Lecture 12:
Object Detection and Image Segmentation

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Announcements

- Homework 2 due **Thursday, March 26**
- Project proposals due **Tuesday, March 31** —
  - Carefully read project proposal guidelines on the course page
  - Submit one proposal per team

Image Classification: A core task in Computer Vision

This image by Nikita is licensed under CC-BY 2.0

(assume given a set of possible labels)
{dog, cat, truck, plane, ...}

$\longrightarrow$ cat

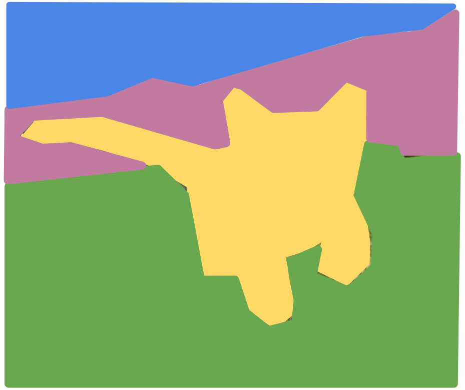
Computer Vision Tasks

Classification

CAT

No spatial extent

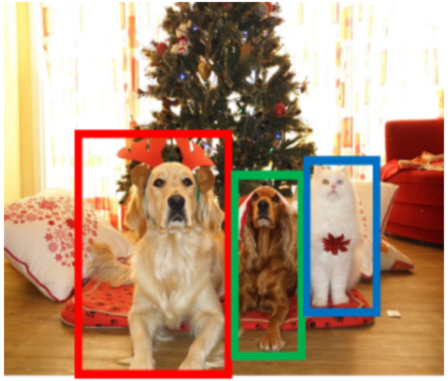
Semantic Segmentation



GRASS, CAT, TREE, SKY

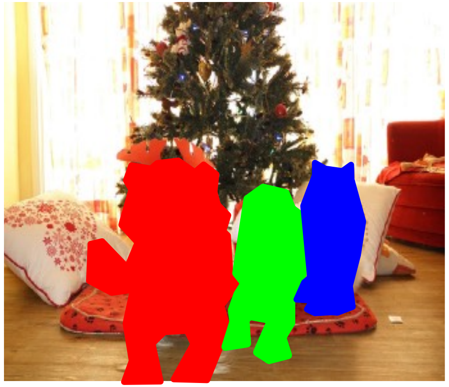
No objects, just pixels

Object Detection



DOG, DOG, CAT

Instance Segmentation



DOG, DOG, CAT

Multiple Object

This image is CC0 public domain

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

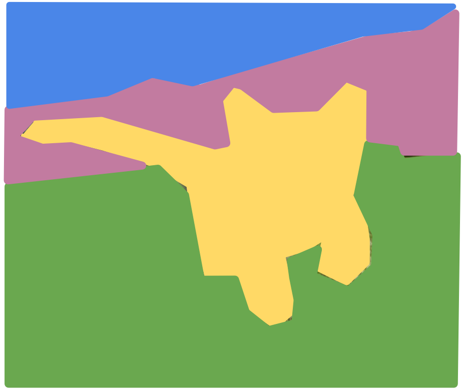
Semantic Segmentation

Classification

CAT

No spatial extent

Semantic Segmentation



GRASS, CAT, TREE, SKY

No objects, just pixels

Object Detection

DOG, DOG, CAT

Instance Segmentation

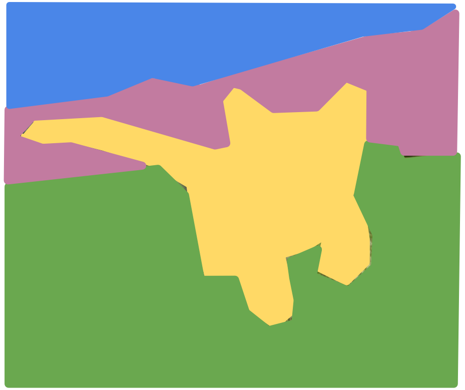
DOG, DOG, CAT

Multiple Object

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Historic note: Why is it called Semantic Segmentation?

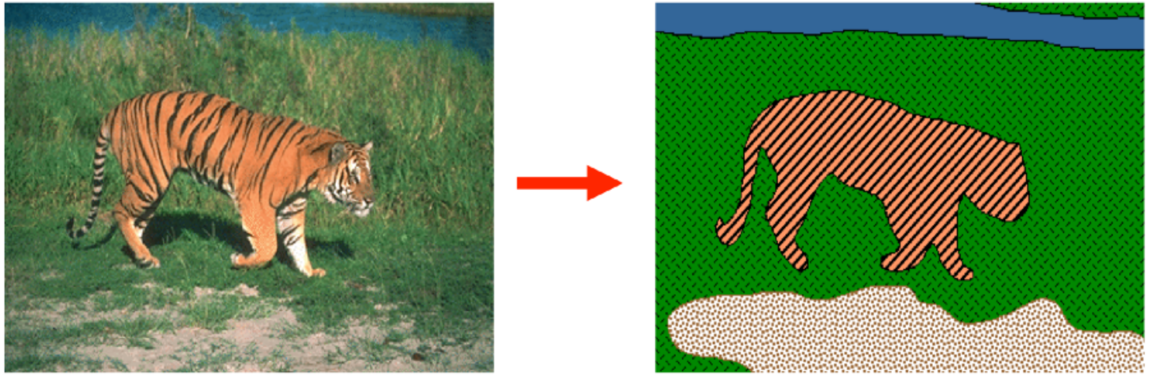
Semantic Segmentation



GRASS, CAT, TREE, SKY

No objects, just pixels

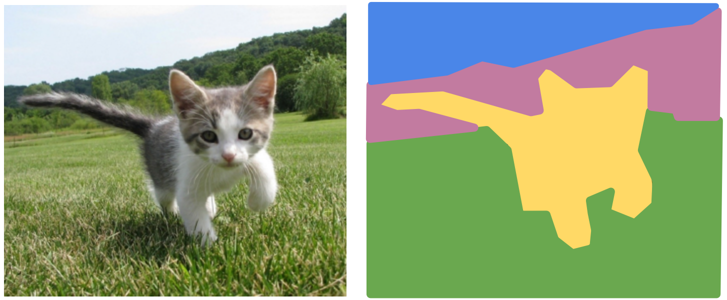
Segmentation



Traditionally covers edge detection, grouping, etc., often done in a category-agnostic manner

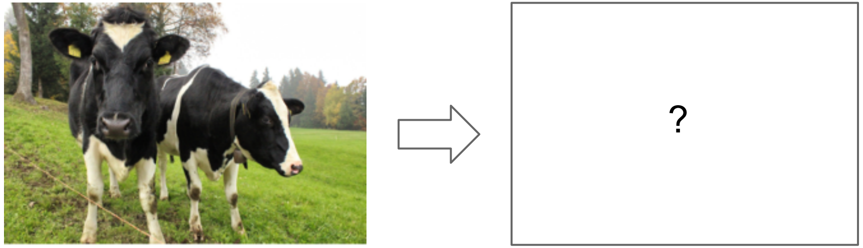
Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Semantic Segmentation: The Problem



GRASS, **CAT**, **TREE**, **SKY**, **...**

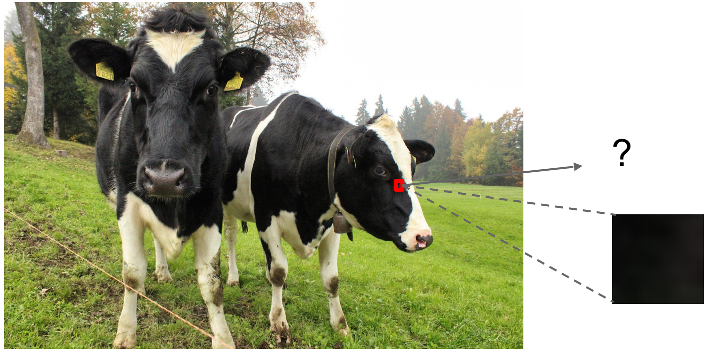
Paired training data: for each training image, each pixel is labeled with a semantic category.



At test time, classify each pixel of a new image.

Semantic Segmentation Idea: Sliding Window

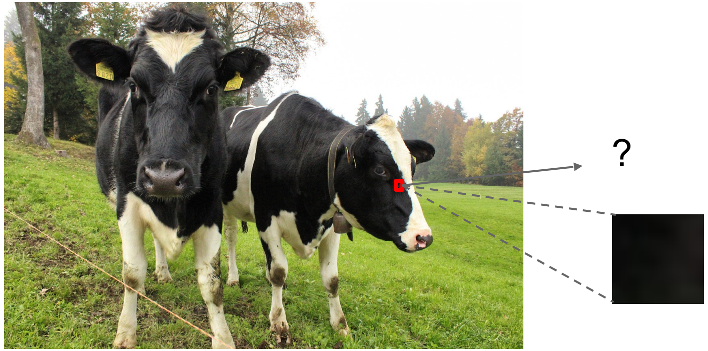
Full image



Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Semantic Segmentation Idea: Sliding Window

Full image

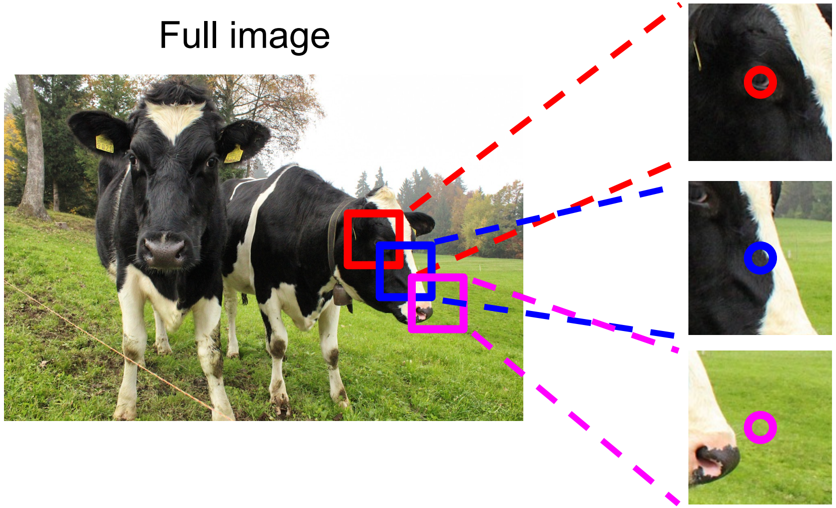


Impossible to classify without context

Q: how do we include context?

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Semantic Segmentation Idea: Sliding Window



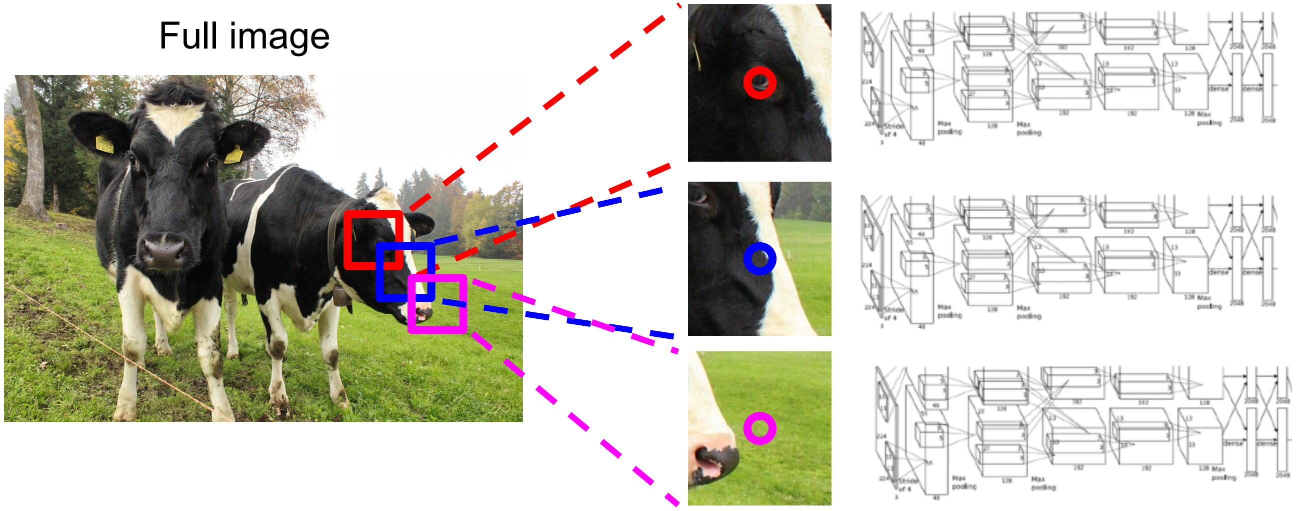
Q: how do we model this?

Semantic Segmentation Idea: Sliding Window

Full image

Extract patch

Classify center pixel with CNN



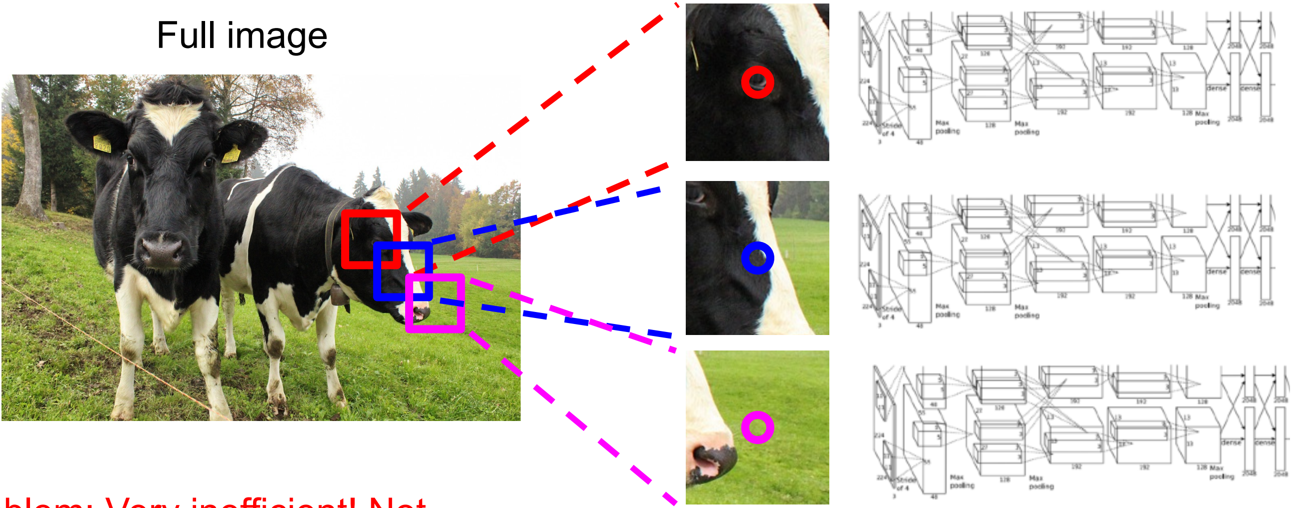
Cow

Cow

Grass

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

Semantic Segmentation Idea: Sliding Window



Full image

Extract patch

Classify center pixel with CNN

Cow

Cow

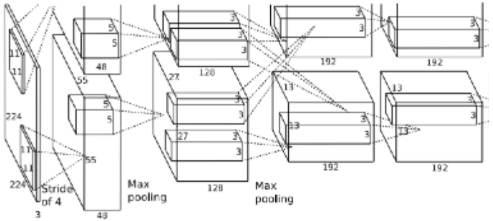
Grass

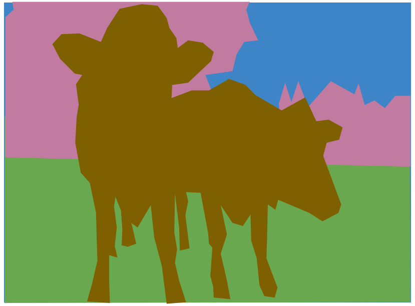
Problem: Very inefficient! Not reusing shared features between overlapping patches

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

Semantic Segmentation Idea: Convolution

Full image





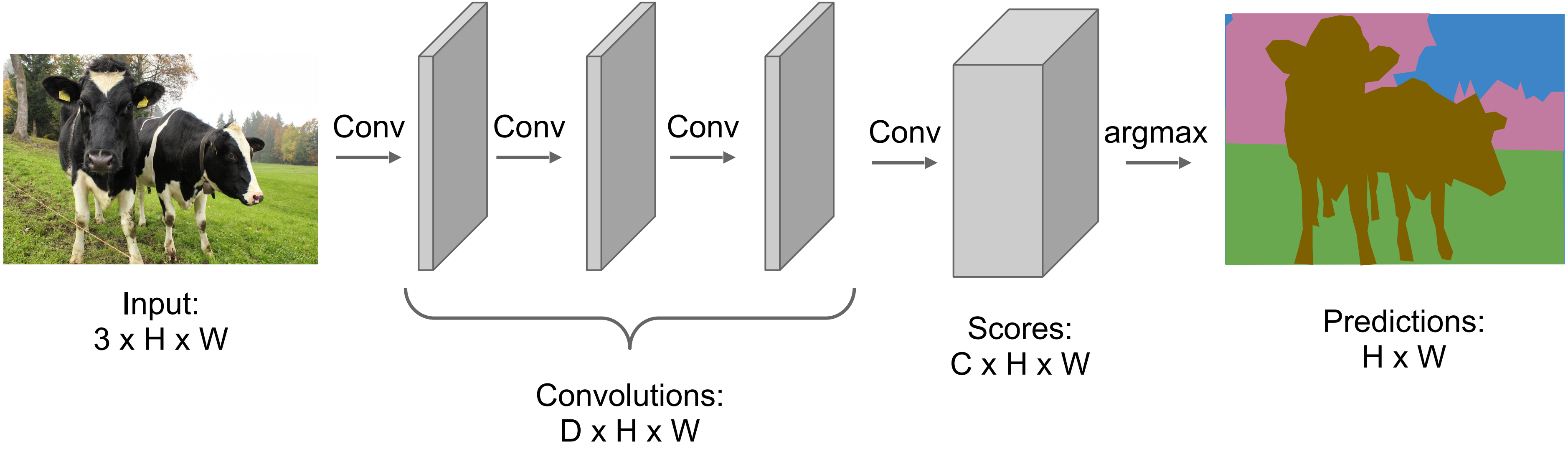
An intuitive idea: encode the entire image with conv net, and do semantic segmentation on top.

Problem: classification architectures often reduce feature spatial sizes to go deeper, but semantic segmentation requires the output size to be the same as input size.

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Semantic Segmentation Idea: Fully Convolutional

Design a network with only convolutional layers without downsampling operators to make predictions for pixels all at once!



Input:
3 x H x W

Convolutions:
D x H x W

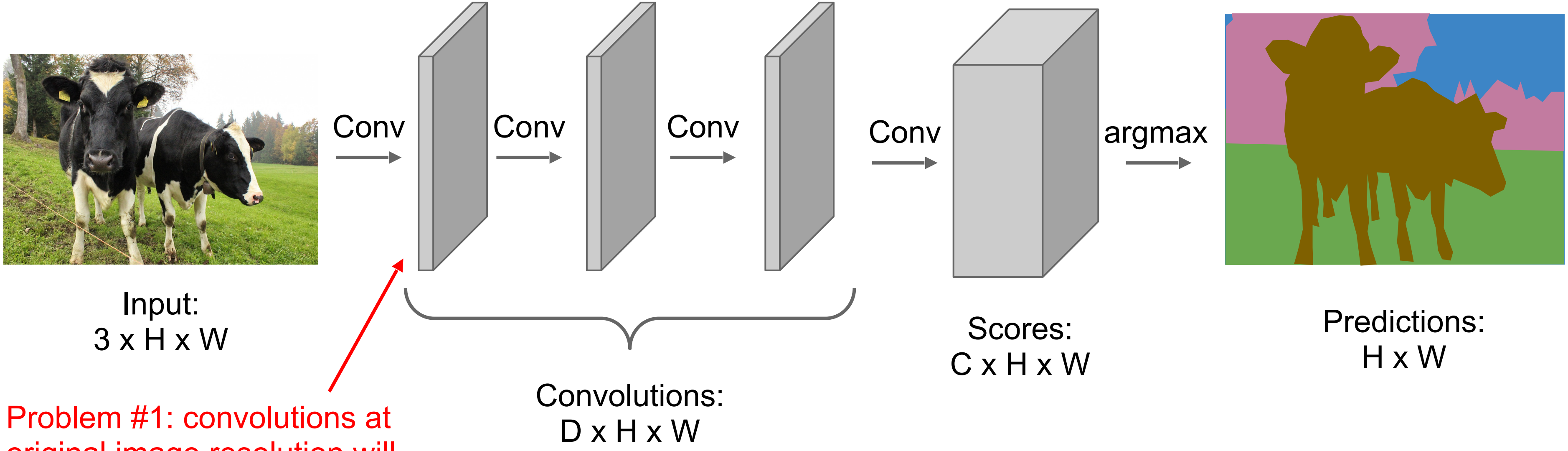
Scores:
C x H x W

Predictions:
H x W

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Semantic Segmentation Idea: Fully Convolutional

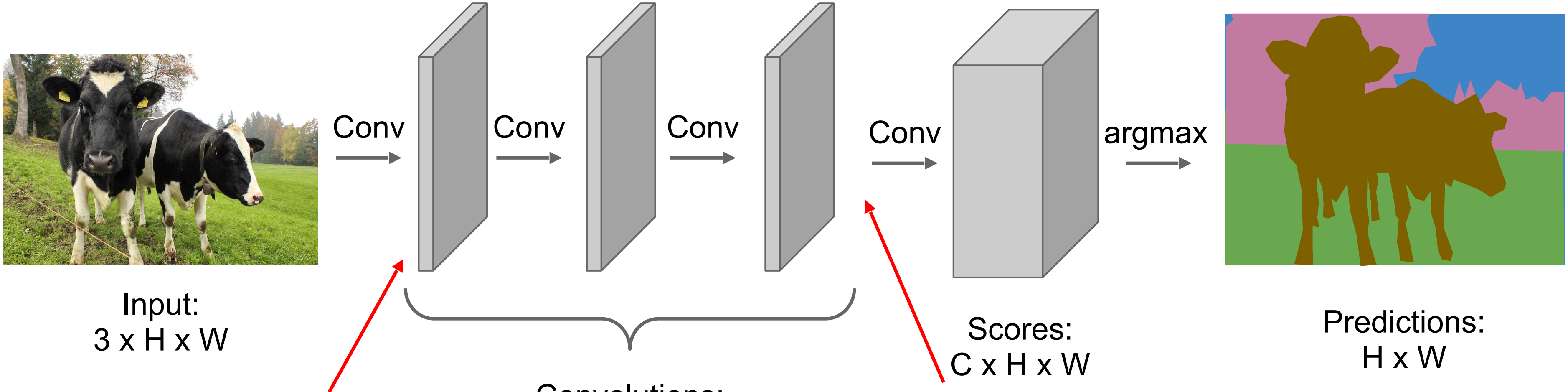
Design a network with only convolutional layers without downsampling operators to make predictions for pixels all at once!



Problem #1: convolutions at original image resolution will be very expensive ...

Semantic Segmentation Idea: Fully Convolutional

Design a network with only convolutional layers without downsampling operators to make predictions for pixels all at once!



Input:
3 x H x W

Convolutions:
D x H x W

Scores:
C x H x W

Predictions:
H x W

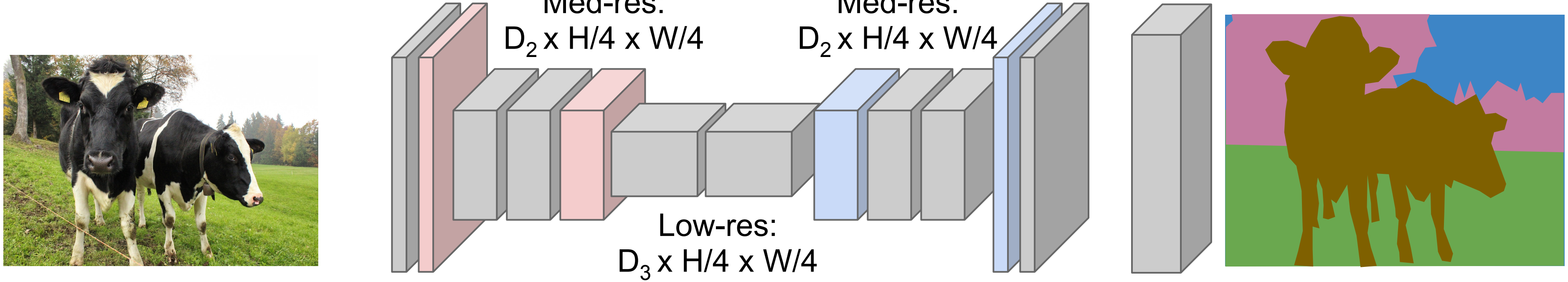
Problem #1: convolutions at original image resolution will be very expensive ...

Problem #2: Effective receptive field size is linear in number of conv layers: With L 3x3 conv layers, receptive field is 1+2L

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Semantic Segmentation Idea: Fully Convolutional

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



Med-res: D_2 x H/4 x W/4

Med-res: D_2 x H/4 x W/4

Low-res: D_3 x H/4 x W/4

Input: 3 x H x W

High-res: D_1 x H/2 x W/2

High-res: D_1 x H/2 x W/2

C x H x W

Predictions: H x W

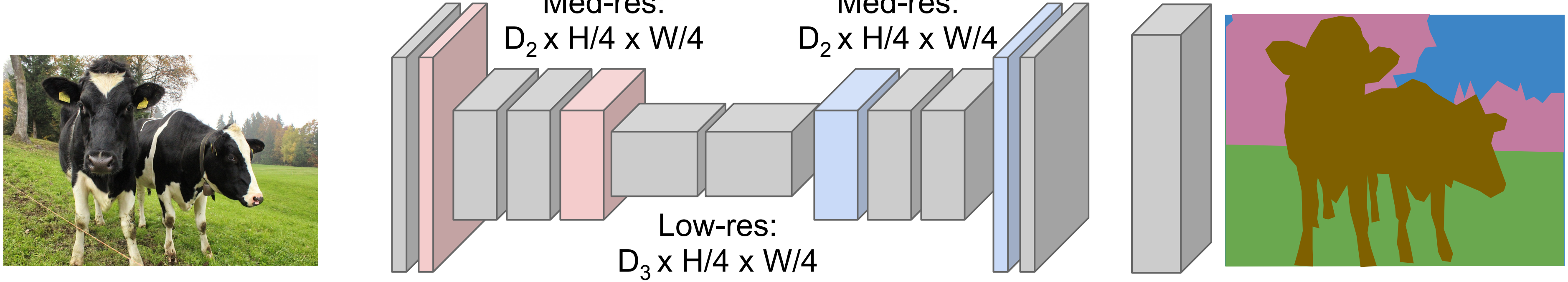
Long, Shelhamer, and Darrell, “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015
Noh et al, “Learning Deconvolution Network for Semantic Segmentation”, ICCV 2015

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: $D_1 \times H/2 \times W/2$

Med-res: $D_2 \times H/4 \times W/4$

Low-res: $D_3 \times H/4 \times W/4$

Med-res: $D_2 \times H/4 \times W/4$

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

$C \times H \times W$

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

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

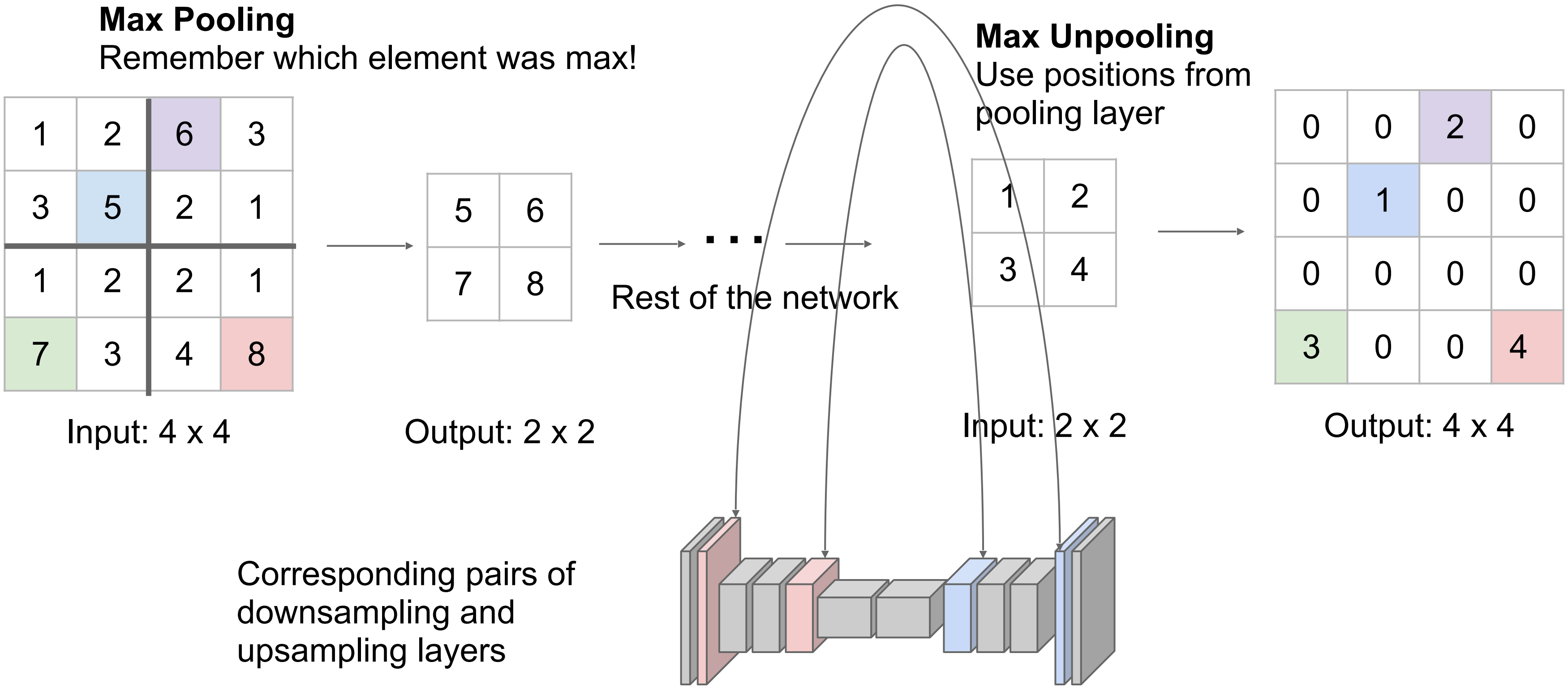
→

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

Output: 4 x 4

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

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

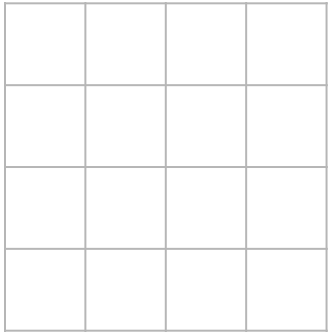
0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

Output: 4 x 4

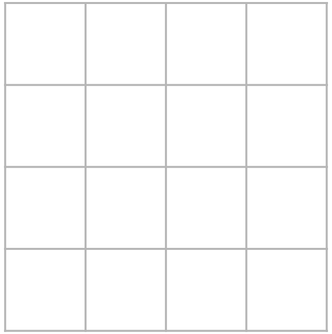
Corresponding pairs of downsampling and upsampling layers

Learnable Upsampling

Recall: Normal 3 x 3 convolution, stride 1 pad 1



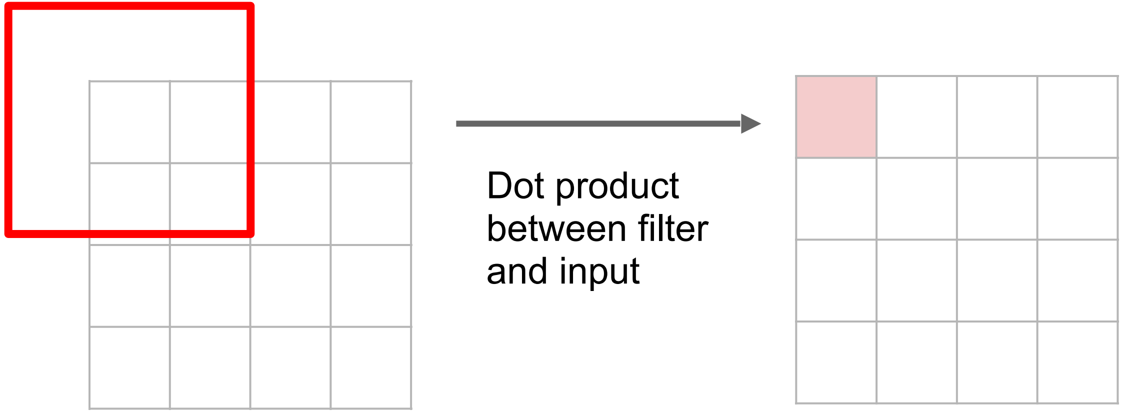
Input: 4 x 4



Output: 4 x 4

Learnable Upsampling

Recall: Normal 3 x 3 convolution, stride 1 pad 1



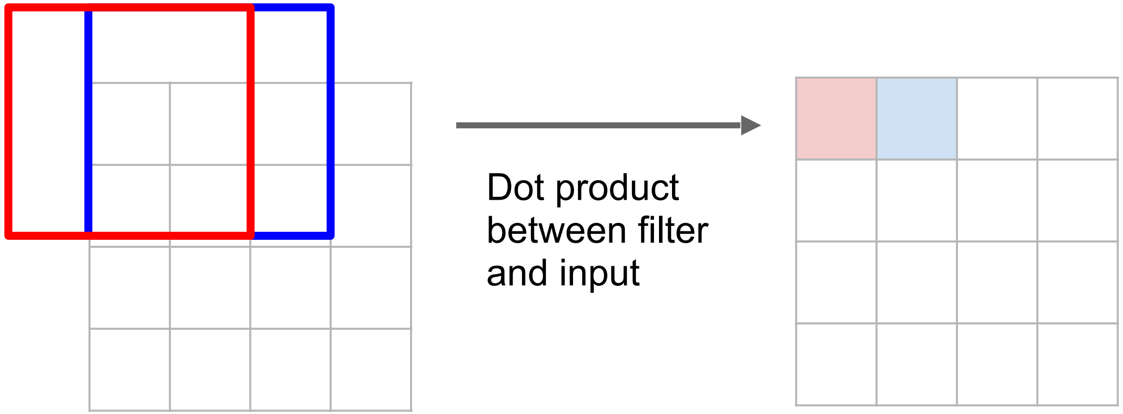
Dot product between filter and input

Input: 4 x 4

Output: 4 x 4

Learnable Upsampling

Recall: Normal 3 x 3 convolution, stride 1 pad 1



Dot product between filter and input

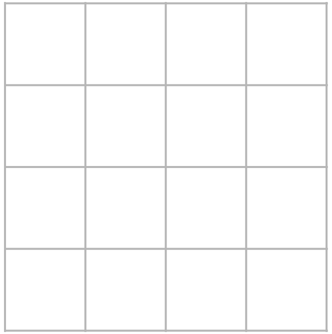
Input: 4 x 4

Output: 4 x 4

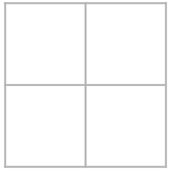
Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Learnable Upsampling

Recall: Normal 3 x 3 convolution, stride 2 pad 1



Input: 4 x 4

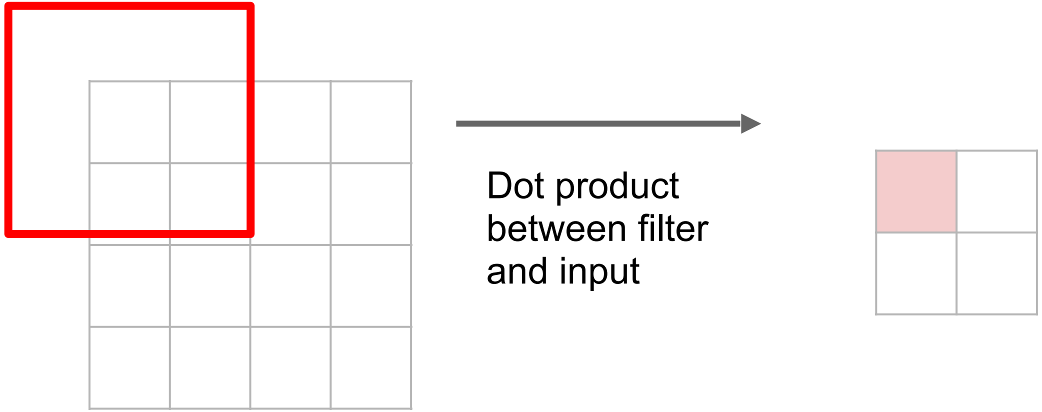


Output: 2 x 2

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Learnable Upsampling

Recall: Normal 3 x 3 convolution, stride 2 pad 1



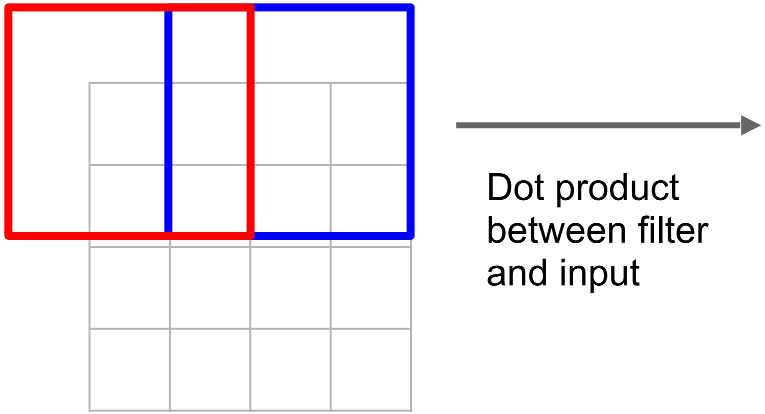
Dot product between filter and input

Input: 4 x 4

Output: 2 x 2

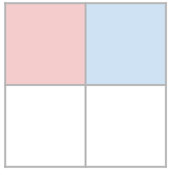
Learnable Upsampling

Recall: Normal 3 x 3 convolution, stride 2 pad 1



Input: 4 x 4

Dot product between filter and input



Output: 2 x 2

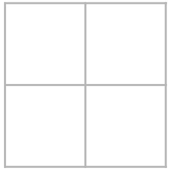
Filter moves 2 pixels in the input for every one pixel in the output

Stride gives ratio between movement in input and output

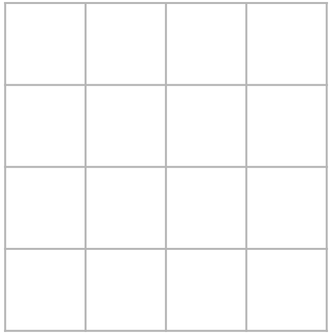
We can interpret strided convolution as “learnable downsampling”.

Learnable Upsampling: Transposed Convolution

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



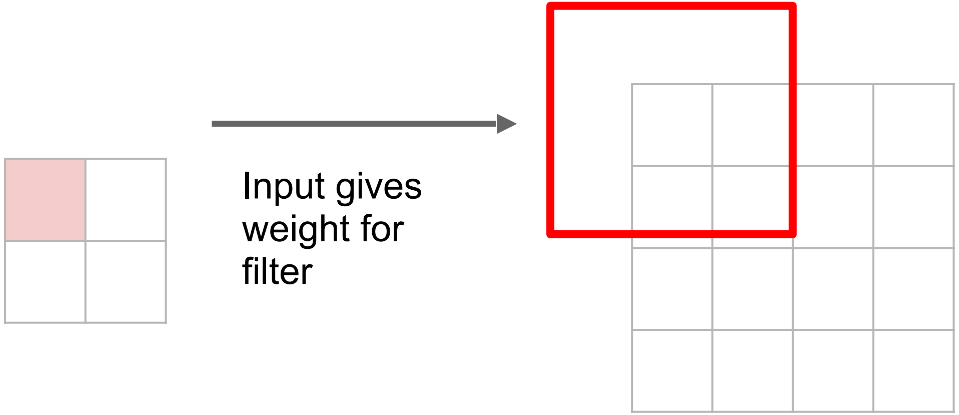
Input: 2 x 2



Output: 4 x 4

Learnable Upsampling: Transposed Convolution

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



Input gives weight for filter

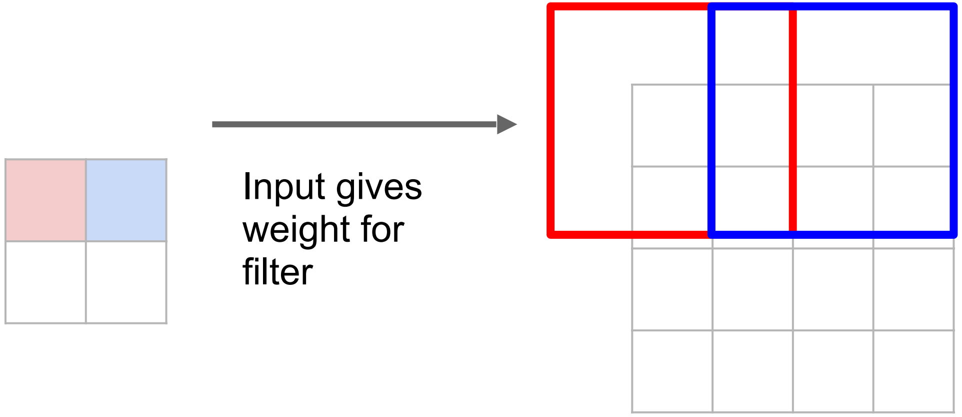
Input: 2 x 2

Output: 4 x 4

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Learnable Upsampling: Transposed Convolution

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



Input gives weight for filter

Input: 2 x 2

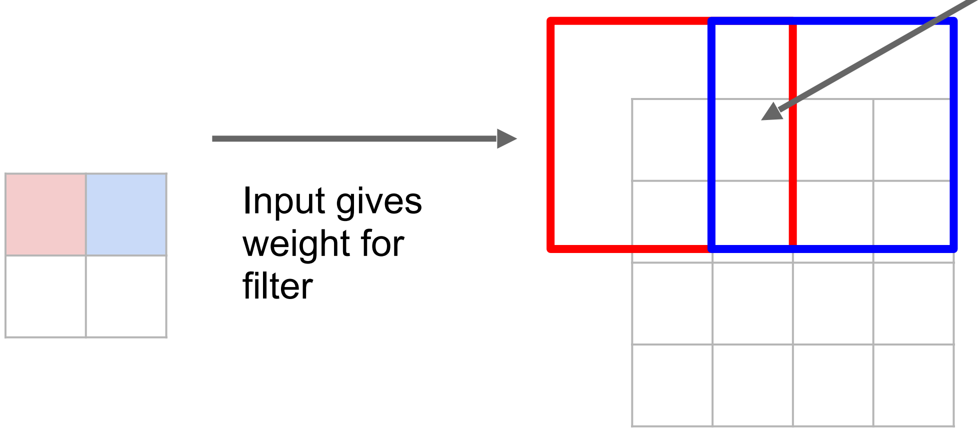
Output: 4 x 4

Filter moves 2 pixels in the <u>output</u> for every one pixel in the <u>input</u>

Stride gives ratio between movement in output and input

Learnable Upsampling: Transposed Convolution

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



Sum where output overlaps

Input gives weight for filter

Filter moves 2 pixels in the <u>output</u> for every one pixel in the <u>input</u>

Stride gives ratio between movement in output and input

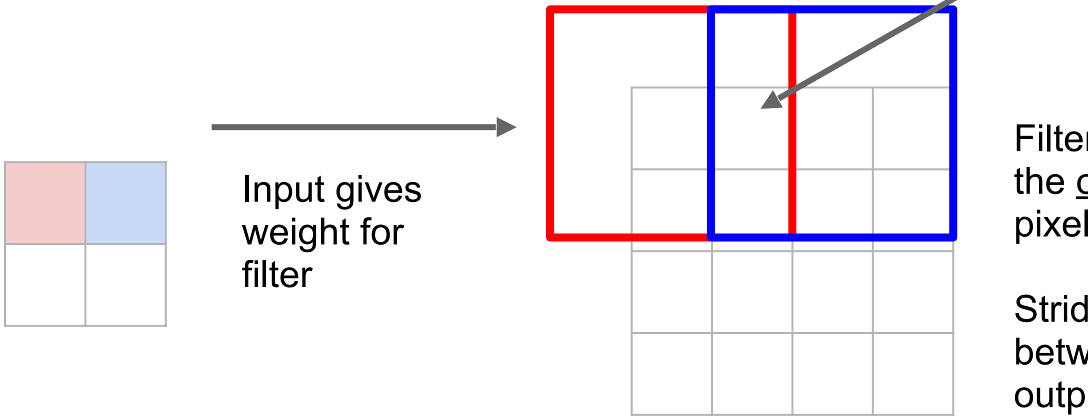
Input: 2 x 2

Output: 4 x 4

Learnable Upsampling: Transposed Convolution

Q: Why is it called transposed convolution?

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



Input gives weight for filter

Sum where output overlaps

Input: 2 x 2

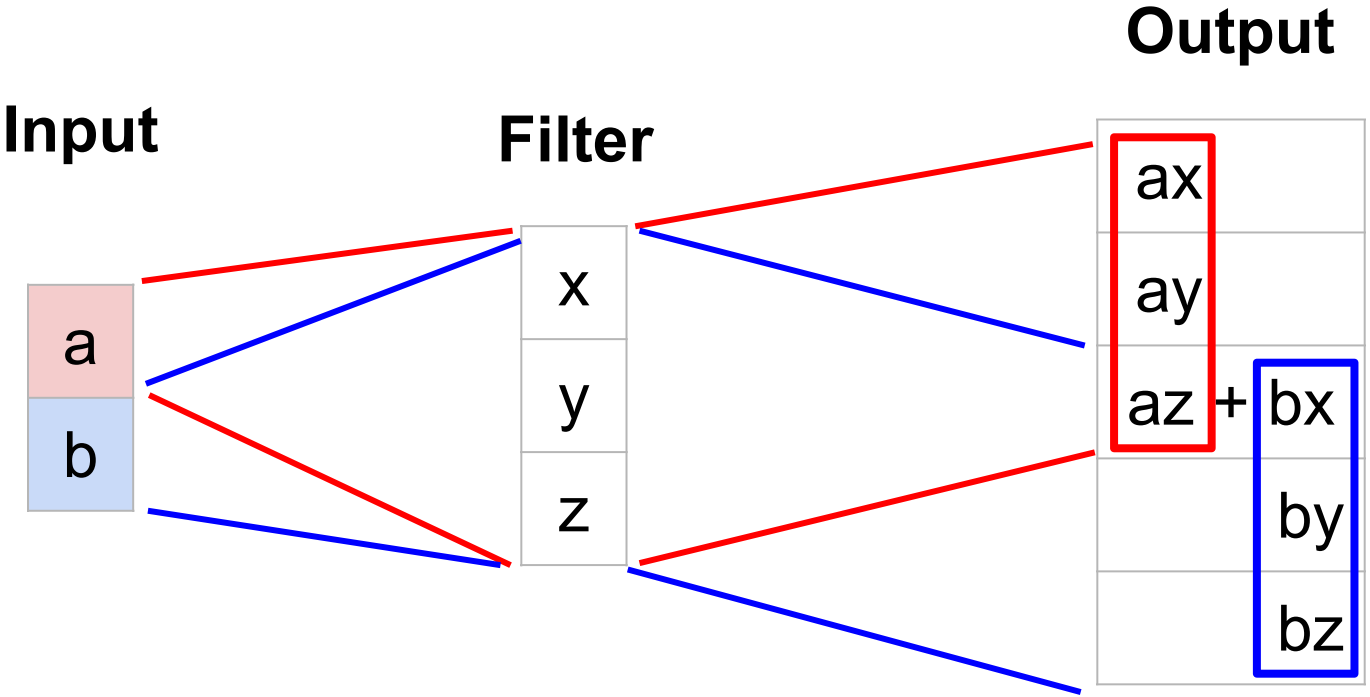
Output: 4 x 4

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

Stride gives ratio between movement in output and input

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Learnable Upsampling: 1D Example



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

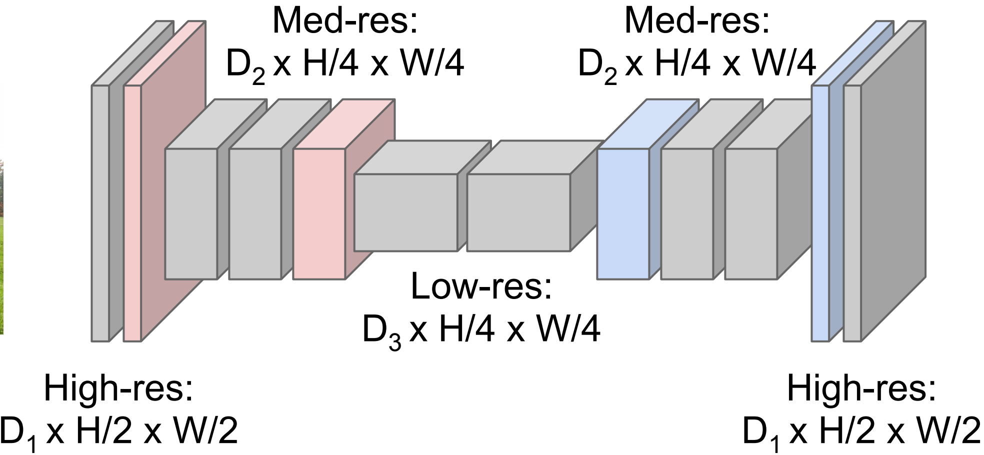
Semantic Segmentation Idea: Fully Convolutional

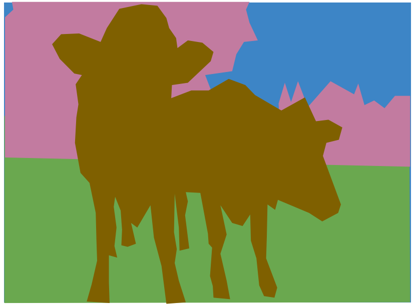
Downsampling: Pooling, strided convolution

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

Upsampling: Unpooling or strided transposed convolution

Input: 3 x H x W


Med-res: D_2 x H/4 x W/4
Med-res: D_2 x H/4 x W/4
Low-res: D_3 x H/4 x W/4
High-res: D_1 x H/2 x W/2
High-res: D_1 x H/2 x W/2



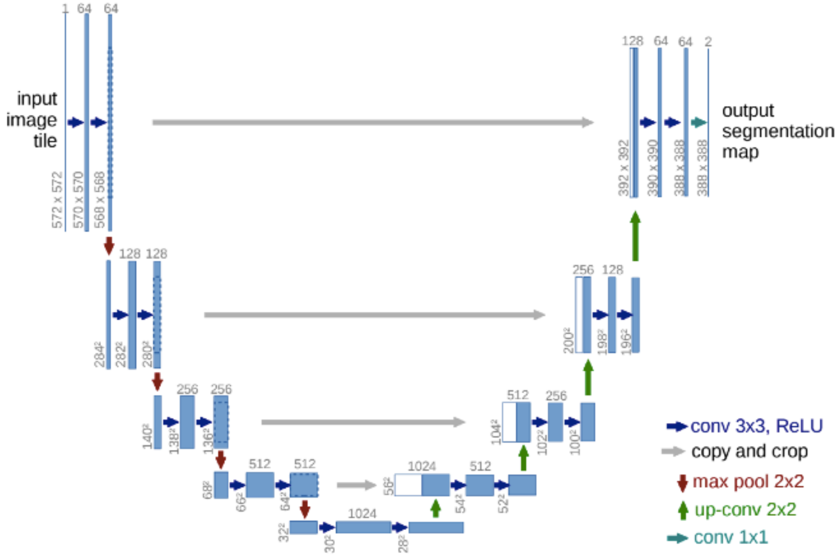
Predictions: H x W

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

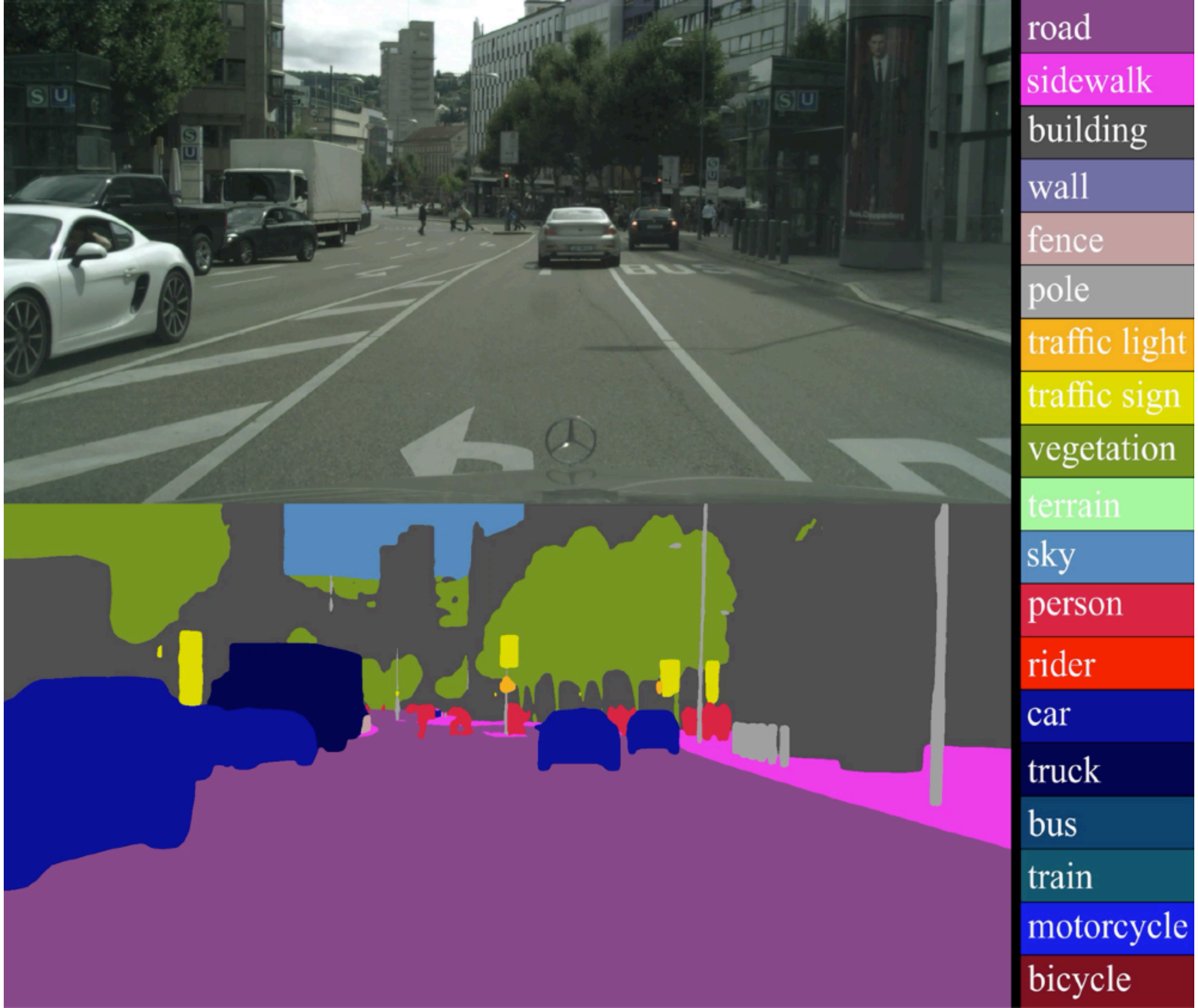
U-Net

O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, MICCAI 2015

- Like FCN, fuse upsampled higher-level feature maps with higher-res, lower-level feature maps
- Unlike FCN, fuse by concatenation, predict at the end

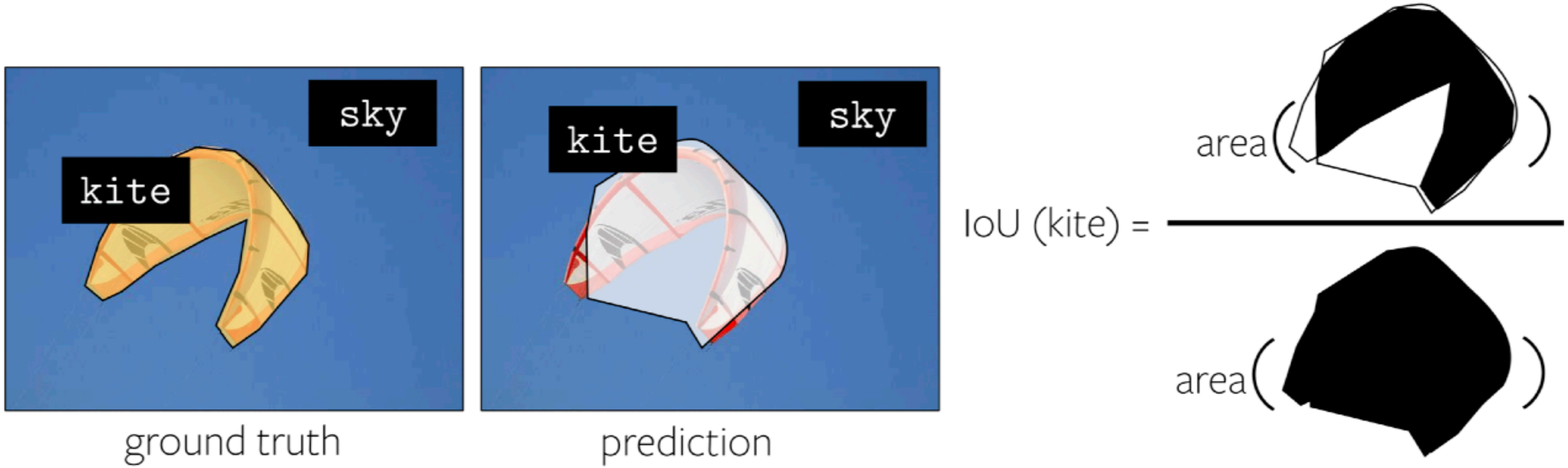


Slide credit: S. Lazebnik



road
sidewalk
building
wall
fence
pole
traffic light
traffic sign
vegetation
terrain
sky
person
rider
car
truck
bus
train
motorcycle
bicycle

Evaluation of Semantic Segmentation



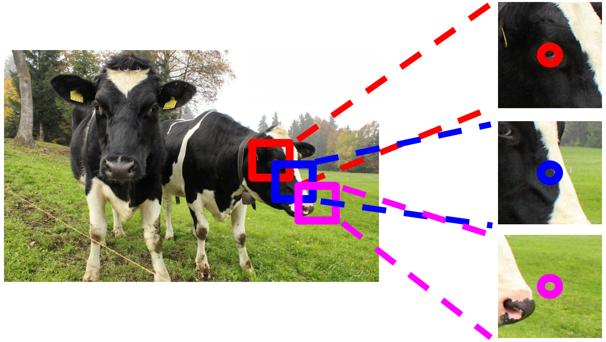
ground truth

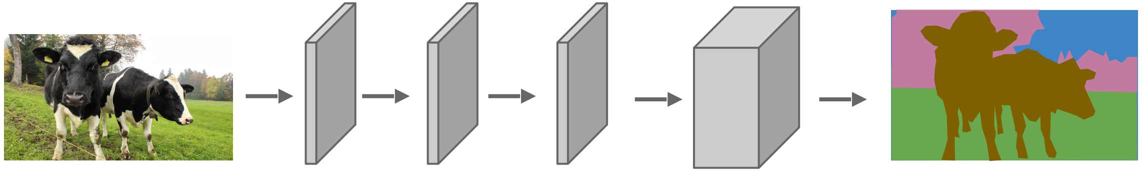
prediction

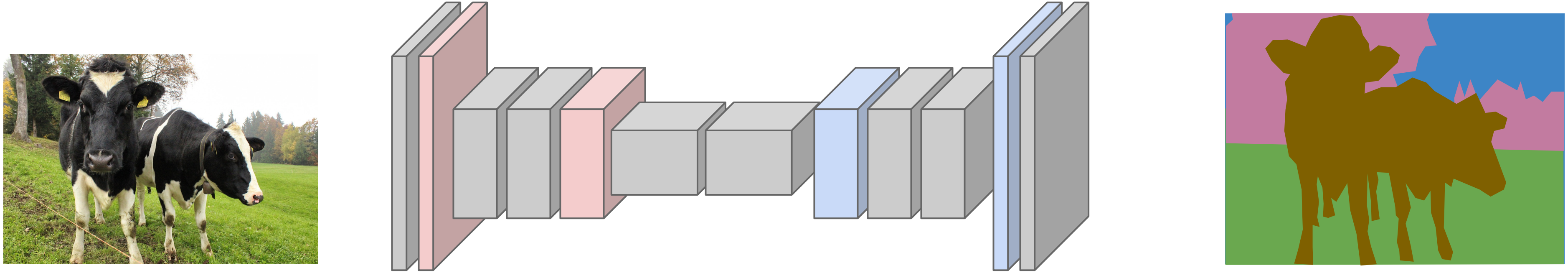
mIoU (mean IoU) per class

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Semantic Segmentation: Summary







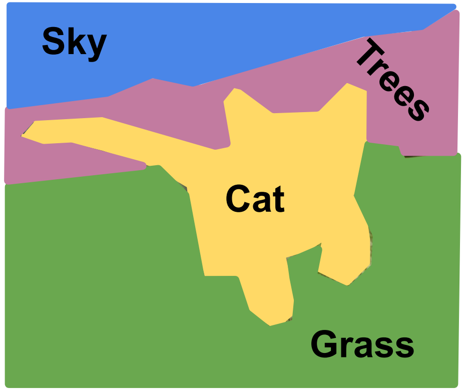
Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

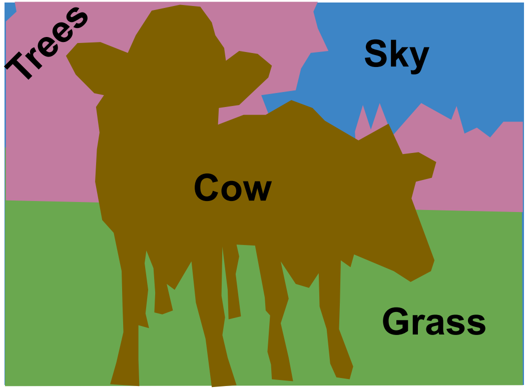
Semantic Segmentation

Label each pixel in the image with a category label

Don’t differentiate instances, only care about pixels

This image is CC0 public domain





Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

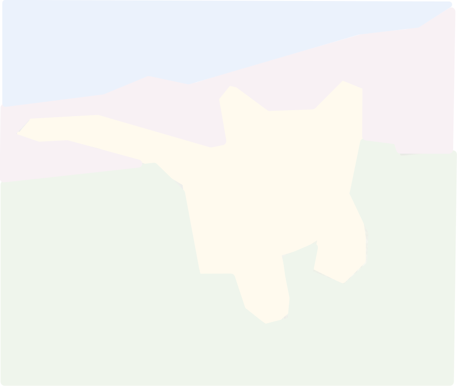
Object Detection

Classification

CAT

No spatial extent

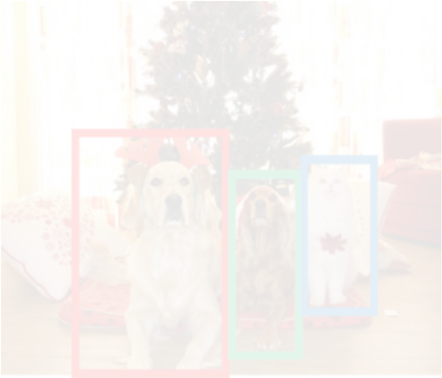
Semantic Segmentation



GRASS, CAT, TREE, SKY

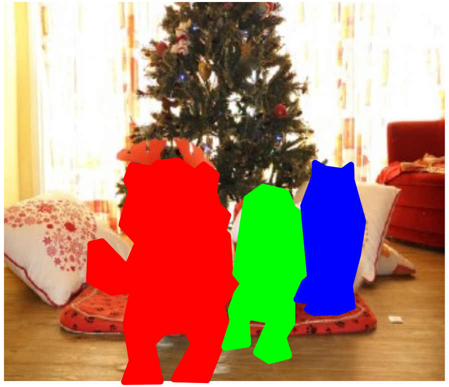
No objects, just pixels

Object Detection



DOG, DOG, CAT

Instance Segmentation



DOG, DOG, CAT

Multiple Object

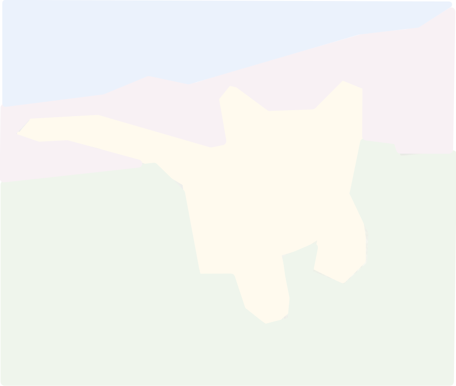
Object Detection

Classification

CAT

No spatial extent

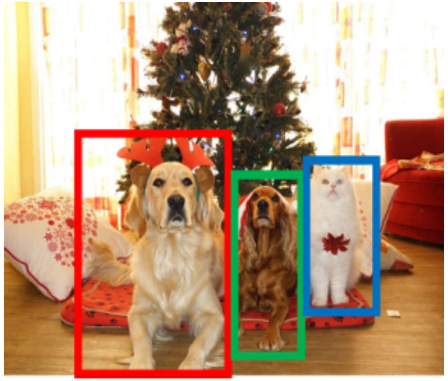
Semantic Segmentation



GRASS, CAT, TREE, SKY

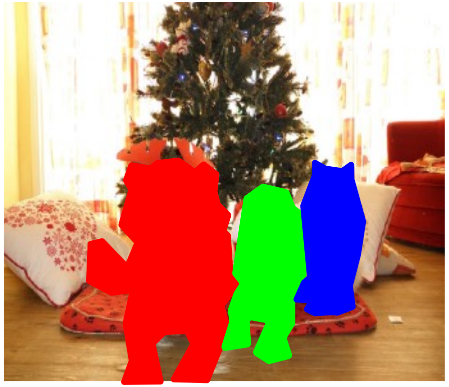
No objects, just pixels

Object Detection



DOG, DOG, CAT

Instance Segmentation

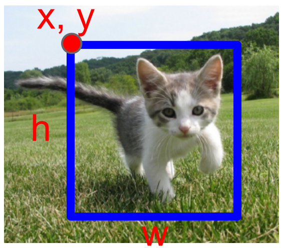


DOG, DOG, CAT

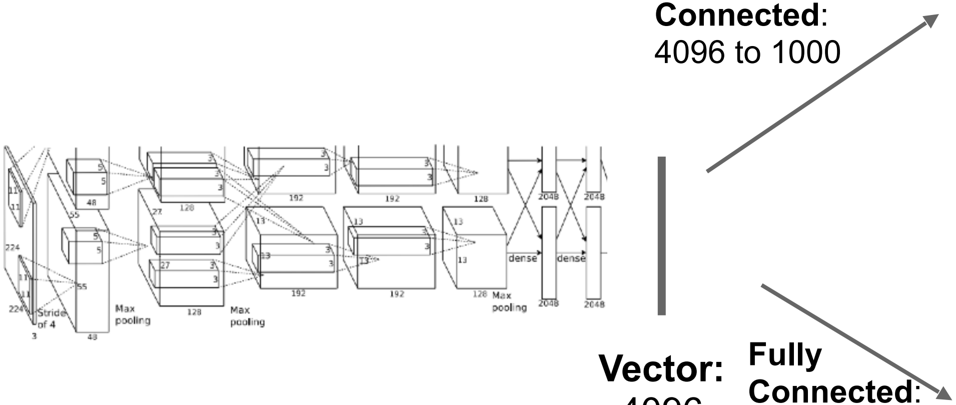
Multiple Object

Object Detection: Single Object

(Classification + Localization)



This image is CC0 public domain



Fully Connected: 4096 to 1000

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

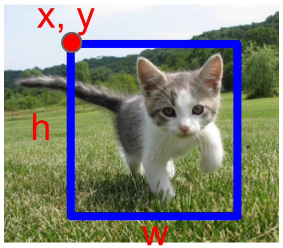
Vector: 4096

Fully Connected: 4096 to 4

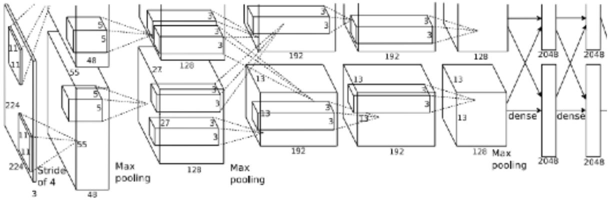
Box Coordinates
(x, y, w, h)

Object Detection: Single Object

(Classification + Localization)



This image is CC0 public domain



Vector: 4096

Fully Connected: 4096 to 1000

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

Softmax Loss

Correct label:
Cat

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

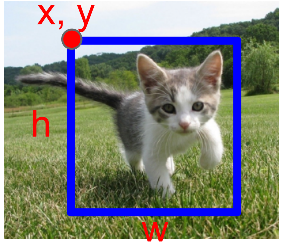
L2 Loss

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

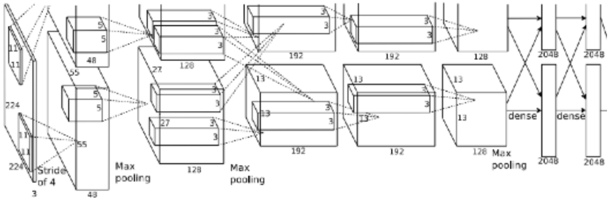
Treat localization as a regression problem!

Object Detection: Single Object

(Classification + Localization)



This image is CC0 public domain



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')

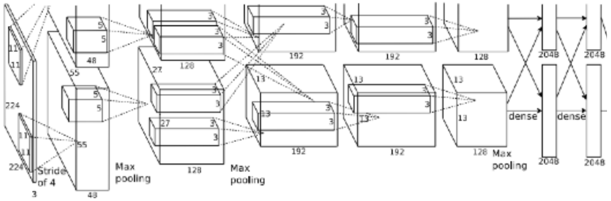
Multitask Loss

+ → **Loss**

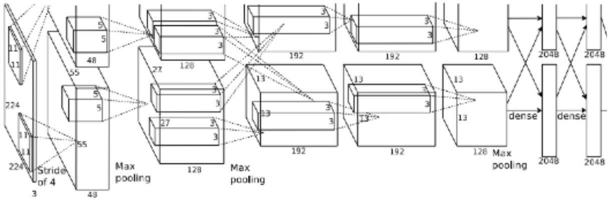
Treat localization as a regression problem!

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

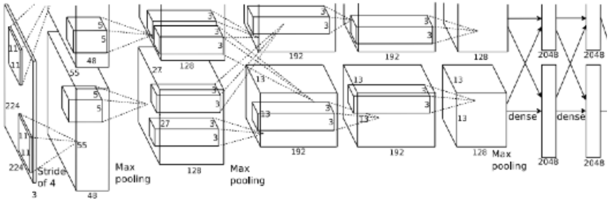
Object Detection: Multiple Objects



CAT: (x, y, w, h)



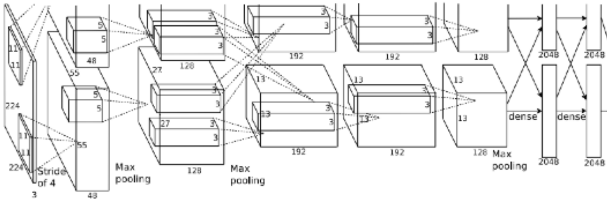
DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)



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

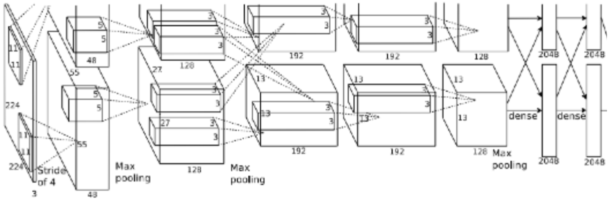
Object Detection: Multiple Objects

Each image needs a different number of outputs!



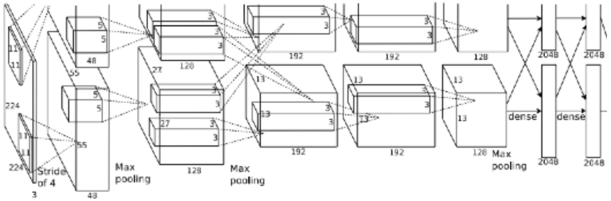
CAT: (x, y, w, h)

4 numbers



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

12 numbers

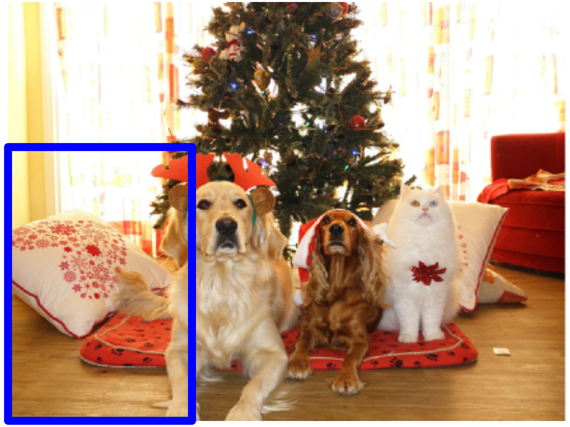


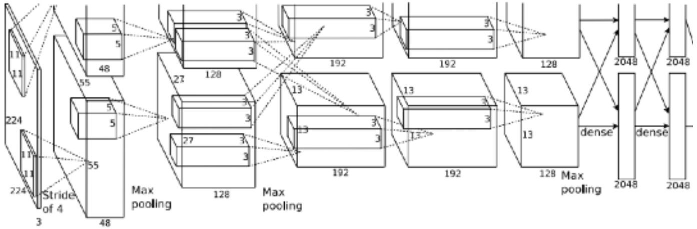
DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
….

Many numbers!

Object Detection: Multiple Objects

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

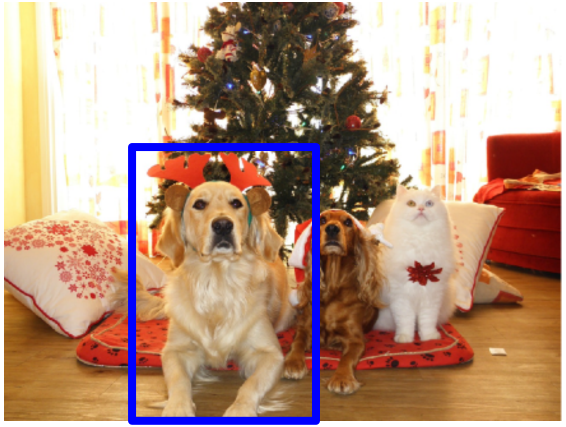


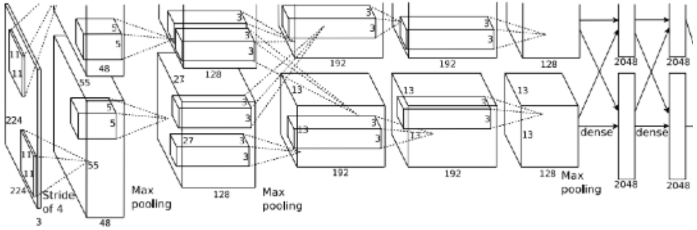


Dog? NO
Cat? NO
Background? YES

Object Detection: Multiple Objects

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

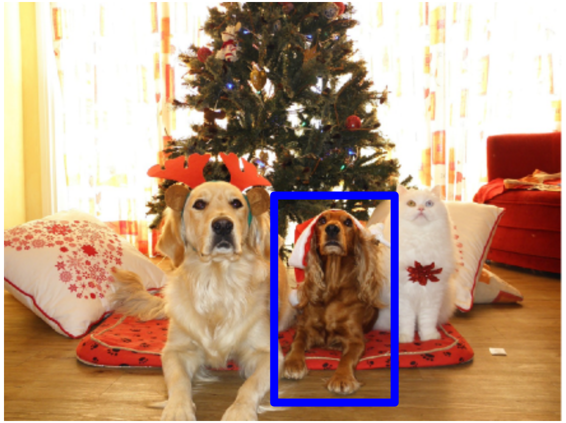


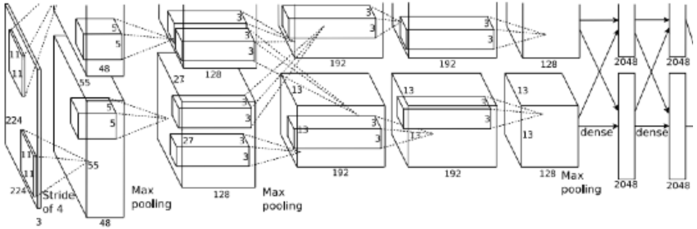


Dog? YES
Cat? NO
Background? NO

Object Detection: Multiple Objects

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



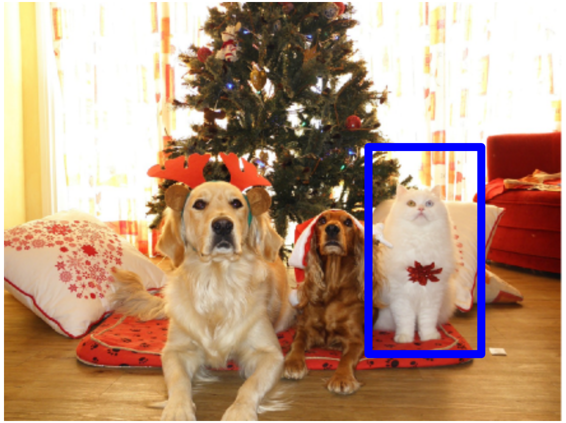


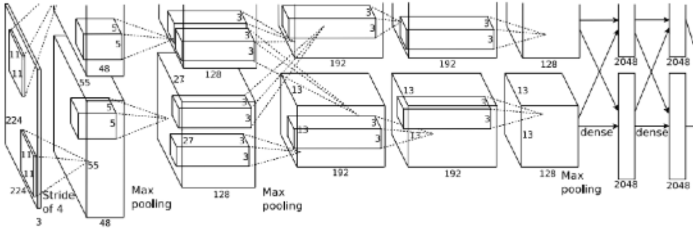
Dog? YES
Cat? NO
Background? NO

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Object Detection: Multiple Objects

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





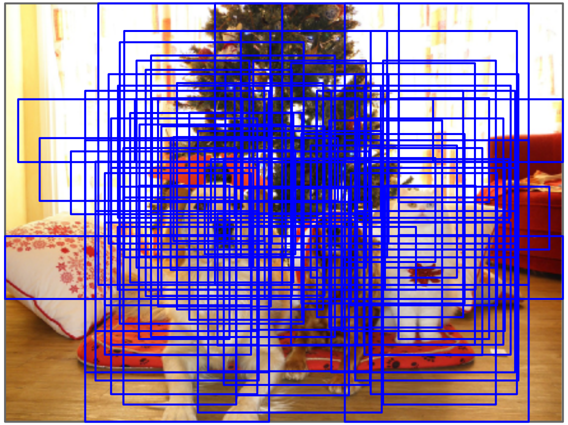
Dog? NO
Cat? YES
Background? NO

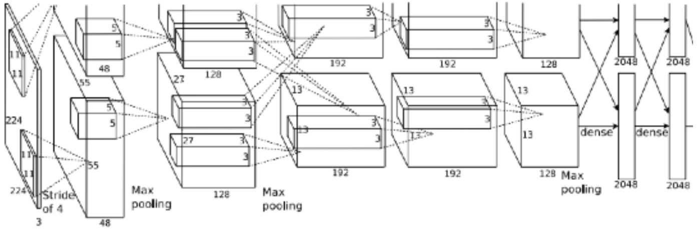
Q: What’s the problem with this approach?

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Object Detection: Multiple Objects

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





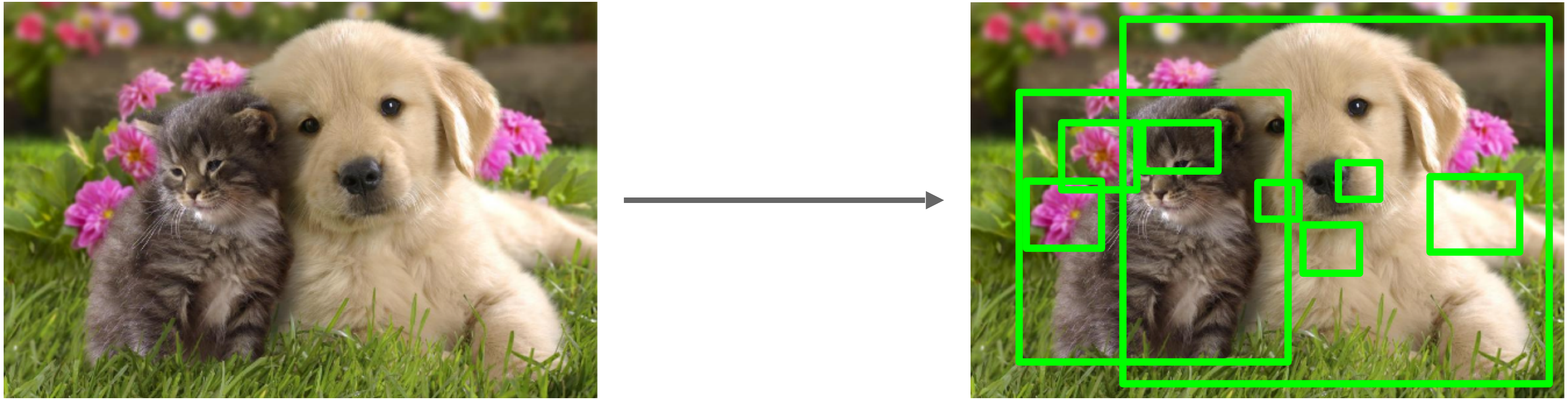
Dog? NO
Cat? YES
Background? NO

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

Slide credits: Erik Learned-Miller, Fei-Fei Li, Jiajun Wu, Ruohan Gao

Region Proposals: Selective Search

- Find “blobby” image regions that are likely to contain objects
- Relatively fast to run; e.g. Selective Search gives 2000 region proposals in a few seconds on CPU



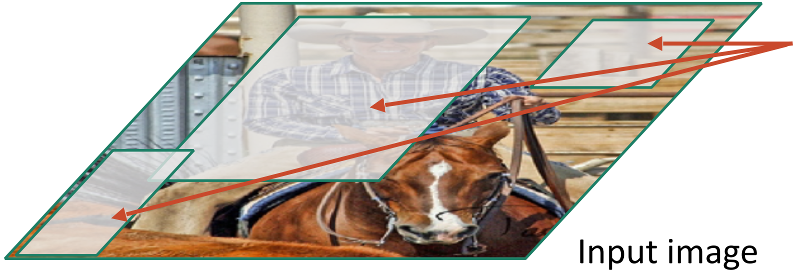
Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012
Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013
Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014
Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”,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

R-CNN

Input image

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

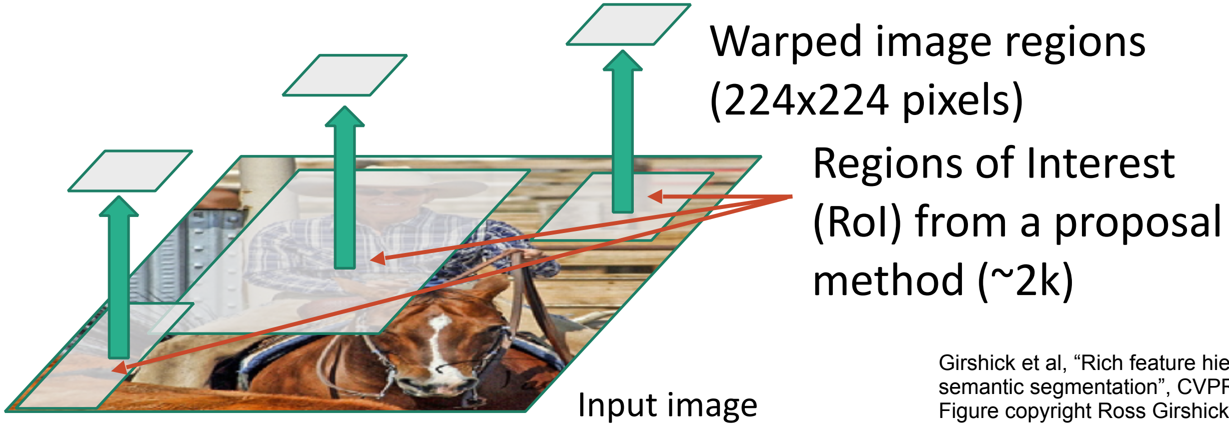
R-CNN



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

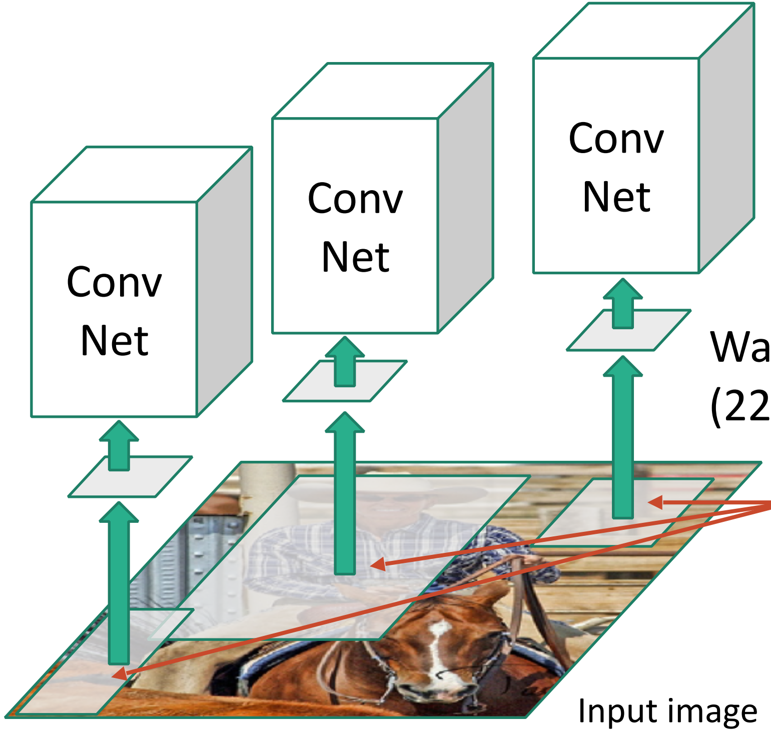
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN



Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN



Forward each region through ConvNet (ImageNet-pretranied)

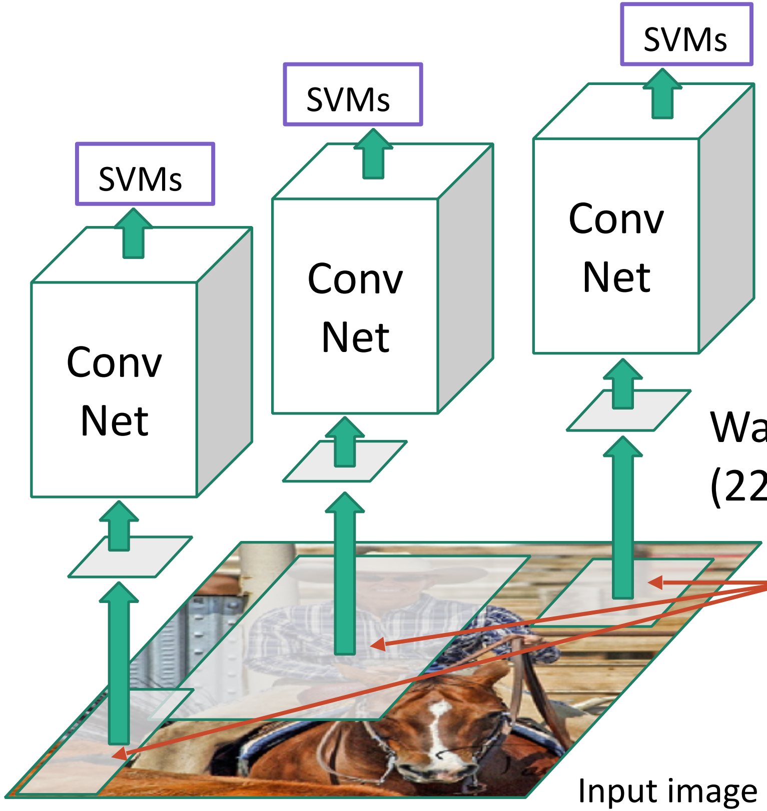
Warped image regions (224x224 pixels)

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

Input image

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN



Classify regions with SVMs

Forward each region through ConvNet (ImageNet-pretranied)

Warped image regions (224x224 pixels)

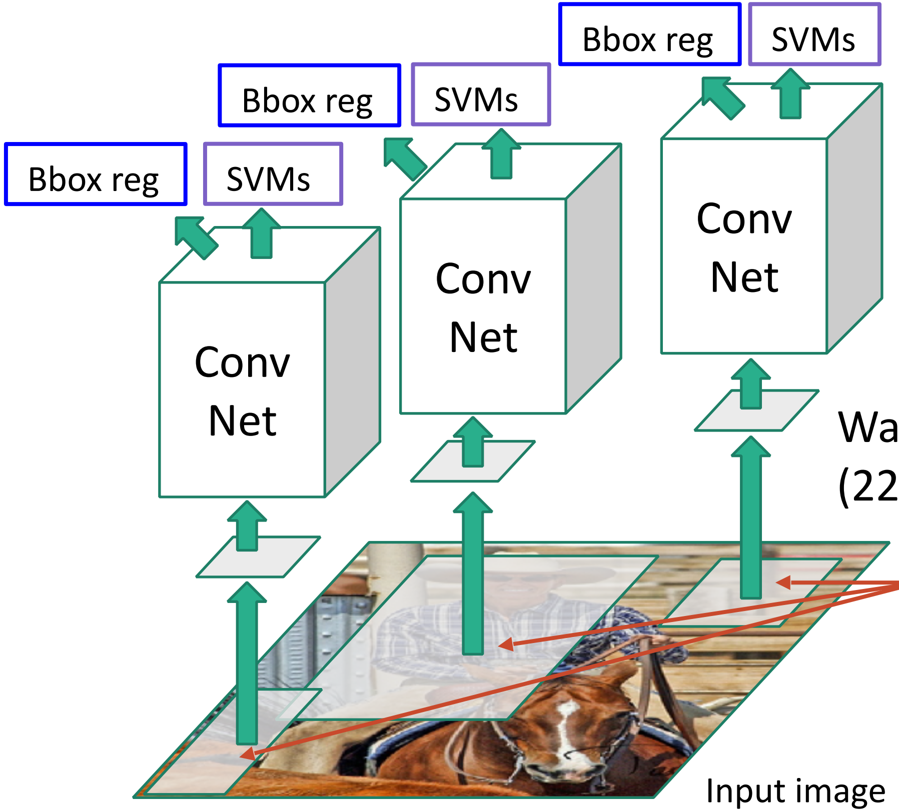
Regions of Interest (RoI) from a proposal method (~2k)

Input image

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN

Predict “corrections” to the RoI: 4 numbers: (dx, dy, dw, dh)



Classify regions with SVMs

Forward each region through ConvNet (ImageNet-pretranied)

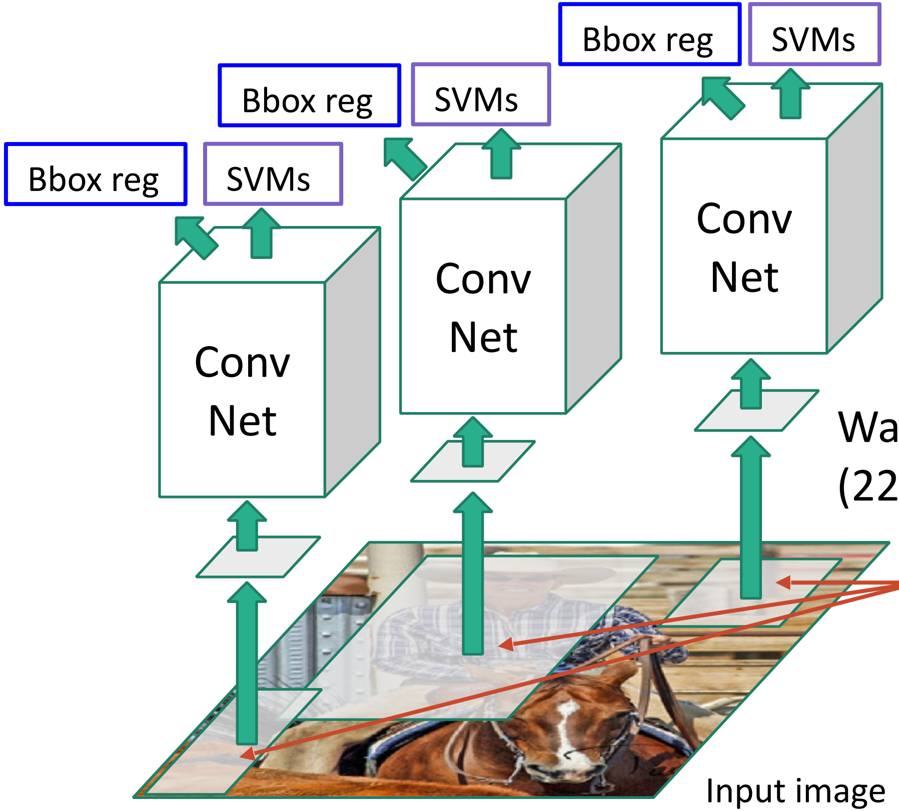
Warped image regions (224x224 pixels)

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

Input image

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN



Predict “corrections” to the RoI: 4 numbers: (dx, dy, dw, dh)

Classify regions with SVMs

Forward each region through ConvNet

Warped image regions (224x224 pixels)

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

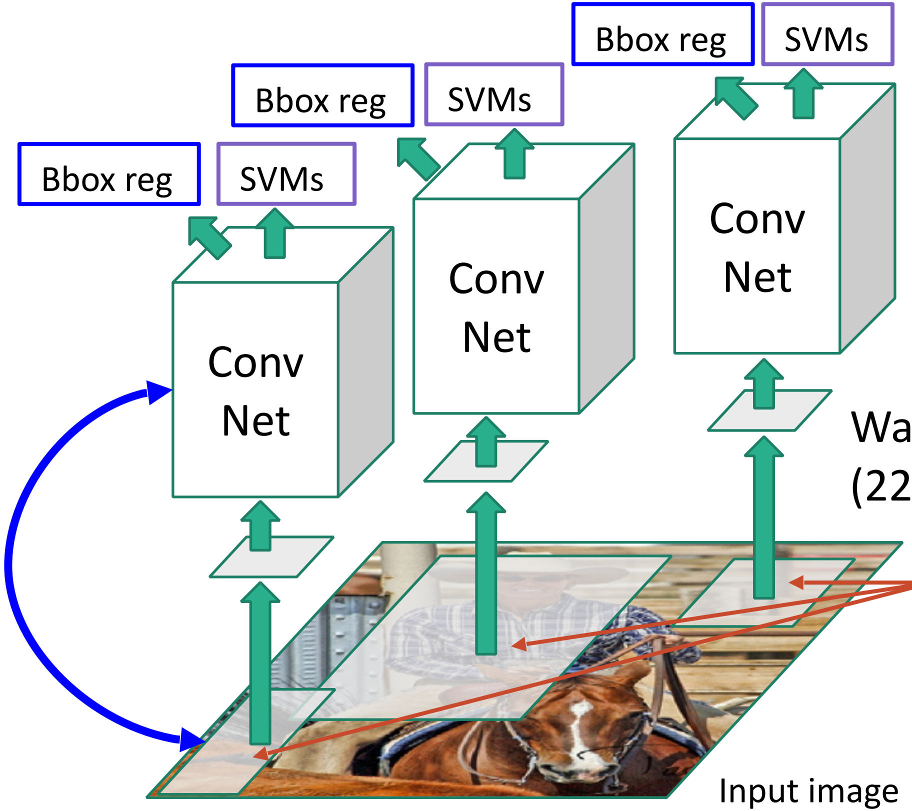
Input image

Problem: Very slow! Need to do ~2k independent forward passes for each image!

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

"Slow" R-CNN

Predict "corrections" to the RoI: 4 numbers: (dx, dy, dw, dh)



Classify regions with SVMs

Forward each region through ConvNet

Warped image regions (224x224 pixels)

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

Input image

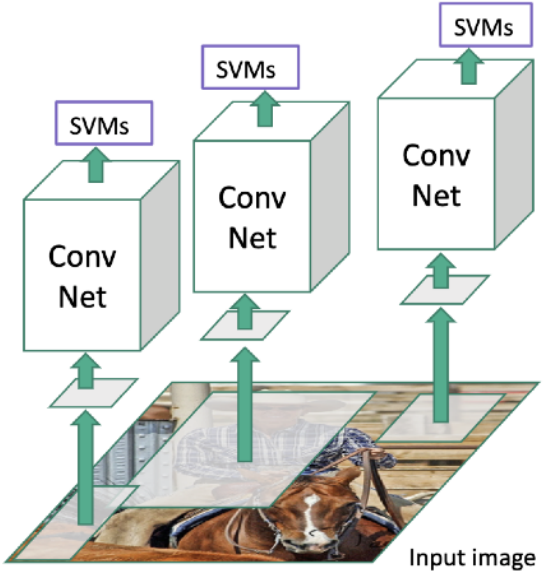
Problem: Very slow! Need to do ~2k independent forward passes for each image!

Idea: Pass the image through convnet before cropping! Crop the conv feature instead!

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN

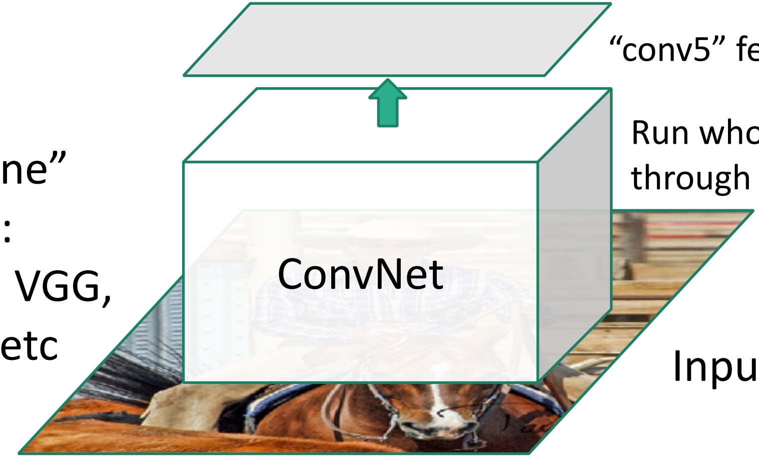
“Slow” R-CNN



Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN

"Backbone" network: AlexNet, VGG, ResNet, etc

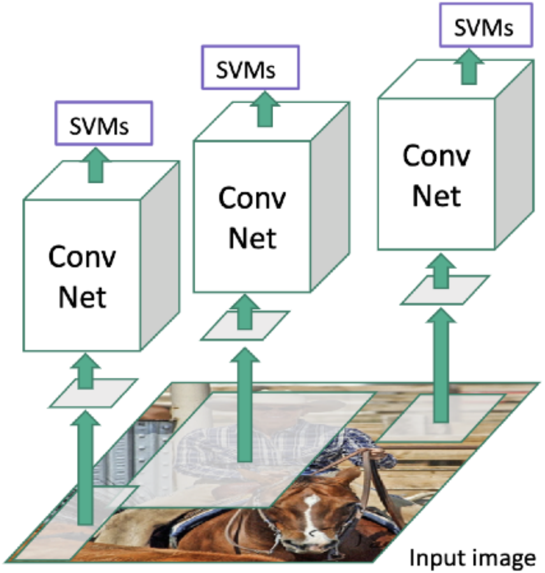


"conv5" features

Run whole image through ConvNet

Input image

"Slow" R-CNN

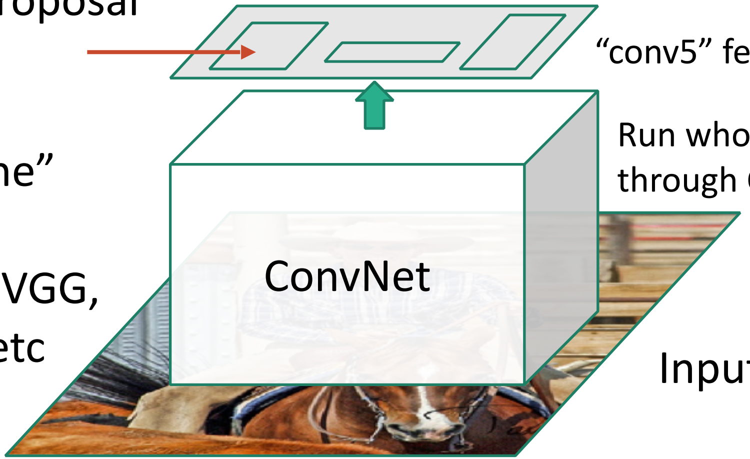


Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN

Regions of Interest (RoIs) from a proposal method

“Backbone” network: AlexNet, VGG, ResNet, etc

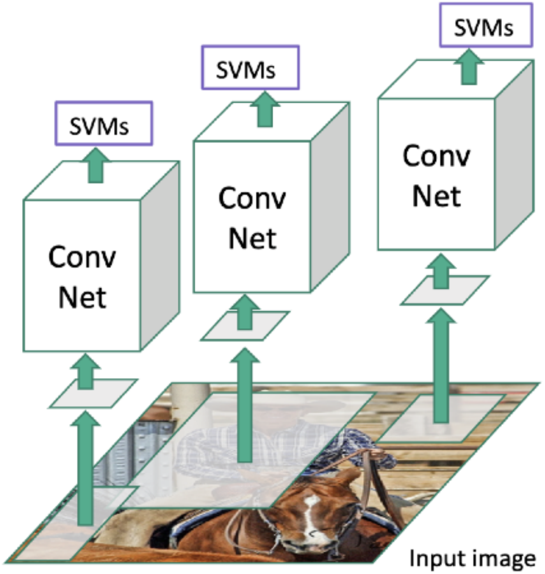


“conv5” features

Run whole image through ConvNet

Input image

“Slow” R-CNN

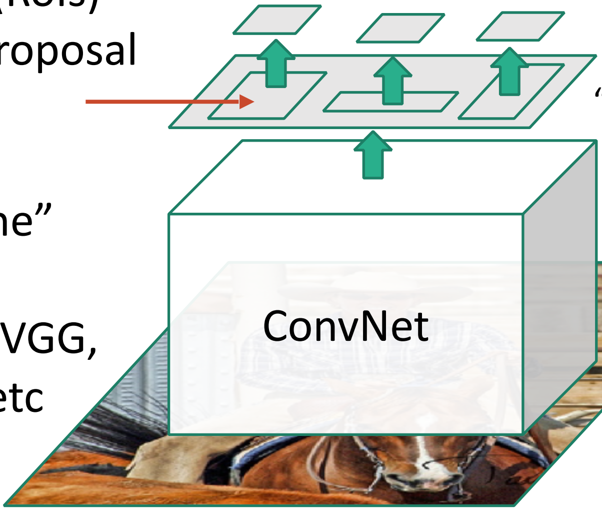


Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN

Regions of Interest (RoIs) from a proposal method

“Backbone” network: AlexNet, VGG, ResNet, etc



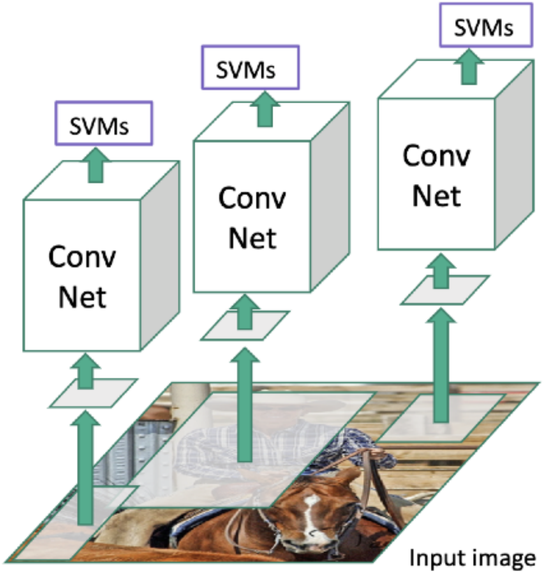
Crop + Resize features

“conv5” features

Run whole image through ConvNet

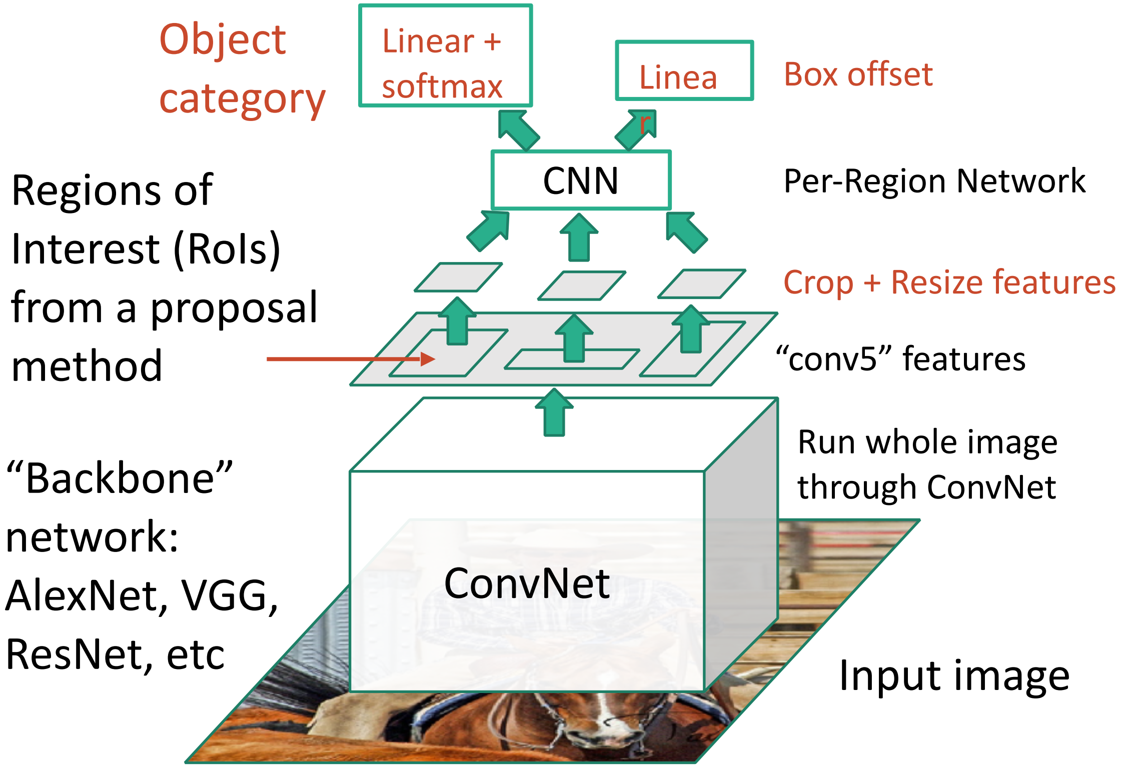
Input image

“Slow” R-CNN



Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN



Object category

Linear + softmax

Linea
r

Box offset

CNN

Per-Region Network

Regions of Interest (RoIs) from a proposal method

Crop + Resize features

“conv5” features

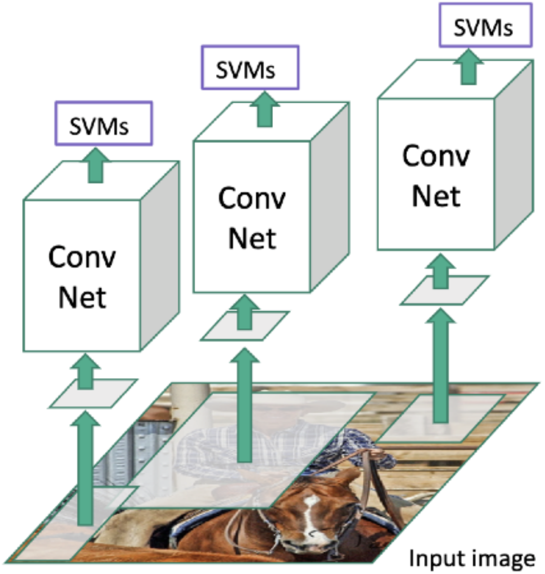
“Backbone” network: AlexNet, VGG, ResNet, etc

ConvNet

Run whole image through ConvNet

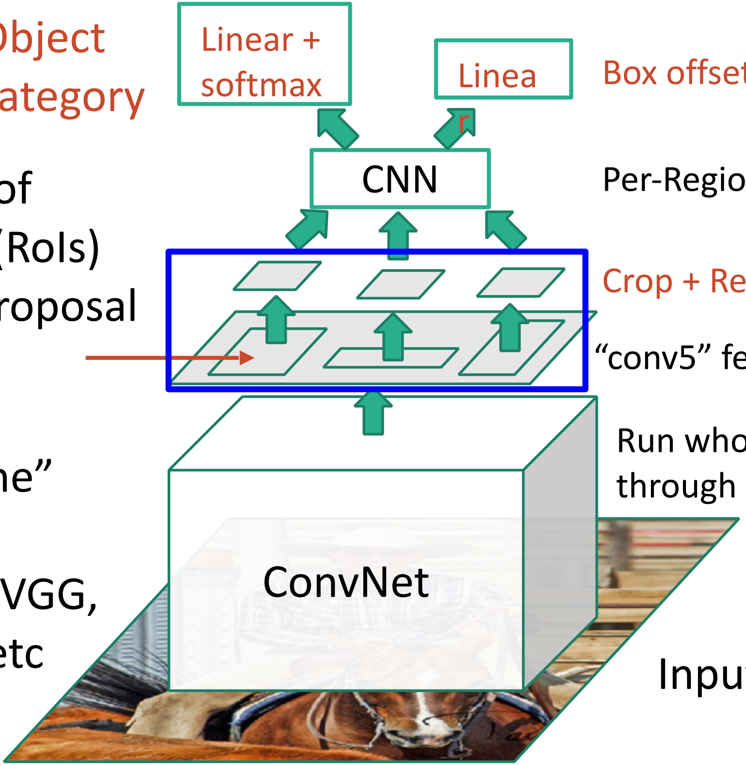
Input image

“Slow” R-CNN



Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN



Object category

Box offset

Per-Region Network

Regions of Interest (RoIs) from a proposal method

Crop + Resize features

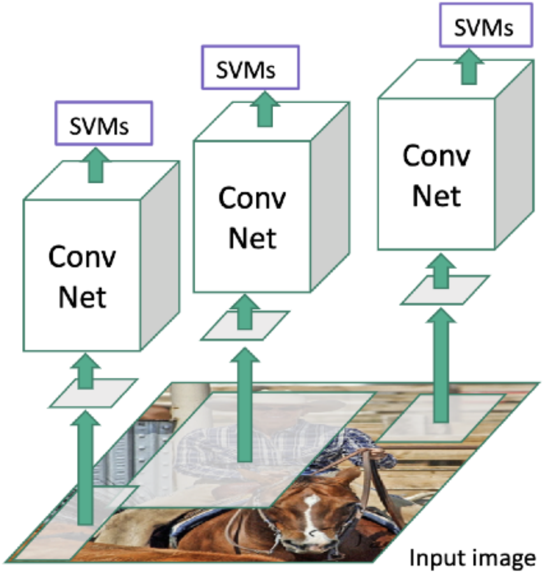
"conv5" features

Run whole image through ConvNet

"Backbone" network: AlexNet, VGG, ResNet, etc

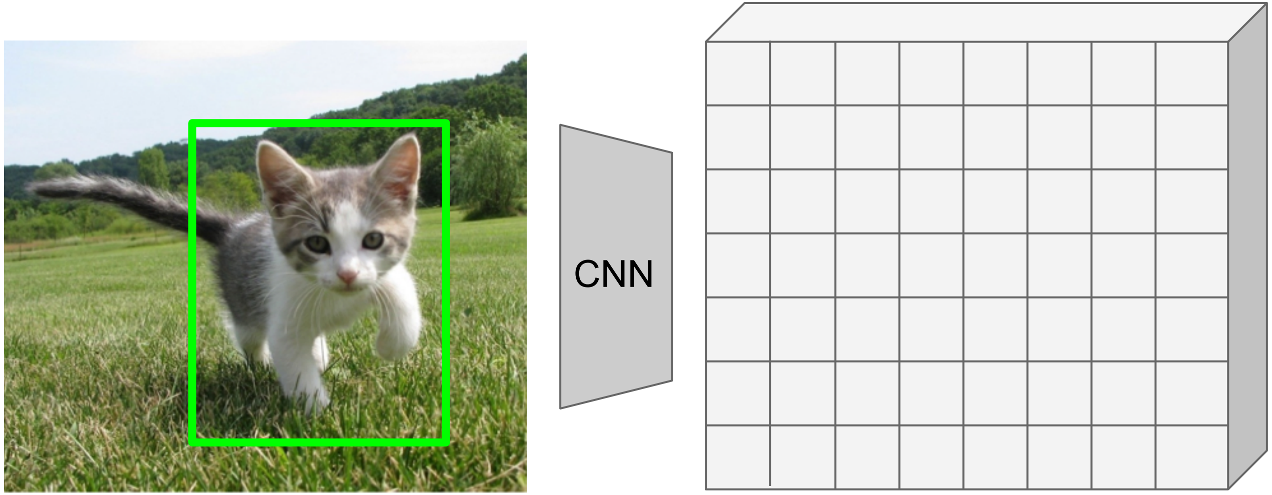
Input image

"Slow" R-CNN



Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Cropping Features: RoI Pool



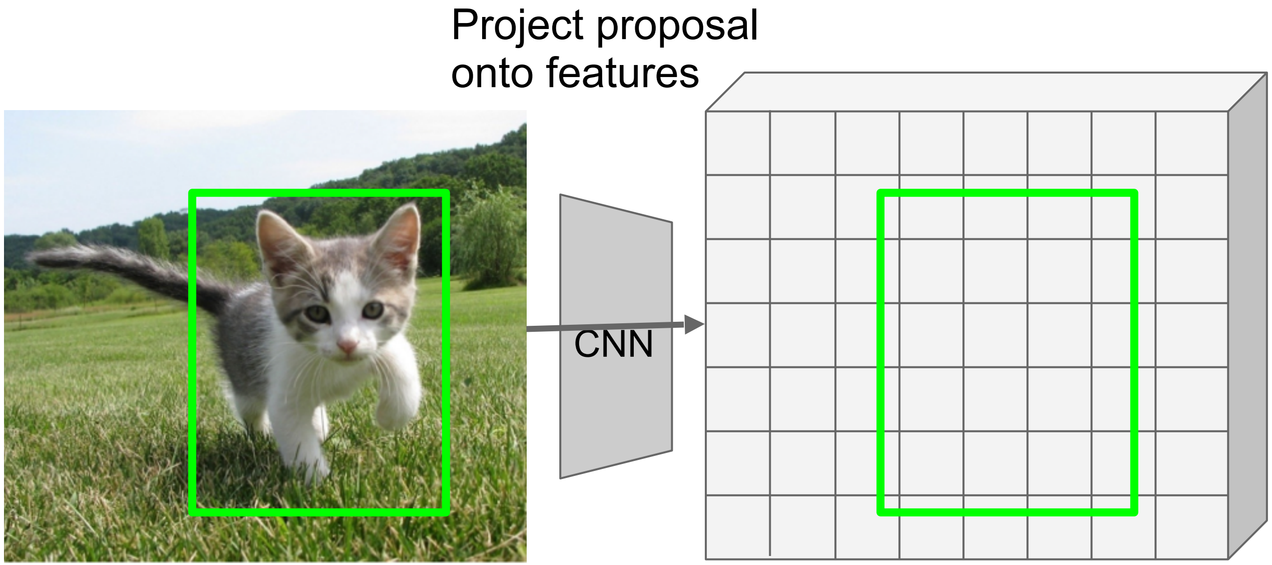
Input Image
(e.g. 3 x 640 x 480)

Image features: C x H x W
(e.g. 512 x 20 x 15)

Girshick, “Fast R-CNN”, ICCV 2015.

Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool



Project proposal
onto features

CNN

Input Image
(e.g. 3 x 640 x 480)

Image features: C x H x W
(e.g. 512 x 20 x 15)

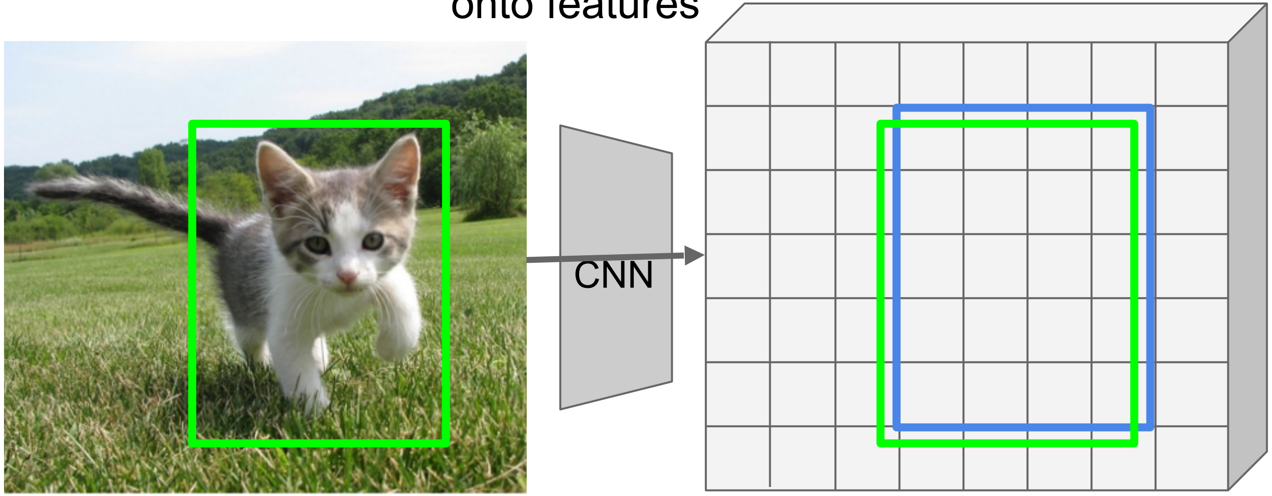
Girshick, “Fast R-CNN”, ICCV 2015.

Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool

“Snap” to grid cells

Project proposal onto features



Input Image
(e.g. 3 x 640 x 480)

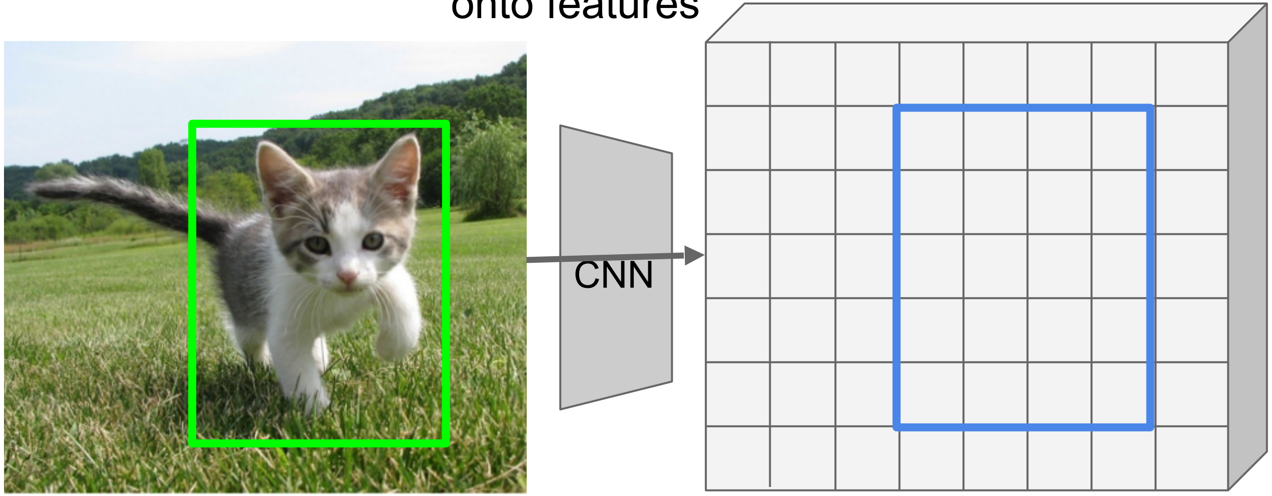
Image features: C x H x W
(e.g. 512 x 20 x 15)

Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool

“Snap” to grid cells

Project proposal onto features



Input Image
(e.g. 3 x 640 x 480)

Image features: C x H x W
(e.g. 512 x 20 x 15)

Q: how do we resize the 512 x 5 x 4 region to, e.g., a 512 x 2 x 2 tensor?.

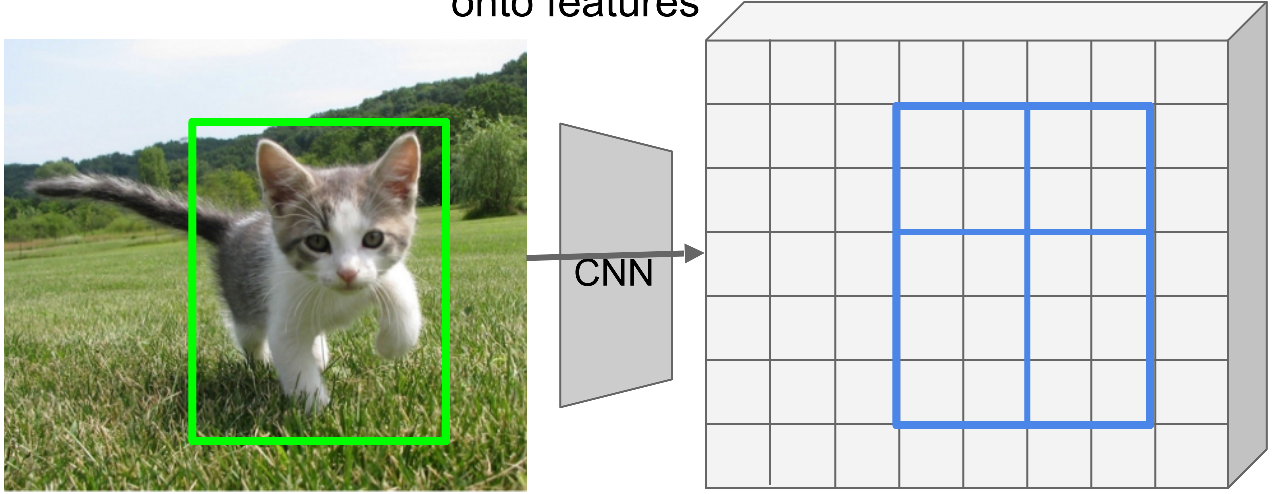
Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool

“Snap” to grid cells

Divide into 2x2 grid of (roughly) equal subregions

Project proposal onto features



CNN

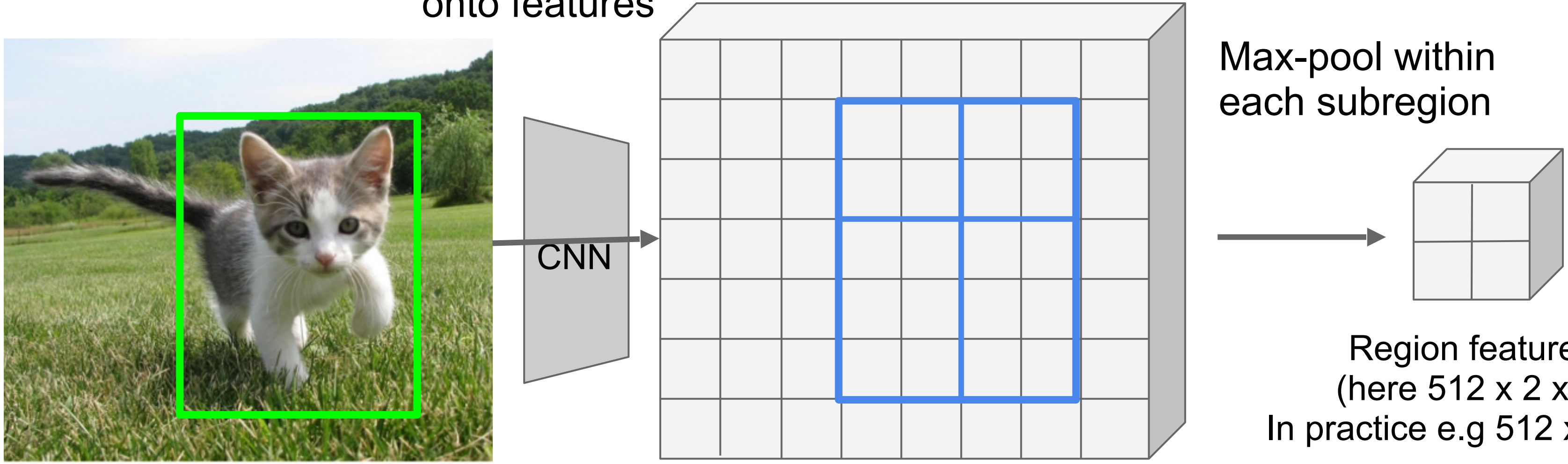
Input Image
(e.g. 3 x 640 x 480)

Image features: C x H x W
(e.g. 512 x 20 x 15)

Q: how do we resize the 512 x 5 x 4 region to, e.g., a 512 x 2 x 2 tensor?.

Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool



Project proposal onto features

“Snap” to grid cells

Divide into 2x2 grid of (roughly) equal subregions

Max-pool within each subregion

Input Image
(e.g. 3 x 640 x 480)

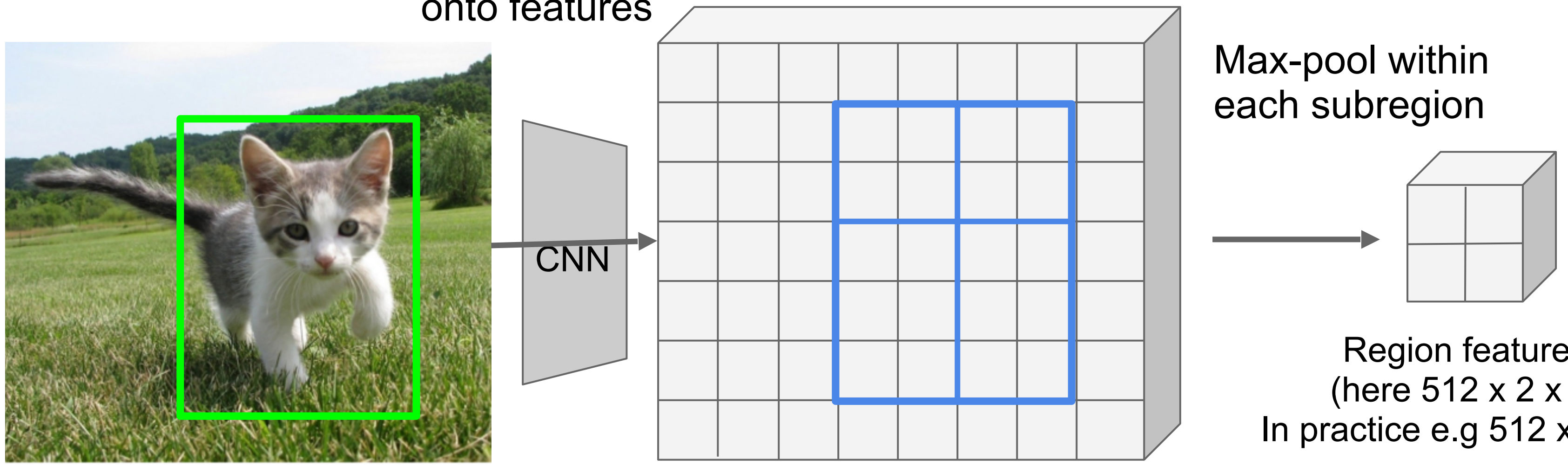
Image features: C x H x W
(e.g. 512 x 20 x 15)

Region features
(here 512 x 2 x 2;
In practice e.g 512 x 7 x 7)

Region features always the same size even if input regions have different sizes!

Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool



“Snap” to grid cells

Divide into 2x2 grid of (roughly) equal subregions

Project proposal onto features

Max-pool within each subregion

Input Image
(e.g. 3 x 640 x 480)

Image features: C x H x W
(e.g. 512 x 20 x 15)

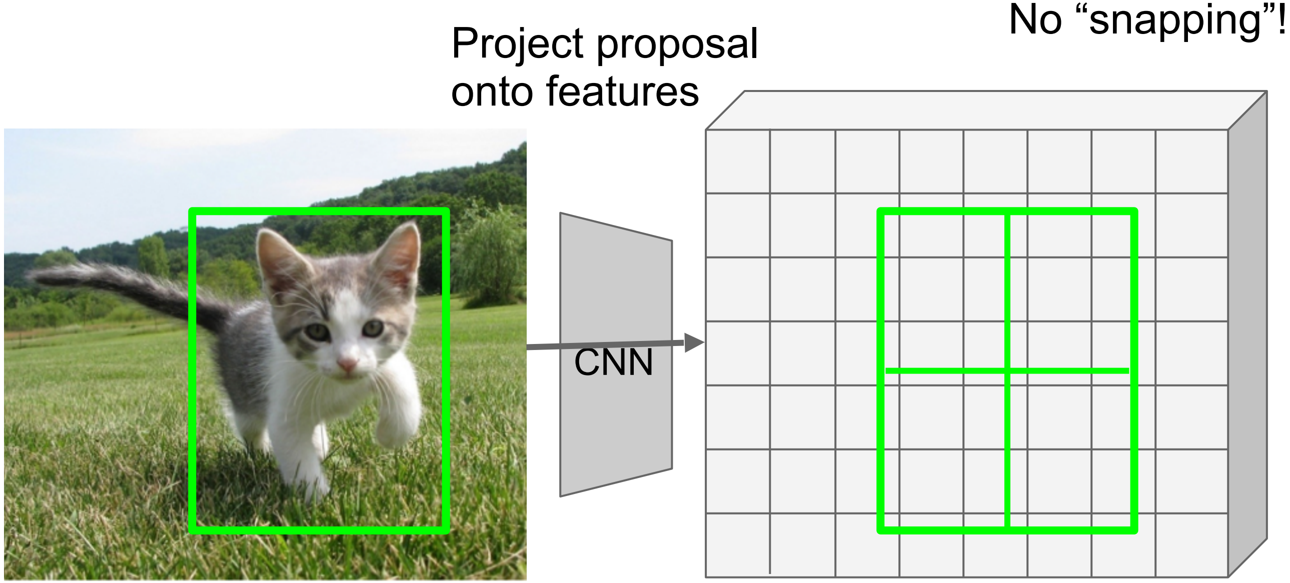
Region features
(here 512 x 2 x 2;
In practice e.g 512 x 7 x 7)

Region features always the same size even if input regions have different sizes!

Problem: Region features slightly misaligned

Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Align



Input Image
(e.g. 3 x 640 x 480)

Image features: C x H x W
(e.g. 512 x 20 x 15)

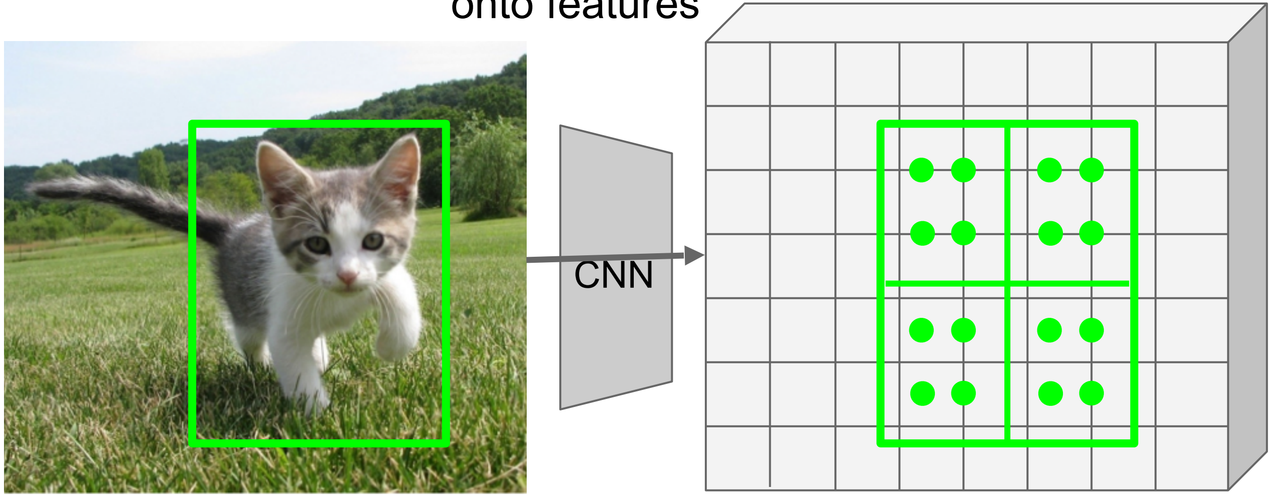
He et al, “Mask R-CNN”, ICCV 2017

Cropping Features: RoI Align

Sample at regular points in each subregion using bilinear interpolation

No “snapping”!

Project proposal onto features

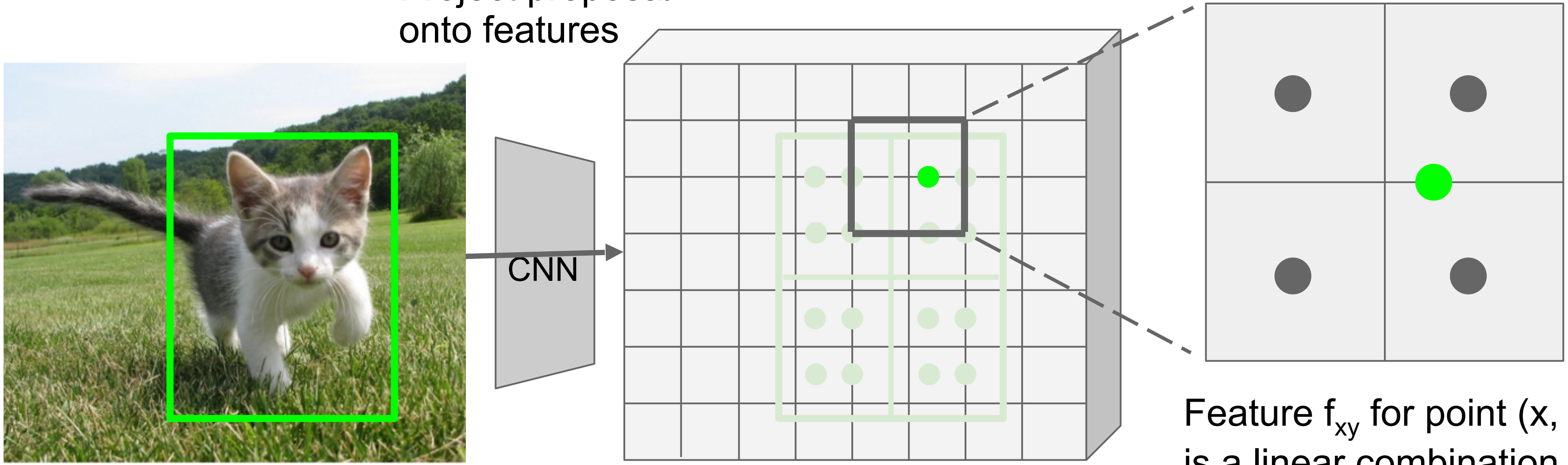


Input Image
(e.g. 3 x 640 x 480)

Image features: C x H x W
(e.g. 512 x 20 x 15)

He et al, “Mask R-CNN”, ICCV 2017

Cropping Features: RoI Align



Project proposal onto features

No "snapping"!

Sample at regular points in each subregion using bilinear interpolation

Input Image
(e.g. 3 x 640 x 480)

Image features: C x H x W
(e.g. 512 x 20 x 15)

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

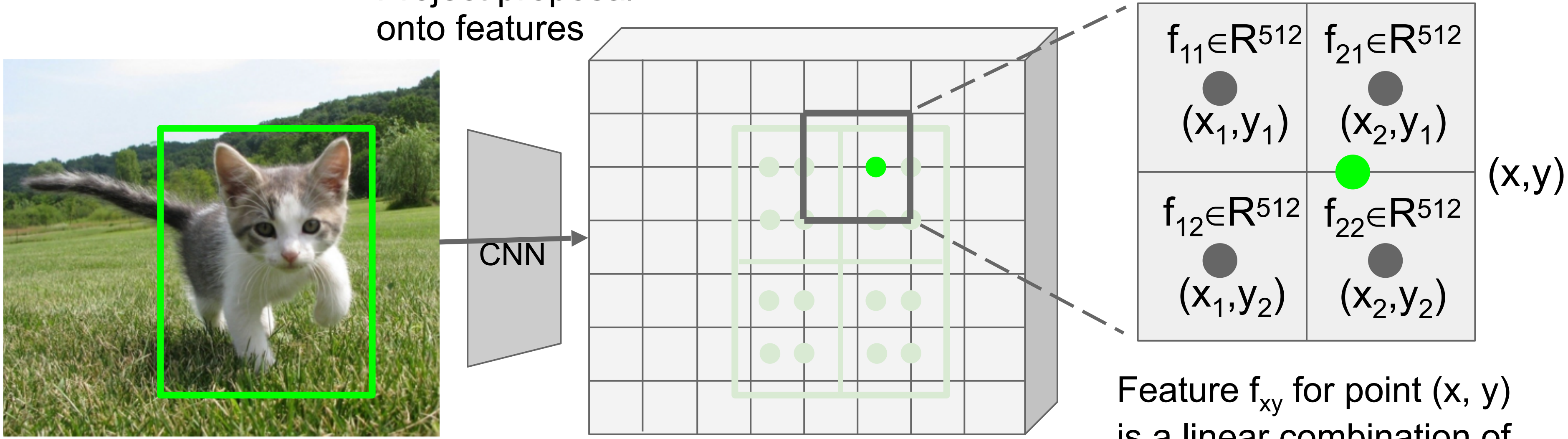
He et al, "Mask R-CNN", ICCV 2017

Cropping Features: RoI Align

Project proposal onto features

No “snapping”!

Sample at regular points in each subregion using bilinear interpolation



Input Image
(e.g. 3 x 640 x 480)

Image features: C x H x W
(e.g. 512 x 20 x 15)

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

$$f_{xy} = \sum_{i,j=1}^{2} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_j|)$$

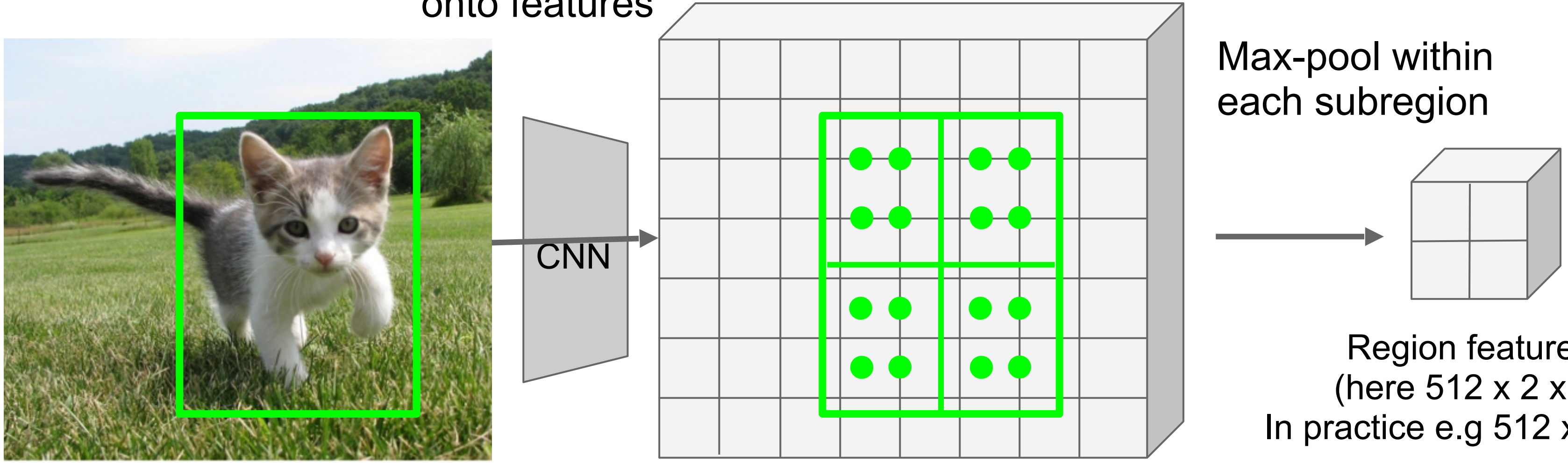
He et al, “Mask R-CNN”, ICCV 2017

Cropping Features: RoI Align

Sample at regular points in each subregion using bilinear interpolation

No “snapping”!

Project proposal onto features



Max-pool within each subregion

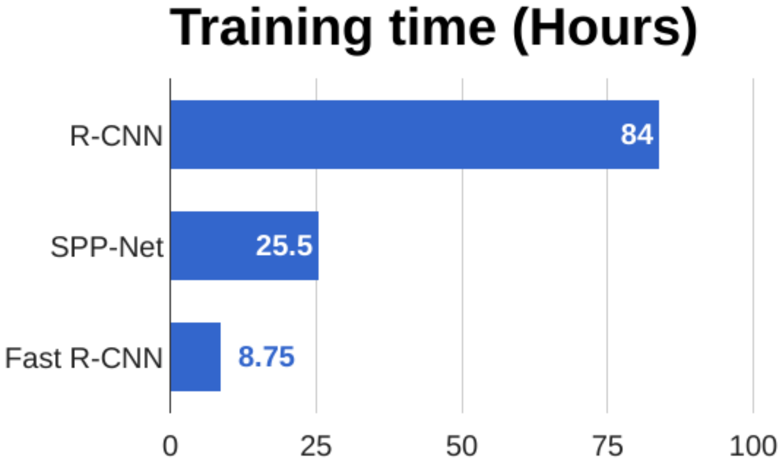
Input Image
(e.g. 3 x 640 x 480)

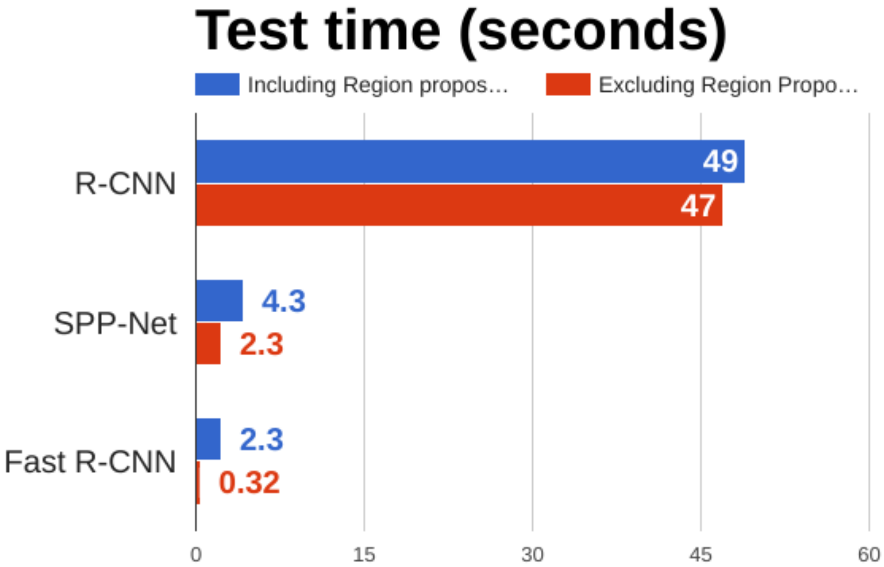
Image features: C x H x W
(e.g. 512 x 20 x 15)

Region features
(here 512 x 2 x 2;
In practice e.g 512 x 7 x 7)

He et al, “Mask R-CNN”, ICCV 2017

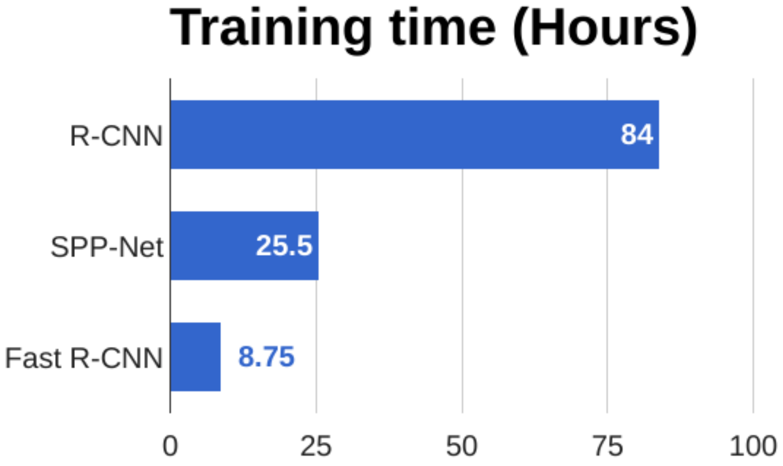
R-CNN vs Fast R-CNN

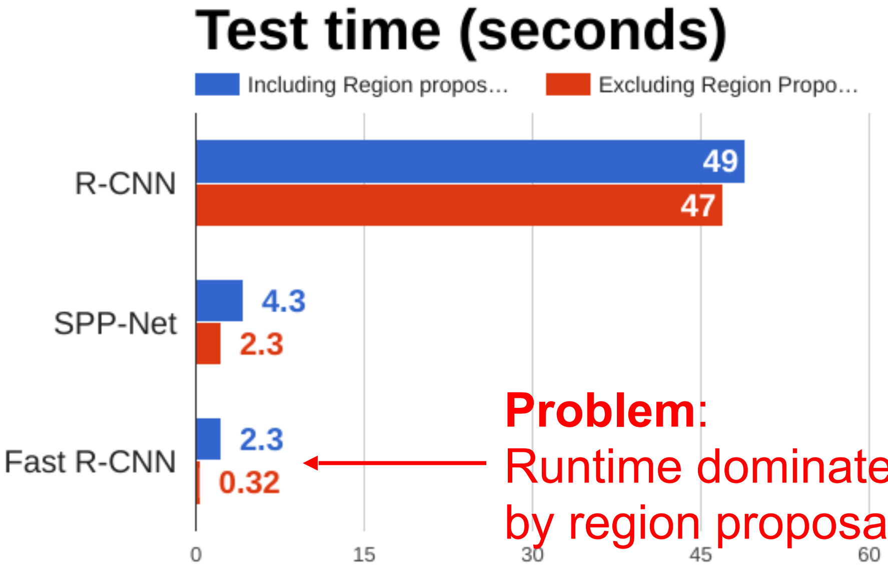




Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
He et al, “Spatial pyramid pooling in deep convolutional networks for visual recognition”, ECCV 2014
Girshick, “Fast R-CNN”, ICCV 2015

R-CNN vs Fast R-CNN





Problem: Runtime dominated by region proposals!

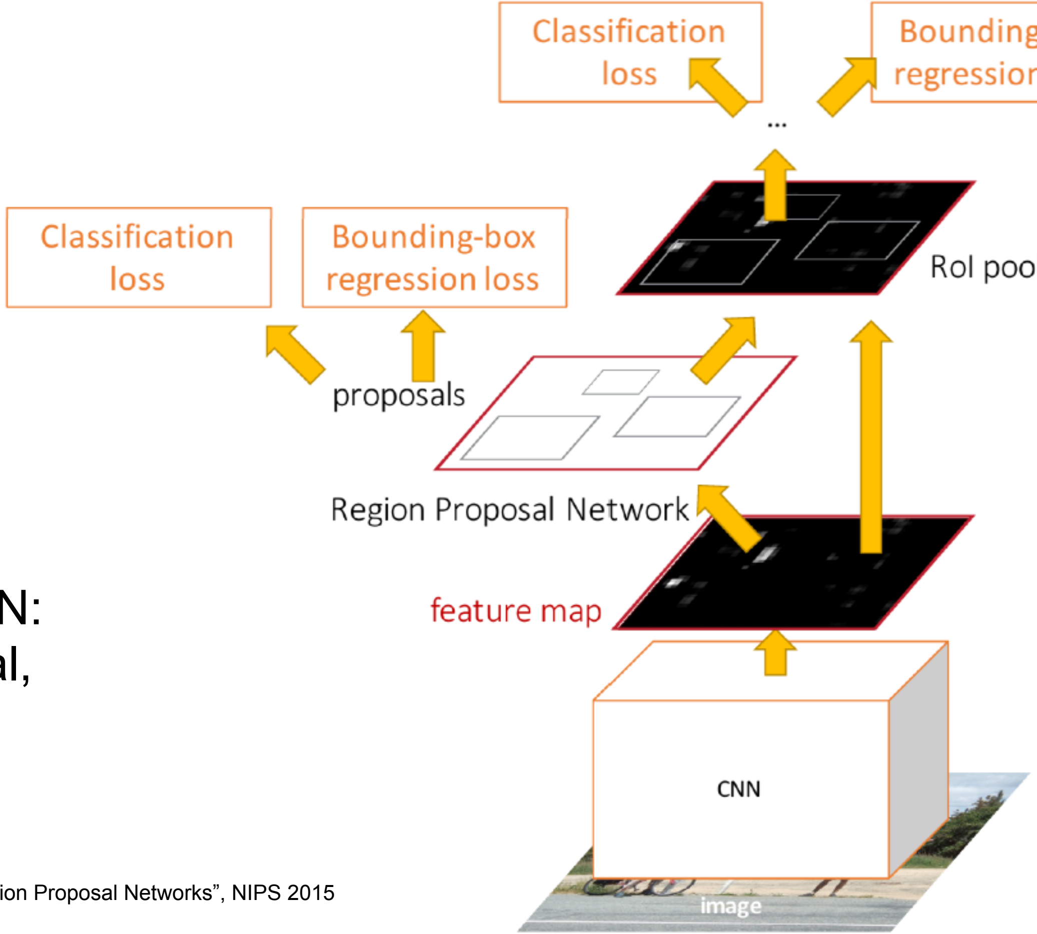
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
He et al, “Spatial pyramid pooling in deep convolutional networks for visual recognition”, ECCV 2014
Girshick, “Fast R-CNN”, ICCV 2015

Fast<u>**er**</u> R-CNN:

Make CNN do proposals!

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

Otherwise same as Fast R-CNN: Crop features for each proposal, classify each one



Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission