CSE 252D: Advanced Computer Vision

Manmohan Chandraker

Lecture 15: Object Detection

Virtual classrooms

- Virtual lectures on Zoom
  - Only host shares the screen
  - Keep video off and microphone muted
  - But please do speak up (remember to unmute!)
  - Slides uploaded on webpage just before class
- Virtual interactions on Zoom
  - Ask and answer plenty of questions
  - “Raise hand” feature on Zoom when you wish to speak
  - Post questions on chat window
  - Happy to try other suggestions!
- Lectures recorded and upload on Canvas
  - Available under “My Media” on Canvas

Overall goals for the course

- Introduce the state-of-the-art in computer vision
- Study principles that make them possible
- Get understanding of tools that drive computer vision
- Enable one or all of several such outcomes
  - Pursue higher studies in computer vision
  - Join industry to do cutting-edge work in computer vision
  - Gain appreciation of modern computer vision technologies
- This is a great time to study computer vision!

Papers for Wed, May 19

- Context Encoding for Semantic Segmentation
  - https://arxiv.org/abs/1803.08904
- Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation
  - https://arxiv.org/abs/1802.02611
- High-Resolution Representations for Labeling Pixels and Regions
  - https://arxiv.org/abs/1904.04514
- MSeg: A Composite Dataset for Multi-Domain Semantic Segmentation
  - https://ieeexplore.ieee.org/document/9157628

Papers for Fri, May 21

- R-FCN: Object Detection via Region-based Fully Convolutional Networks
  - https://arxiv.org/abs/1605.06409
- Deformable Convolutional Networks
  - https://arxiv.org/abs/1703.06211
- Feature Pyramid Networks for Object Detection
  - https://arxiv.org/abs/1612.03144
- Training Region-Based Object Detectors with Online Hard Example Mining
  - https://arxiv.org/abs/1604.03540

Papers for Wed, May 26

- You Only Look Once: Unified, Real-Time Object Detection
  - https://arxiv.org/abs/1506.02640
- Focal Loss for Dense Object Detection
  - https://arxiv.org/abs/1708.02002
- Single-Shot Refinement Neural Network for Object Detection
  - https://arxiv.org/abs/1711.06897
- CornerNet: Detecting Objects as Paired Keypoints
  - https://arxiv.org/abs/1808.01244

Recap

Output Going to Image Resolution

- Encoder aggressively pools and subsamples image
- Necessary to capture context information which is necessary for segmentation
- But spatial detail is also necessary

- Goal for decoder: obtain output at image *resolution*
- Goal for decoder: recover *detail* in encoder feature maps before subsampling

- Option 1
  - Use transposed convolution to upsample to image resolution
  - Concatenate encoder features to upsampled features during decoding

- Option 2
  - Similar as above, but gradual upsampling to image resolution

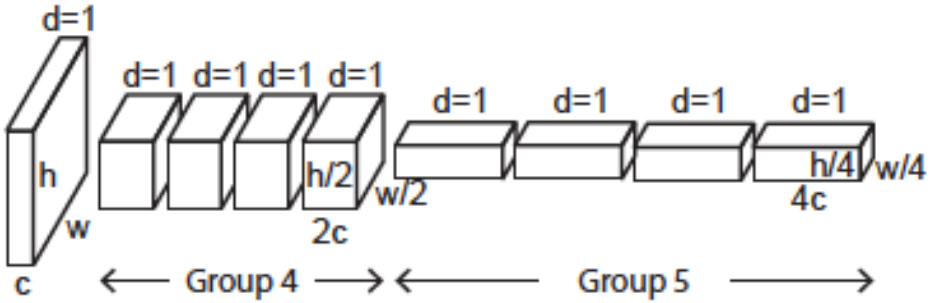
- Option 3
  - Unpool based on encoder locations and convolve to densify

Wide Receptive Fields

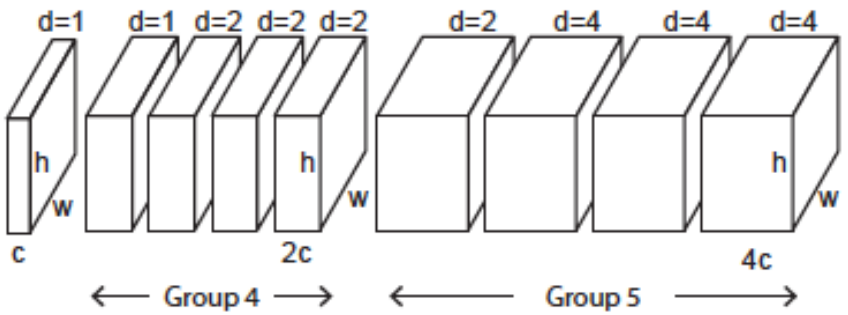
- Most networks have similar encoders (inspired by classification networks)
- Downsample to save memory and obtain large receptive fields
- Consider not downsampling features, but still achieve large receptive fields
  - Once downsampled, signal might be lost for small objects
  - Hard to recover by subsequent layers during training
- Option 1:
  - Can process input image at multiple scales and combine the predictions
- Option 2:
  - Do not let encoder subsample too much in the first place
  - Challenge: maintain spatial resolution along with a large receptive field
  - Solution: use dilated convolutions
- Design goals:
  - A module specifically for dense prediction
  - Rectangular prism of convolutional layers, instead of pyramid
  - No pooling or subsampling.

Dilated Residual Network

- Goals: Exploit the power of residual networks with advantages of dilation
  - Once training signal lost by downsampling, hard to recover it
  - Preserve spatial resolution of feature maps
  - Provide training signals that densely cover the input field
  - Backpropagation can now learn to preserve small but salient details
  - Also beneficial when classification network is transferred to other tasks
- ResNet uses stride 2 to downsample from block G_k to G_{k+1}
- Do not use stride to downsample, maintain resolution
- Dilate subsequent layers by another factor of 2 to maintain receptive field



ResNet



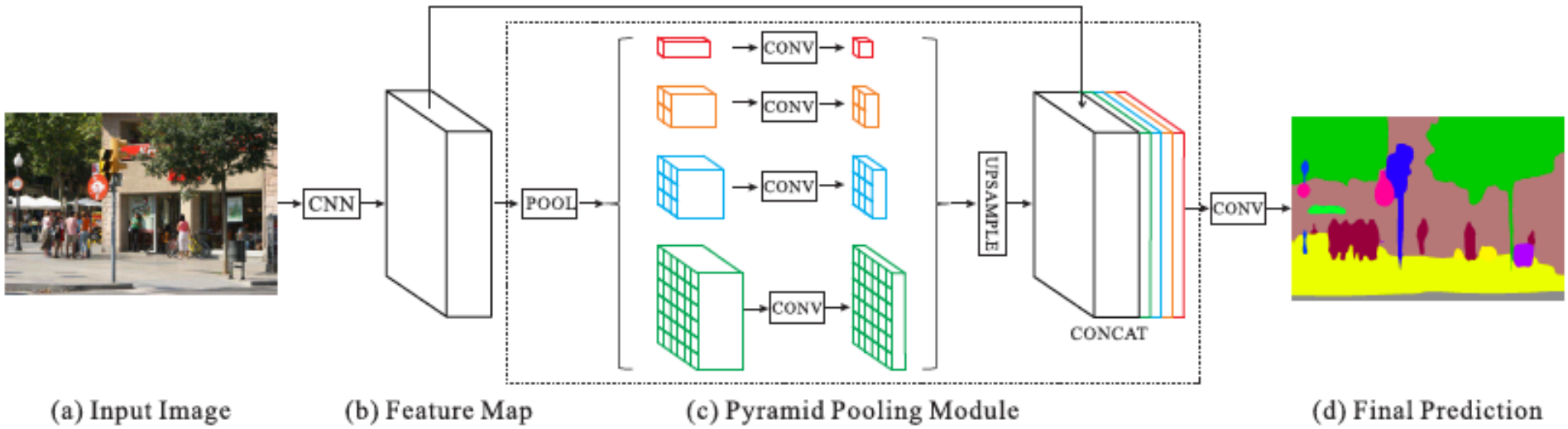
Dilated ResNet

[Yu et al., Dilated Residual Networks]

More Context Information: Pyramid Pooling

- Role in segmentation: Extract both global and regional context
  - A hierarchical global prior, with information across scales and regions

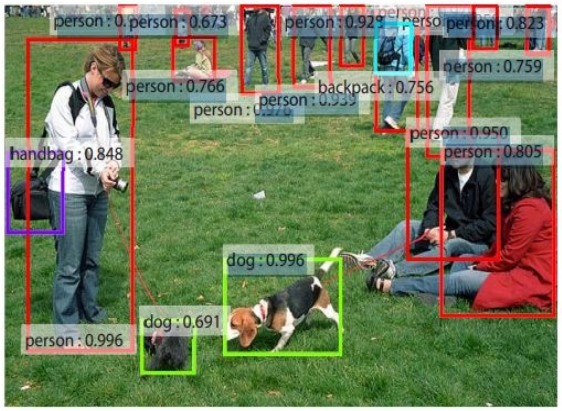
- Fuse features in several different pyramid scales
  - Pool input feature map into hierarchy of context features
  - Do 1x1 convolution to ensure each pyramid level gets equal weight
  - Upsample to original resolution and concatenate

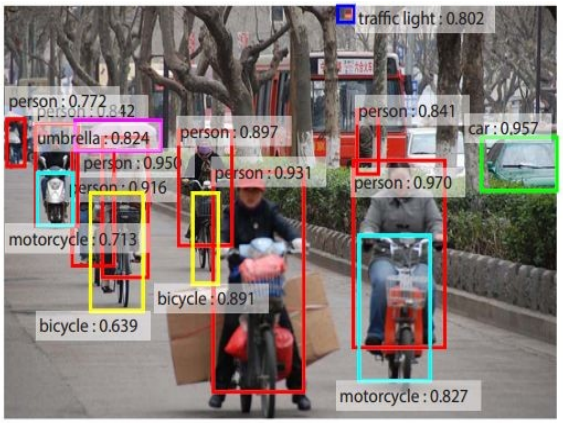


[Zhao et al., PSPNet]

Object Detection

Object Detection





[Ren et al., Faster R-CNN]

Object Detection

Object 1: (x_1, y_1, w_1, h_1), dog

Object 2: (x_2, y_2, w_2, h_2), cat

Object Detection

Object 1: (x_1, y_1, w_1, h_1), dog

Object 2: (x_2, y_2, w_2, h_2), cat

Object 3: (x_3, y_3, w_3, h_3), dog

Object Detection

Object 1: (x_1, y_1, w_1, h_1), dog

Object 2: (x_2, y_2, w_2, h_2), cat

Object 3: (x_3, y_3, w_3, h_3), dog

Object 4: (x_4, y_4, w_4, h_4), cat

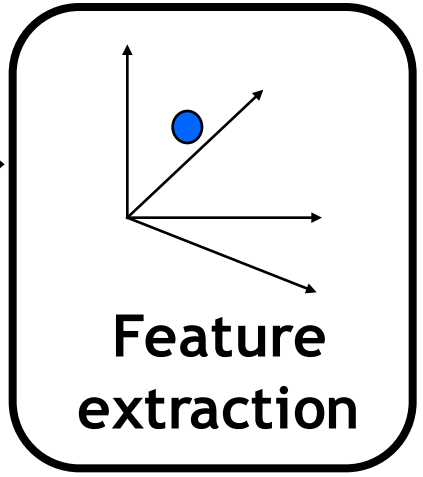
Object 5: (x_5, y_5, w_5, h_5), dog

Need to handle outputs of variable lengths

Object Detection: Sliding Windows

Given new image:

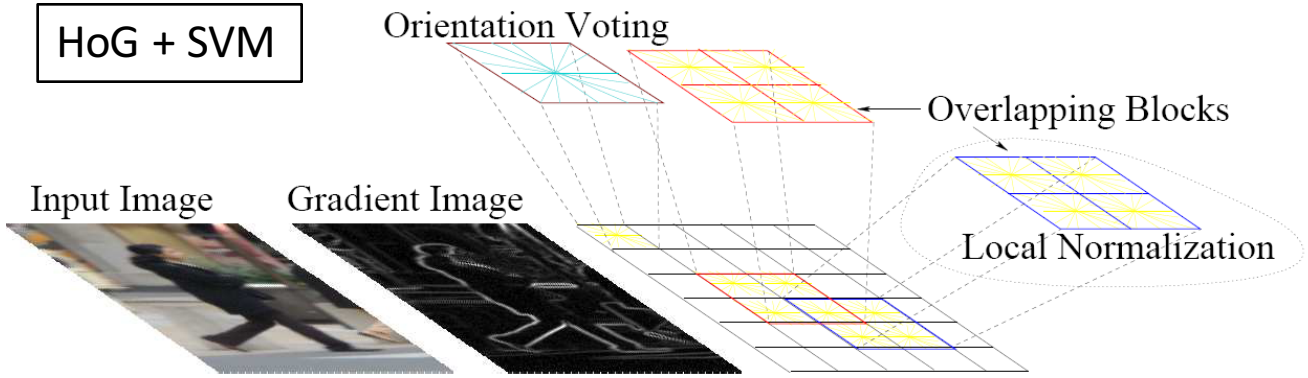
1. Slide window
2. Score by classifier

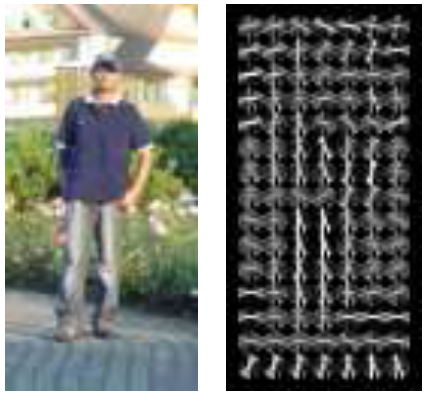


Car or non-car Classifier

Object Detection: Sliding Windows

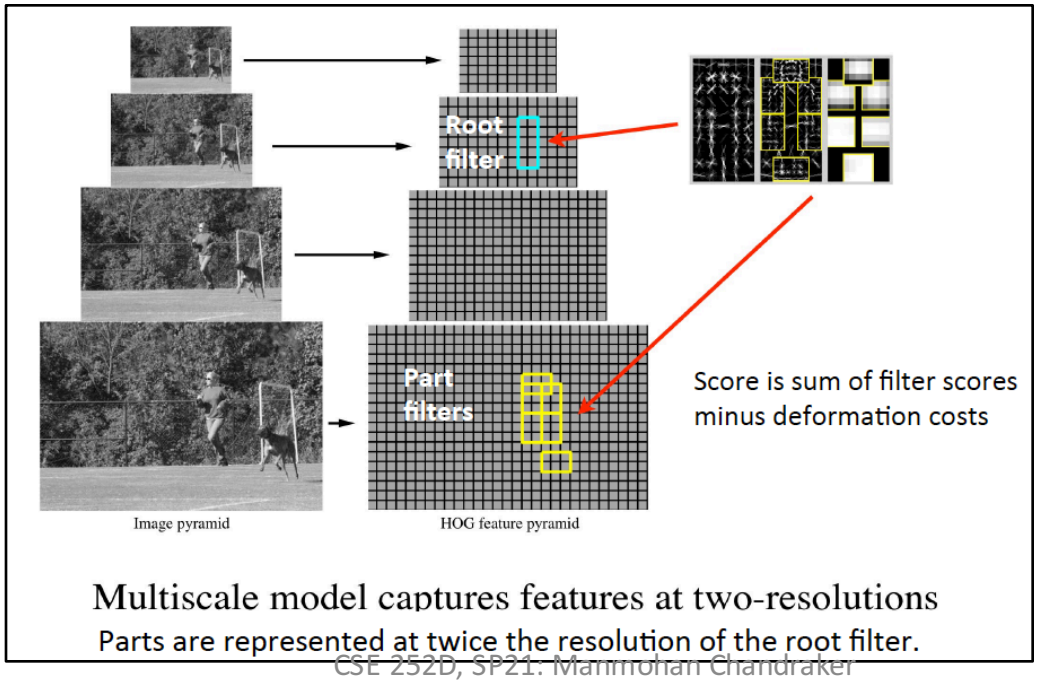
HoG + SVM





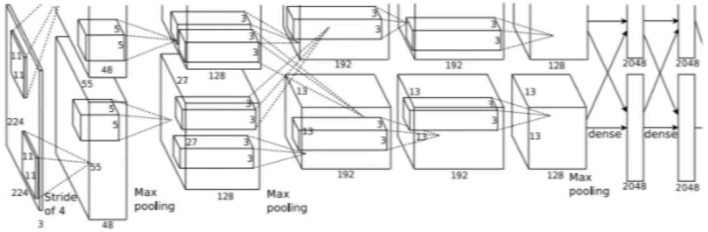
Dalal and Triggs, CVPR 2005

DPM

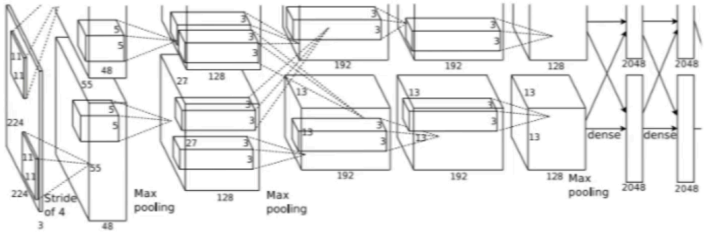


Felzenszwalb, Pedro, et al. "Visual object detection with deformable part models." *Communications of the ACM* 56.9 (2013): 97-105.

Sliding Windows



Dog? YES
Cat? NO
Background? NO

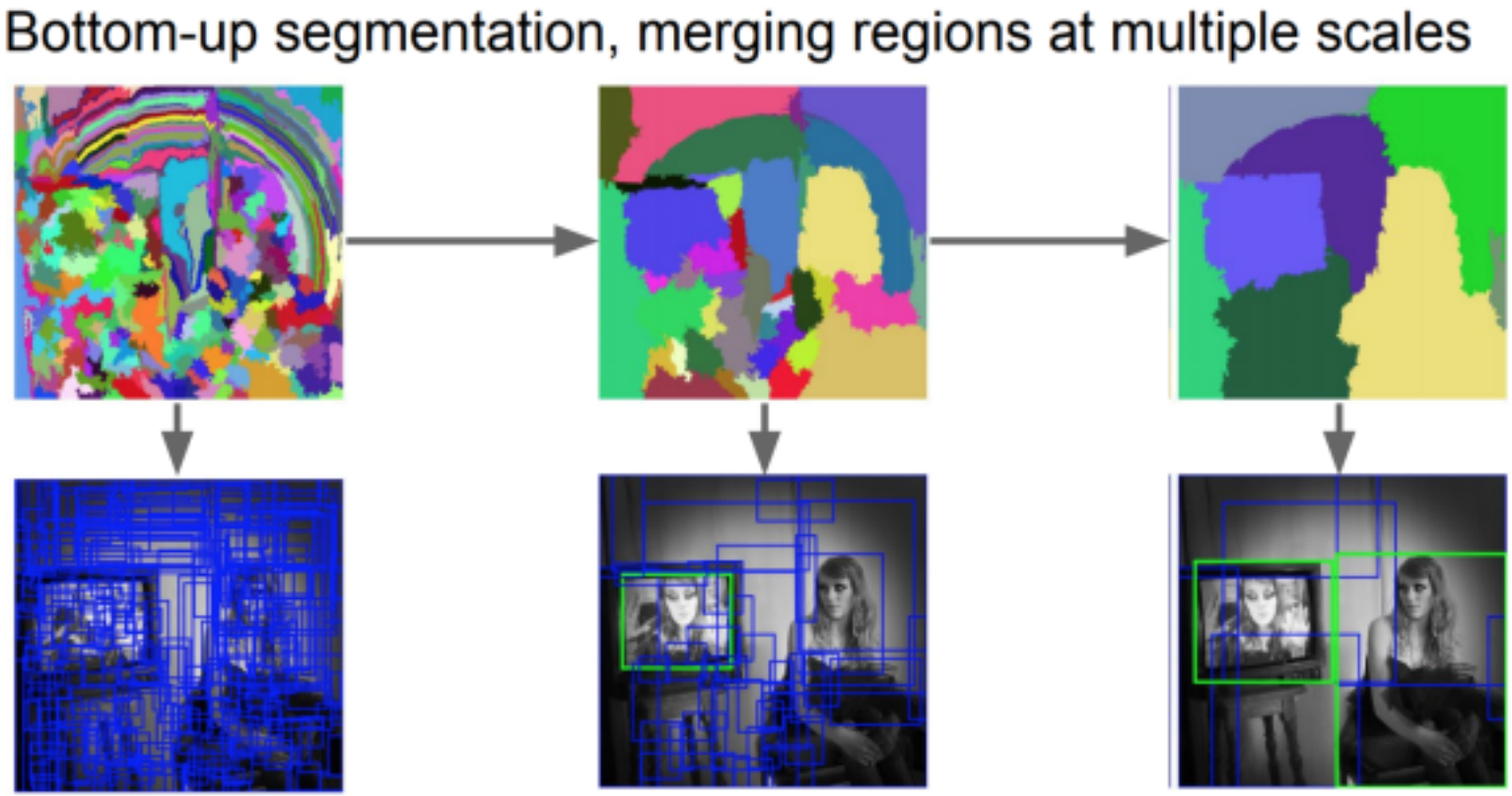


Dog? NO
Cat? NO
Background? YES

- Need to consider windows at several different positions and scales
- Either use a simple feature extractor, or evaluate only few candidates

Region Proposals

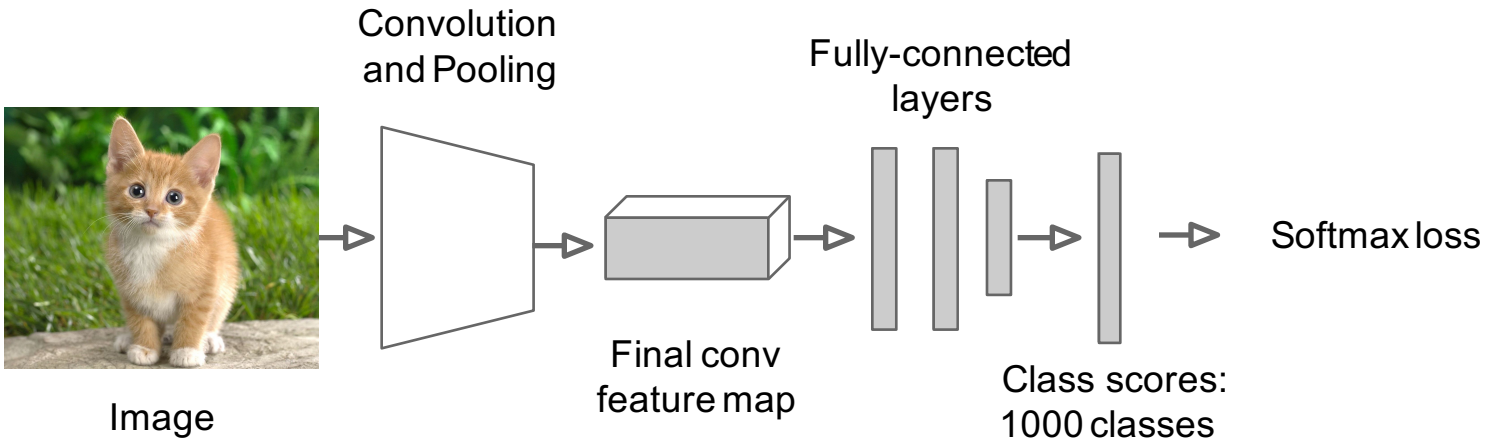
- Find blob-like regions of image that might be objects
- Do not consider object type and tolerate high rate of false positives



[Uijlings et al., Selective search for object recognition]

R-CNN Training

Step 1: Train (or download) a classification model for ImageNet (AlexNet)

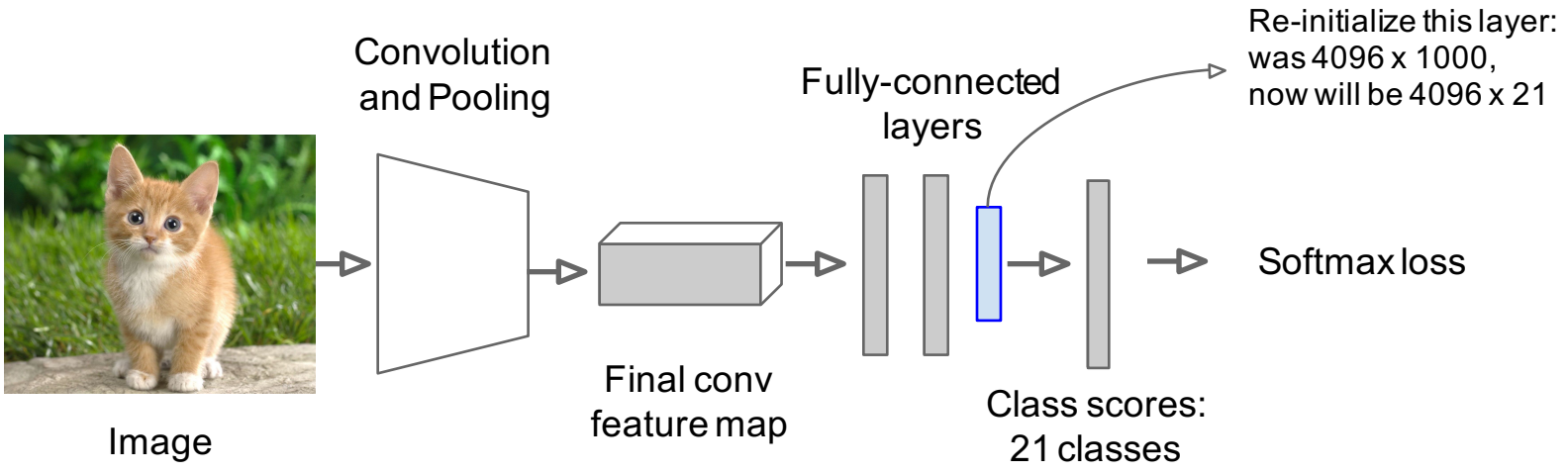


[Girshick et al., Rich feature hierarchies]

R-CNN Training

Step 2: Fine-tune model for detection

- Instead of 1000 ImageNet classes, want 20 object classes + background
- Throw away final fully-connected layer, reinitialize from scratch
- Keep training model using positive and negative regions from detection images



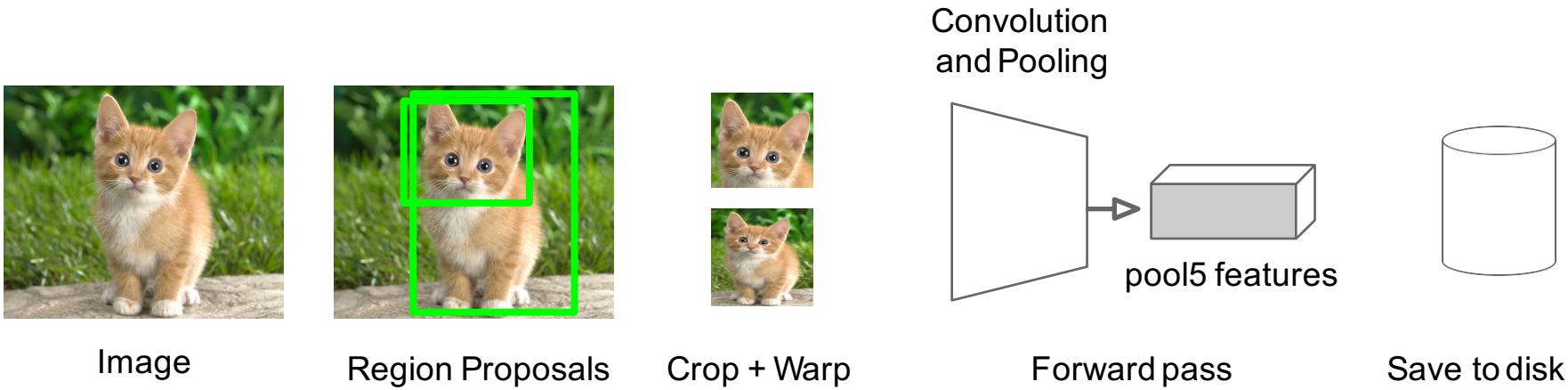
[Slide from: Tamara Berg]

[Girshick et al., Rich feature hierarchies]

R-CNN Training

Step 3: Extract features

- Extract region proposals for all images
- For each region: warp to CNN input size, run forward through CNN, save pool5 features to disk
- Have a big hard drive: features are ~200GB for PASCAL dataset!

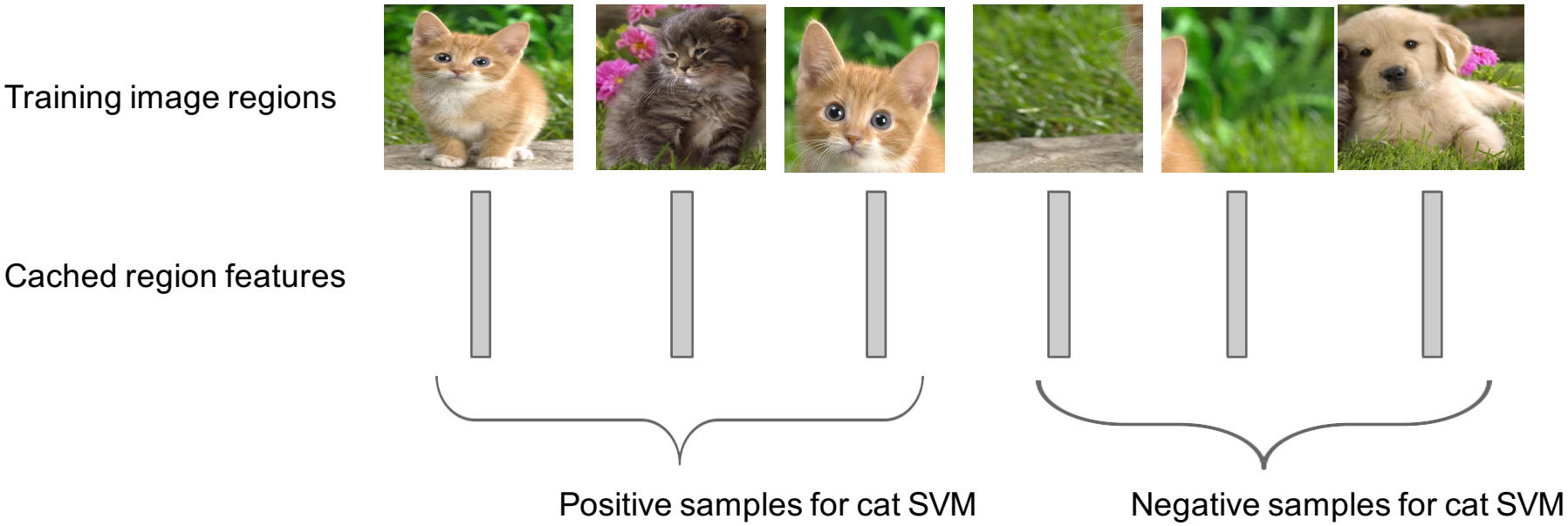


[Slide from: Tamara Berg]

[Girshick et al., Rich feature hierarchies]

R-CNN Training

Step 4: Train one binary SVM per class to classify region features

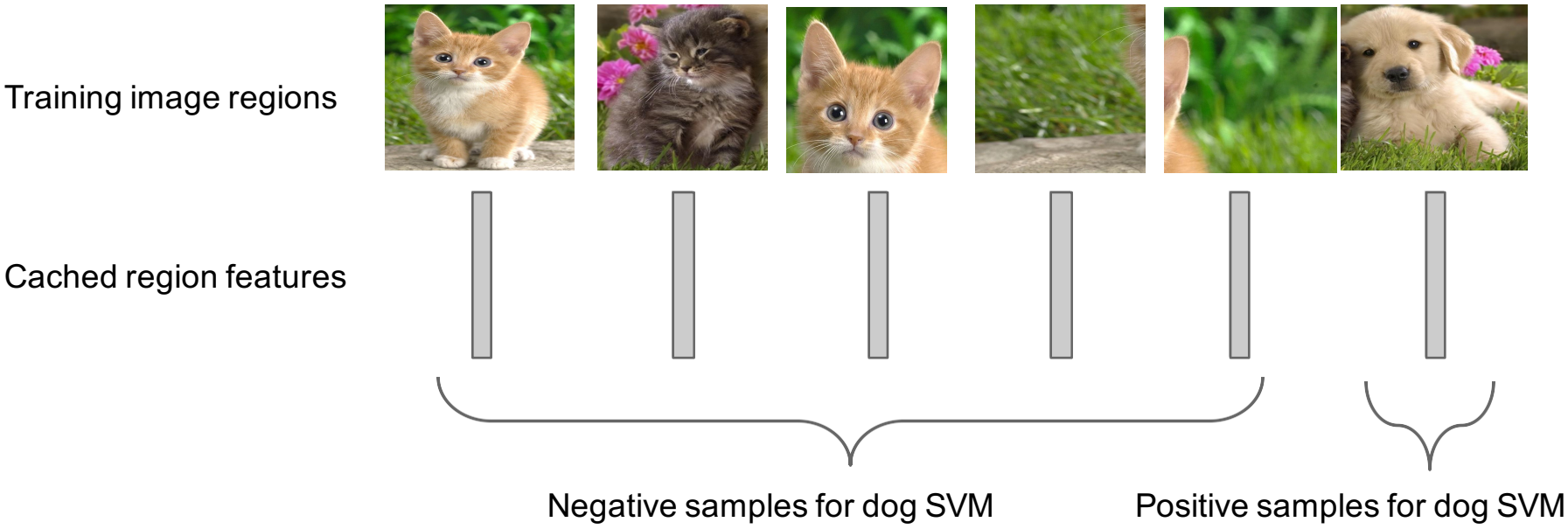


[Slide from: Tamara Berg]

[Girshick et al., Rich feature hierarchies]

R-CNN Training

Step 4: Train one binary SVM per class to classify region features



[Girshick et al., Rich feature hierarchies]

R-CNN Training

Step 5 (bbox regression): For each class, train a linear regression model to map from cached features to offsets to GT boxes to make up for "slightly wrong" proposals

Training image regions

Cached region features

Regression targets
(dx, dy, dw, dh)
Normalized coordinates

(0, 0, 0, 0)
Proposal is good

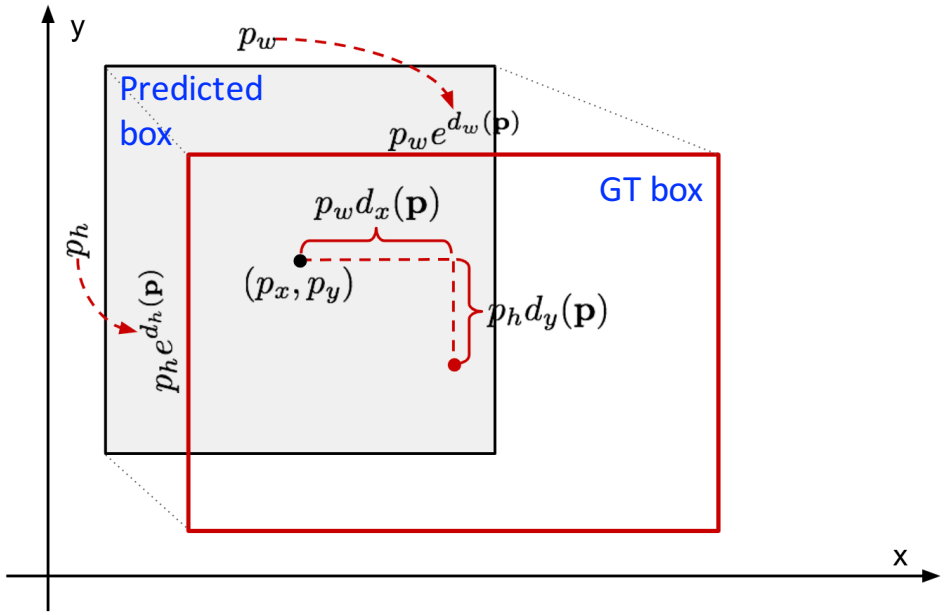
(.25, 0, 0, 0)
Proposal too
far to left

(0, 0, -0.125, 0)
Proposal too
wide

[Girshick et al., Rich feature hierarchies]

R-CNN Regression

Step 5 (bbox regression): For each class, train a linear regression model to map from cached features to offsets to GT boxes to make up for "slightly wrong" proposals



Regression targets:

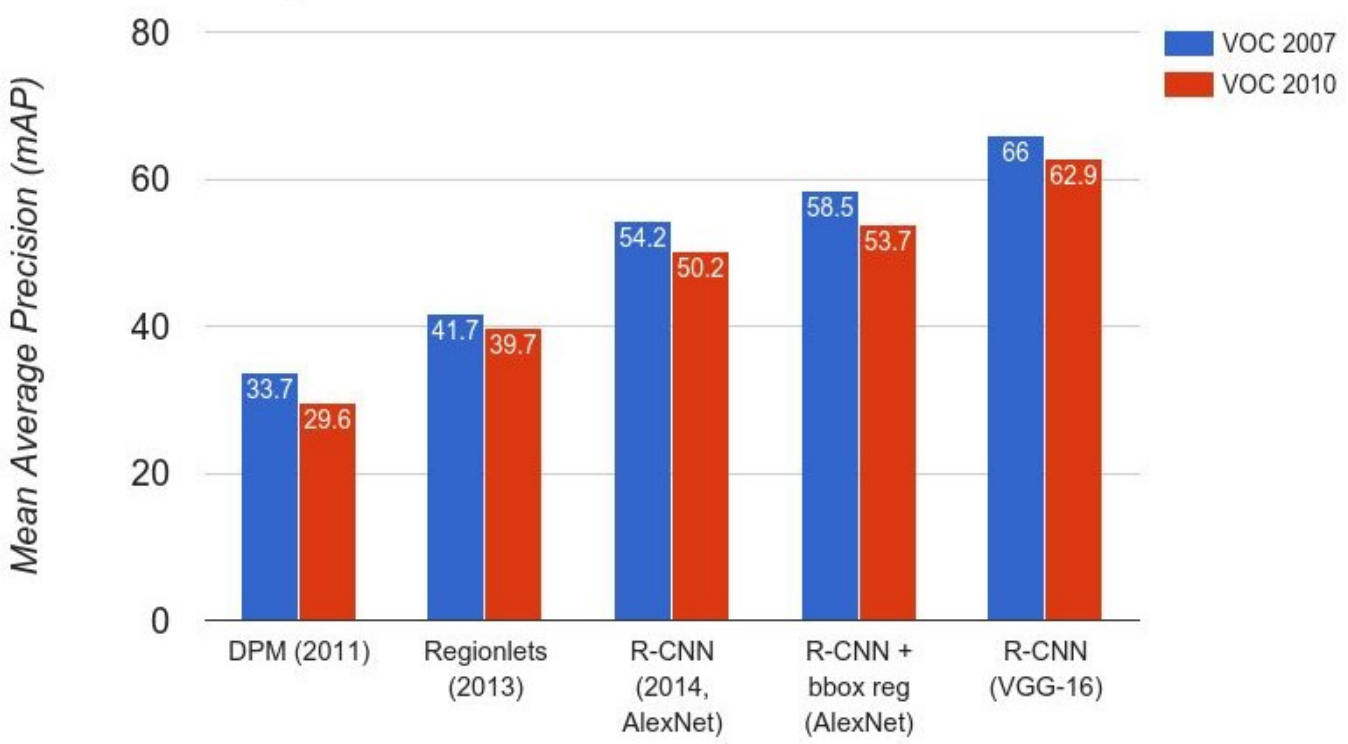
$$t_x = (g_x - p_x)/p_w$$
$$t_y = (g_y - p_y)/p_h$$
$$t_w = \log(g_w/p_w)$$
$$t_h = \log(g_h/p_h)$$

Train a regressor to find offset function d_i that reaches closest to regression target:

$$\mathcal{L}_{\text{reg}} = \sum_{i \in \{x,y,w,h\}} (t_i - d_i(\mathbf{p}))^2 + \lambda \|\mathbf{w}\|^2$$

R-CNN Improvements

- Significant gain over pre-CNN methods
- Bounding box regression helps
- Deeper feature extractor leads to large gain



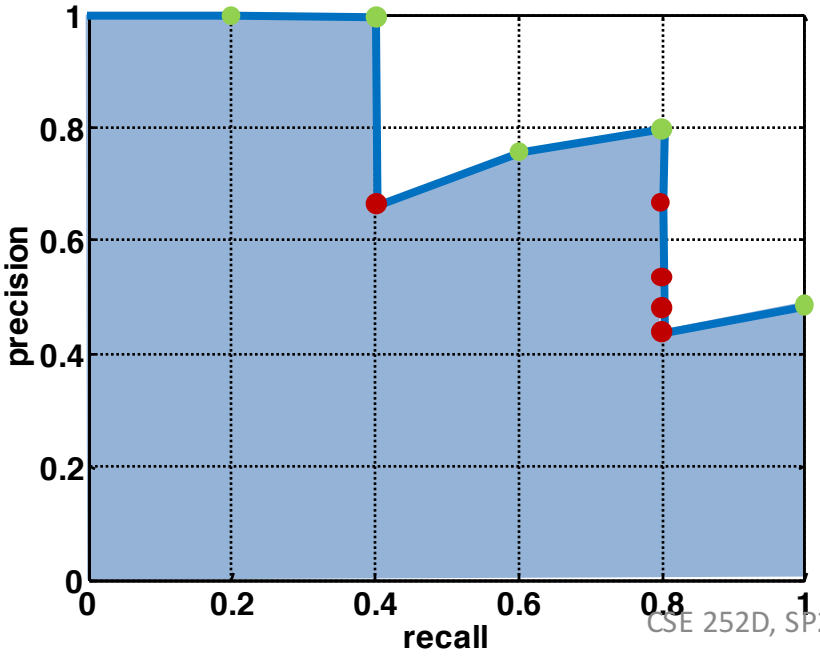
Object Detection Evaluation: Precision and Recall

Query

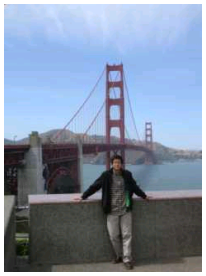
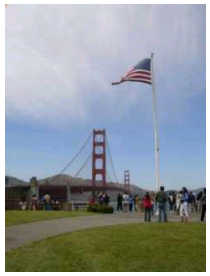
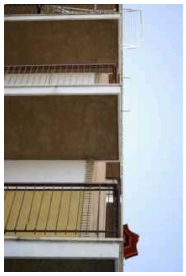
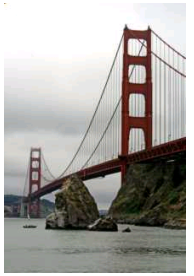
Database size: 10 images
Relevant (total): 5 images

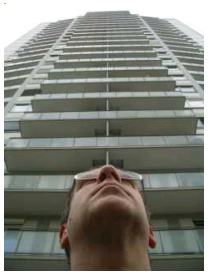
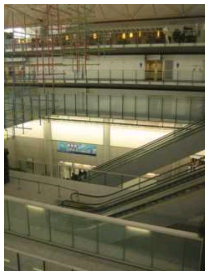
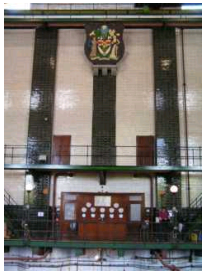
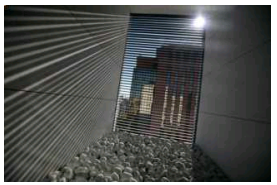
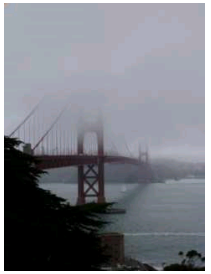
$$\text{precision} = \frac{\text{Number of relevant}}{\text{Number of returned}}$$

$$\text{recall} = \frac{\text{Number of relevant}}{\text{Number of total relevant}}$$



Results (ordered):

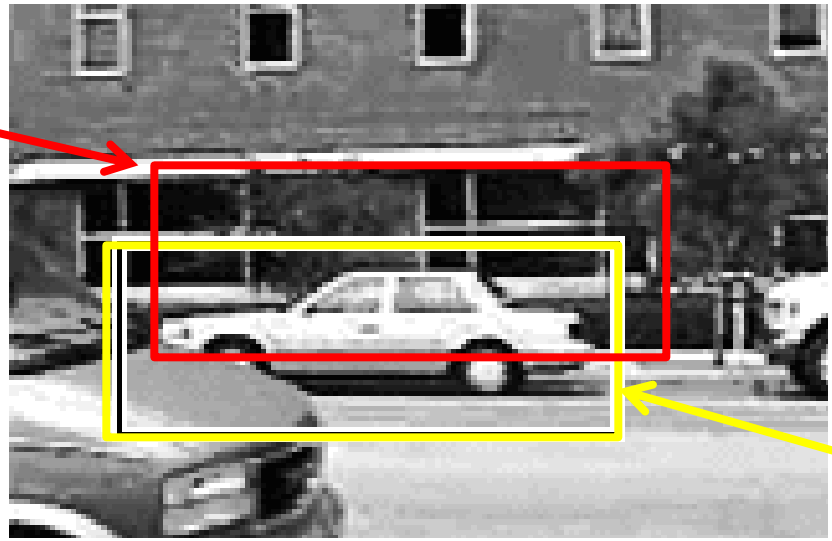






[Ondrej Chum]

Object Detection Evaluation: Scoring a Bounding Box

If prediction and ground truth are *bounding boxes*, when do we have a correct detection?

Object Detection Evaluation: Scoring a Bounding Box



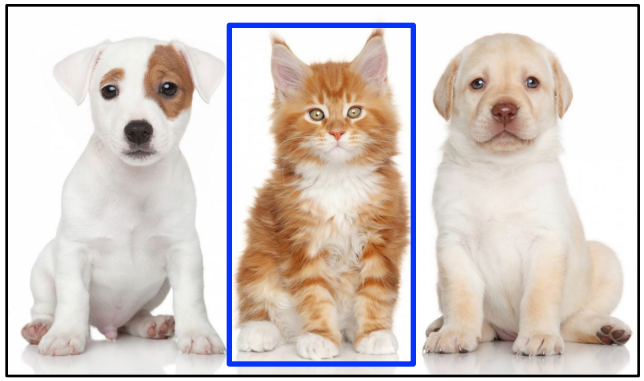
$$a_o = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}$$

$$a_o > 0.5 \qquad correct$$

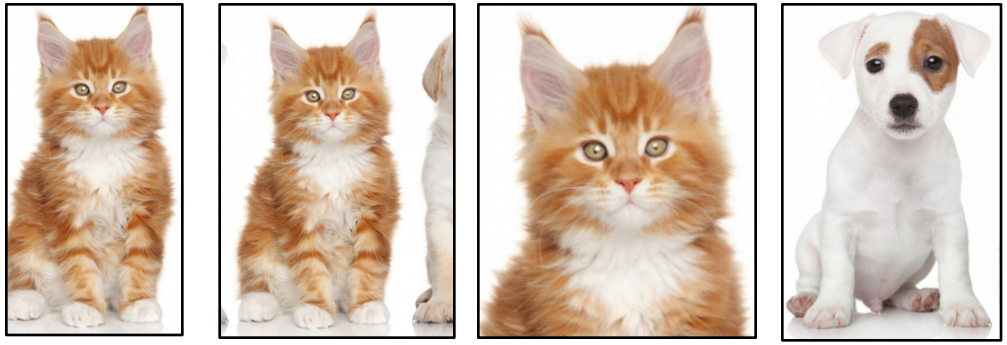
We say the detection is correct (a "true positive") if the intersection of the bounding boxes, divided by their union, is greater than a threshold.

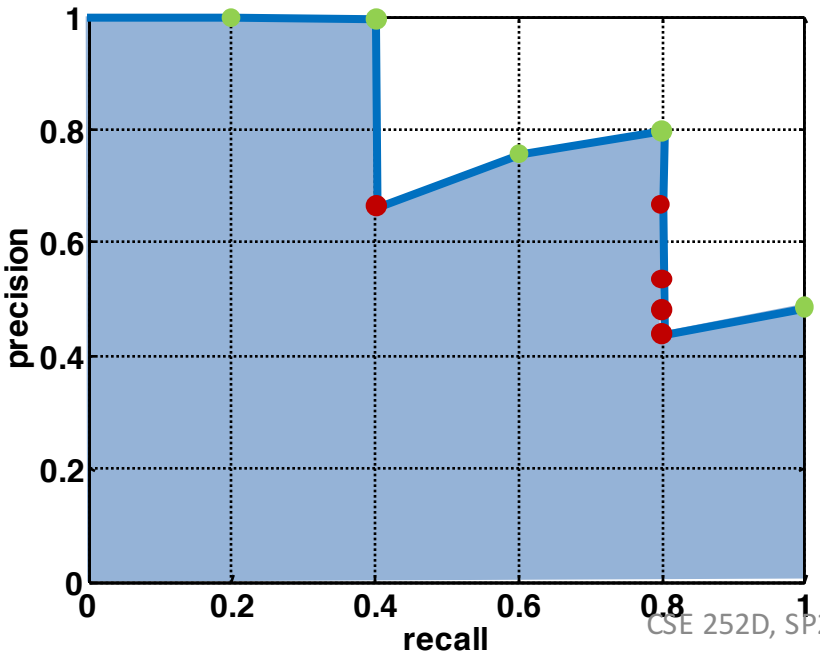
Object Detection Evaluation: Average Precision

Query



Results (ordered):





- Ordering determined by detector confidence (say, softmax score)
- Choose the IoU threshold
- Plot the precision-recall curve
- Average precision: area under the curve
- Mean AP: average AP across all categories

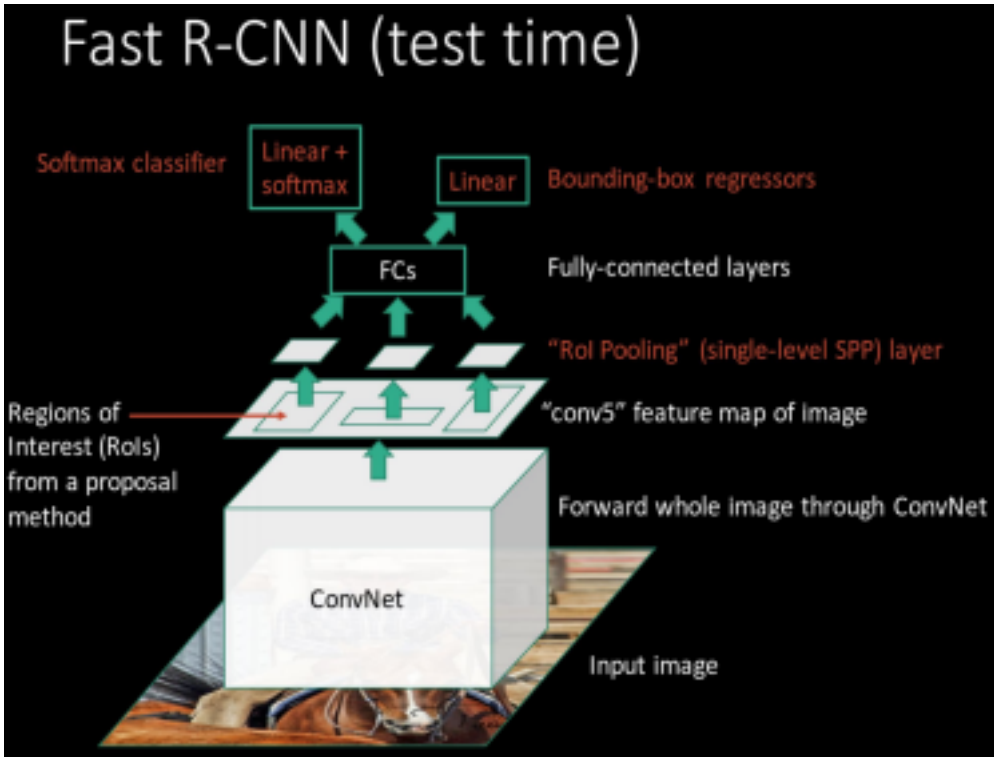
- AP-k: AP value with k% IoU
- Another metric: report the average of AP-k, for various k = [50, 55, …., 95].

Issues with R-CNN

- Very expensive for inference
  - Full forward pass of CNN for each region proposal
- Classification and regression are disjoint from feature extraction
  - CNN features not updated together with detection
- Complex training pipeline
- Need mechanism to “connect” CNN features to classifier and regressor

Fast R-CNN

- Extract convolutional features just once for whole image
- Need method to share computation of convolutional layers between proposals for an image
- Allows several advantages
  - Faster inference
  - Connects CNN to classifier
  - Easier training
- Proposals still from selective search

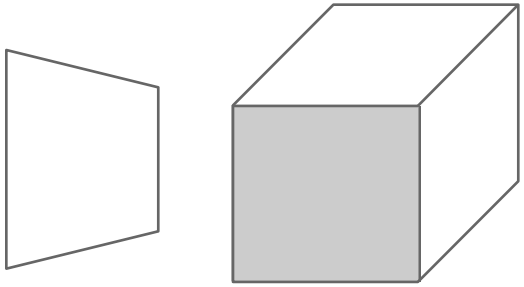


[Girschick, Fast R-CNN]

Fast R-CNN: RoI Pooling

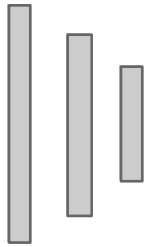
Hi-res input image:
3 x 800 x 600
with region
proposal

Convolution
and Pooling



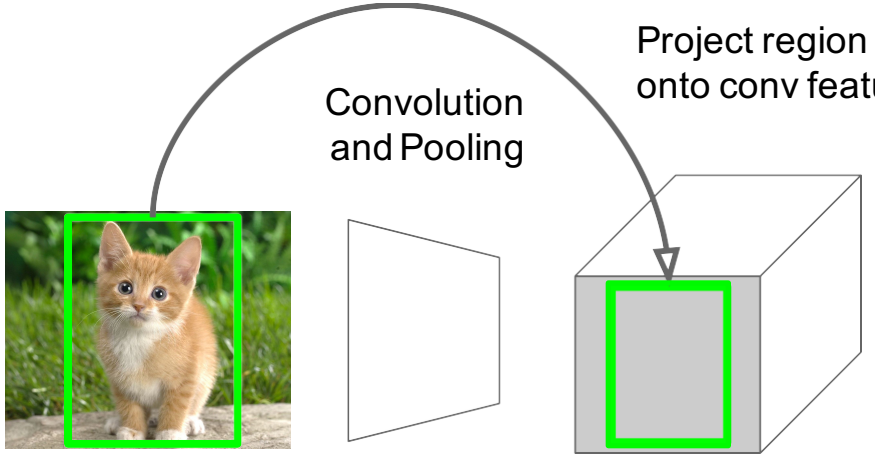
Hi-res conv features:
C x H x W
with region proposal

Fully-connected
layers



Problem: Fully-connected
layers expect fixed size
low-res features: C x h x w

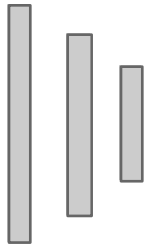
Fast R-CNN: RoI Pooling



Hi-res input image:
3 x 800 x 600
with region
proposal

Hi-res conv features:
C x H x W
with region proposal

Fully-connected
layers

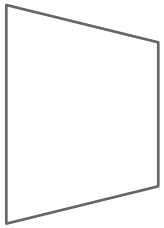


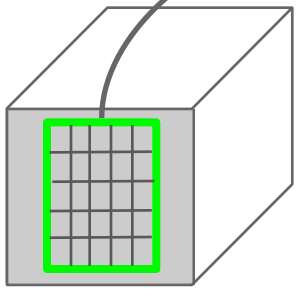
Problem: Fully-connected layers expect fixed size low-res features: C x h x w

Fast R-CNN: RoI Pooling

Hi-res input image:
3 x 800 x 600
with region
proposal

Convolution
and Pooling

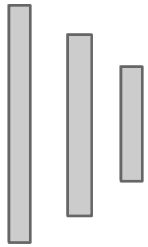




Hi-res conv features:
C x H x W
with region proposal

Divide projected
region into h x w grid

Fully-connected
layers

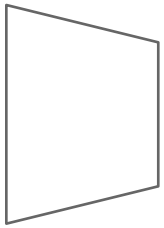


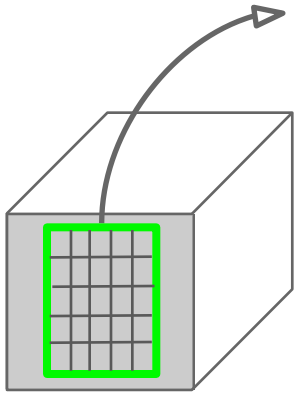
Problem: Fully-connected
layers expect fixed size
low-res features: C x h x w

Fast R-CNN: RoI Pooling

Hi-res input image:
3 x 800 x 600
with region
proposal

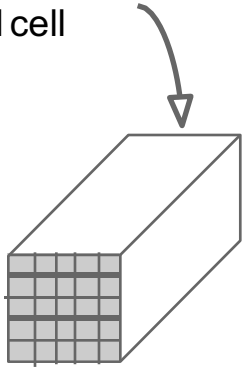
Convolution
and Pooling





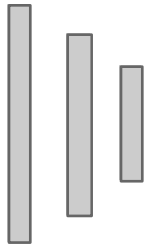
Hi-res conv features:
C x H x W
with region proposal

Max-pool within
each grid cell



RoI conv features:
C x h x w
for region proposal

Fully-connected
layers



Problem: Fully-connected
layers expect fixed size
low-res features: C x h x w

RoI Pooling

$$y_{rj} = x_{i^*(r,j)}$$ (RoI *r*, output *j*)

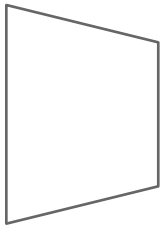
$$i^*(r,j) = \text{argmax}_{i' \in \mathcal{R}(r,j)} x_{i'}$$

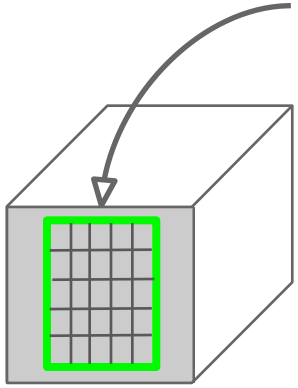
Pooling region

Fast R-CNN: RoI Pooling

Hi-res input image:
3 x 800 x 600
with region
proposal

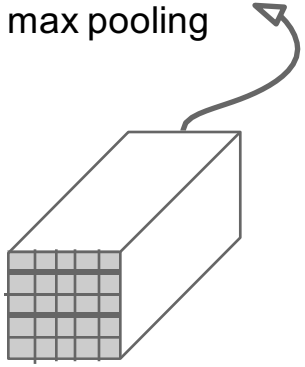
Convolution
and Pooling





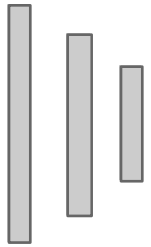
Hi-res conv features:
C x H x W
with region proposal

Can back propagate
similar to max pooling



RoI conv features:
C x h x w
for region proposal

Fully-connected
layers



Problem: Fully-connected
layers expect fixed size
low-res features: C x h x w

RoI Pooling

$$y_{rj} = x_{i^*(r,j)} \quad \text{(RoI } r\text{, output } j\text{)}$$

$$i^*(r,j) = \text{argmax}_{i' \in \mathcal{R}(r,j)} x_{i'}$$

Pooling region

Backpropagation

$$\frac{\partial L}{\partial x_i} = \sum_r \sum_j [i = i^*(r,j)] \frac{\partial L}{\partial y_{rj}}$$

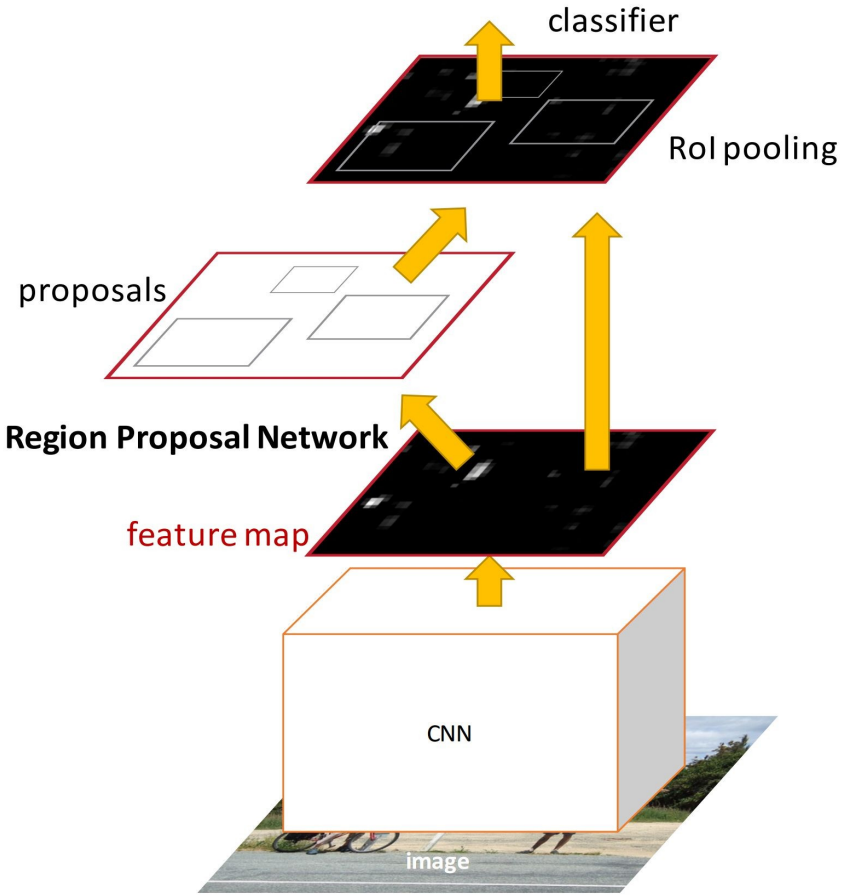
Gradient accumulated if *i* is the
index chosen for max pooling

Fast R-CNN: Improvements over R-CNN

	R-CNN	**Fast R-CNN**
Training Time:	84 hours	**9.5 hours**
Test time per image	47 seconds	**0.32 seconds**
mAP (VOC 2007)	66.0	**66.9**

Using VGG-16 CNN on Pascal VOC 2007 dataset

Faster R-CNN

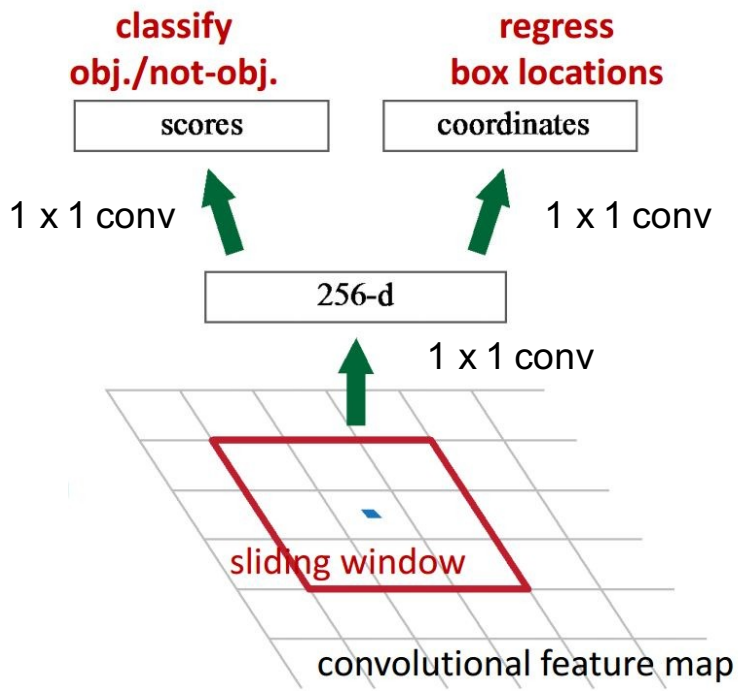


- Proposals still from separate mechanism in Fast R-CNN
- Insert a **Region Proposal Network (RPN)** after last convolutional layer
- RPN trained to produce region proposals directly, no need for external region proposals!
- After RPN, use RoI Pooling and an upstream classifier and regressor just like Fast R-CNN

[Ren et al., Faster R-CNN]

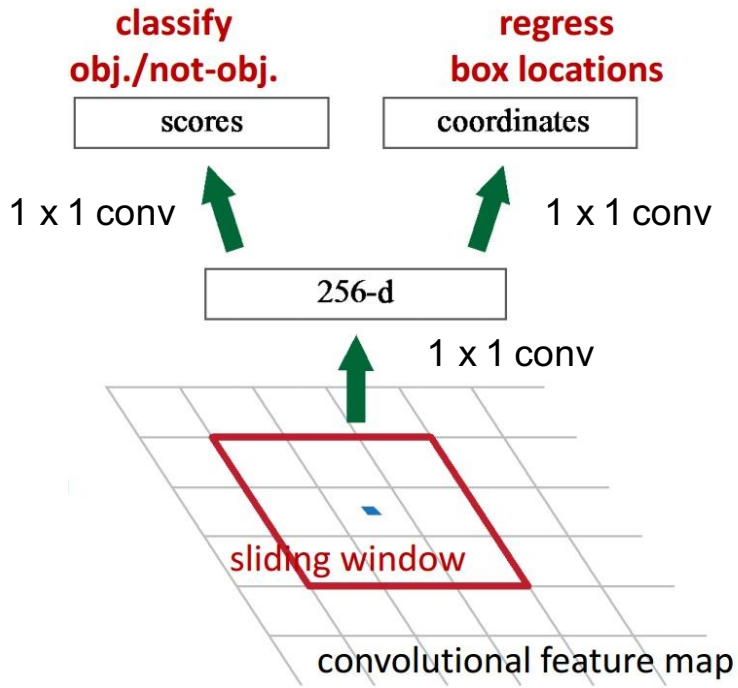
Faster R-CNN: Region Proposal Network

- Slide a small window on the feature map
- Build a small network for:
  - classifying object or not-object
  - regressing bounding box locations
- Position of sliding window gives location information with respect to the image
- Box regression gives finer localization with respect to this sliding window



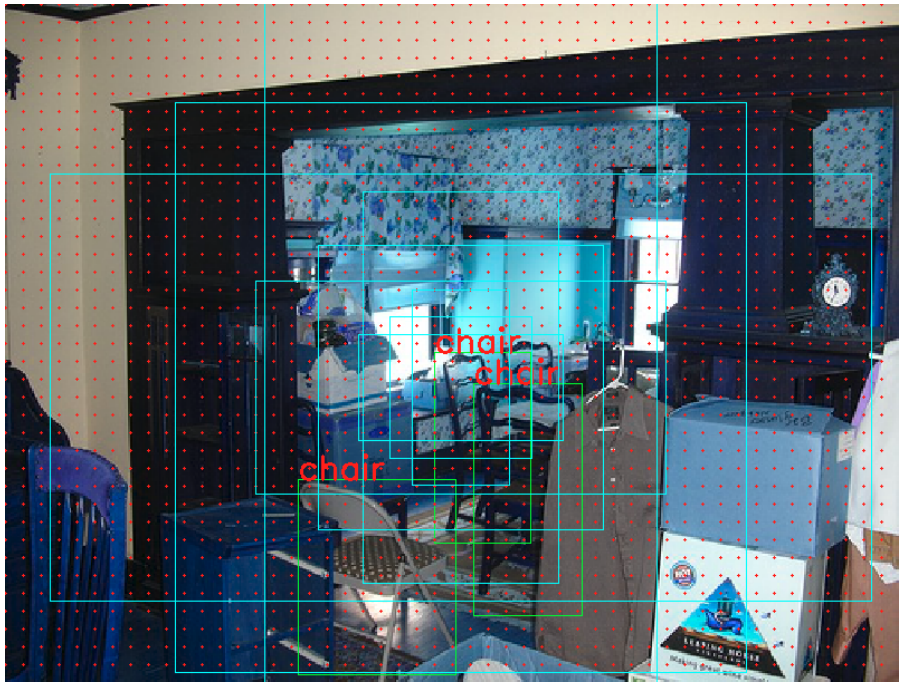
Faster R-CNN: Need for Multiscale Proposals

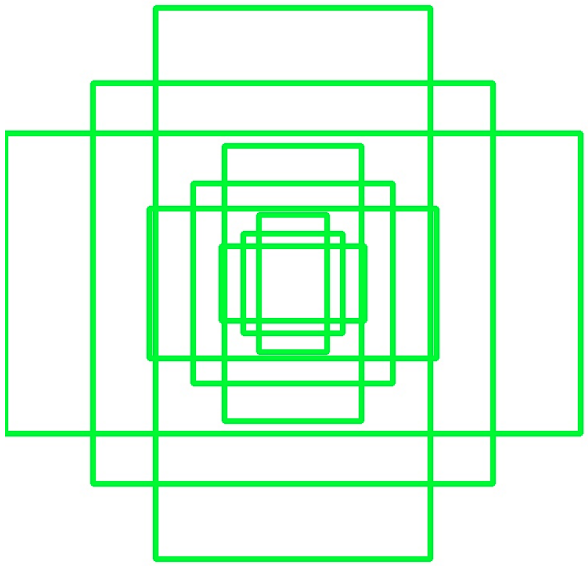
- Problem with region proposal using a convolutional network
  - Multiple scales of objects possible
- Possible solutions
  - Multiple scaled images
  - Filters of different sizes (usually together with above)
- But such approaches are time-consuming.



Faster R-CNN: Anchors

- Consider single object localization
  - Simply have a classification and regression head
- But now consider multiple objects, possibly overlapping
  - Regression for objects will interfere with each other
- Anchors: a set of reference positions on the feature map
  - Anchor box with high ground truth overlap responsible for regressing position
  - Determines reference and spatial extent for predicting object



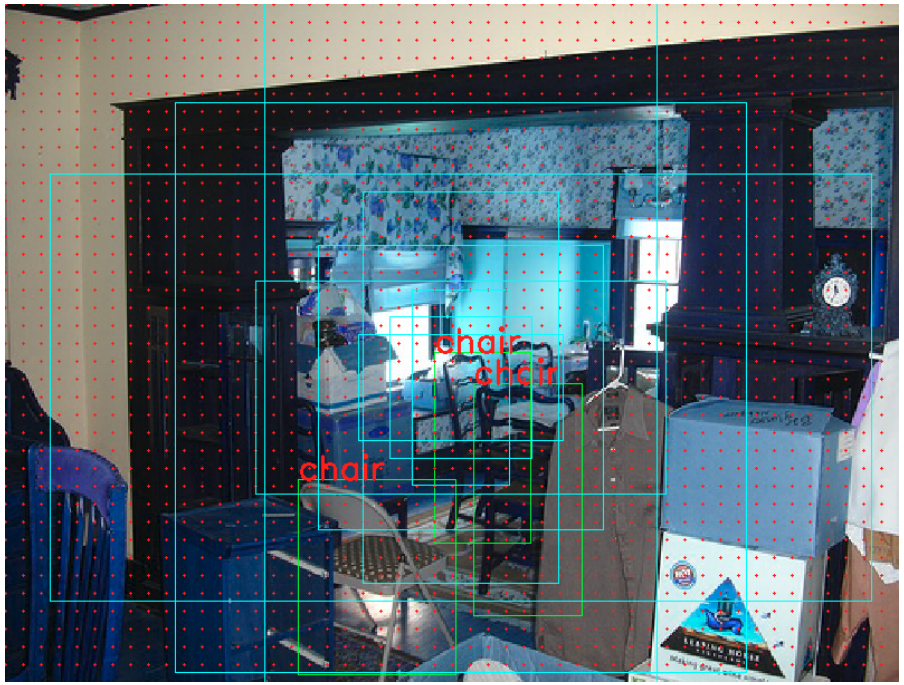


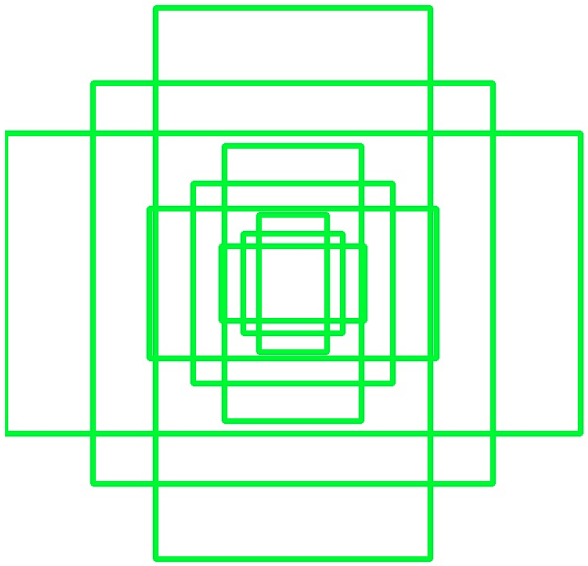
Typically: 3 scales and 3 aspect ratios

Faster R-CNN: Anchors

- A cost-efficient way to achieve multi-scale outputs
  - Relies on image and feature maps at single scale
  - Feature computation is shared across anchor boxes at different scales

- Translation invariance
  - Use same convolutional RPN, anchor boxes for spatial localization
  - A translated object will lead to accordingly shifted proposals





Typically: 3 scales and 3 aspect ratios

Training RPN

- Place anchors uniformly across image, n boxes at every position (typically n = 9)
  - For 40 x 60 feature map, about 21k anchor boxes
- RPN output: 4n regression (x, y, w, h) and 2n classification (object, background)
- Non-maximum suppression to pick about 2k boxes
  - Remove boxes that overlap with others of higher score
- Labels for RPN classification: compute IoU of anchor boxes with ground truth
  - Assign IoU > 0.7 as object and IoU < 0.3 as background
- Bounding box regression
  - Displacement target: distance between centers of ground truth and anchor boxes
  - Size target: log ratio of anchor and ground truth dimensions
- Sample 256 anchors to form mini-batch for training

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*).$$

Cross-entropy

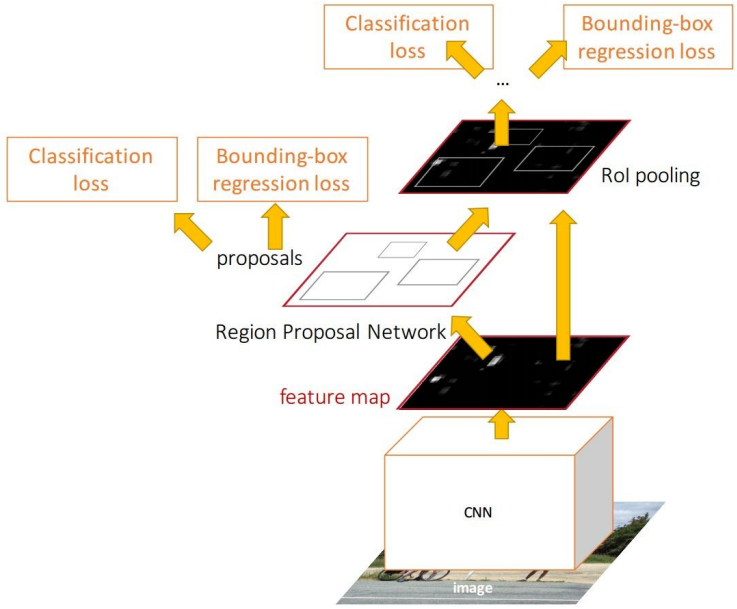
Bounding box label

Smooth L1

Regression target

Training Overall Detector

- Train CNN + RPN
  - Initialize with ImageNet pretrained weights
  - Train end-to-end for region proposal task
- Train CNN + Detector
  - Initialize with ImageNet pretrained weights
  - Use fixed proposals from above RPN
  - Train Fast R-CNN for detection task
- Fine-tune RPN
  - Use CNN from above step
  - Fine-tune RPN for proposal task
- Fine-tune detector
  - Keep CNN fixed
  - Fine-tune Fast R-CNN layers for detection task

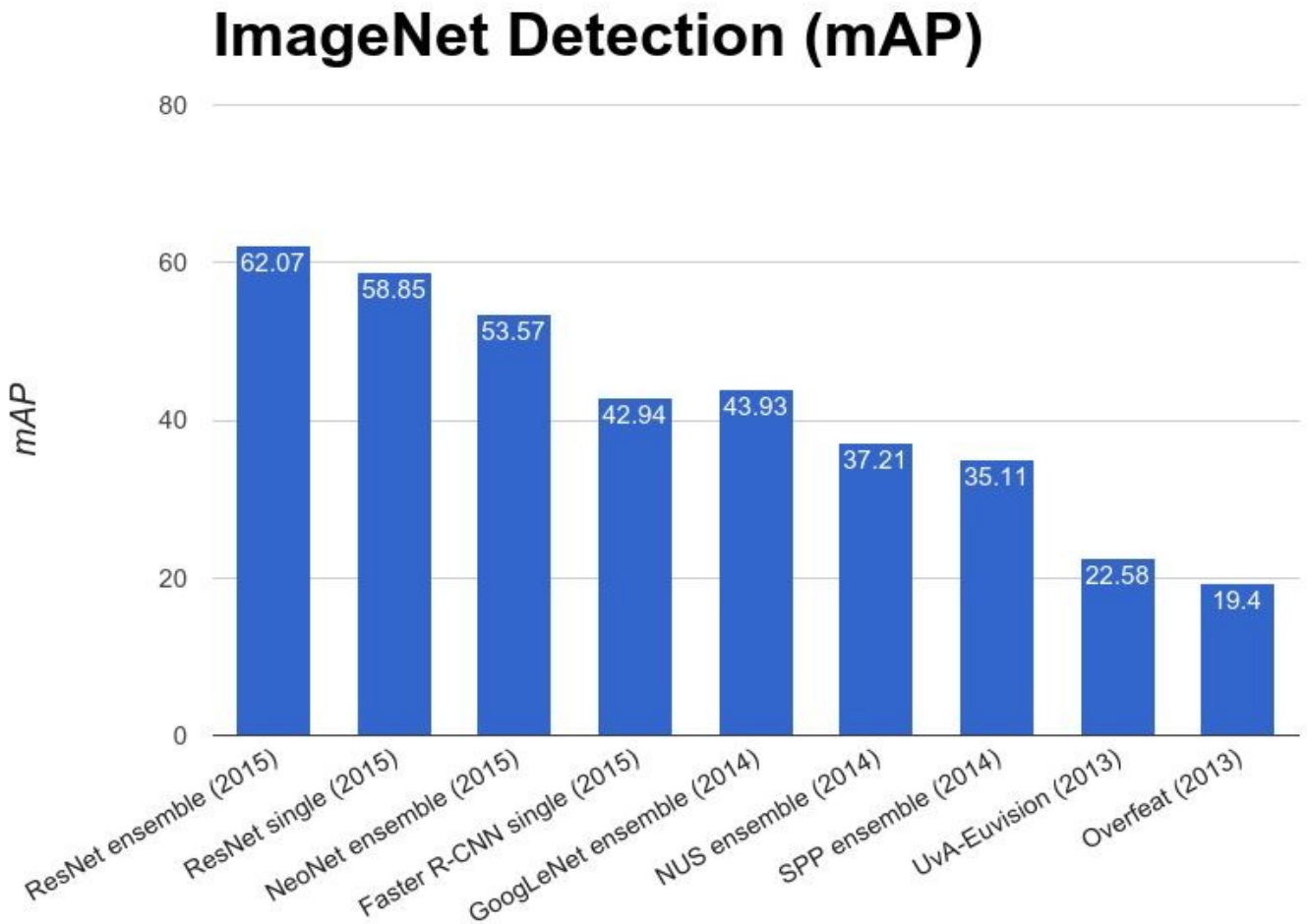


- Subsequently, a joint training is also available with all four losses
  - RPN: classification (object or background), regression (anchor to proposal)
  - Fast R-CNN: classification (object category), regression (proposal to bounding box)

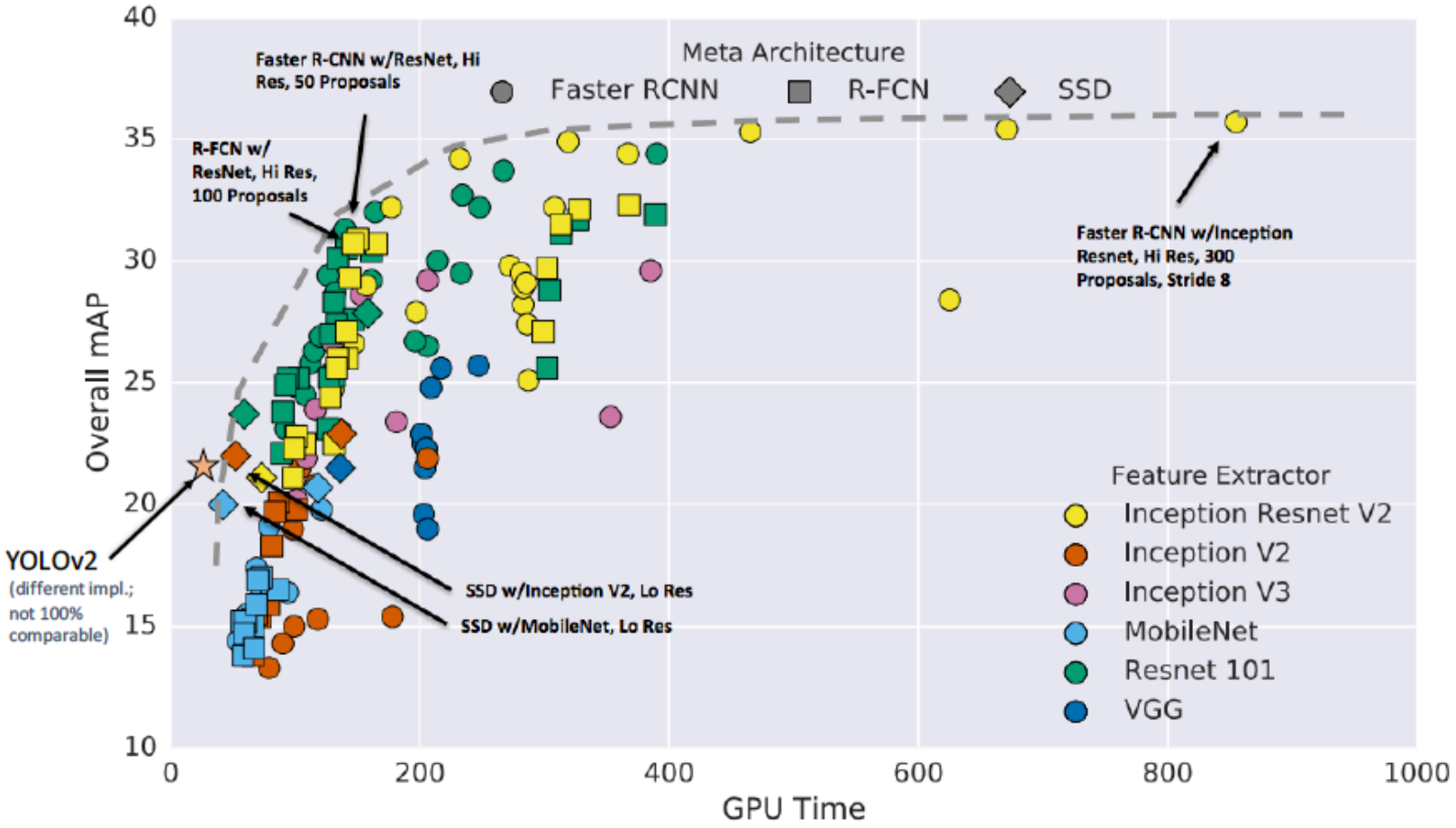
Faster R-CNN Improvements

	R-CNN	**Fast R-CNN**	**Faster R-CNN**
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

Faster R-CNN Improvements



Object Detection



Huang et al. Speed/Accuracy Tradeoffs for Modern Convolutional Object Detectors. CVPR 2017