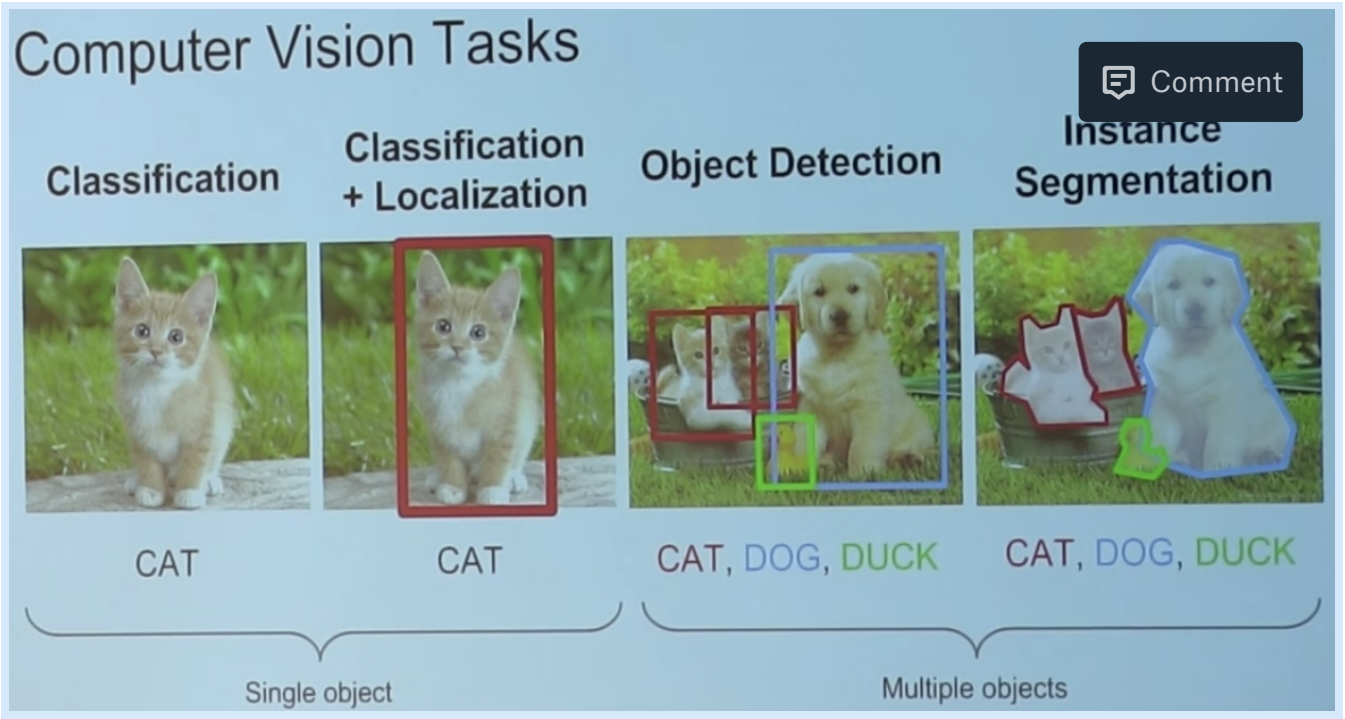
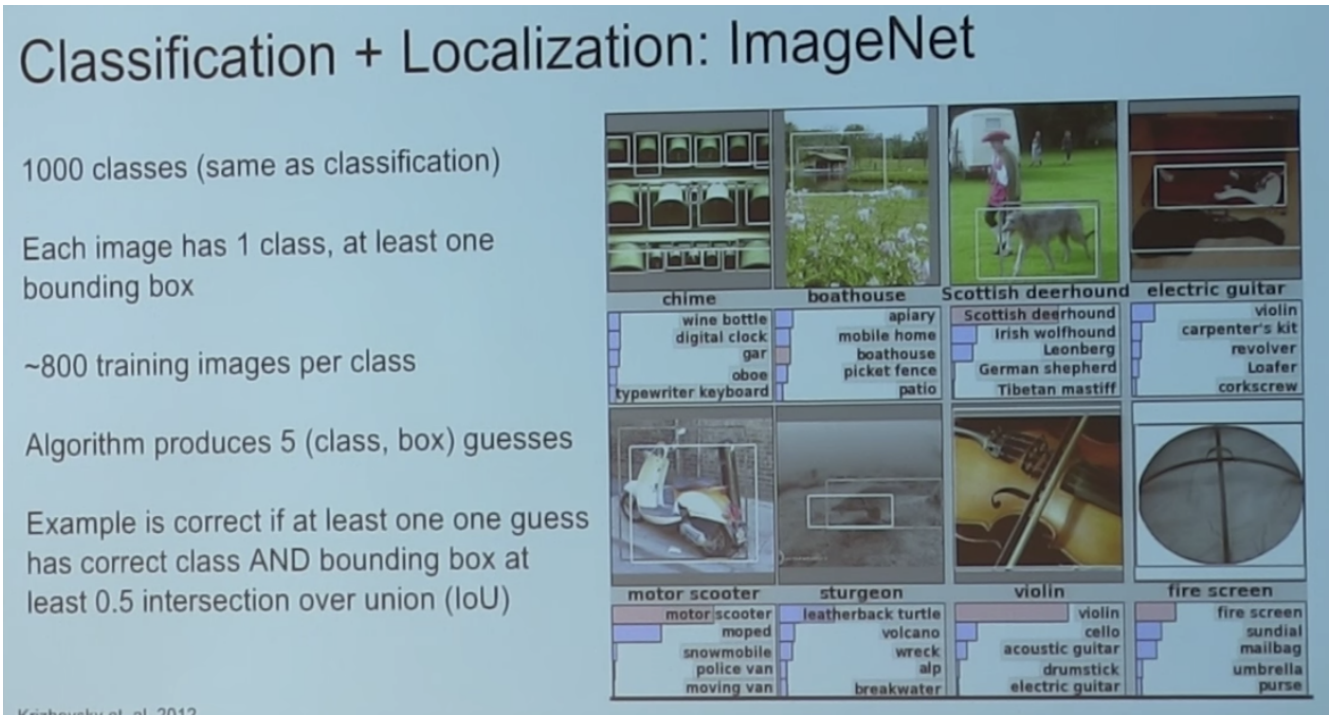
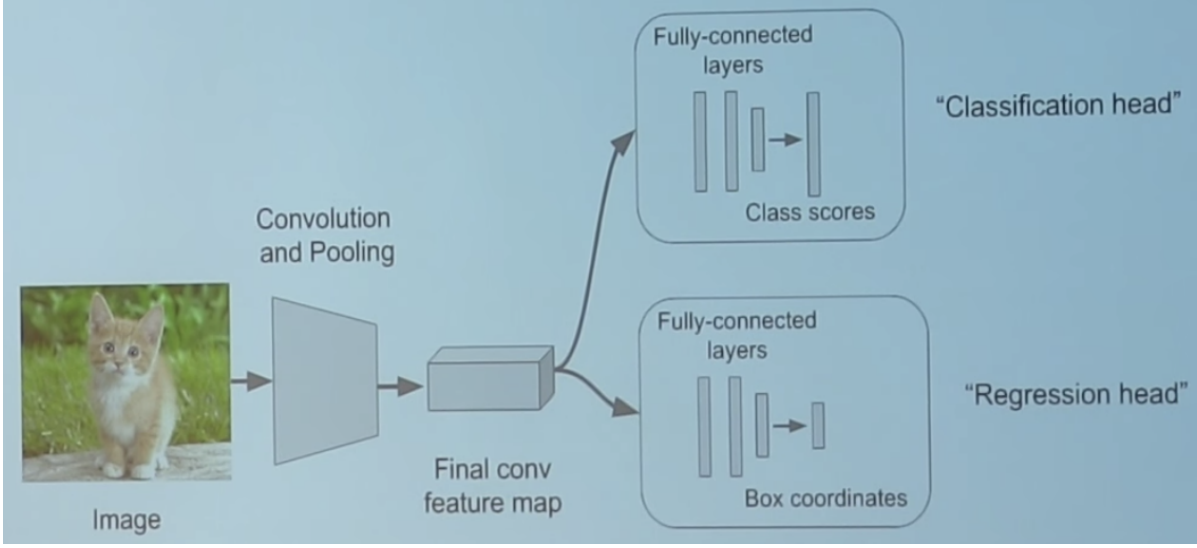
segmentation & attention





Simple Recipe for Classification + Localization

Step 2: Attach new fully-connected "regression head" to the network

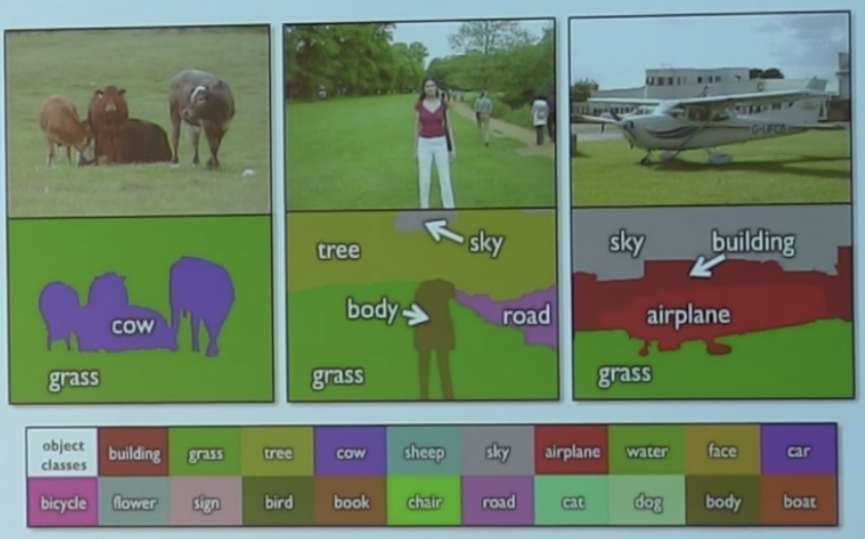


Semantic Segmentation

Label every pixel!

Don't differentiate instances (cows)

Classic computer vision problem

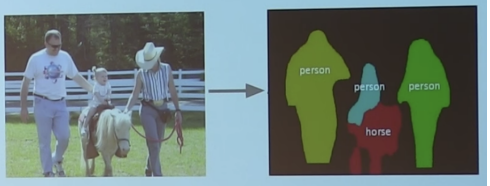


Instance Segmentation

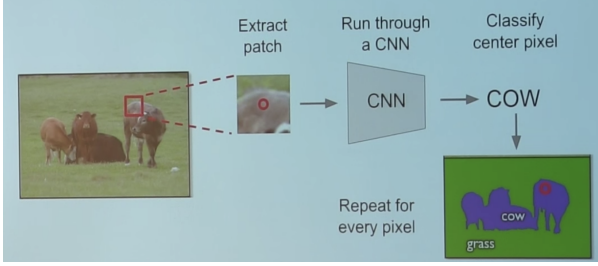
Detect instances, give category, label pixels

"simultaneous detection and segmentation" (SDS)

Lots of recent work (MS-COCO)



Semantic Segmentation



Semantic Segmentation

Run "fully convolutional" network to get all pixels at once

CNN

Smaller output due to pooling

Learnable Upsampling: "Deconvolution"

3 x 3 "deconvolution", stride 2 pad 1

Sum where output overlaps

Same as backward pass for normal convolution!

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

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

Input gives weight for filter

Input: 2 x 2

Output: 4 x 4

Learnable Upsampling: "Deconvolution"

3 x 3 "deconvolution", stride 2 pad 1

Sum where output overlaps

Input gives weight for filter

Input: 2 x 2

Output: 4 x 4

Segmentation Overview

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

Soft Attention for Captioning

Distribution over L locations

Distribution over vocab

a1 a2 d1

CNN

h0 h1

Image: H x W x 3

Features: L x D

Weighted features: D

z1 y1 z2

Weighted combination of features

First word

Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Soft Attention for Captioning

Distribution over L locations

Distribution over vocab

a1 a2 d1 a3 d2

CNN

h0 h1 h2 • • •

Image: H x W x 3

Features: L x D

Weighted features: D

z1 y1 z2 y2

Weighted combination of features

First word

Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Soft vs Hard Attention

CNN

Image: H x W x 3

Grid of features (Each D-dimensional)

a b c d

From RNN:

p_a p_b p_c p_d

Distribution over grid locations

$p_a + p_b + p_c + p_c = 1$

Context vector z (D-dimensional)

Soft attention:
Summarize ALL locations
$z = p_a a + p_b b + p_c c + p_d d$

Derivative dz/dp is nice!
Train with gradient descent

Hard attention:
Sample ONE location according to p, z = that vector

With argmax, dz/dp is zero almost everywhere …
Can't use gradient descent; need reinforcement learning

Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Spatial Transformer Networks

Can we make this function differentiable?

Input image: H x W x 3

Box Coordinates: (xc, yc, w, h)

Cropped and rescaled image: X x Y x 3

Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

$\mathcal{T}_\theta(G)$

Repeat for all pixels in *output* to get a **sampling grid**

Attention Recap

- Soft attention:
  - Easy to implement: produce distribution over input locations, reweight features and feed as input
  - Attend to arbitrary input locations using spatial transformer networks
- Hard attention:
  - Attend to a single input location
  - Can't use gradient descent!
  - Need **reinforcement learning!**