Final Projects

Final Projects - Dates

- Project Proposals:
  - Project proposal due date: December 20th 11.59 p.m.
  - Project proposal feedback date: December 21th 11.59 p.m.
  - Starting date: December 22th
- Poster presentation and due date: February 6th

Projects with no proposal won't be graded!

Replace with your project title

Team Member 1
first@i1.org

Team Member 2
second@i1.org

Team Member 3
third@i1.org

Team Member 4
fourth@i1.org

Project Proposal

The following bullet points and remarks are meant to guide through the process of writing this proposal. Nevertheless we expect a coherent text.

1. Introduction

Explain your general idea and state the problem you are trying to solve.

1.1. Related Works

- Related and previous work on your topic
- A small overview of the SOTA (state-of-the-art)
- What is new/different in your approach?
- ...

2. Dataset

- Are you working with an existing dataset or is data collection part of your project?
- Transfer learning and training from scratch
- Resource management (please consider GPU memory)
- ...

4. Outcome

What is your desired outcome or aspiration for the project?

Final Project – Project Proposals

- Send your project proposals to this email address: dl4cv-dropbox.vision.in@tum.de
- Remember to add all members of the team in the CC as well as your team ID

- Deadline: **December 20th 11.59 p.m.!**

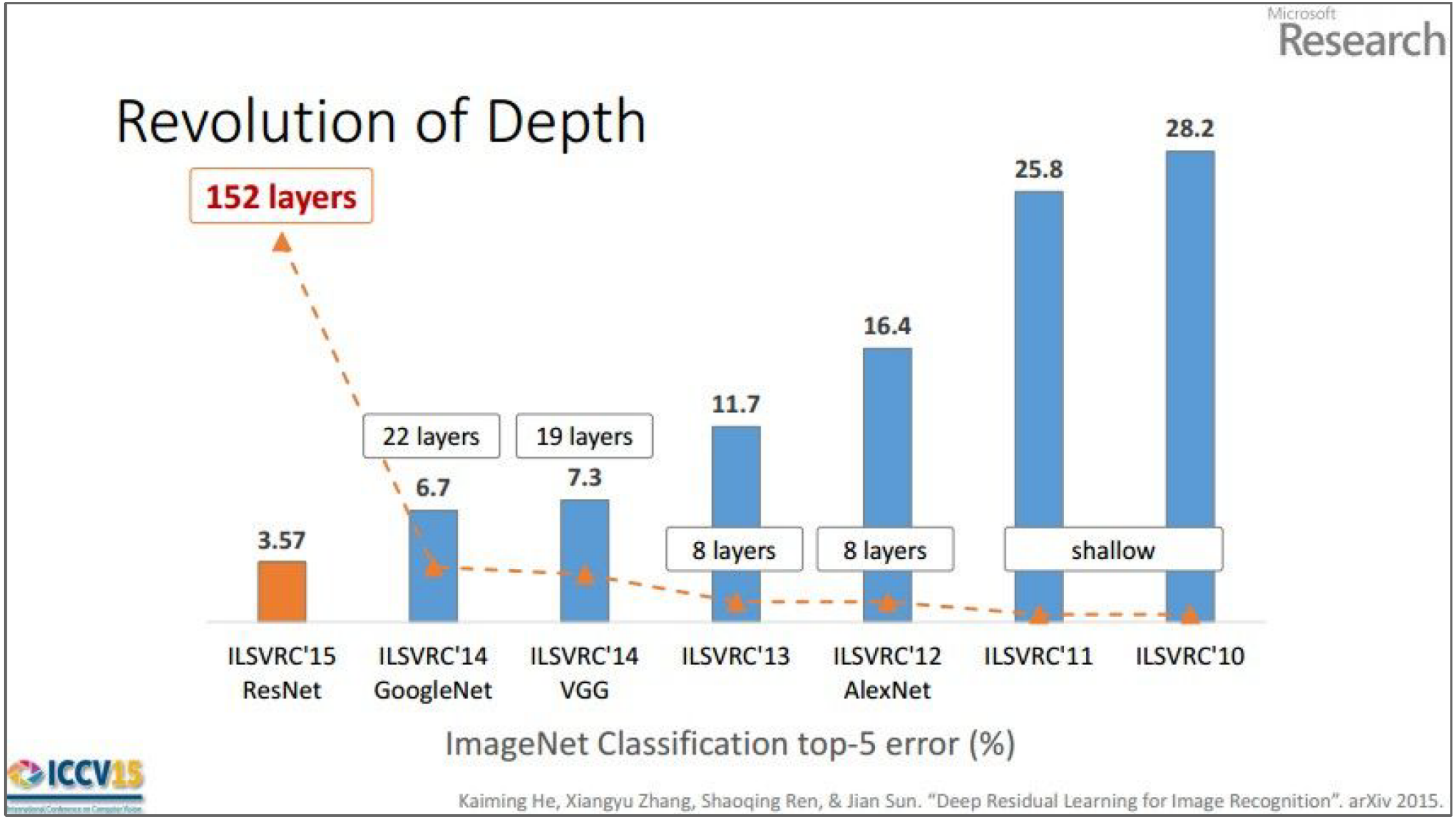
Lecture 8 Recap

Using CNNs in Computer Vision

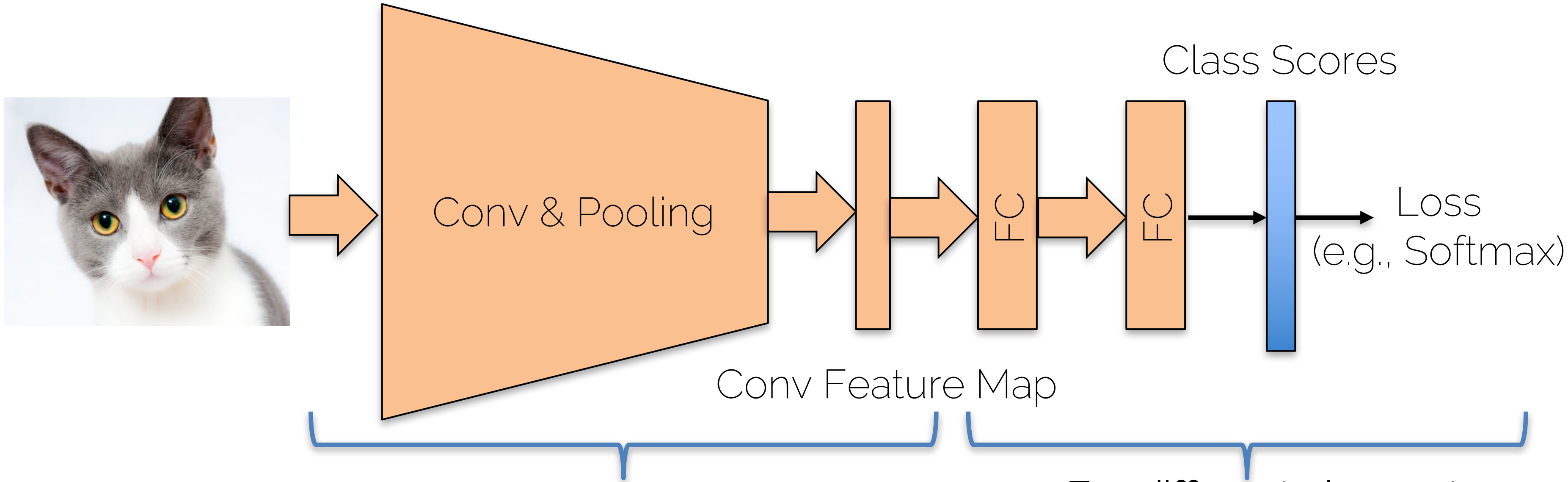
- We have CNNs (Convs, Pooling, FCs, Losses)
- We can employ them for classification
- We can employ them for regression

- Somewhat oversimplified: the "rest" is smart architectures and application of these tools
  - > of course it's more complicated ☺

CNN Architectures



How to Train in Practice?



E.g. AlexNet, VGG, GoogLeNet

Use Pre-Trained Network (e.g., download model)
-> keep ConvLayers fixed

For different class set,
only train FCs
-> new class scores
-> less training data
-> faster training

Don't be afraid to use newer architectures!

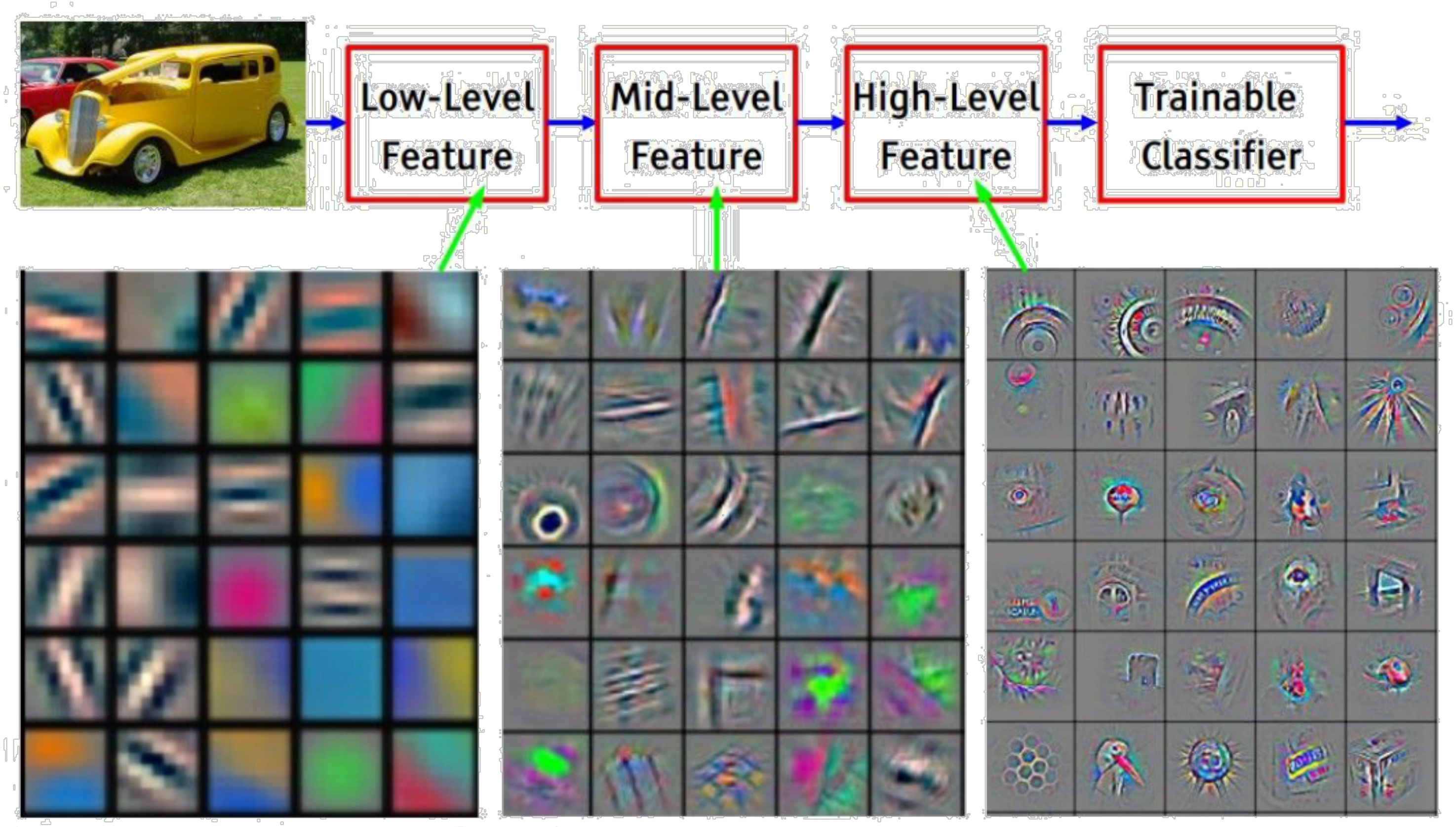
Network	Layers	Top-1 error	Top-5 error	Speed (ms)	Citation
AlexNet	8	42.90	19.80	14.56	[1]
Inception-V1	22	-	10.07	39.14	[2]
VGG-16	16	27.00	8.80	128.62	[3]
VGG-19	19	27.30	9.00	147.32	[3]
ResNet-18	18	30.43	10.76	31.54	[4]
ResNet-34	34	26.73	8.74	51.59	[4]
ResNet-50	50	24.01	7.02	103.58	[4]
ResNet-101	101	22.44	6.21	156.44	[4]

"Poor" mans choice:
- Resnet 50

Better performance:
- Inception-V3
- Xception

(credit: Justin Johnson, **jcjohnson on github**)

Convolutional Neural Network

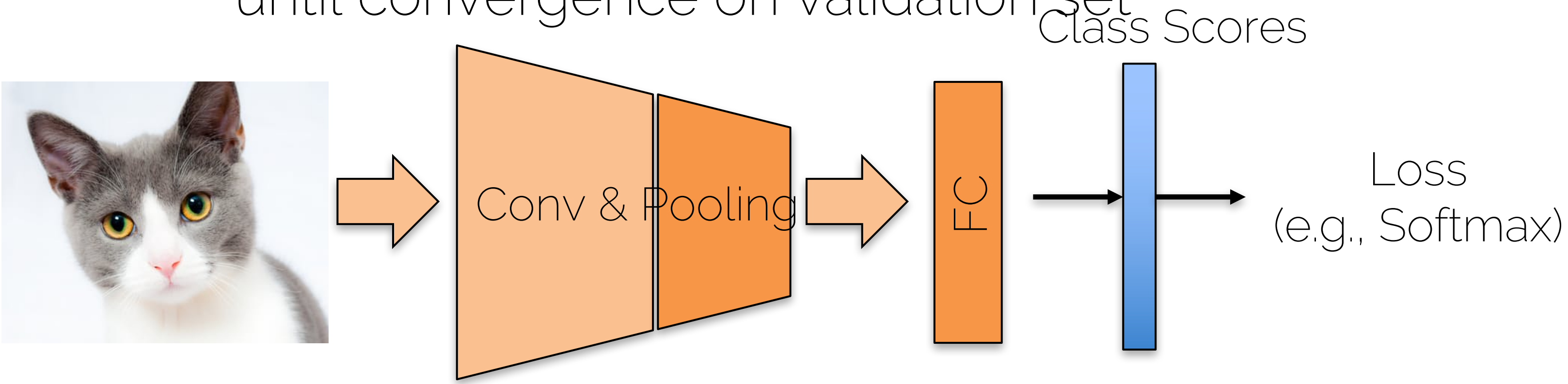


Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Slide by LeCun

How to finetune convolutional layers?

- Take network pretrained on big dataset (ImageNet)
- Reinitialize fully connected layer(s)
- Train a new fully connected network for a few epochs with fixed convolutional layers (or choose low(er) learning rate)
- Set a subset of convolutional layers to trainable and train until convergence on validation set



Using CNNs in Computer Vision

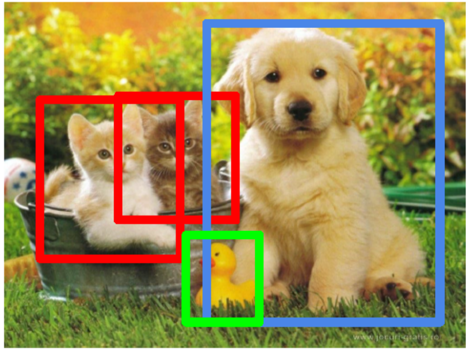
Classification

CAT

Classification + Localization

CAT

Object Detection



CAT, DOG, DUCK

Instance Segmentation

CAT, DOG, DUCK

Single object

Multiple objects

Credit: Li/Karpathy/Johnson

Using CNNs in Computer Vision

Classification

CIFAR 10 +
"raw" CNN ☺

Classification + Localization

Object Detection

Instance Segmentation

Credit: Li/Karpathy/Johnson

Important Datasets to Know

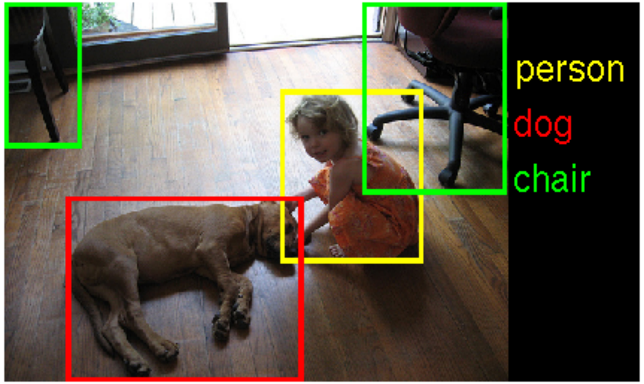
CIFAR-10: single object, centered, Krizhevsky et al.

MNIST: handwritten digits, LeCun et al.

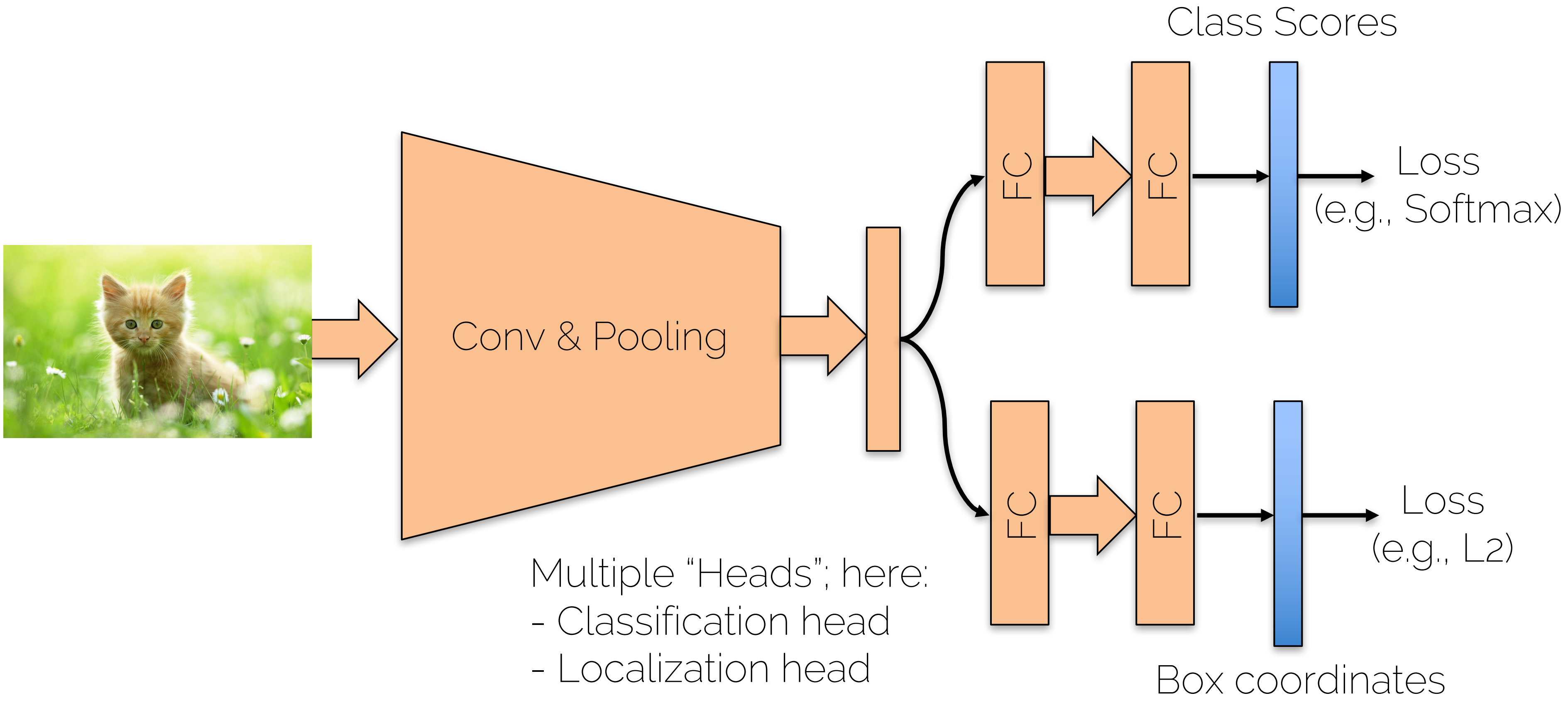
Pascal VOC, 20 classes, 10k images, Everingham et al.

ImageNet: 10 mio images, Deng et al.

MSCoco, 300k images, Lin et al. 15



Classification + Localization: Regression



Multiple "Heads"; here:
- Classification head
- Localization head

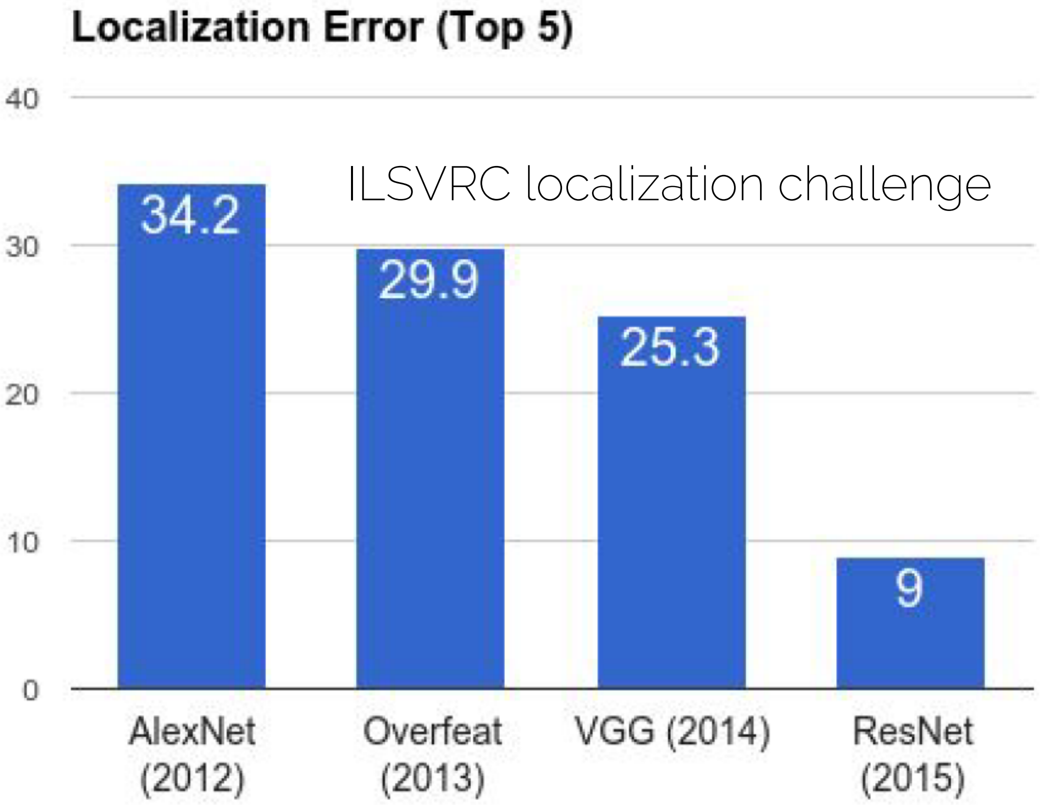
Classification + Localization: Sliding Window

Class score (cat):

Box location 0	-> score 0.02
Box location 1	-> score 0.2
Box location 2	-> score 0.42
Box location 3	-> score 0.31
Box location 4	-> score 0.8

Take winning box location as result

ImageNet Classification + Localization



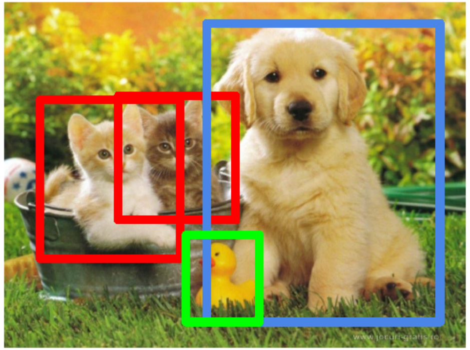
Overfeat: Multiscale conv regression with box merging

VGG: Mostly the same, but better network (also fewer scales and location, gain by better features)

ResNet: Crazy network, and different localization method (region proposals, RPN)

Credit: Li/Karpathy/Johnson

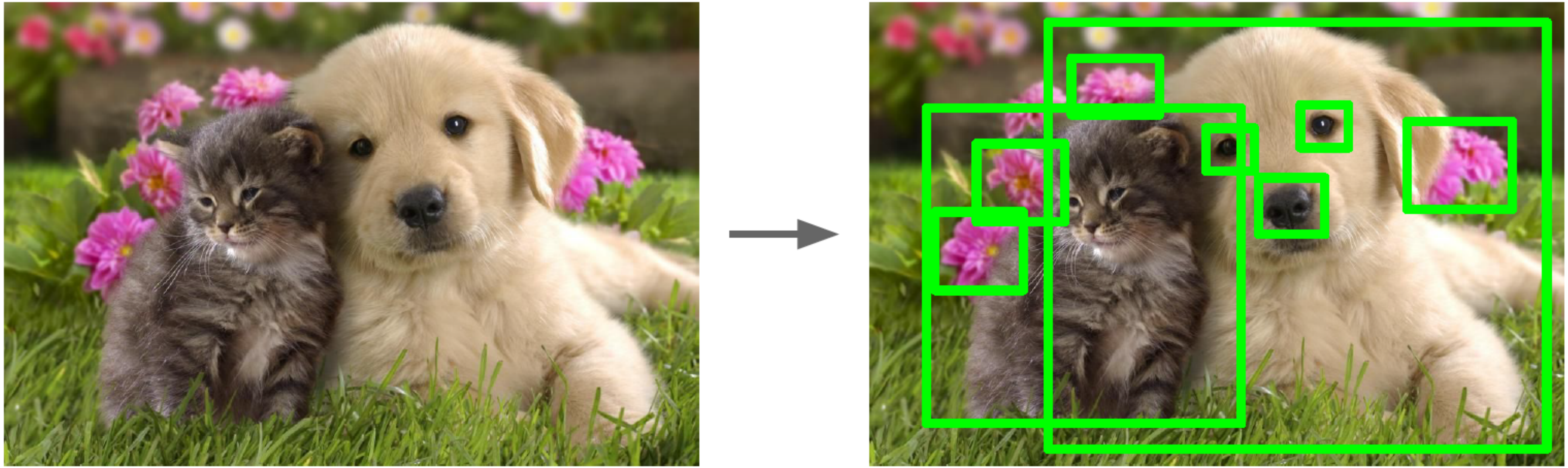
Using CNNs in Computer Vision

Classification	Classification + Localization	Object Detection	Instance Segmentation
			
CIFAR 10 + "raw" CNN ☺	Regression and/or sliding window		

Multiple objects!
(but we don't know how many)

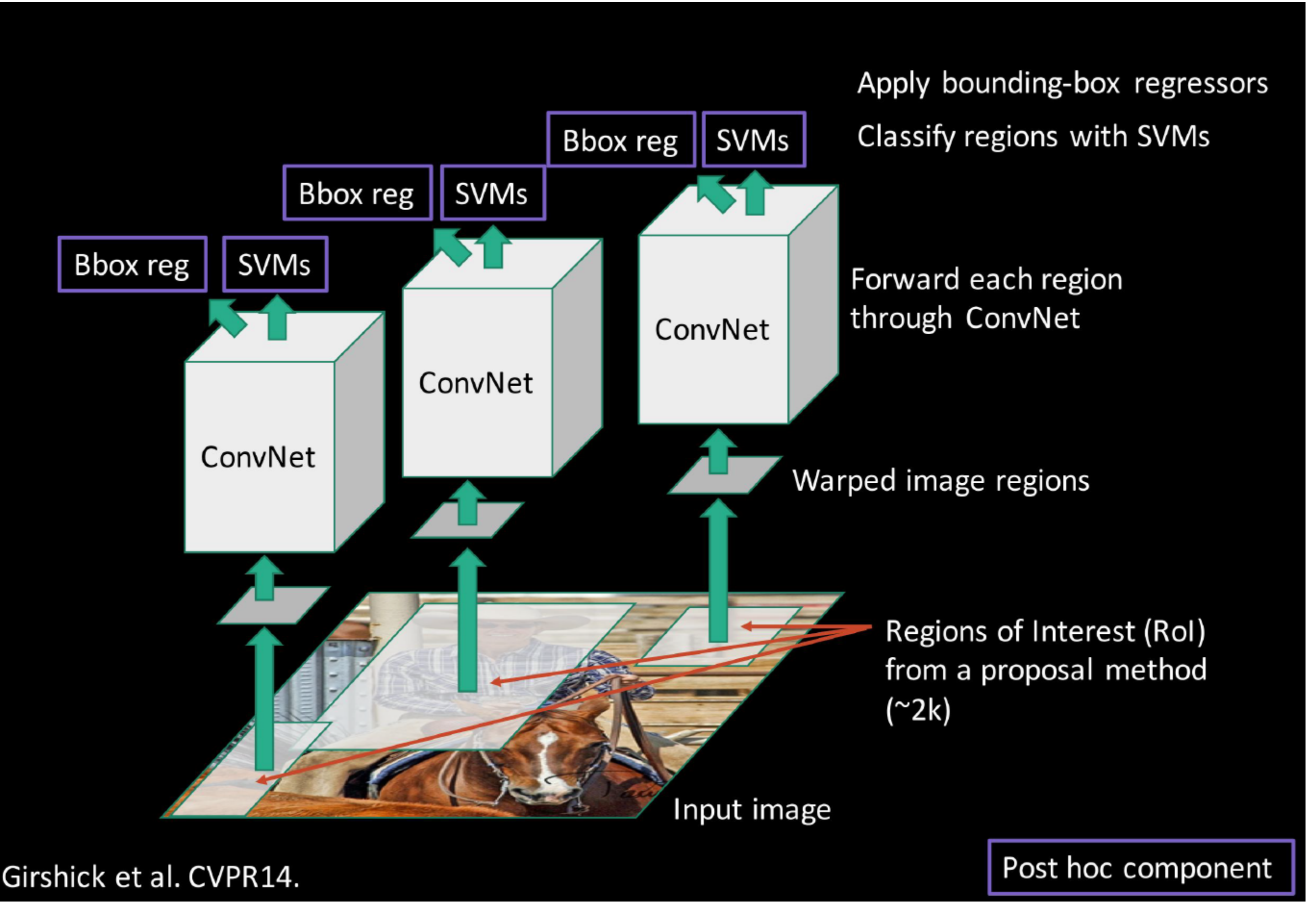
Credit: Li/Karpathy/Johnson

Region Proposals



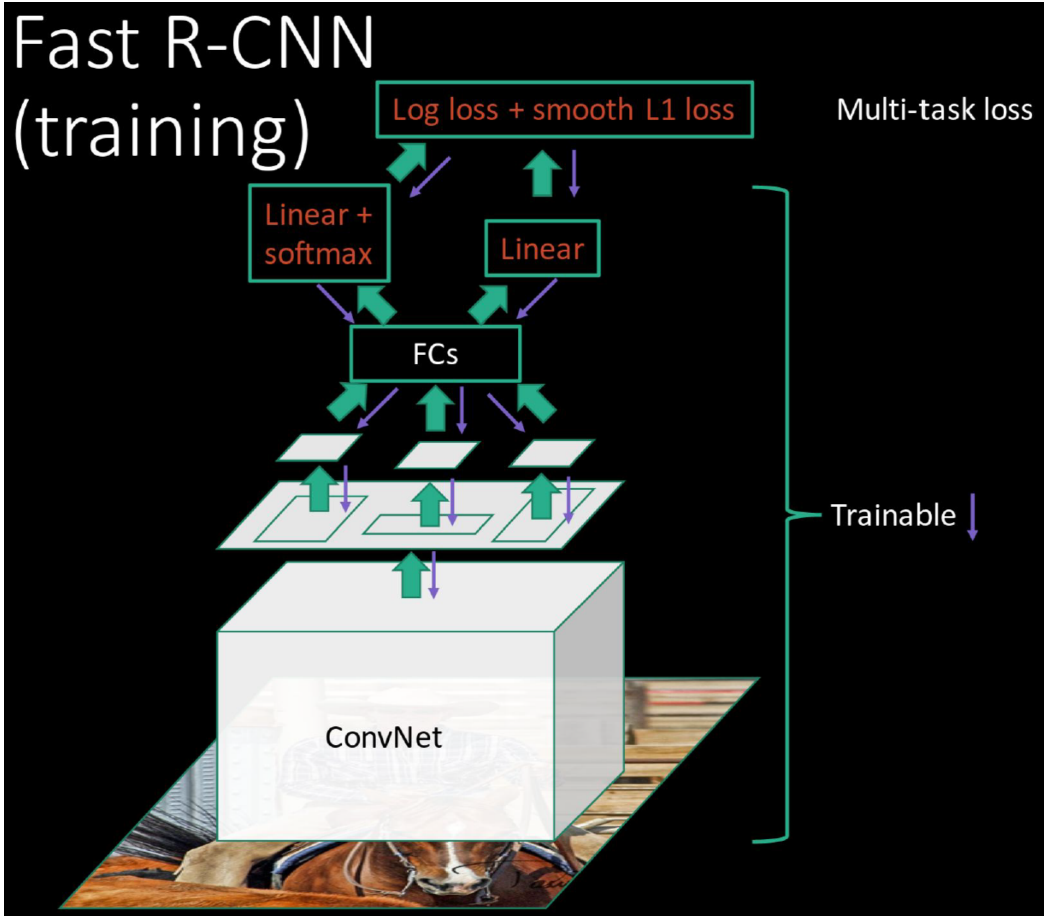
Credit: Li/Karpathy/Johnson

Putting it Together: R-CNN



1) Run region proposal (e.g., selective search)
2) Warp (i.e., re-scale, re-size) to a fixed image size
3) This fixed output is fit into a CNN with class + regression head, which corrects for slightly off proposals

Fast R-CNN (training)

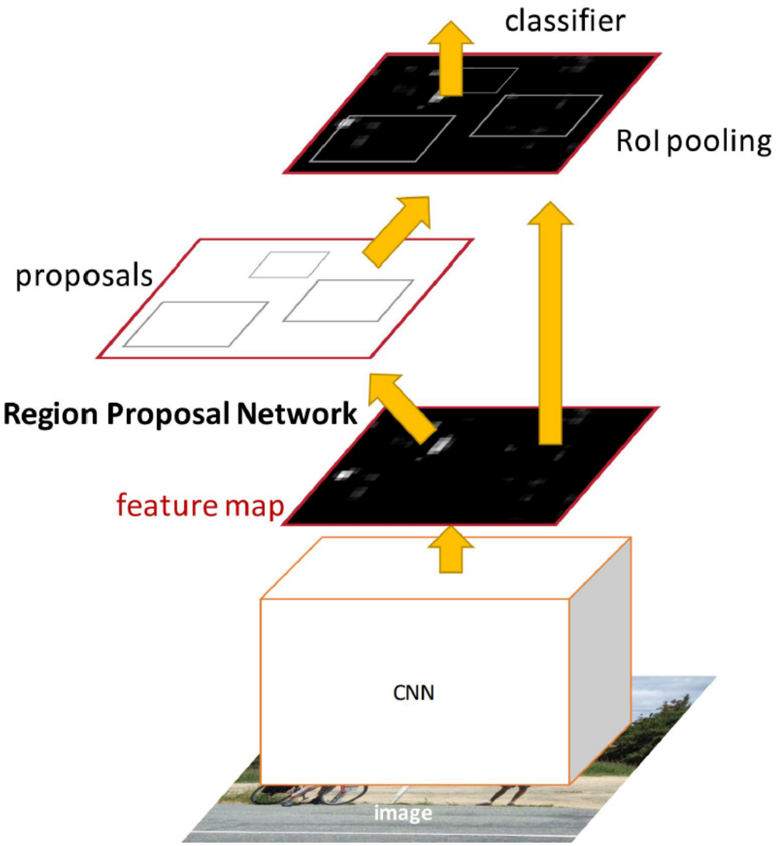


Solves training time issue: 1) CNN not updated with SVM losses. 2) Complex training pipeline

-> Just train whole thing end-to-end

[Girshick 15, Fast R-CNN]

Faster R-CNN



Solution: make the CNN also do region proposals!

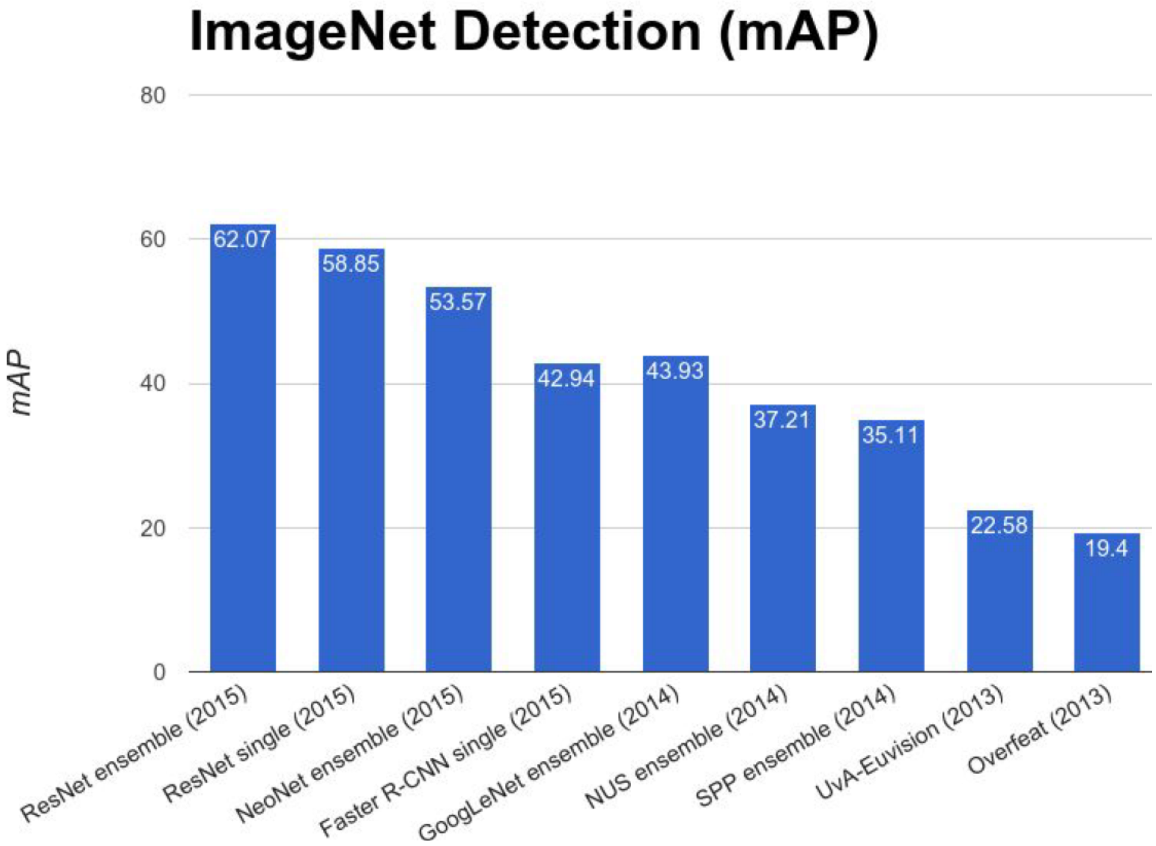
Insert a Region Proposal Network (RPN) after last conv layer

RPN produces region proposals (one shot) -> no need for external proposals

After RPN, region of interest pooling, and use similar classifier and bbox regressor like Fast R-CNN

[Girshick 15, Faster R-CNN]

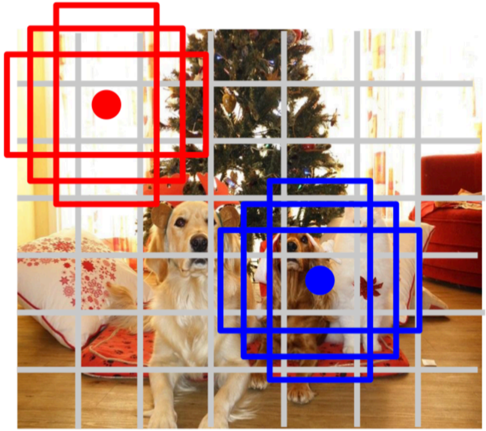
ImageNet Detection 2013 - 2015



Credit: Li/Karpathy/Johnson

Detection without Proposals: Yolo/SSD

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

Credit: Li, Johnson, Yeung

Object Detection: Lots of variables ...

Base Network
VGG16
ResNet-101
Inception V2
Inception V3
Inception
ResNet
MobileNet

Object Detection architecture
Faster R-CNN
R-FCN
SSD

Image Size
 # Region Proposals
…

Takeaways
Faster R-CNN is slower but more accurate

SSD is much faster but not as accurate

Huang et al, “Speed/accuracy trade-offs for modern convolutional object detectors”, CVPR 2017

R-FCN: Dai et al, “R-FCN: Object Detection via Region-based Fully Convolutional Networks”, NIPS 2016
Inception-V2: Ioffe and Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”, ICML 2015
Inception V3: Szegedy et al, “Rethinking the Inception Architecture for Computer Vision”, arXiv 2016
Inception ResNet: Szegedy et al, “Inception-V4, Inception-ResNet and the Impact of Residual Connections on Learning”, arXiv 2016
MobileNet: Howard et al, “Efficient Convolutional Neural Networks for Mobile Vision Applications”, arXiv 2017

Credit: Li, Johnson, Yeung

Lecture 9

Image Segmentation and Instance Segmentation

Using CNNs in Computer Vision

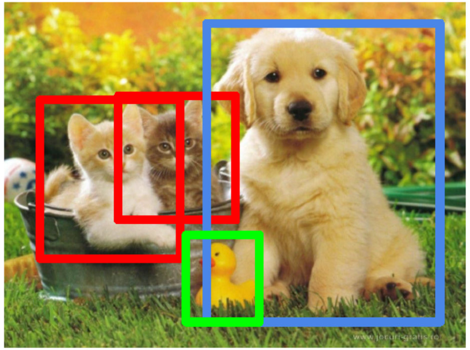
Classification

CIFAR 10 +
"raw" CNN ☺

**Classification
+ Localization**

Regression and/or
sliding window

Object Detection



Selective Search, (D)RP
(Fast(er)) R-CNN

**Instance
Segmentation**

Credit: Li/Karpathy/Johnson

Using CNNs in Computer Vision

Classification

CIFAR 10 +
"raw" CNN ☺

Classification + Localization

Regression and/or
sliding window

Object Detection

Selective Search, RP
(Fast(er)) R-CNN

Instance Segmentation

Credit: Li/Karpathy/Johnson

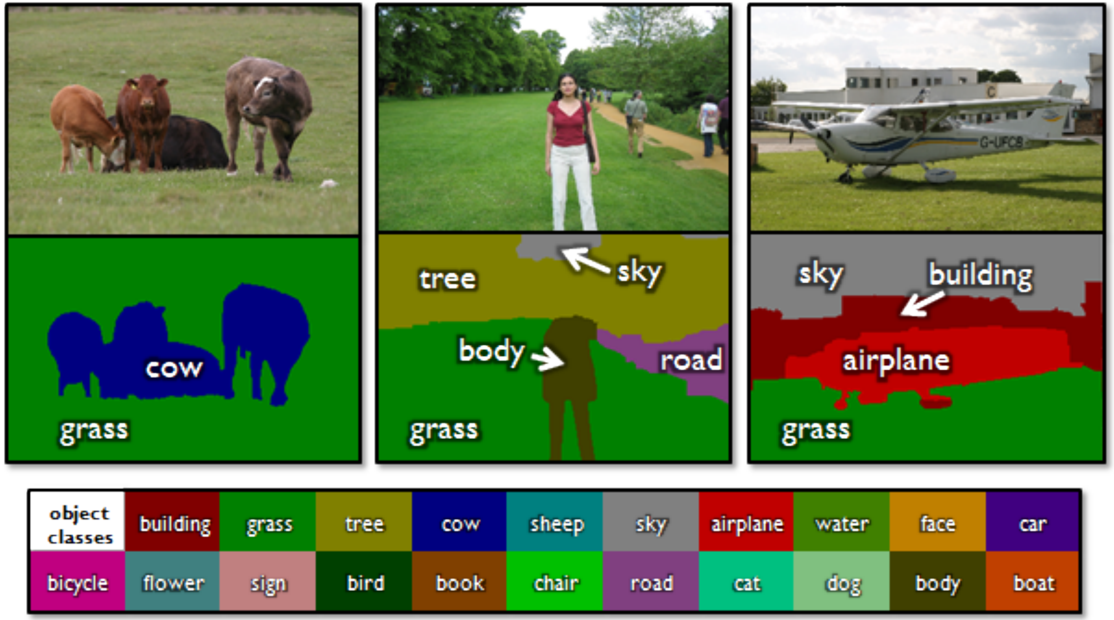
Semantic Segmentation

Predict class label for every pixel (i.e., dense pixel labeling)

No differentiation between instances

I.e., all objects of the same class receive same class label

Traditional computer vision task



[Shotton et al. 07] TextonBoost

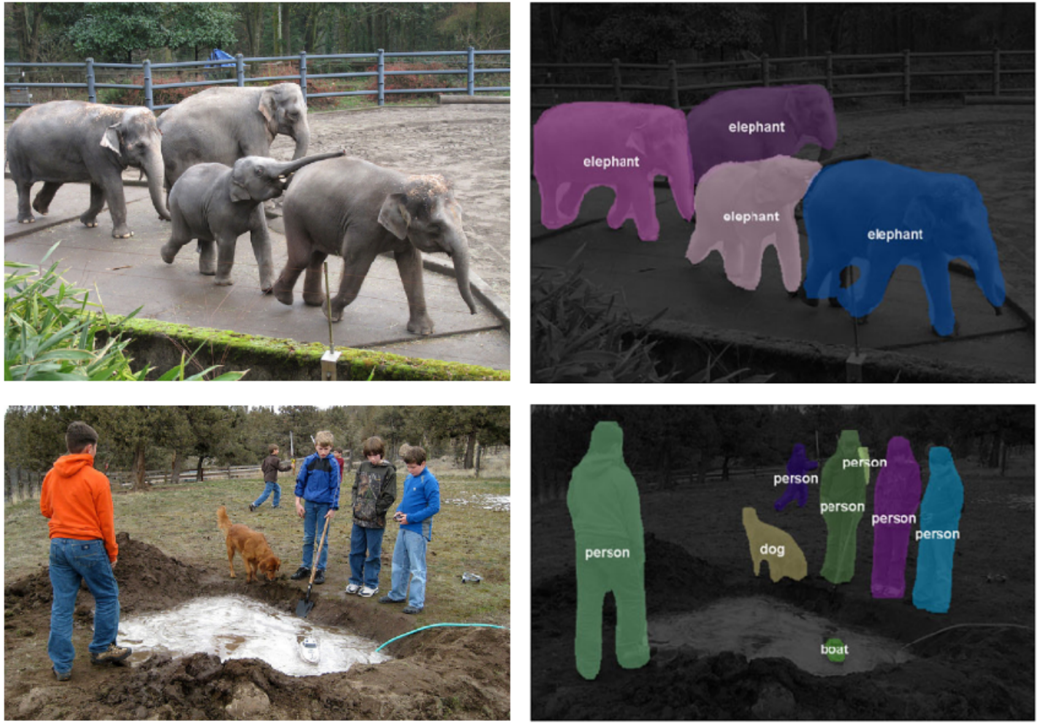
Instance Segmentation

Detect instances, classify category, label pixels of each instance;

Distinguish between instances within a category;
e.g., elephant1, elephant2, etc.

Simultaneous detection and segmentation (SDS)

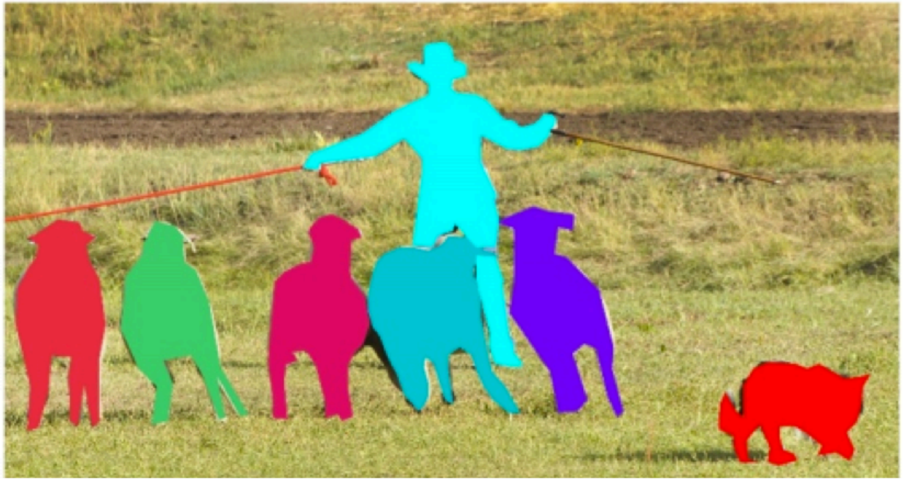
MS COCO is core dataset
-> lots of work around it



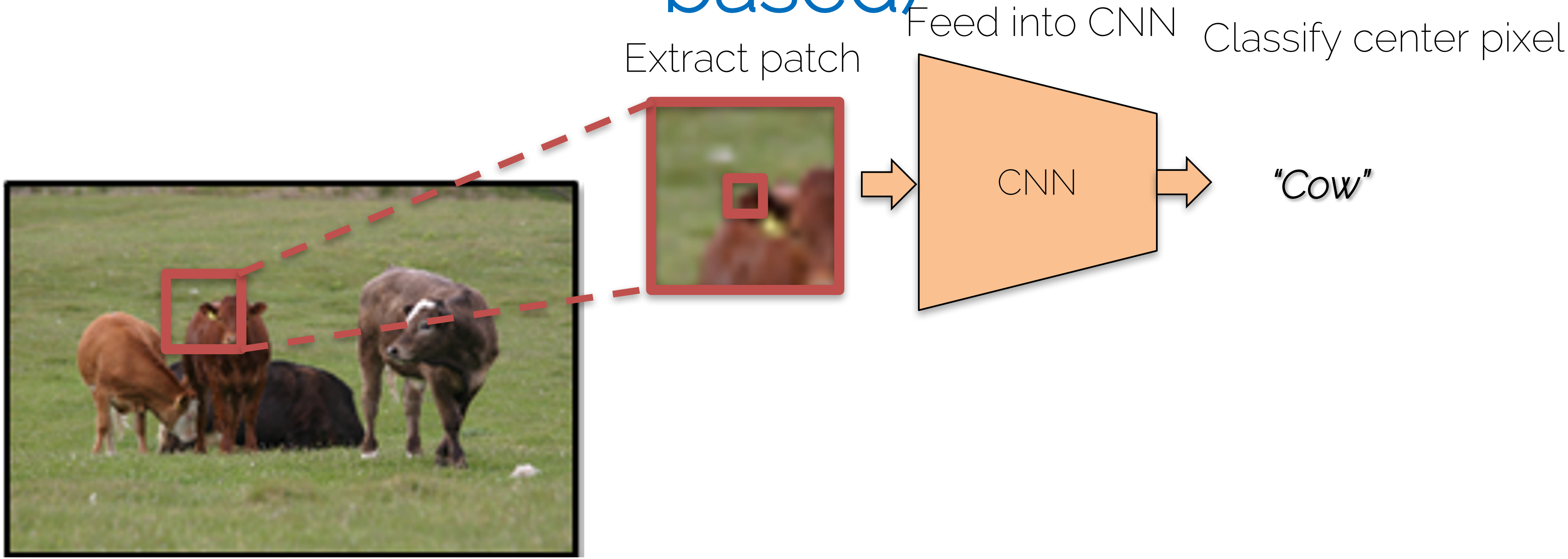
[Dai et al. 15] Instance-aware Semantic Segmentation

Training Data

- Have a number of fixed classes
- We must label every pixel in our training set!
- Very expensive!
- Usual way of handling this: crowdsourcing



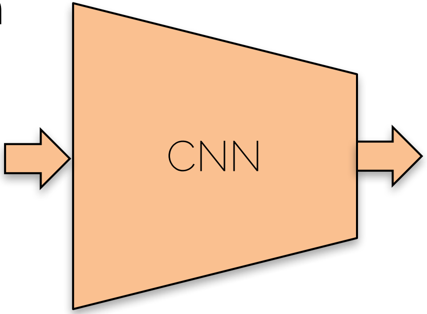
Semantic Segmentation (Patch-based)



Semantic Segmentation (Patch-based)

Extract patch

Feed into CNN



Classify center pixel

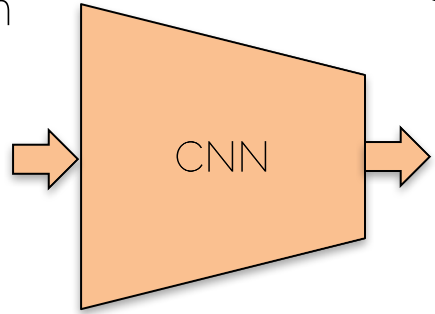
Run CNN for every pixel!

Semantic Segmentation (Patch-based)

Extract patch

Feed into CNN

Classify center pixel



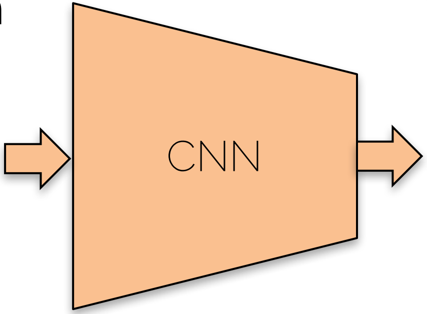
Run CNN for every pixel!

Semantic Segmentation (Patch-based)

Extract patch

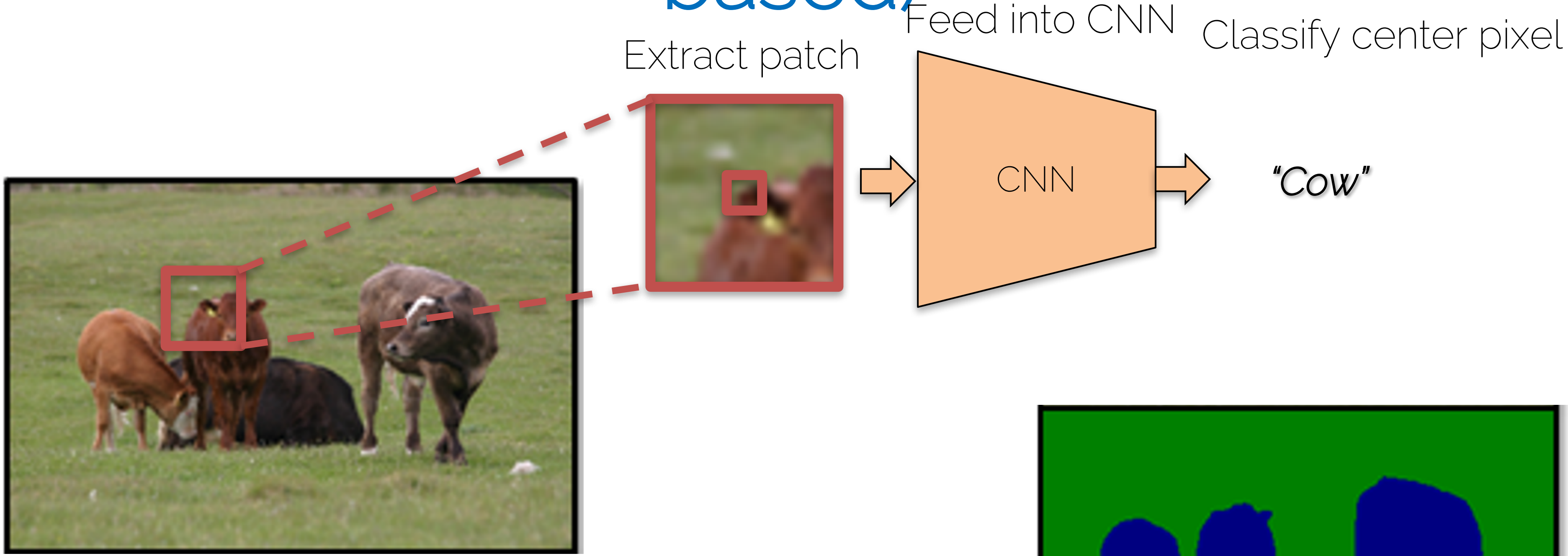
Feed into CNN

Classify center pixel



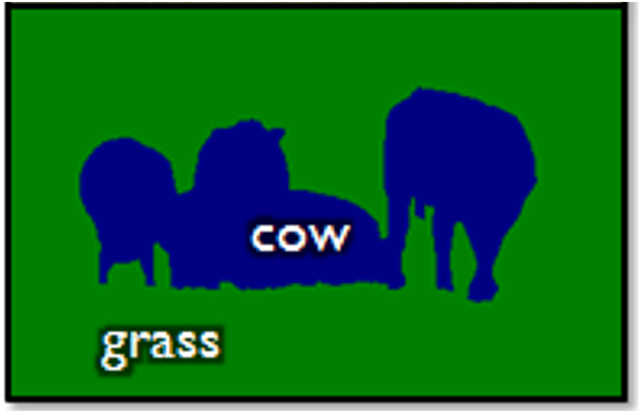
Run CNN for every pixel!

Semantic Segmentation (Patch-based)



Run CNN for every pixel!

Possibly run a CRF at the end

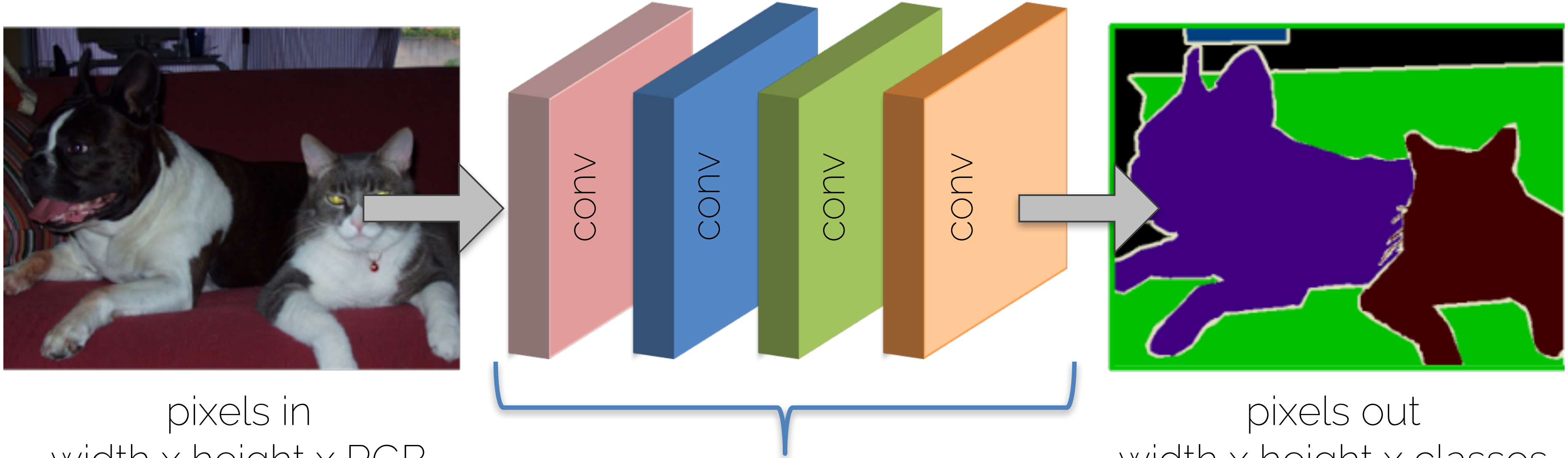


Semantic Segmentation (Patch-based)

- Extract patch from image for every pixel
- Run every patch independently through a CNN

- Easy architecture: just classify -- use VGG/ResNet
- Easy to train: just use pixel center label for patch
- Expensive at test time

Semantic Segmentation (Fully Convolutional)



pixels in
width x height x RGB

Just convs & activations

pixels out
width x height x classes

Fully Convolutional Network

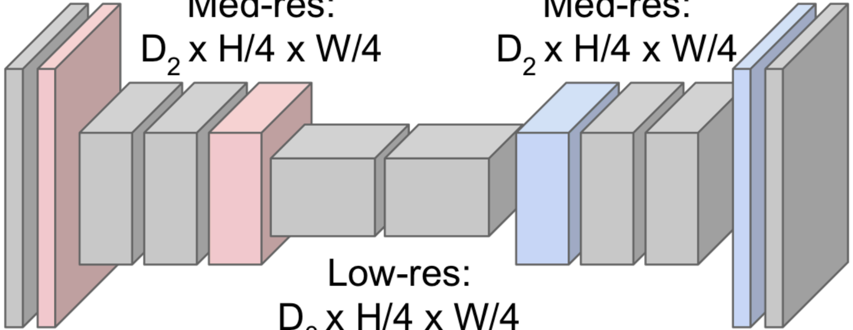
Super expensive!

Semantic Segmentation (FC)

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

Pooling, strided convolutions

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

Credit: Li, Johnson, Yeung

Unpooling

Nearest Neighbor

1	2
3	4

2x2

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

4x4

"Bed of Nails"

1	2
3	4

2x2

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

4x4

Max Unpooling

Max Pooling

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

4X4

6	8
7	6

2X2

Max Unpooling

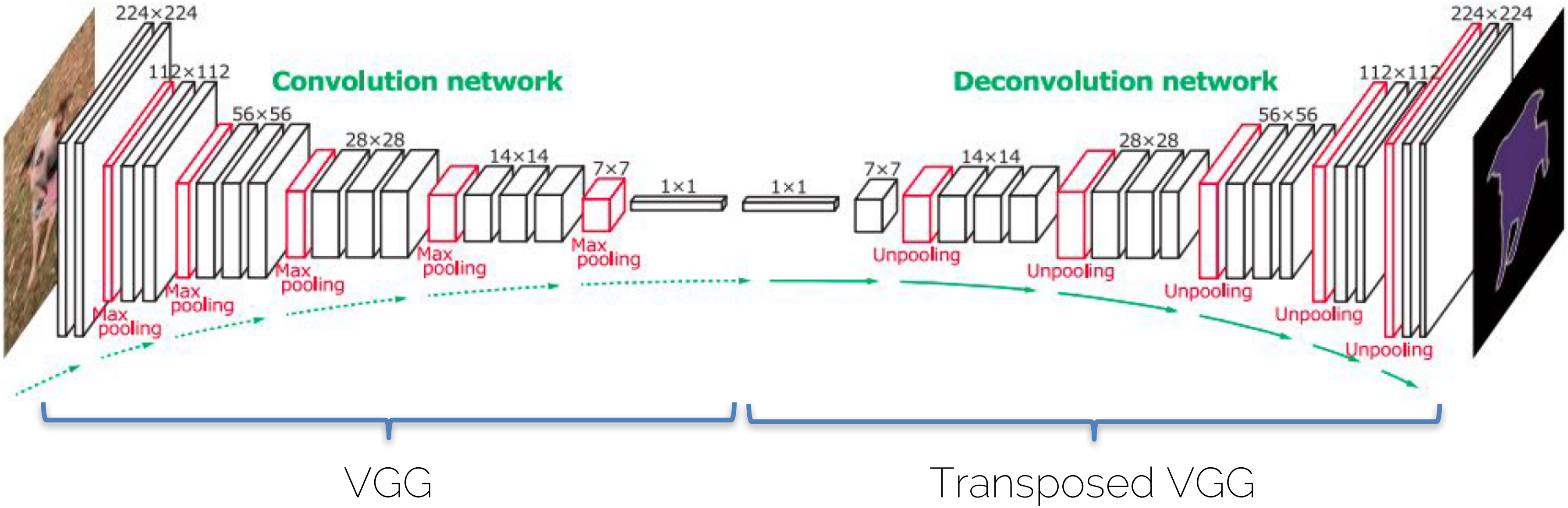
1	2
3	4

2X2

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

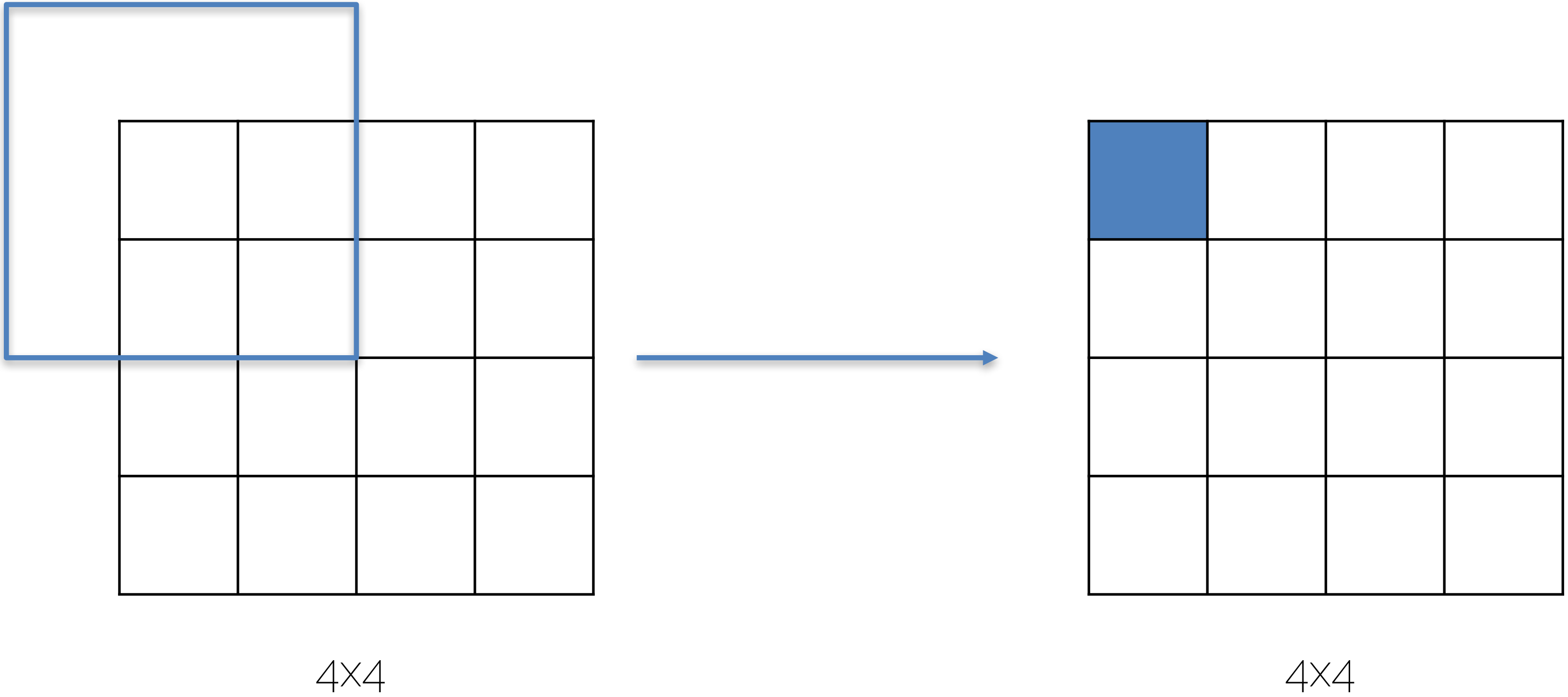
4X4

Unpooling in Action

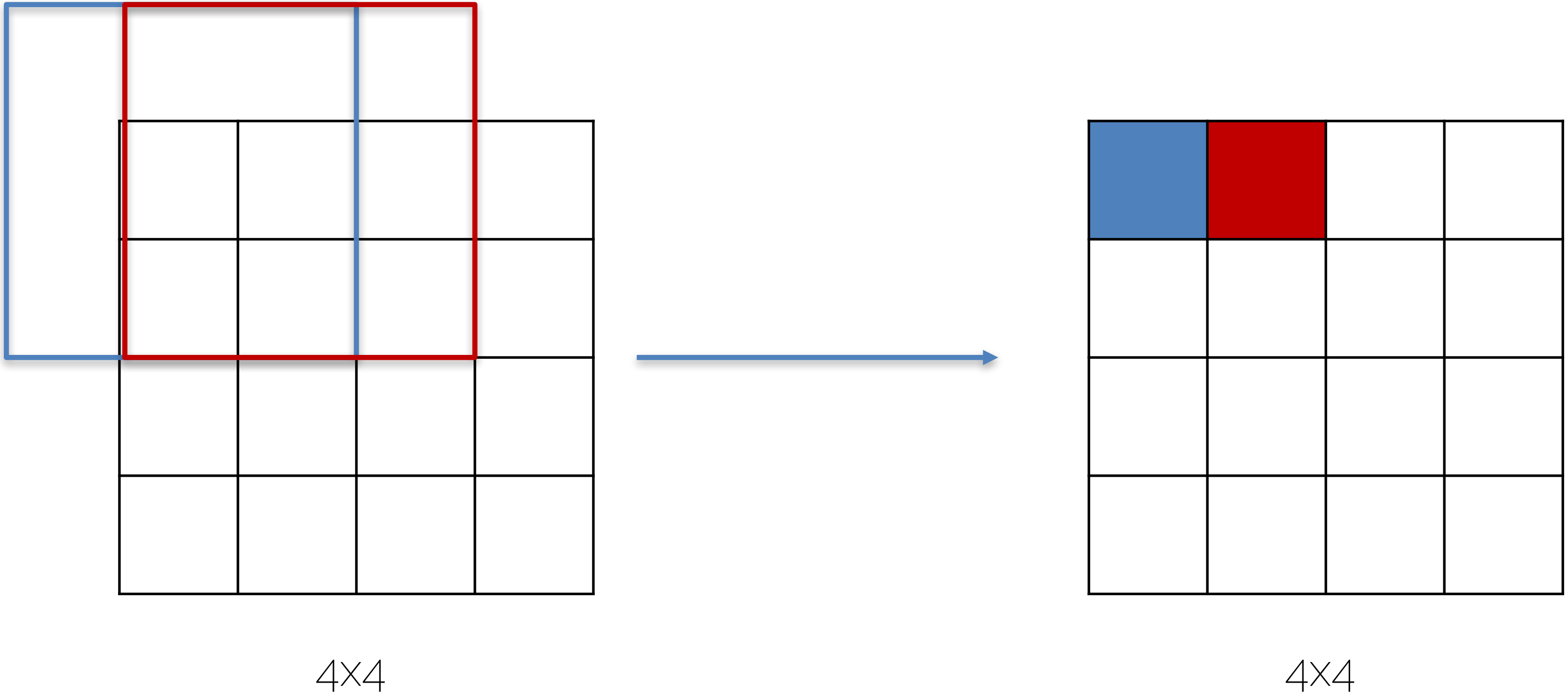


[Noh et al. 15] Learning Deconvolution Network for Semantic Segmentation

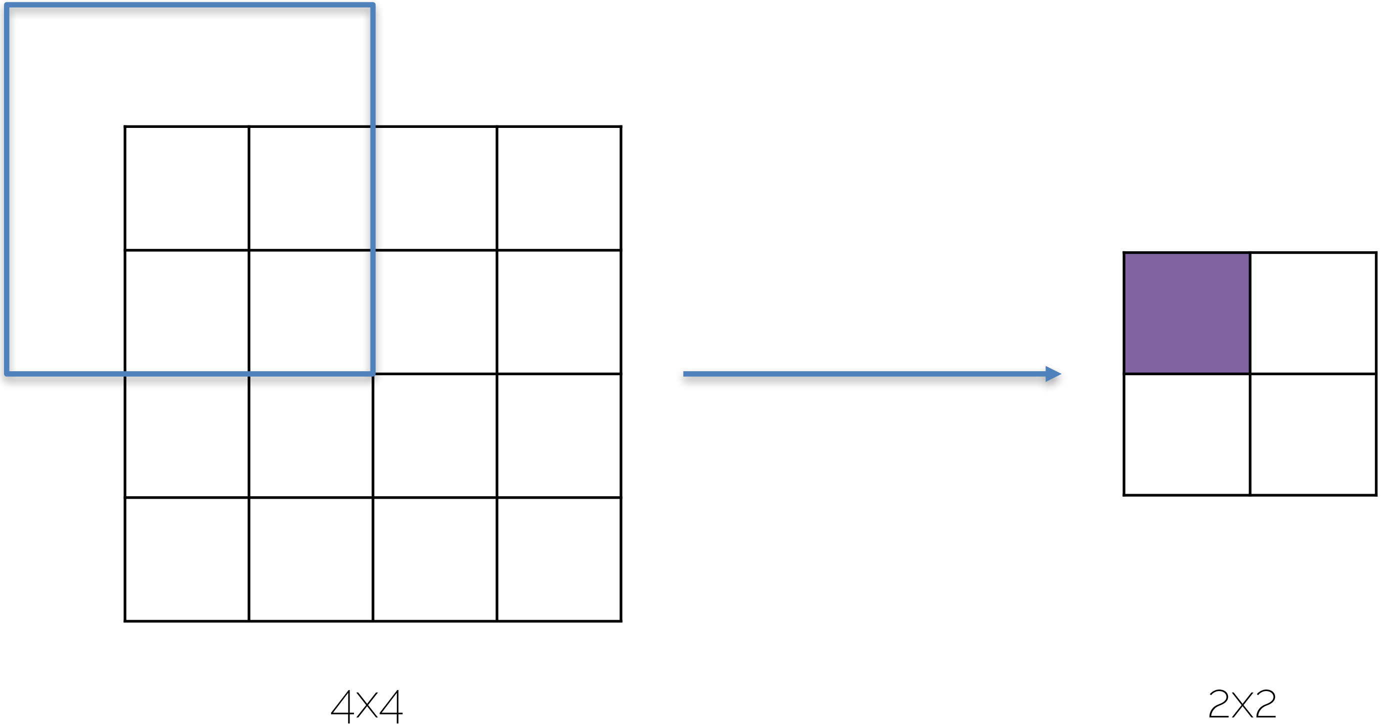
Recall: 3x3 Convolution, Stride 1, Pad 1



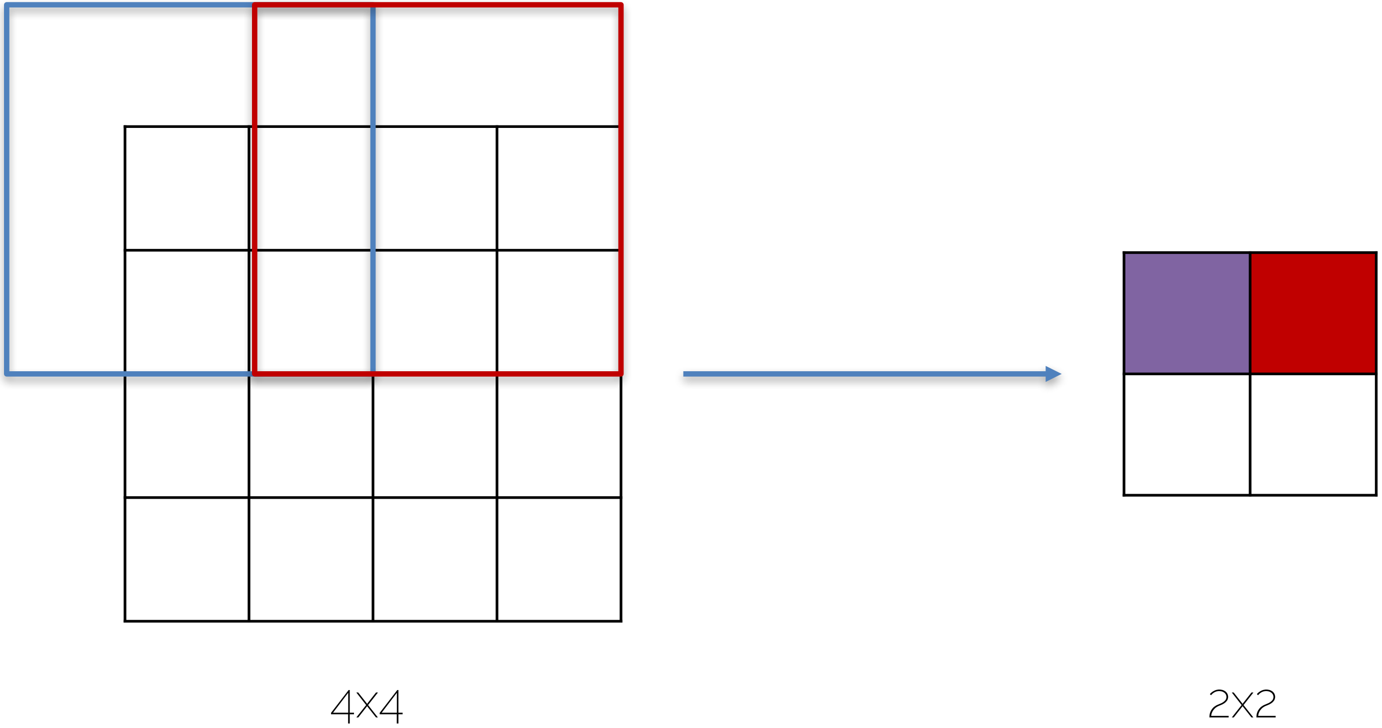
Recall: 3x3 Convolution, Stride 1, Pad 1



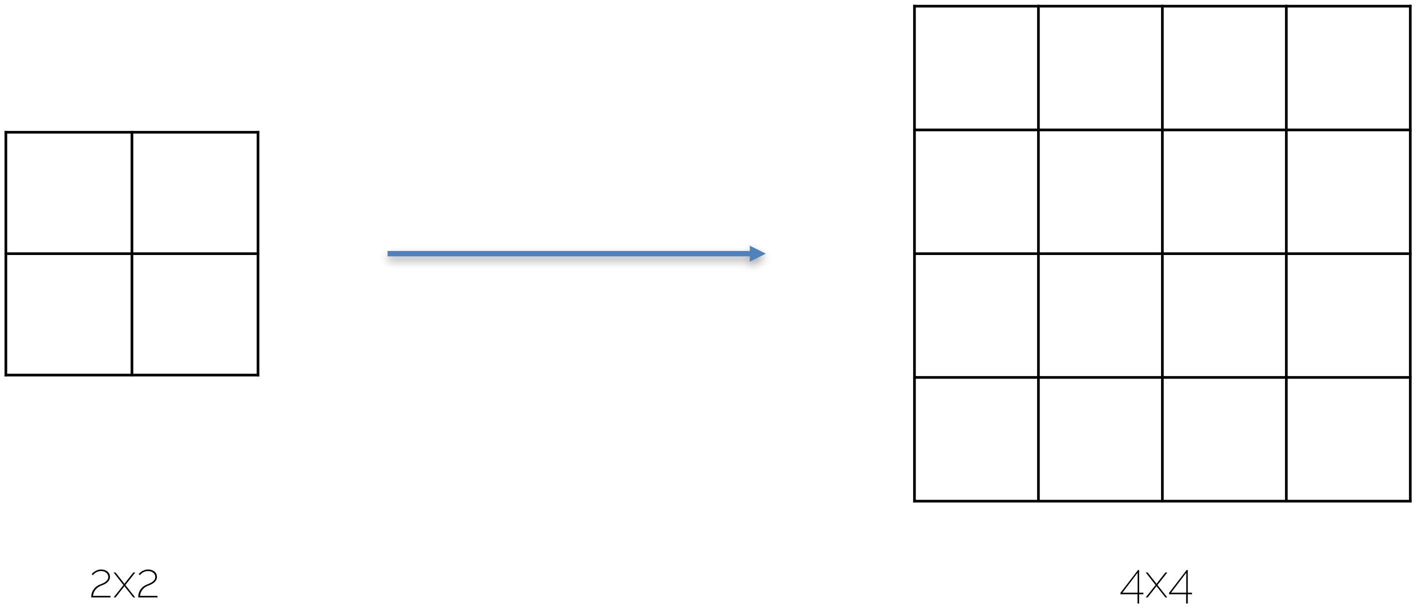
Recall: 3x3 Convolution, Stride 2, Pad 1



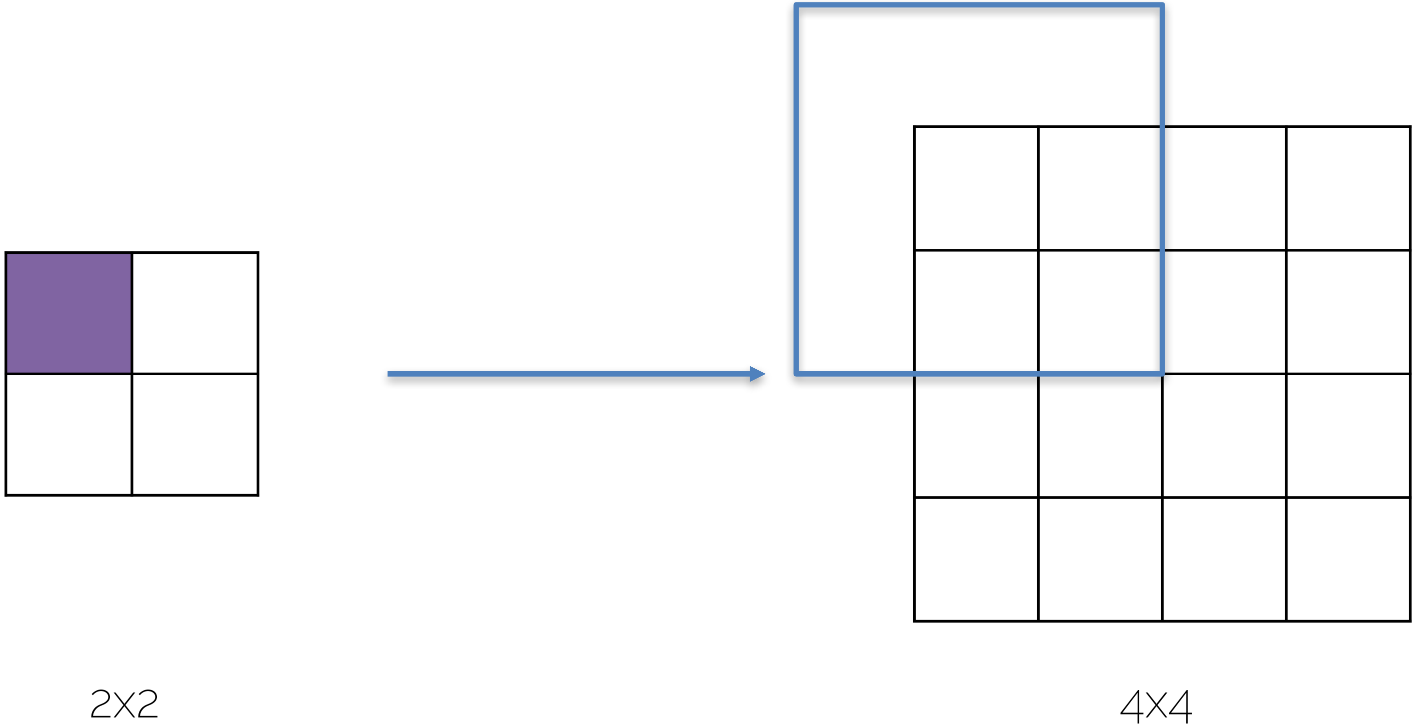
Recall: 3x3 Convolution, Stride 2, Pad 1



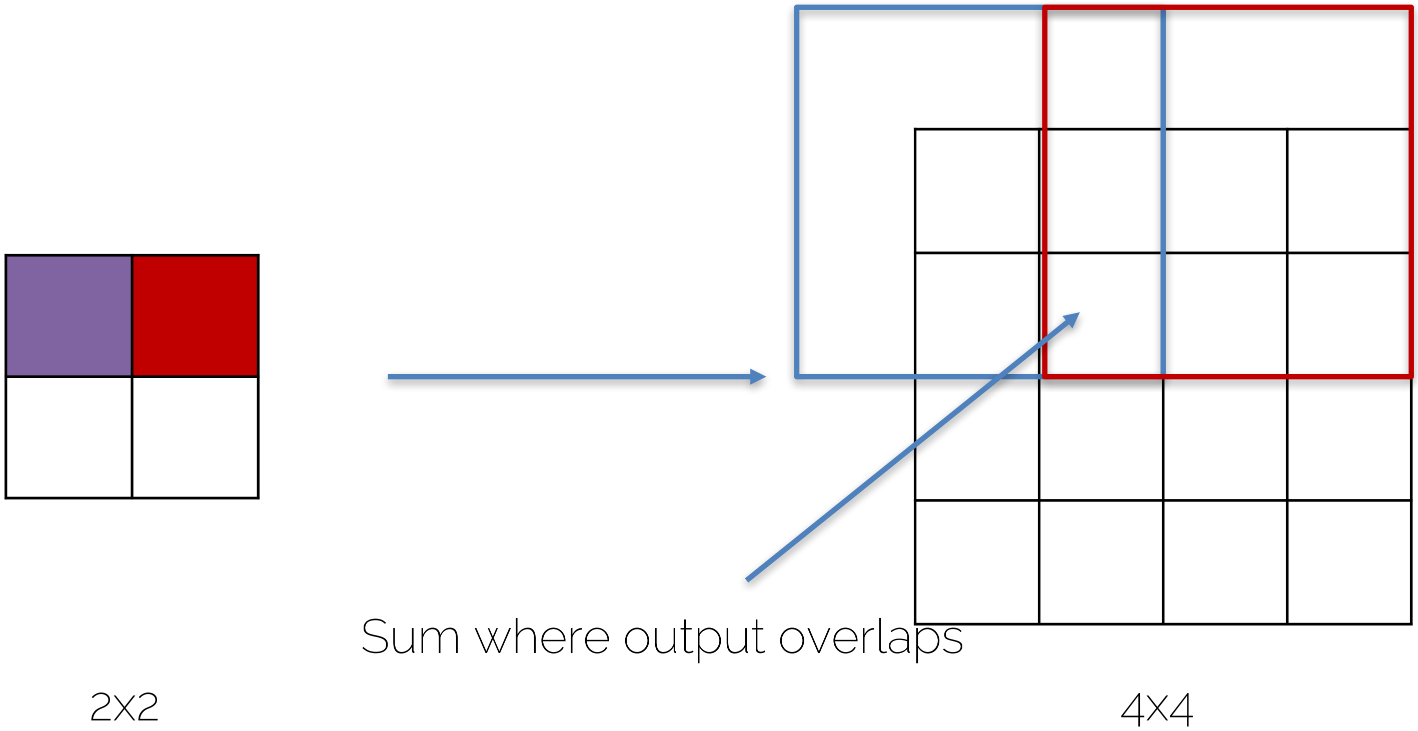
3x3 Transpose Convolution, <u>Stride 2</u>, Pad 1



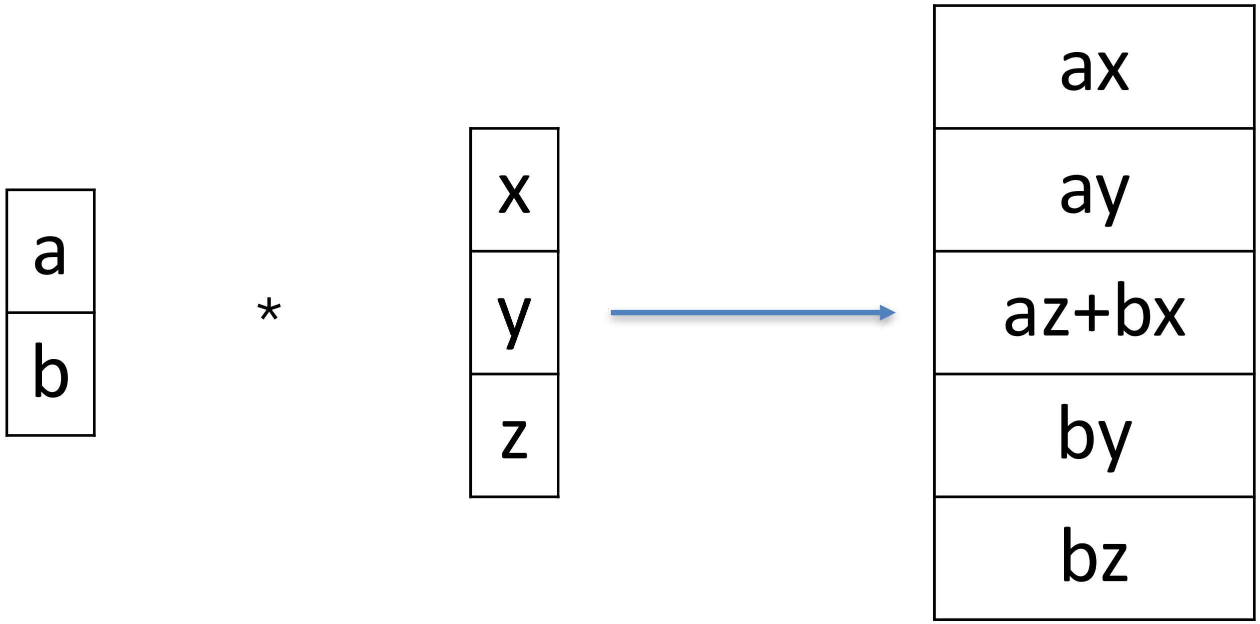
3x3 Transpose Convolution, Stride 2, Pad 1



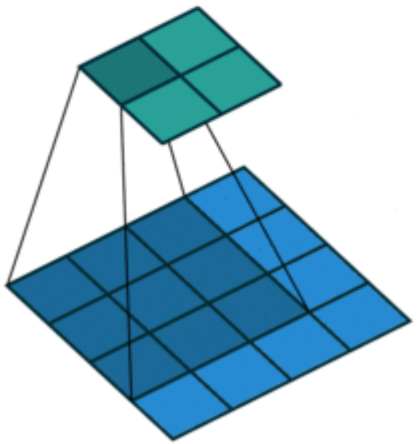
3x3 Transpose Convolution, Stride 2, Pad 1



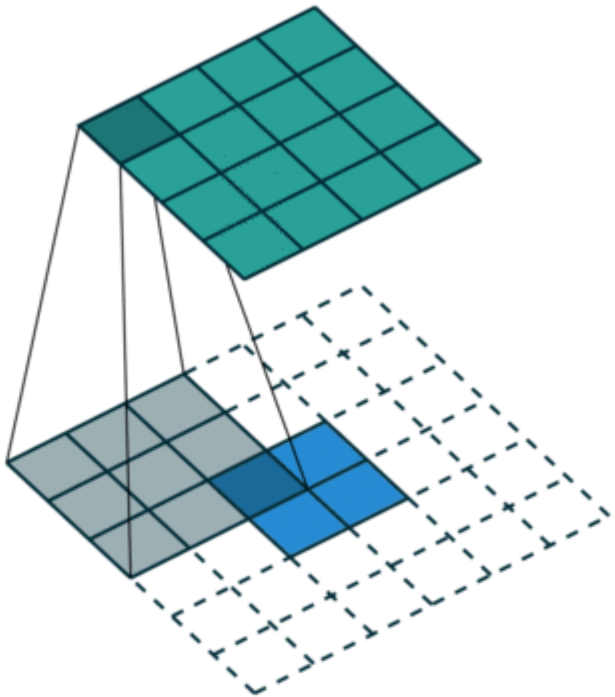
Example in 1D, stride 2



Transpose Convolution, Stride 1



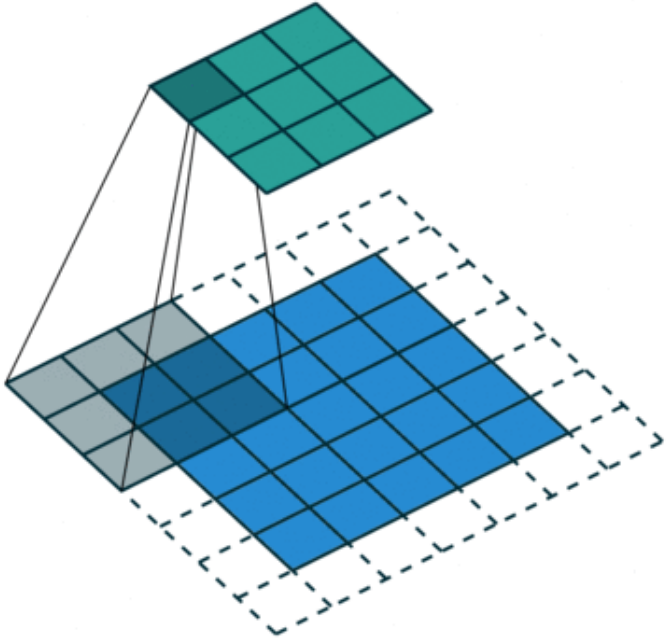
Convolution
no padding, no stride



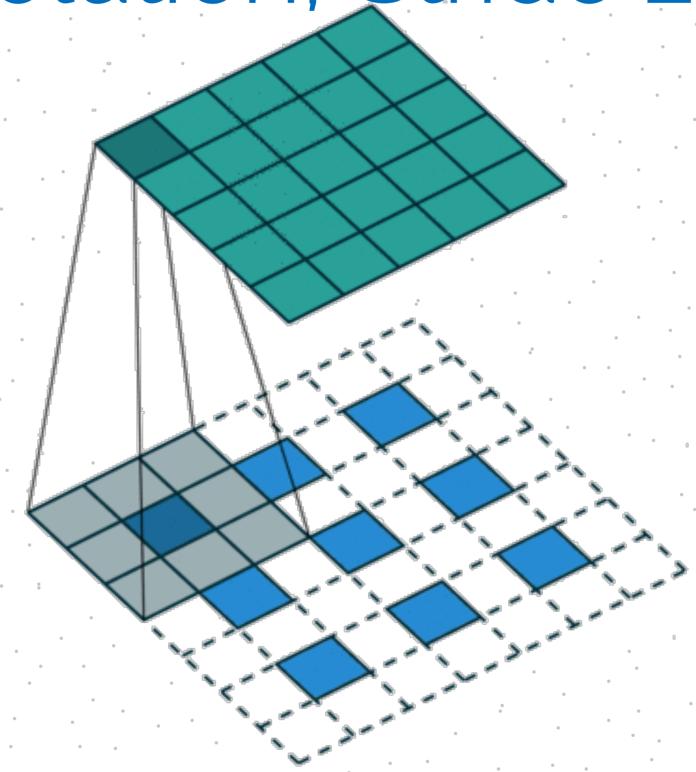
Transposed convolution
no padding, no stride

https://github.com/vdumoulin/conv_arithmetic

Transpose Convolution, Stride 2



Convolution
padding, stride



Transposed convolution
padding, stride

https://github.com/vdumoulin/conv_arithmetic

Convolution as Matrix Multiplication

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, <u>stride=2</u>, padding=1

Convolution as Matrix Multiplication

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, <u>stride=2</u>, padding=1

Convolution transpose multiplies by the transpose of the same matrix:

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

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

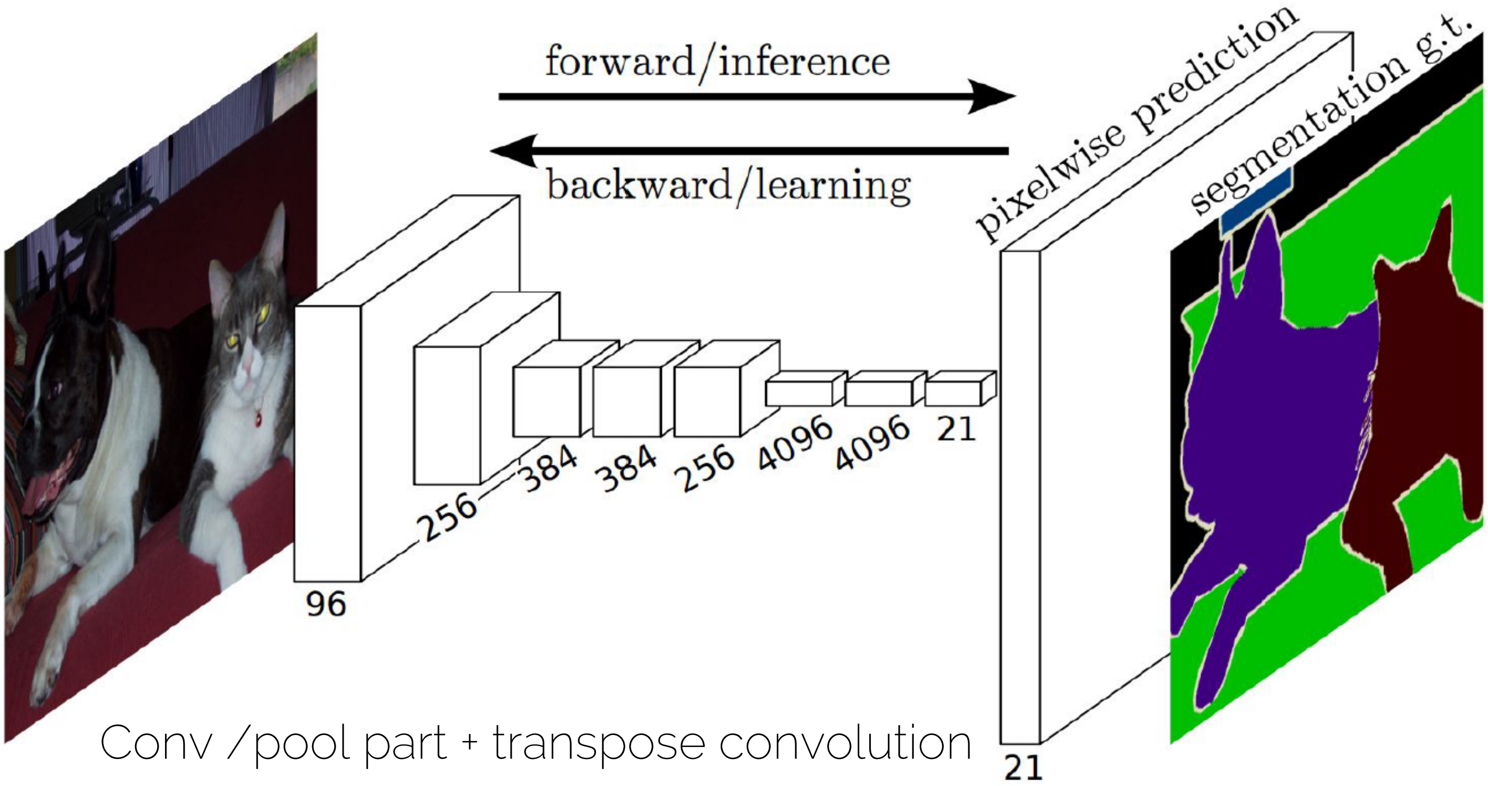
No longer a convolution if stride > 1

Transpose Convolution

- Input gives weight for filter
- Stride gives ratio between movement in output and input
- Replace backward and forward pass of convolutional layer
- Avoid "checkerboard" patterns by using a even number as convolutional kernel size (e.g. k=4, s=2 instead of k=3, s=2)
- Alternate names:
  – Deconvolution
  – Upconvolution

How to trigger mathematicians…

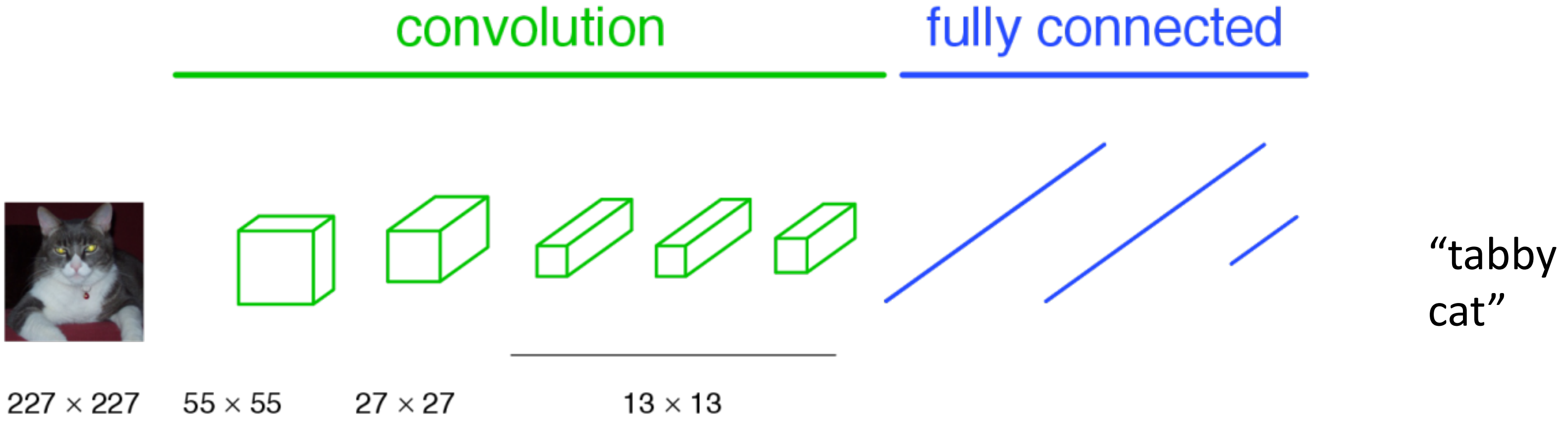
Semantic Segmentation (FCN)



Conv /pool part + transpose convolution

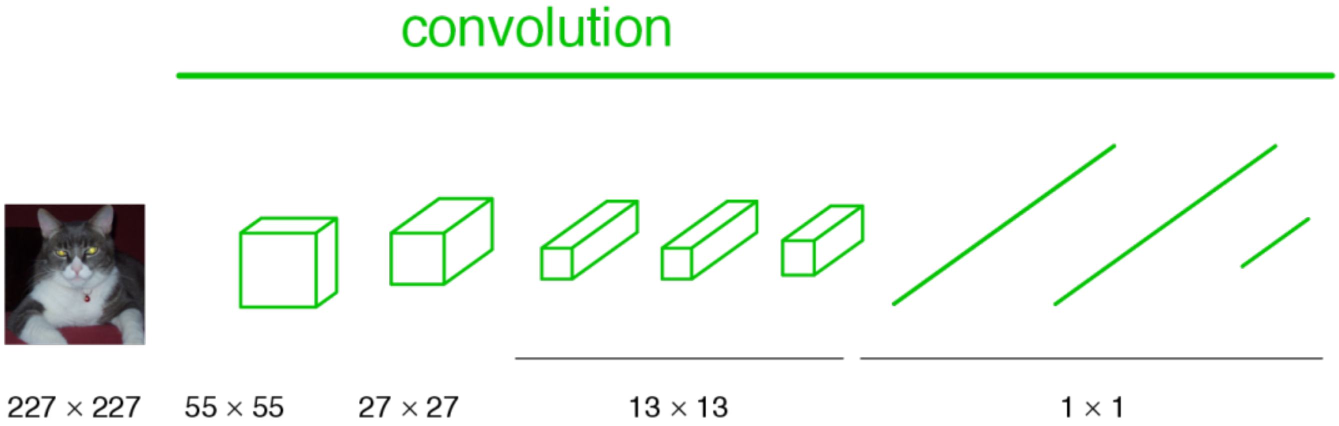
[Long et al. 15] Fully Convolutional Networks for Semantic Segmetnation (FCN)

Classification Network



[Long et al. 15] Fully Convolutional Networks for Semantic Segmentation (FCN)

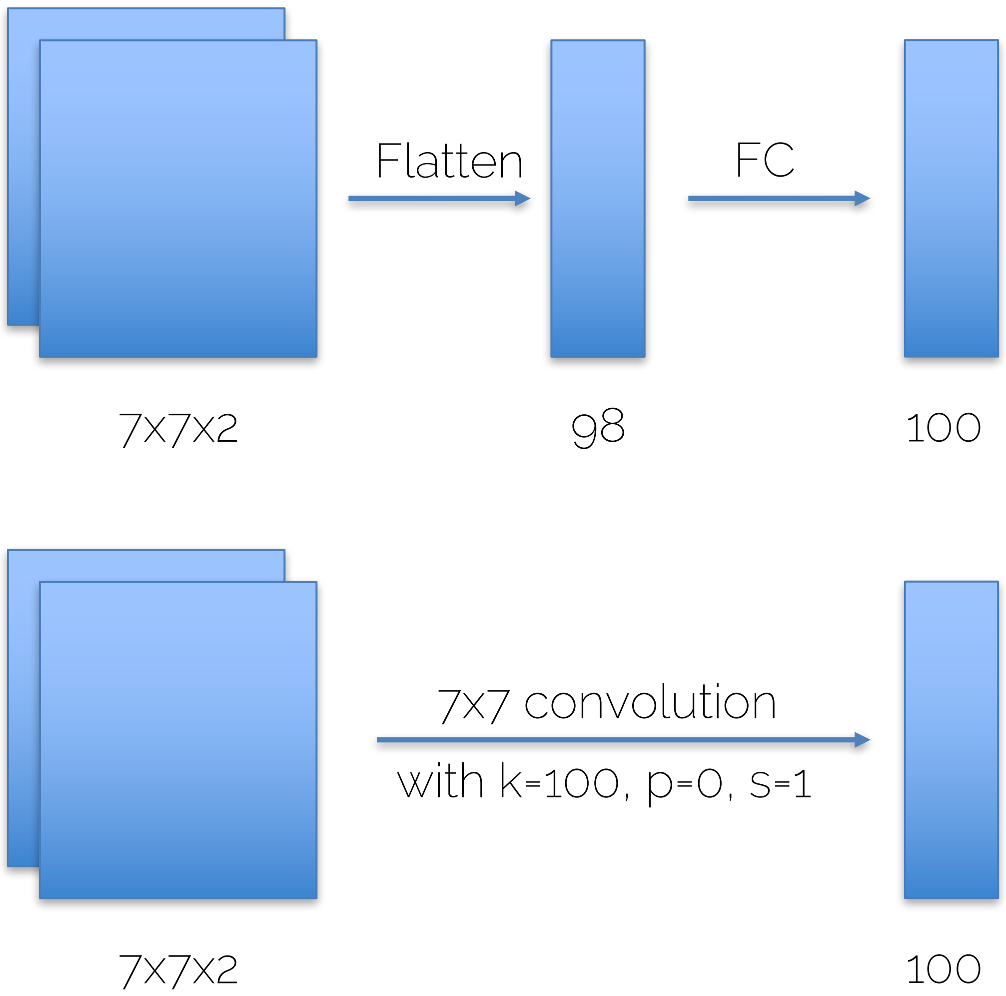
FCN: Becoming Fully Convolutional



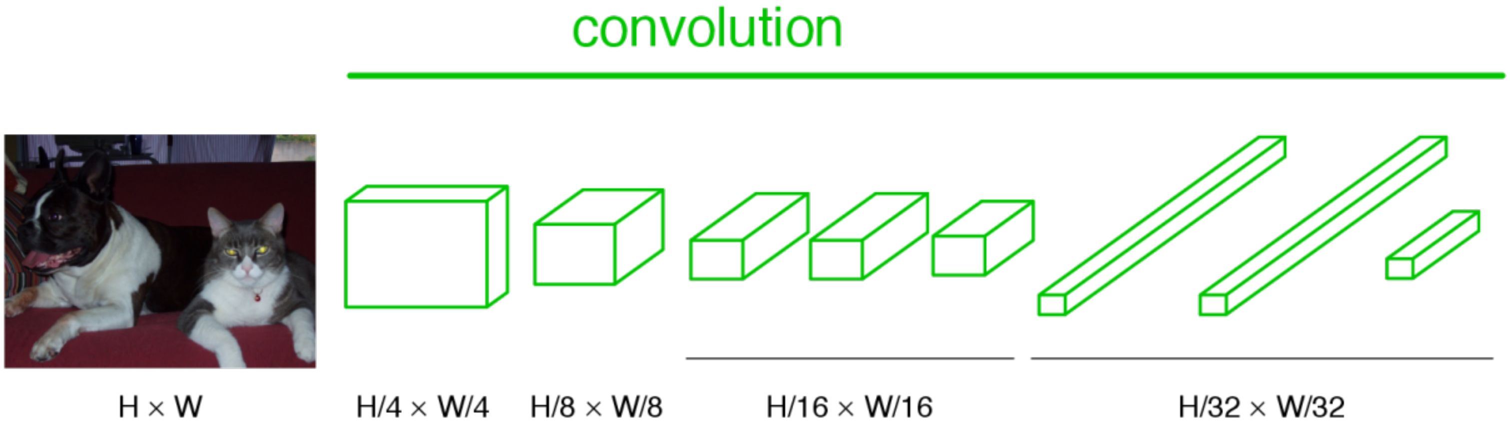
Convert fully connected layers to convolutional layers!

[Long et al. 15] Fully Convolutional Networks for Semantic Segmentation (FCN)

Example: Convert FC to Conv

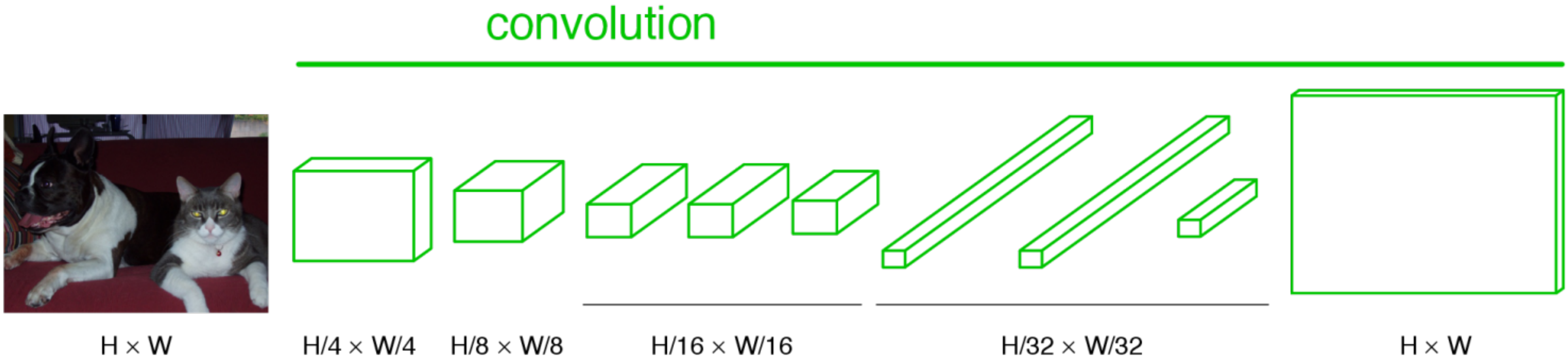


FCN: Becoming Fully Convolutional



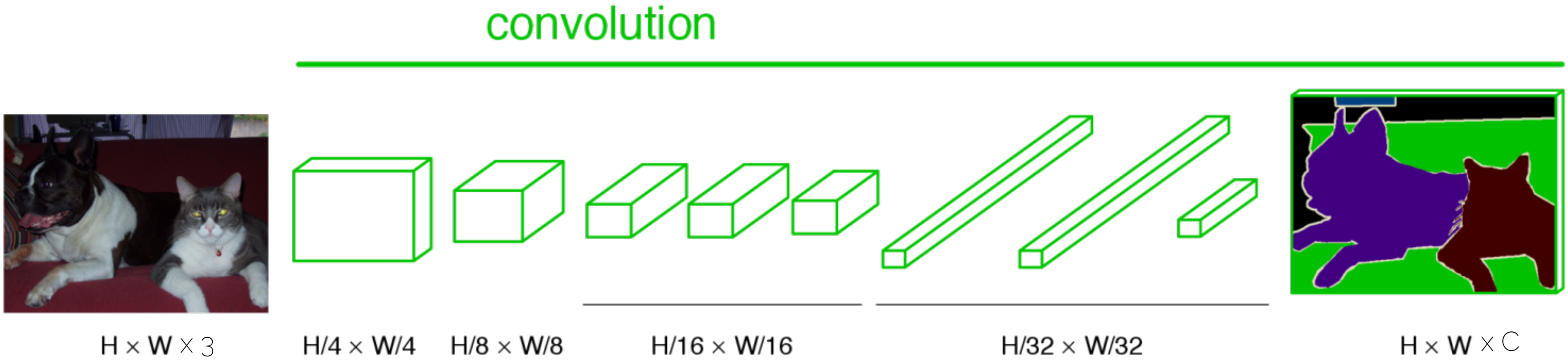
[Long et al. 15] Fully Convolutional Networks for Semantic Segmentation (FCN)

FCN: Upsampling Output



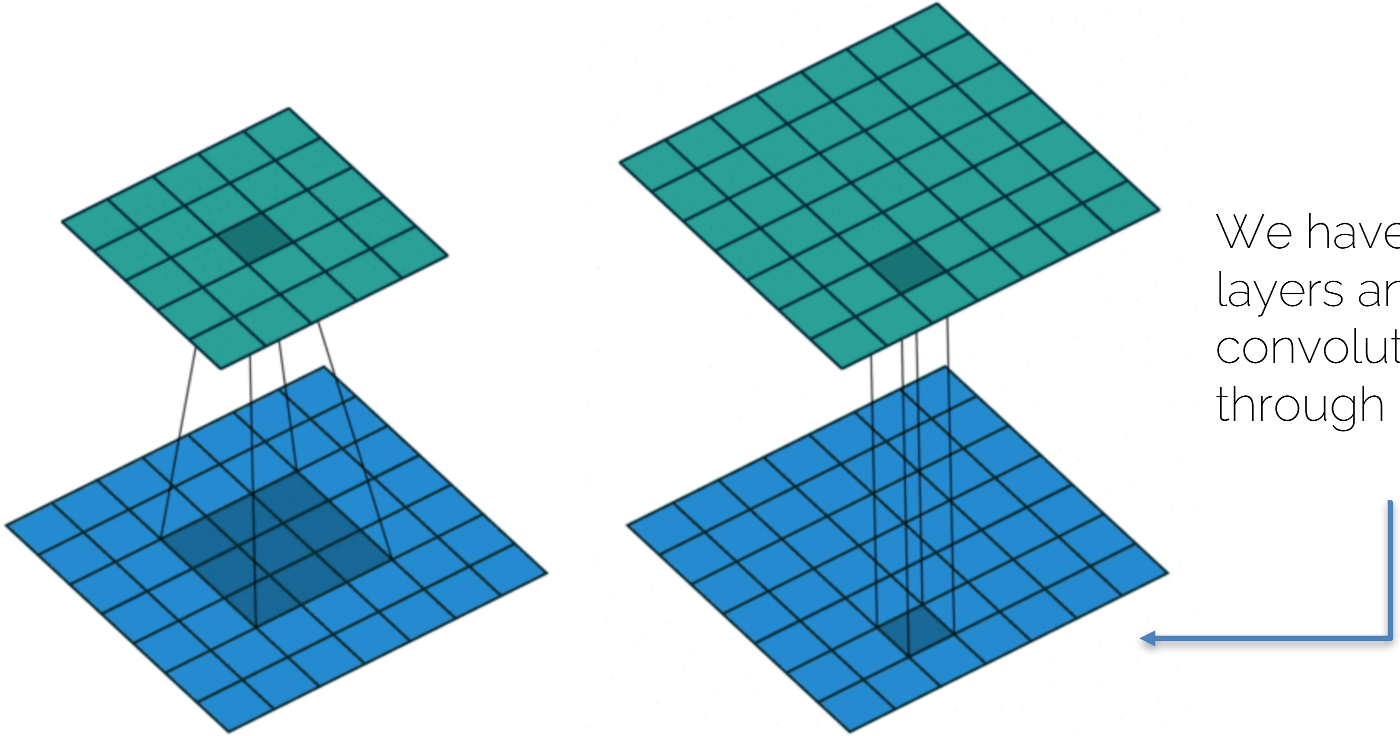
[Long et al. 15] Fully Convolutional Networks for Semantic Segmentation (FCN)

FCN: End-to-end, Pixels-to-pixels Network



[Long et al. 15] Fully Convolutional Networks for Semantic Segmentation (FCN)

Short Aside: 1x1 Convolutions?

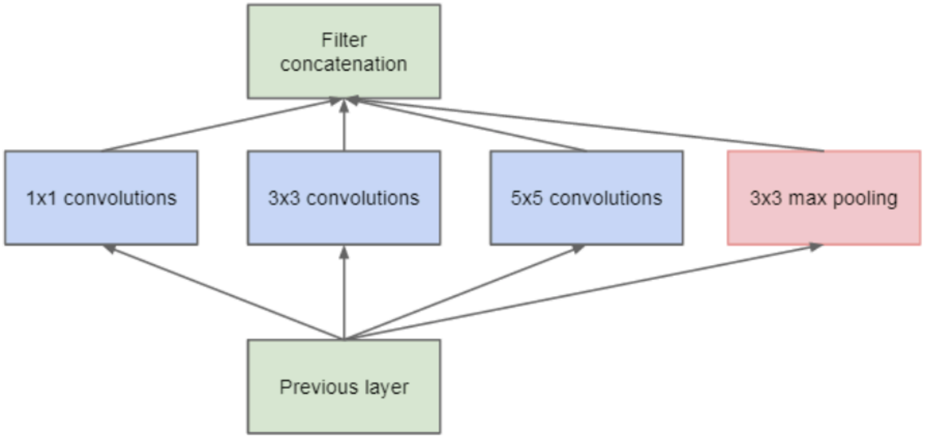


We have multiple layers and convolutions go through all layers

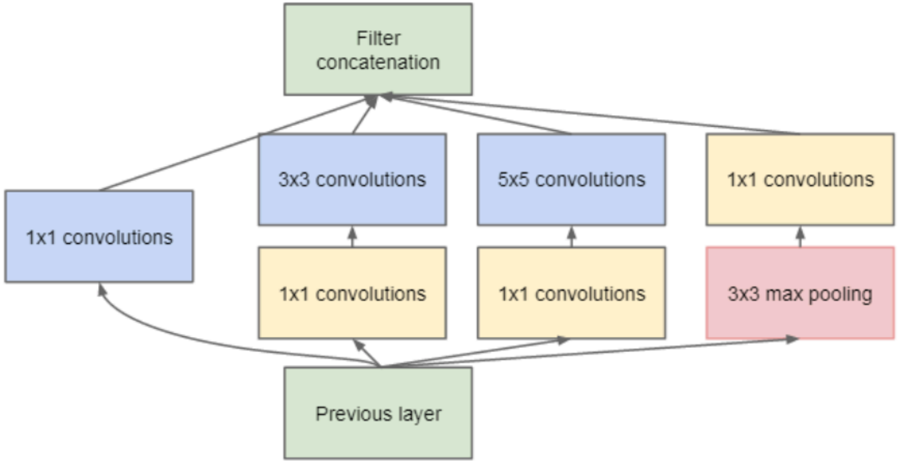
left : **Convolution with kernel of size 3x3** right : **Convolution with kernel of size 1x1**

Credit: Aaditya Prakash

Short Aside: 1x1 Convolutions



(a) Inception module, naïve version



(b) Inception module with dimension reductions

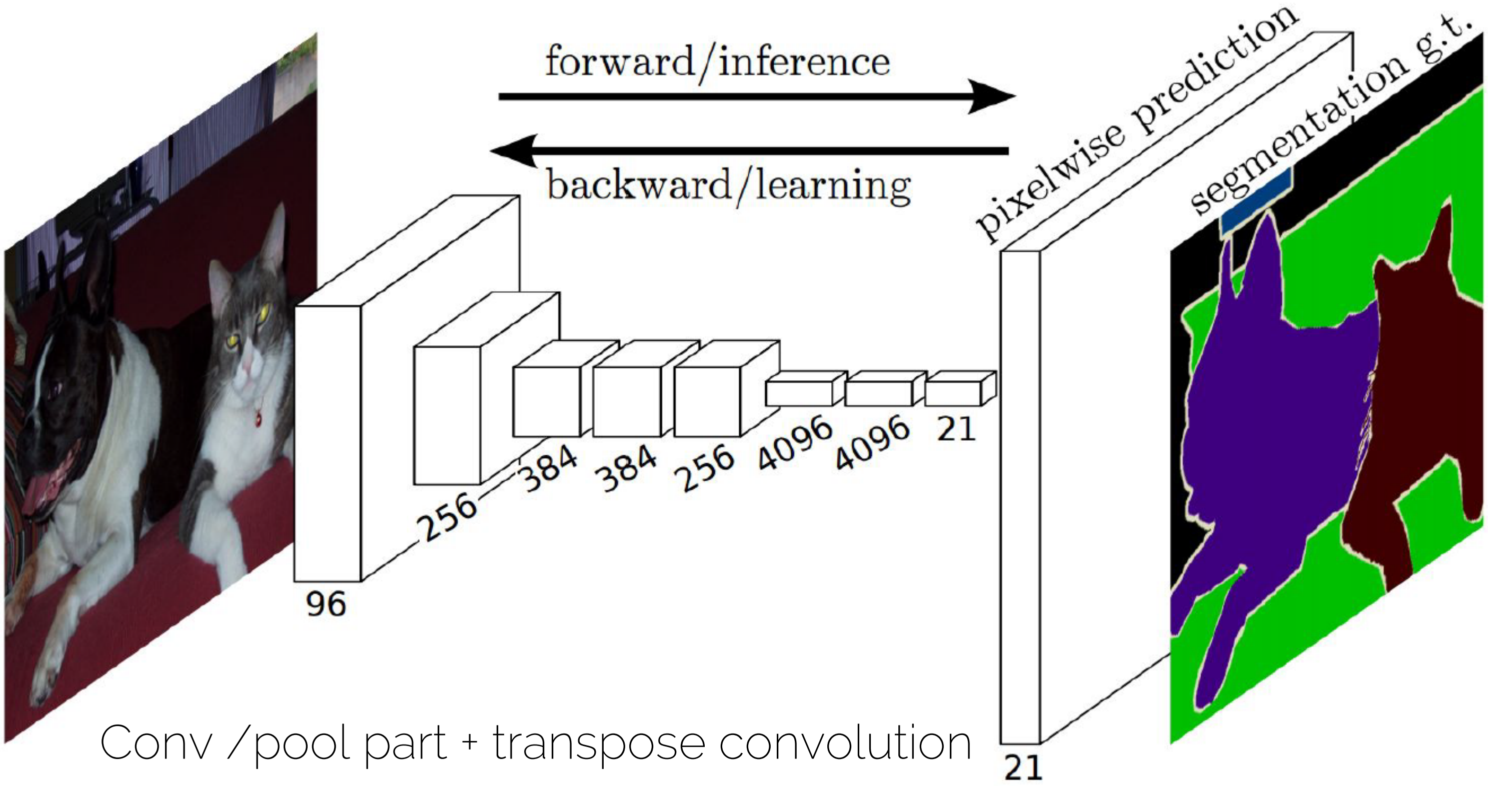
1x1 convolutions in GoogLeNet

Semantic Segmentation (FCN)

- Run "fully convolutional" network (FCN)
- Take all pixels at once as input
- Bottle neck + learnable upsampling
- Predict class for every pixel simultaneously

[Long et al. 15] Fully Convolutional Networks for Semantic Segmetnation (FCN)

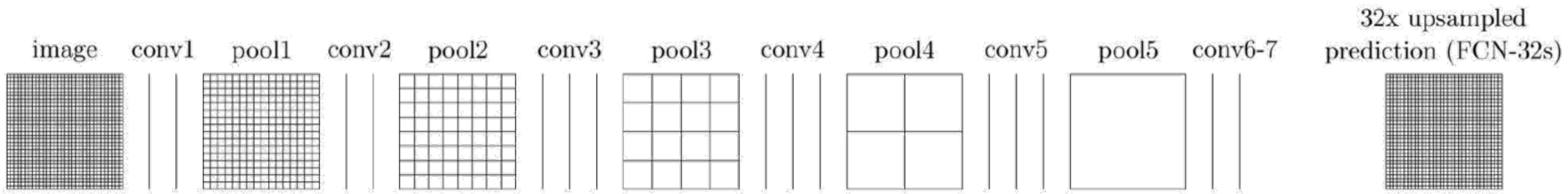
Semantic Segmentation (FCN)



Conv /pool part + transpose convolution

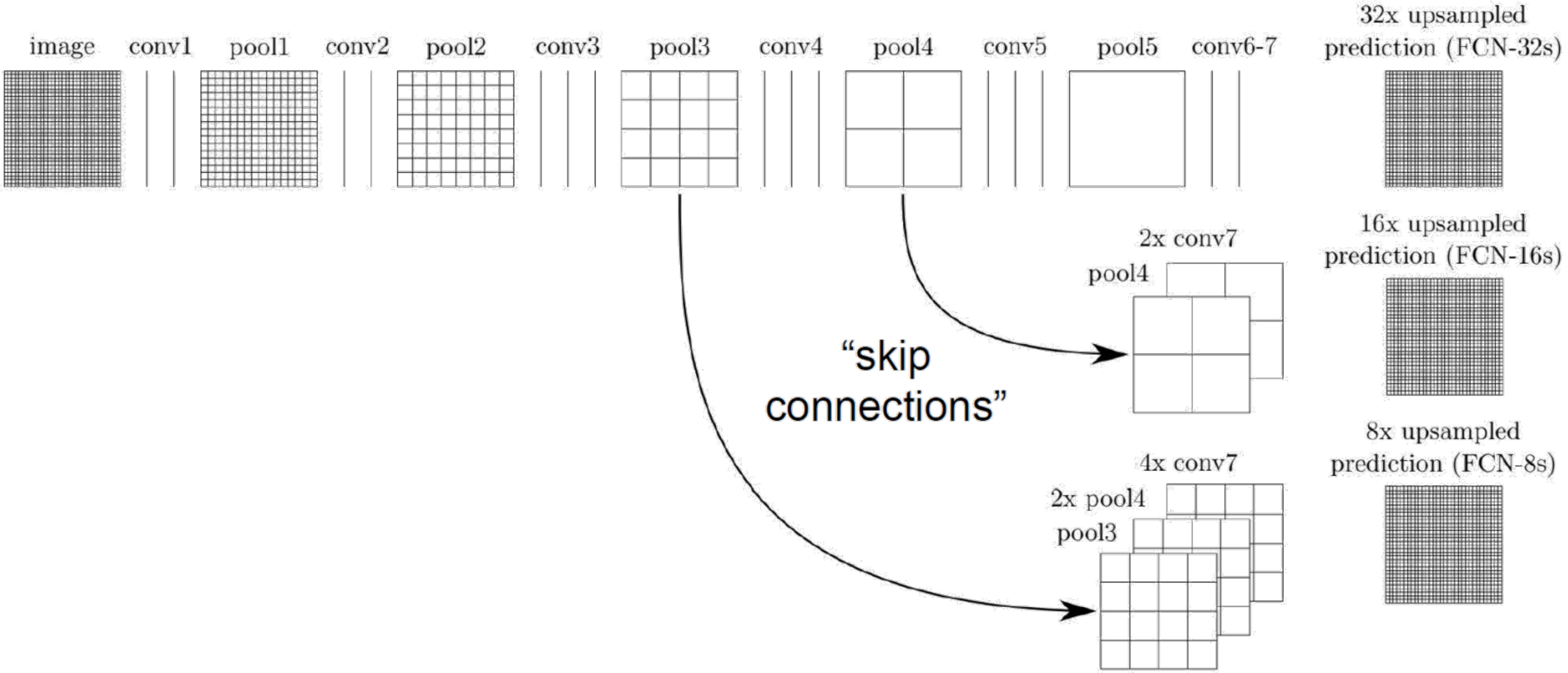
[Long et al. 15] Fully Convolutional Networks for Semantic Segmetnation (FCN)

Semantic Segmentation (FCN)



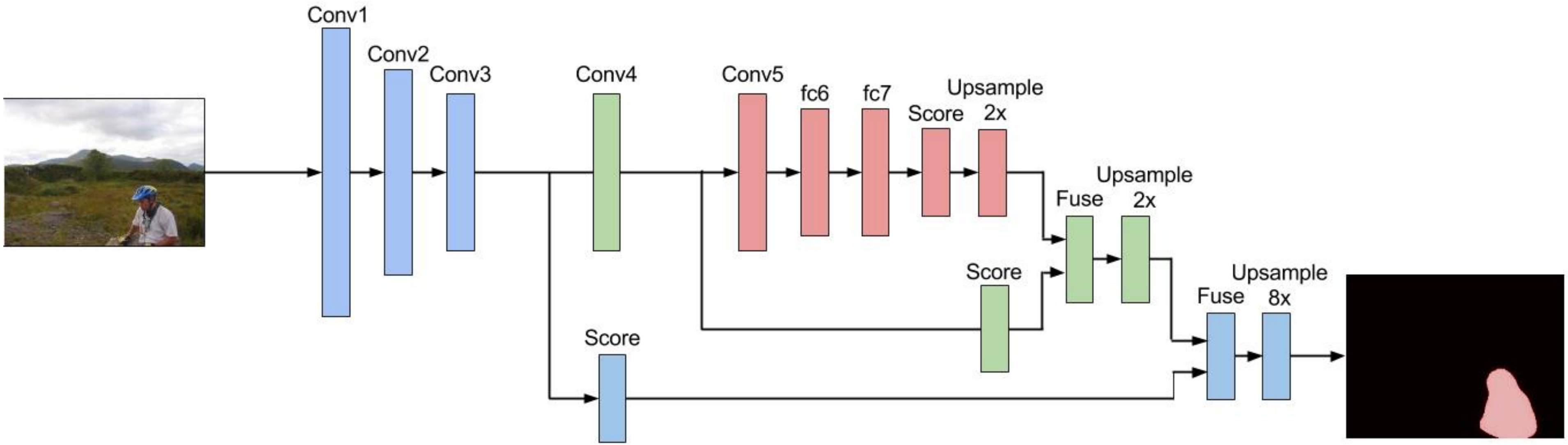
[Long et al. 15] Fully Convolutional Networks for Semantic Segmetnation (FCN)

Semantic Segmentation (FCN)



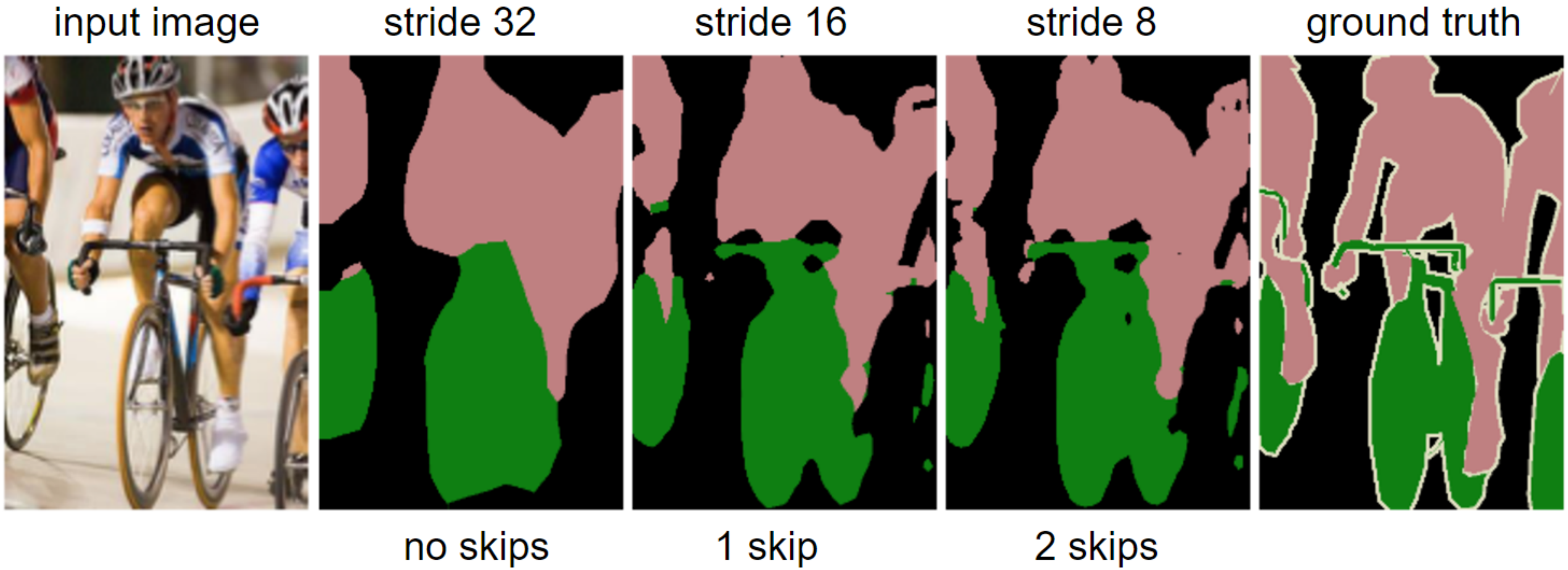
[Long et al. 15] Fully Convolutional Networks for Semantic Segmetnation (FCN)

FCN: Architecture



[Long et al. 15] Fully Convolutional Networks for Semantic Segmentation (FCN)

Semantic Segmentation (FCN)



Skip connections -> better results

[Long et al. 15] Fully Convolutional Networks for Semantic Segmetnation (FCN)

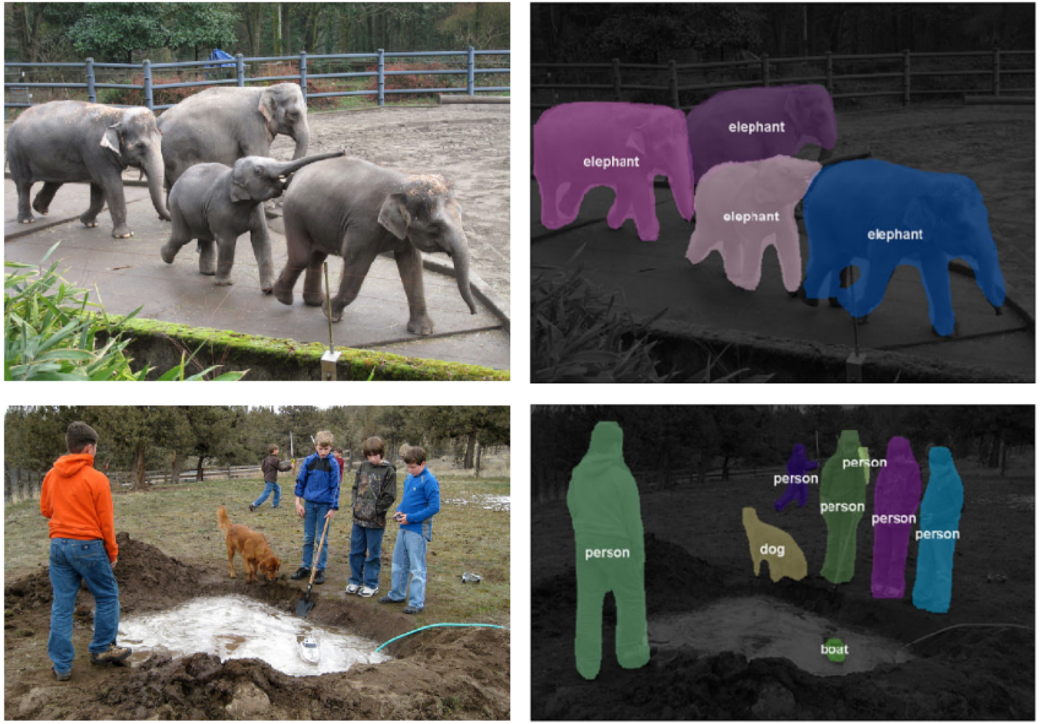
Instance Segmentation

Detect instances, classify category, label pixels of each instance;

Distinguish between instances within a category;
e.g., elephant1, elephant2, etc.

Simultaneous detection and segmentation (SDS)

MS COCO is core dataset
-> lots of work around it



[Dai et al. 15] Instance-aware Semantic Segmentation

Using CNNs in Computer Vision

Classification

CIFAR 10 +
"raw" CNN ☺

Classification + Localization

Regression and/or
sliding window

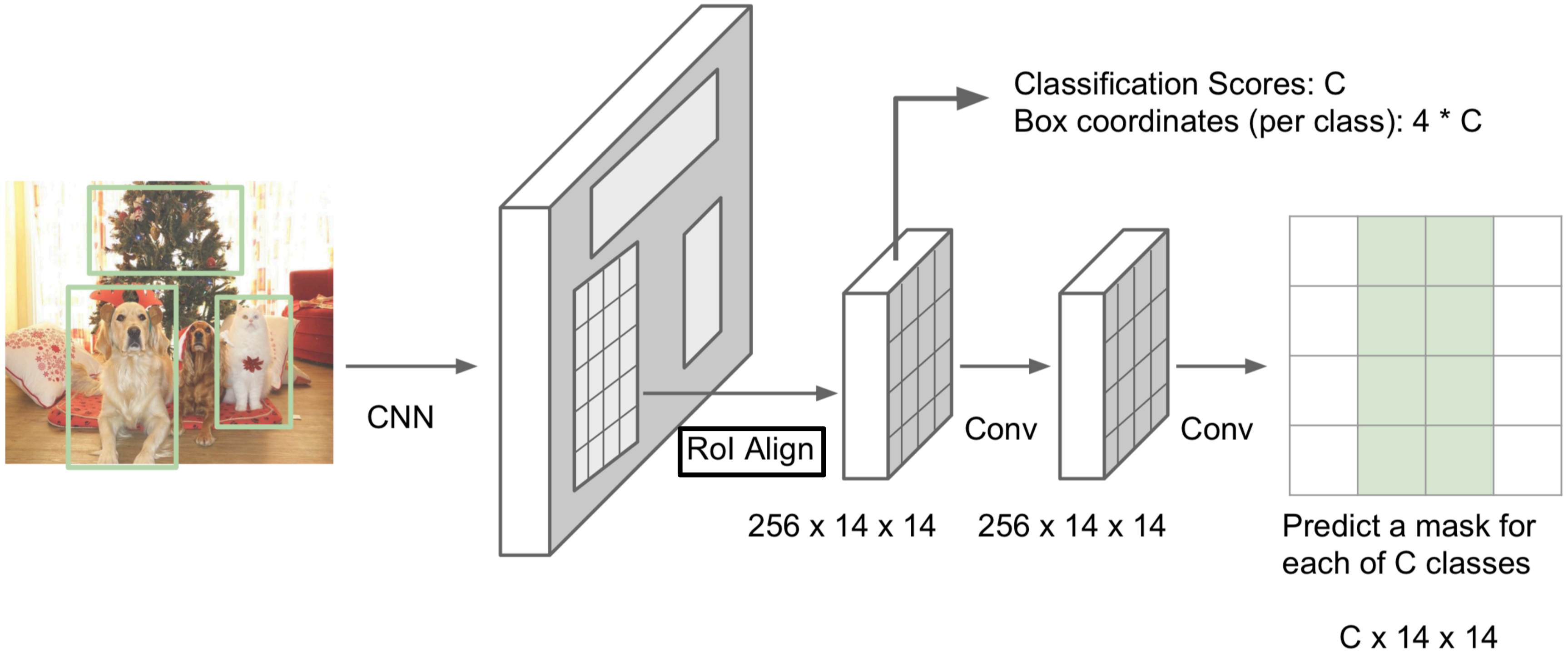
Object Detection

Selective Search, RP
(Fast(er)) R-CNN

Instance Segmentation

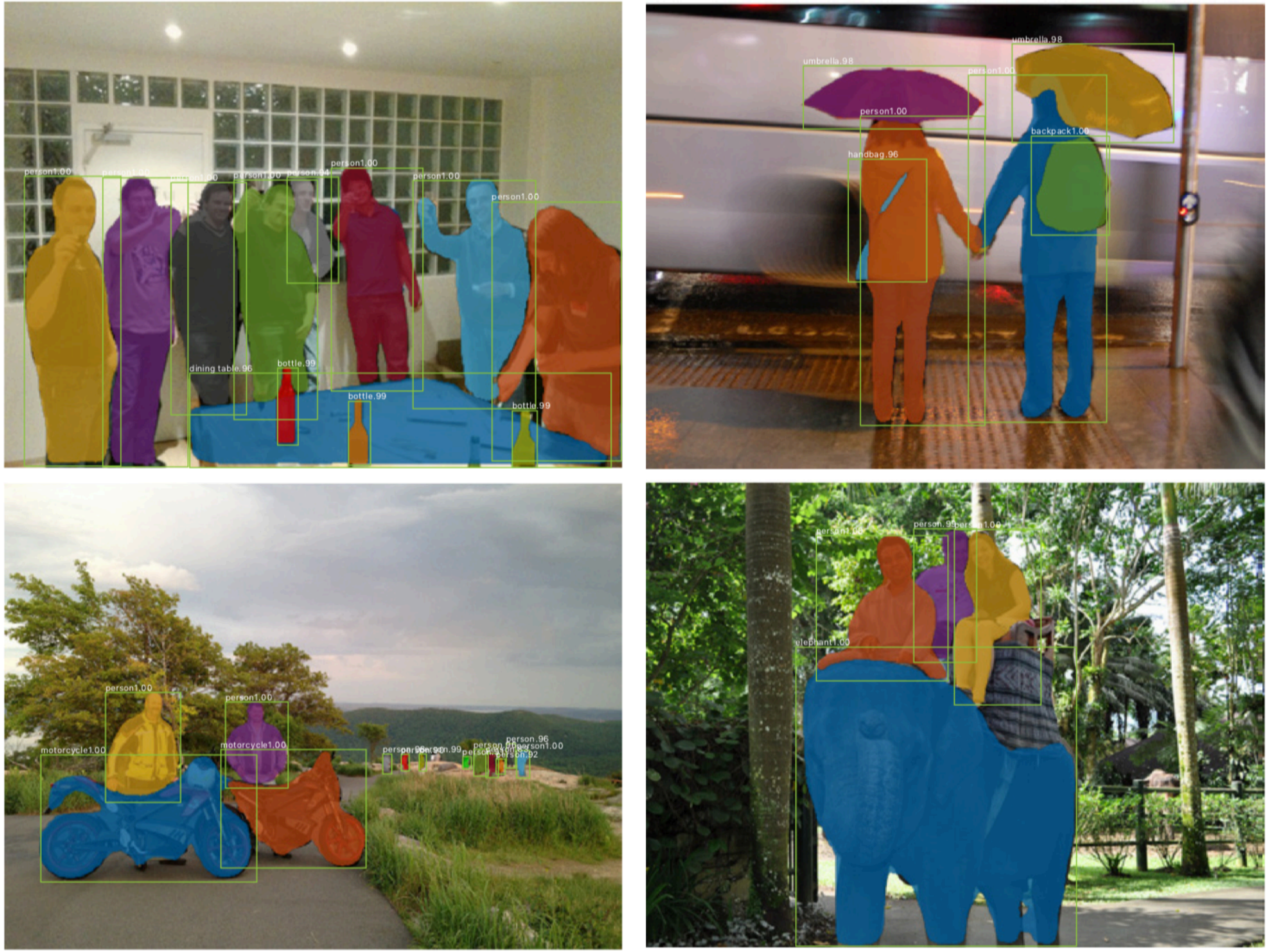
Credit: Li/Karpathy/Johnson

Putting it all together: Mask R-CNN



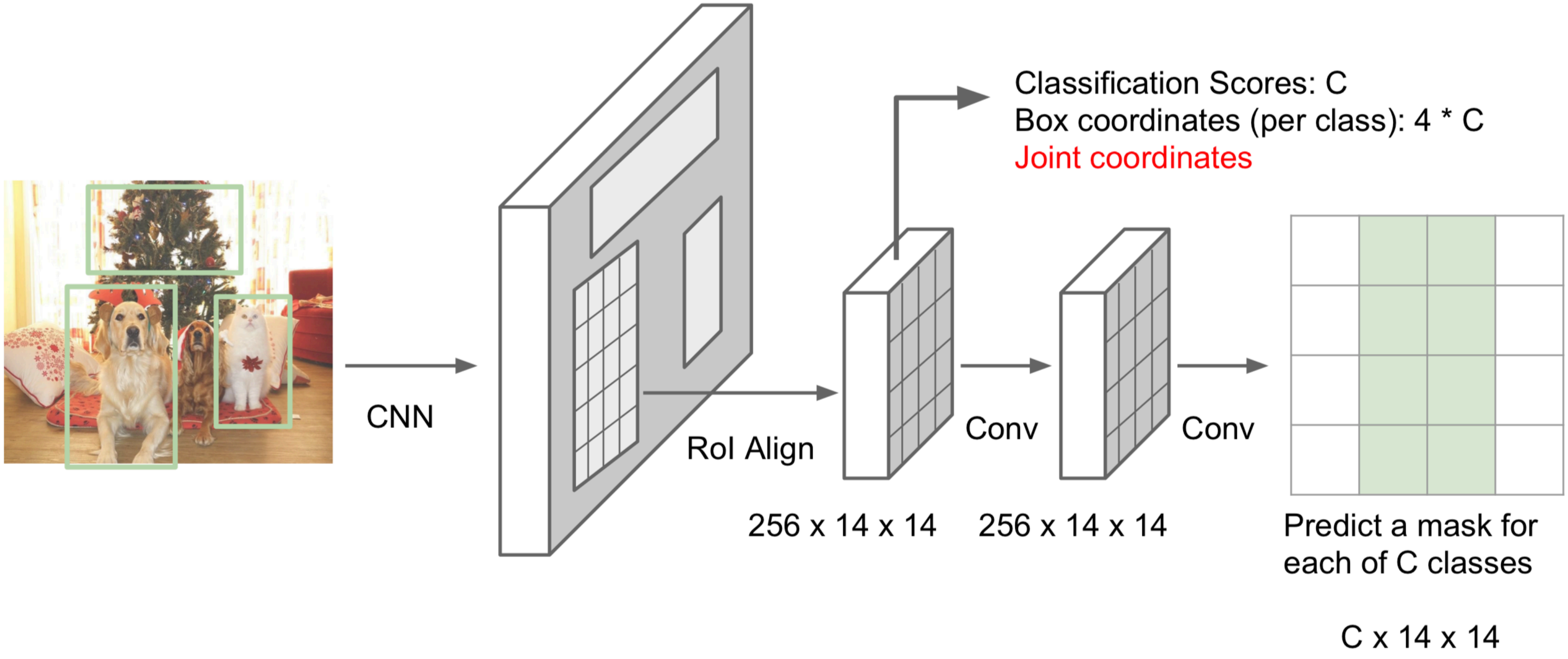
He et al, "Mask R-CNN", arXiv 2017

Credit: Li, Johnson, Yeung



He et al. “Mask R-CNN”

We can also add Pose!



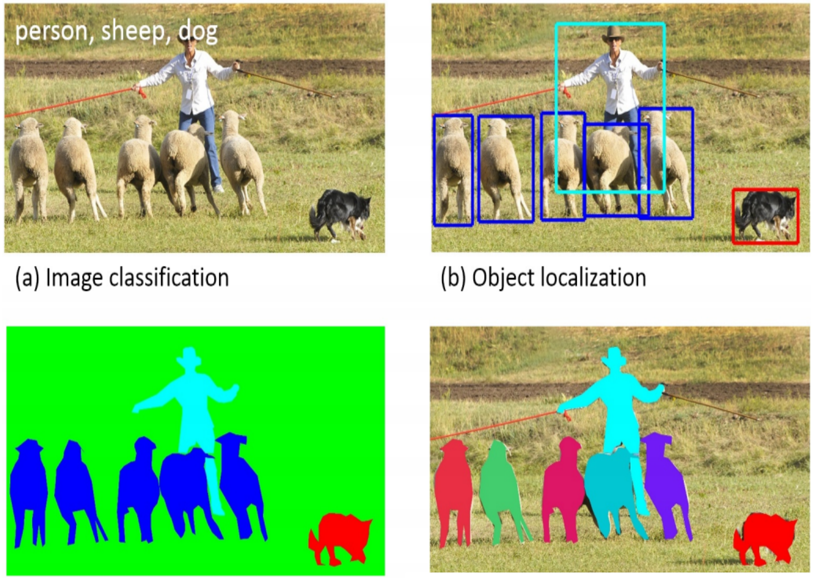
He et al, "Mask R-CNN", arXiv 2017

Credit: Li, Johnson, Yeung

He et al. "Mask R-CNN"

Segmentation Overview

- Semantic segmentation
  - Classify all pixels
  - Fully convolutional models, downsample, then upsample
  - Learnable upsampling (transpose convs
  - Skip connection can help (more later)



(a) Image classification (b) Object localization

(c) Semantic segmentation (d) Instance segmentation

- Instance segmentation
  - Combine object localization with semantic segmentation

Unsupervised Learning: Autoencoders

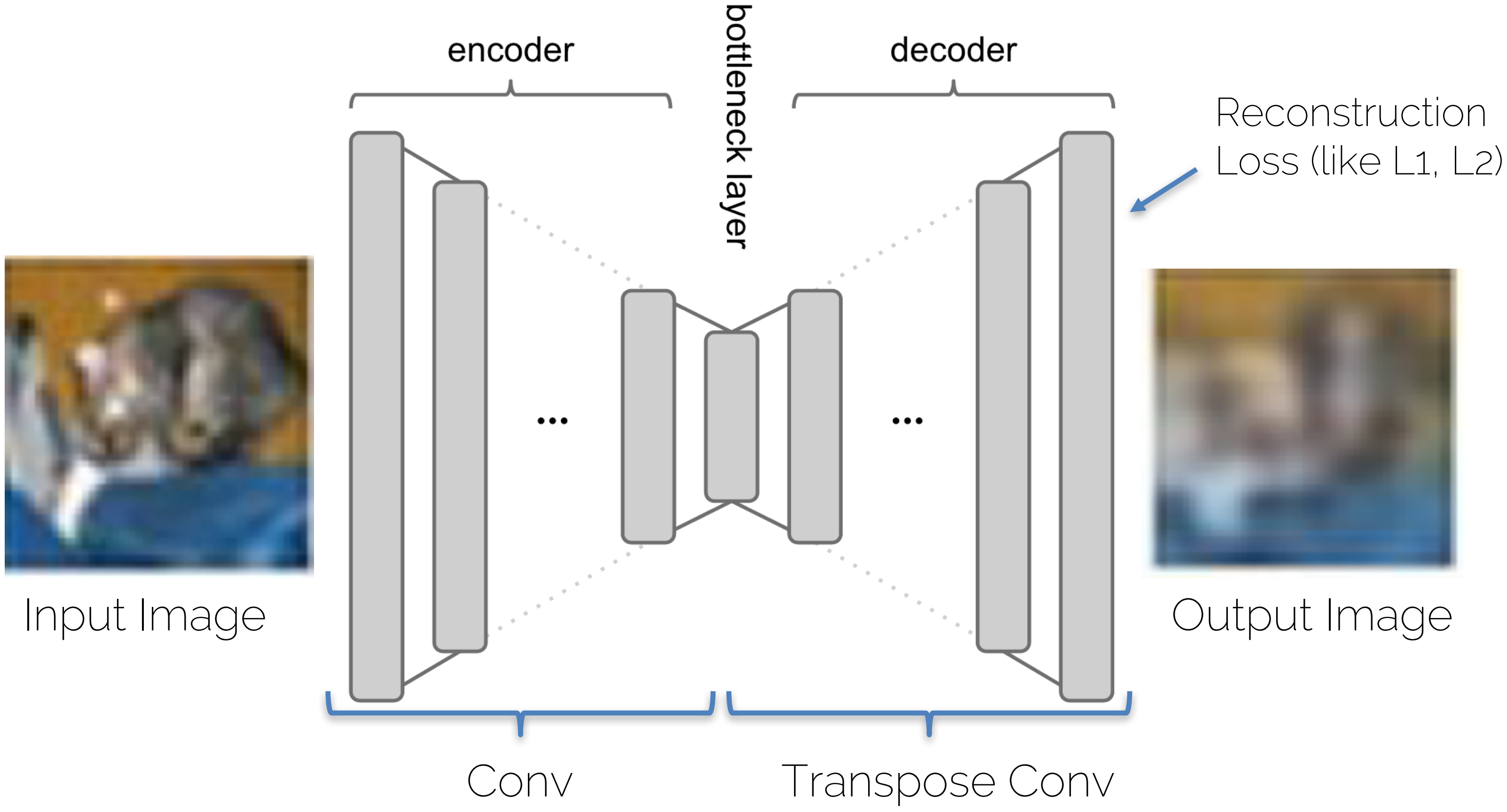
Training Classifiers vs Autoencoders

- Supervised Learning
  - Data (x, y)
    x is data, y is label
  - Goal: learn mapping x -> y
  - Examples: classification, regression

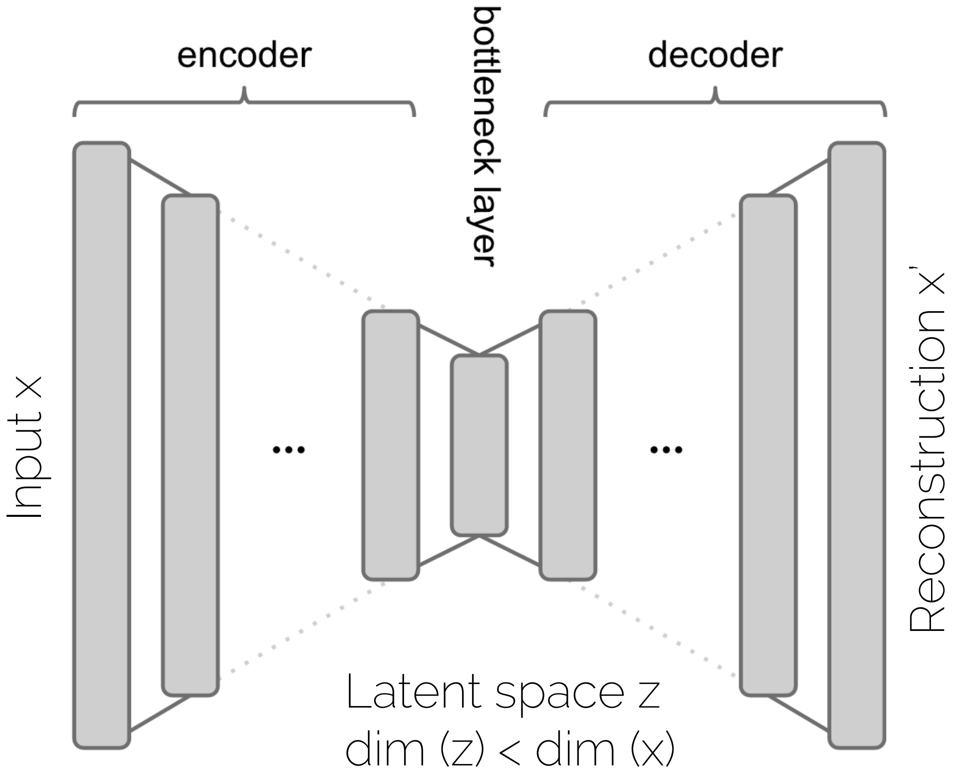
- Unsupervised Learning
  - Data (x)
    only data, no labels
  - Goal: learn structure (e.g., clustering)
  - Example: K-Means clustering, PCA, Autoencoder, density estimation

Super exciting! ☺
("holy grail")

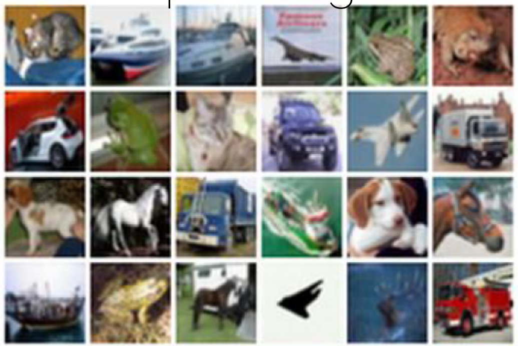
Reconstruction: Autoencoder



Training Autoencoders



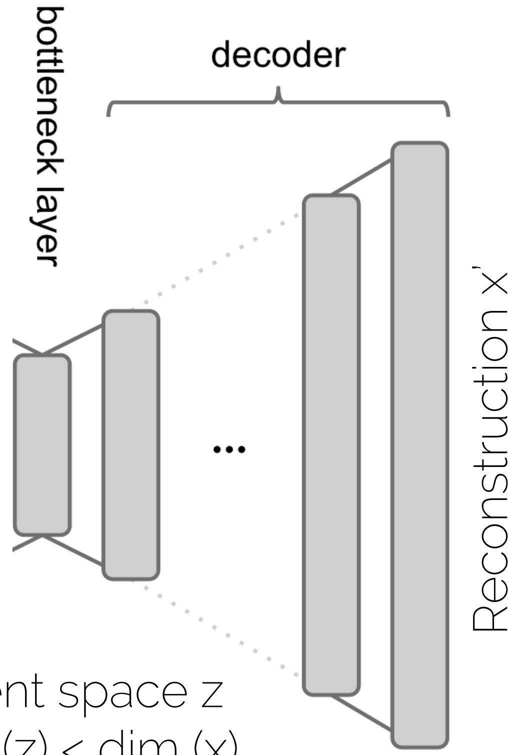
Input images



Reconstructed images

Testing Autoencoders

“Test time”:
-> reconstruction from ‘random’ vector

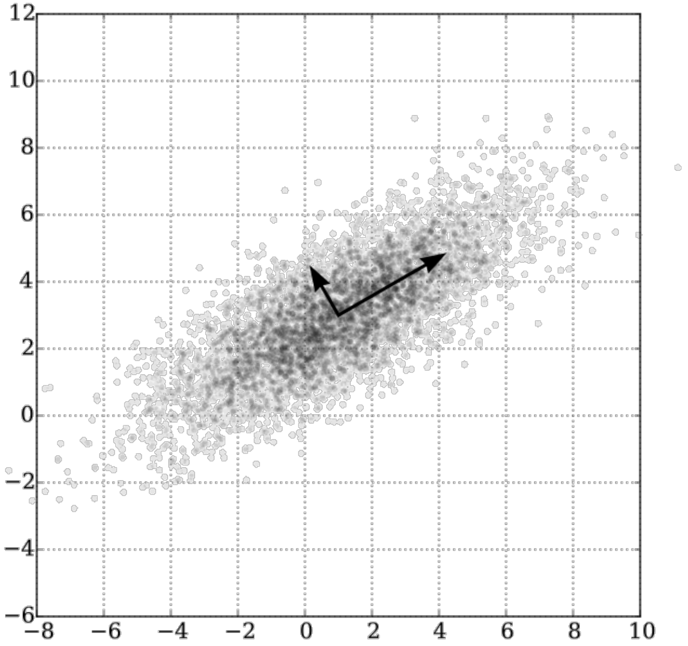


Latent space z
dim (z) < dim (x)

Reconstructed images

Typically pretty blurry… why?

Autoencoder vs PCA

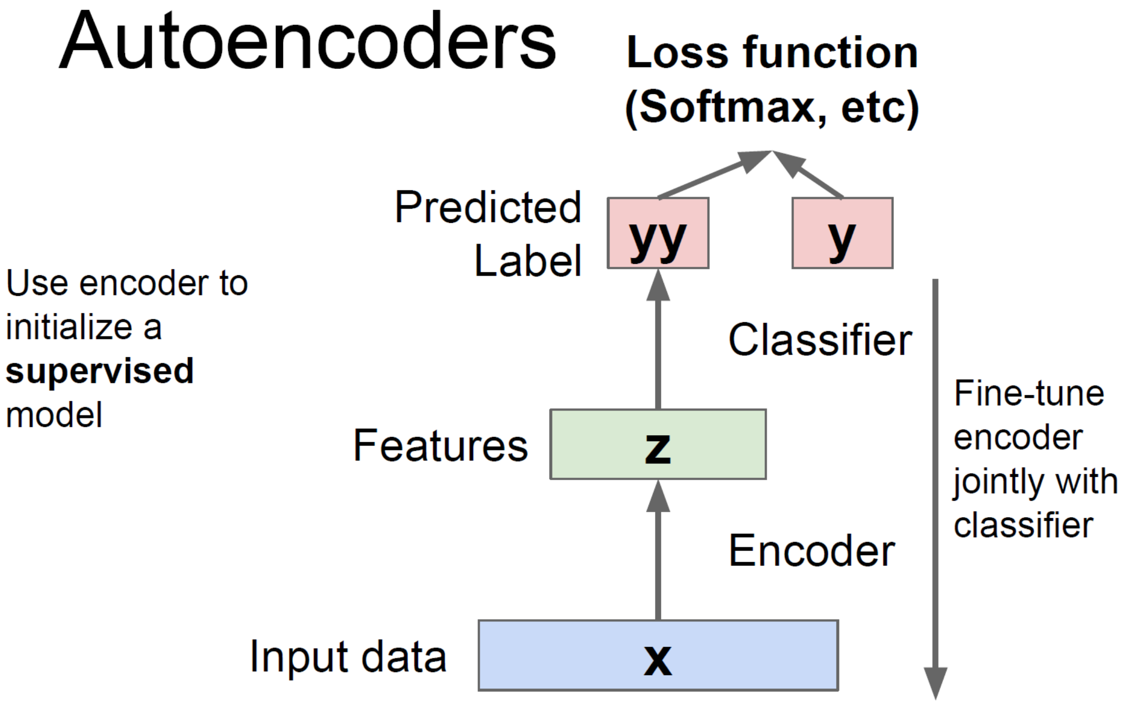


Principal Component Analysis
(low rank approximation)

What is the connection between Autoencoder and PCA?

Autoencoder: Use Cases

- Clustering
- Feature learning
- Embeddings

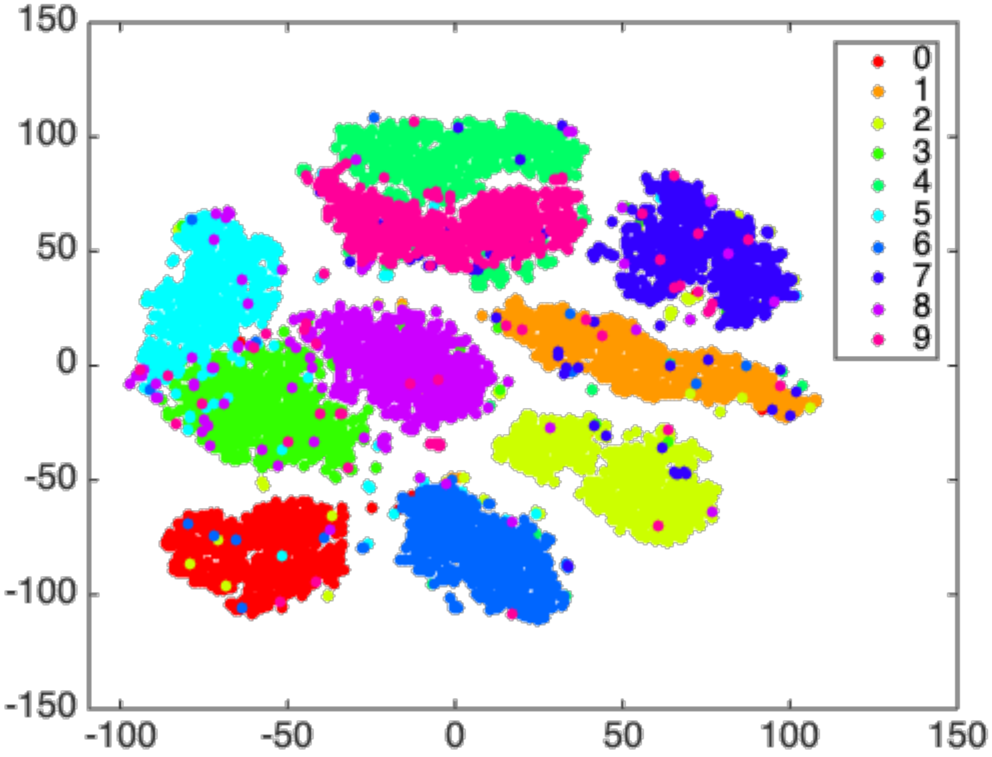


Pre-train AE -> fine-tune with small labeled data

figure by Li/Karpathy/Johnson

Autoencoder: Use Cases

Embedding of
MNIST numbers



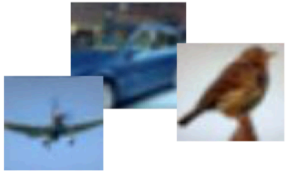
Autoencoder: Use Cases

3D shape embedding

Outlook: Lecture 10

Outlook Thursday 12.01.18

- Generative models
  - Given training data, generate new samples from same distribution



Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

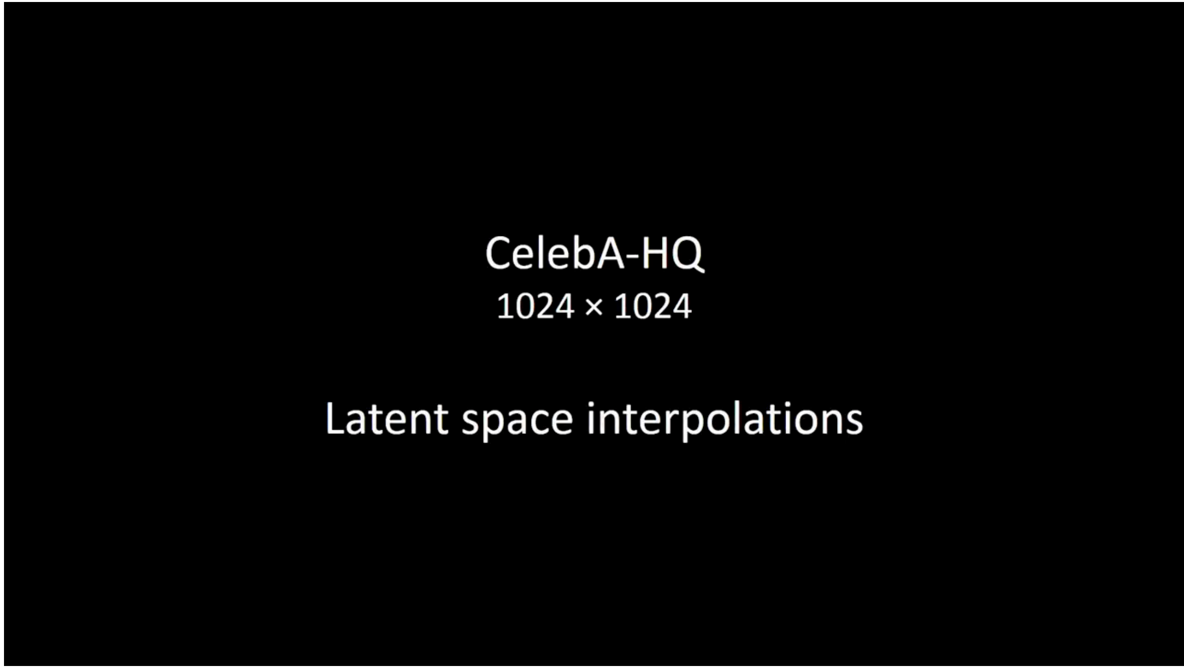
Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

  - Usually start from a random vector

Why Generative Models?

- Realistic samples for artwork, super-resolution, colorization, etc.
- Generative models of time-series data can be used for simulation and planning (reinforcement learning applications!)
- Training generative models can also enable inference of latent representations that can be useful as general features

Basicall we do cool stuff like this…



Credit: Nvidia (Progressive Growing of GANs), Yijun Li (Universal Style Transform), Alec Radford (DcG