

**CMP717**

**Image Processing**

# **Semantic Segmentation**

Erkut Erdem  
Hacettepe University  
Computer Vision Lab (HUCVL)

# Semantic Segmentation

- Joint recognition & segmentation
  - segmenting all the objects in a given image and identifying their visual categories
- aka scene parsing or image parsing
- Early studies aim at segmenting out a single object of a known category
  - Borenstein & Ullman, 2002, Liebe & Schiele, 2003, etc.
- More recent work depends on CNNs
  - Farabet et al., 2013, Pinheiro and Collobert, 2014, Long et al., 2015, Noh et al., 2015

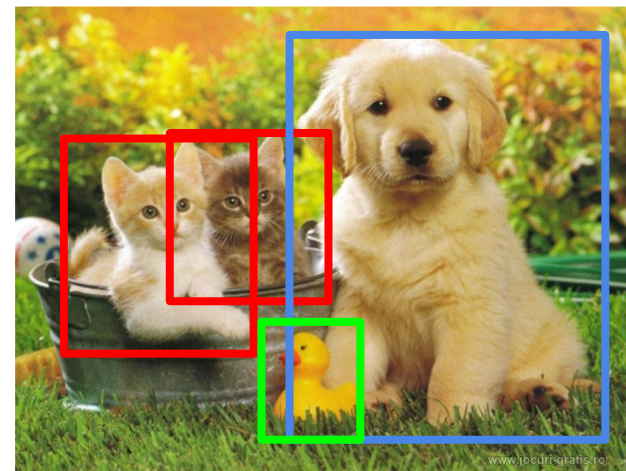
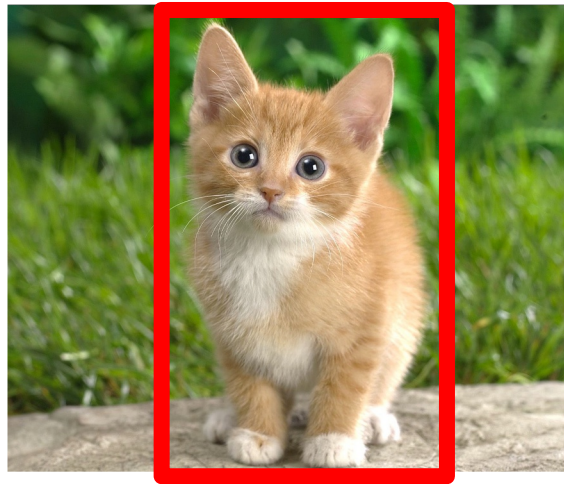
# Computer Vision Tasks

Classification

Classification  
+ Localization

Object  
Detection

Semantic  
Segmentation



CAT

CAT

CAT, DOG, DUCK

CAT, DOG, DUCK

Single  
object

Multiple  
objects

# Computer Vision Tasks

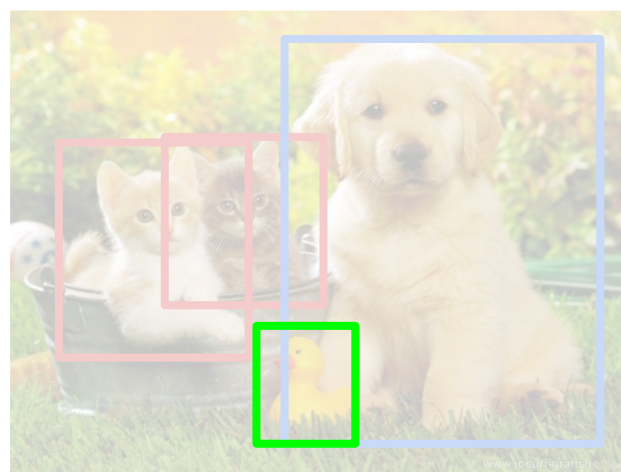
Classification



Classification  
+ Localization



Object  
Detection



Semantic  
Segmentation



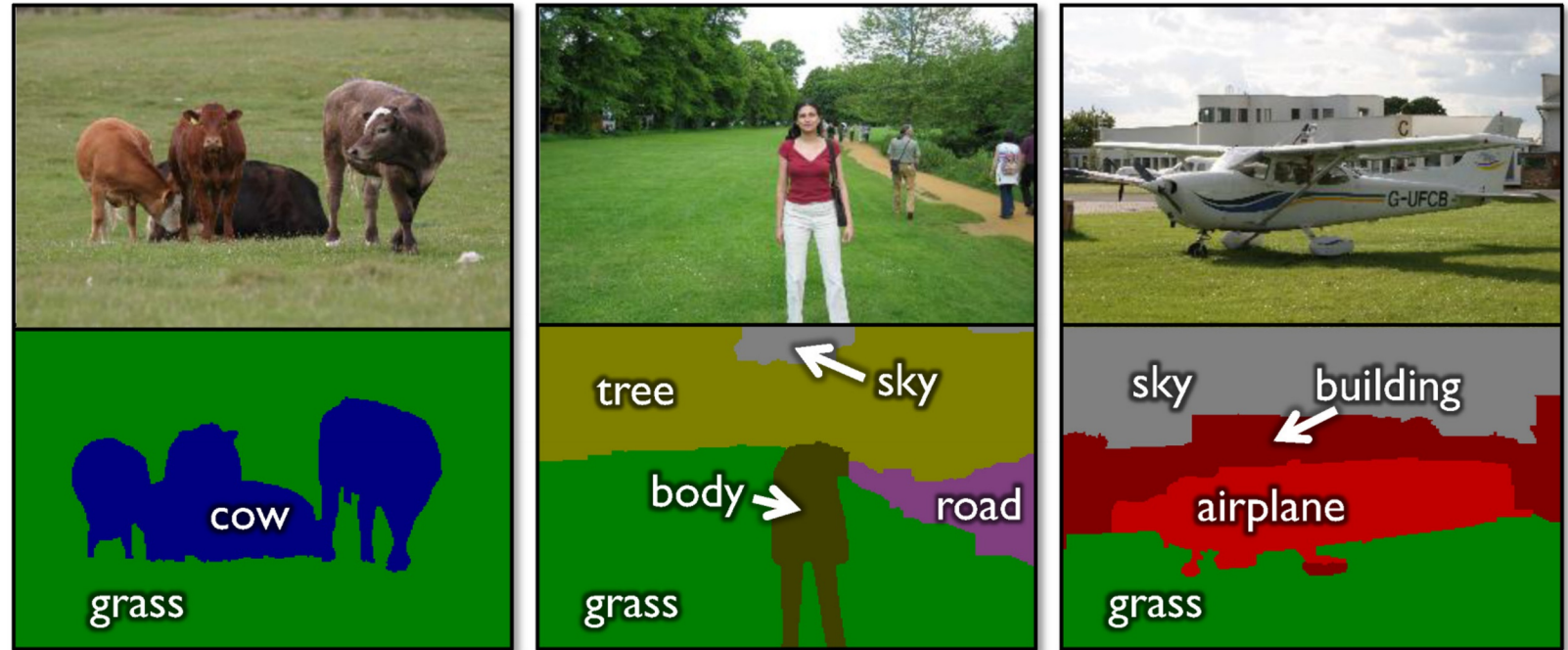
Today

# Semantic Segmentation

Label every pixel in the image with a category label

Don't differentiate instances (cows)

Classic computer vision problem



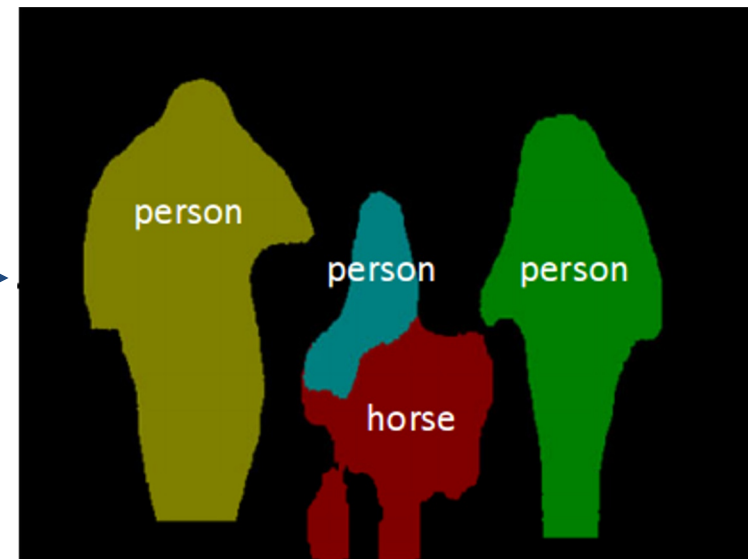
|                |          |       |      |      |       |      |          |       |      |      |
|----------------|----------|-------|------|------|-------|------|----------|-------|------|------|
| object classes | building | grass | tree | cow  | sheep | sky  | airplane | water | face | car  |
| bicycle        | flower   | sign  | bird | book | chair | road | cat      | dog   | body | boat |

# Instance Segmentation

Detect instances,  
give category, label  
pixels

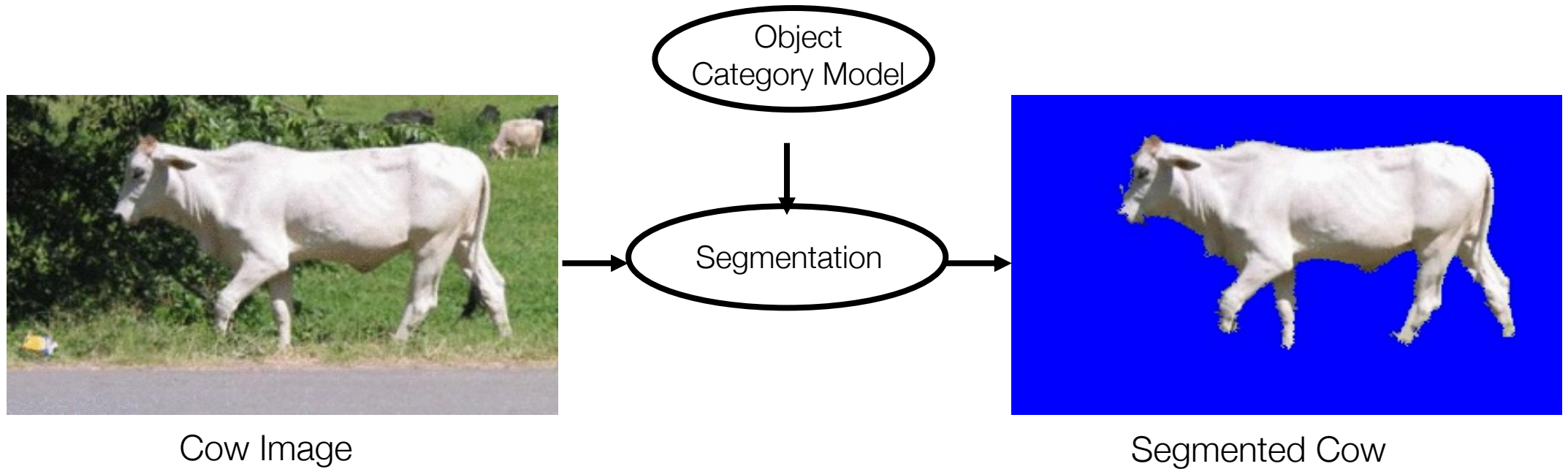
“simultaneous  
detection and  
segmentation” (SDS)

Lots of recent work  
(MS-COCO)



# Early Studies of Semantic Segmentation

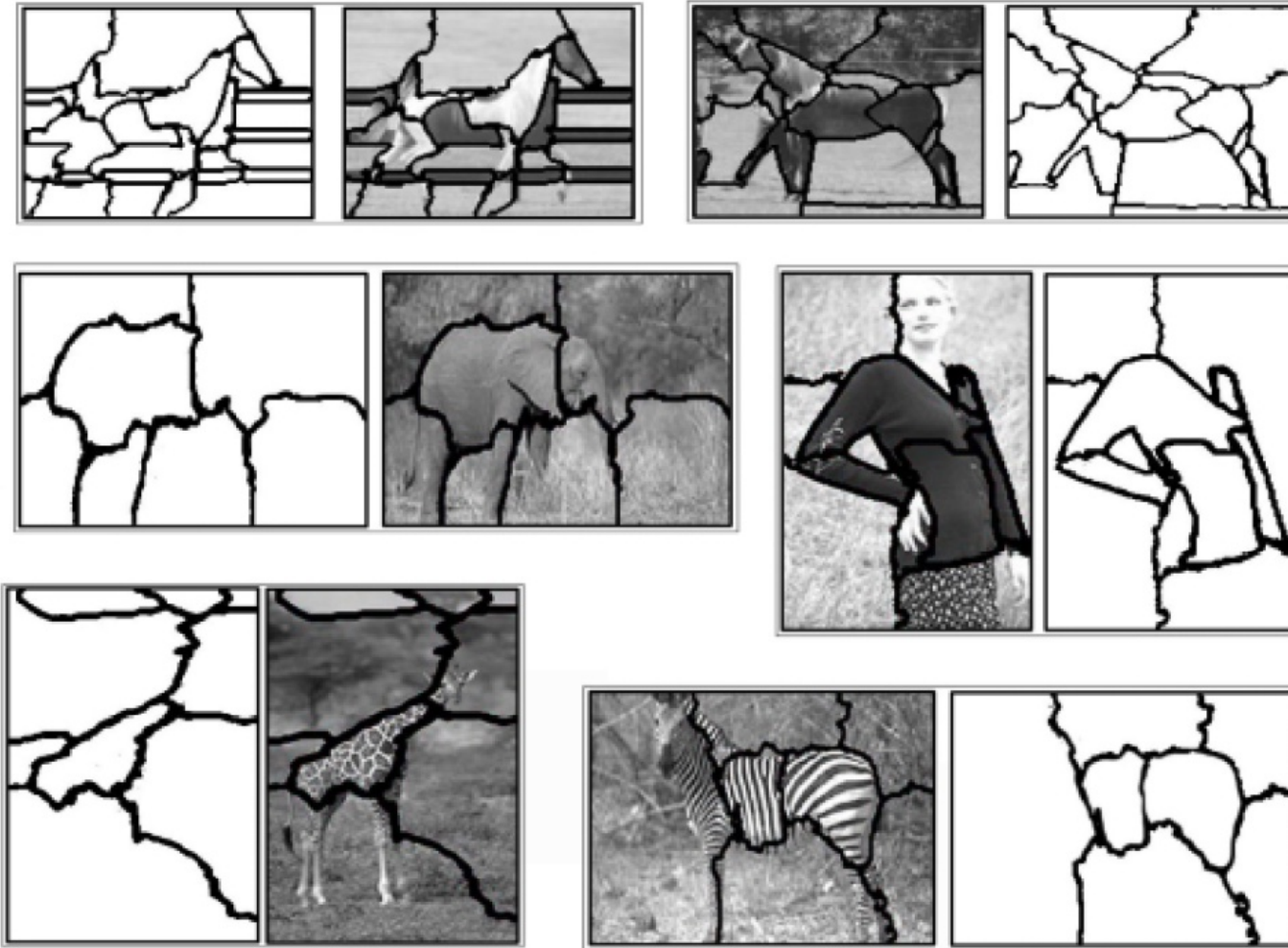
- Given an image and object category, to segment the object



- Segmentation should (ideally) be
  - shaped like the object e.g. cow-like
  - obtained efficiently in an unsupervised manner
  - able to handle self-occlusion

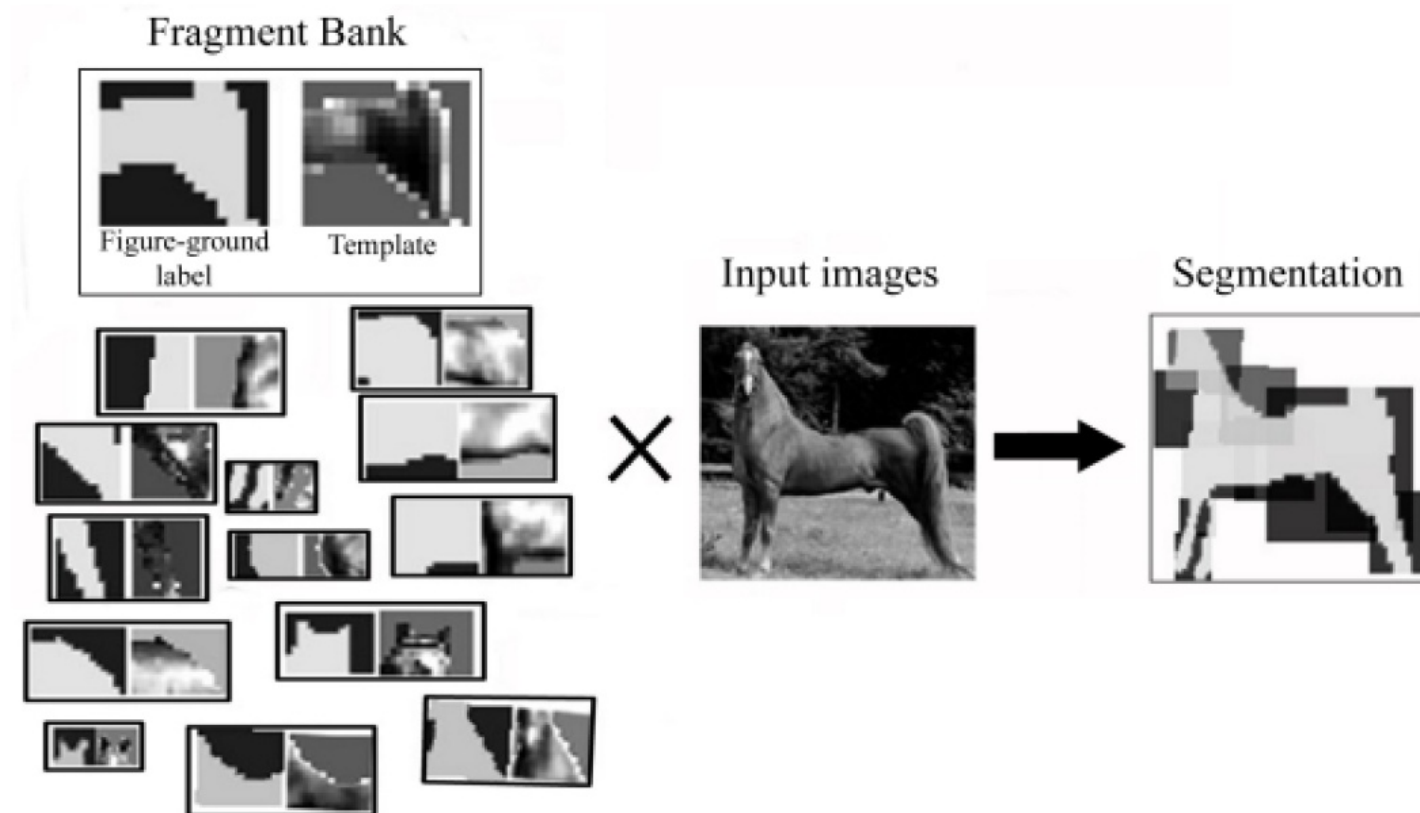
# Examples of Bottom-up Segmentation

Using Normalized Cuts, Shi & Malik, 1997

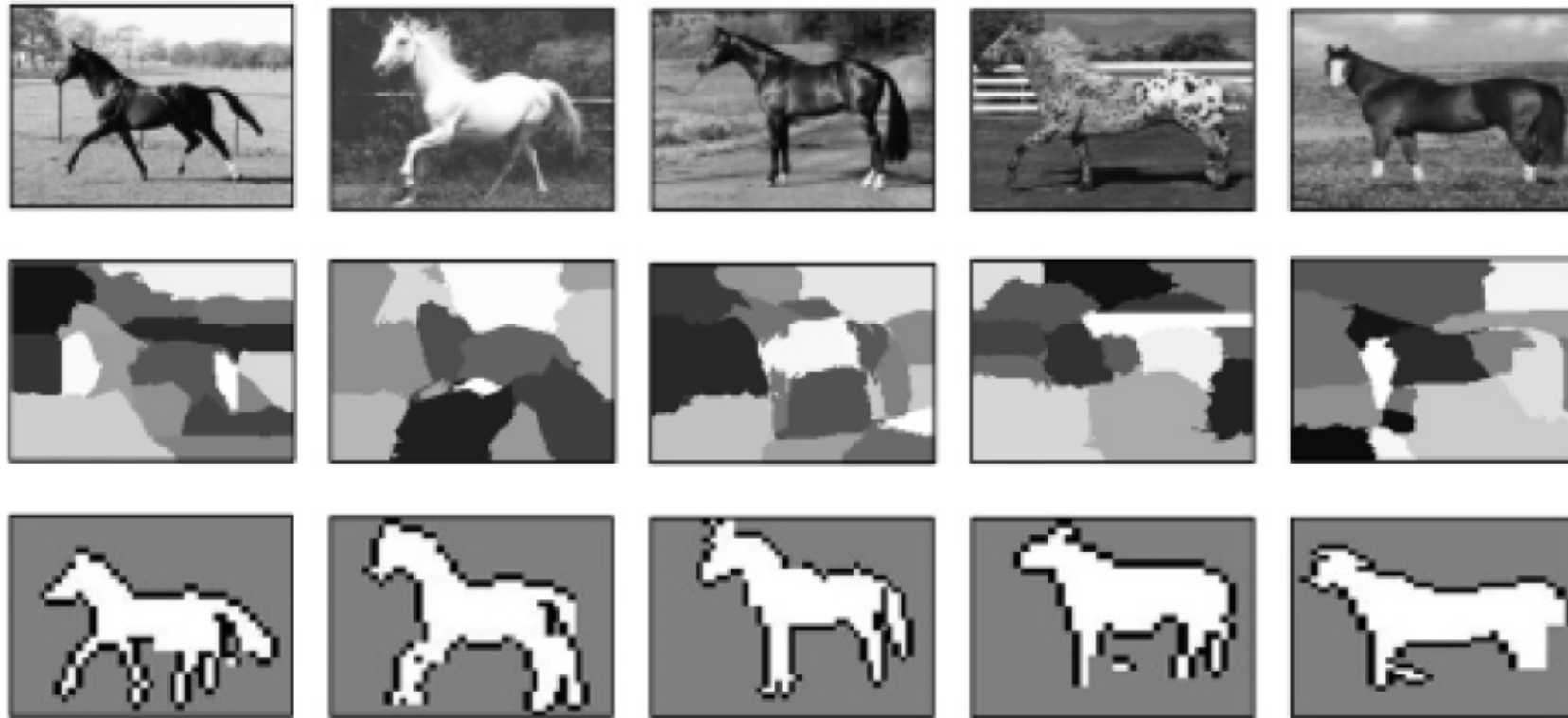




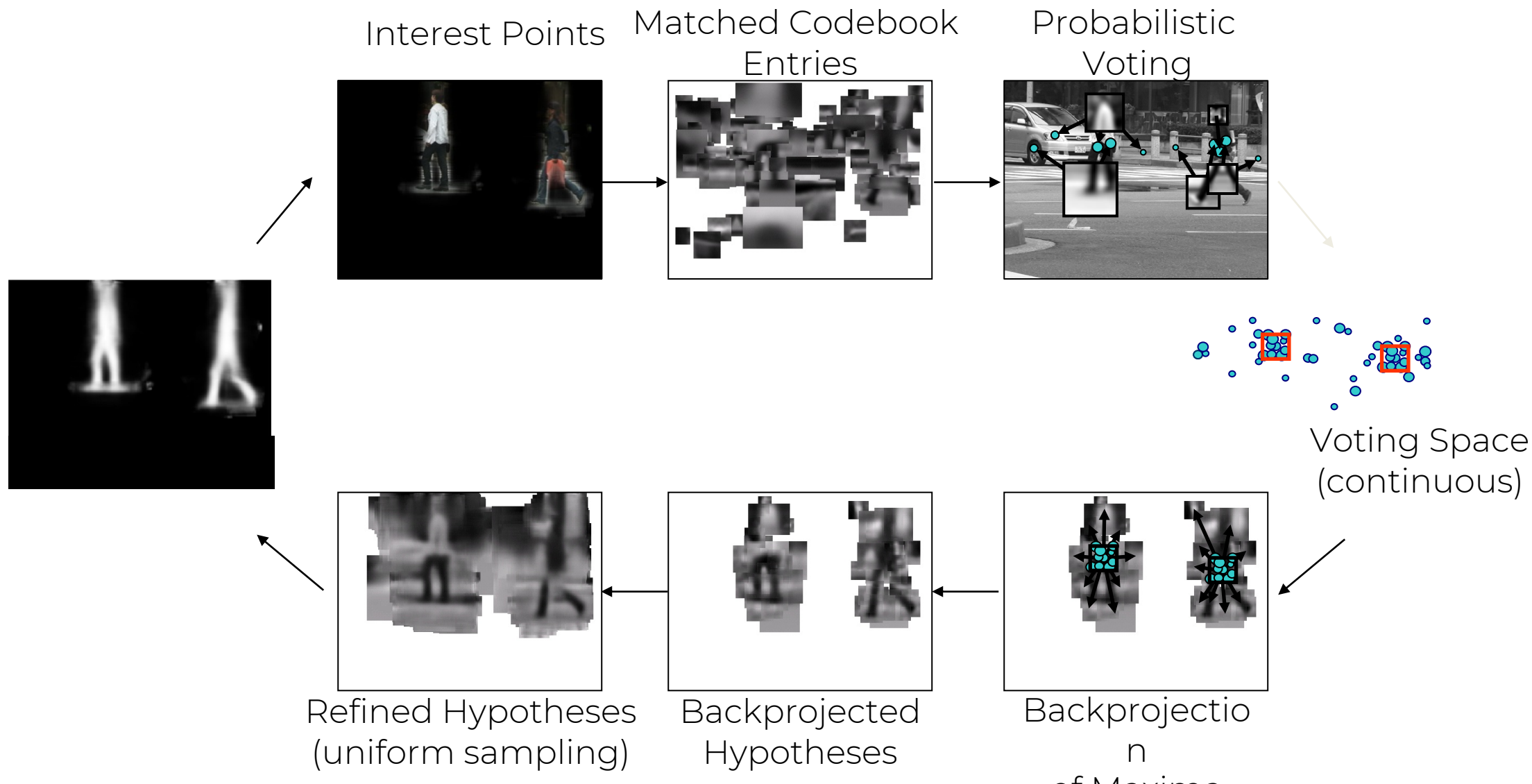
# Jigsaw approach: Borenstein and Ullman, 2002



# Jigsaw approach: Borenstein and Ullman, 2002



# Implicit Shape Model - Liebe and Schiele, 2003



# Random Fields for segmentation

$I$  = Image pixels (observed)

$h$  = foreground/background labels (hidden) – one label per pixel

$\theta$  = Parameters

$$\underbrace{p(h | I, \theta)}$$

Posterior

# Random Fields for segmentation

$I$  = Image pixels (observed)

$h$  = foreground/background labels (hidden) – one label per pixel

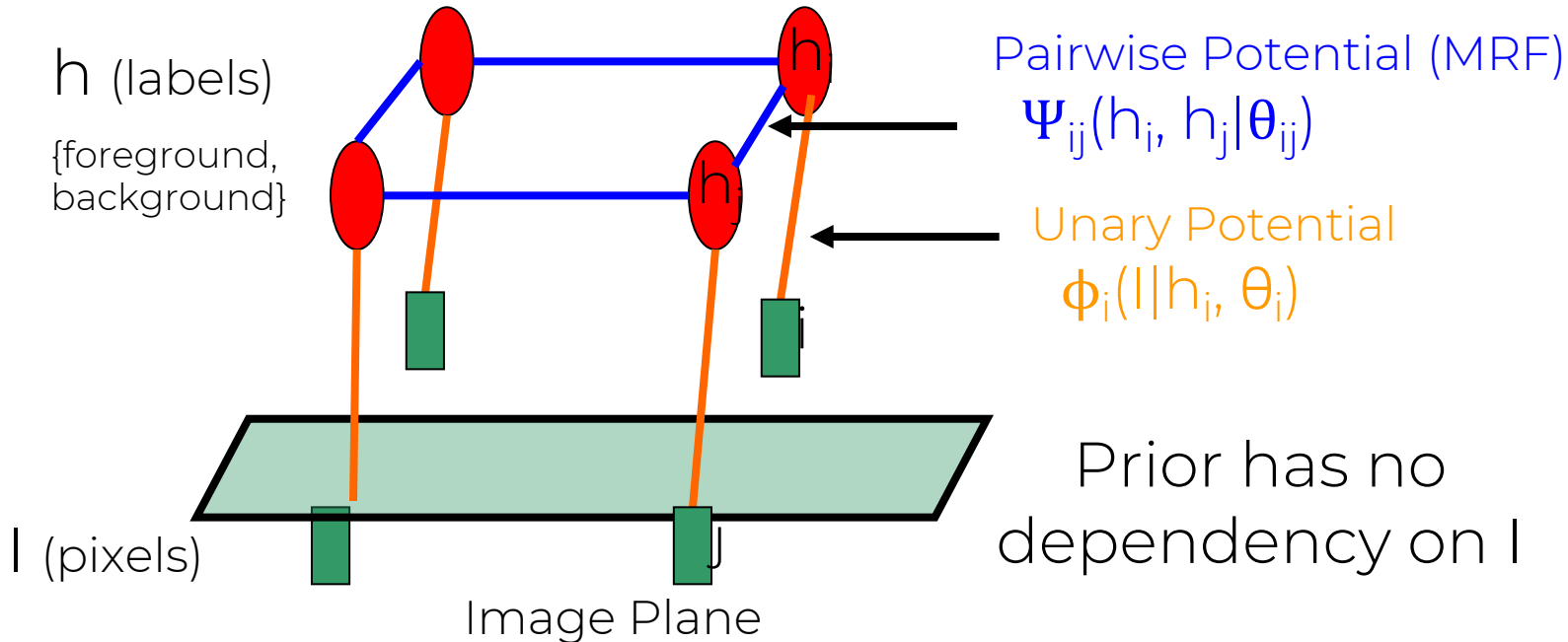
$\theta$  = Parameters

$$\underbrace{p(h | I, \theta)}_{\text{Posterior}} \propto \underbrace{p(I, h | \theta)}_{\text{Joint}} = \underbrace{p(I | h, \theta)}_{\text{Likelihood}} \underbrace{p(h | \theta)}_{\text{Prior}}$$

1. Generative approach models joint  
→ Markov random field (MRF)
2. Discriminative approach models posterior directly  
→ Conditional random field (CRF)

# Generative Markov Random Field

$$\begin{aligned}
 p(h, I | \theta) &= p(I | h, \theta) p(h | \theta) \\
 &= \frac{1}{Z(\theta)} \left[ \underbrace{\prod_i \phi_i(I | h_i, \theta_i)}_{\text{Likelihood}} \underbrace{\prod_{ij} \psi_{ij}(h_i, h_j | \theta_{ij})}_{\text{MRF Prior}} \right]
 \end{aligned}$$



# Conditional Random Field

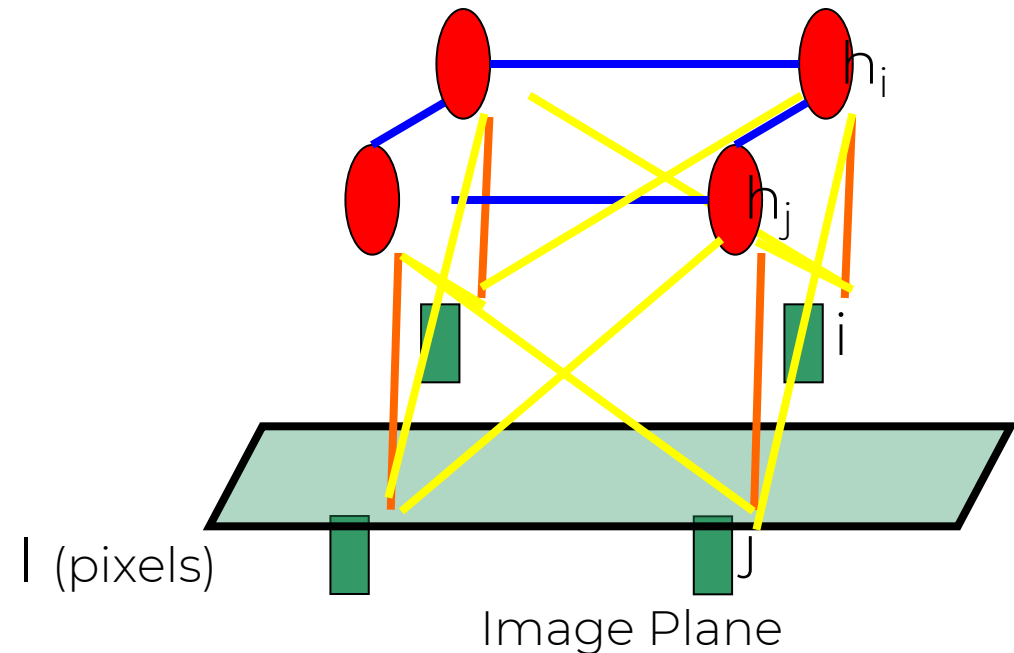
Lafferty, McCallum and Pereira 2001

Discriminative approach

$$p(h | I, \theta) = \frac{1}{Z(I, \theta)} \left[ \underbrace{\prod_i \phi_i(h_i, I | \theta_i)}_{\text{Unary}} \underbrace{\prod_{ij} \psi_{ij}(h_i, h_j, I | \theta_{ij})}_{\text{Pairwise}} \right]$$

- Dependency on  $I$  allows introduction of pairwise terms that make use of image.
- For example, neighboring labels should be similar only if pixel colors are similar  $\rightarrow$  Contrast term

e.g Kumar and Hebert 2003



# Conditional Random Fields for Segmentation

- Segmentation map  $x$
- Image  $I$

$$E(x; I) = \nu \underbrace{\sum_{i,j} w_{ij} |x(i) - x(j)|}_{\text{Low-level pairwise term}} + \underbrace{\sum_k \lambda_k |x - x_{F_k, I}|}_{\text{High-level local term}}$$

Pixel-wise similarity

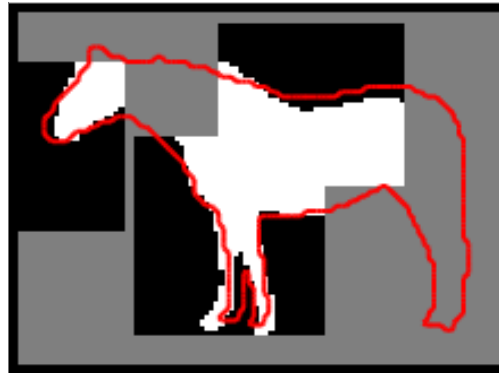
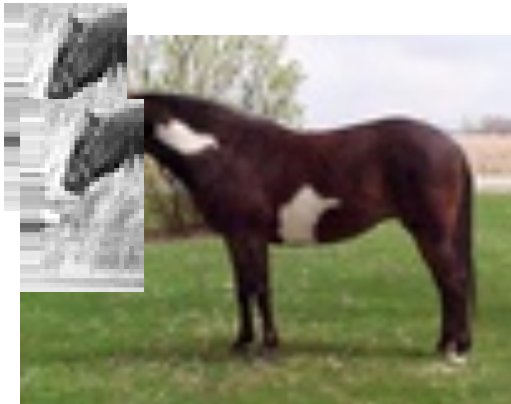


# Levin & Weiss [ECCV 2006]

$$E(h; I) = \sum_i \lambda_i |h - h_{F_i, I}| + \sum_{ij} w(i, j) |h_i - h_j|$$

Consistency  
with fragments  
segmentation

Segmentation  
alignment  
with image  
edges

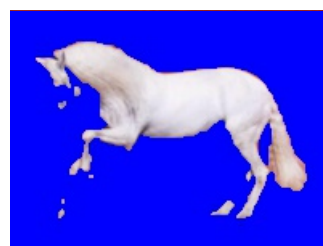
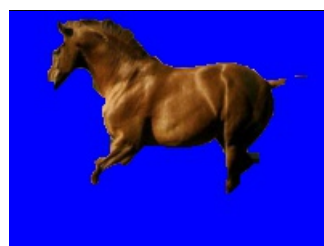
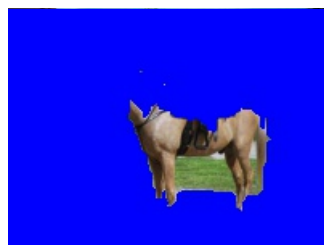
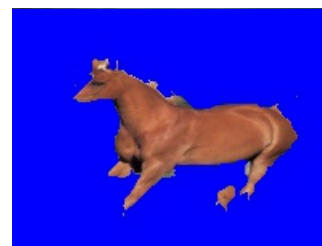
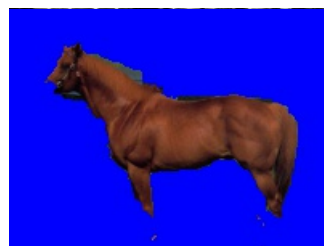
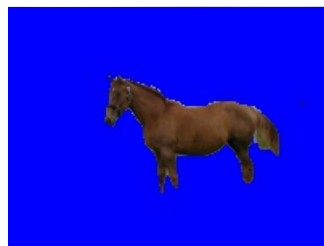


Resulting min-cut  
segmentation

# Levin & Weiss [ECCV 2006]

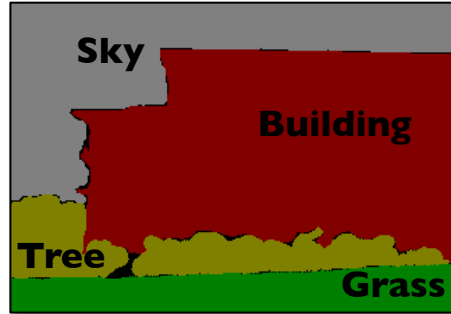


# Levin & Weiss [ECCV 2006]



# Semantic Segmentation

## Joint Object recognition & segmentation

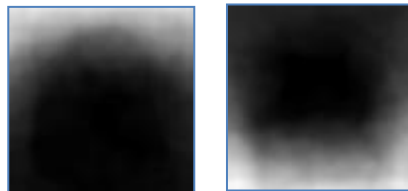


$$E(x, \omega) = \sum_i \theta_i(\omega, x_i) + \sum_i \theta_i(x_i) + \sum_i \theta_i(x_i) + \sum_{i,j} \theta_{ij}(x_i, x_j)$$

(color)
(location)
(class)
(edge aware Ising prior)

$x_i \in \{1, \dots, K\}$  for  $K$  object classes

Location



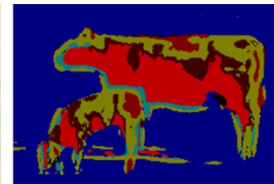
sky

grass

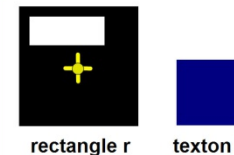
Class (boosted textons)



(a) Input image



(b) Texton map



(c) Feature pair = (r,t)



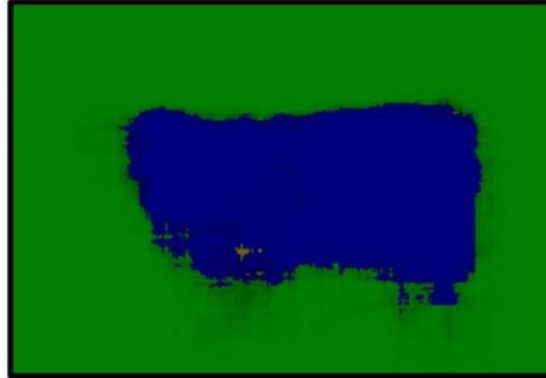
(d) Superimposed rectangles

# Semantic Segmentation

Joint Object recognition & segmentation

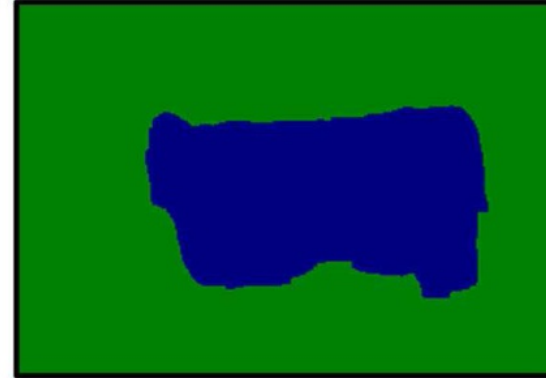


(a)



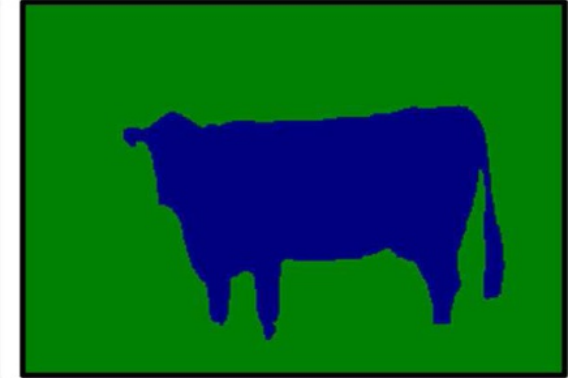
(b) 69.6%

Class +  
location



(c) 70.3%

+  
edges



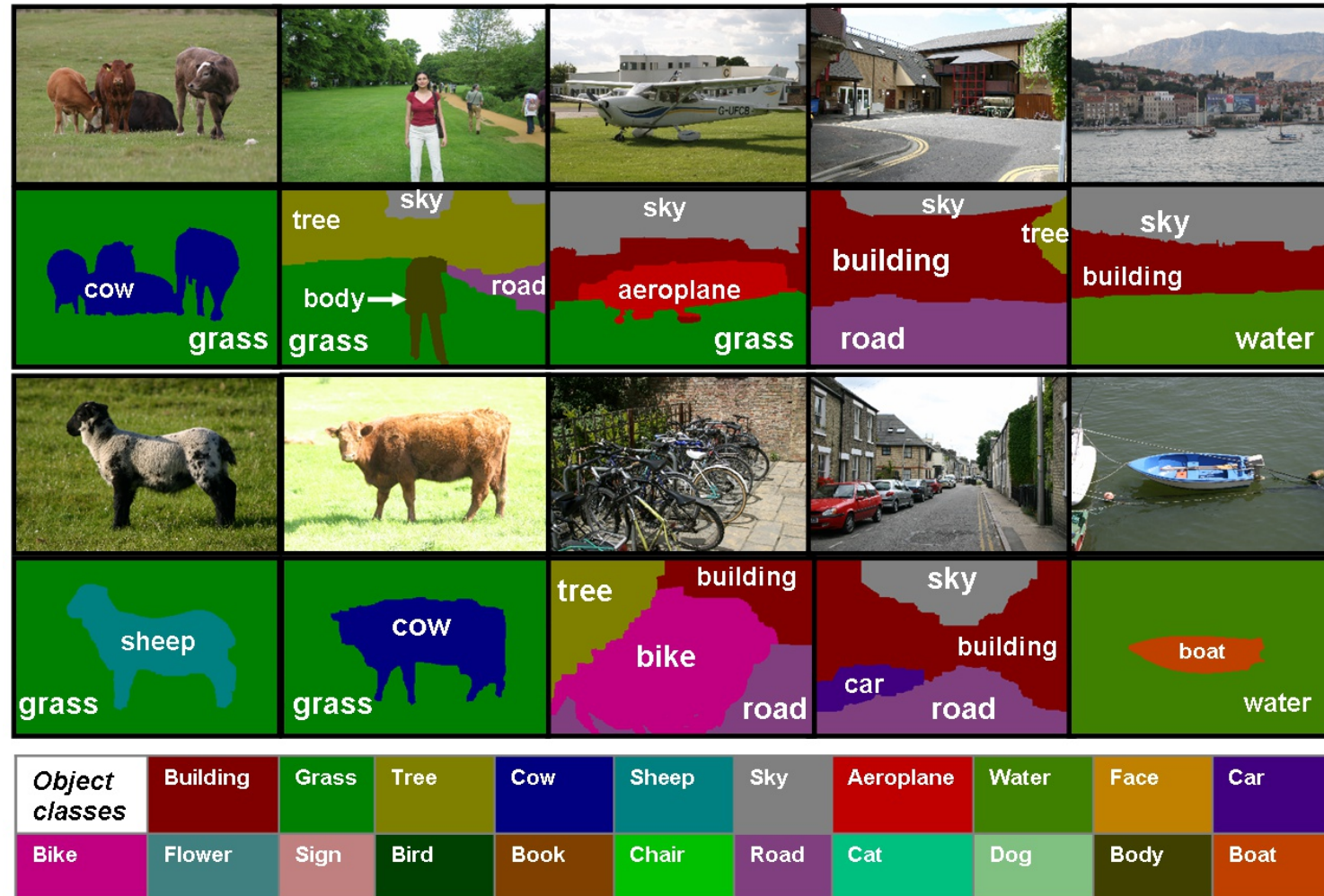
(d) 72.2%

+  
color

# Semantic Segmentation

## Joint Object recognition & segmentation

Good results ...



# Semantic Segmentation

Joint Object recognition & segmentation

Failure cases...



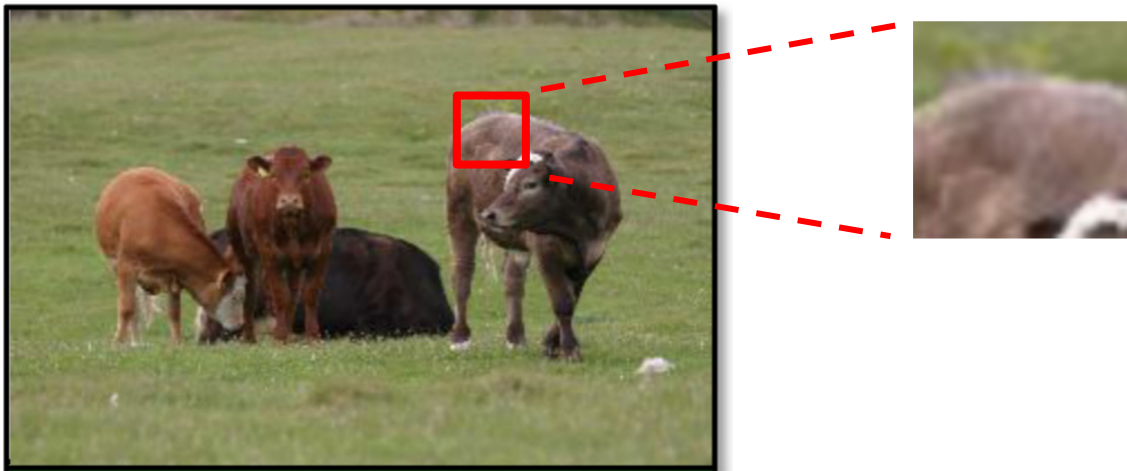
# Deep Semantic Segmentation



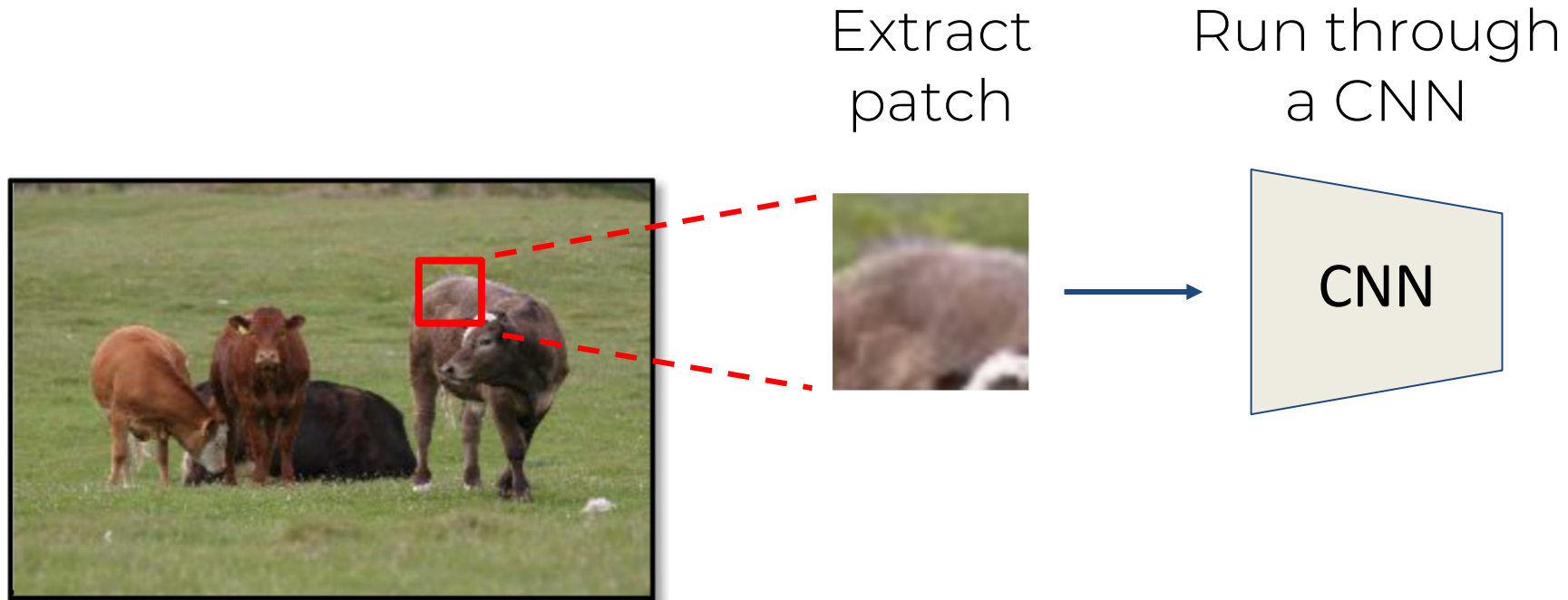


# Deep Semantic Segmentation

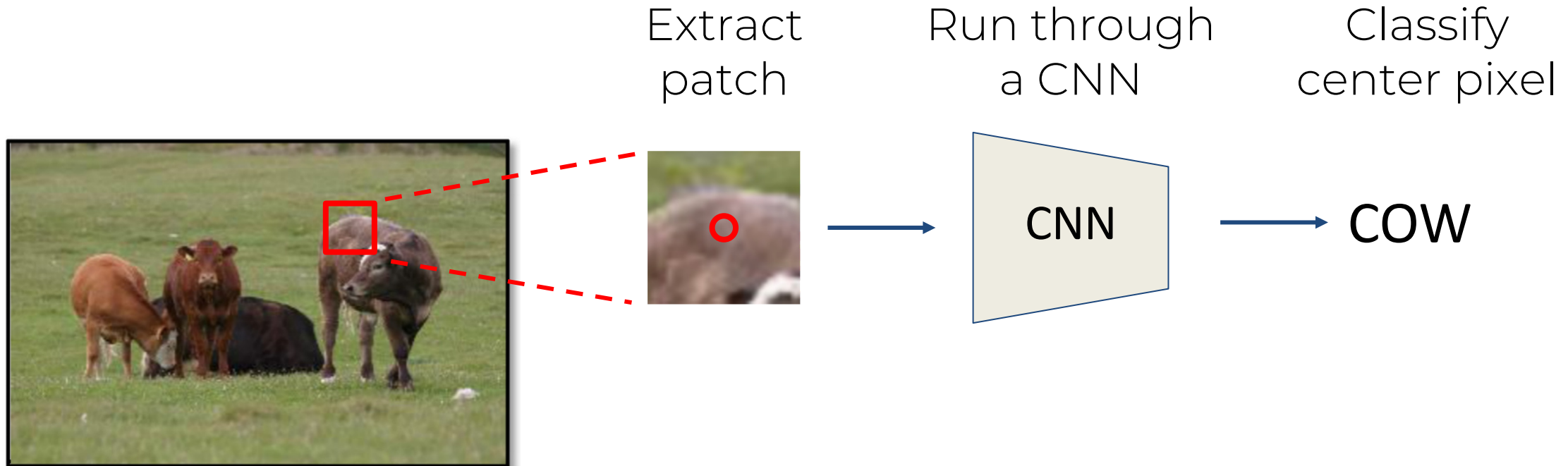
Extract  
patch



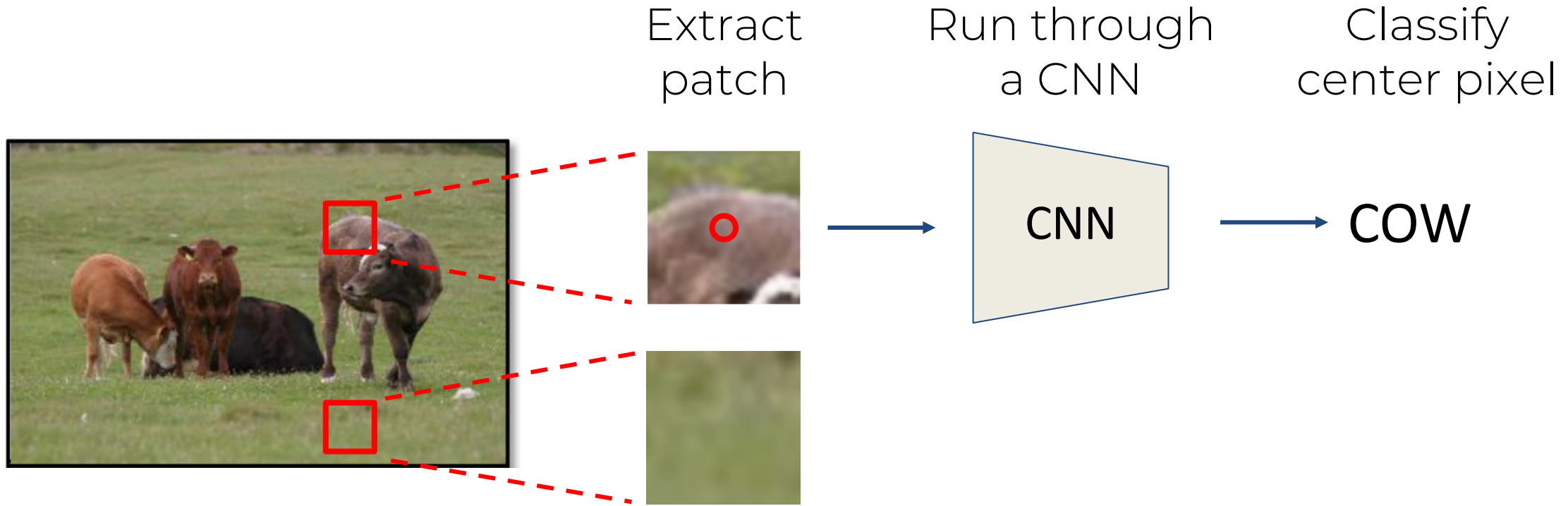
# Deep Semantic Segmentation



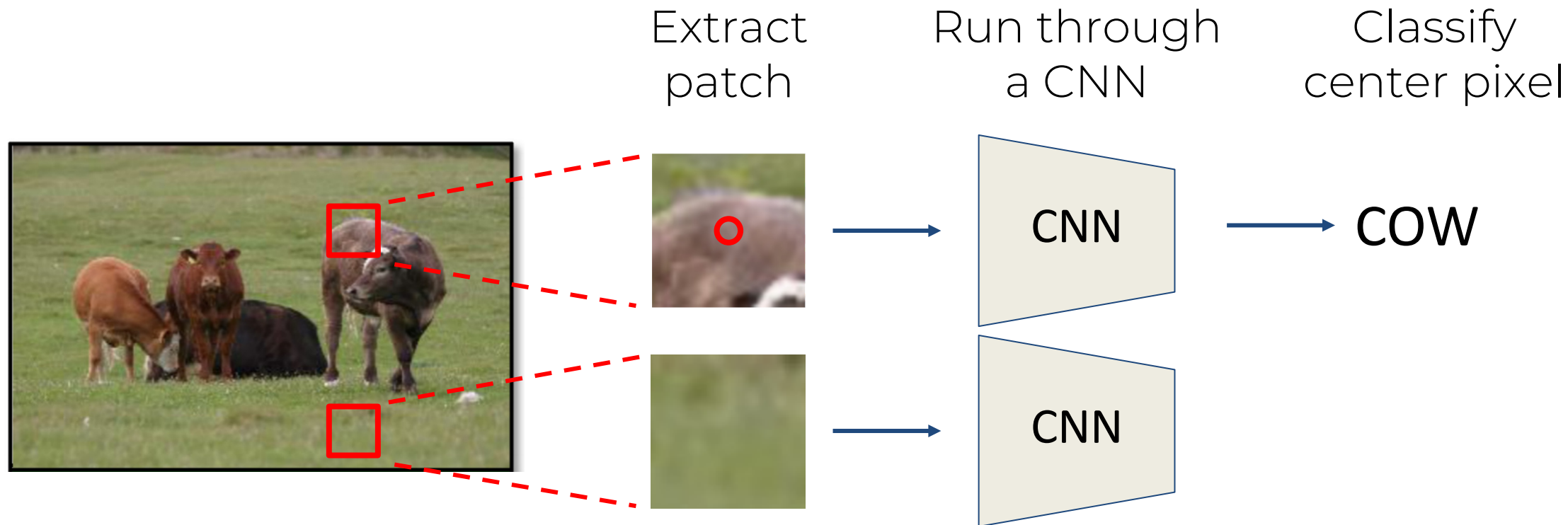
# Deep Semantic Segmentation



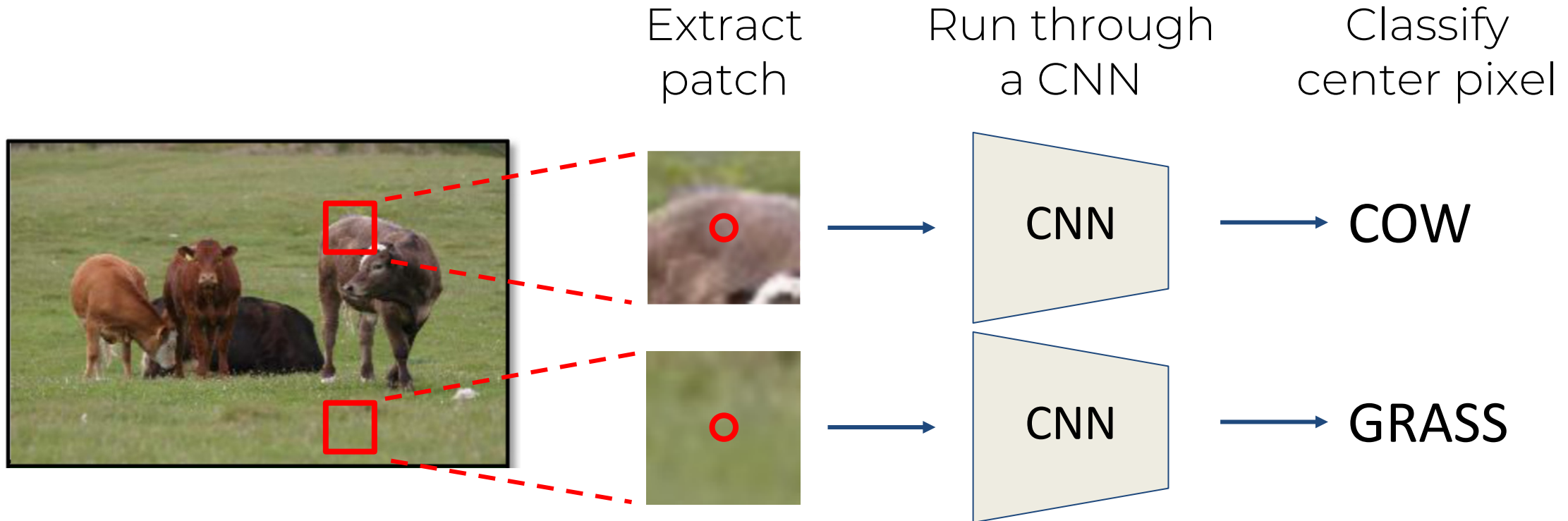
# Deep Semantic Segmentation



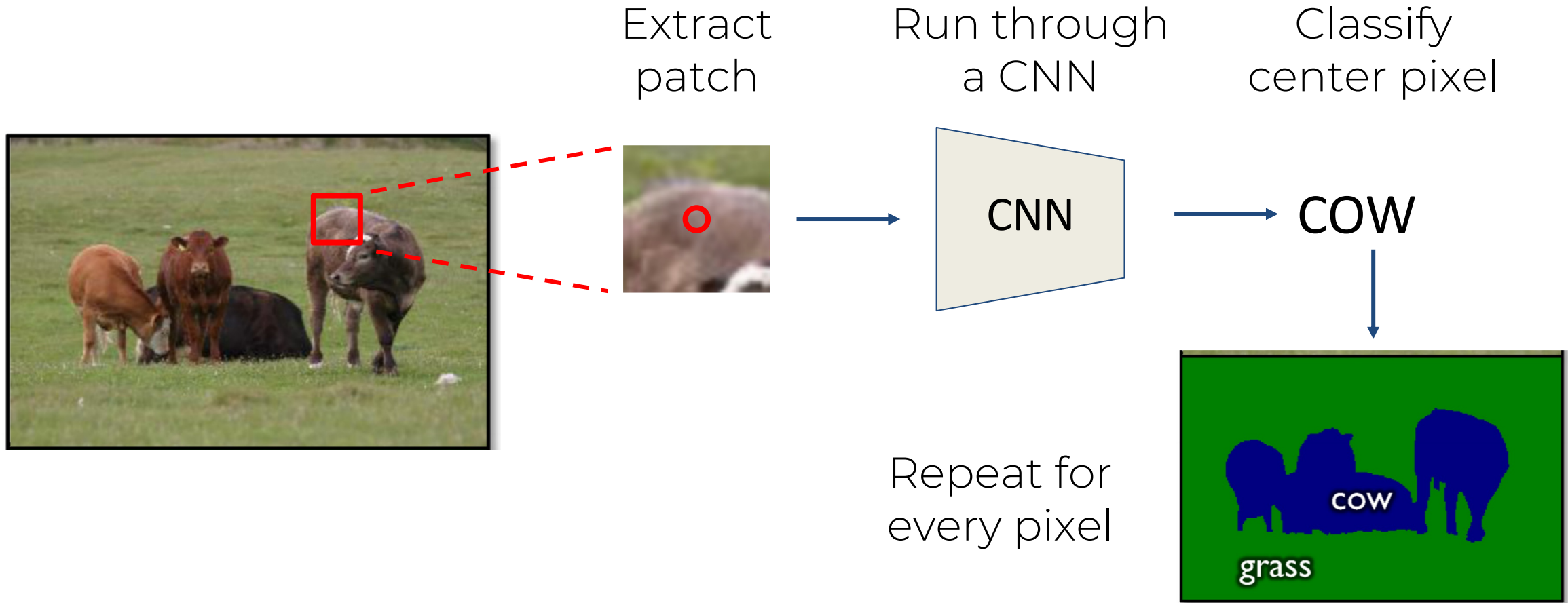
# Deep Semantic Segmentation



# Deep Semantic Segmentation



# Deep Semantic Segmentation

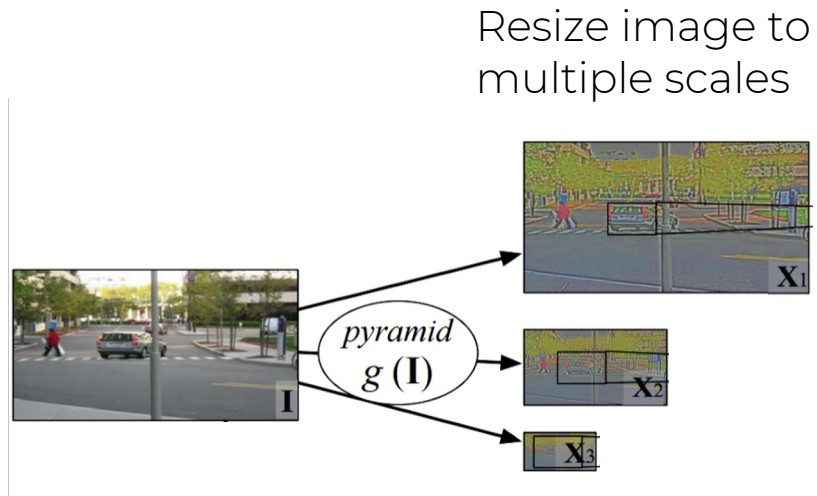


# Semantic Segmentation: Multi-Scale

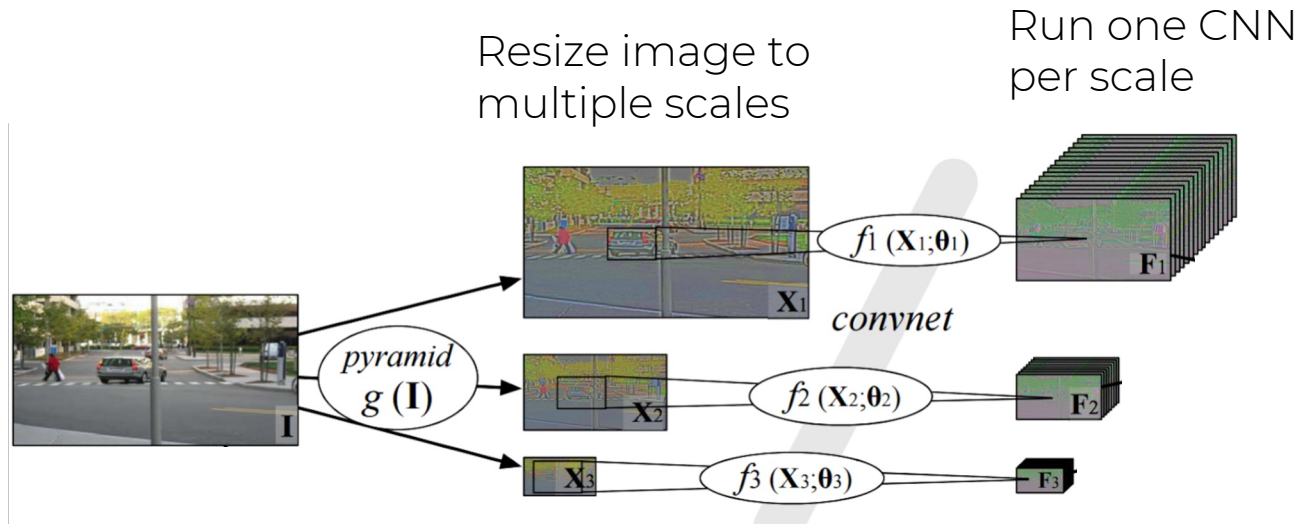




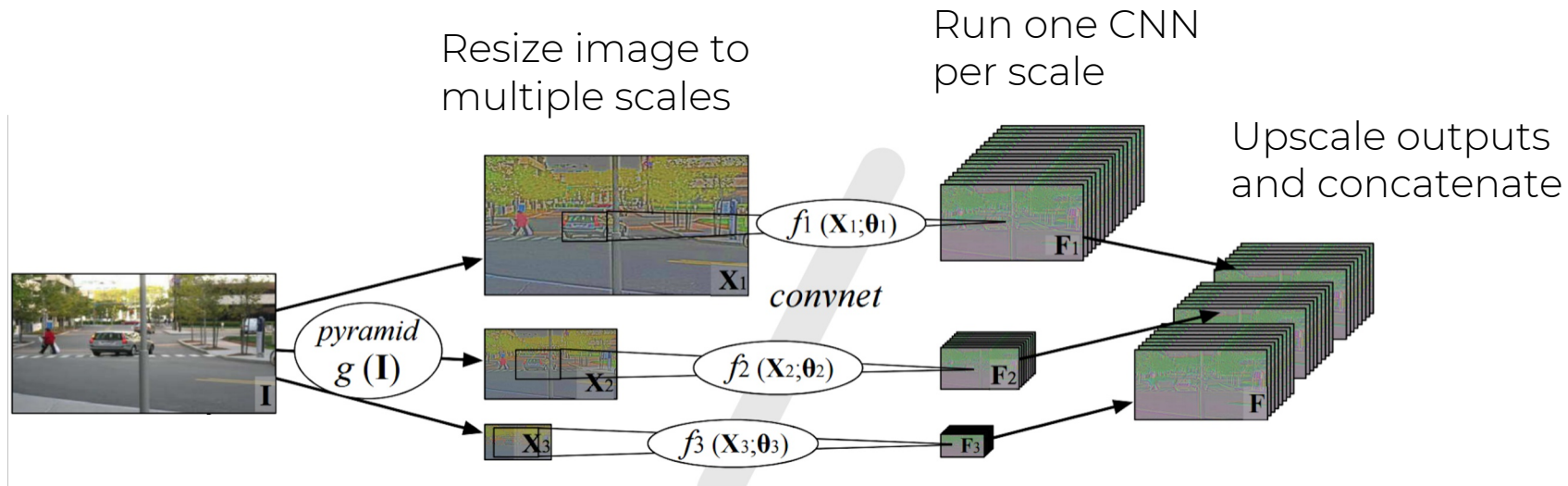
# Semantic Segmentation: Multi-Scale



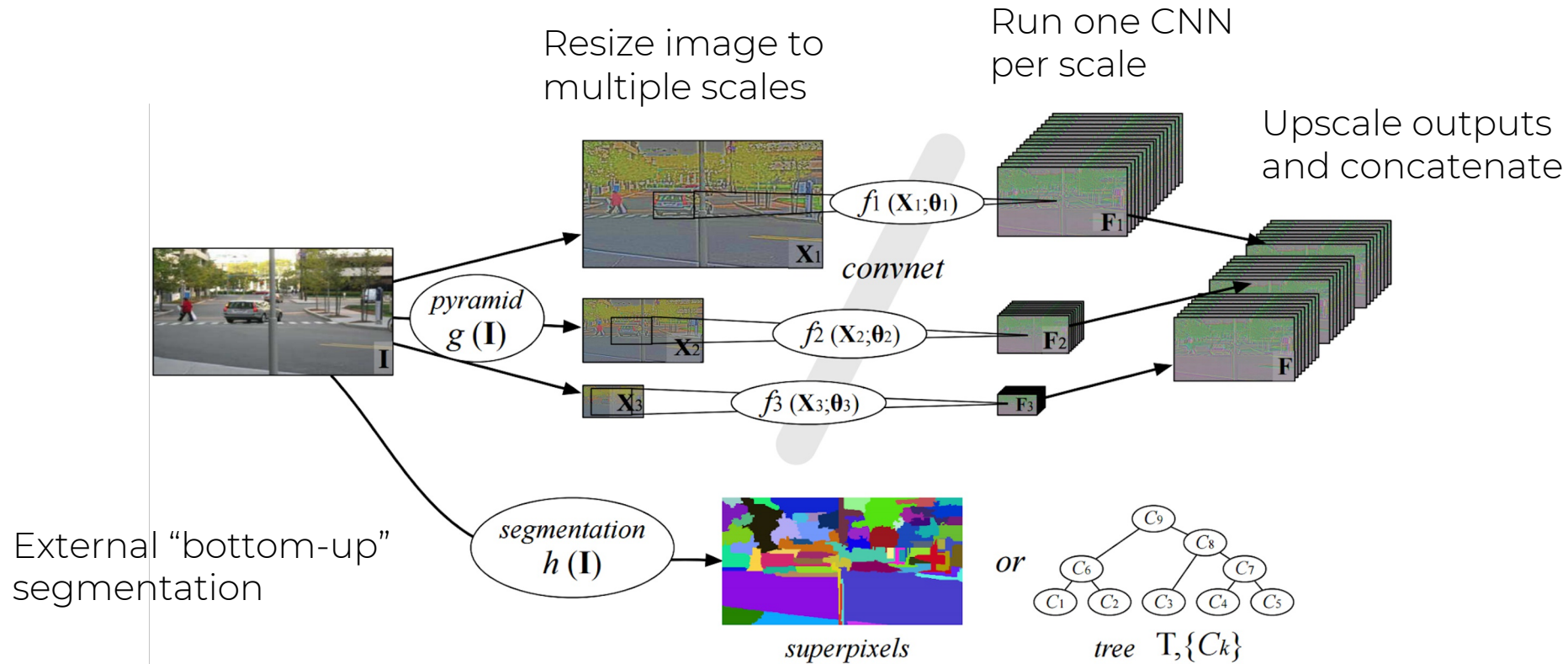
# Semantic Segmentation: Multi-Scale



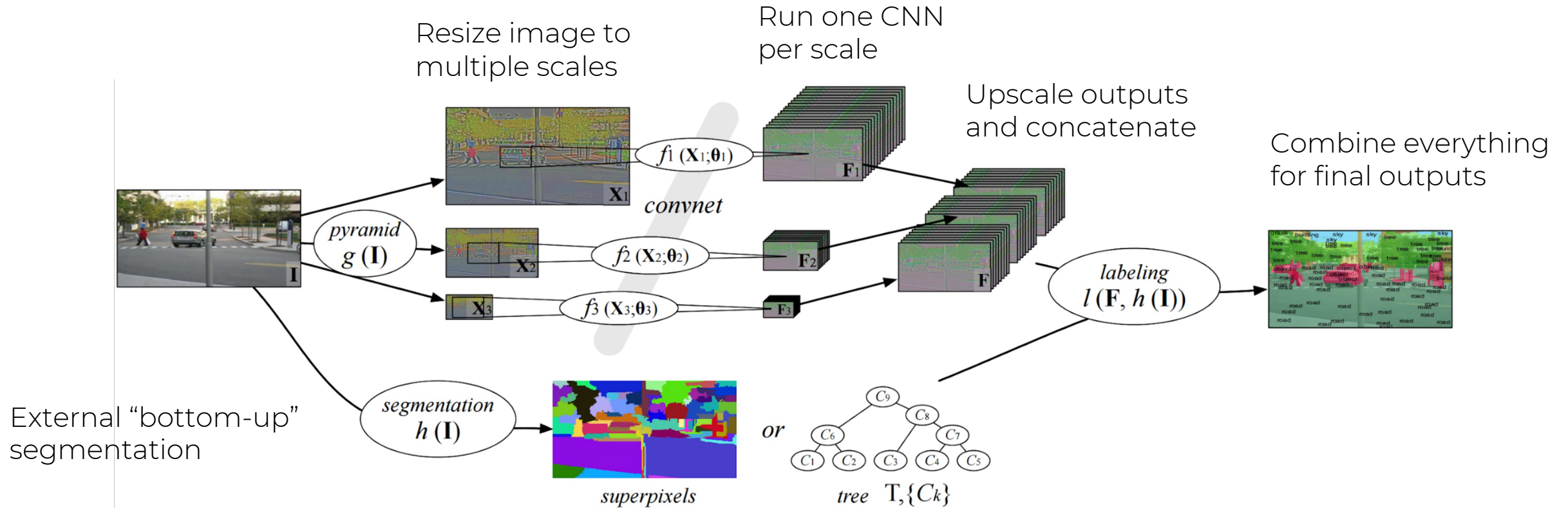
# Semantic Segmentation: Multi-Scale



# Semantic Segmentation: Multi-Scale

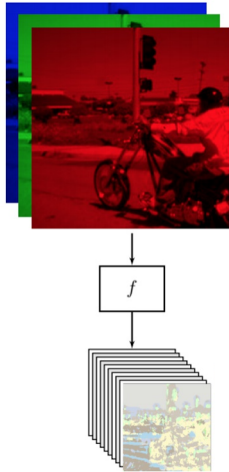


# Semantic Segmentation: Multi-Scale

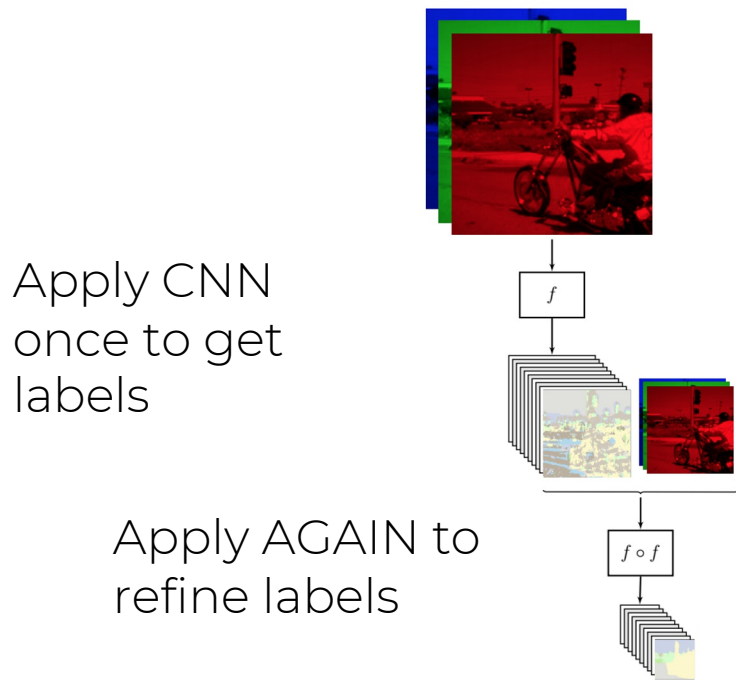


# Semantic Segmentation: Refinement

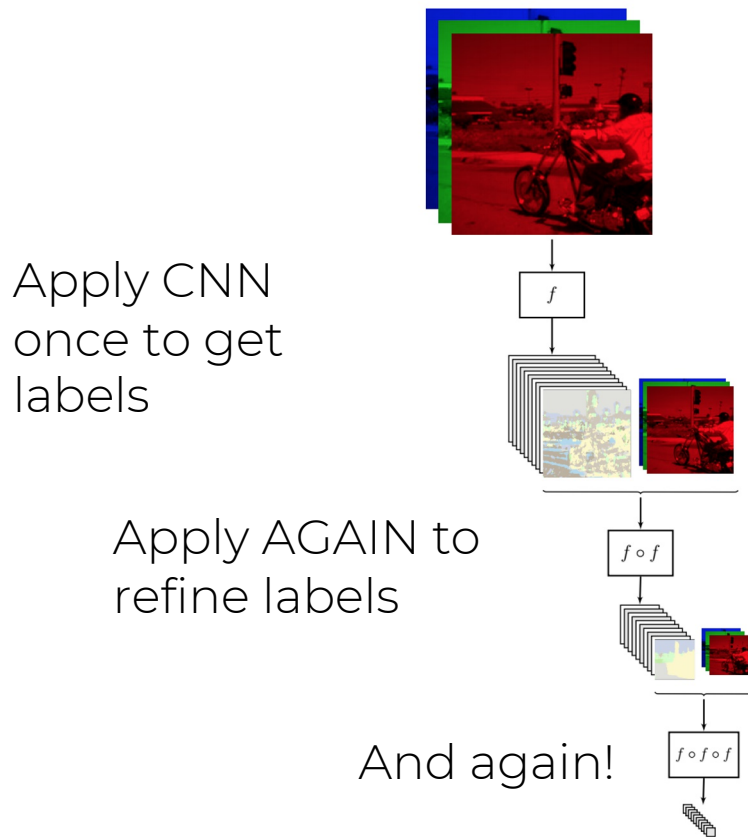
Apply CNN  
once to get  
labels



# Semantic Segmentation: Refinement

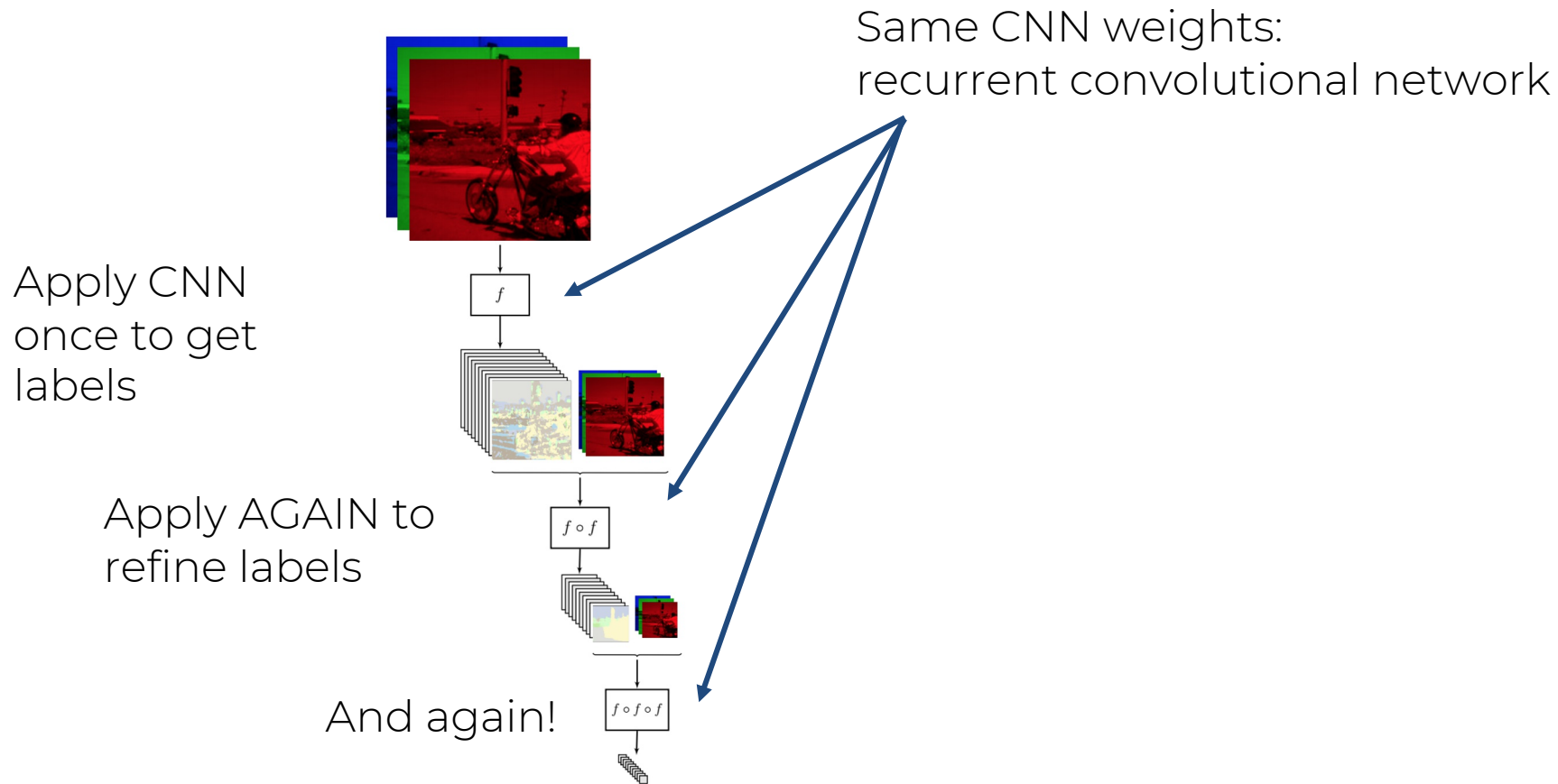


# Semantic Segmentation: Refinement

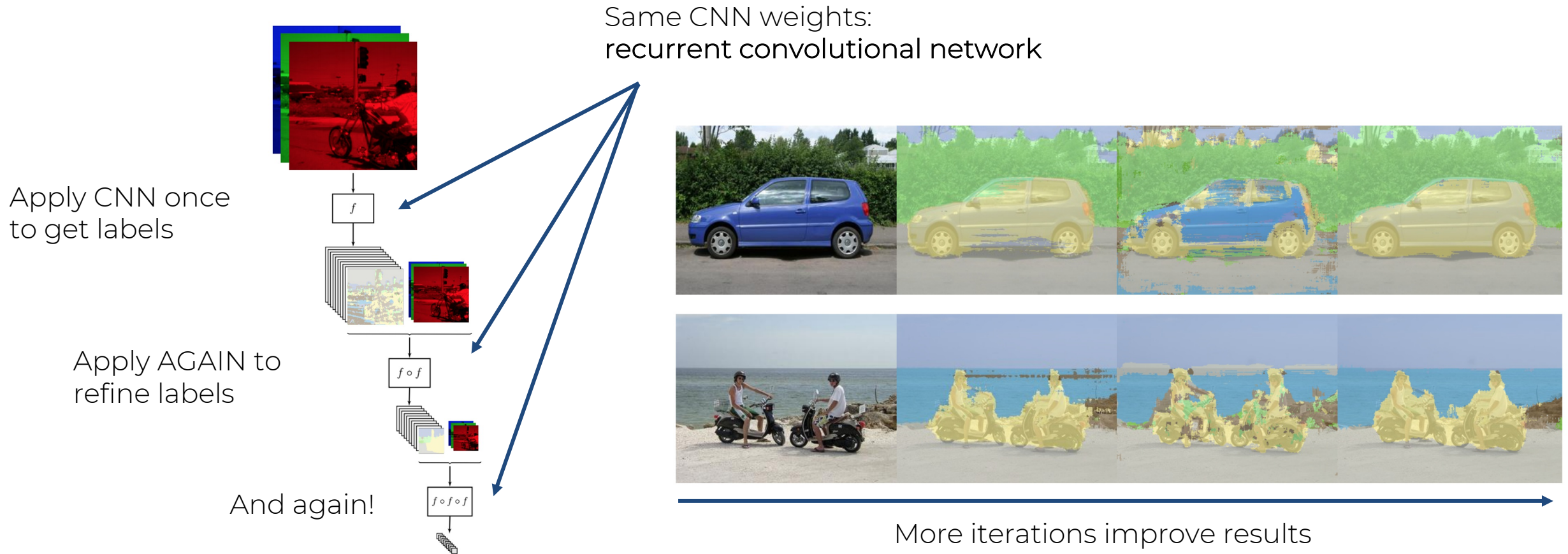




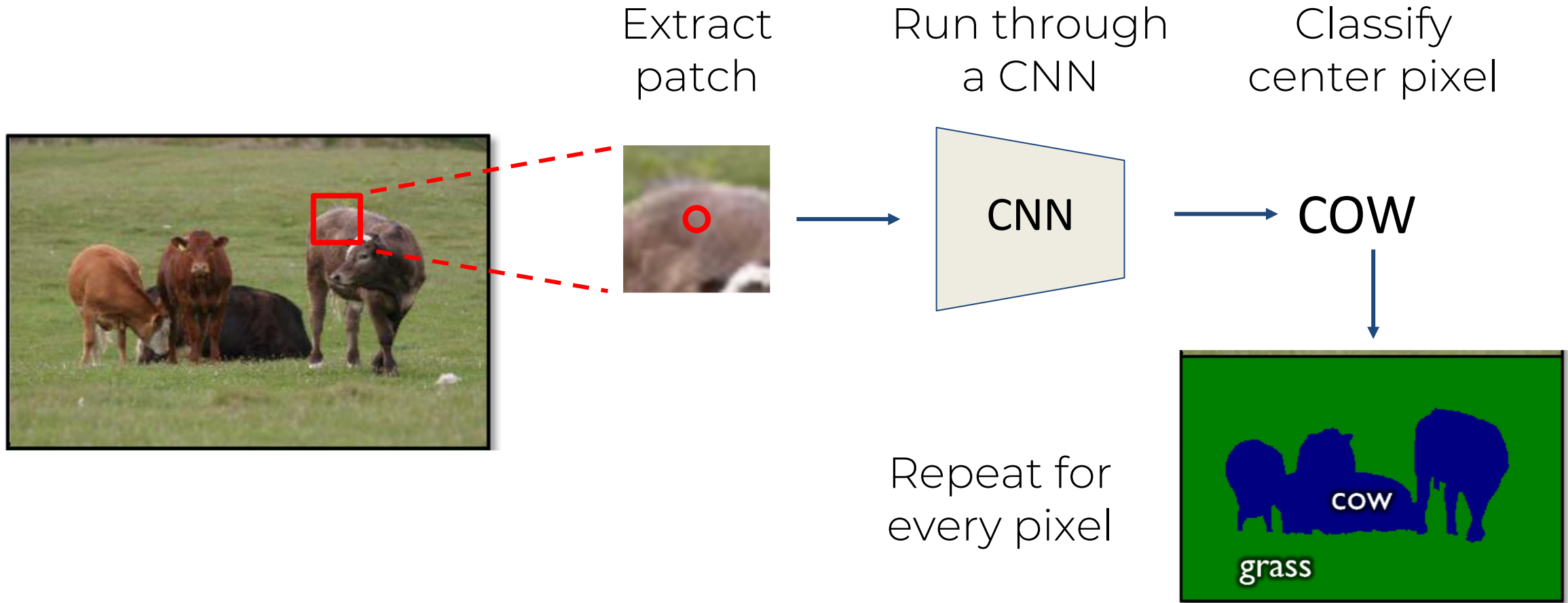
# Semantic Segmentation: Refinement



# Semantic Segmentation: Refinement

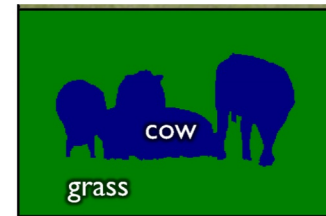
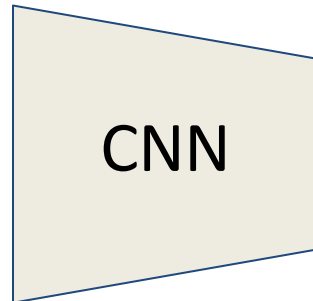


# Deep Semantic Segmentation



# Semantic Segmentation Idea: Fully Convolutional

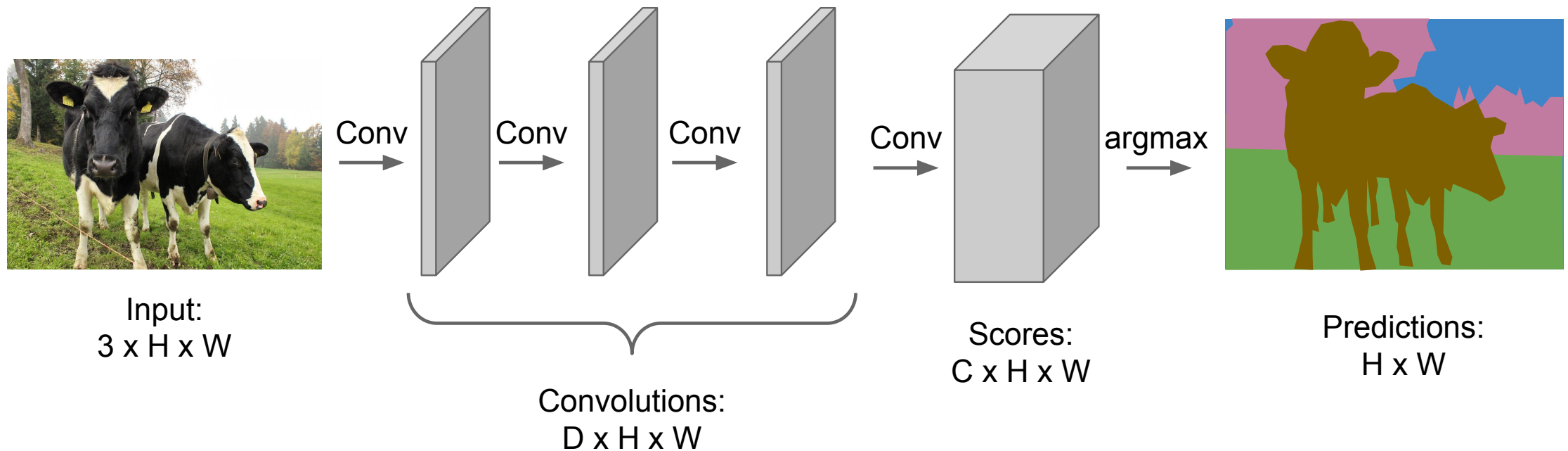
Run “fully convolutional”  
network to get all pixels at  
once



Smaller output  
due to pooling

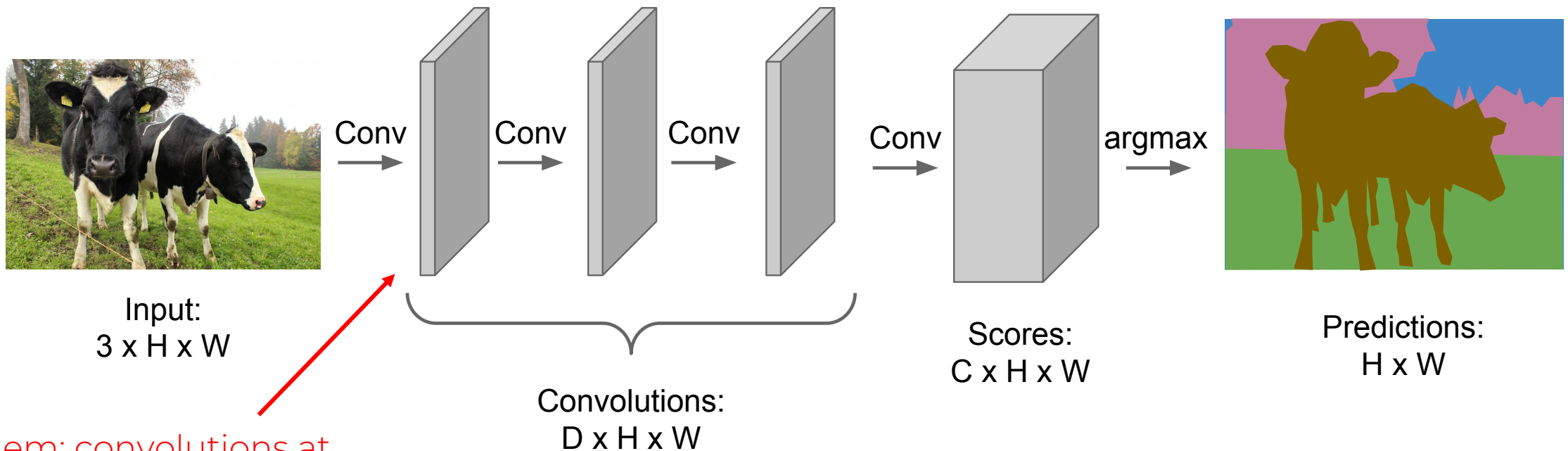
# Semantic Segmentation Idea: Fully Convolutional

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



# Semantic Segmentation Idea: Fully Convolutional

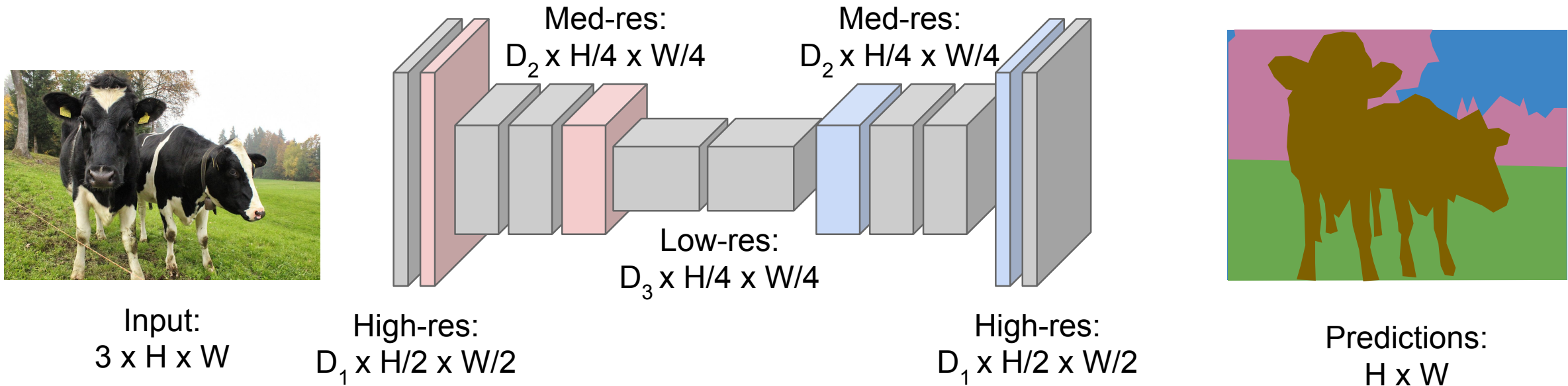
- Design a network as a bunch of convolutional layers to make predictions for pixels all at once!



Problem: convolutions at original image resolution will be very expensive ...

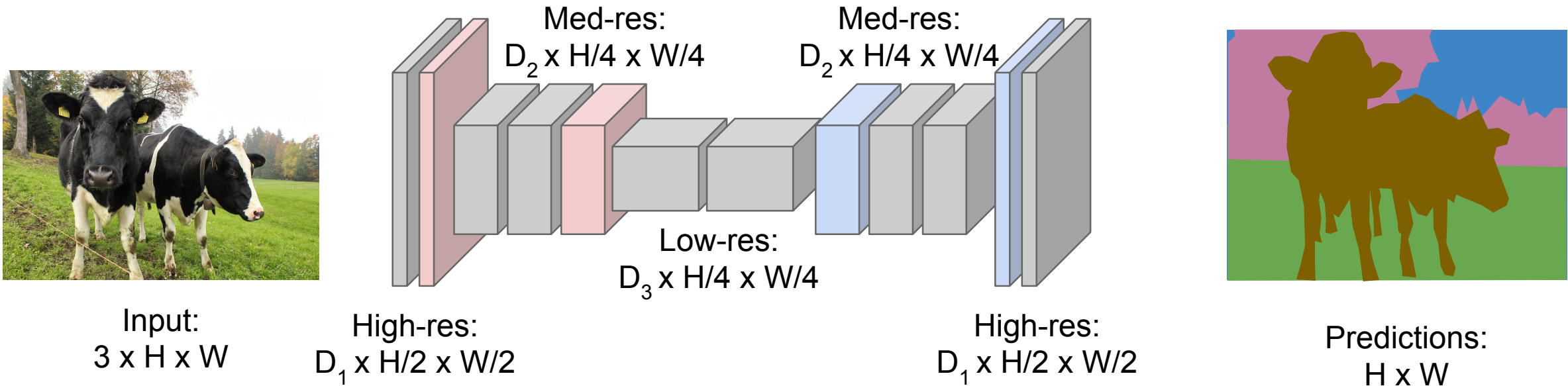
# Semantic Segmentation Idea: Fully Convolutional

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



# Semantic Segmentation Idea: Fully Convolutional

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

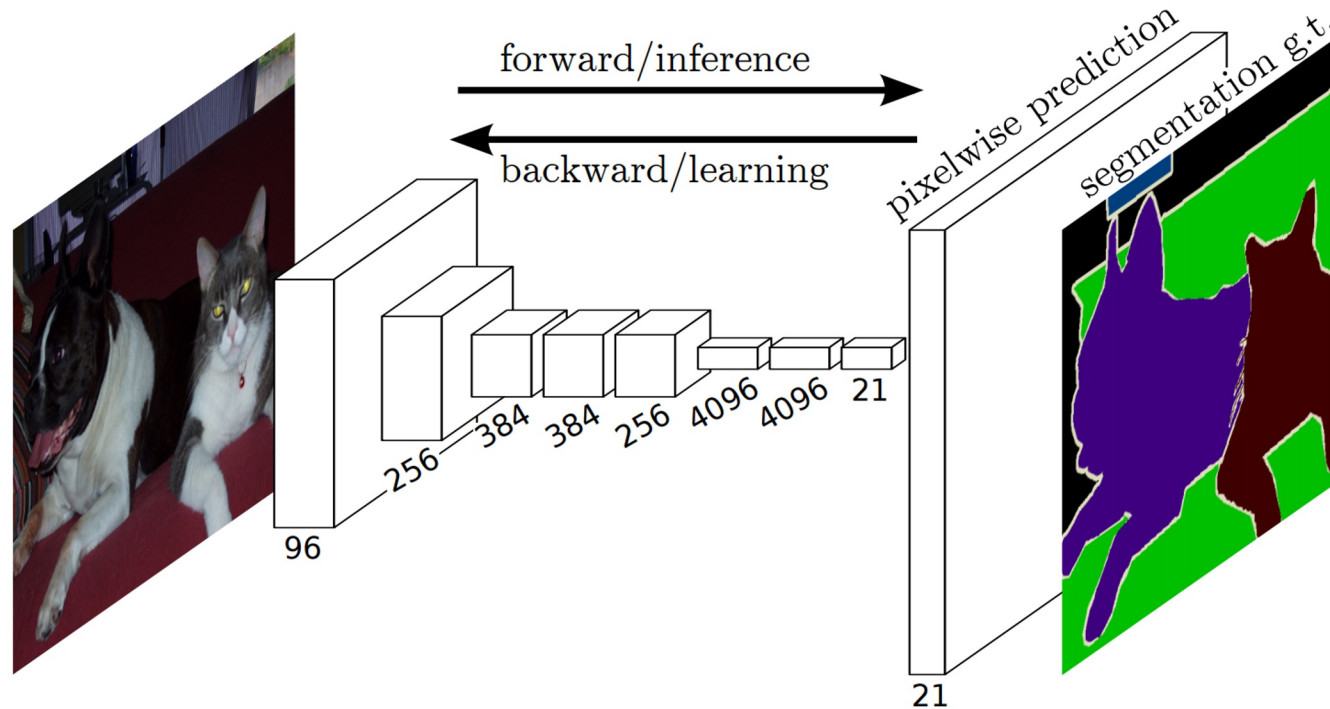


Downsampling:  
Pooling, strided  
convolution

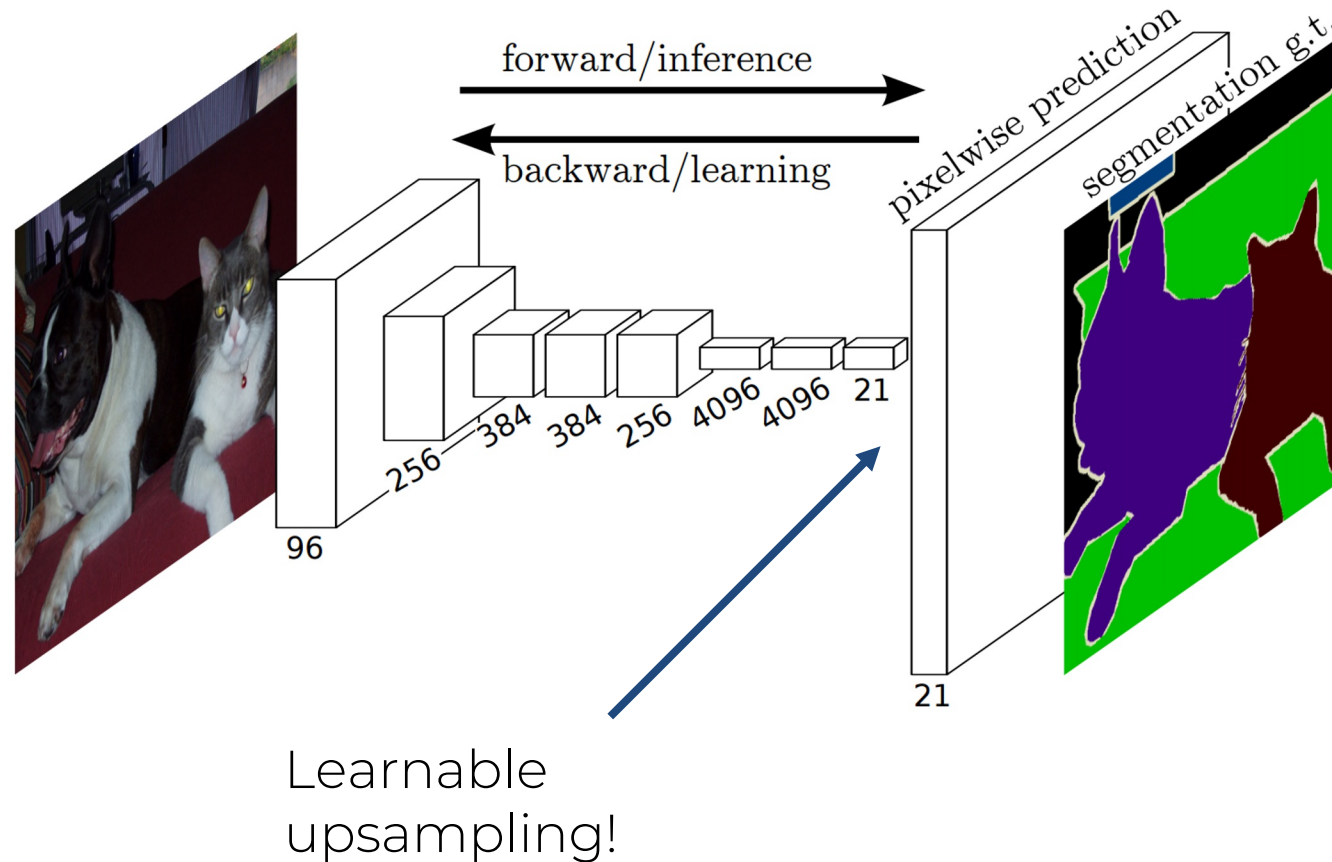
Upsampling: ???



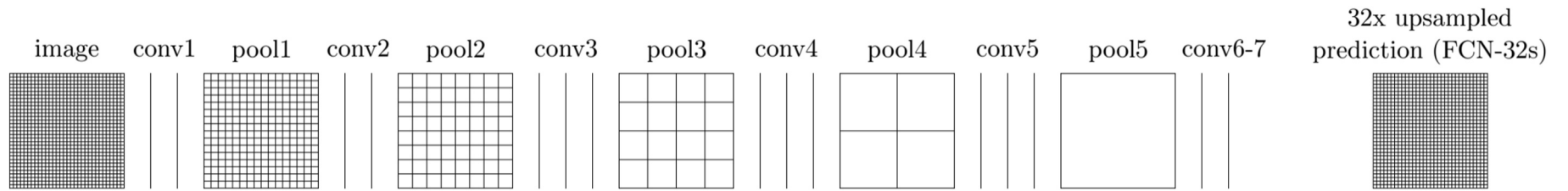
# Semantic Segmentation: Upsampling



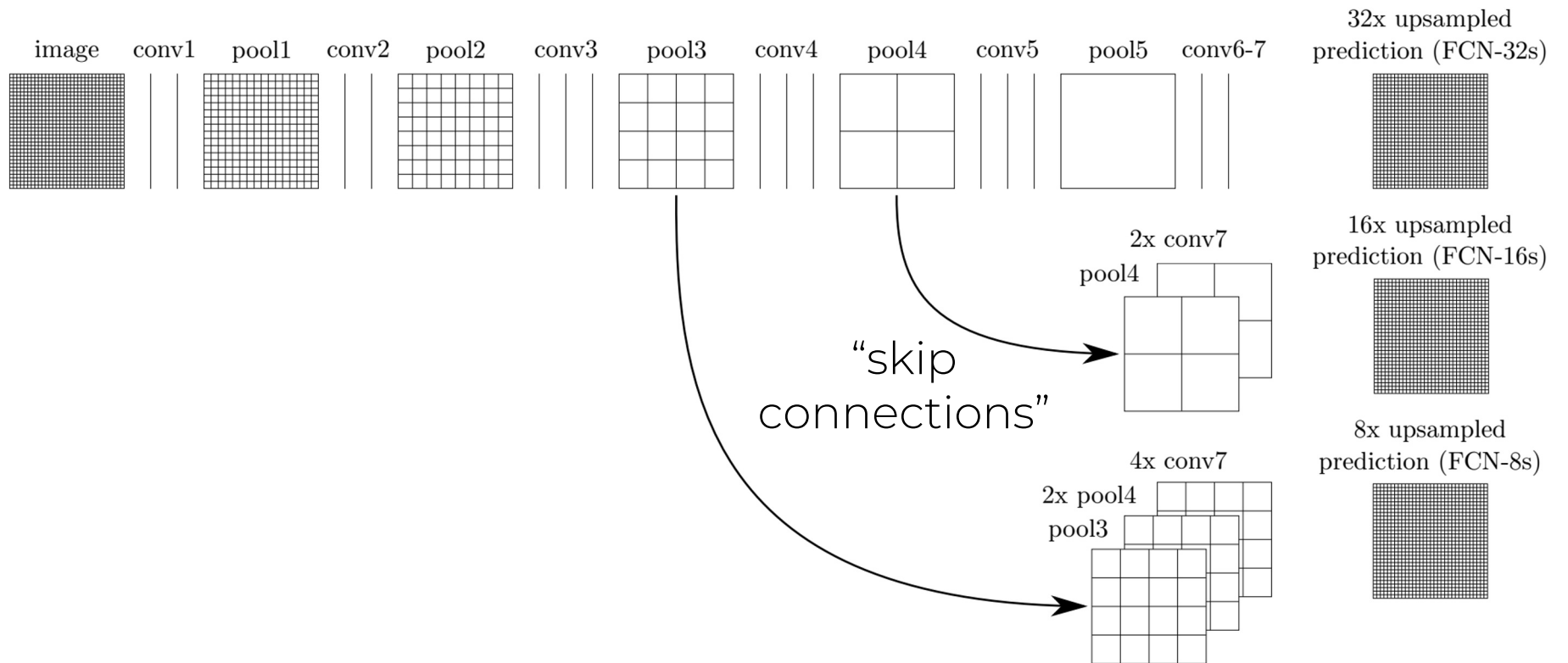
# Semantic Segmentation: Upsampling



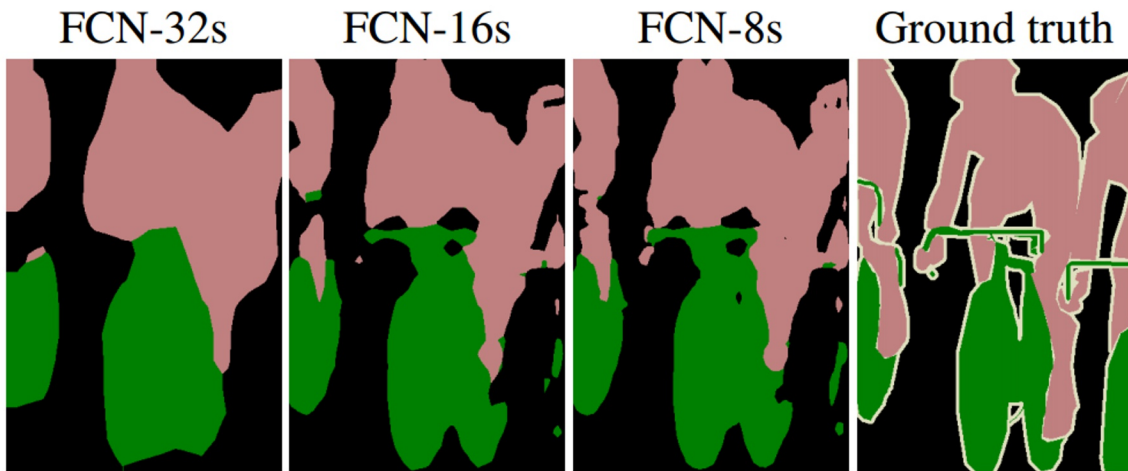
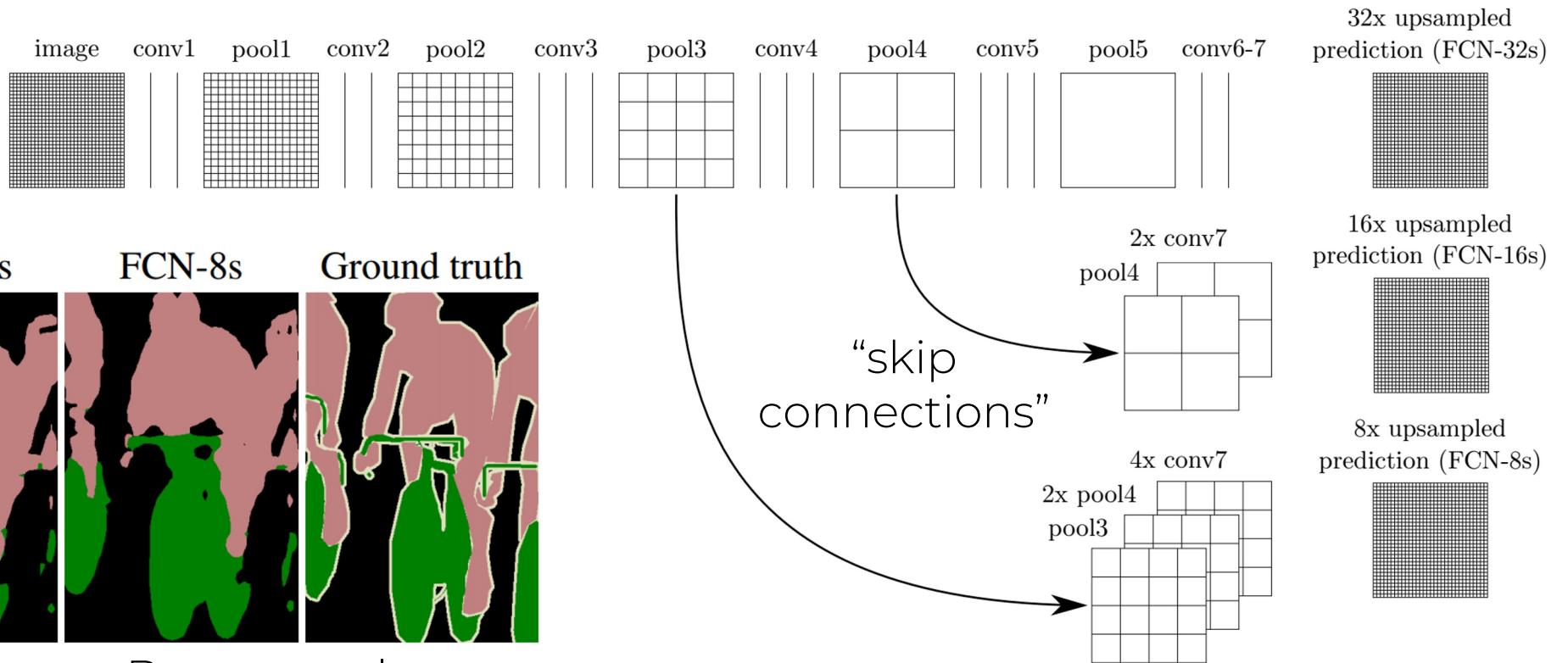
# Semantic Segmentation: Upsampling



# Semantic Segmentation: Upsampling



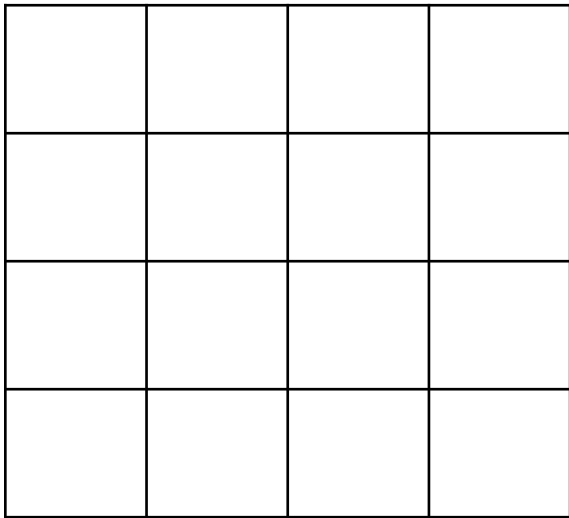
# Semantic Segmentation: Upsampling



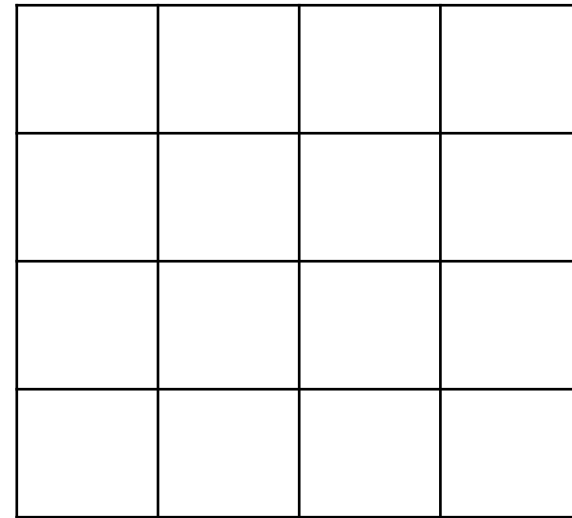
Skip connections = Better results

# Learnable Upsampling: “Deconvolution”

Typical 3 x 3 convolution, stride 1 pad 1



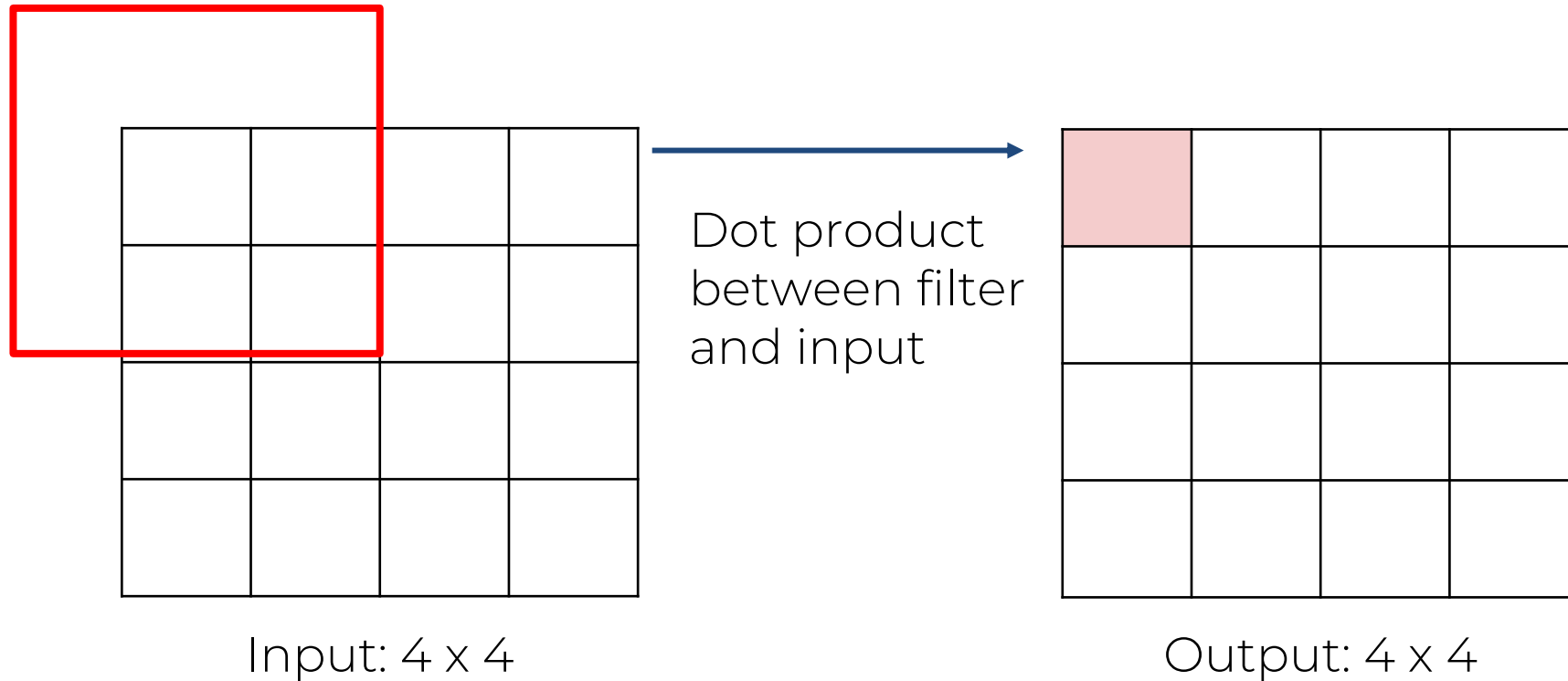
Input: 4 x 4



Output: 4 x 4

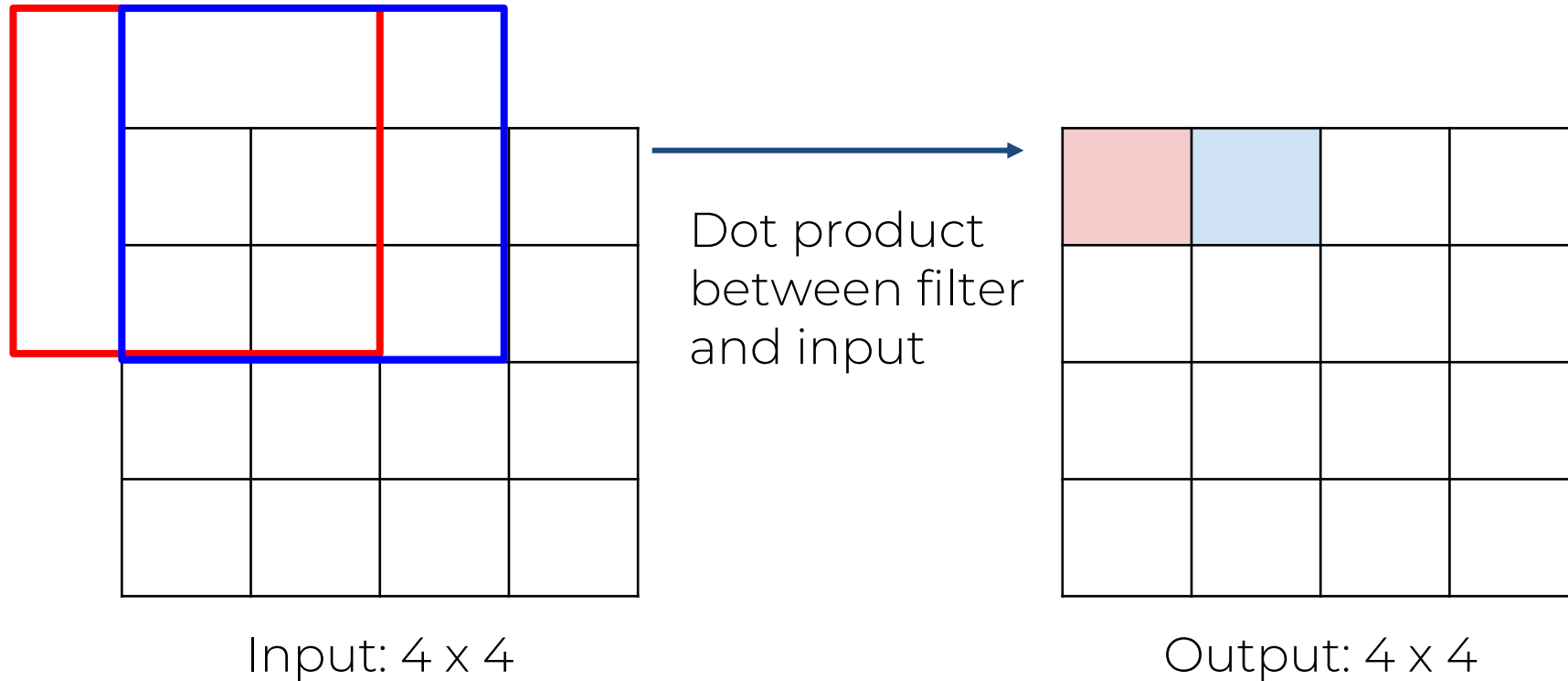
# Learnable Upsampling: “Deconvolution”

Typical 3 x 3 convolution, stride 1 pad 1



# Learnable Upsampling: “Deconvolution”

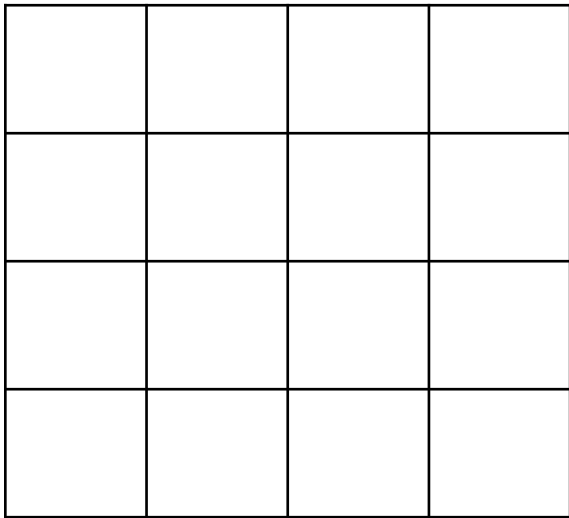
Typical 3 x 3 convolution, stride 1 pad 1



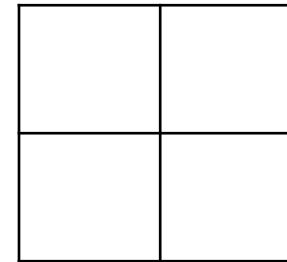


# Learnable Upsampling: “Deconvolution”

Typical 3 x 3 convolution, stride 2 pad 1



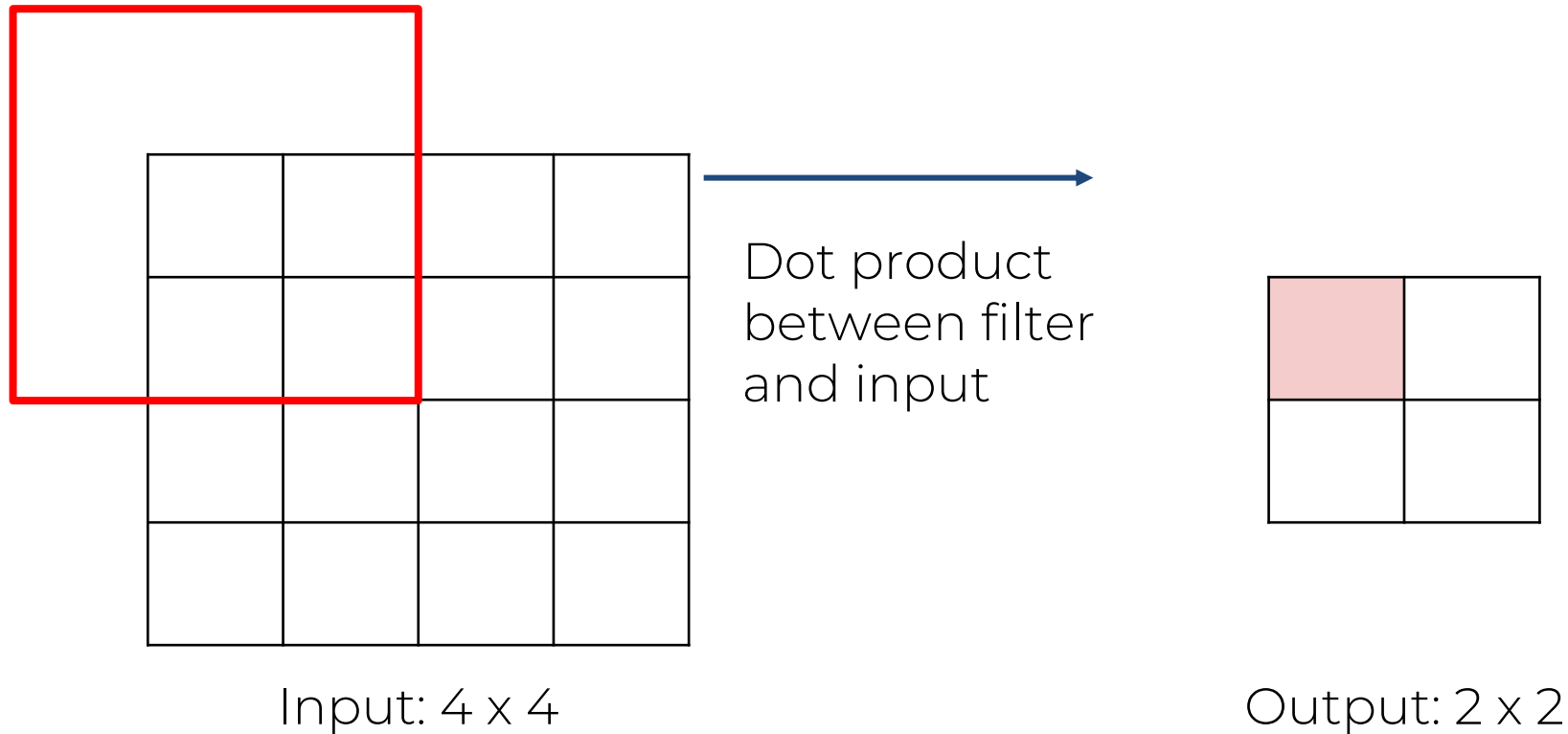
Input: 4 x 4



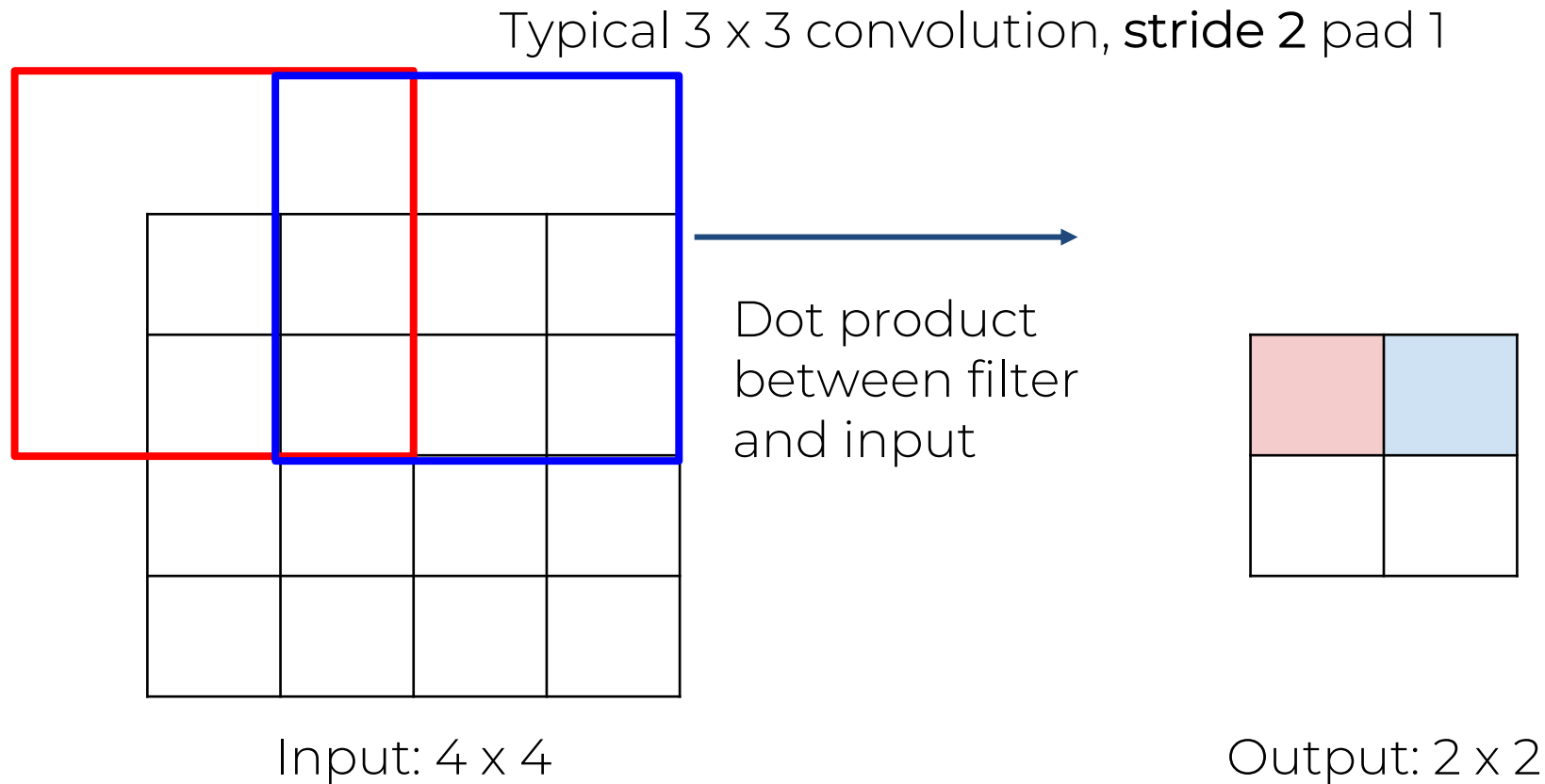
Output: 2 x 2

# Learnable Upsampling: “Deconvolution”

Typical 3 x 3 convolution, stride 2 pad 1

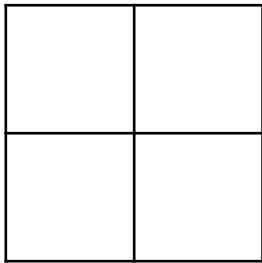


# Learnable Upsampling: “Deconvolution”

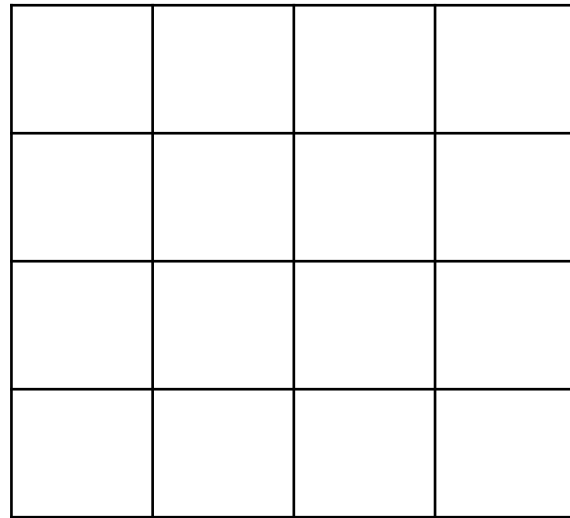


# Learnable Upsampling: “Deconvolution”

3 x 3 deconvolution, stride 2 pad 1



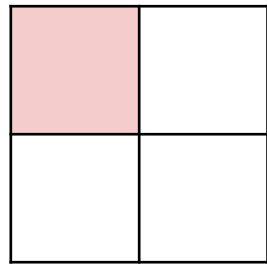
Input: 2 x 2



Output: 4 x 4

# Learnable Upsampling: “Deconvolution”

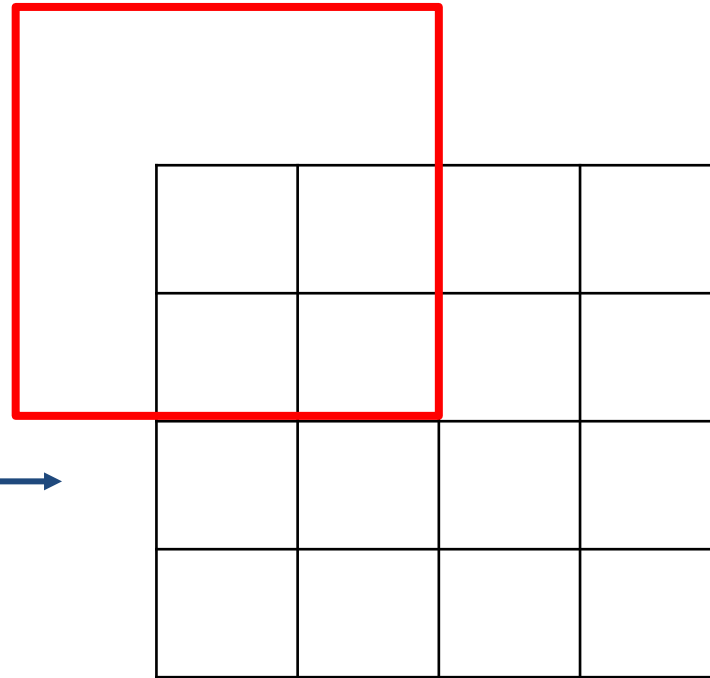
3 x 3 deconvolution, stride 2 pad 1



Input: 2 x 2



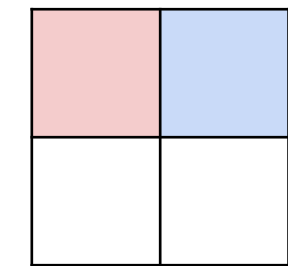
Input gives  
weight for  
filter



Output: 4 x 4

# Learnable Upsampling: “Deconvolution”

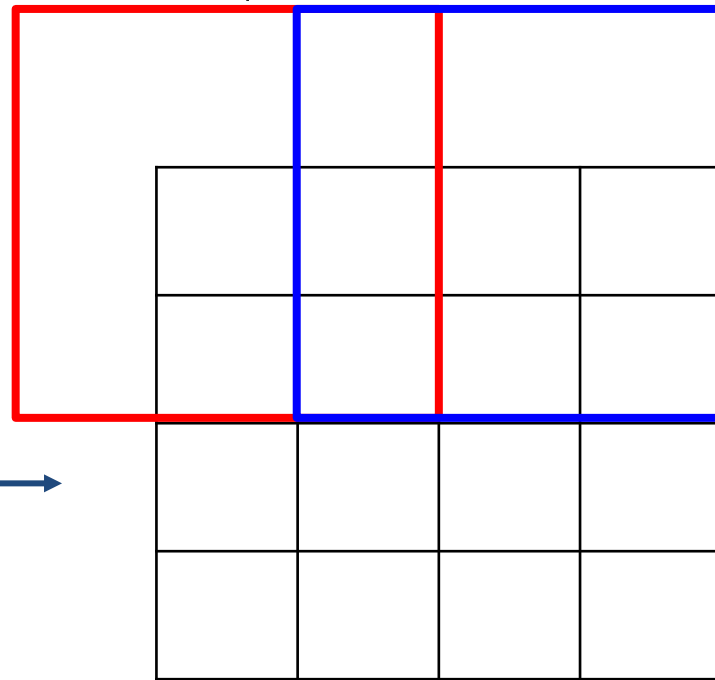
3 x 3 deconvolution, stride 2 pad 1



Input: 2 x 2

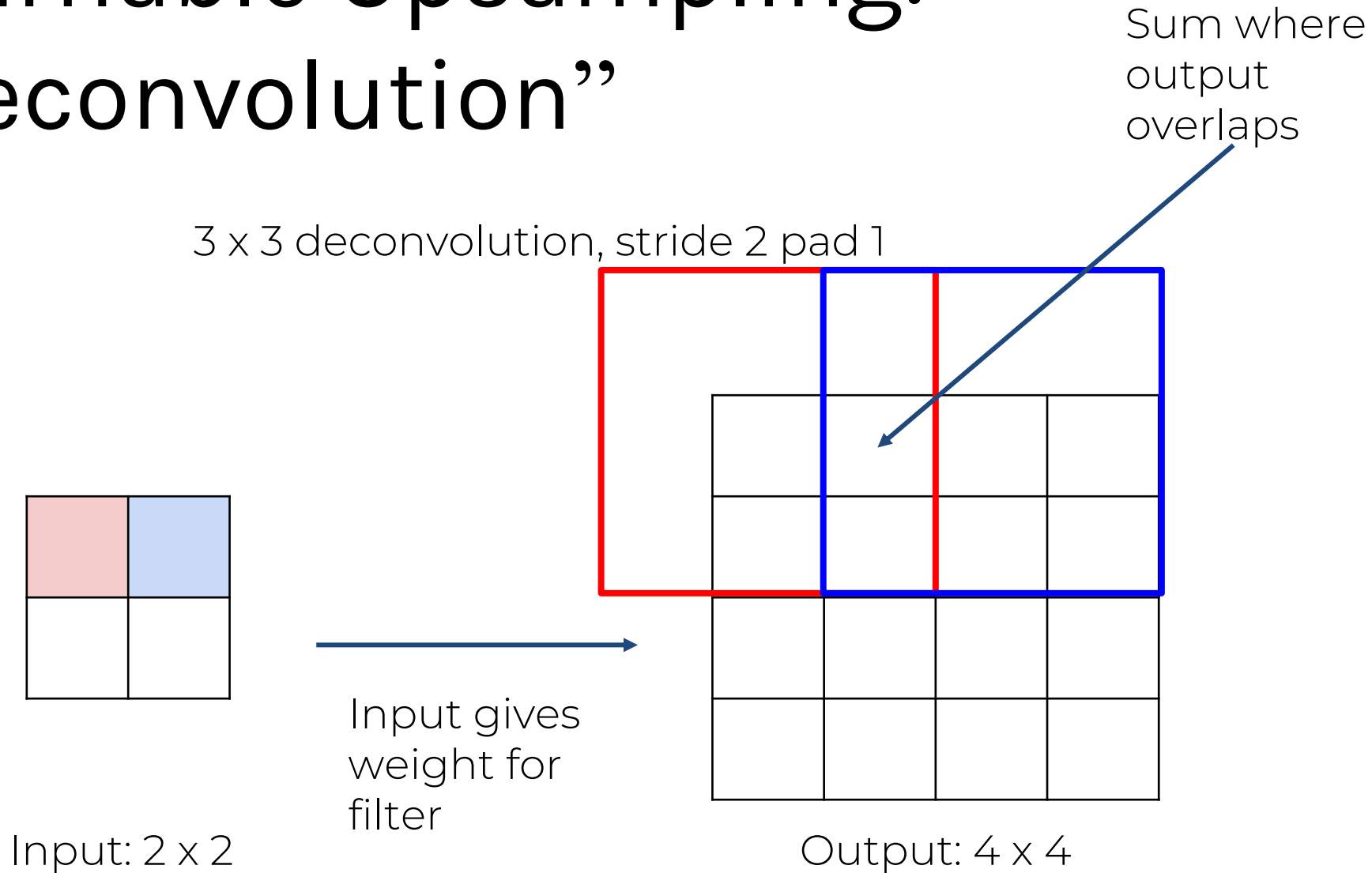


Input gives  
weight for  
filter

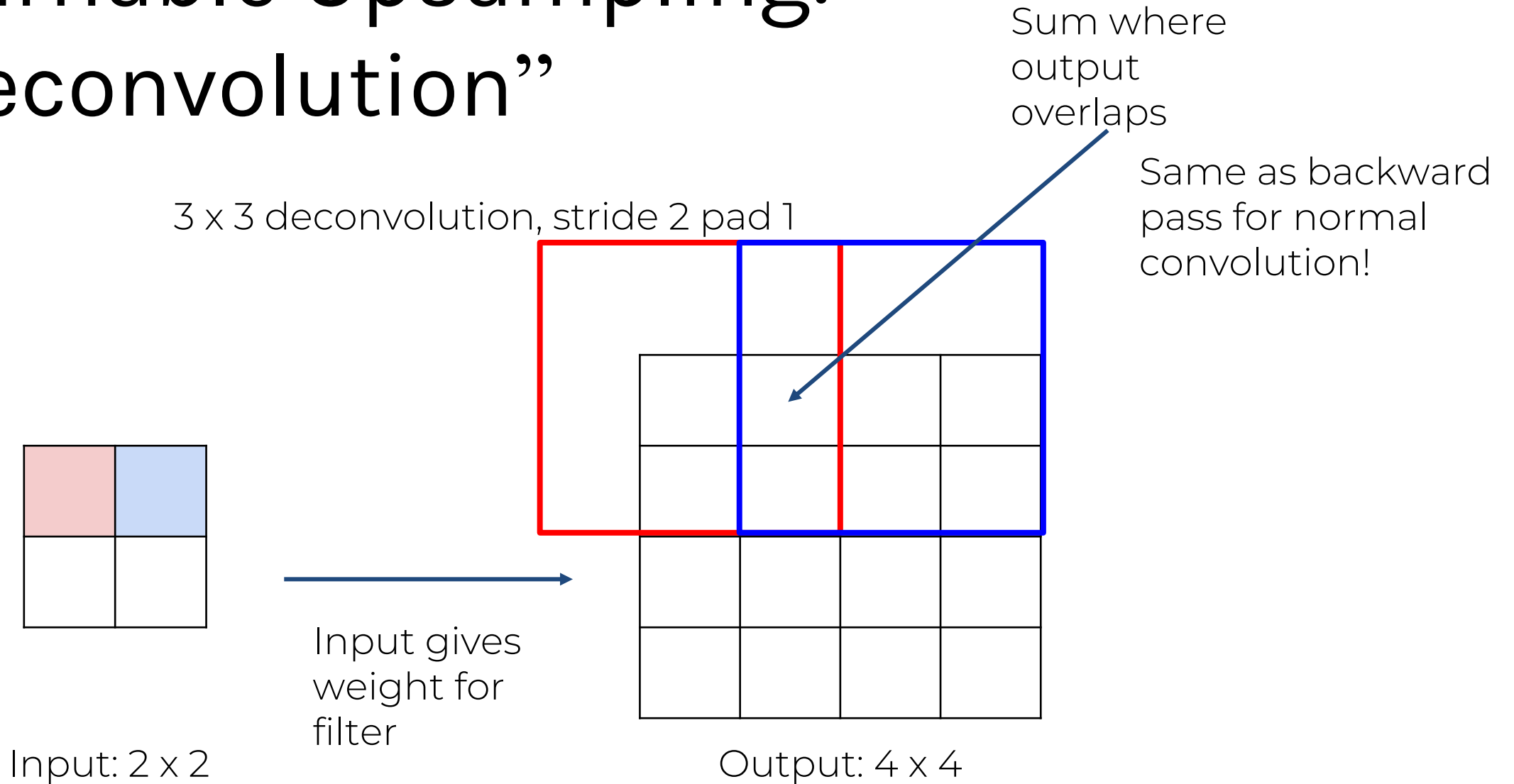


Output: 4 x 4

# Learnable Upsampling: “Deconvolution”

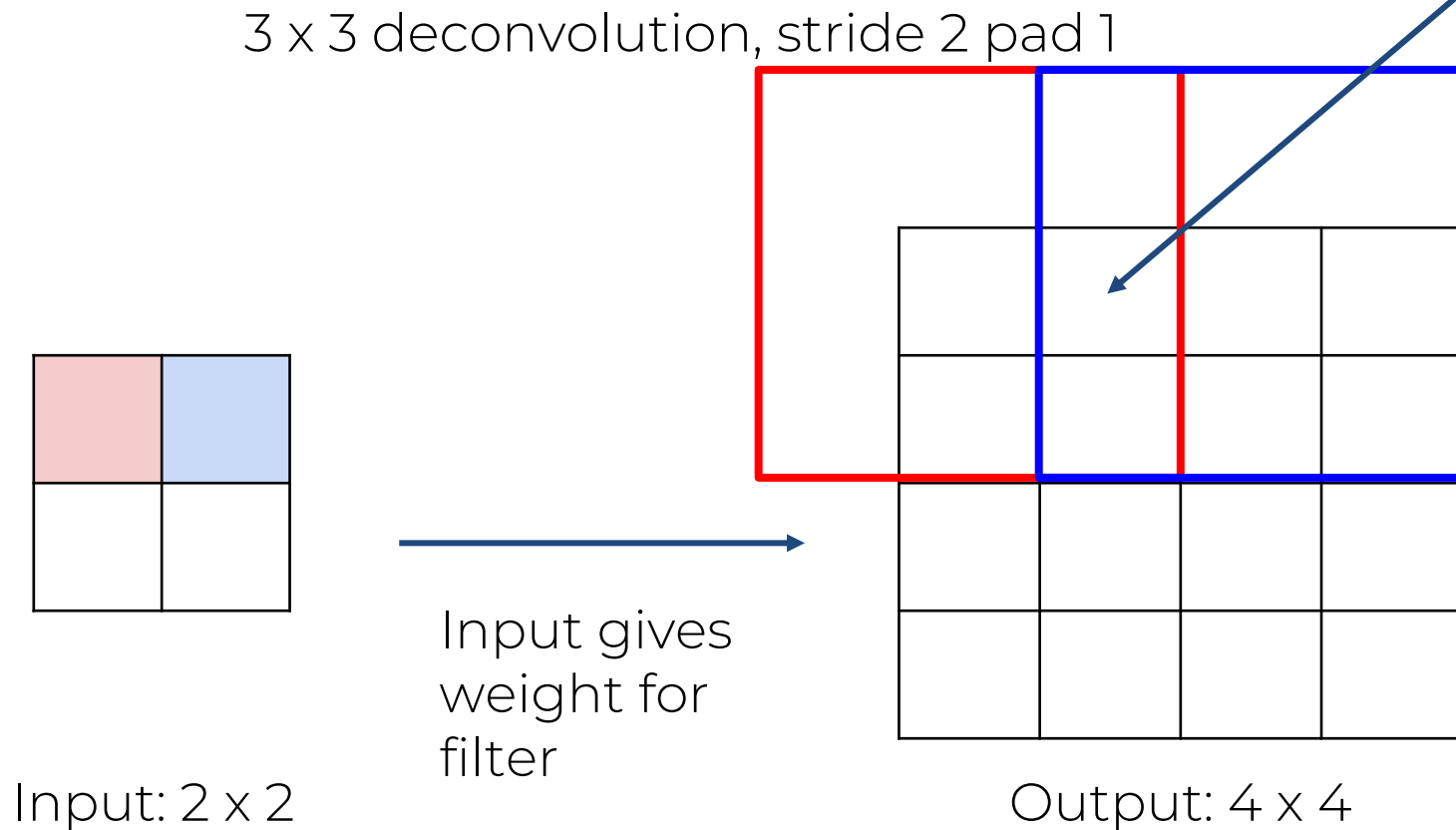


# Learnable Upsampling: “Deconvolution”





# Learnable Upsampling: “Deconvolution”



Sum where  
output  
overlaps

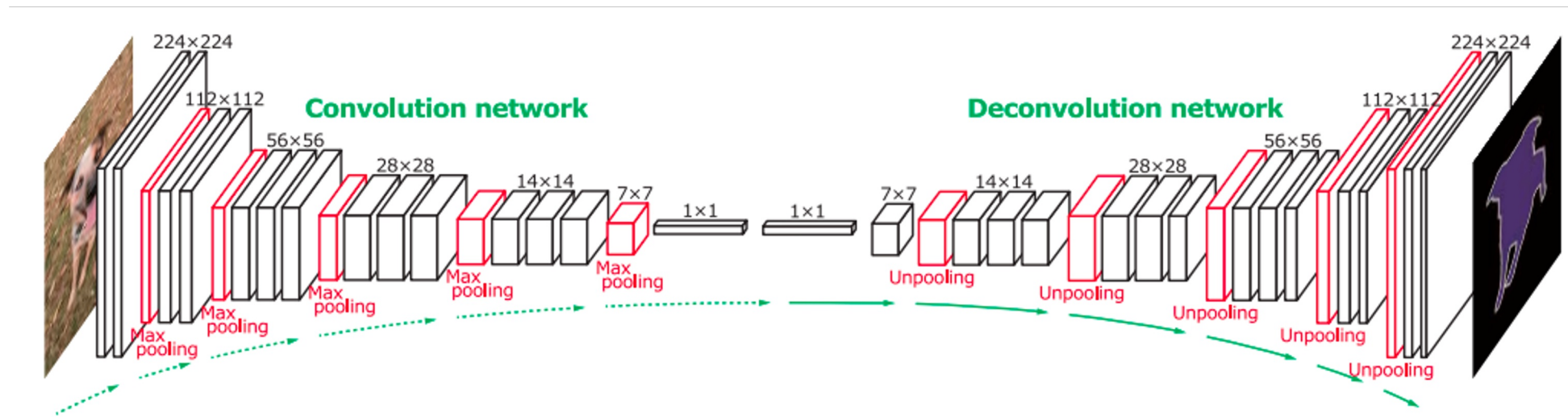
Same as backward  
pass for normal  
convolution!

“Deconvolution” is a bad  
name, already defined  
as “inverse of  
convolution”

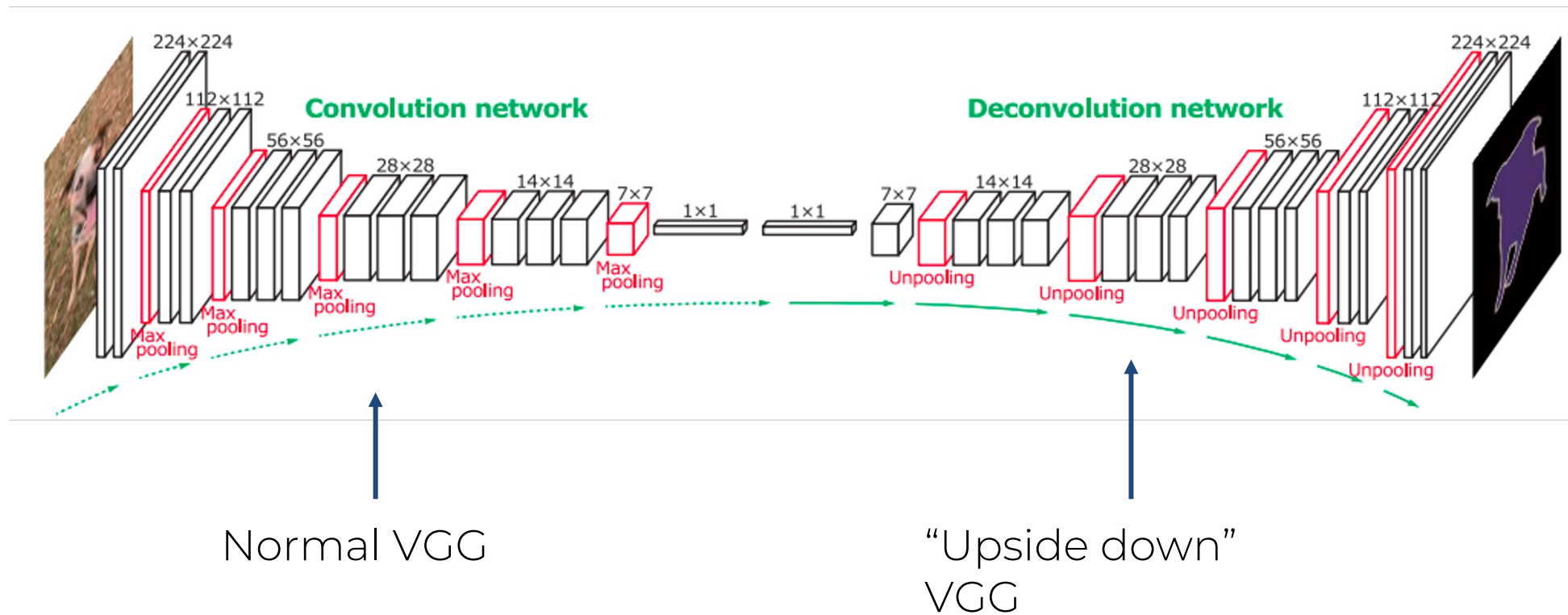
**Better names:**  
convolution transpose,  
backward strided  
convolution,  
1/2 strided convolution,  
upconvolution

F.-F. Li, A. Karpathy and J. Johnson

# Semantic Segmentation: Upsampling



# Semantic Segmentation: Upsampling



Normal VGG

“Upside down”  
VGG

6 days of training on Titan X...

# Instance Segmentation

Detect instances,  
give category, label  
pixels

“simultaneous  
detection and  
segmentation”  
(SDS)

Lots of recent work  
(MS-COCO)

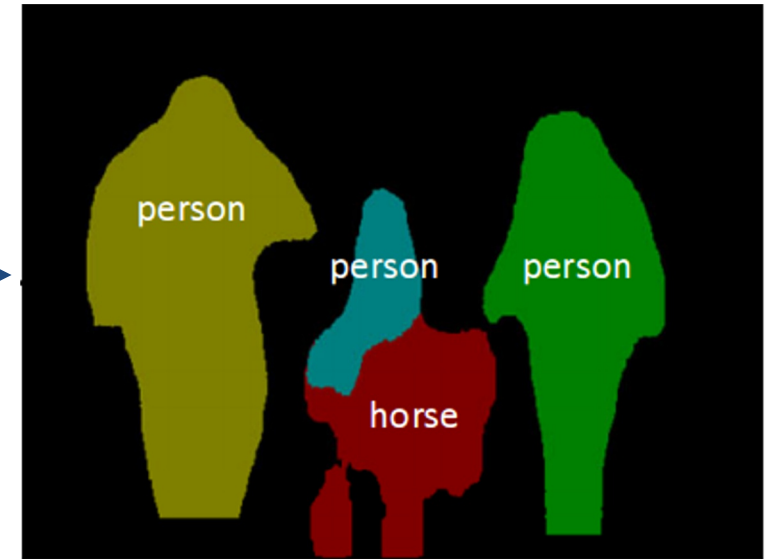


Figure credit: Dai et al, “Instance-aware Semantic Segmentation via Multi-task Network Cascades”,  
arXiv 2015

# Instance Segmentation

Similar to R-CNN,  
but with segments

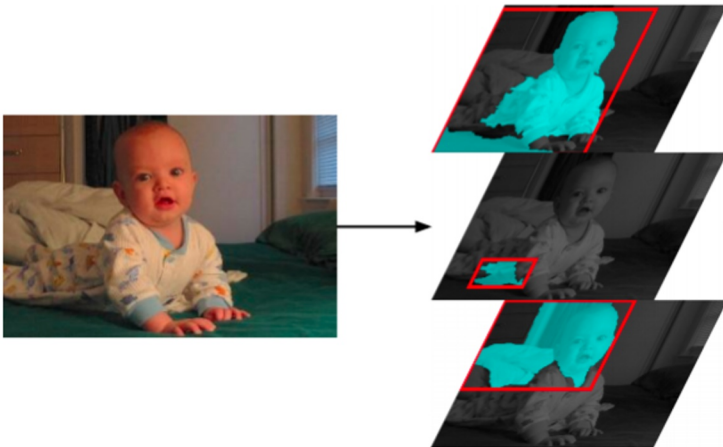


# Instance Segmentation

Similar to R-CNN,  
but with segments

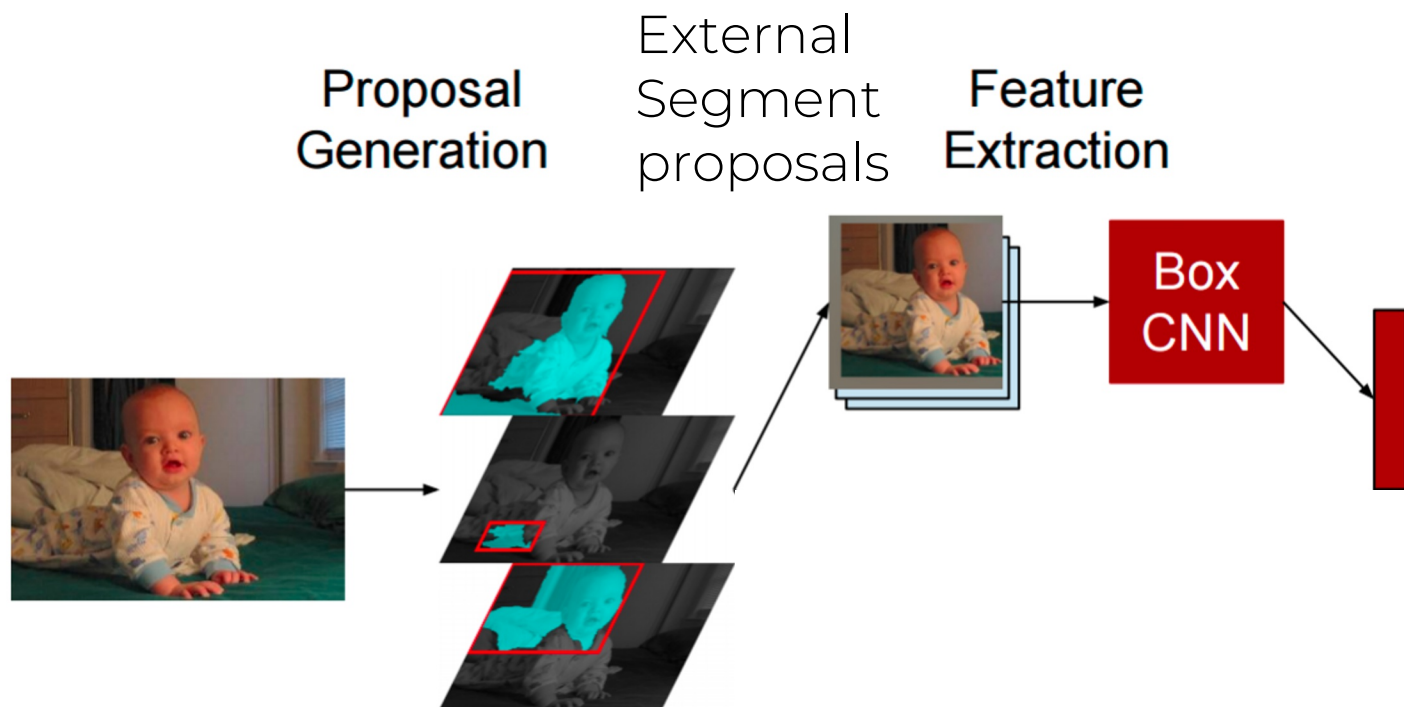
Proposal  
Generation

External  
Segment  
proposals



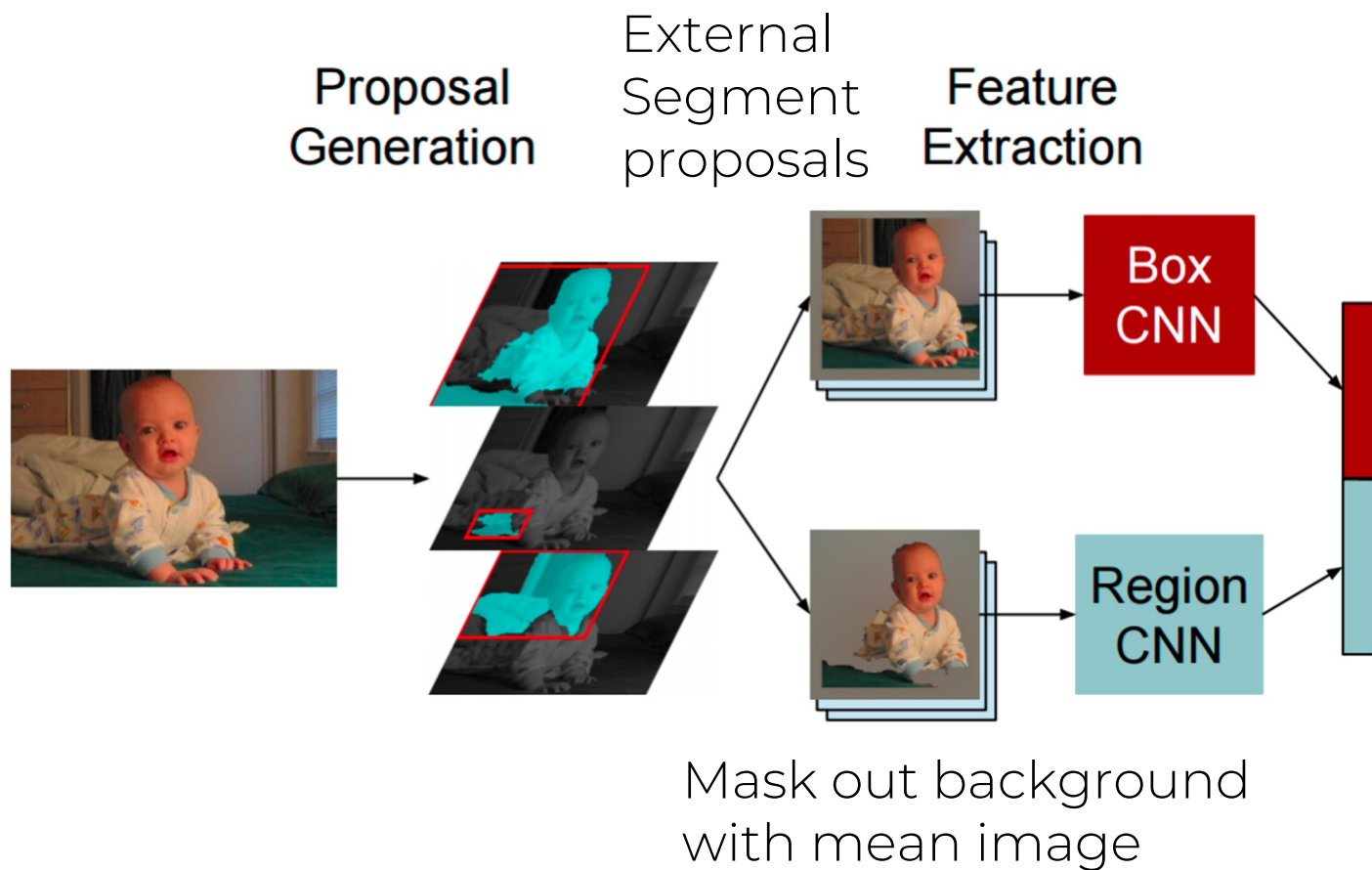
# Instance Segmentation

Similar to R-CNN,  
but with segments



# Instance Segmentation

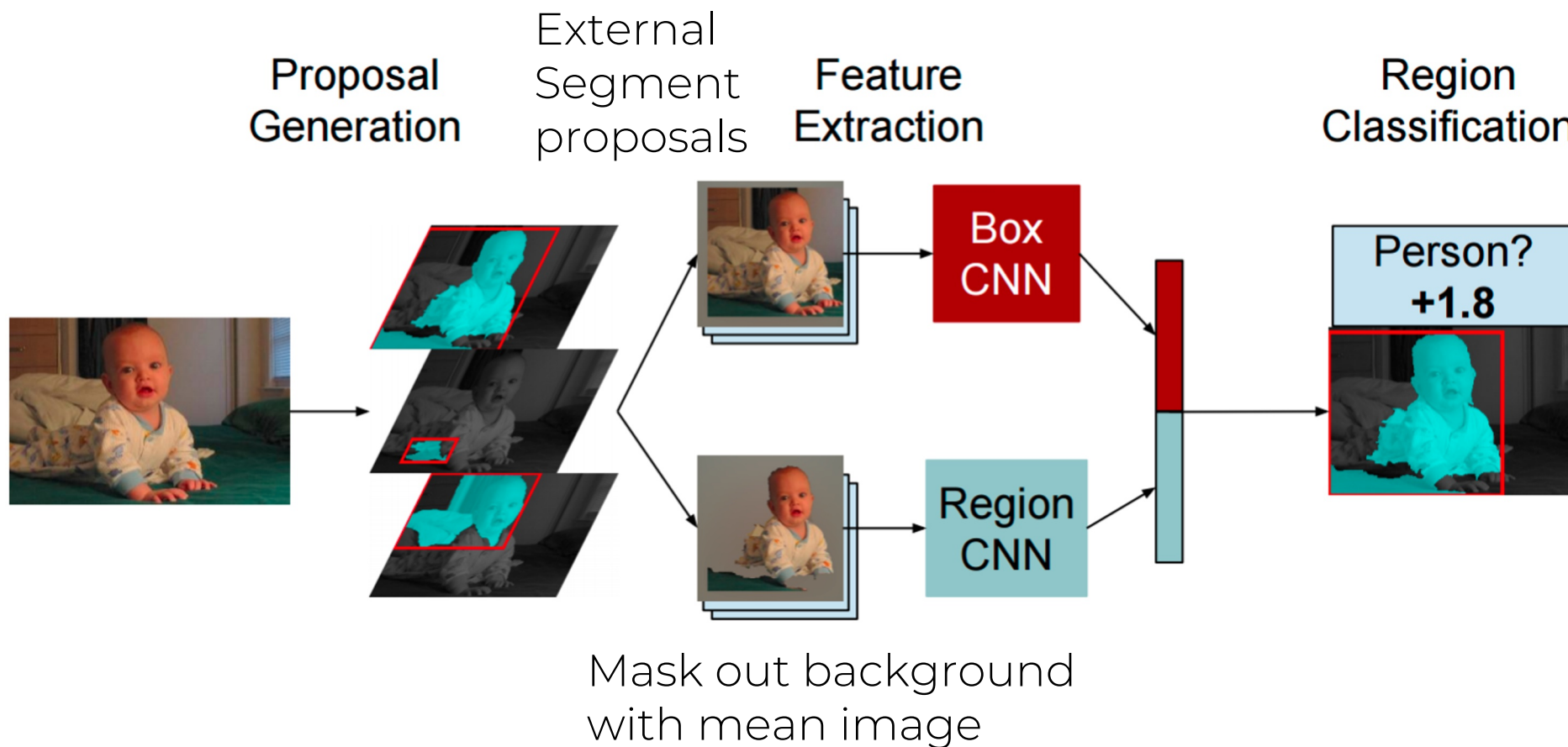
Similar to R-CNN,  
but with segments





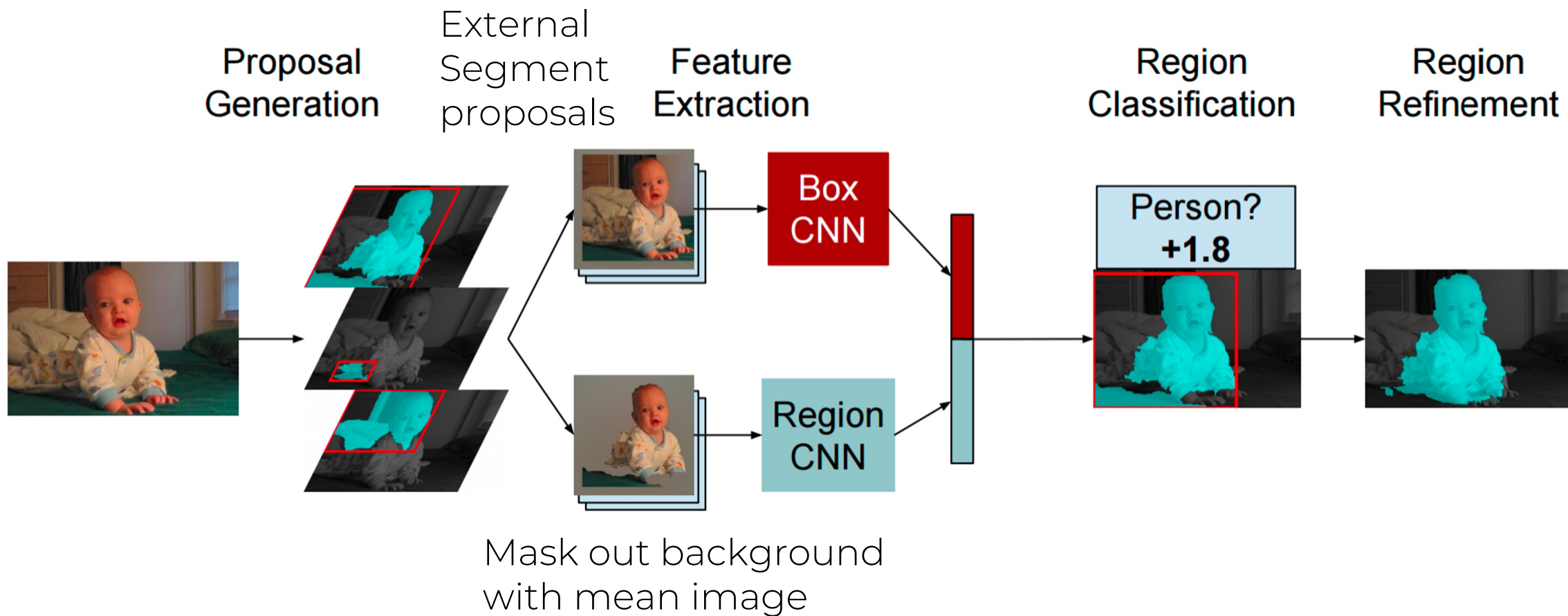
# Instance Segmentation

Similar to R-CNN,  
but with segments



# Instance Segmentation

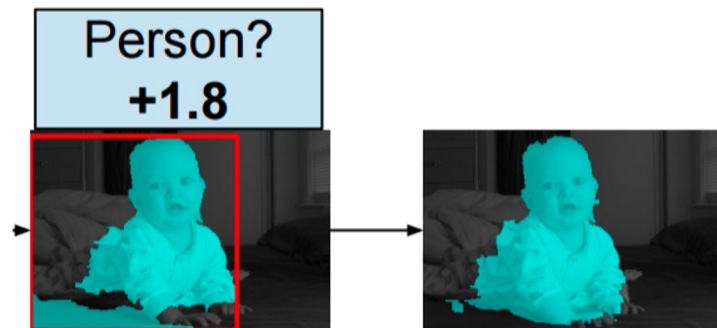
Similar to R-CNN,  
but with segments



# Instance Segmentation: Hypercolumns

Region  
Classification

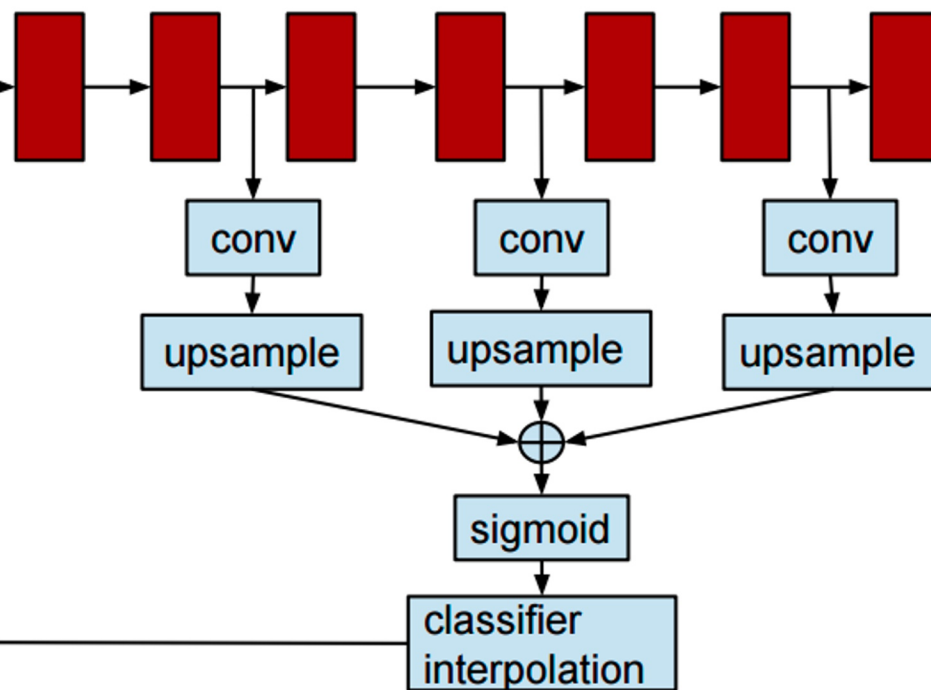
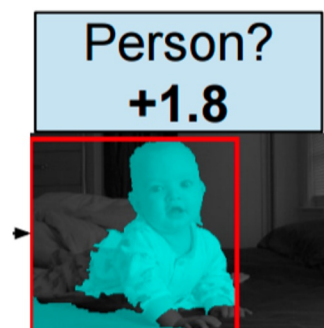
Region  
Refinement



# Instance Segmentation: Hypercolumns

Region Classification

Region Refinement



# Instance Segmentation: Cascades

Similar to  
Faster R-CNN



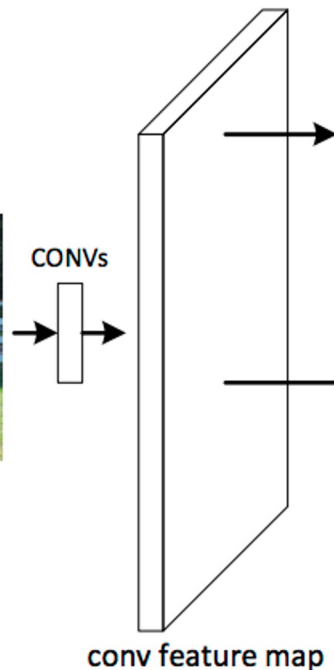
Won COCO 2015  
challenge  
(with ResNet)

Dai et al, "Instance-aware Semantic Segmentation via Multi-task Network Cascades",  
arXiv 2015

F.-F. Li, A. Karpathy and J. Johnson

# Instance Segmentation: Cascades

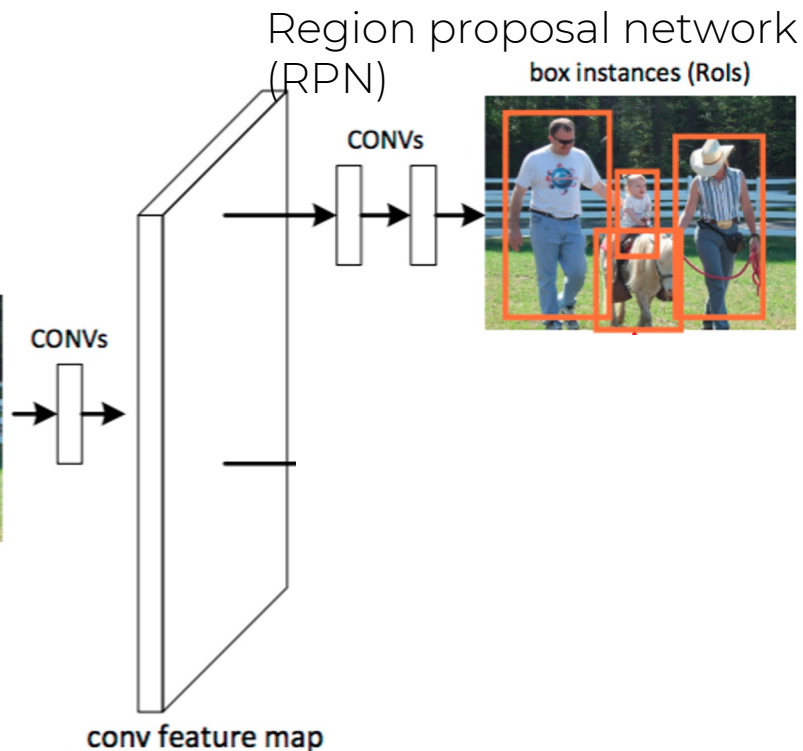
Similar to  
Faster R-CNN



Won COCO 2015  
challenge  
(with ResNet)

# Instance Segmentation: Cascades

Similar to  
Faster R-CNN



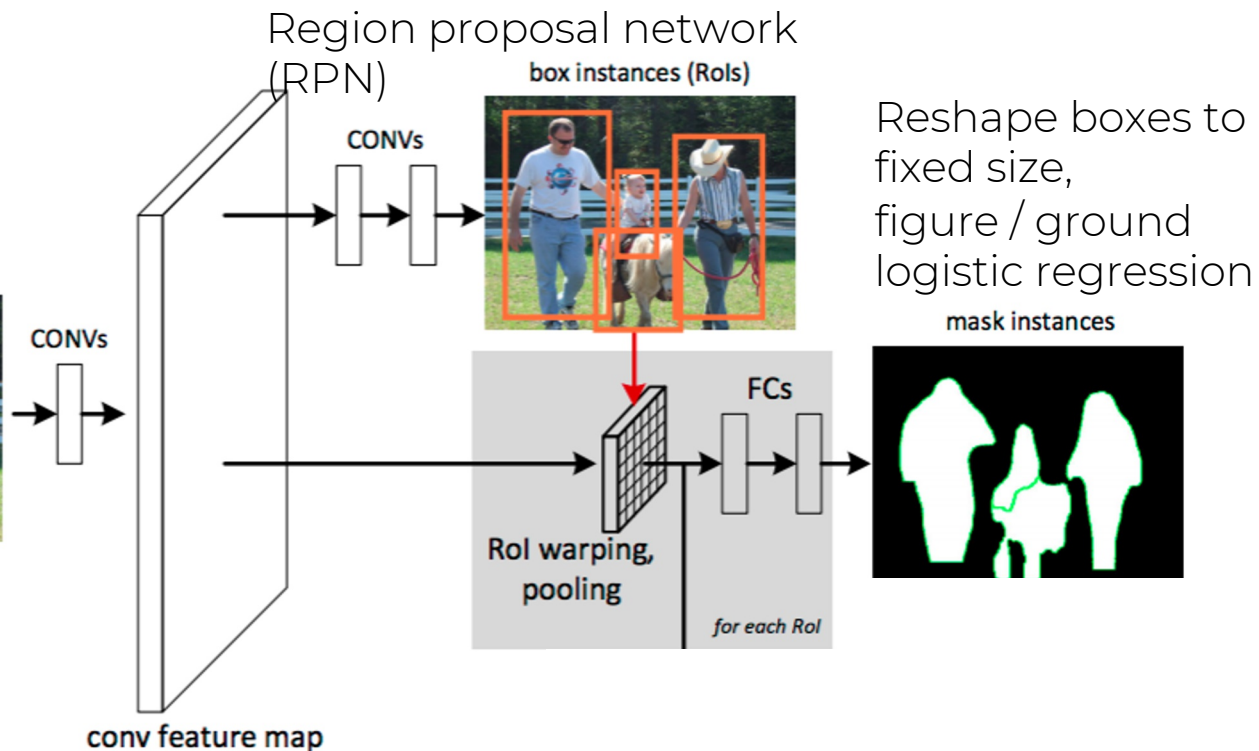
Won COCO 2015  
challenge  
(with ResNet)

# Instance Segmentation: Cascades

Similar to  
Faster R-CNN



Won COCO 2015  
challenge  
(with ResNet)



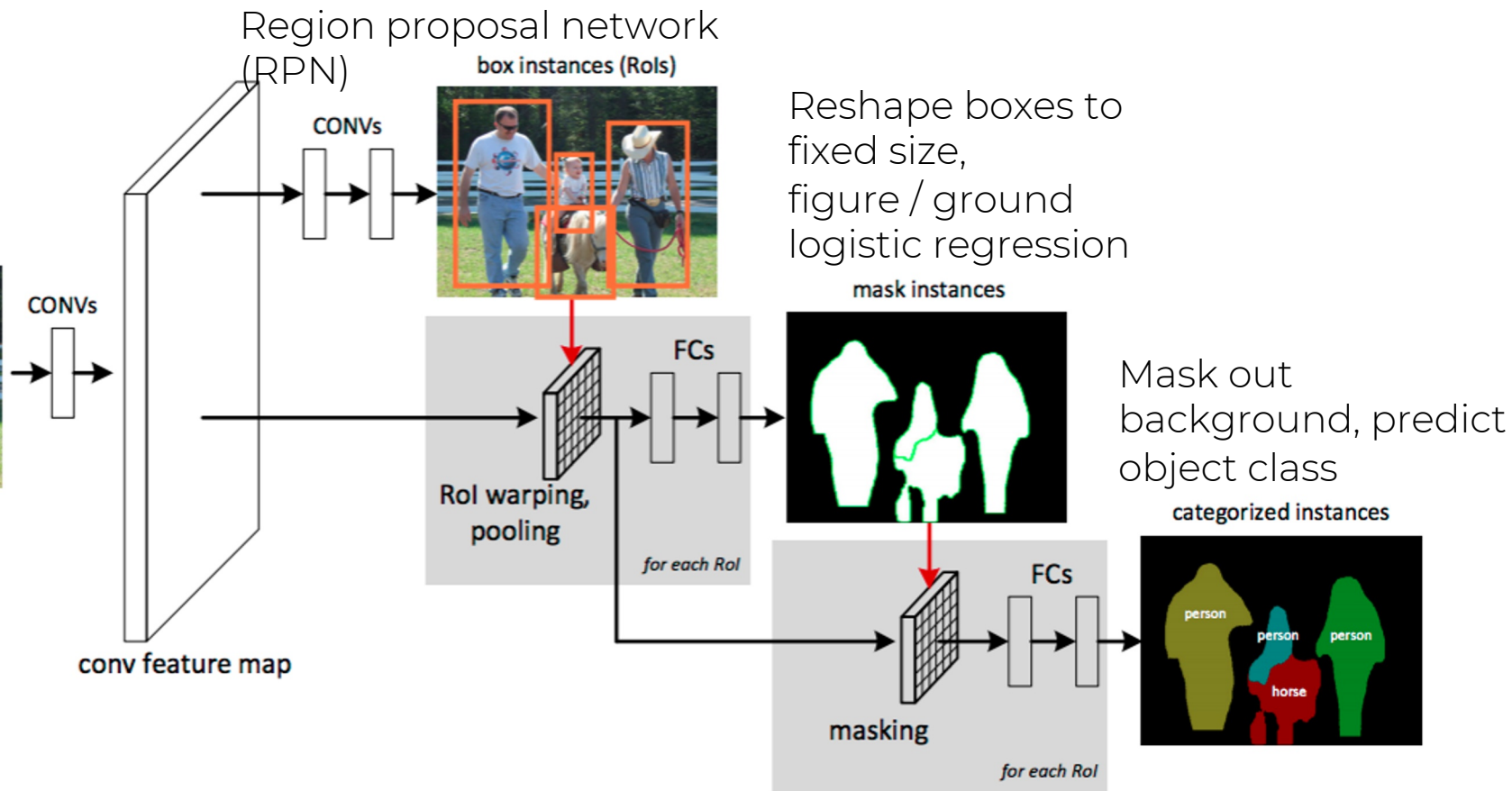


# Instance Segmentation: Cascades

Similar to  
Faster R-CNN



Won COCO 2015  
challenge  
(with ResNet)

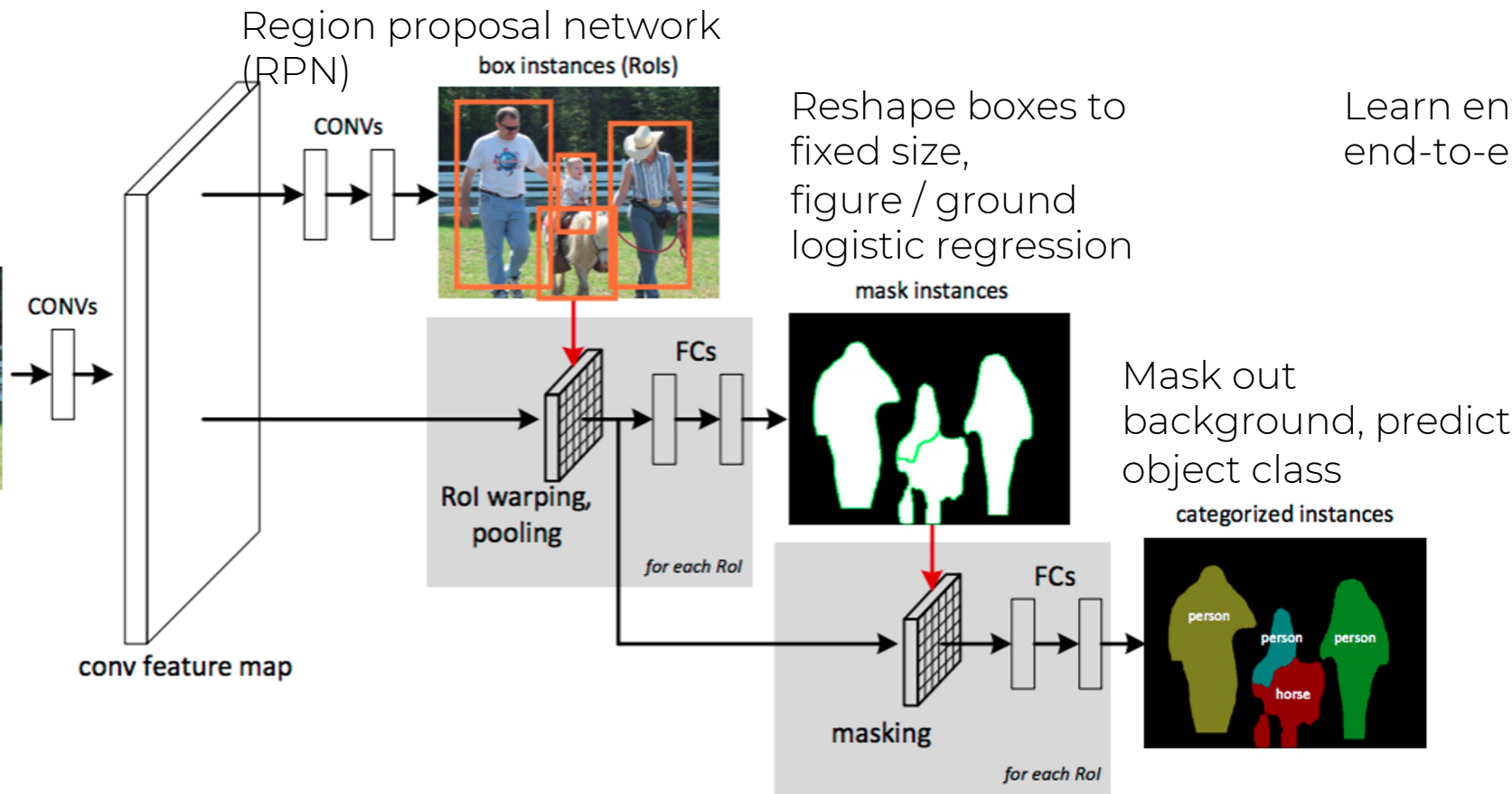


# Instance Segmentation: Cascades

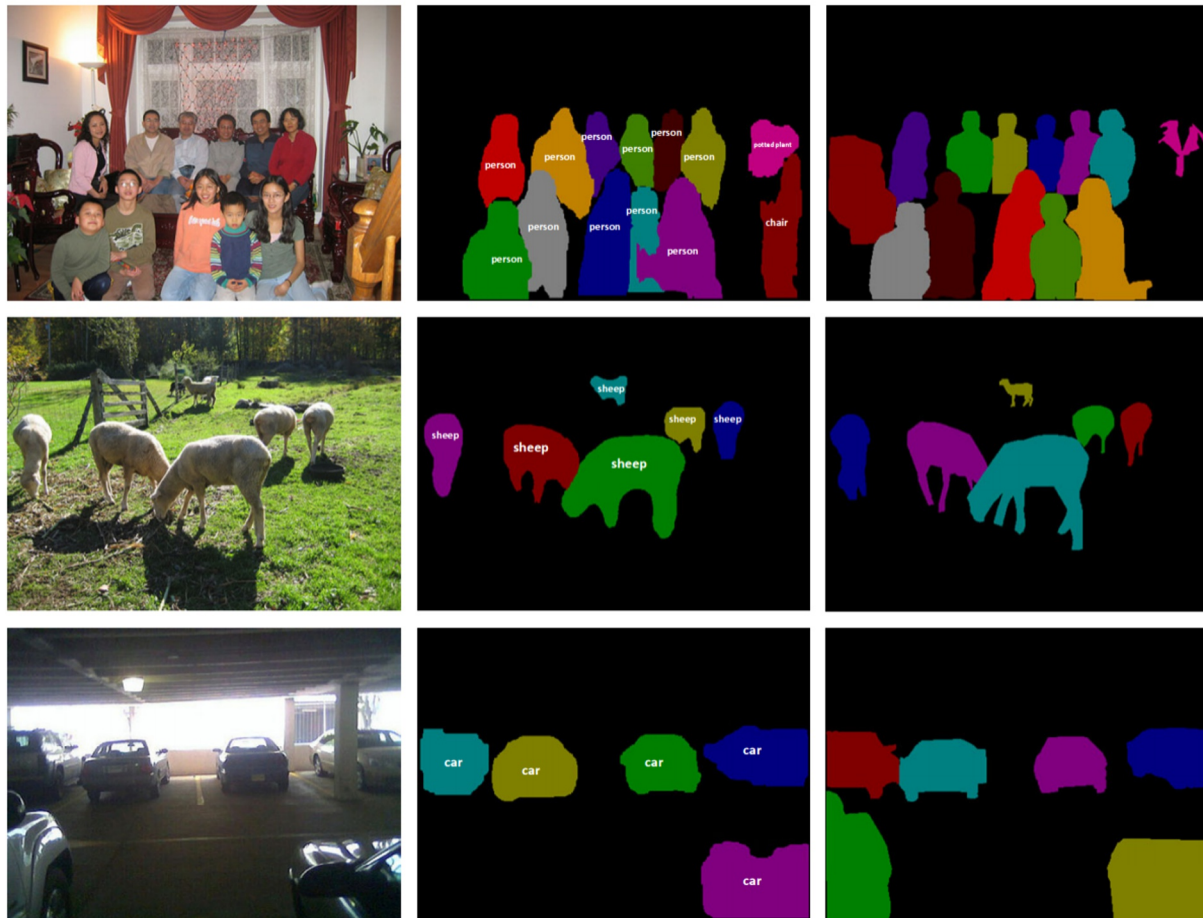
Similar to  
Faster R-CNN



Won COCO 2015  
challenge  
(with ResNet)



# Instance Segmentation: Cascades




Predictions

Ground truth

# Mask R-CNN: Model Overview

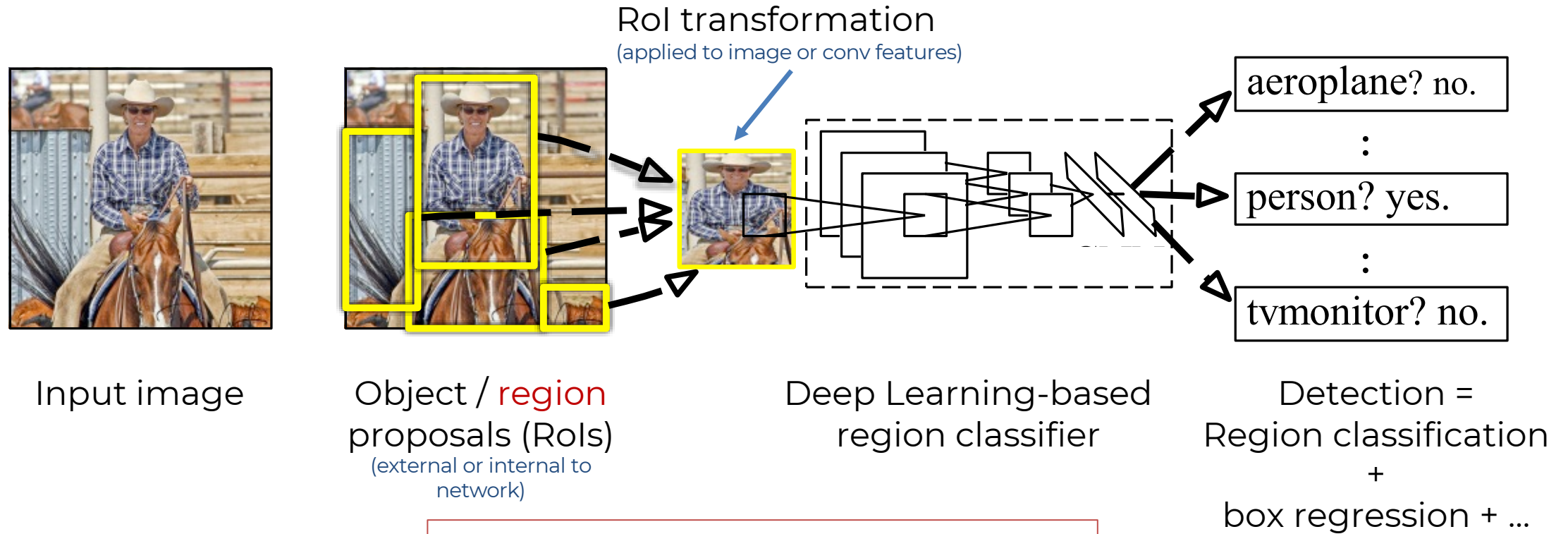
## R-CNN-style detection system

1. Backbone architecture
2. Feature Pyramid Network (FPN)
3. Region Proposal Network (RPN)
4. Region of interest feature alignment (RoIAlign)
5. Multi-task network head
  - Box classifier
  - Box regressor
  - Mask predictor
  - Keypoint predictor



Modular  
composition  
of  
many recent  
ideas

# 0. R-CNN-style Approach to Object Detection



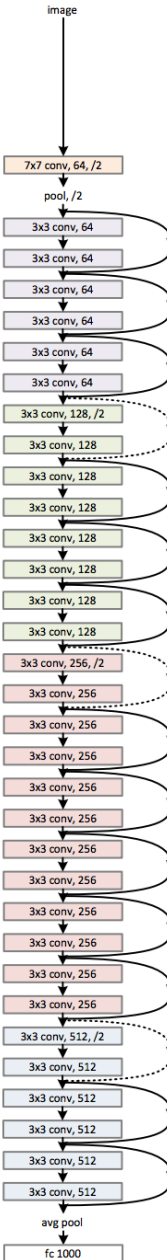
General formula for  
**Region-based CNN** models

# 1. Backbone ConvNet

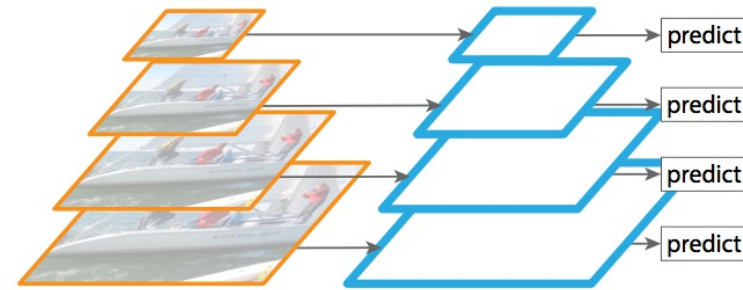
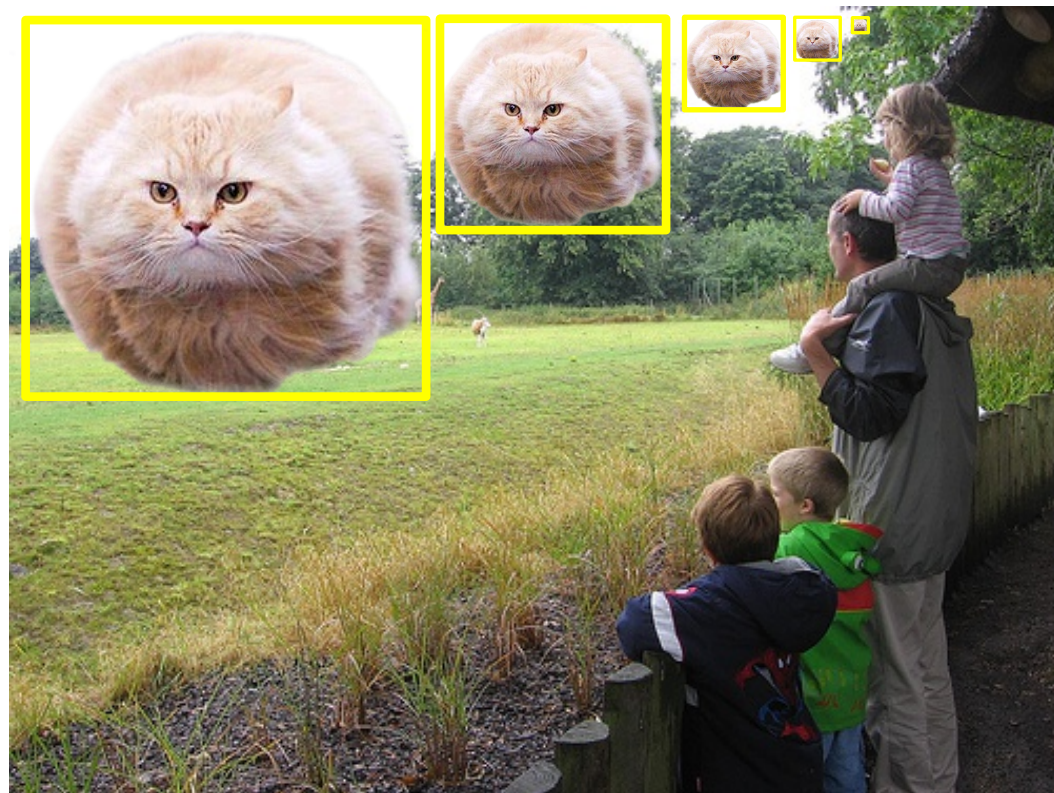
Use **any standard ConvNet** as a “backbone architecture”

- AlexNet, VGG, ResNet, Inception, Inception-ResNet, ResNeXt, ...
- Use “same” padding everywhere (preserves integer scales)
- Prefer fully convolutional networks (ignoring cls head; see next slide)
- **Pre-train on ImageNet** classification (or similar)

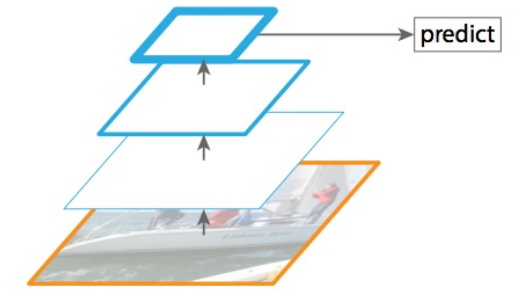
Example:  
ResNet-34



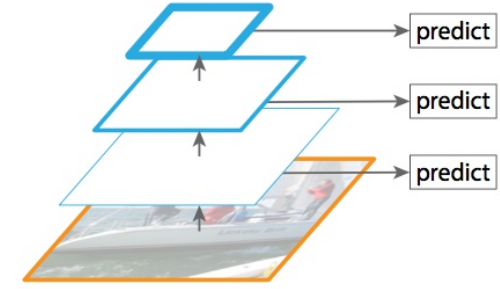
# 2. Scale Invariant Detection



(a) Featurized image pyramid  
SLOW!  
(Viola & Jones, HOG, DPM, multiscale Fast R-CNN, ...)



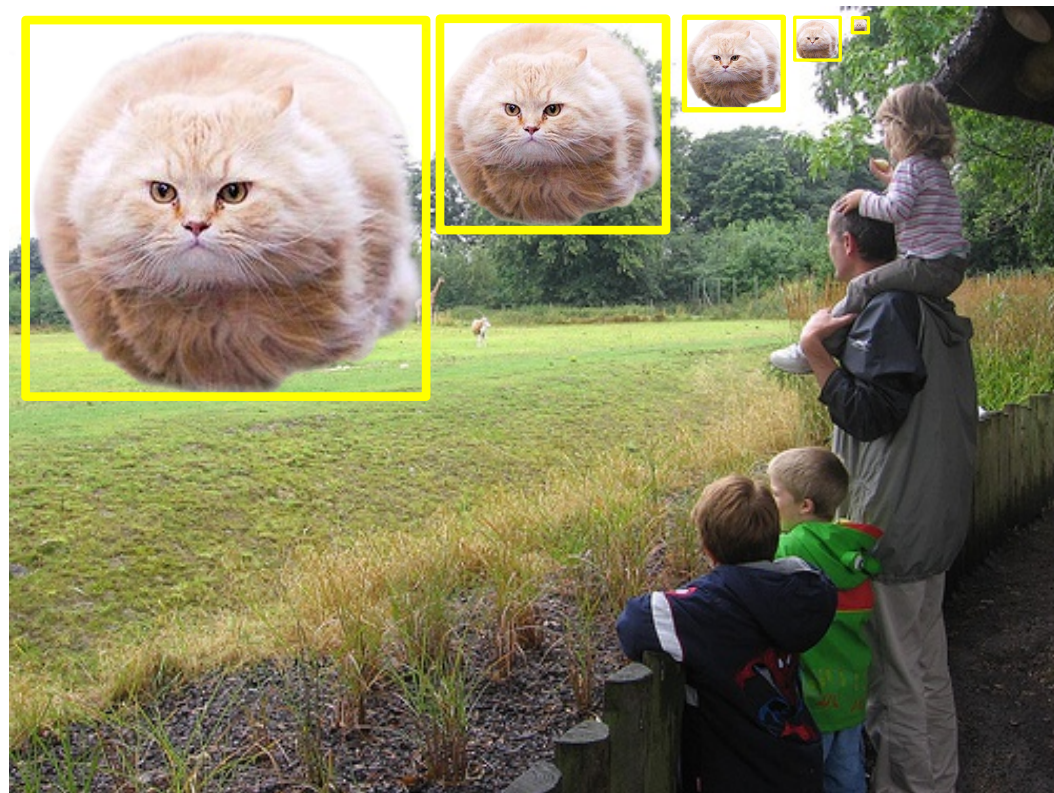
(b) Single feature map  
Do nothing; give up; Fast  
(Fast R-CNN, YOLO, ...)



(c) Pyramidal feature hierarchy  
Fast!  
( $\approx$  SSD, ...)

Popular strategies for detecting objects over large scale changes

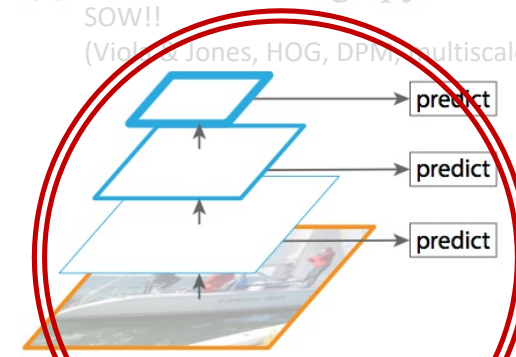
# 2. Scale Invariant Detection



(a) Featurized image pyramid  
SOW!!  
(Viola & Jones, HOG, DPM, multiscale Fast R-CNN, ...)



(b) Single feature map  
Do nothing; give up; Fast!  
(Fast R-CNN, YOLO, ...)

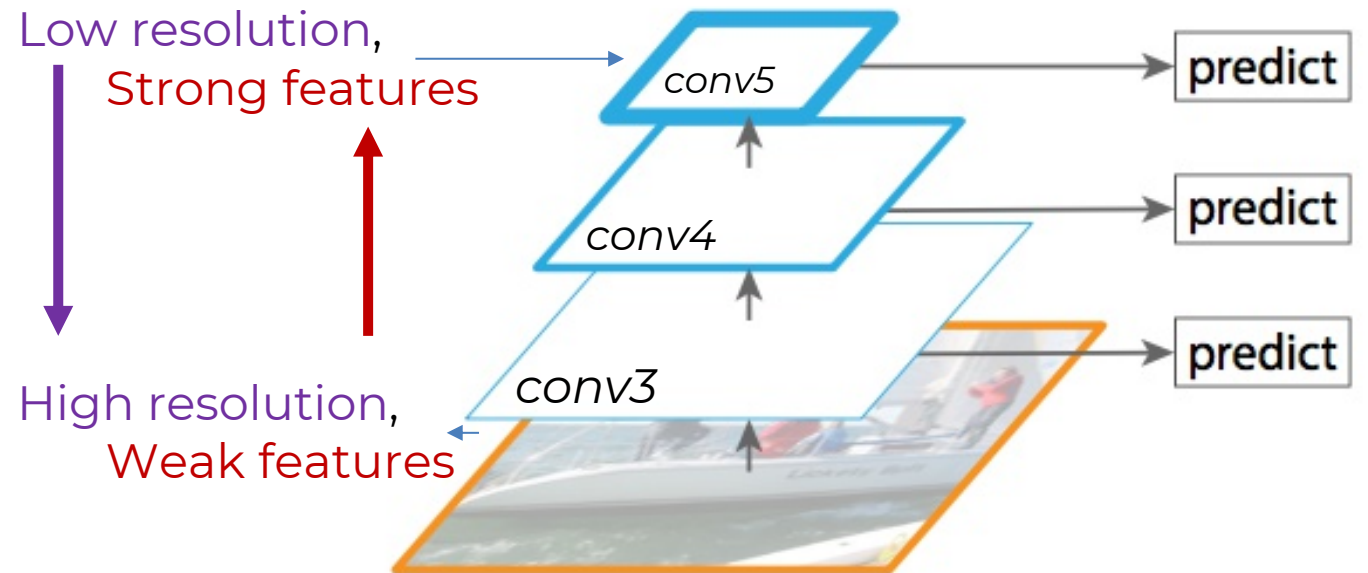
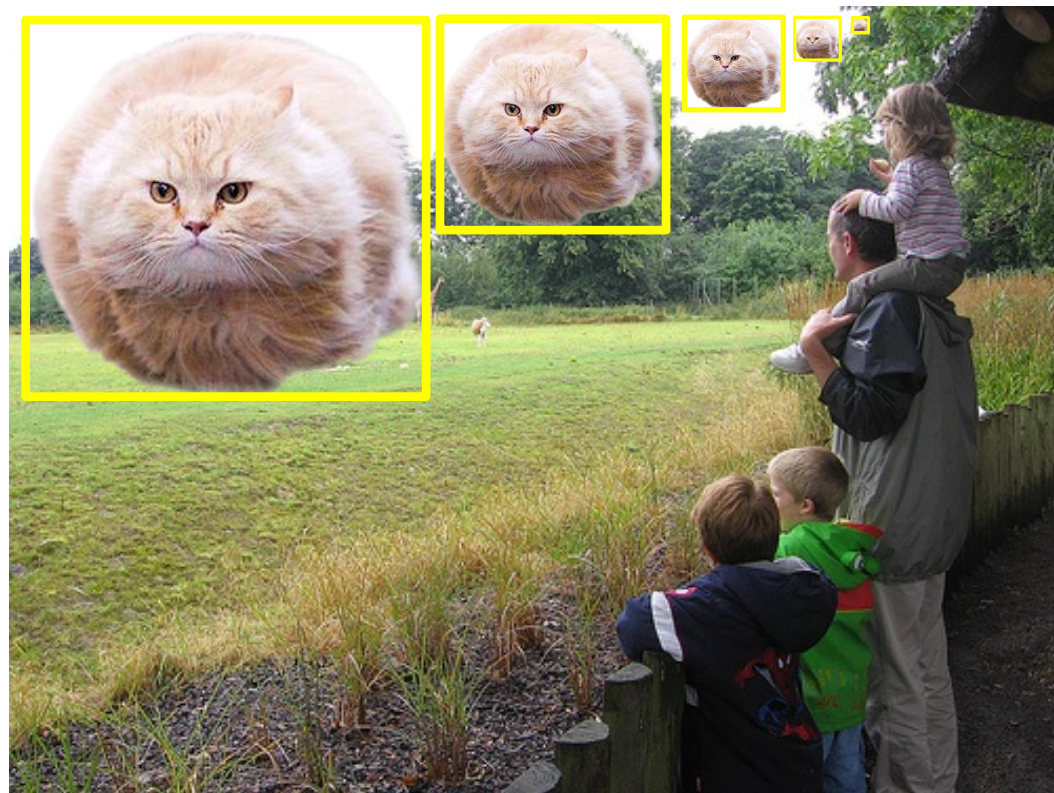


(c) Pyramidal feature hierarchy  
Fast!  
( $\approx$  SSD, ...)

Let's examine this one



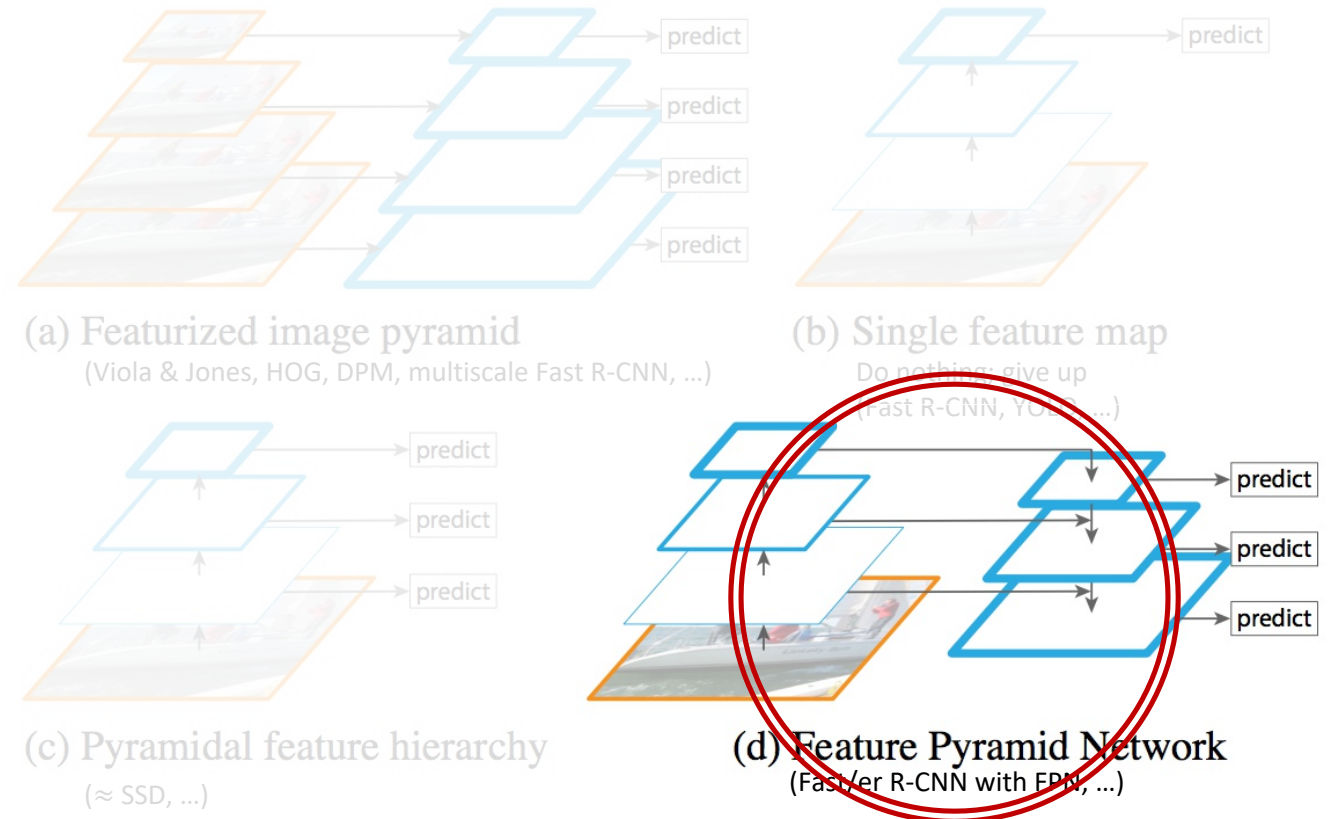
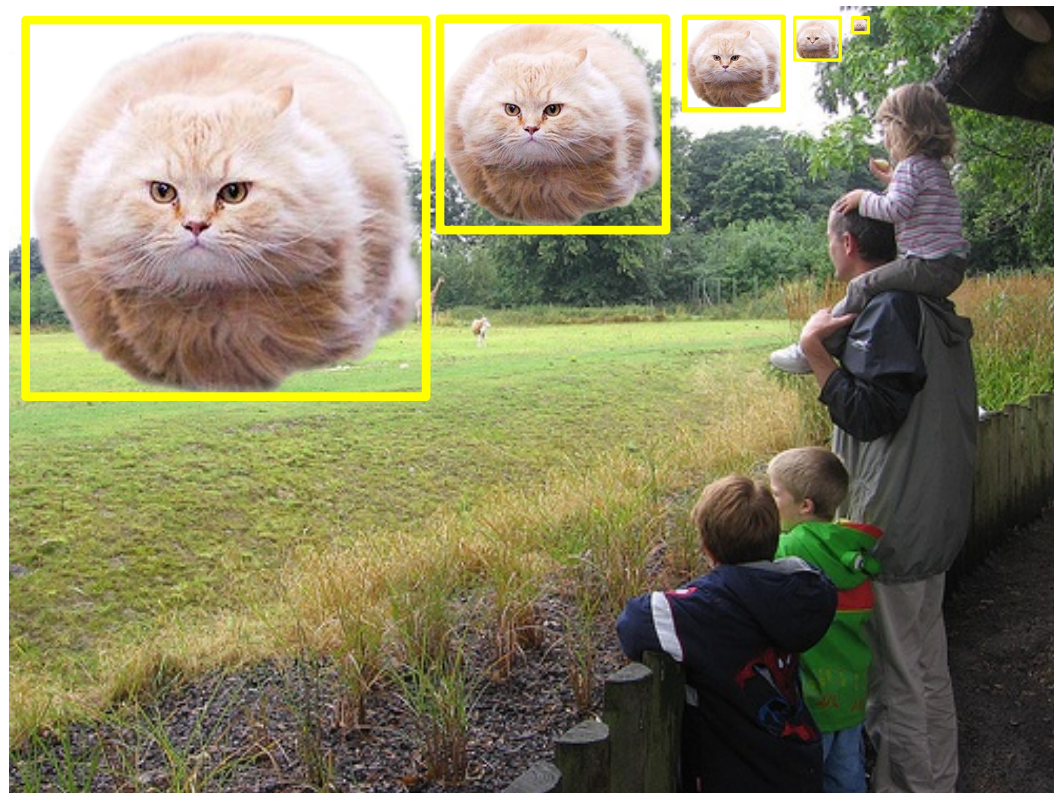
# Compromise Feature Quality, but Fast (“Free”)

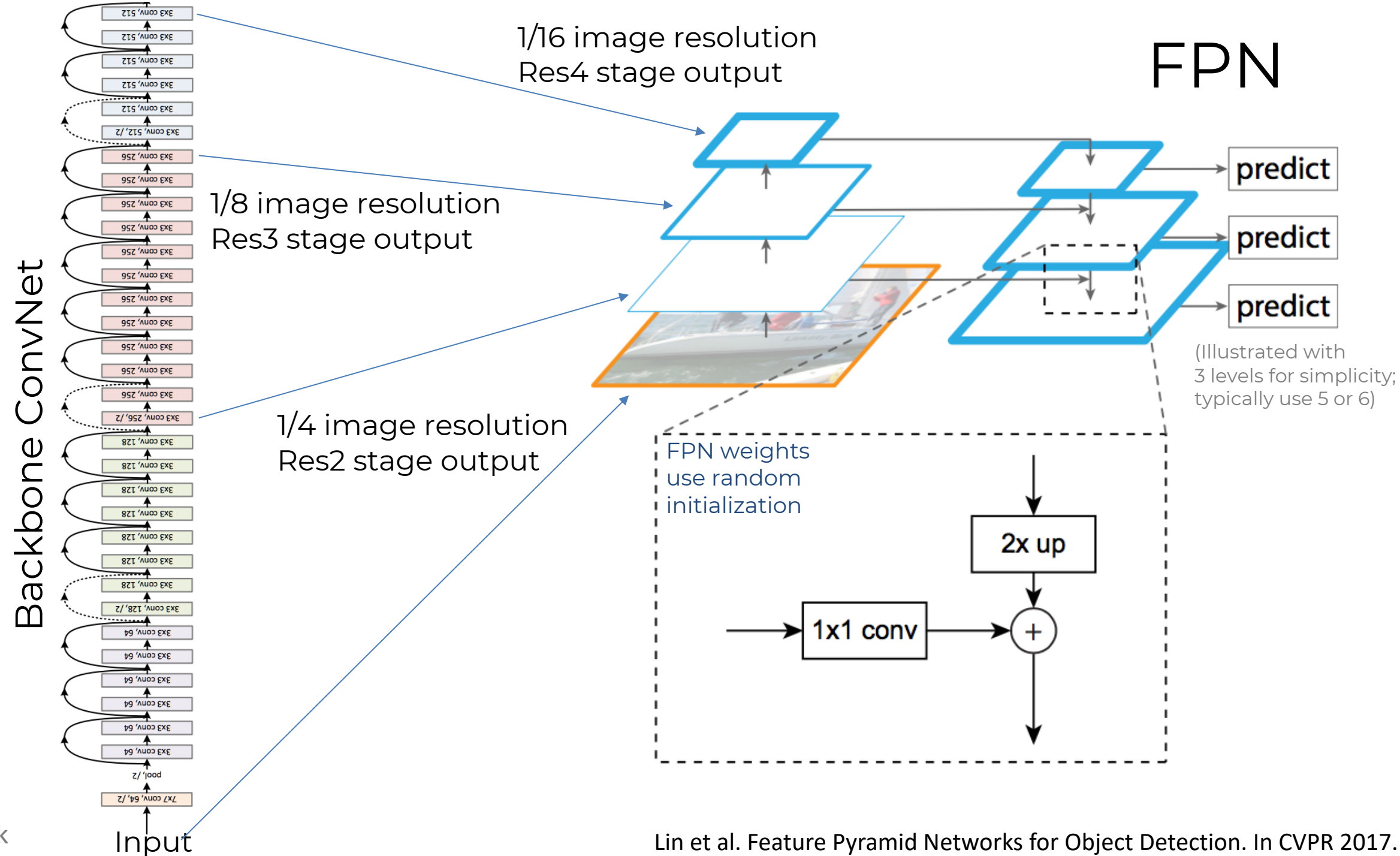


The “native” in-network feature pyramid poses an inherent tradeoff

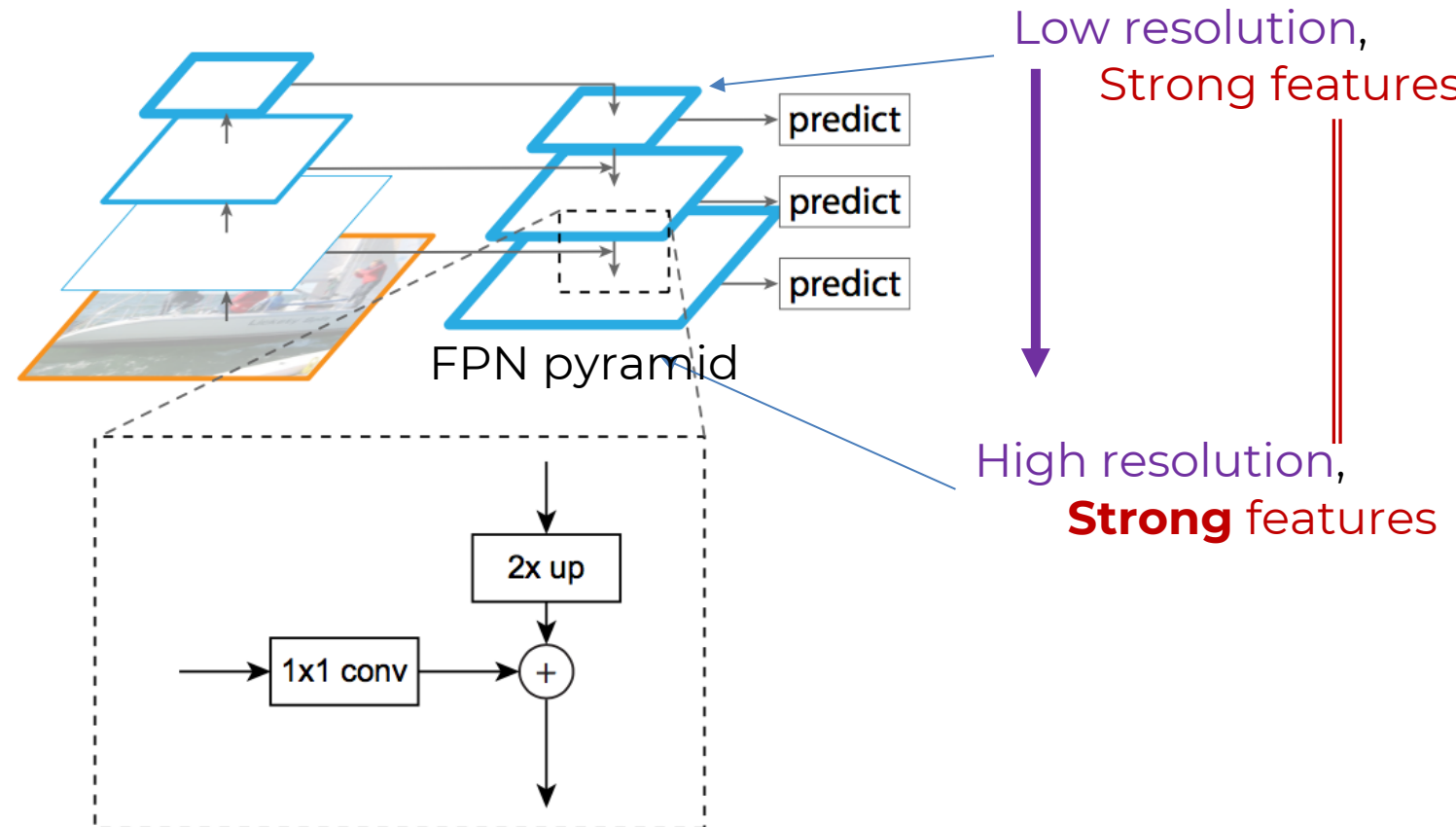
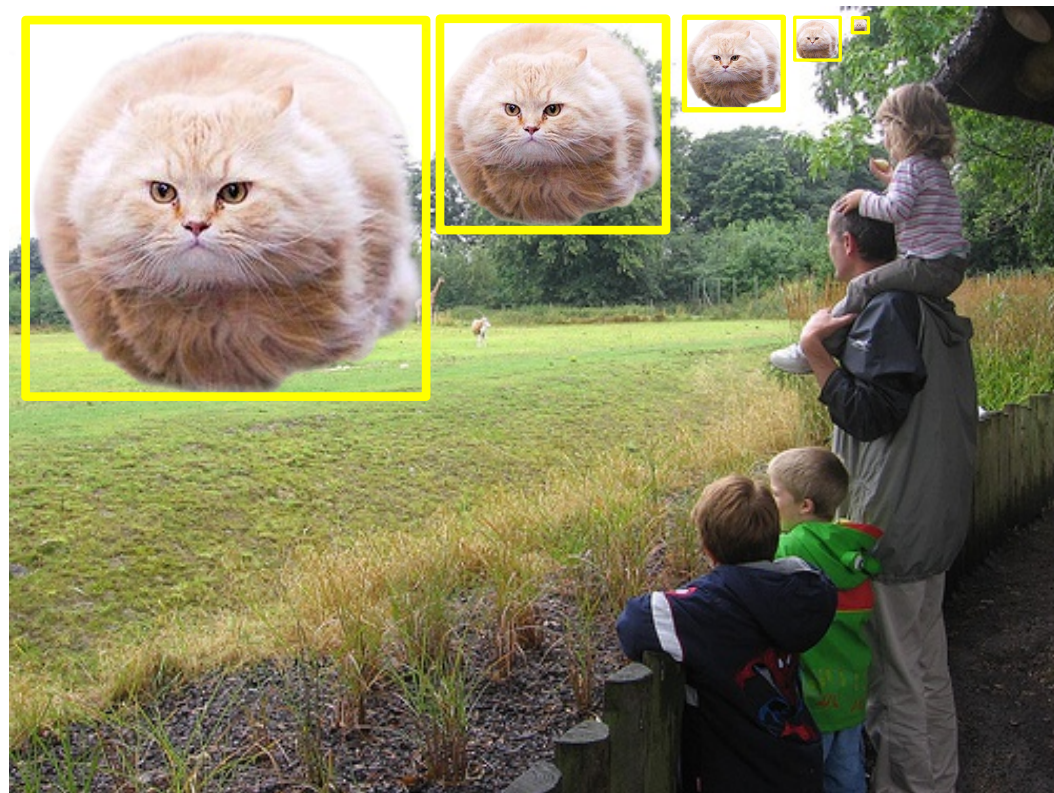
# 2. ... + Feature Pyramid Network (FPN)

[Optional, but recommended]



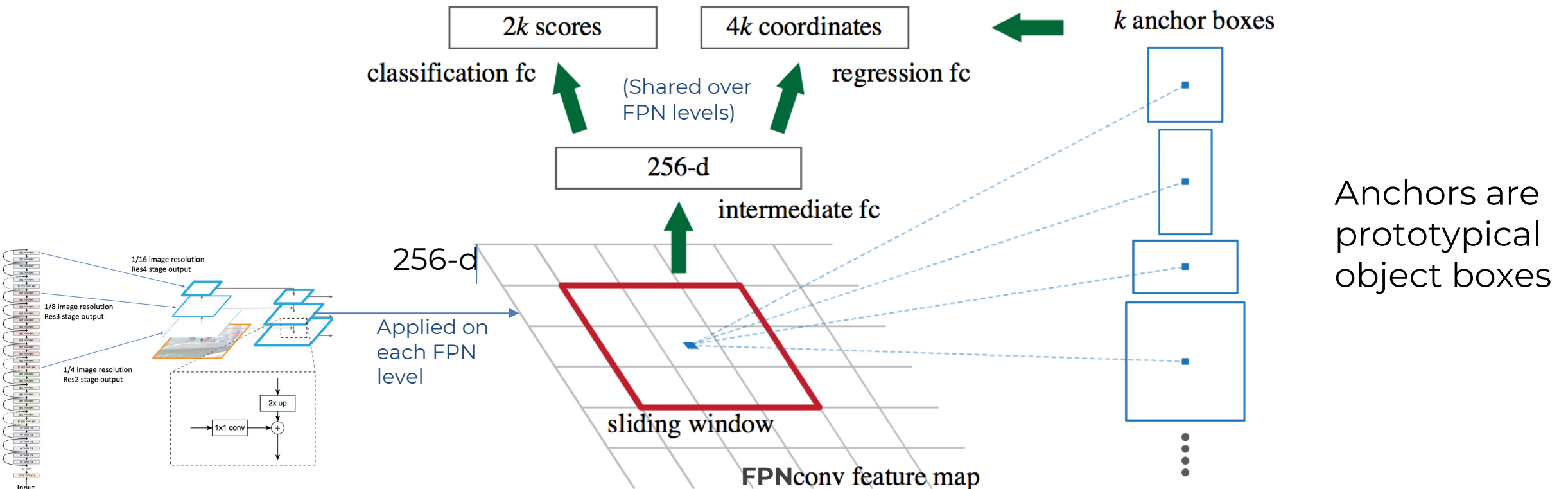


# No Compromise on Feature Quality, still Fast



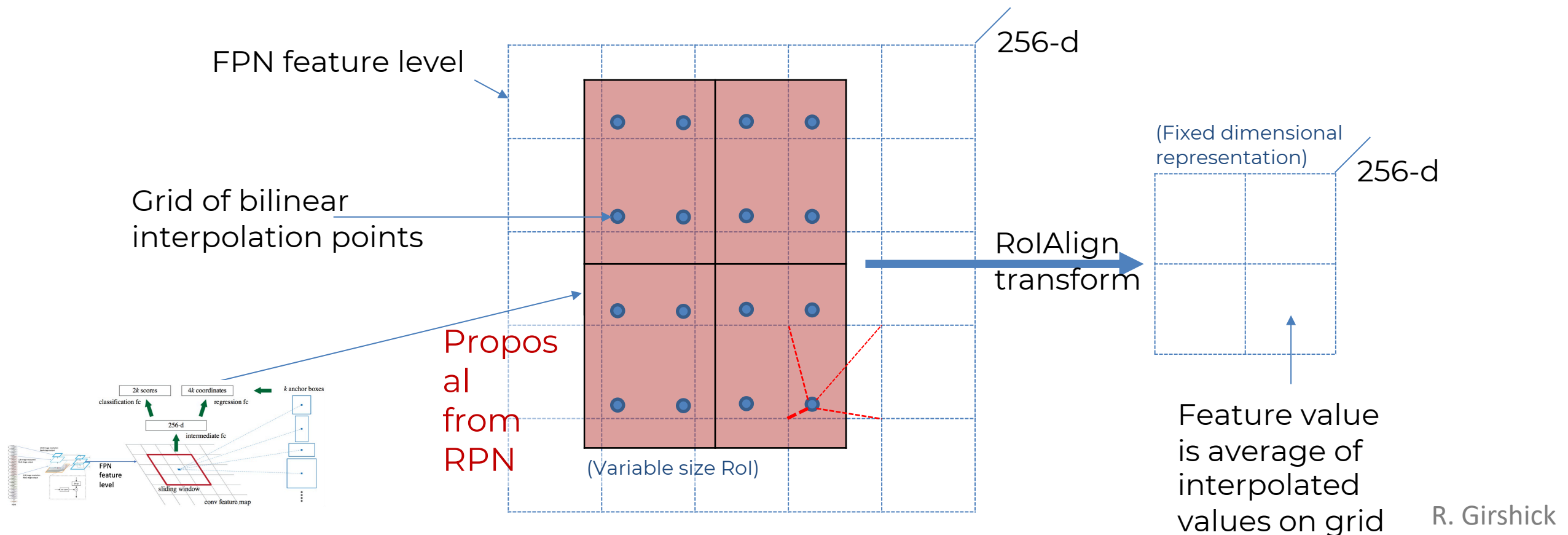
# 3. ... + Region Proposal Network (RPN)

Proposals = sliding window object/not-object classifier  
+ box regression *inside the same network*



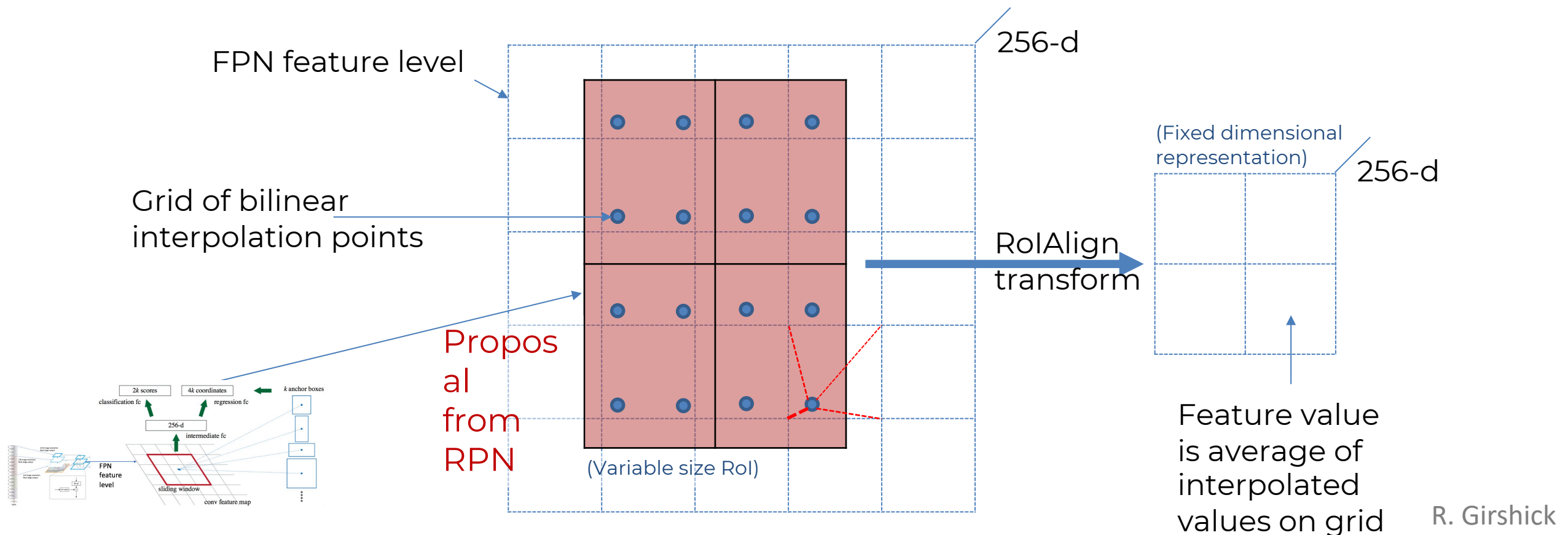
# 4. ... + RoIAlign Transform (on each Proposal)

Smoothly normalize features and predictions into coordinate frame free of scale and aspect ratio



# 4. ... + RoIAlign Transform (on each Proposal)

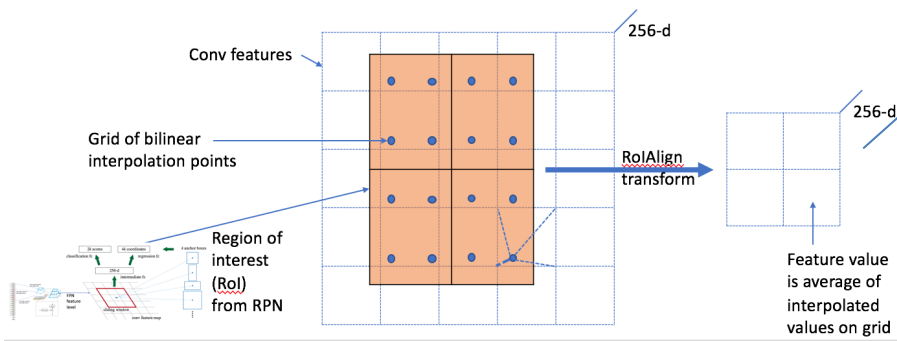
Key: **No coordinate quantization**  
(cf. RoIPool in Fast R-CNN, etc.)



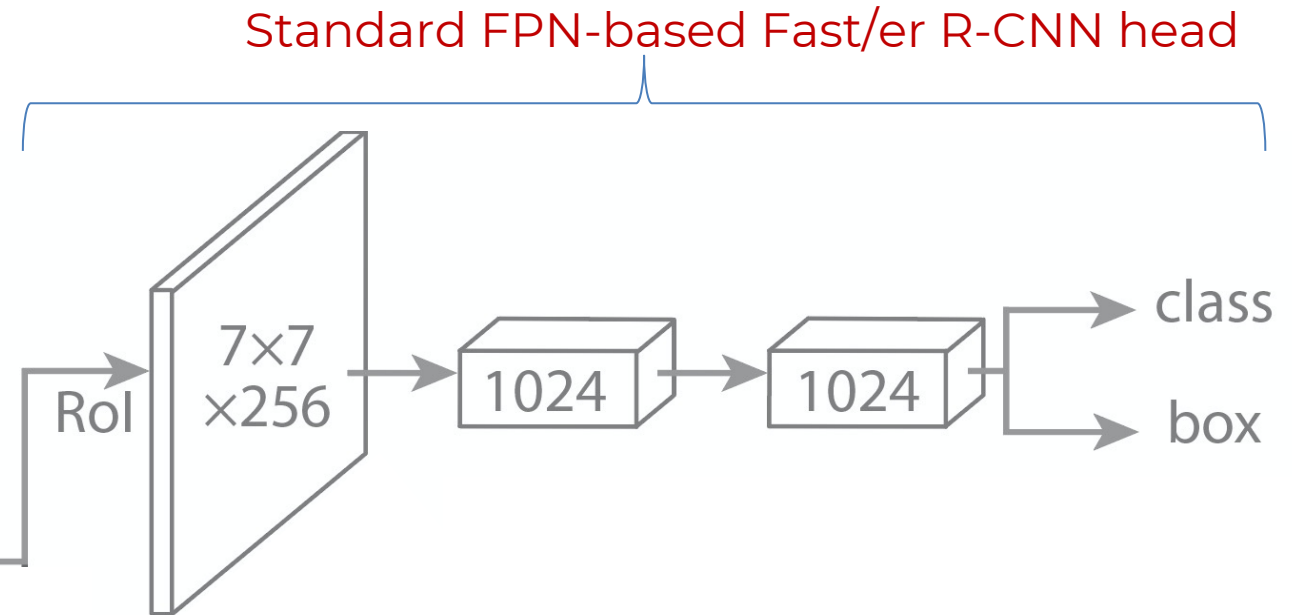
# 5. ... + Task-specific Heads (on each Proposal)

Task specific heads for ...

- Bounding box detection
- Object classification
- Instance mask prediction
- Human keypoint prediction



RoIAlign transformed features

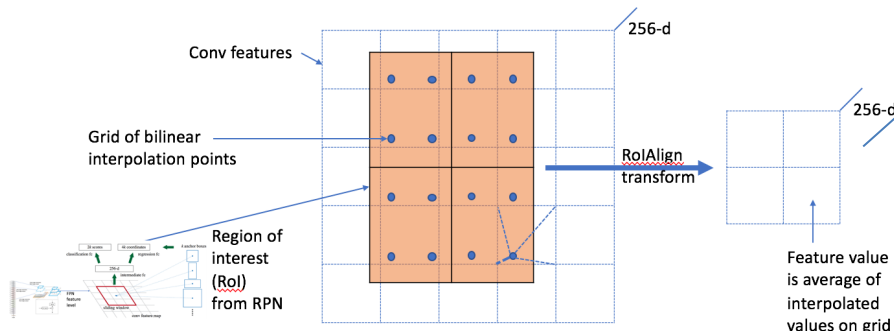




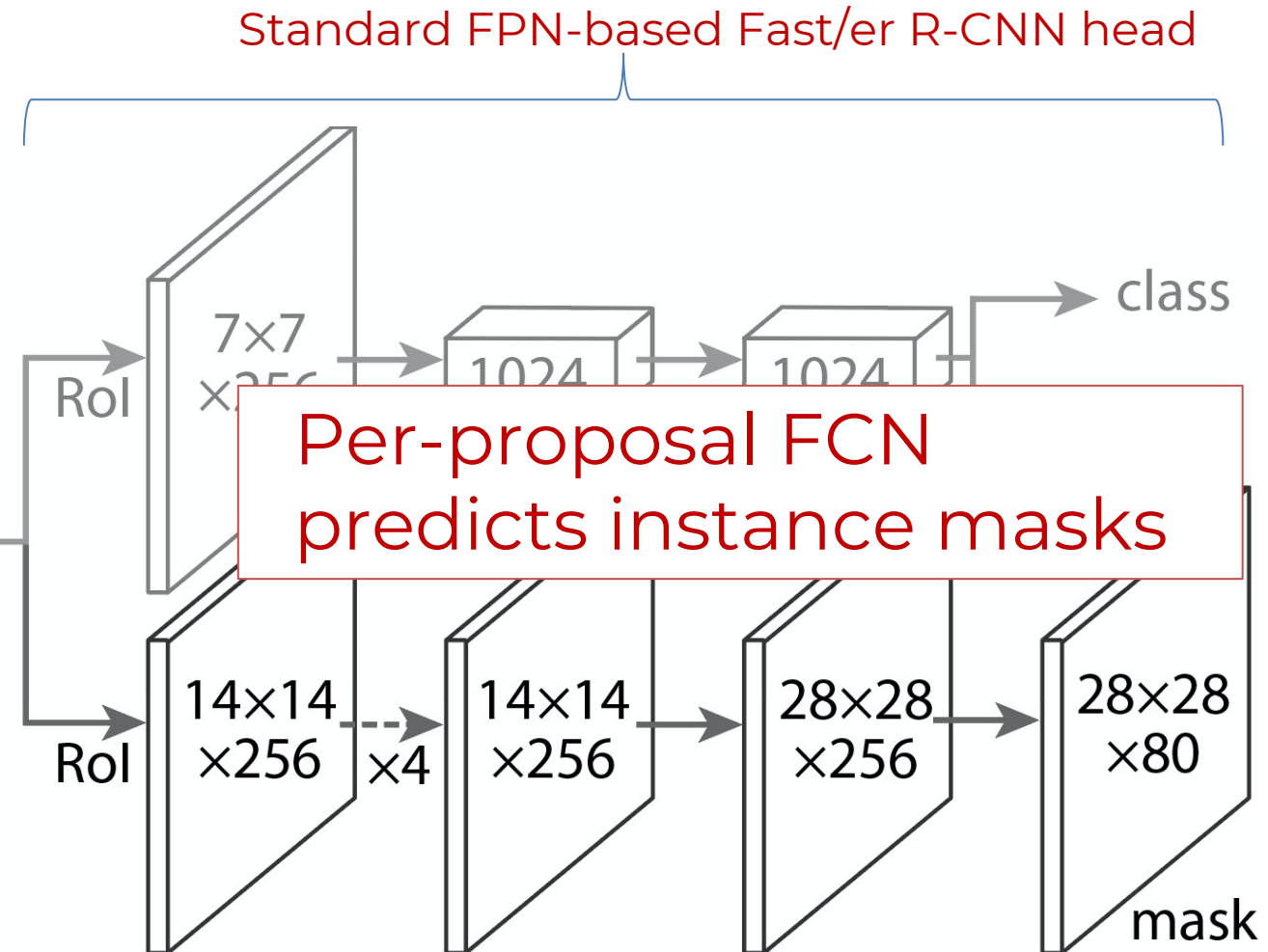
# 5. ... + Task-specific Heads (on each Proposal)

Task specific heads for ...

- Bounding box detection
- Object classification
- Instance mask prediction
- Human keypoint prediction



RoIAlign transformed features



# Mask R-CNN: Training

Same as “image centric” Fast/er R-CNN training

- Use precomputed proposals for faster experimentation
- Use joint / end-to-end training for sharing features

But with **training targets for masks**

# Example Mask Training Targets

Image with training proposal



28x28 mask target

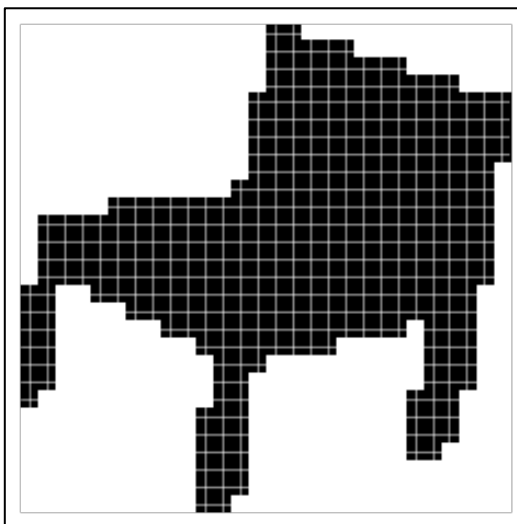
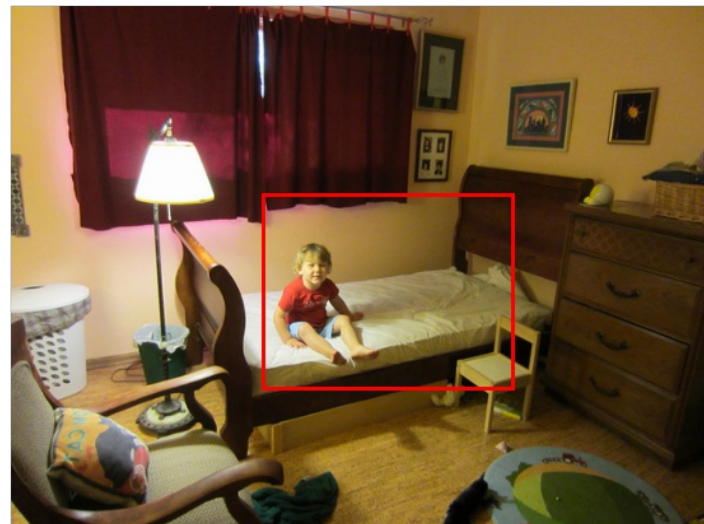
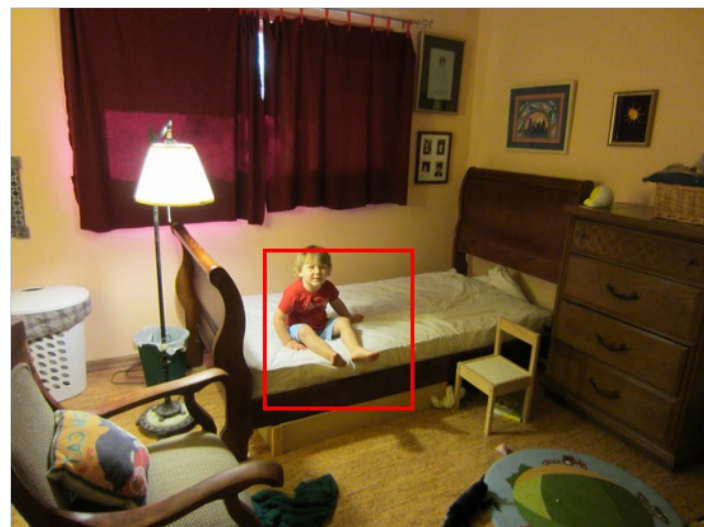
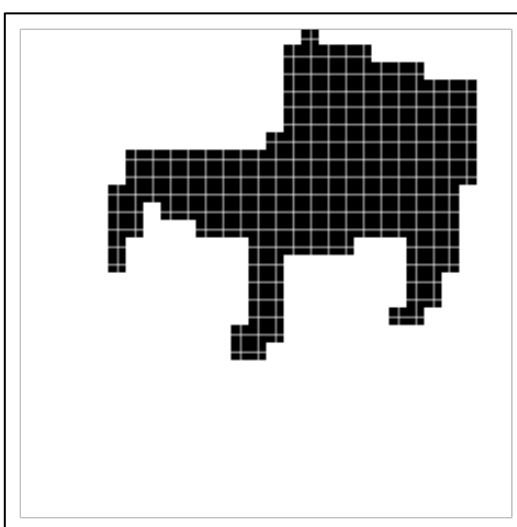
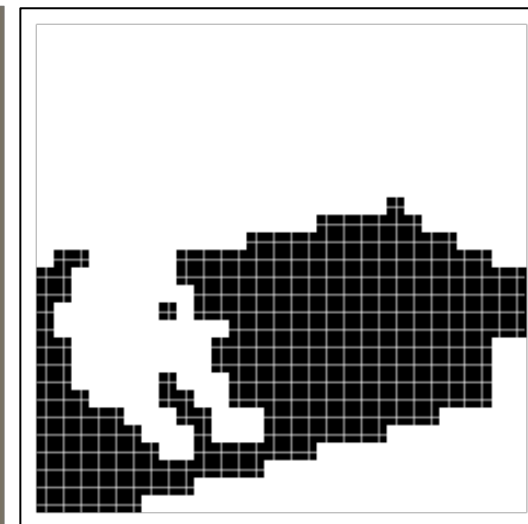


Image with training proposal



28x28 mask target



# Mask R-CNN: Inference

## 1. Perform Faster R-CNN inference

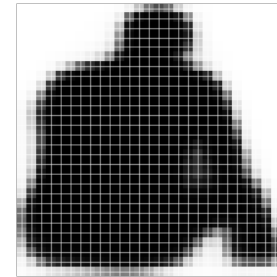
- Generate proposals (RPN)
- Score the proposals
- Regress from proposals to refined detection boxes
- Apply NMS and take the top  $K$  (= 100, e.g.)

## 2. Run RoIAlign and mask head on top- $K$ refined, post-NMS boxes

- Fast (only compute masks for top- $K$  detections)
- Improves accuracy (uses *refined* detection boxes, not proposals)

# Mask Prediction

28x28 soft prediction from Mask R-CNN  
(enlarged)



Soft prediction **resampled to image coordinates**  
(bilinear and bicubic interpolation work equally well)



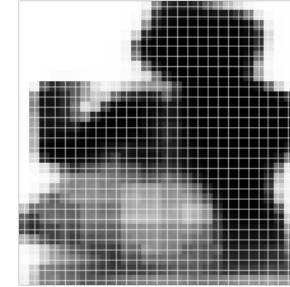
Final prediction (threshold at 0.5)



Validation image with box detection shown in red

# Mask Prediction

28x28 soft prediction



Resized soft prediction



Final mask

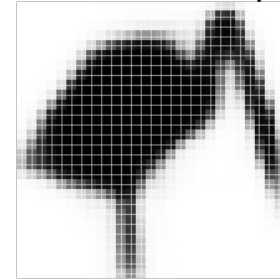


Validation image with box detection shown in red

# Mask Prediction



28x28 soft prediction



Resized Soft prediction



Final mask

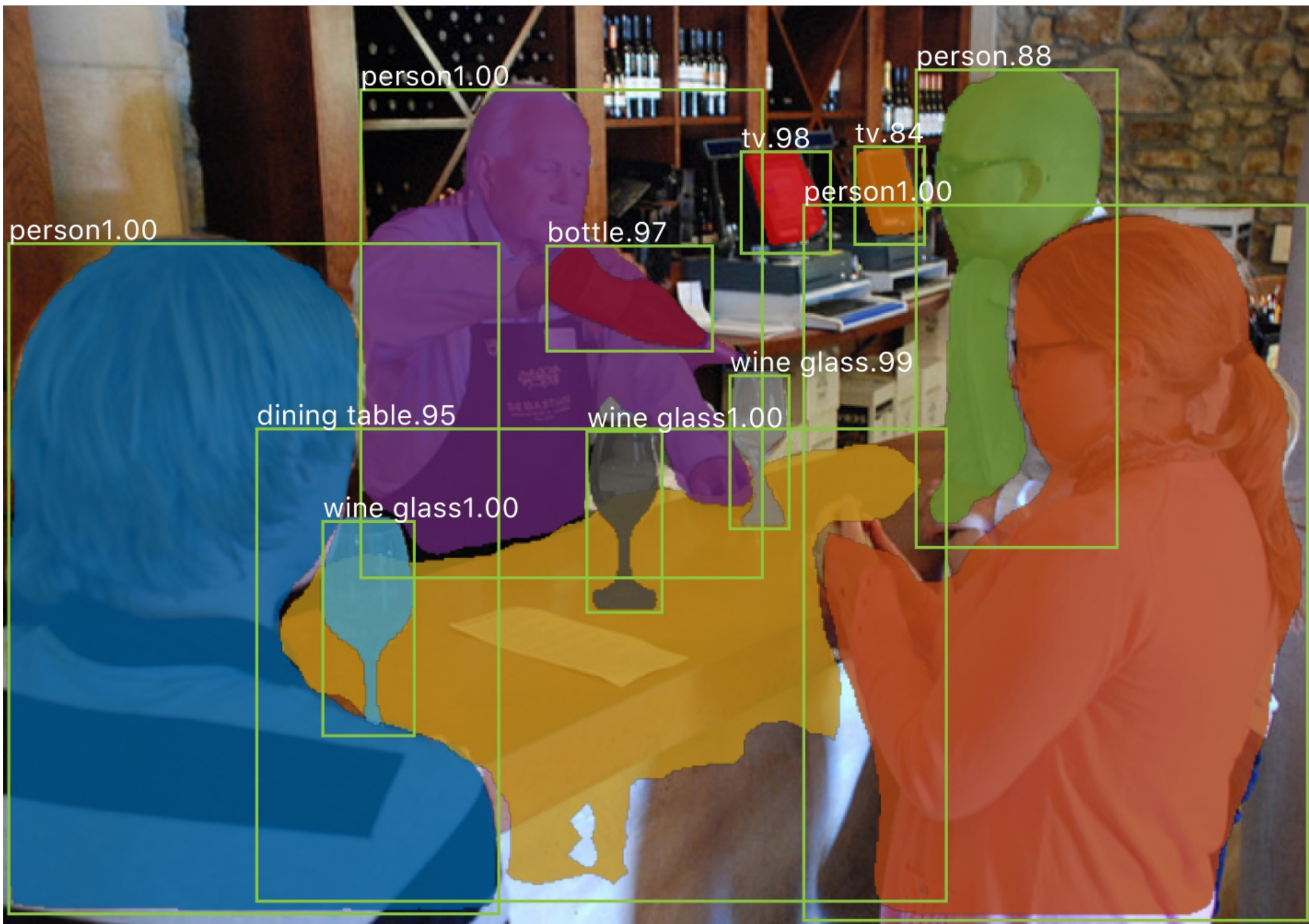


Validation image with box detection shown in red



person1.00 person1.00 person1.00 person.98  
person1.00 person.91 surfboard1.00 surfboard.98 surfboard1.00 surfboard1.00  
surfboard1.00 person.74





person1.00

person.88

tv.98

tv.84

person1.00

person1.00

bottle.97

wine glass.99

dining table.95

wine glass1.00

wine glass1.00

# Segmentation Overview

- Semantic segmentation
  - Classify all pixels
  - Fully convolutional models, downsample then upsample
  - Learnable upsampling: fractionally strided convolution
  - Skip connections can help
- Instance Segmentation
  - Detect instance, generate mask
  - Similar pipelines to object detection