max planck institut informatik

SIC Saarland Informatics Campus

High Level Computer Vision

Object Detection and Segmentation @ May 22, 2024

Bernt Schiele

https://cms.sic.saarland/hlcvss24/

Max Planck Institute for Informatics & Saarland University,
Saarland Informatics Campus Saarbrücken

So far: Image Classification

This image is CC0 public domain

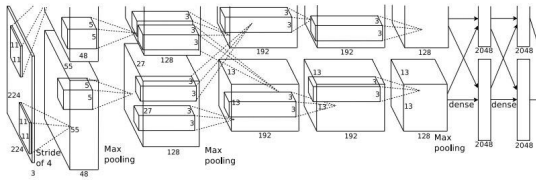


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

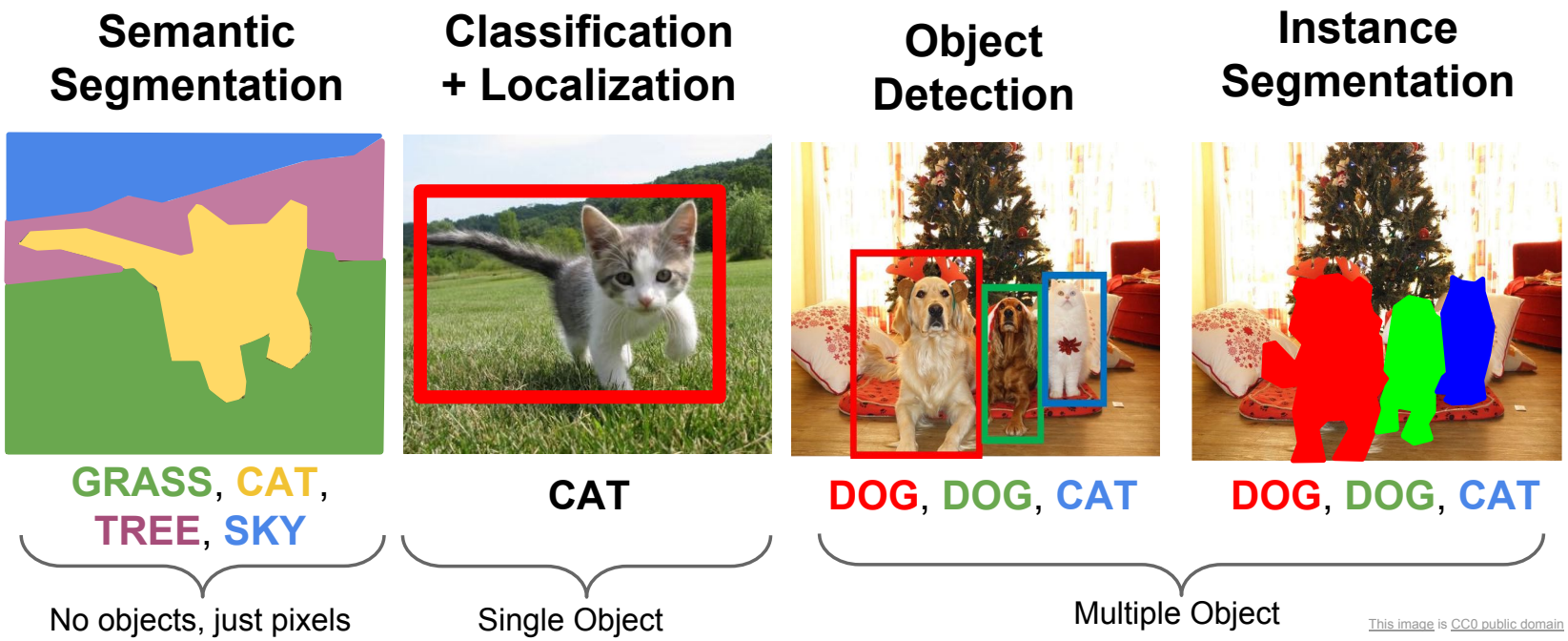
Vector:
4096

Fully-Connected:
4096 to 1000

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

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

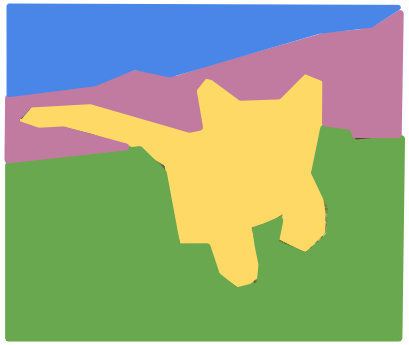
Other Computer Vision Tasks



slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Other Computer Vision Tasks

Semantic Segmentation



GRASS, CAT, TREE, SKY

No objects, just pixels

Classification + Localization

CAT

Single Object

Object Detection

DOG, DOG, CAT

Instance Segmentation

DOG, DOG, CAT

Multiple Object

This image is CC0 public domain

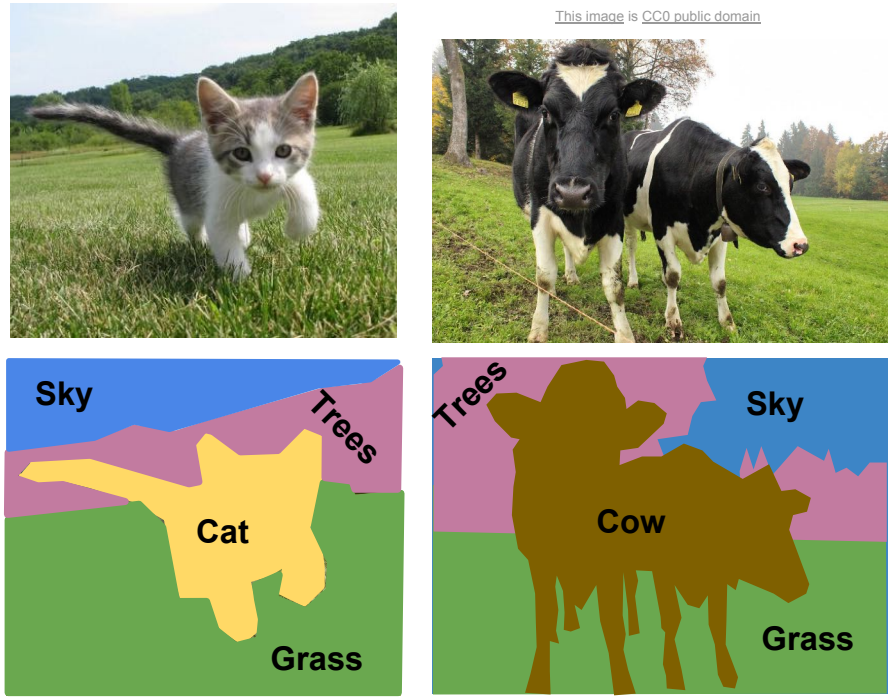
slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Semantic Segmentation

Label each pixel in the image with a category label

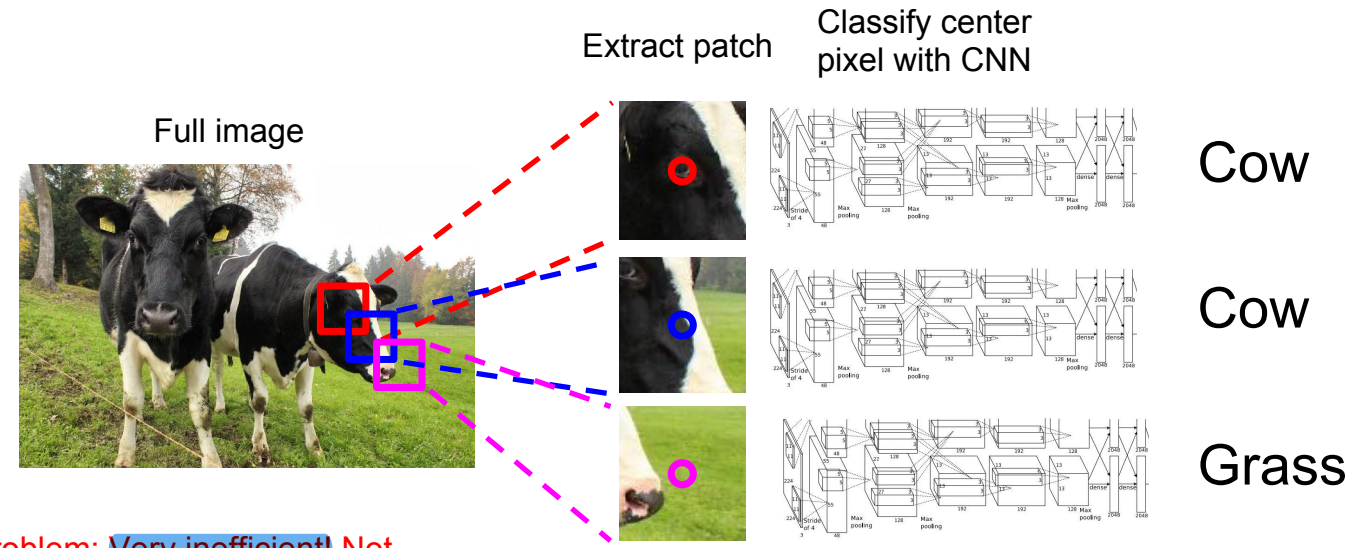
Don’t differentiate instances, only care about pixels

dont care about objects only semantic class



slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Semantic Segmentation Idea: Sliding Window

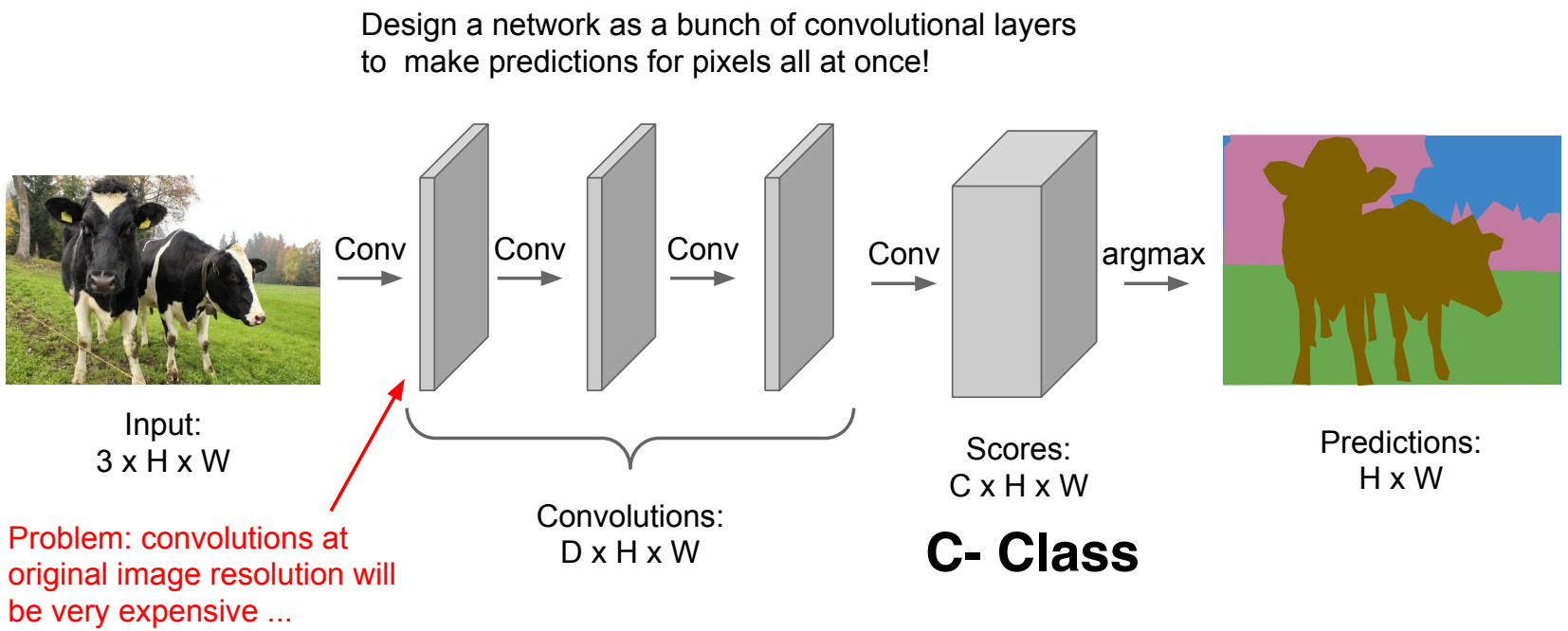


Problem: Very inefficient! Not reusing shared features between overlapping patches

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

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Semantic Segmentation Idea: Fully Convolutional

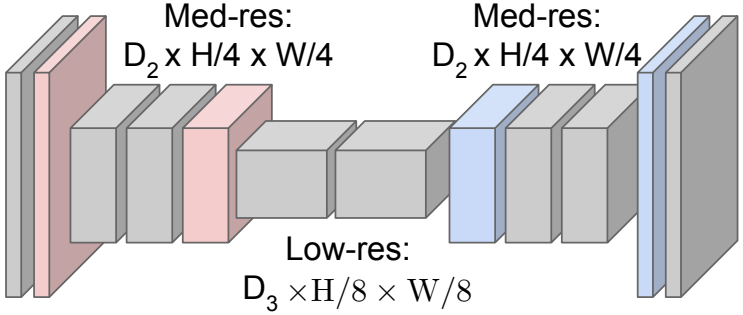


slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Semantic Segmentation Idea: Fully Convolutional

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

Input:
3 x H x W



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

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

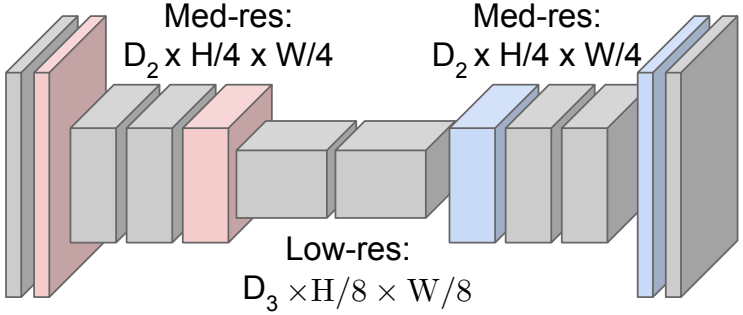
Semantic Segmentation Idea: Fully Convolutional

Downsampling:
Pooling, strided convolution

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

Upsampling:
???

Input:
3 x H x W



High-res:
D1 x H/2 x W/2

High-res:
D1 x H/2 x W/2

Predictions:
H x W

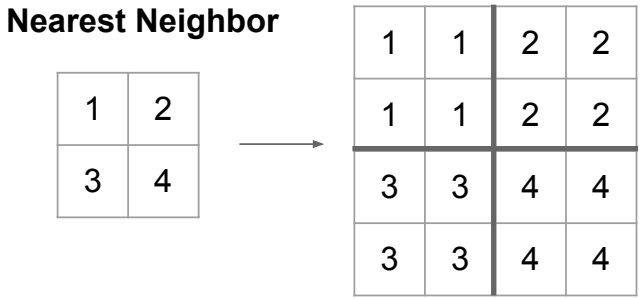
Long, Shelhamer, and Darrell, "Fully Convolutional Networks for Semantic Segmentation", CVPR 2015
Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV 2015

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

In-Network Upsampling: "Unpooling"

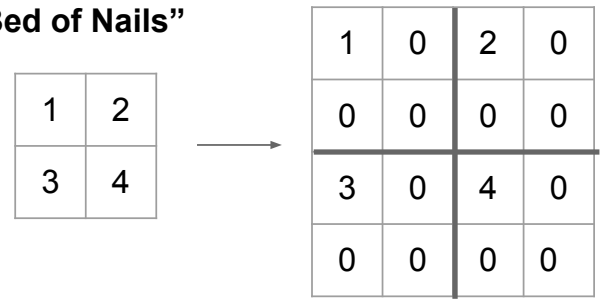
No params involved

Nearest Neighbor



Input: 2 x 2 Output: 4 x 4

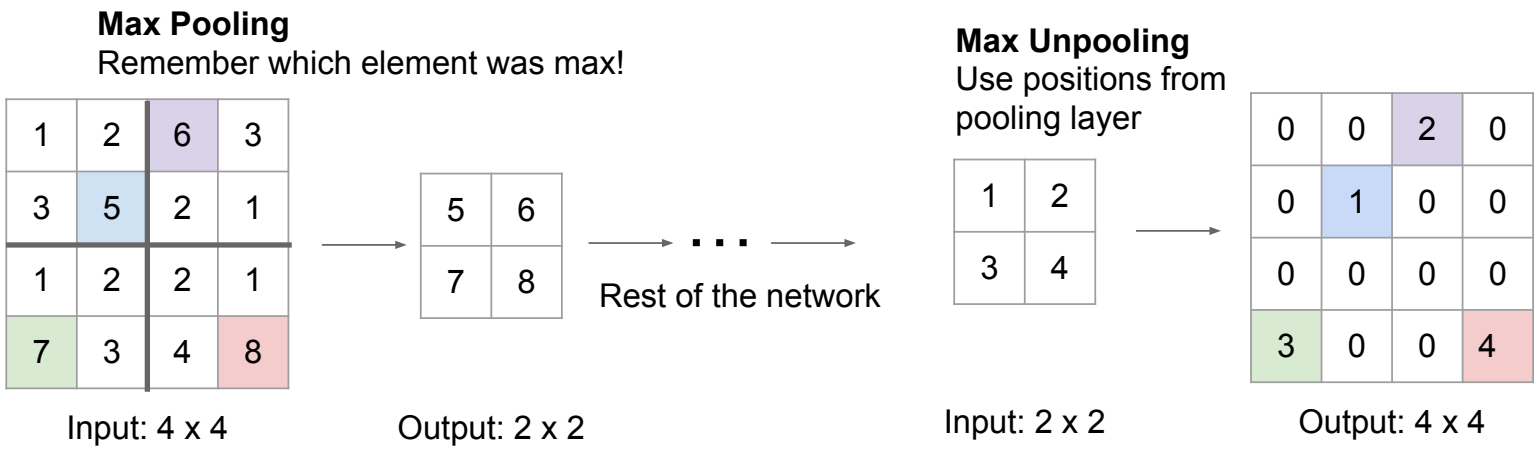
“Bed of Nails”



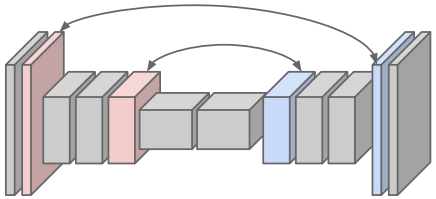
Input: 2 x 2 Output: 4 x 4

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

In-Network Upsampling: "Max Unpooling"



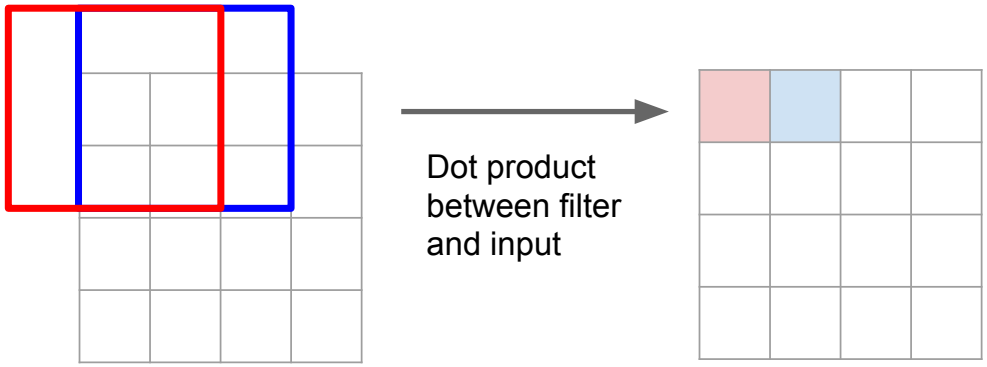
Corresponding pairs of downsampling and upsampling layers



slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Learnable Upsampling: Transpose Convolution

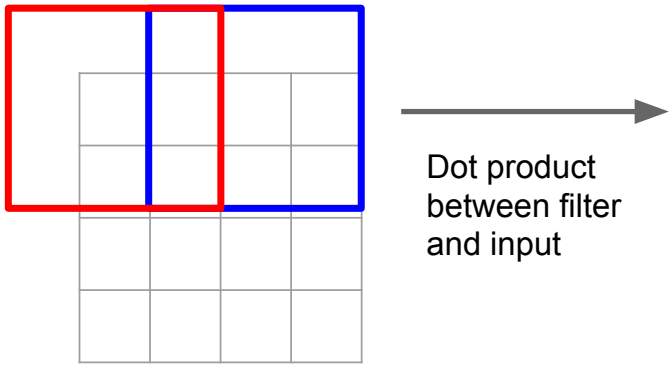
Recall: Normal 3 x 3 convolution, stride 1 pad 1



slide credit: Fei-Fei, Justin Johnson, Serena Yeung

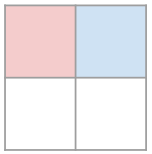
Learnable Upsampling: Transpose Convolution

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



Input: 4 x 4

Dot product between filter and input



Output: 2 x 2

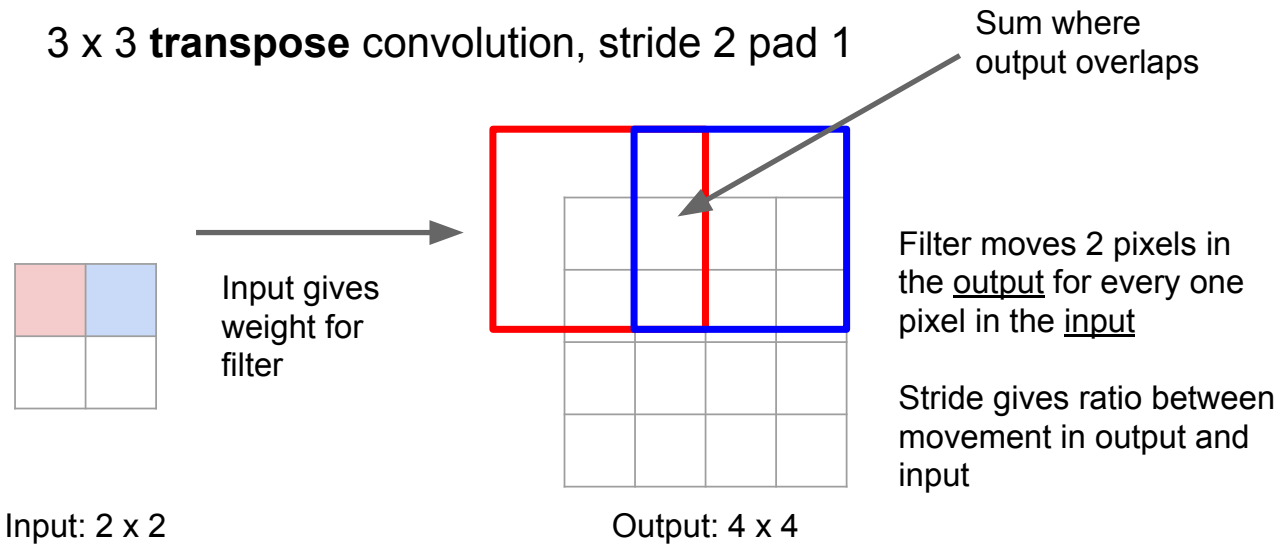
Filter moves 2 pixels in the input for every one pixel in the output

Stride gives ratio between movement in input and output

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

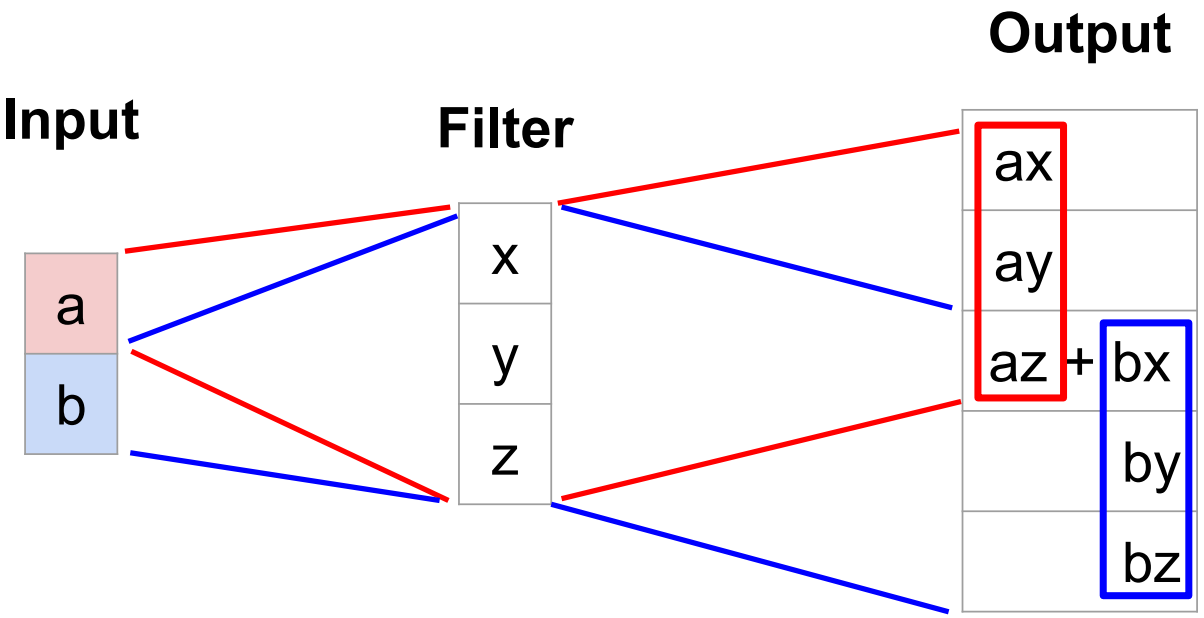
Learnable Upsampling: Transpose Convolution

Other names:
-Deconvolution (bad)
-Upconvolution
-Fractionally strided convolution
-Backward strided convolution



slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Learnable Upsampling: 1D Example



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

Need to crop one pixel from output to make output exactly 2x input

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Convolution as Matrix Multiplication (1D Example)

We can express convolution in terms of a matrix multiplication

$$\vec{x} * \vec{a} = X\vec{a}$$

$$\begin{bmatrix} x & y & z & 0 & 0 & 0 \\ 0 & x & y & z & 0 & 0 \\ 0 & 0 & x & y & z & 0 \\ 0 & 0 & 0 & x & y & z \end{bmatrix} \begin{bmatrix} 0 \\ a \\ b \\ c \\ d \\ 0 \end{bmatrix} = \begin{bmatrix} ay + bz \\ ax + by + cz \\ bx + cy + dz \\ cx + dy \end{bmatrix}$$

Example: 1D conv, kernel size=3, stride=1, padding=1

1D input (e.g. image)

1D Convolution Kernel expanded into Matrix

Convolution transpose multiplies by the transpose of the same matrix:

$$\vec{x} *^T \vec{a} = X^T \vec{a}$$

$$\begin{bmatrix} x & 0 & 0 & 0 \\ y & x & 0 & 0 \\ z & y & x & 0 \\ 0 & z & y & x \\ 0 & 0 & z & y \\ 0 & 0 & 0 & z \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} ax \\ ay + bx \\ az + by + cx \\ bz + cy + dx \\ cz + dy \\ dz \end{bmatrix}$$

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Convolution as Matrix Multiplication (1D Example)

We can express convolution in terms of a matrix multiplication

$$\vec{x} * \vec{a} = X\vec{a}$$

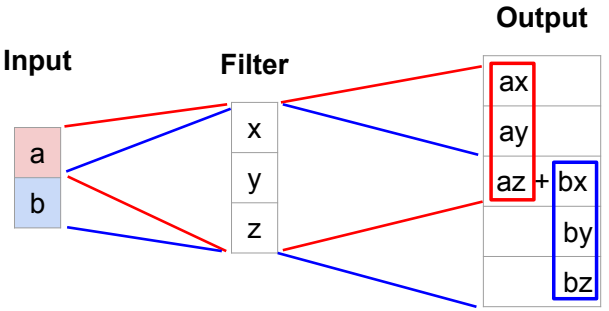
$$\begin{bmatrix} x & y & z & 0 & 0 & 0 \\ 0 & 0 & x & y & z & 0 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ b \\ c \\ d \\ 0 \end{bmatrix} = \begin{bmatrix} ay + bz \\ bx + cy + dz \end{bmatrix}$$

Example: 1D conv, kernel size=3, stride=2, padding=1

Convolution transpose multiplies by the transpose of the same matrix:

$$\vec{x} *^T \vec{a} = X^T \vec{a}$$

$$\begin{bmatrix} x & 0 \\ y & 0 \\ z & x \\ 0 & y \\ 0 & z \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} ax \\ ay \\ az + bx \\ by \\ bz \\ 0 \end{bmatrix}$$



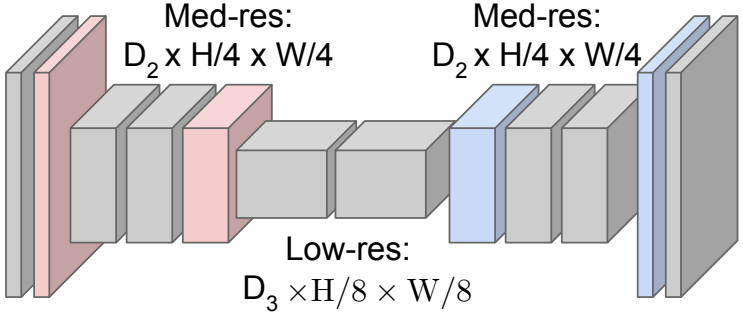
Semantic Segmentation Idea: Fully Convolutional

Downsampling:
Pooling, strided convolution

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

Upsampling:
Unpooling or strided transpose convolution

Input:
3 x H x W



High-res:
D1 x H/2 x W/2

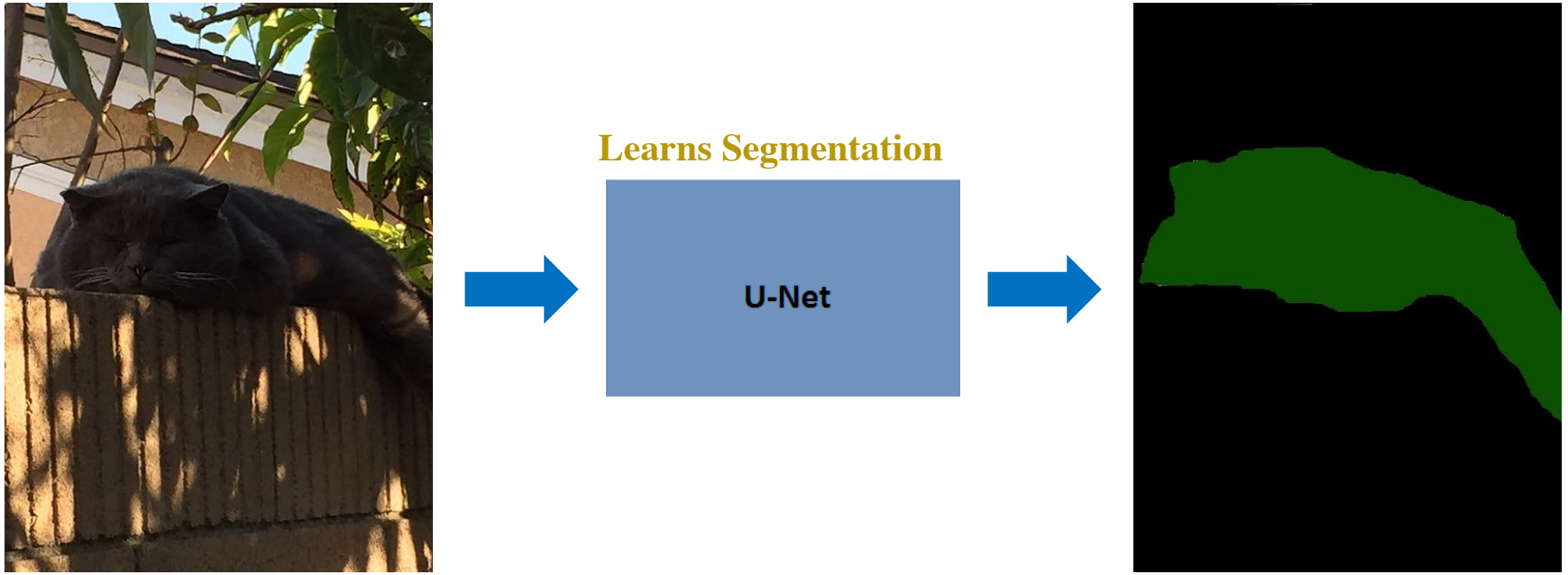
High-res:
D1 x H/2 x W/2

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

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

What does a U-Net do?

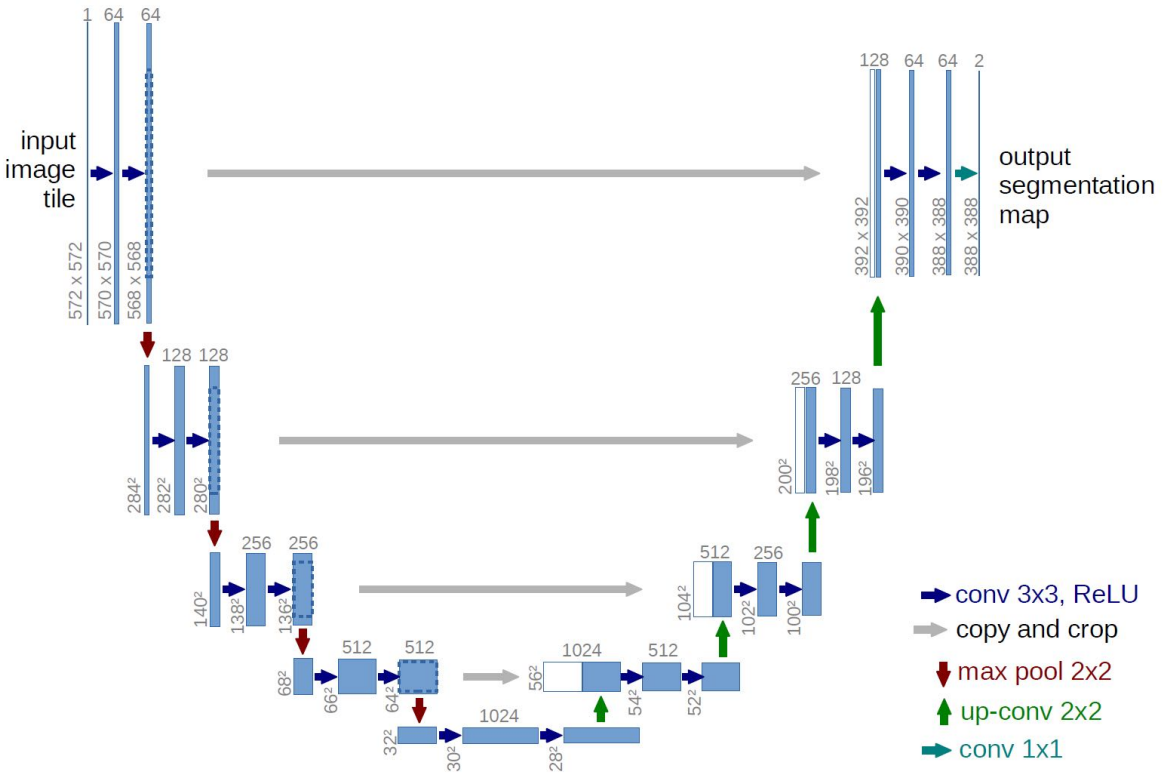


Input Image

Output Segmentation Map

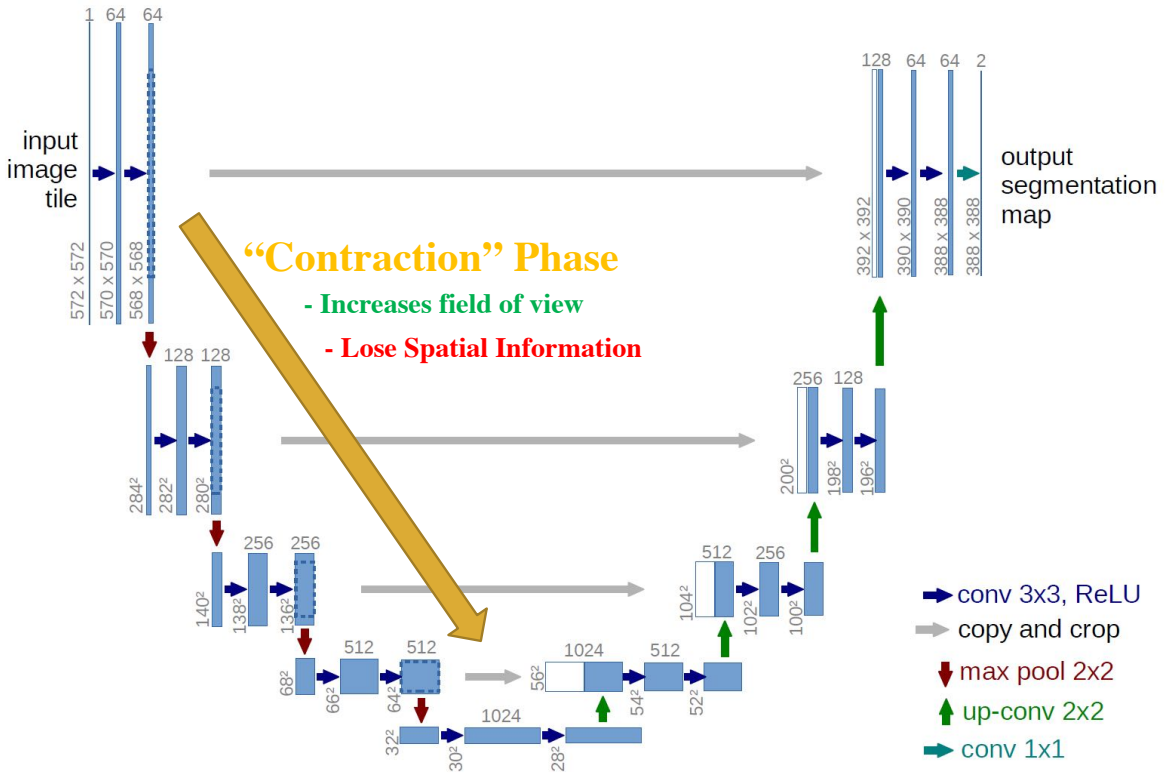
U-Net Architecture

no padding:
pa makes
things
complex



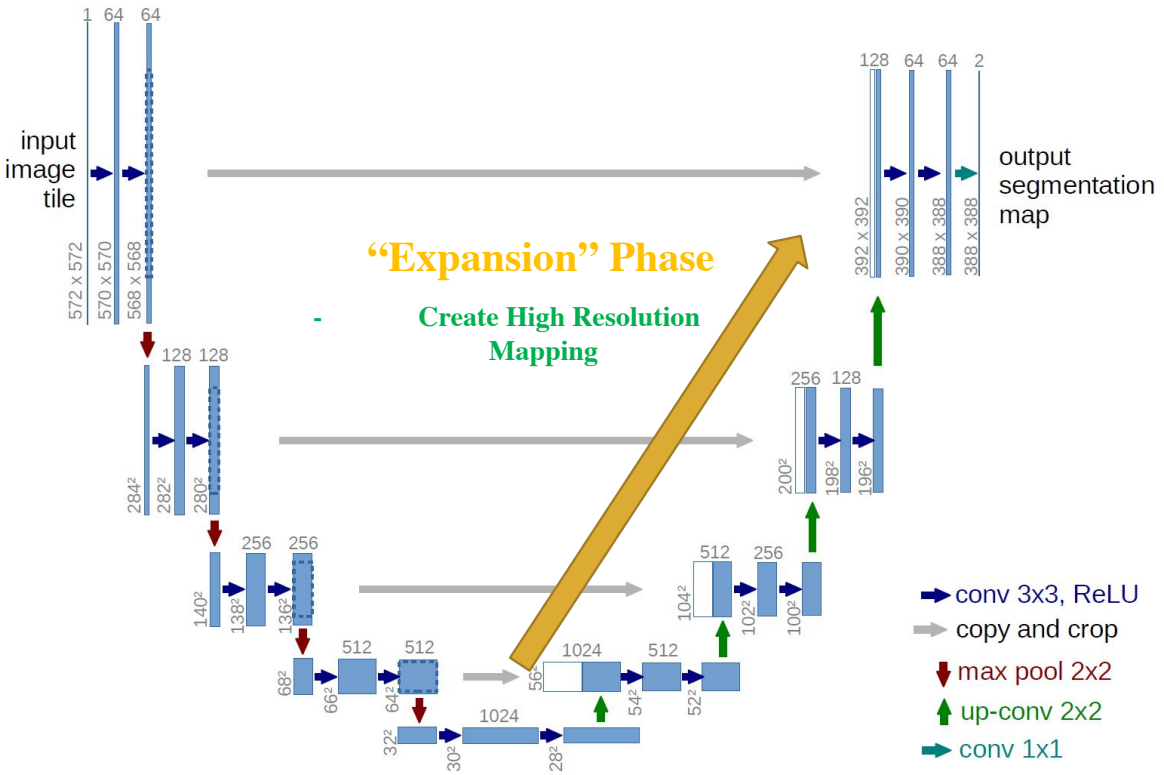
Ronneberger et al. (2015) U-net Architecture

U-Net Architecture



Ronneberger et al. (2015) U-net Architecture

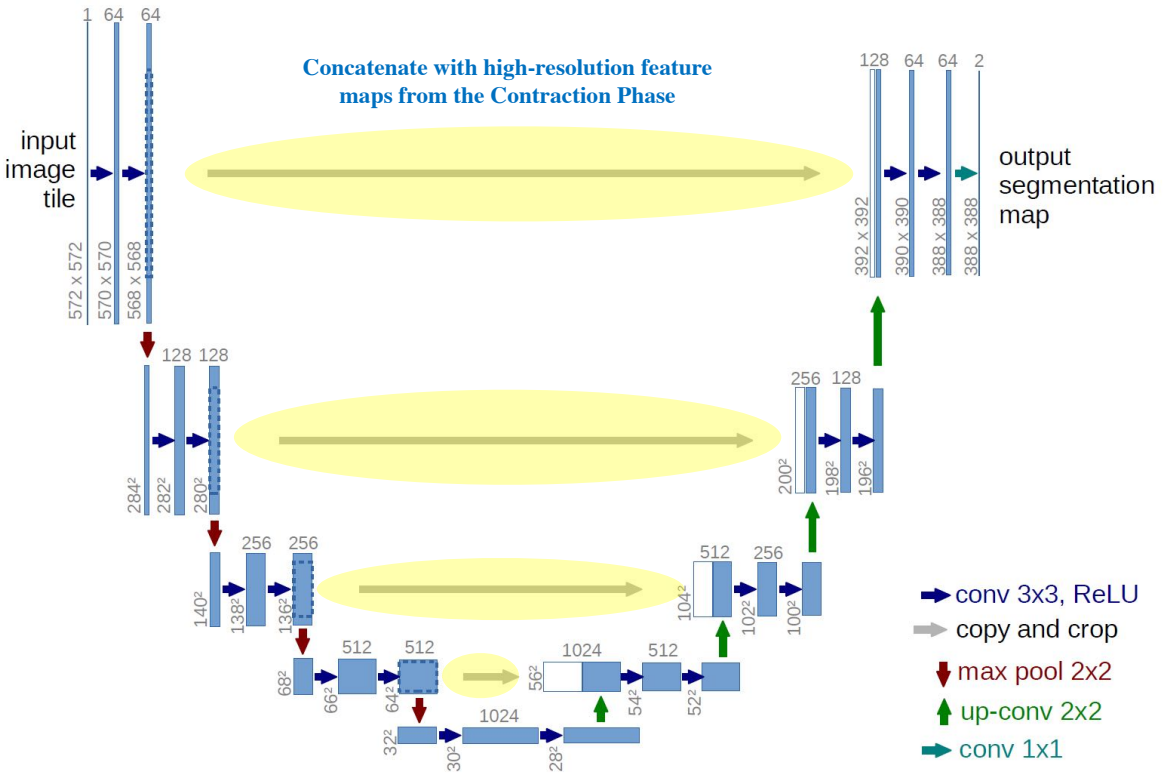
U-Net Architecture



Ronneberger et al. (2015) U-net Architecture

U-Net Architecture

**always center cropping,
cuz other parts are lost**

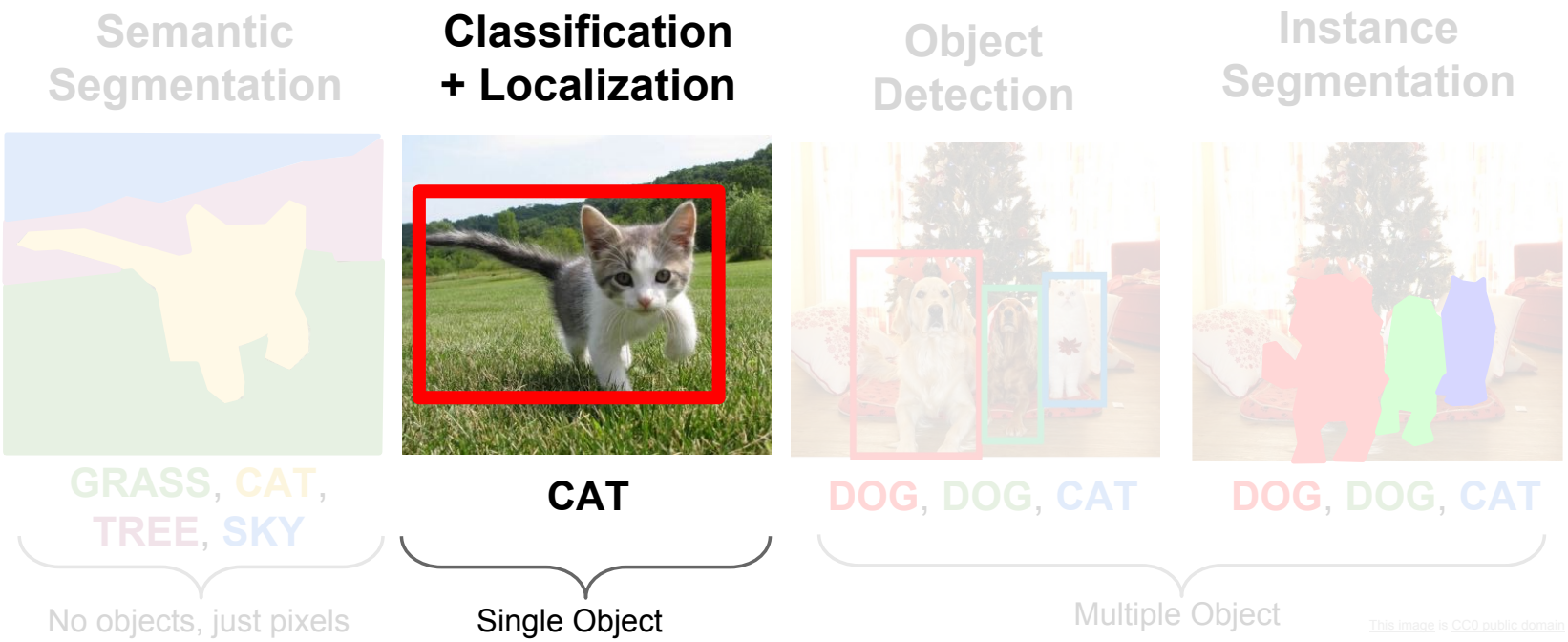


Ronneberger et al. (2015) U-net Architecture

U-Net Summary

- Contraction Phase
  - Reduce spatial dimension, but increases the “what.”
- Expansion Phase
  - Recovers object details and the dimensions, which is the “where.”
- Concatenating feature maps from the Contraction phase helps the Expansion phase with recovering the “where” information.

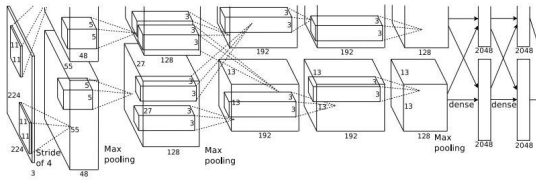
Other Computer Vision Tasks



slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Classification + Localization

This image is CC0 public domain



Fully Connected: 4096 to 1000

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

Softmax Loss

Correct label:
Cat

Multitask Loss

+ → **Loss**

Vector: 4096

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

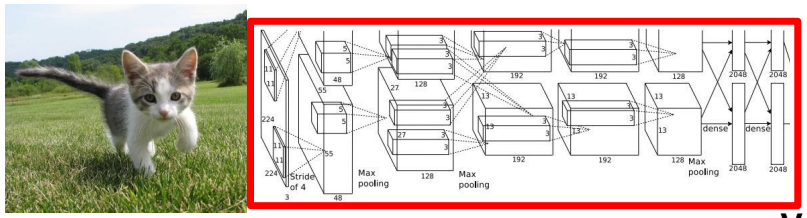
L2 Loss

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

Treat localization as a regression problem!

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Classification + Localization



This image is CC0 public domain

Often pretrained on ImageNet
(Transfer learning)

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')

Loss

Treat localization as a regression problem!

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Other Computer Vision Tasks

Semantic Segmentation

GRASS, CAT, TREE, SKY

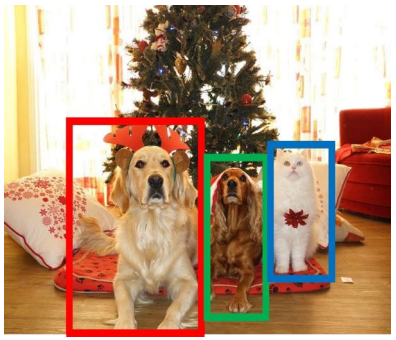
No objects, just pixels

Classification + Localization

CAT

Single Object

Object Detection



DOG, DOG, CAT

Instance Segmentation

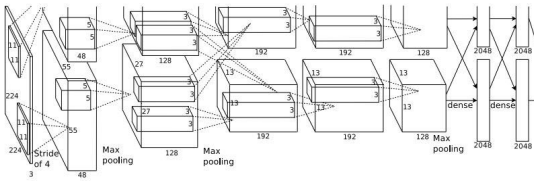
DOG, DOG, CAT

Multiple Object

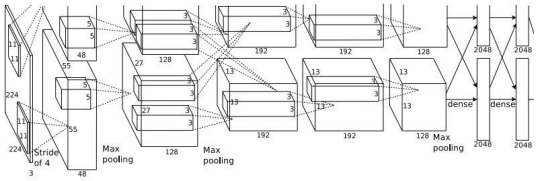
This image is CC0 public domain

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

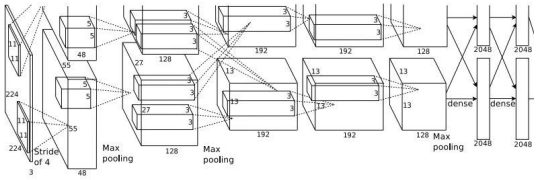
Object Detection as Regression?



CAT: (x, y, w, h)



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

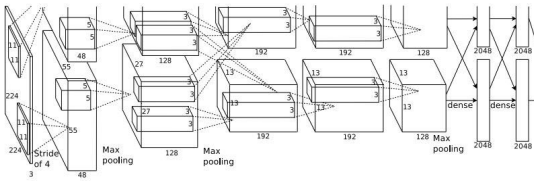


DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

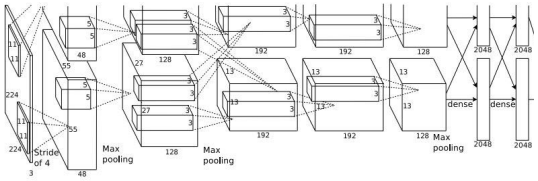
slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Object Detection as Regression?

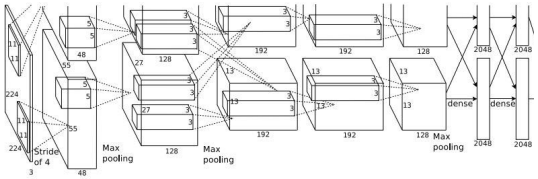
Each image needs a different number of outputs!



CAT: (x, y, w, h) 4 numbers



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

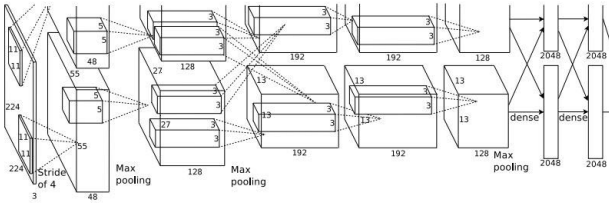


DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....
Many numbers!

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Object Detection as Classification: Sliding Window

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

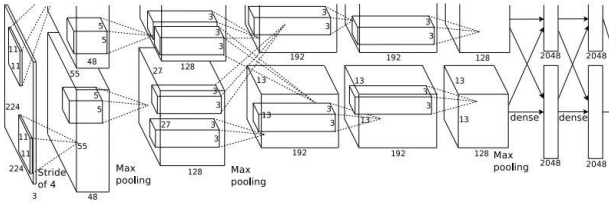


Dog? NO
Cat? NO
Background? YES

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Object Detection as Classification: Sliding Window

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

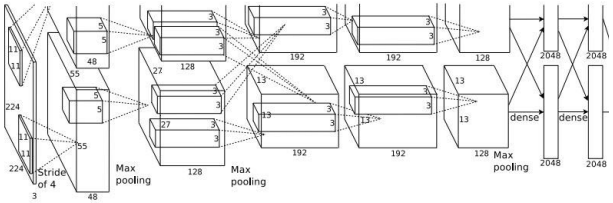


Dog? YES
Cat? NO
Background? NO

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Object Detection as Classification: Sliding Window

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

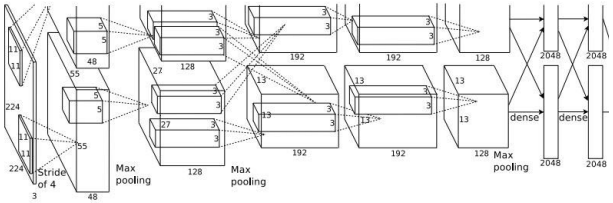


Dog? YES
Cat? NO
Background? NO

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Object Detection as Classification: Sliding Window

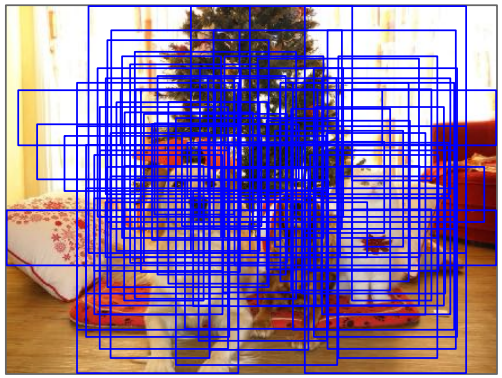
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



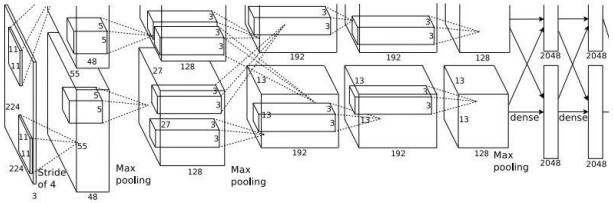
Dog? NO
Cat? YES
Background? NO

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Object Detection as Classification: Sliding Window



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



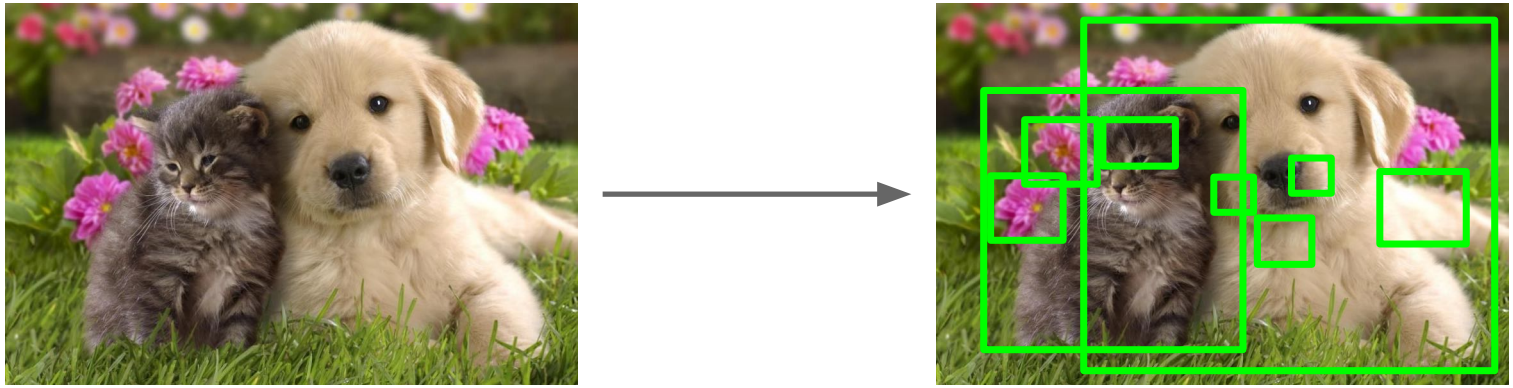
Problem: Need to apply CNN to huge number of locations, scales, and aspect ratios, very computationally expensive!

Dog? NO
Cat? YES
Background? NO

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Region Proposals / e.g. 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

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Region Proposal Step

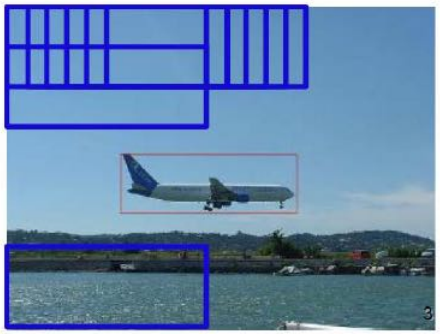
Segmentation as Selective Search for Object Recognition

K. van de Sande[1], J. Uijlings[2], T. Gevers[1], and A. Smeulders[1]
University of Amsterdam[1] and University of Trento[2]

Selective Search: Motivation

Viola IJCV 2004
Dalal CVPR 2005
Felzenszwalb TPAMI 2010
Vedaldi ICCV 2009

- Many approaches (at the time) use exhaustive search:
  - visit **every** location in an image
  - problem: computationally expensive:
    - number of possible locations should be small
      -> number of grid locations & aspect ratio(s) need to be small
    - evaluation cost per location should be low
      -> simple features / classifiers



- to go beyond this - we should aim for something more “sophisticated”

Selective Search: Main Design Criteria

- **High recall**
  - We do not want to lose any objects, since they cannot be recovered later.
- Coarse locations are sufficient
  - Accurate delineation is not necessary for recognition
  - In contrary, nearby context might be useful
    -> use bounding boxes
- Fast to compute
  - Necessary when operating with large datasets
    -> <10s/image

Selective Search: How to Obtain High Recall?

- Images are intrinsically hierarchical

- Segmentation at single scale are not enough
  -> hypotheses based on hierarchical grouping

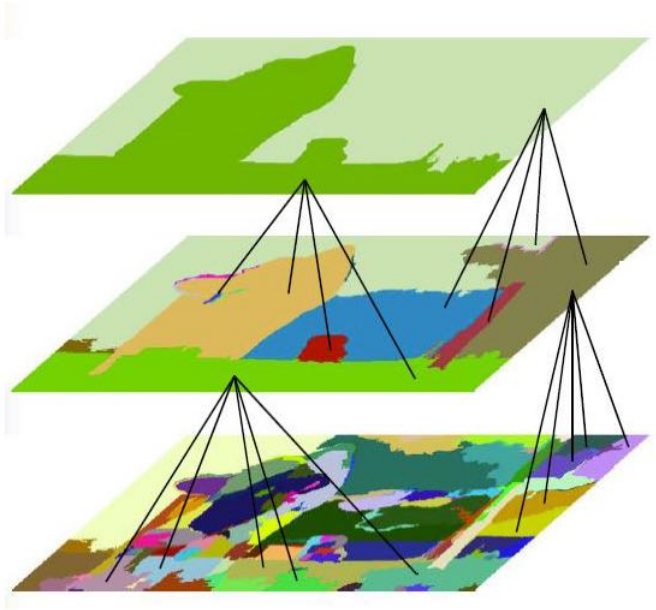
Selective Search: Method

- Start by oversegmenting the input image

"Efficient graph-based image segmentation"
Felzenszwalb and Huttenlocher, IJCV 2004

Selective Search: Method

1. Merge two most similar regions based on *S*.
2. Update similarities between the new region and its neighbors.
3. Go back to step 1. until the whole image is a single region.



Selective Search: Method

- compute similarity measure between all adjacent region pairs a and b (e.g.) as:

$$S(a,b) = \alpha S_{size}(a,b) + \beta S_{color}(a,b)$$

  - with

$$S_{size}(a,b) = 1 - \frac{\text{size}(a) + \text{size}(b)}{\text{size}(image)}$$

    encourages small regions to merge early

  - and

$$S_{color}(a,b) = \sum_{k=1}^{n} \min(a^k, b^k)$$

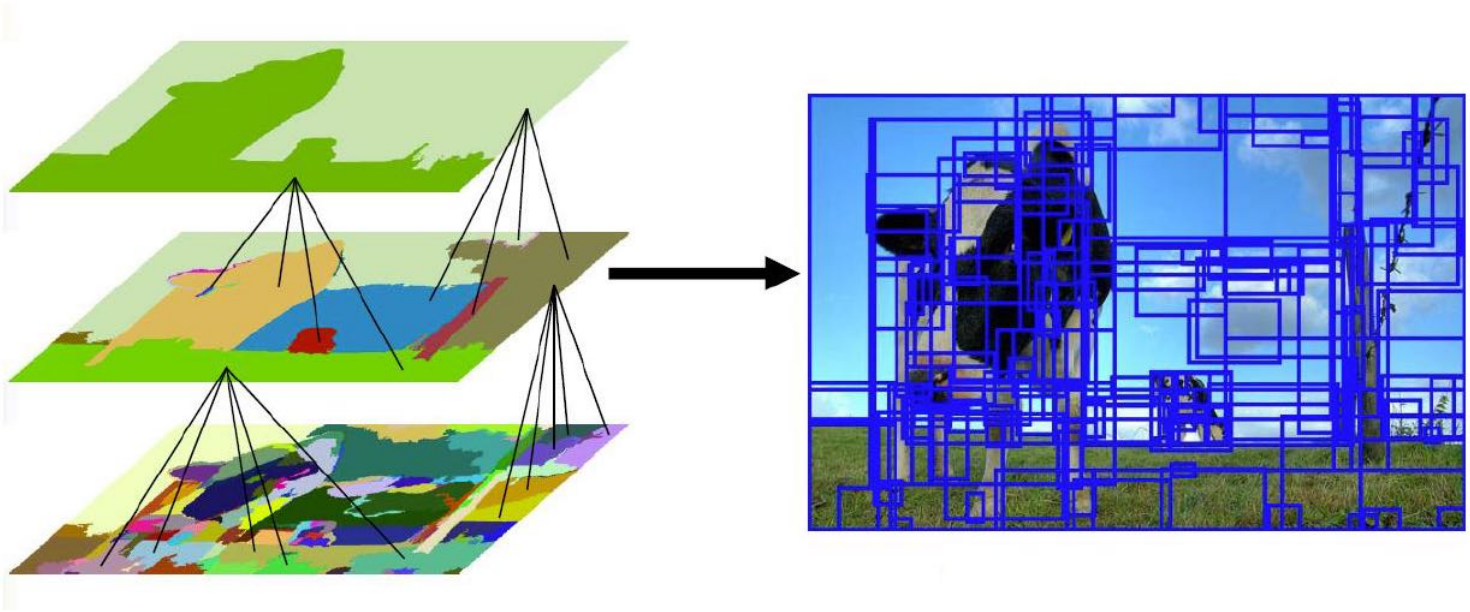
    a^k, b^k are color histograms, encouraging “similar (color)” regions to merge

  - for slightly more elaborated similarities see their IJCV-paper

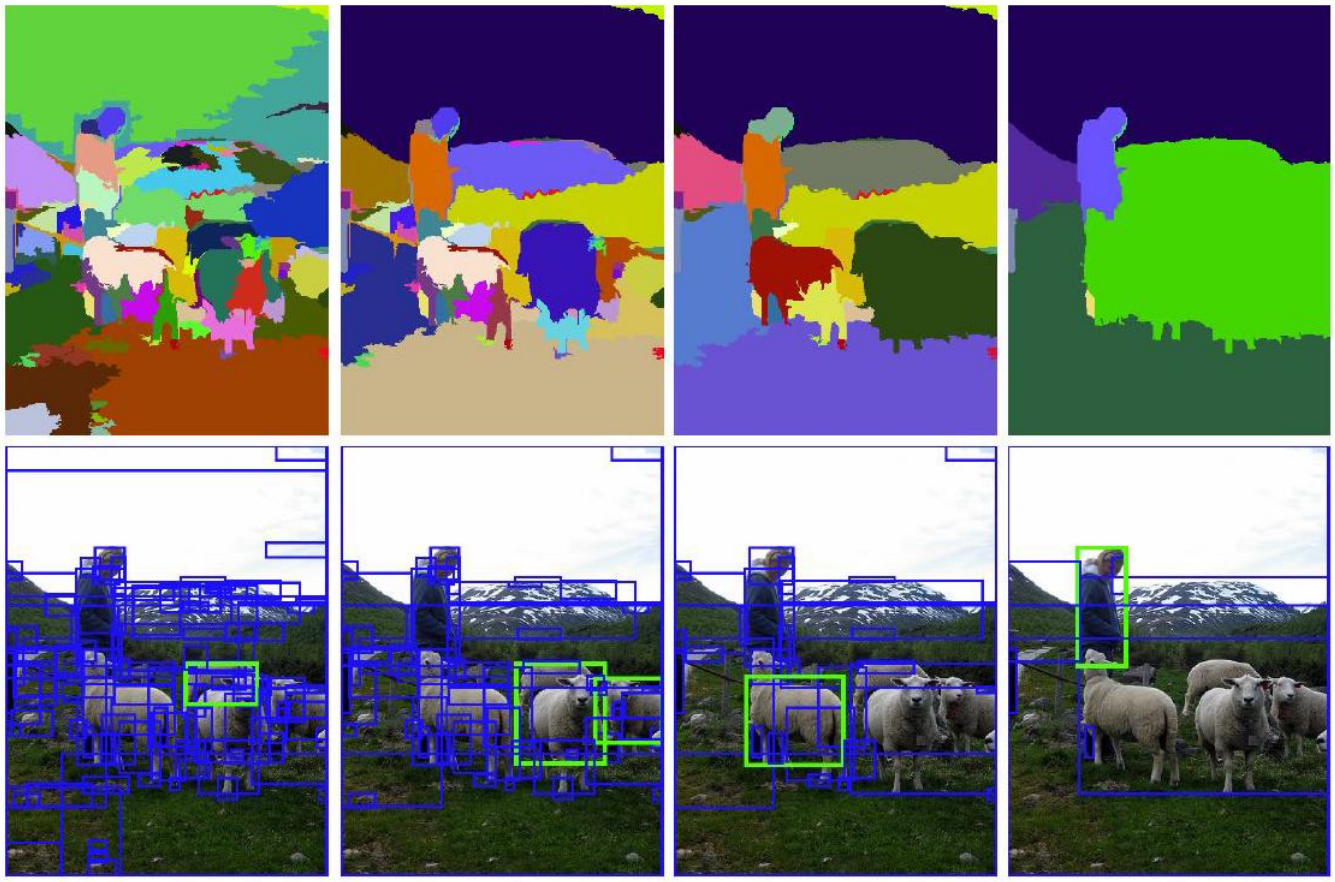
how many similar colors are there in two bins

Selective Search: Method

- Take bounding boxes of all generated regions and treat them as possible object locations.



Selective Search: Method

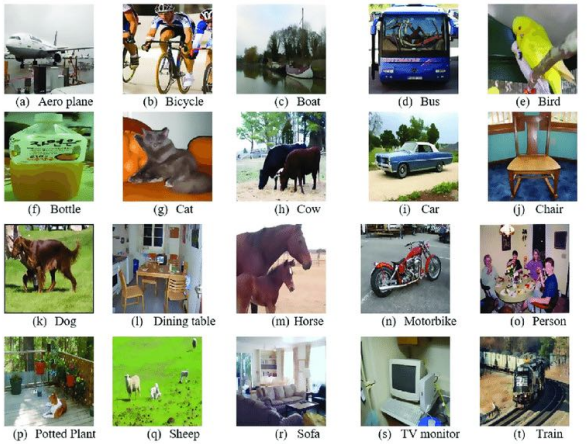


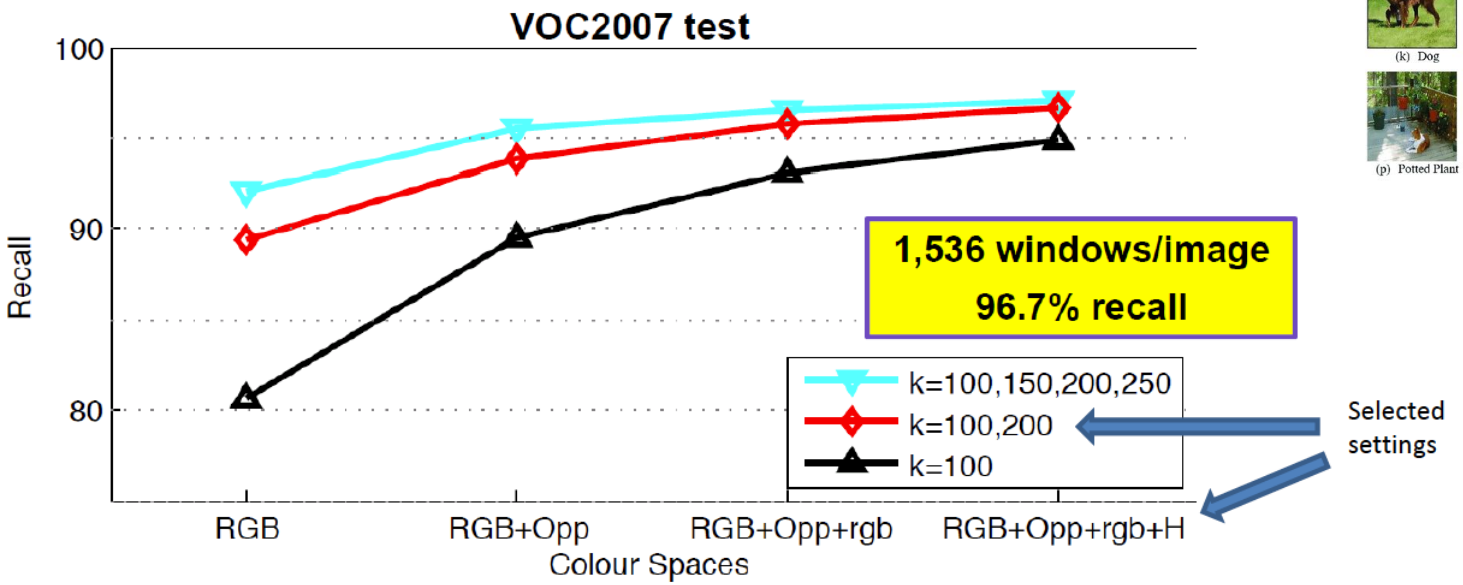
Selective Search: High Recall Revisited

- No single segmentation works for all cases
  -> diversify the set of segmentations
- Use different color spaces
  - RGB, Opponent color, normalized RGB, and hue
- Use different parameters in Felzenswalb method
  - k = [100, 150, 200, 250] (k = threshold parameter)

Selective Search: Evaluation of Object Hypotheses

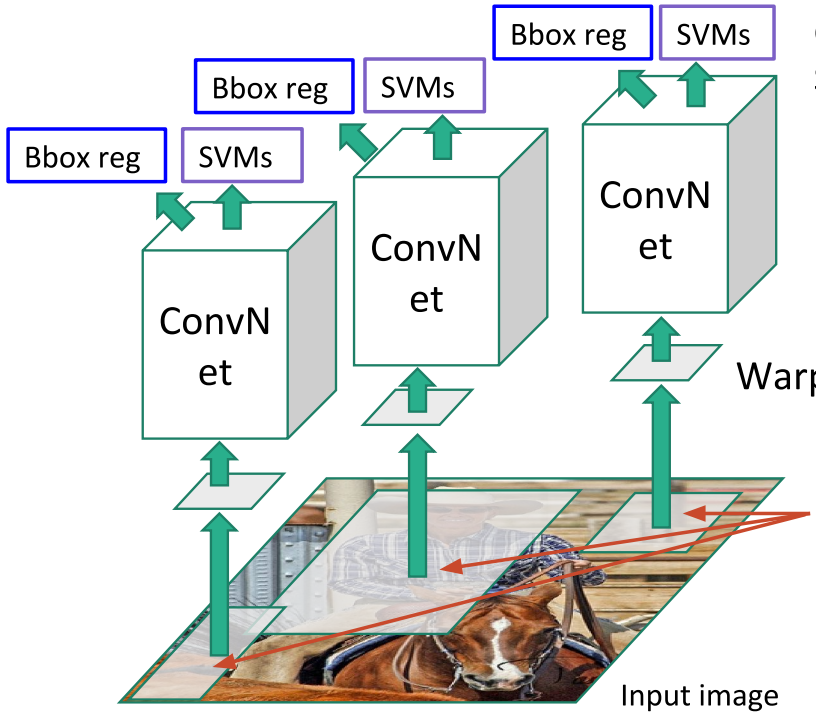
- Recall is a proportion of objects that are covered by some box with >0.5 overlap





R-CNN — Region Based CNN

tend to crop a bit larger: can have more context info



Linear Regression for bounding box offsets

Classify regions with SVMs

Forward each region through ConvNet

Warped image regions

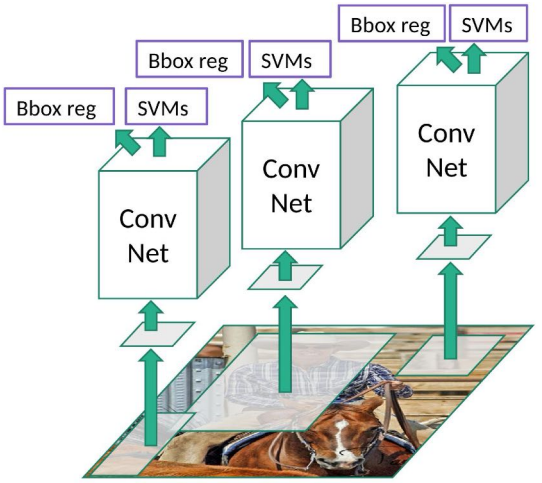
Regions of Interest (RoI) from a proposal method (~2k)

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.

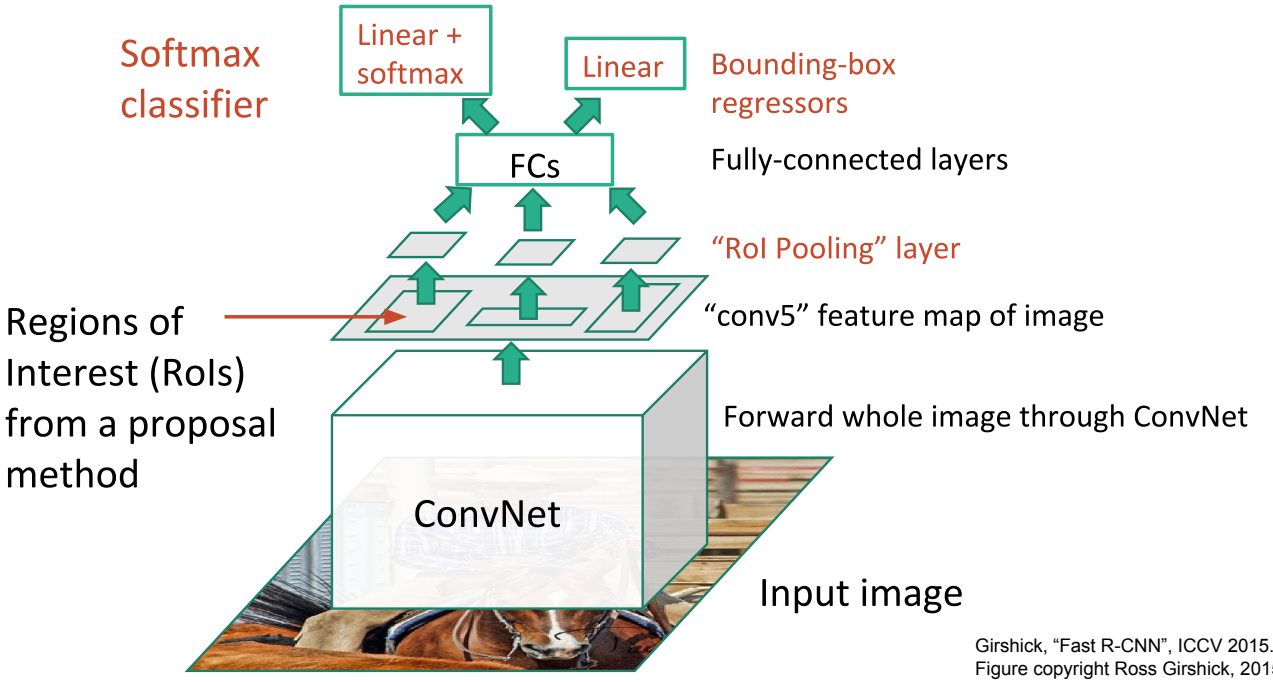
R-CNN — Region Based CNN: Problems

- 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
- Inference (detection) is slow
  - 47s / image with VGG16 [Simonyan & Zisserman. ICLR15]
  - Fixed by SPP-net [He et al. ECCV14]



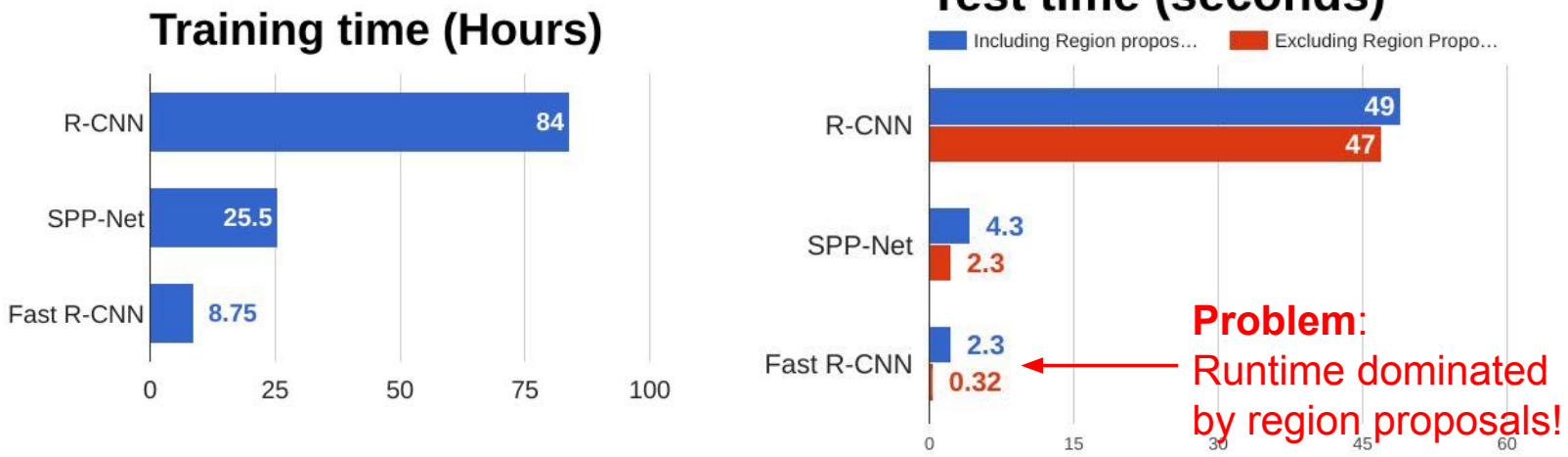
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Slide copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN



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

R-CNN vs. SPP vs. Fast-RCNN



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

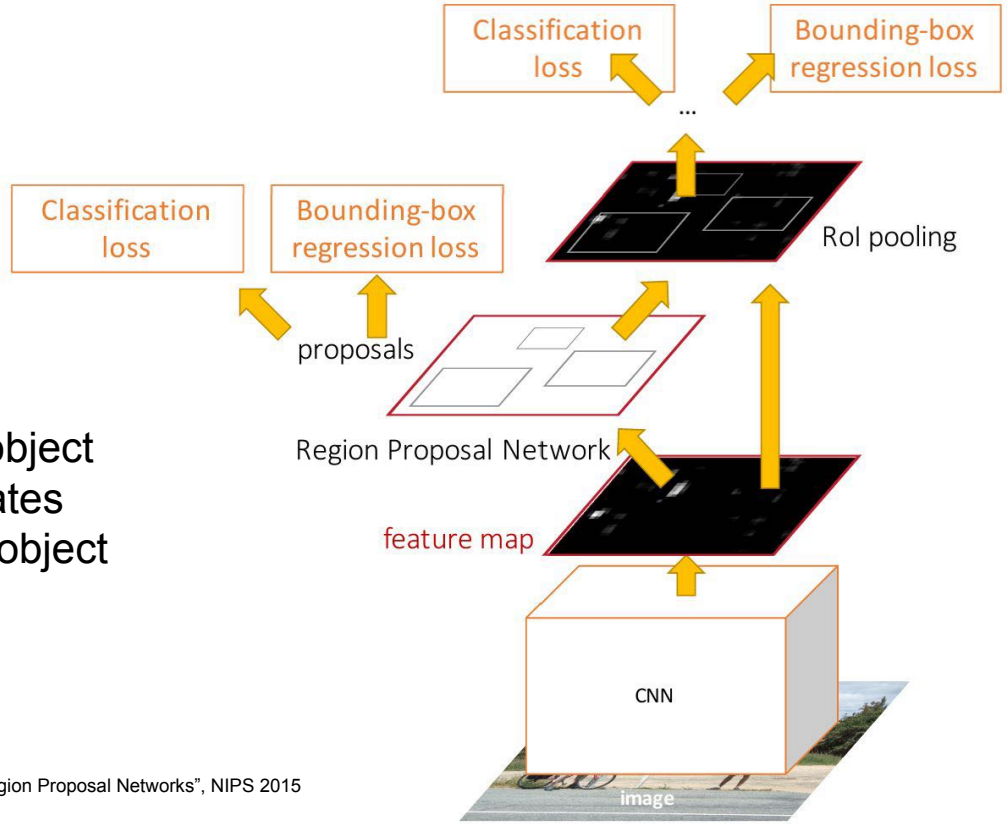
slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Faster - R-CNN: Make CNN do Proposals also !

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

Jointly train with 4 losses:

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

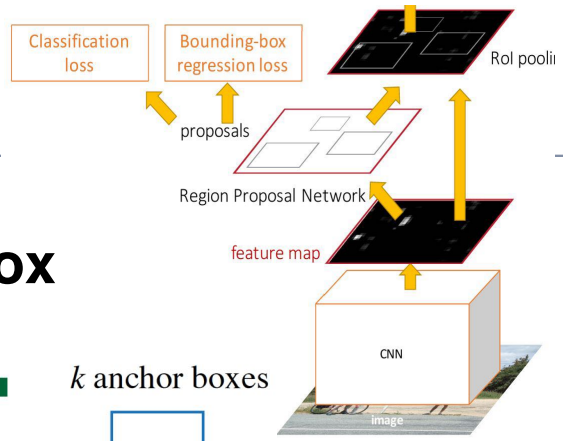


Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

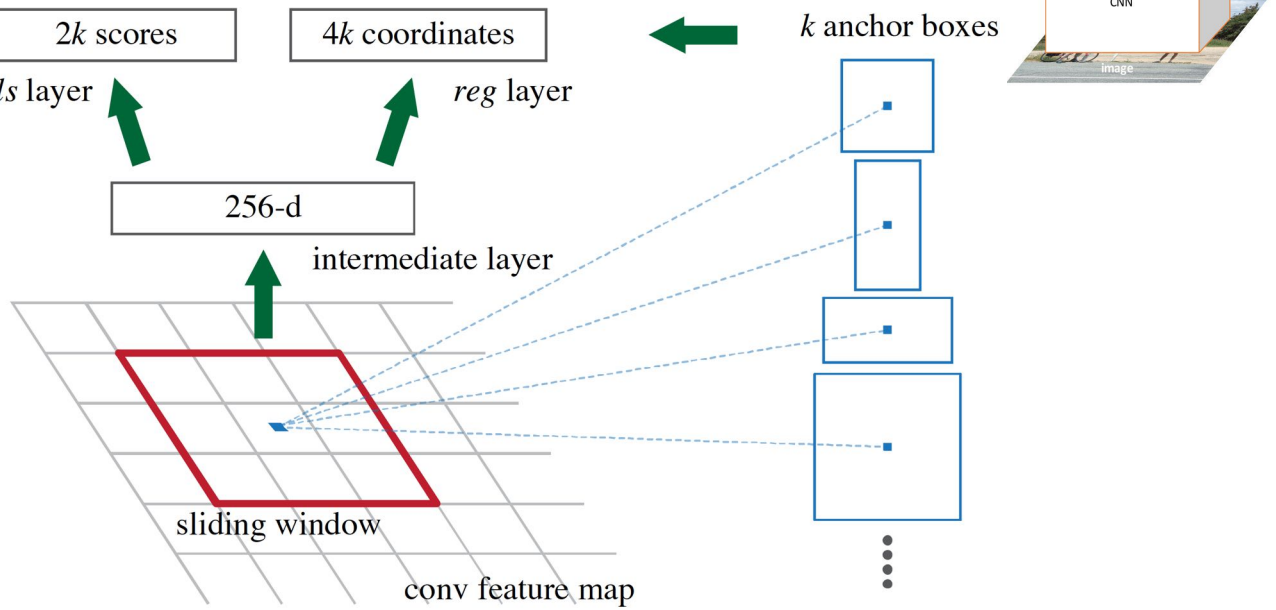
Region Proposal Network (RPN)

- components
  - 3x3 sliding window
  - 256-dimensional vector for each location (convolutions)
  - k anchor boxes, e.g. k=9:
    - 3 scales x 3 aspect ratios
  - for each box
    - class score (here 2-class softmax) for (any) object present or not
    - 4 coordinates for bounding box

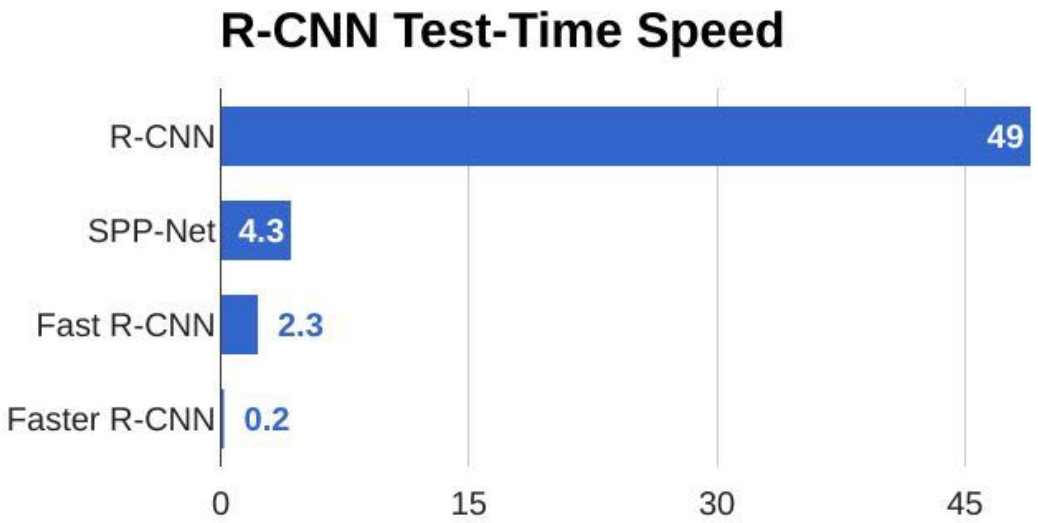


for objects: yes or no

for bounding box

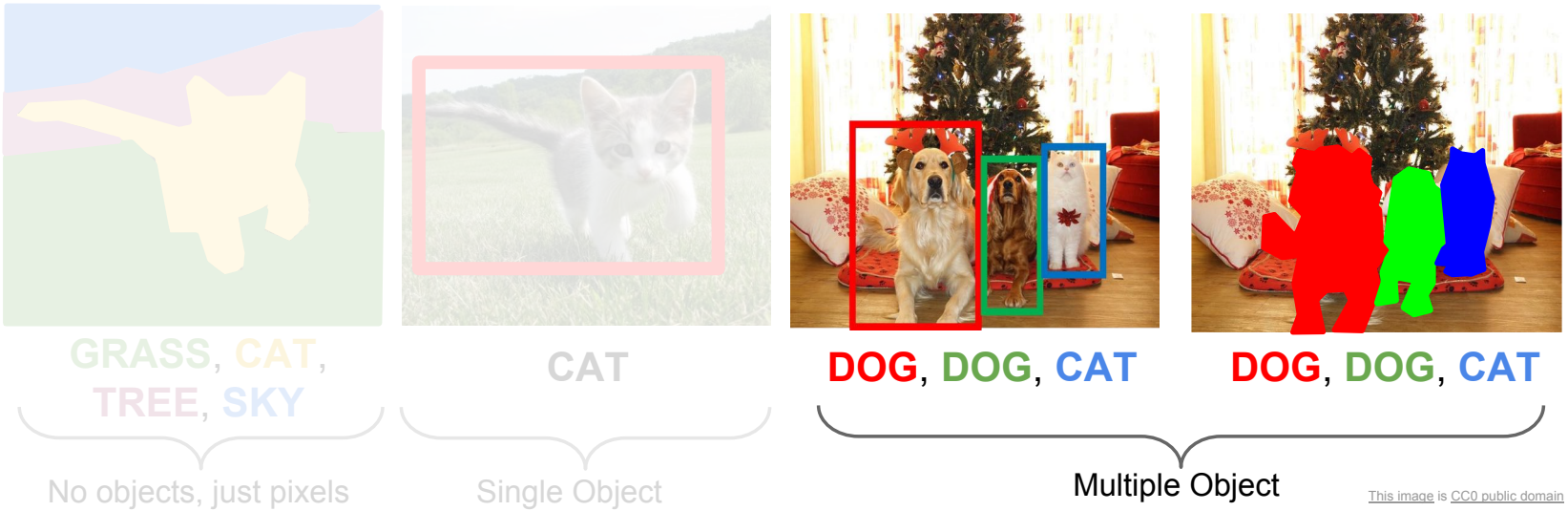


Faster - R-CNN



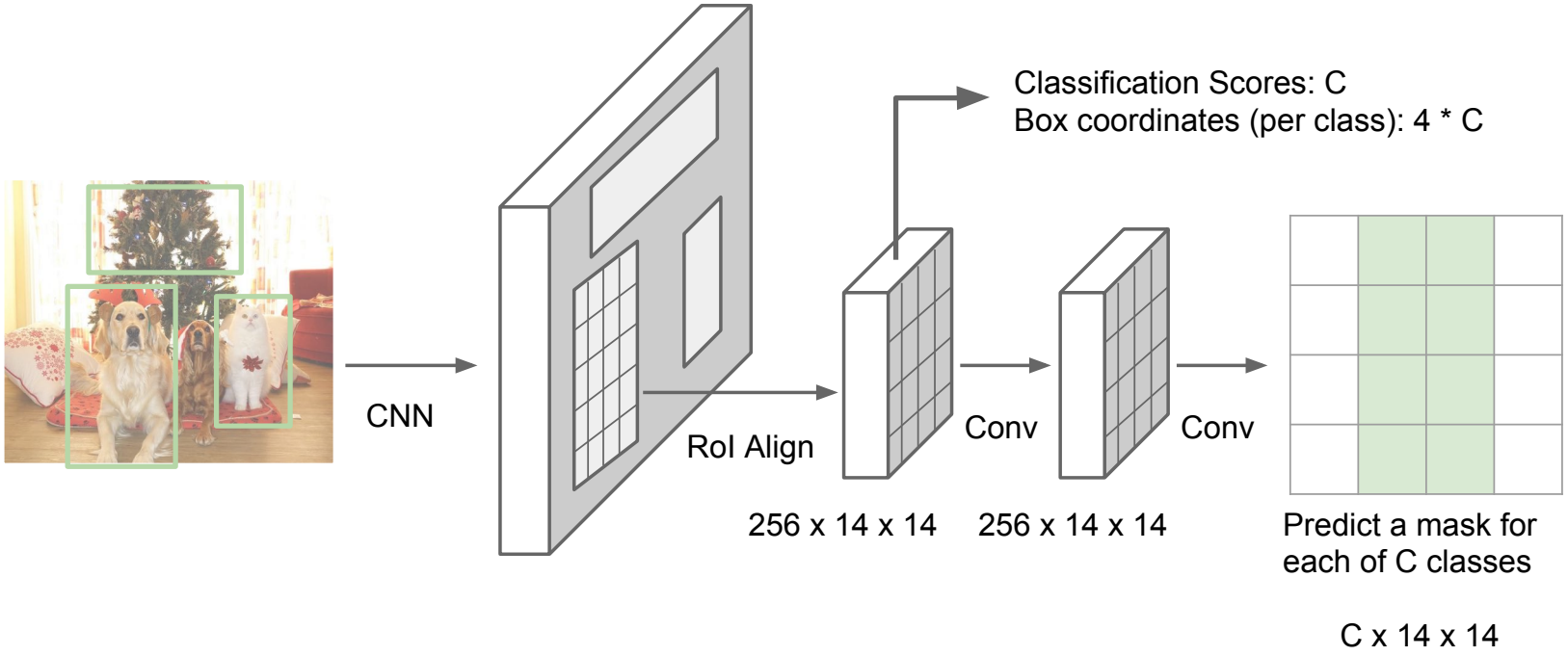
slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Object Detection vs. Instance Segmentation



slide credit: Fei-Fei, Justin Johnson, Serena Yeung

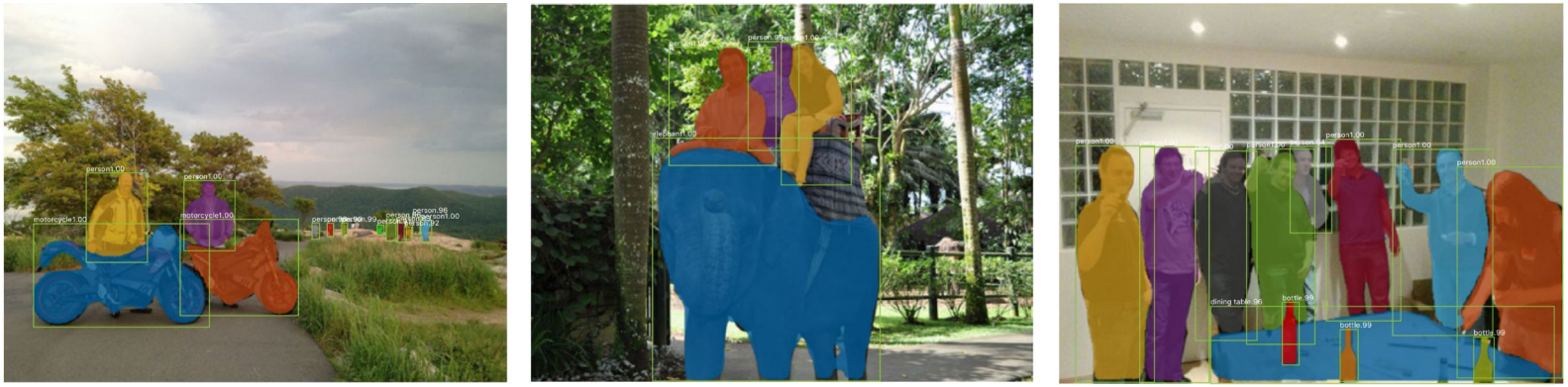
Mask R-CNN = Faster R-CNN + Segmentation Output for each ROI



He et al, “Mask R-CNN”, arXiv 2017

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Mask R-CNN: Very Good Results



He et al, “Mask R-CNN”, arXiv 2017
Figures copyright Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, 2017.
Reproduced with permission.

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Mask R-CNN: Also does Pose

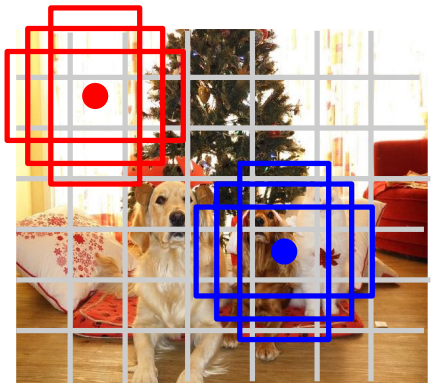
He et al, “Mask R-CNN”, arXiv 2017
Figures copyright Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, 2017.
Reproduced with permission.

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

you only look once
Detection without Proposals: YOLO (alternative: SSD)

Go from input image to tensor of scores with one big convolutional network!

Input image
3 x H x W



Divide image into grid
7 x 7

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

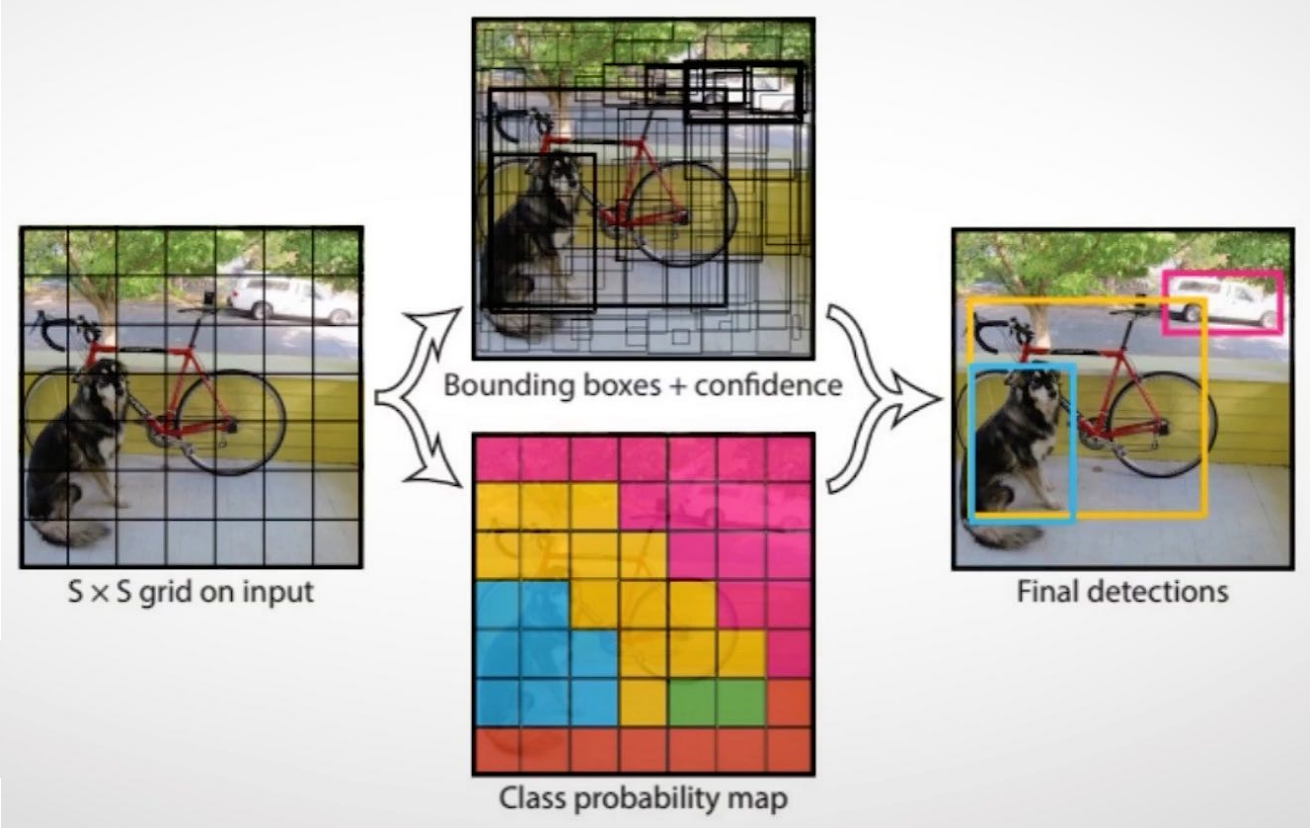
Within each grid cell:
- Regress from each of the B base boxes to a final box with 5 numbers: (dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)

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

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

slide credit: Fei-Fei, Justin Johnson, Serena Yeung

Detection without Proposals: YOLO



slide credit: https://youtu.be/YmMZkCstui0

YOLO Family

YOLO
CVPR 2016

YOLO v2 / YOLO9000
CVPR 2017

Anchor boxes, Batch normalization, …
Darknet-19
Joint training: both detection and classification

Darknet-53

YOLO v3
arXiv 2016

YOLO v5
GitHub 2020

YOLO v4
arXiv 2020

YOLOv4 consists of:

- Backbone: CSPDarknet53 [81]
- Neck: SPP [25], PAN [49]
- Head: YOLOv3 [63]

Object Detection: Impact of Deep Learning

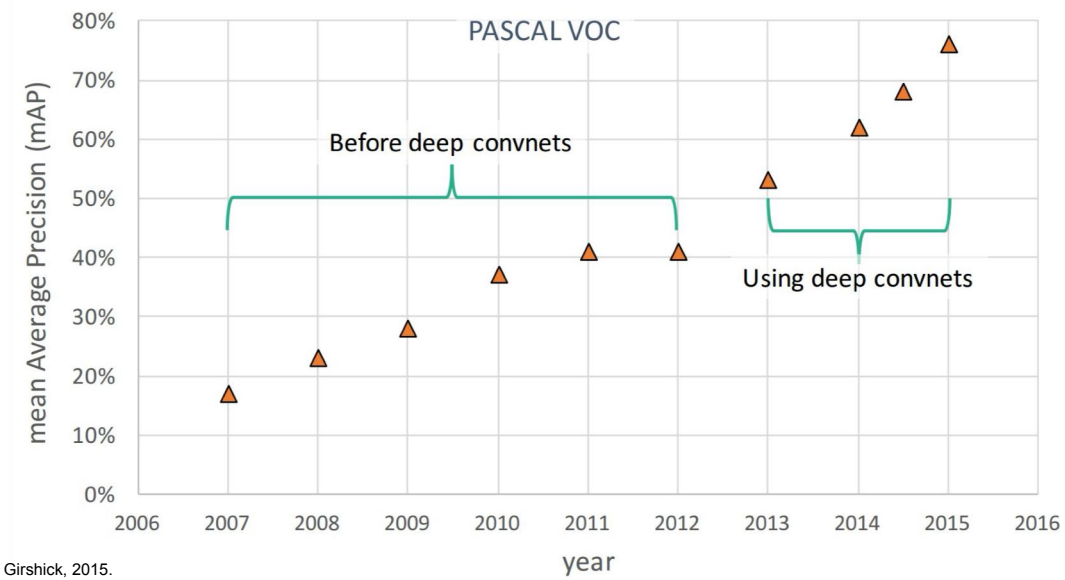


Figure copyright Ross Girshick, 2015.
Reproduced with permission.

slide credit: Fei-Fei, Justin Johnson, Serena Yeung