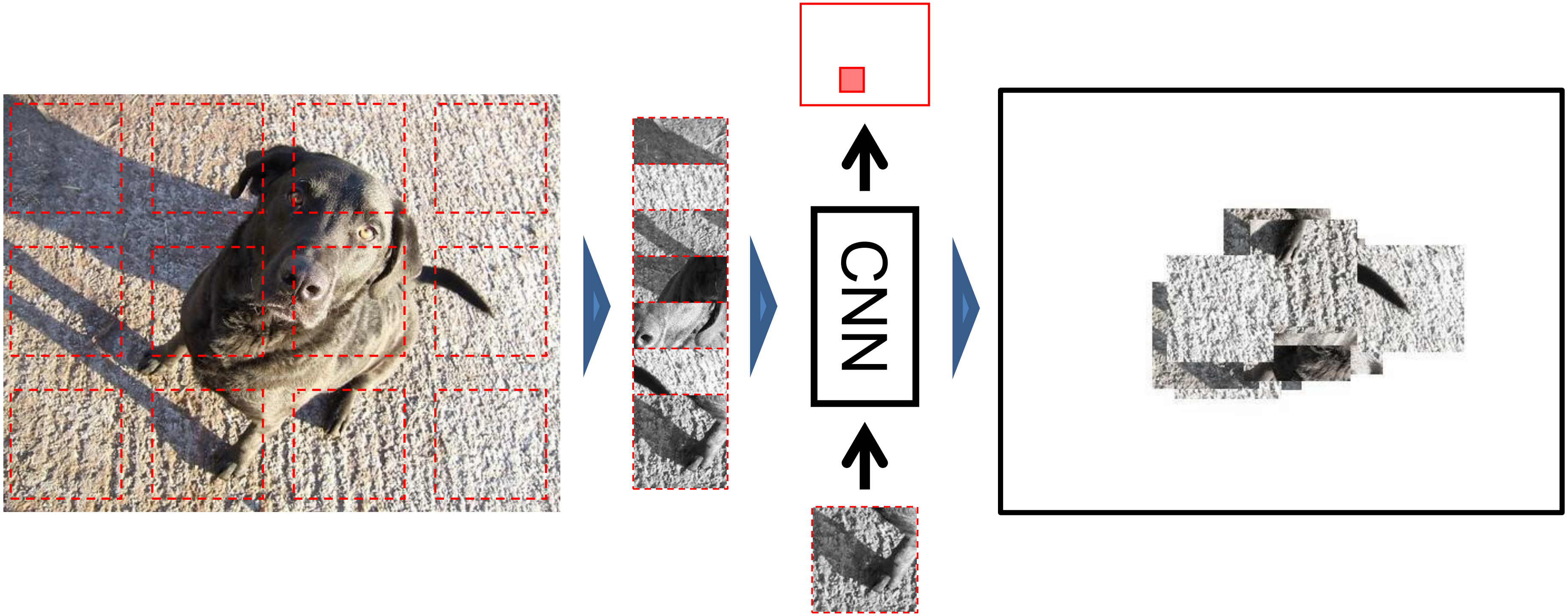


Slide Credit: C. Doersch



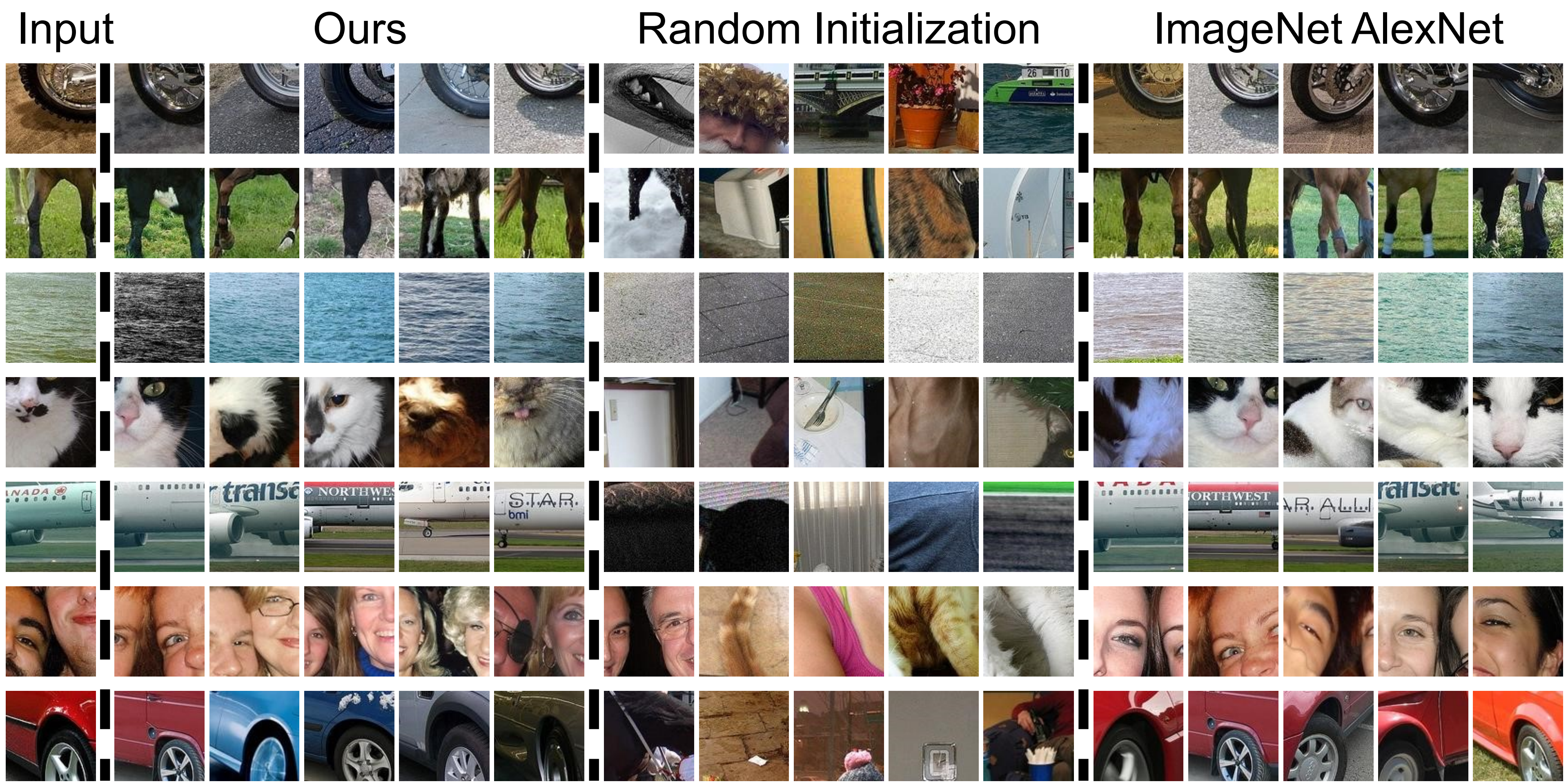
Chromatic Aberration





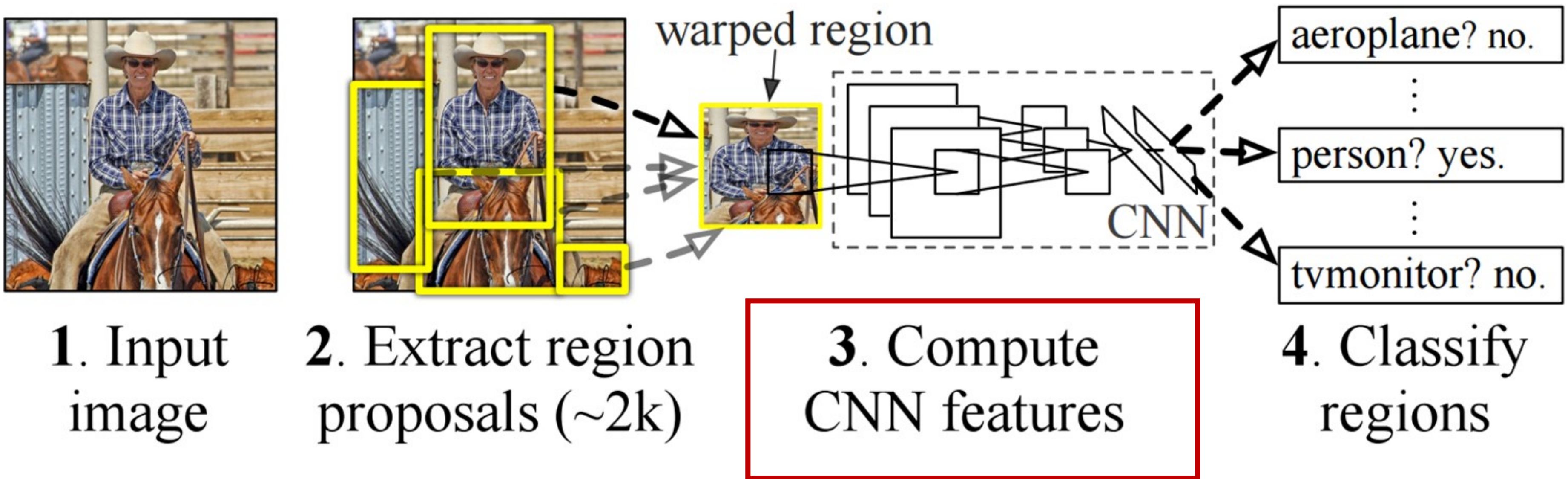
Slide Credit: C. Doersch

What is learned?



Slide Credit: C. Doersch

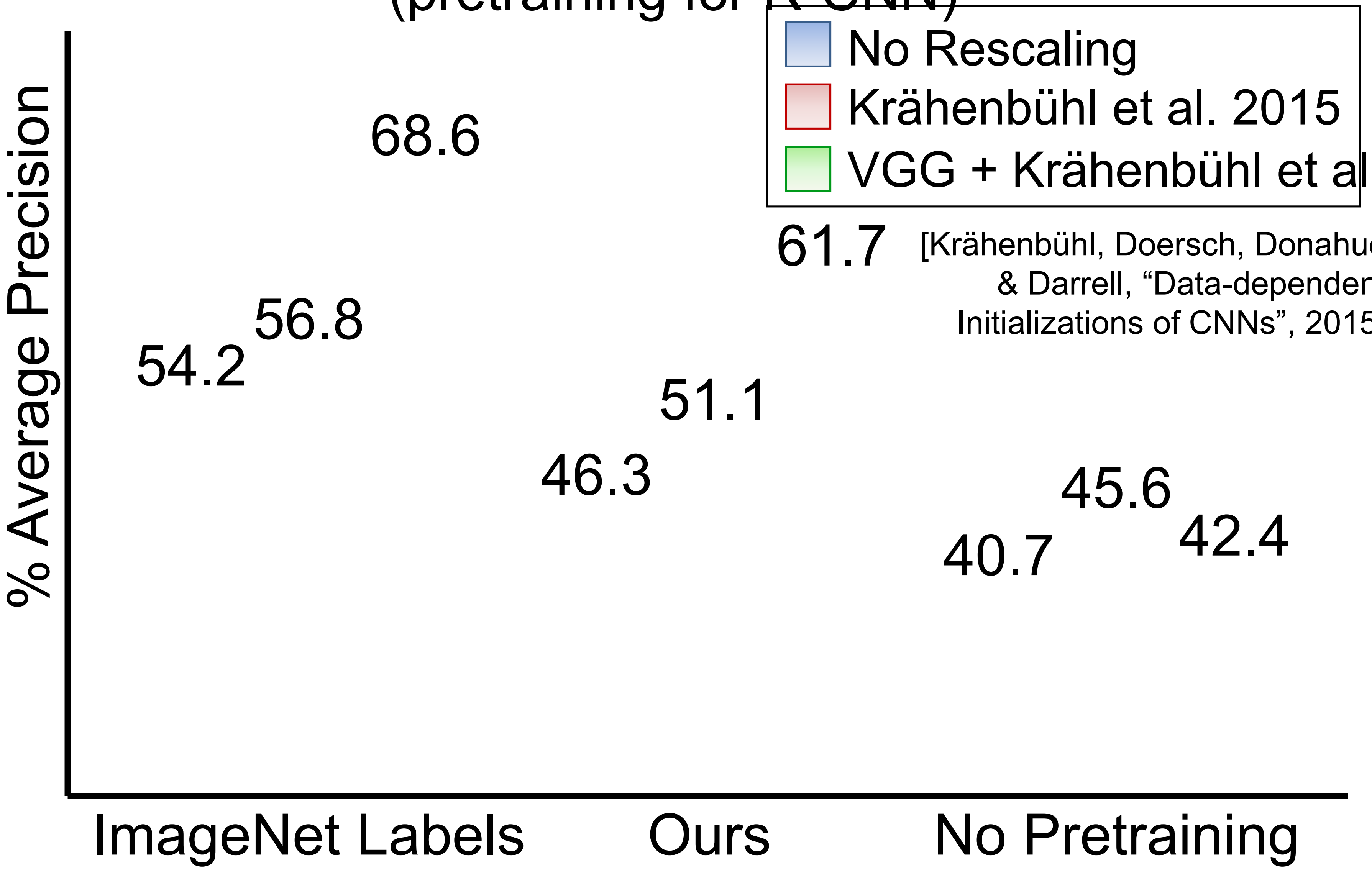
Pre-Training for R-CNN



Pre-train on relative-position task, w/o labels

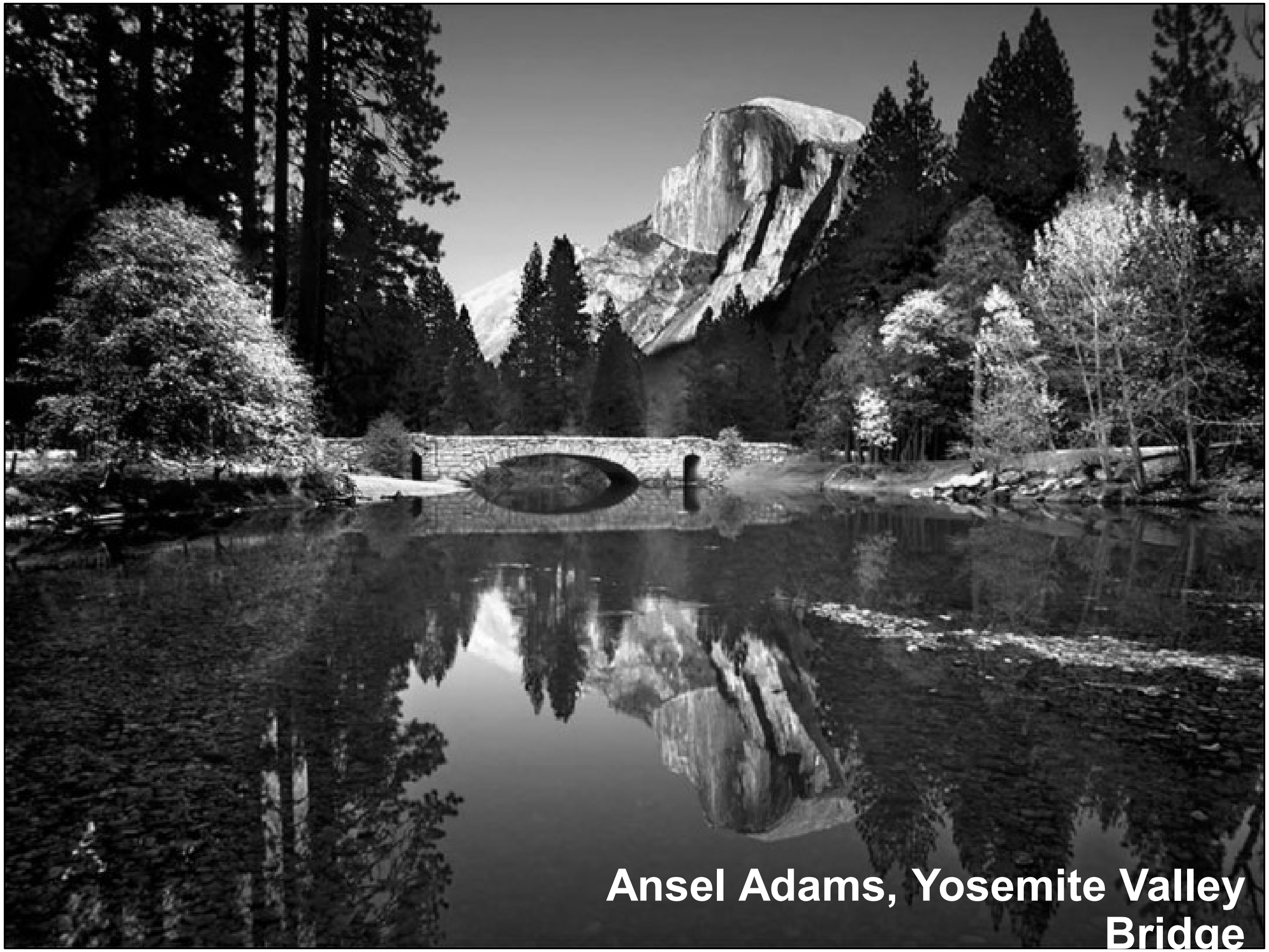
[Girshick et al. 2014]

VOC 2007 Performance
(pretraining for R-CNN)

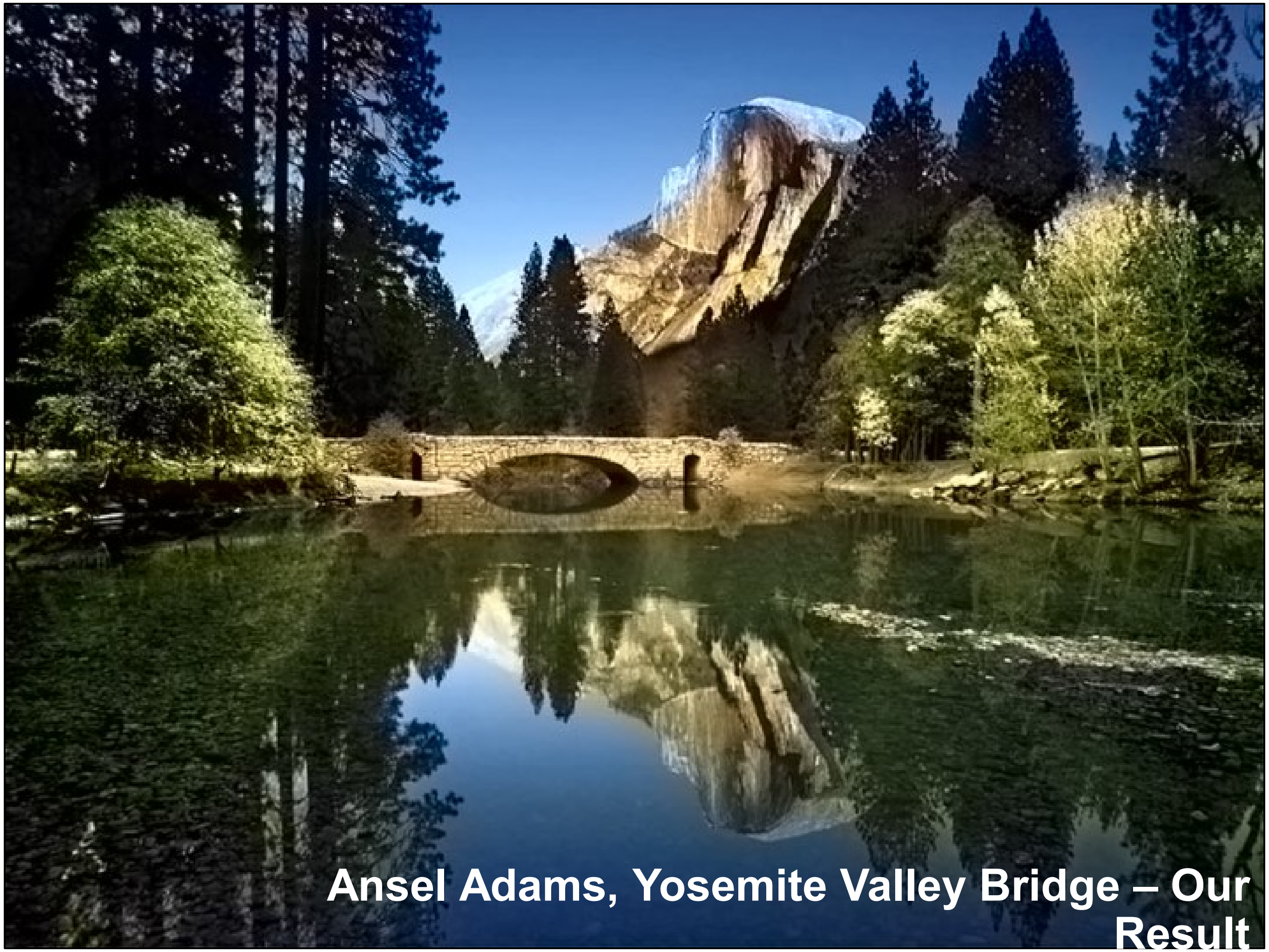


[Krähenbühl, Doersch, Donahue & Darrell, "Data-dependent Initializations of CNNs", 2015]

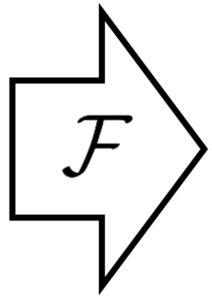
Other Sources Of Signal



Slide Credit: R. Zhang



Slide Credit: R. Zhang

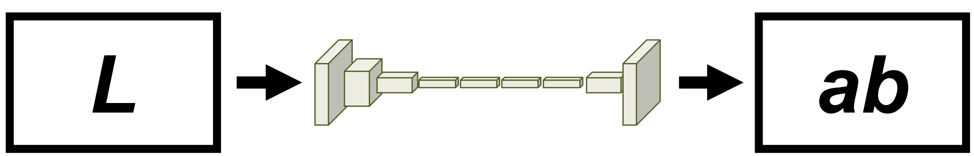


Grayscale image: *L* channel

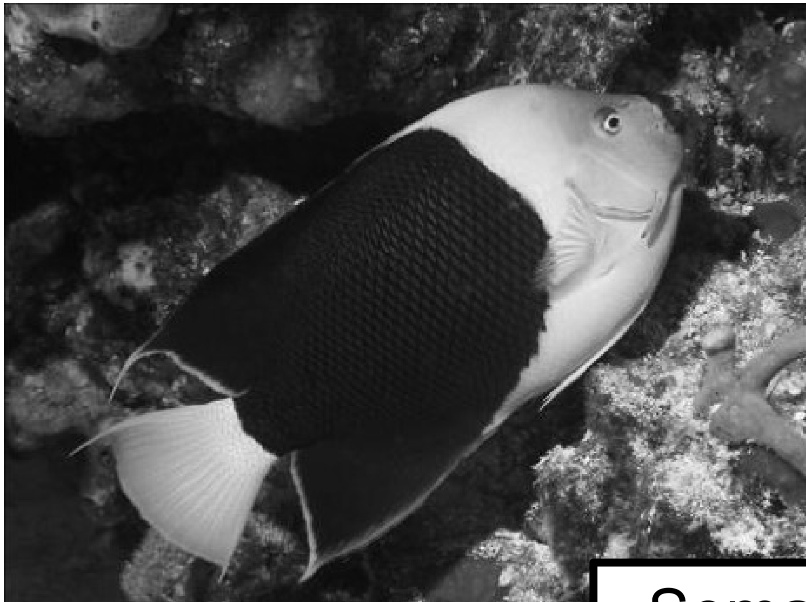
$$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$$

Color information: *ab* channels

$$\widehat{\mathbf{Y}} \in \mathbb{R}^{H \times W \times 2}$$



Slide Credit: R. Zhang



$\mathcal{F}$

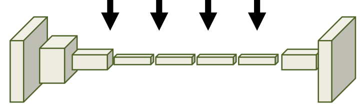
Grayscale image: *L*

$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$

Semantics? Higher-level abstraction?

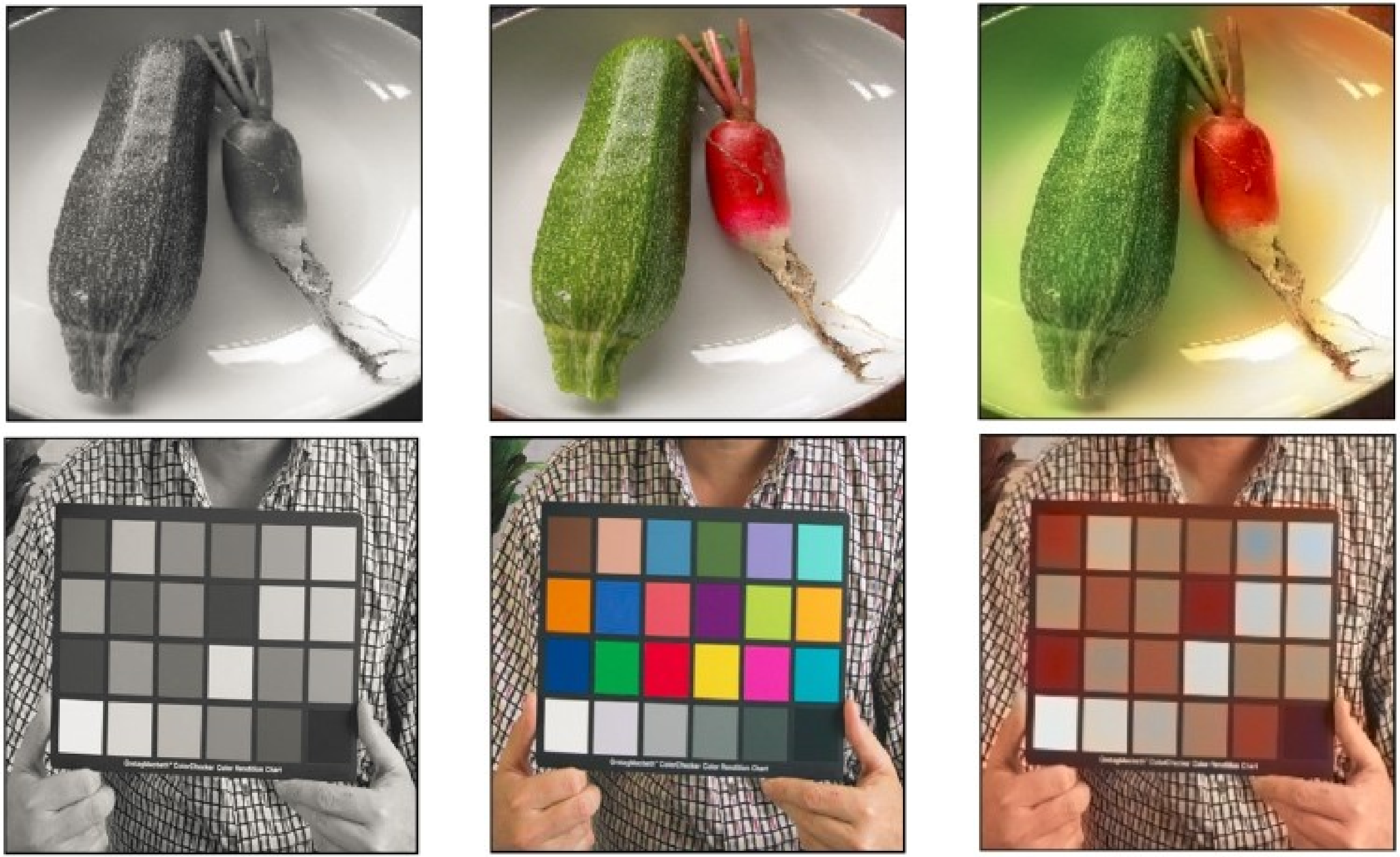
oncatenate (L,ab)

$(\mathbf{X}, \widehat{\mathbf{Y}})$

L →  → ***ab*** ← "Free" supervisory signal

Slide Credit: R. Zhang

Input **Ground Truth** **Output**



Slide Credit: R. Zhang


Visually Indicated Sounds
Andrew Owens
Phillip Isola
Josh McDermott
Antonio Torralba
Edward Adelson
William Freeman

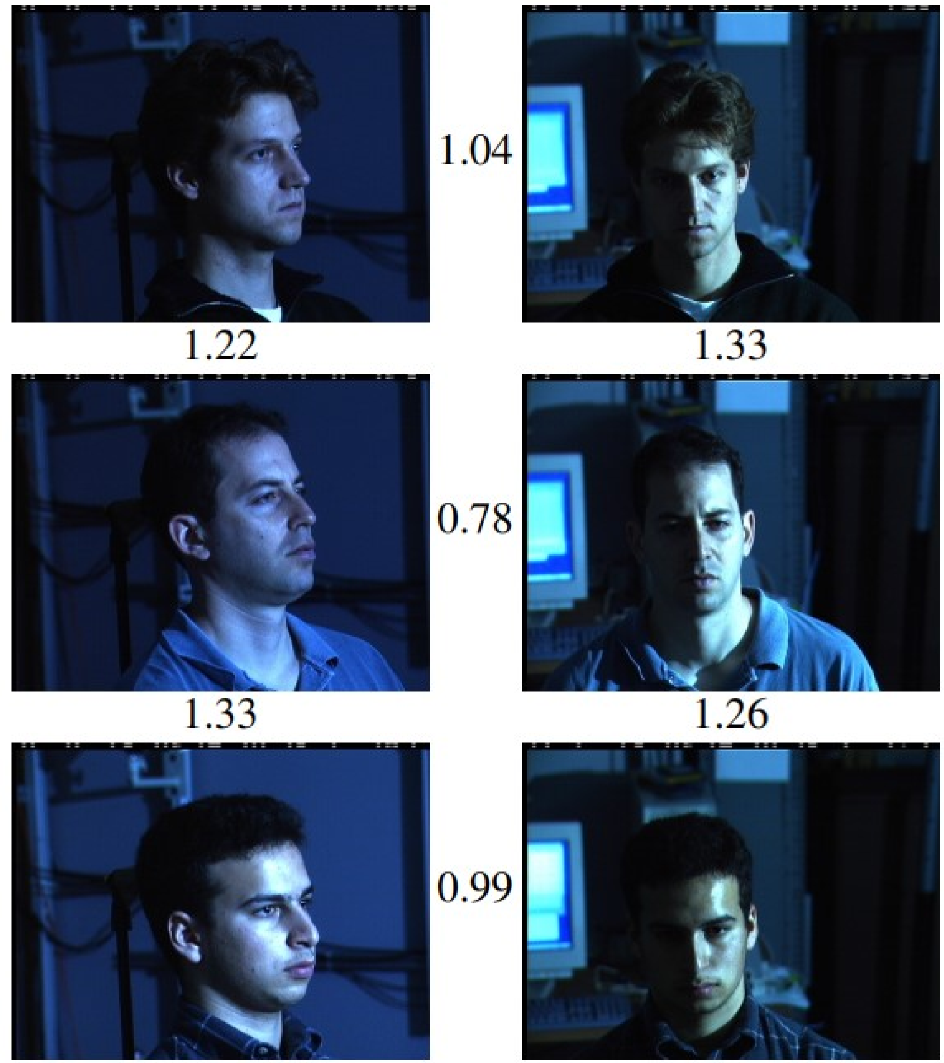
Face Recognition

- Some goals for face/person/recognition:
  - Should be able to recognize lots of people
  - Shouldn’t require lots of data per-person
  - Should be able to recognize a new person post-training
- Classification doesn’t handle any of these well.

Face Recognition

Goal: given images, would like to be able to compute distances between the images.

Face recognition is then: given reference image, is this person the same as the reference image (here: < 1.1)

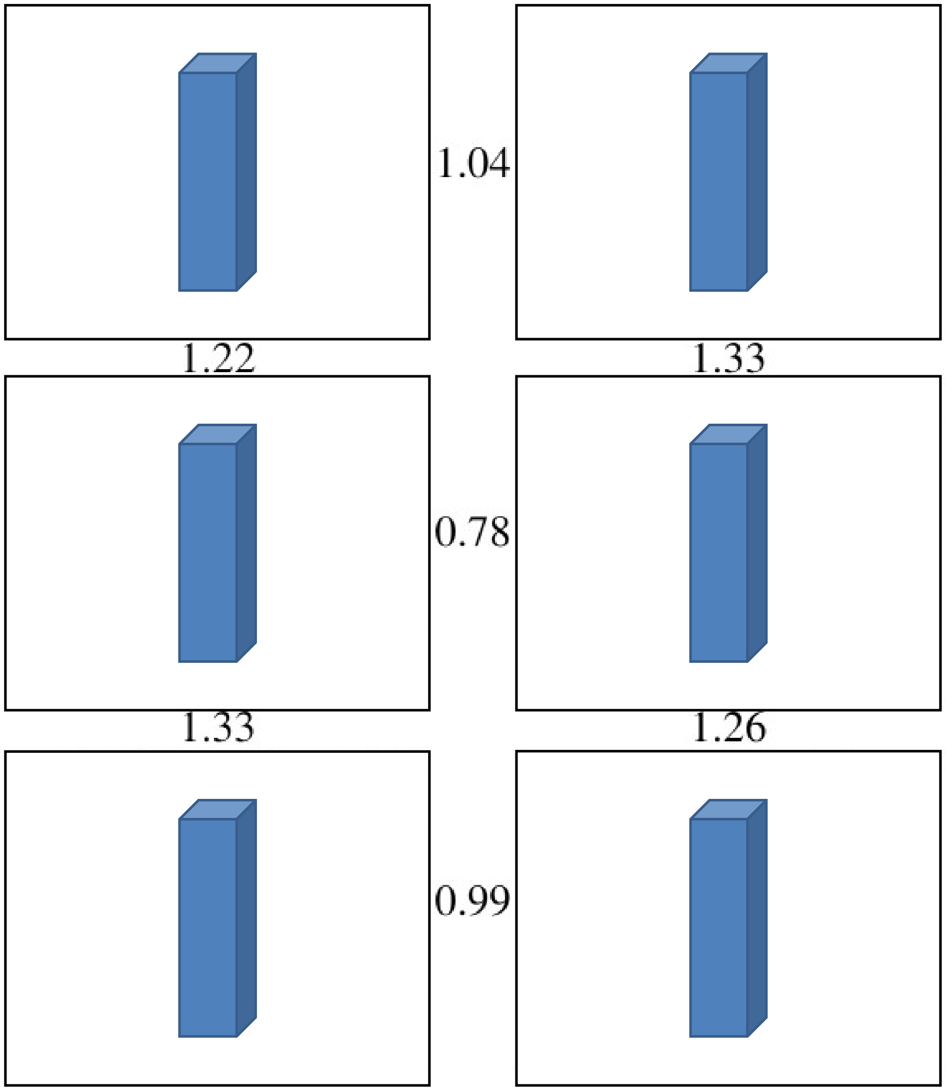


Face Recognition

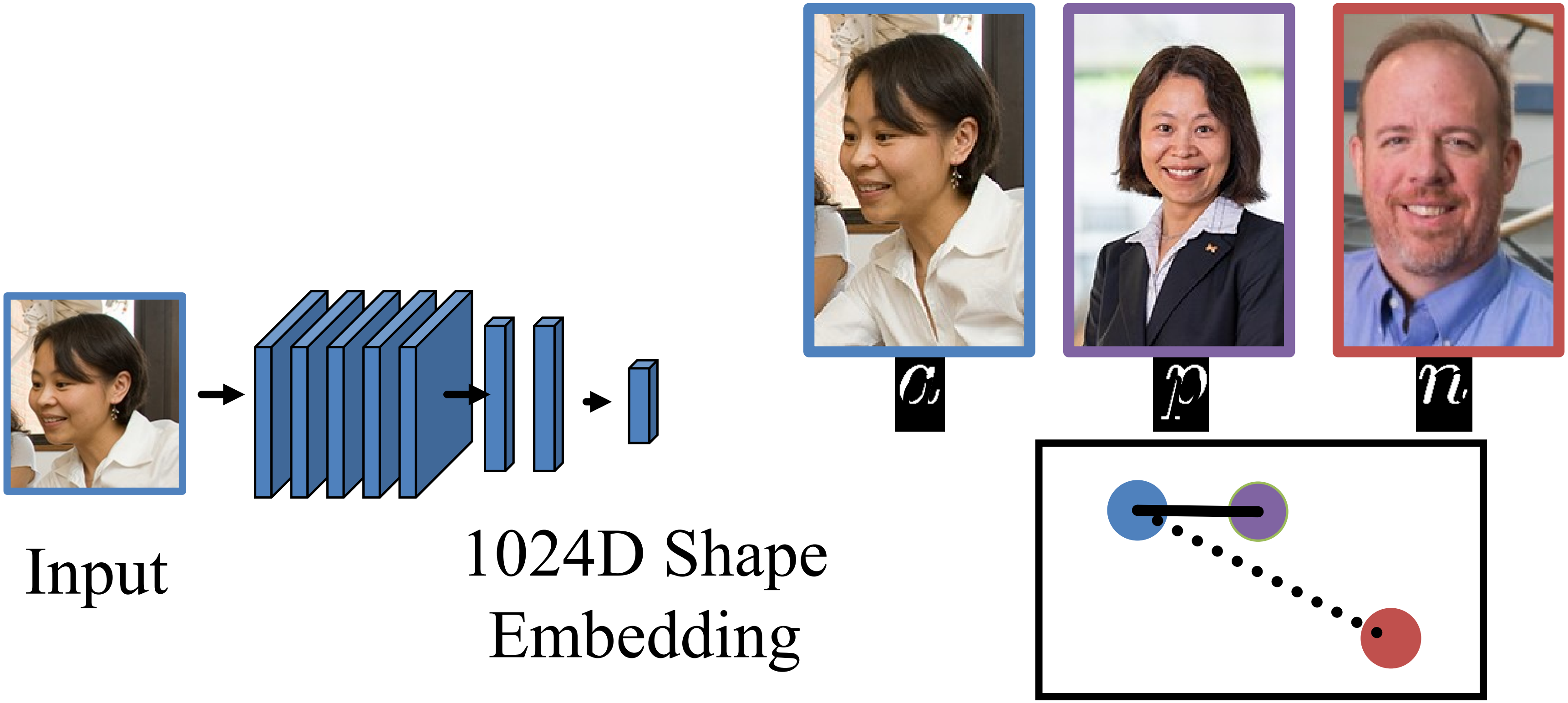
Distances between images are not meaningful.

Need to convert image into vector (typically normalized to unit sphere)

How?



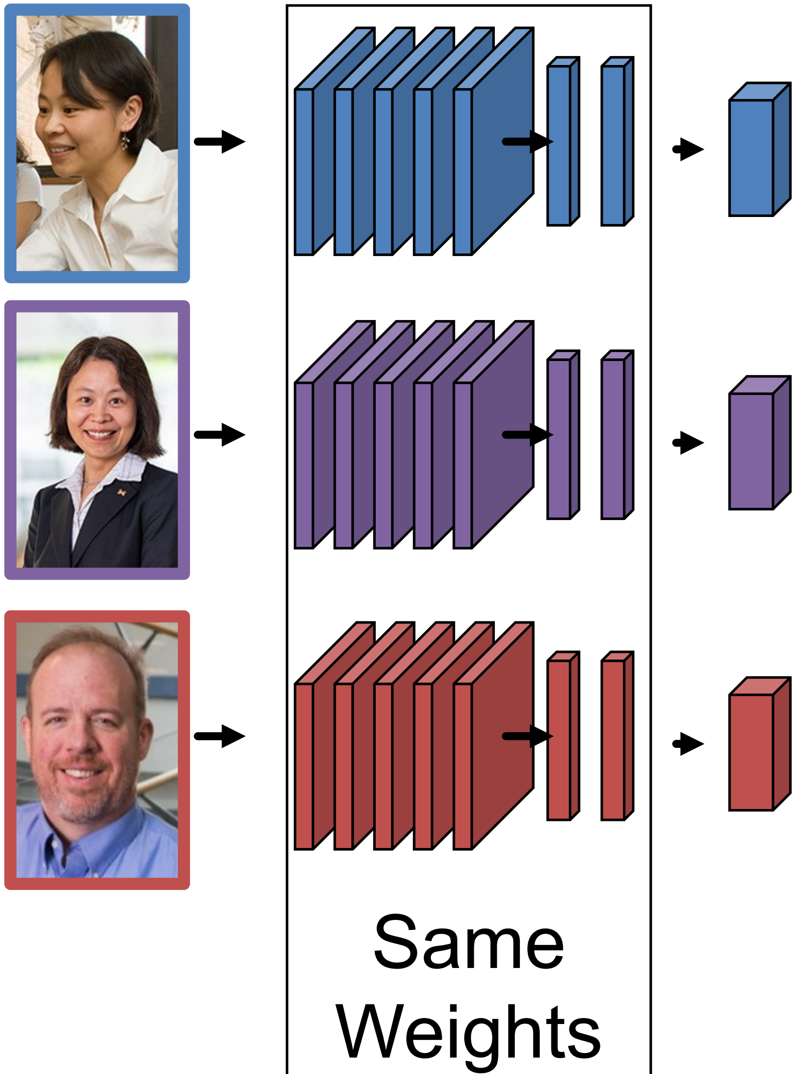
Triplet Network

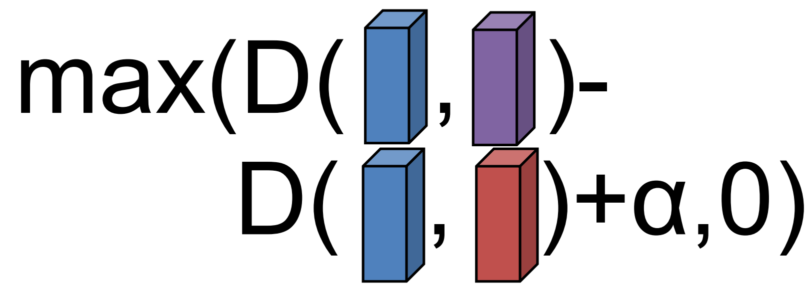


$$L(a, p, n) = \max(D(a, p) - D(a, n) + \alpha, 0)$$

Diagram credit: Fouhey et al. *3D Shape Attributes.* CVPR 2016.

Training





Idea: Pass all three images through same network (e.g., in same batch).

In Practice

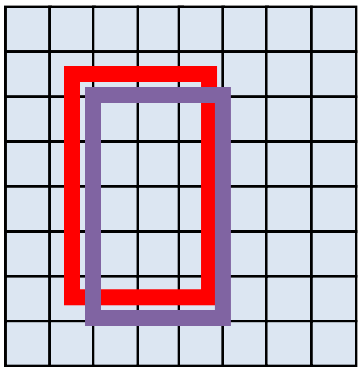
- Picking triplets <u>crucial</u>:
  - Lots of easy triplets (e.g., me and Shaquille O'Neal) – don't learn anything
  - Only hard triplets (e.g., me and my doppleganger) – training fails since it's too hard
  - During training, some of the worst mistakes in practice (triplets with highest loss) are often just mislabeling mistakes
- More generally called "Metric Learning"
- Lots of applications beyond face recognition

Next Time

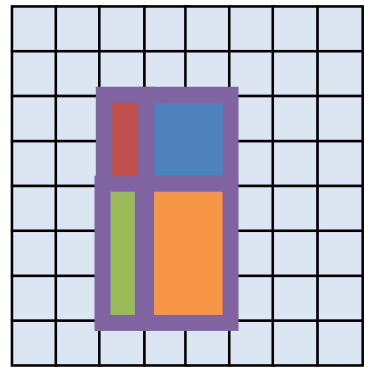
- Video

Extra Stuff

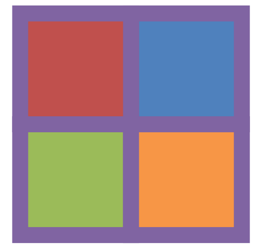
Fast R-CNN – ROI-Pool



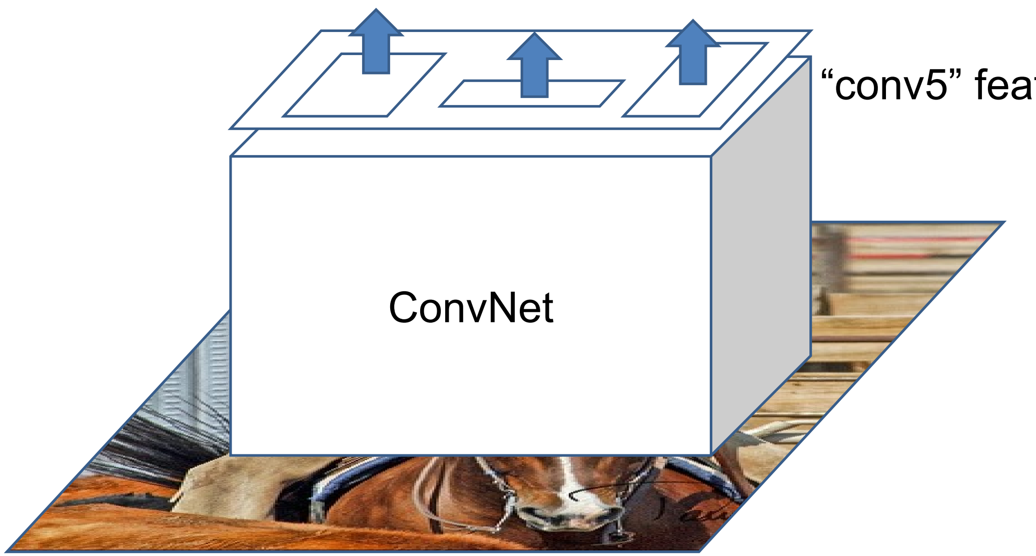
Line up



Divide



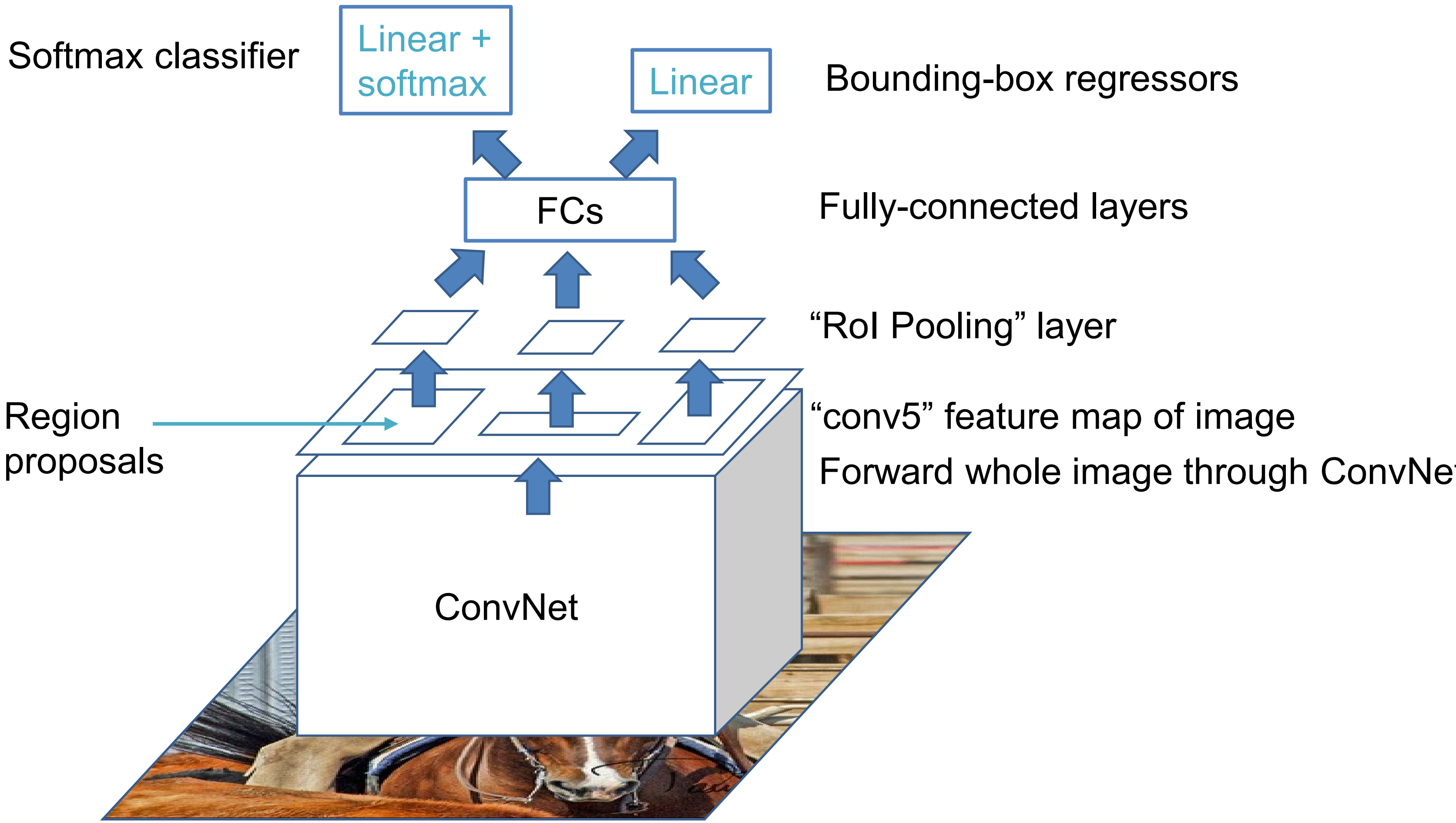
Pool



Source: R. Girshick

R. Girshick, Fast R-CNN, ICCV 2015

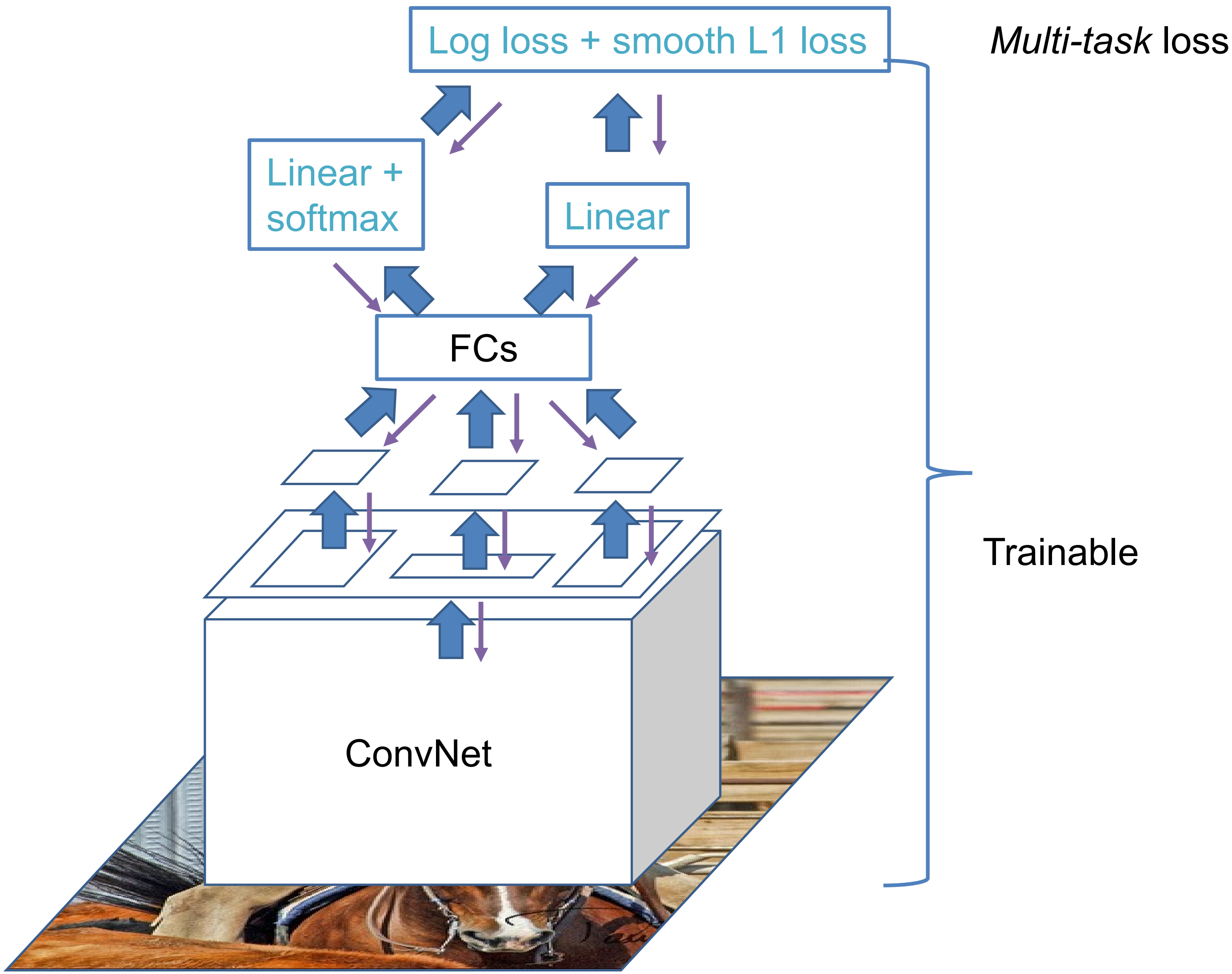
Fast R-CNN



Source: R. Girshick

R. Girshick, [Fast R-CNN](), ICCV 2015

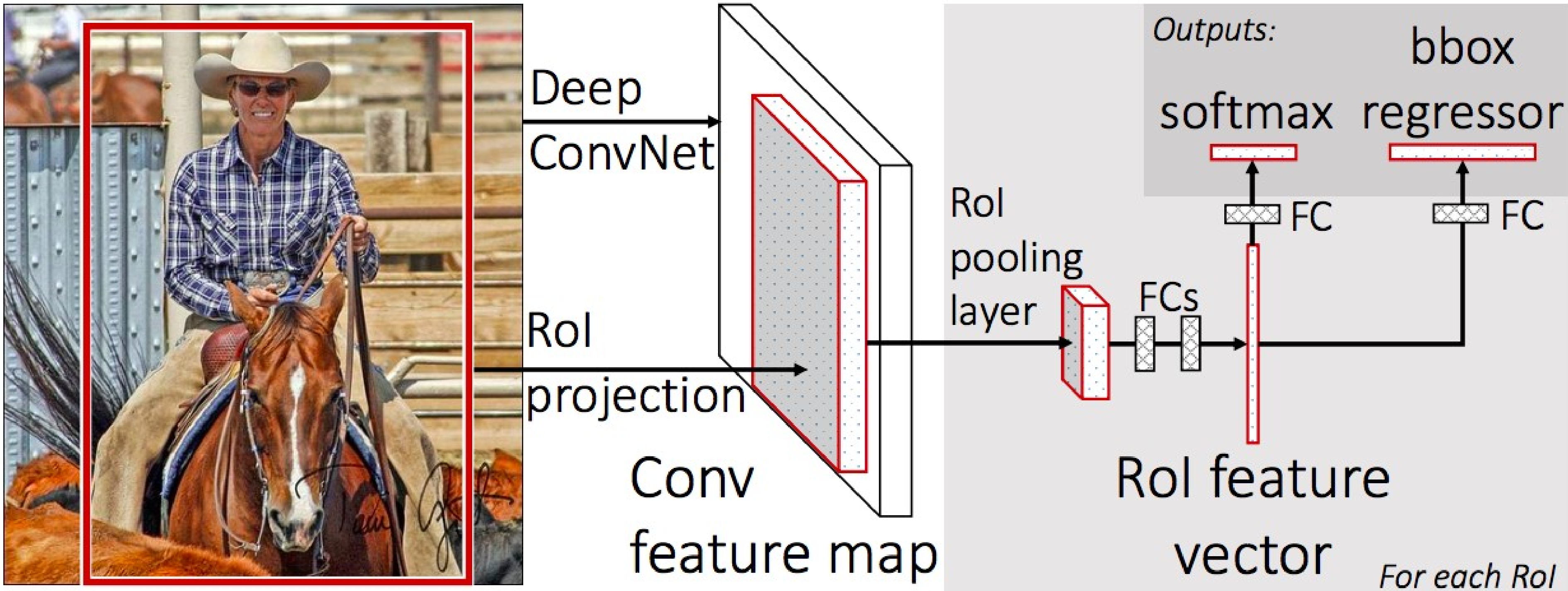
Fast R-CNN training



Source: R. Girshick

R. Girshick, [Fast R-CNN](), ICCV 2015

Fast R-CNN: Another view



Source: R. Girschick

R. Girshick, [Fast R-CNN](), ICCV 2015

Fast R-CNN results

	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.

Source: R. Girshick