Object detection and localization

Classification + Localization: Task

Classification: C classes

Input: Image

Output: Class label

Evaluation metric: Accuracy

CAT

Localization:

Input: Image

Output: Box in the image (x, y, w, h)

Evaluation metric: Intersection over Union

(x, y, w, h)

Classification + Localization: Do both

Idea #1: Localization as Regression

Input: image

Only one object, simpler than detection

Neural Net

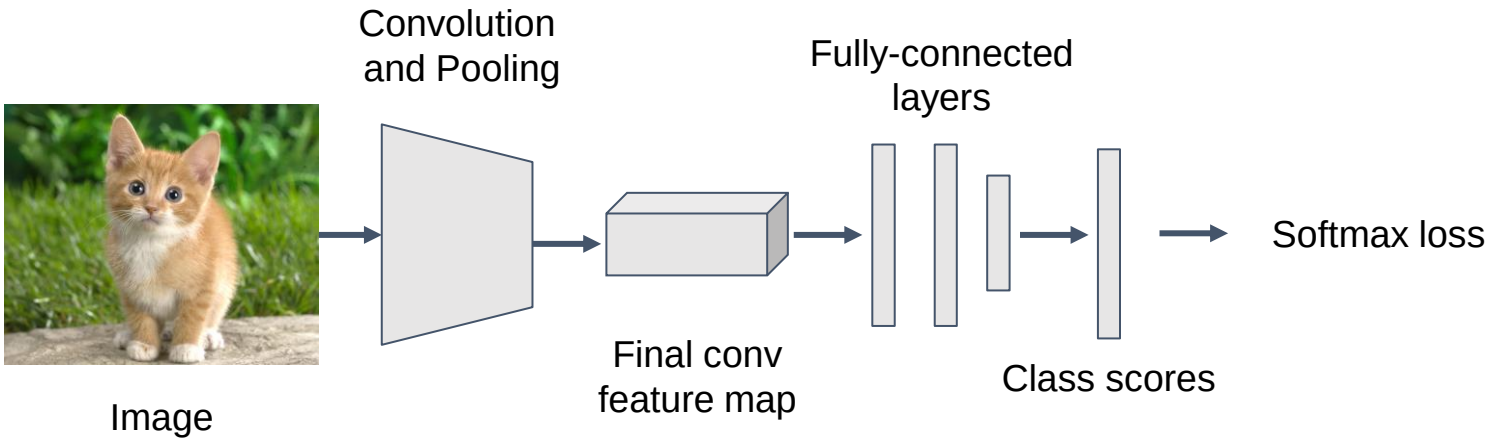
Output:
Box coordinates
(4 numbers)

Correct output:
box coordinates
(4 numbers)

Loss:
L2 distance

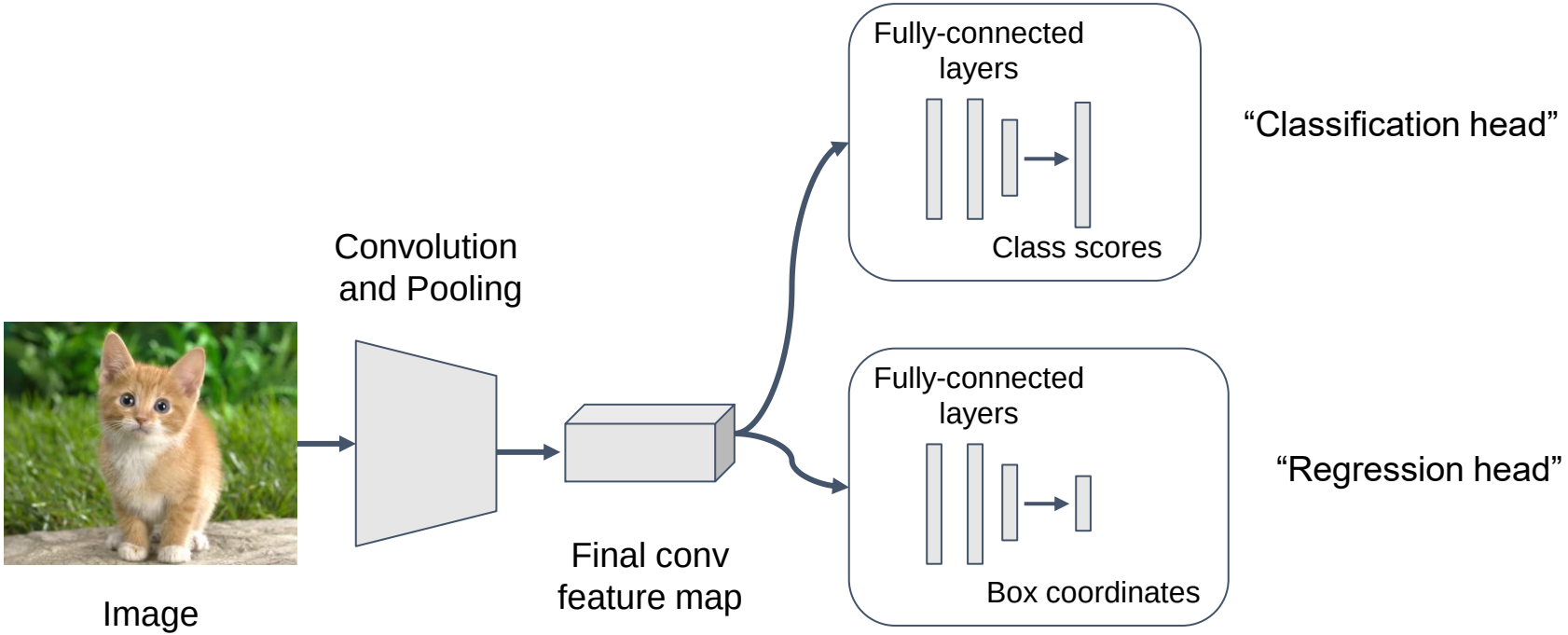
Simple Recipe for Classification + Localization

Step 1: Train (or download) a classification model (AlexNet, VGG, GoogLeNet)



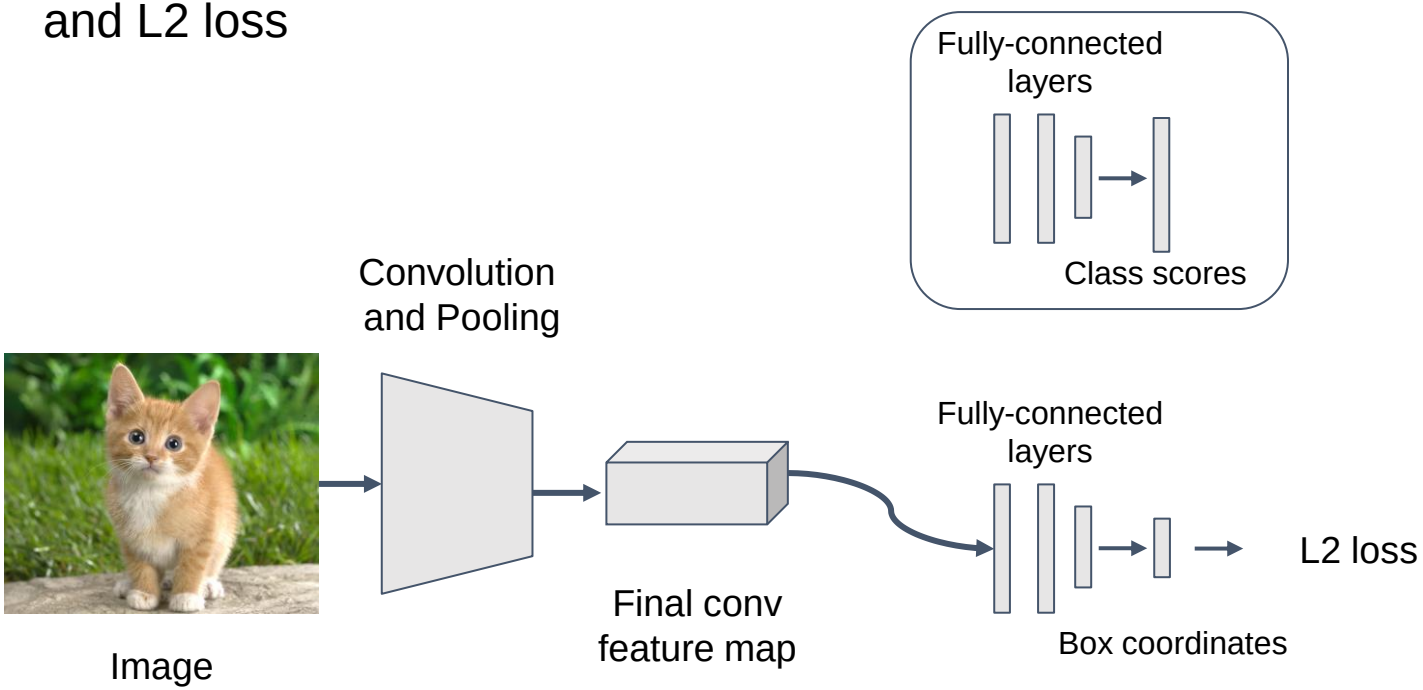
Simple Recipe for Classification + Localization

Step 2: Attach a new fully-connected “regression head” to the network



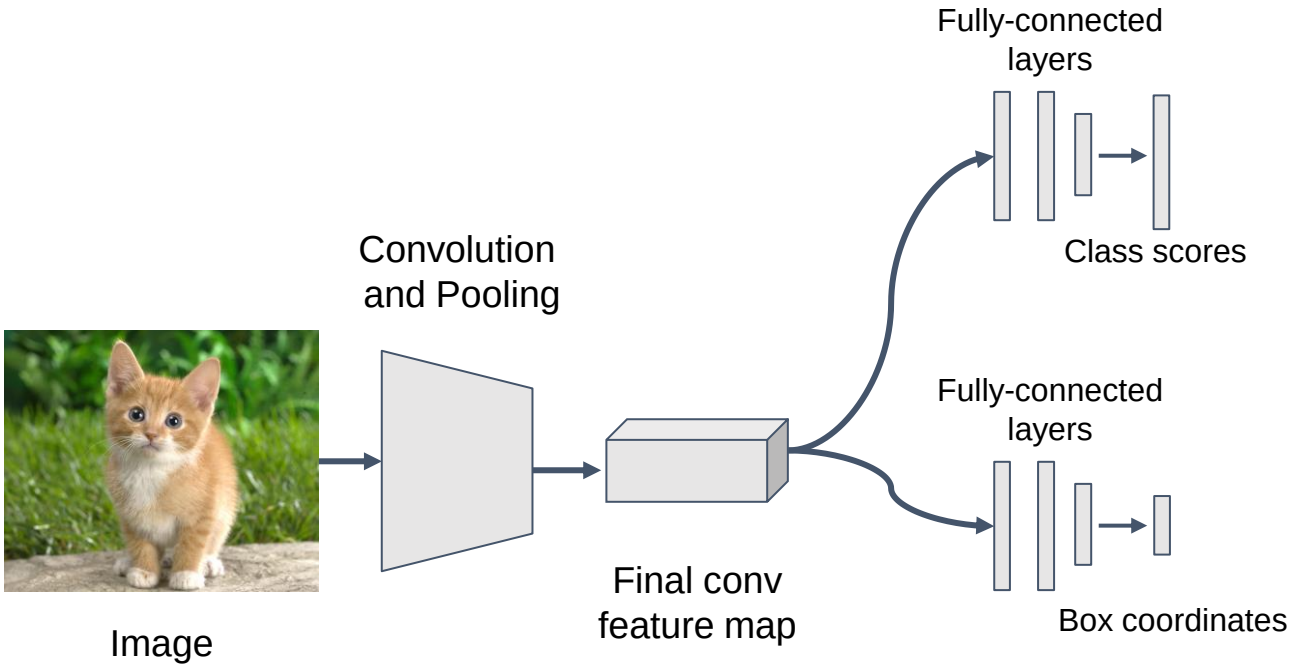
Simple Recipe for Classification + Localization

Step 3: Train the regression head only with stochastic gradient descent (SGD) and L2 loss



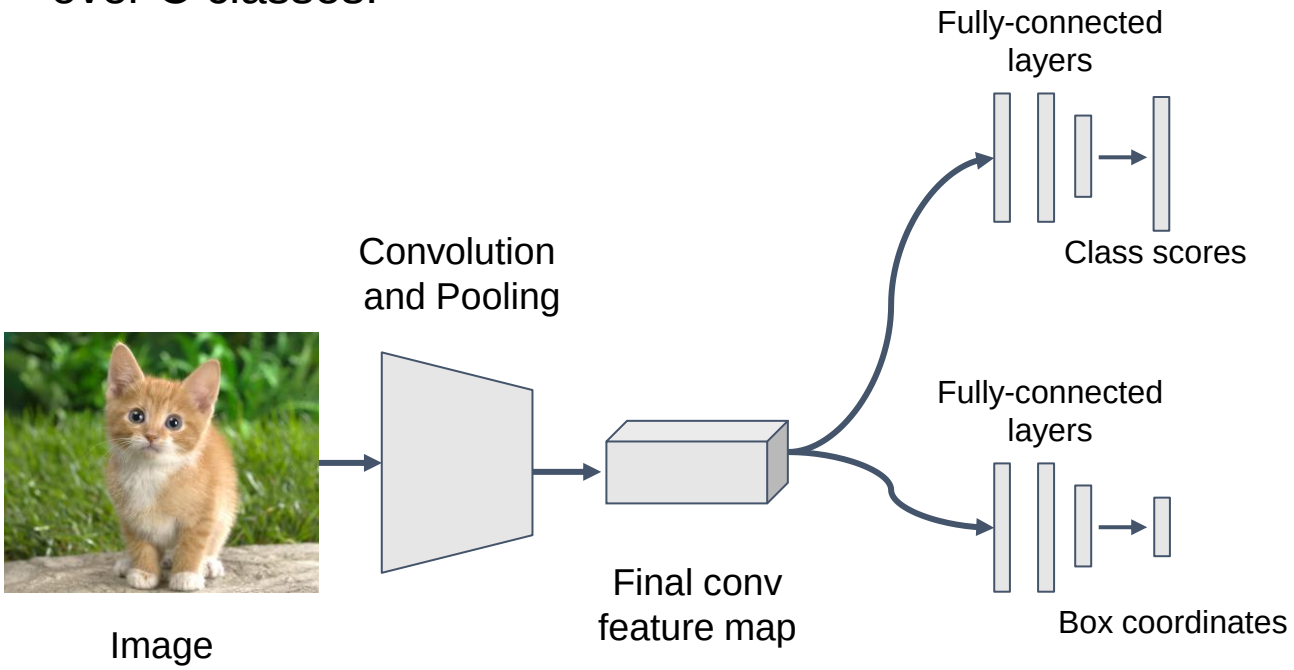
Simple Recipe for Classification + Localization

Step 4: At test time use both heads



Per-class vs class agnostic regression

Assume classification over C classes:

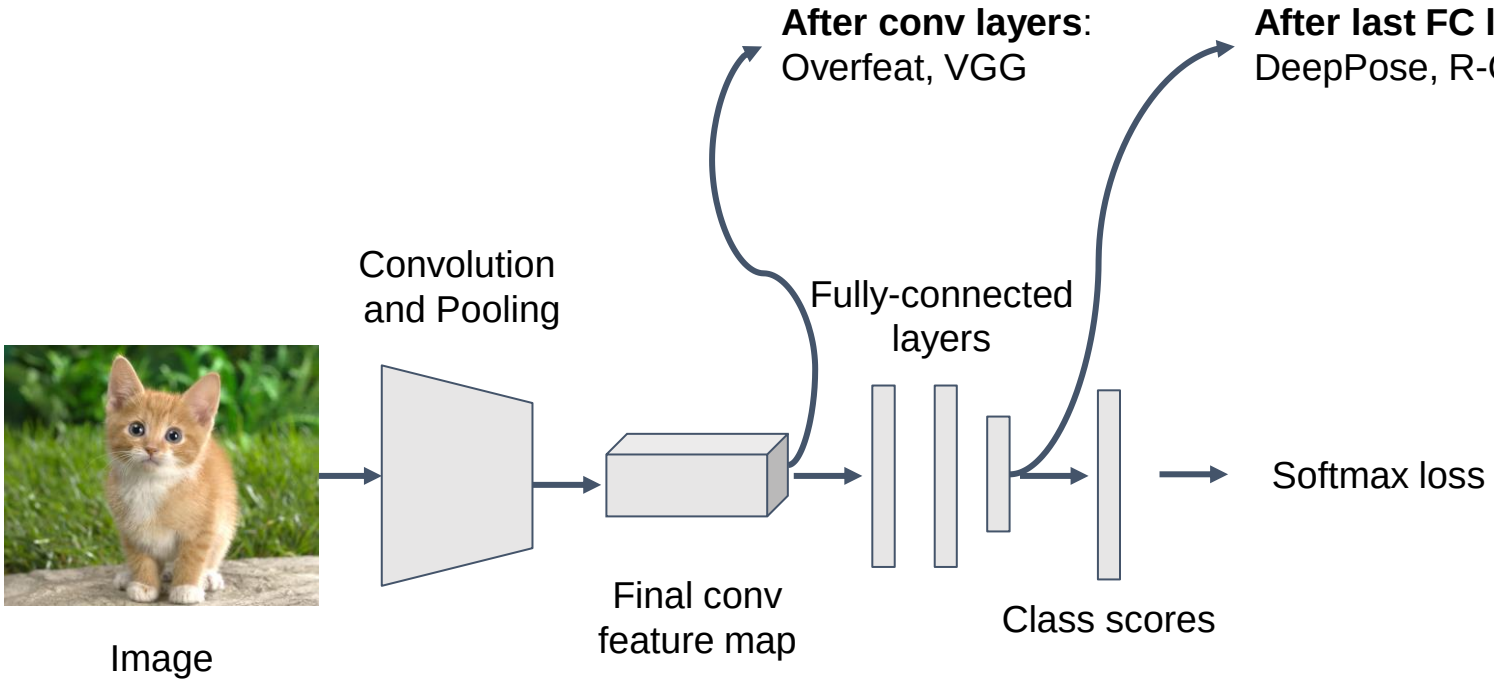


Classification head:
C numbers
(one per class)

Class agnostic:
4 numbers
(one box)

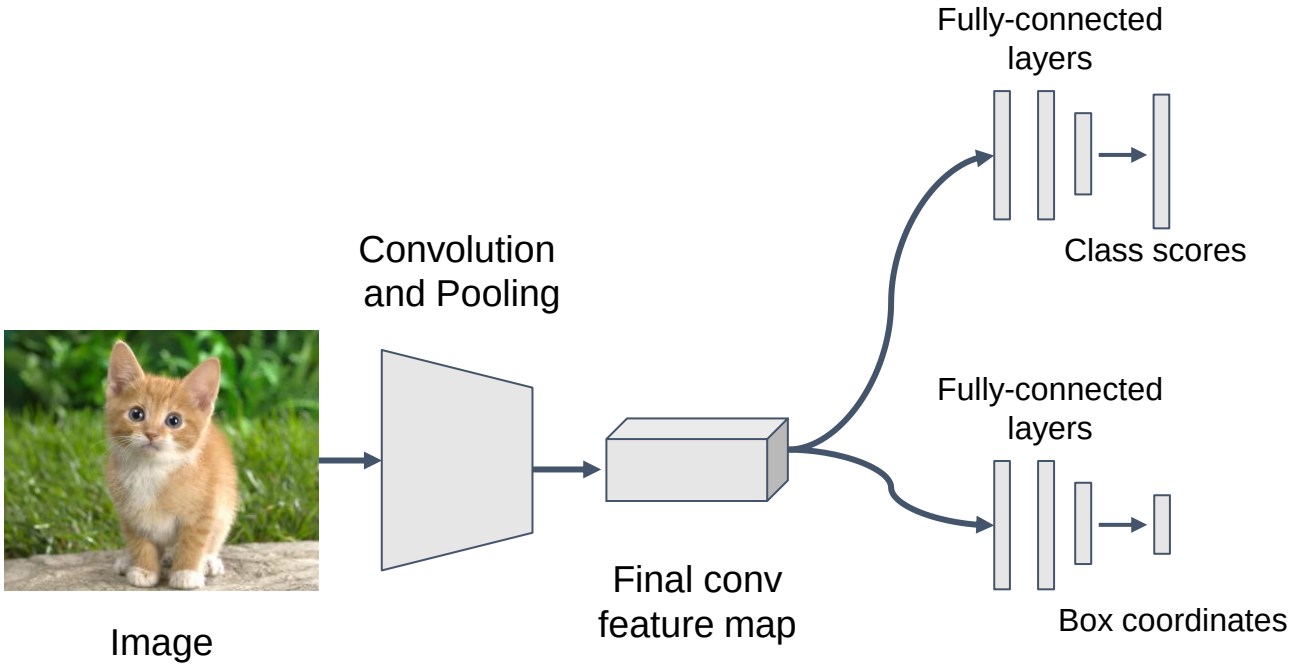
Class specific:
C x 4 numbers
(one box per class)

Where to attach the regression head?



Aside: Localizing multiple objects

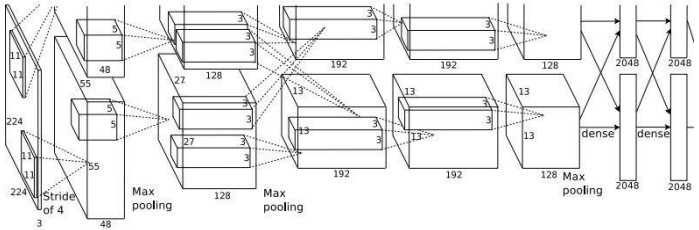
Want to localize **exactly** K objects in each image (e.g. whole cat, cat head, cat's left ear, cat 's ear for K=4)



K x 4 numbers
(one box per object)

Object Detection: Multiple Objects

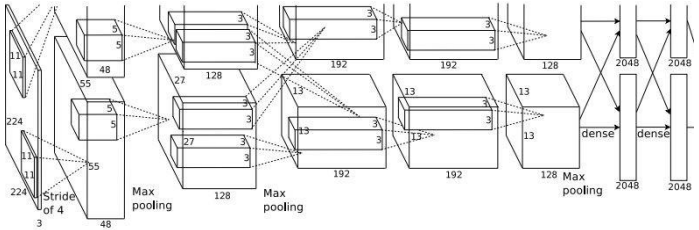
Apply a classifier to many different crops of the image; the classifier classifies each crop as object or background



Dog? NO
Cat? NO
Background? YES

Object Detection: Multiple Objects

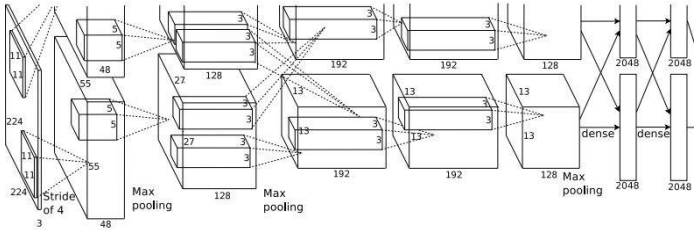
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Object Detection: Multiple Objects

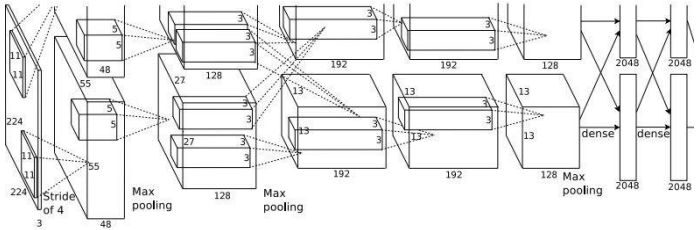
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Object Detection: Multiple Objects

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

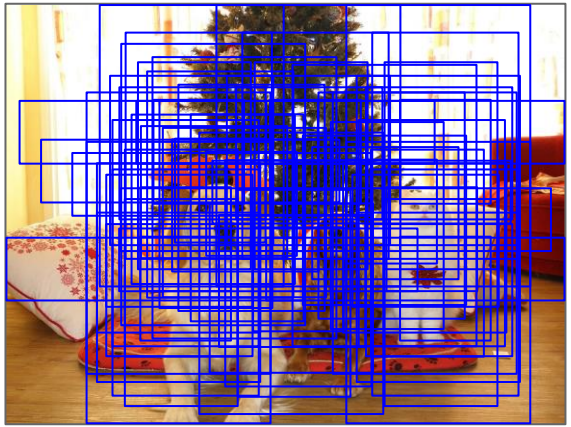


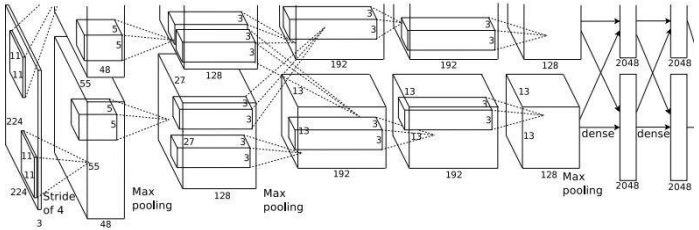
Dog? NO
Cat? YES
Background? NO

Q: What's the problem with this approach?

Object Detection: Multiple Objects

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





Dog? NO
Cat? YES
Background? NO

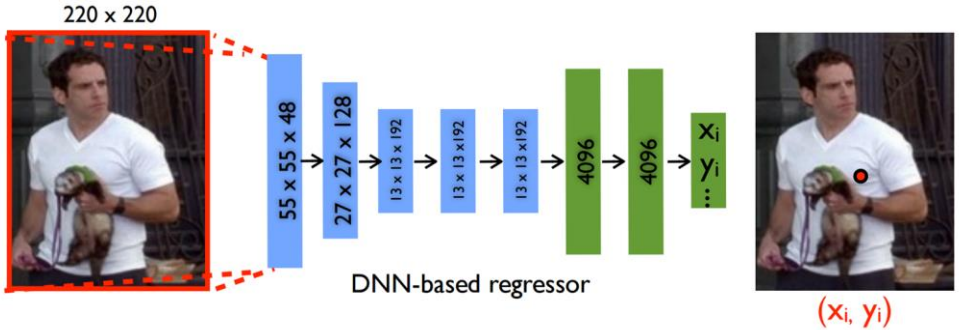
Problem: Need to apply CNN to huge number of locations, scales, and aspect ratios, very computationally expensive!

Aside: Human Pose Estimation

Represent a person by K joints

Regress (x, y) for each joint from last fully-connected layer of AlexNet

(Details: Normalized coordinates, iterative refinement)



Toshev and Szegedy, “DeepPose: Human Pose Estimation via Deep Neural Networks”, CVPR 2014

Idea #2: Sliding Window

- Run classification + regression network at multiple locations on a high-resolution image
- Convert fully-connected layers into convolutional layers for efficient computation
- Combine classifier and regressor predictions across all scales for final prediction

Sliding Window: Overfeat

Winner of ILSVRC 2013
localization challenge

Image:
3 x 221 x 221

Convolution
+ pooling

Feature map:
1024 x 5 x 5

FC

FC

4096

FC

4096

FC

Class scores:
1000

Softmax
loss

FC

4096

FC

1024

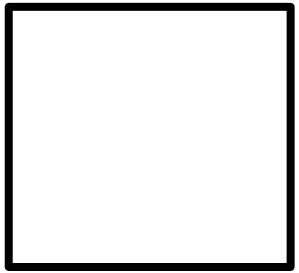
FC

Boxes:
1000 x 4

Euclidean
loss

Sermanet et al, “Integrated Recognition, Localization and Detection using Convolutional Networks”, ICLR 2014

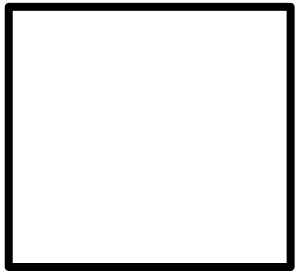
Sliding Window: Overfeat



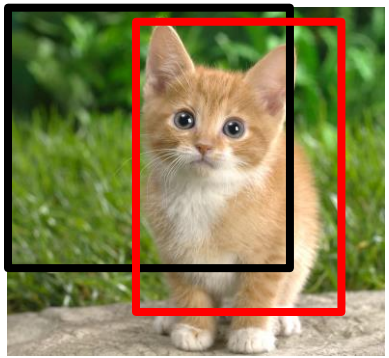
Network input:
3 x 221 x 221

Larger image:
3 x 257 x 257

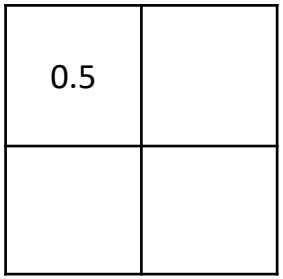
Sliding Window: Overfeat



Network input:
3 x 221 x 221

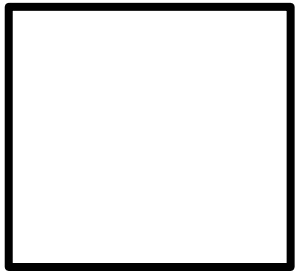


Larger image:
3 x 257 x 257

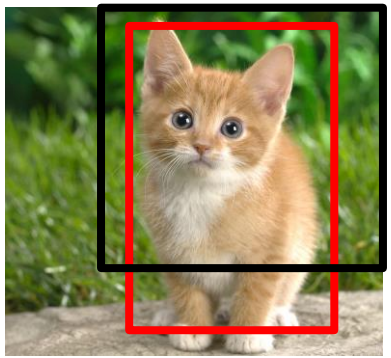


Classification scores:
P(cat)

Sliding Window: Overfeat



Network input:
3 x 221 x 221

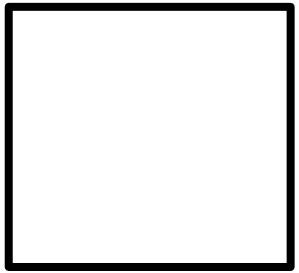


Larger image:
3 x 257 x 257

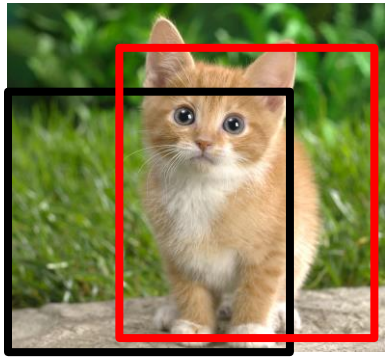
0.5	0.75

Classification scores:
P(cat)

Sliding Window: Overfeat



Network input:
3 x 221 x 221

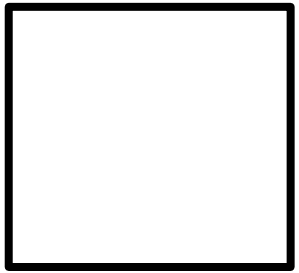


Larger image:
3 x 257 x 257

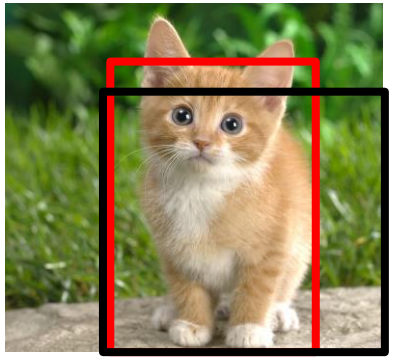
0.5	0.75
0.6	

Classification scores:
P(cat)

Sliding Window: Overfeat



Network input:
3 x 221 x 221

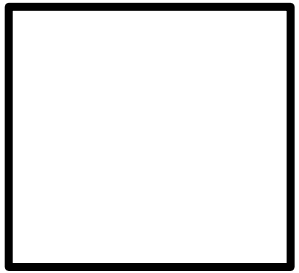


Larger image:
3 x 257 x 257

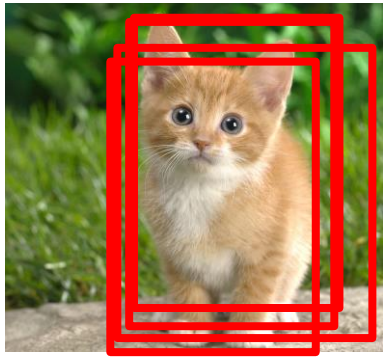
0.5	0.75
0.6	0.8

Classification scores:
P(cat)

Sliding Window: Overfeat



Network input:
3 x 221 x 221



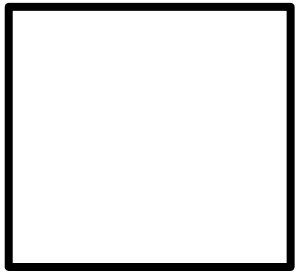
Larger image:
3 x 257 x 257

0.5	0.75
0.6	0.8

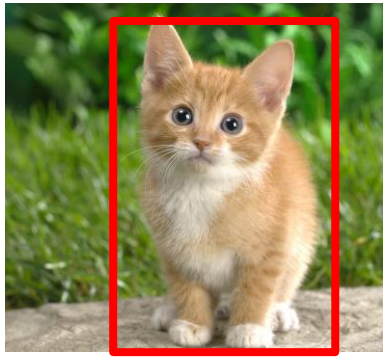
Classification scores:
P(cat)

Sliding Window: Overfeat

Greedily merge boxes and scores (details in paper)



Network input:
3 x 221 x 221



Larger image:
3 x 257 x 257

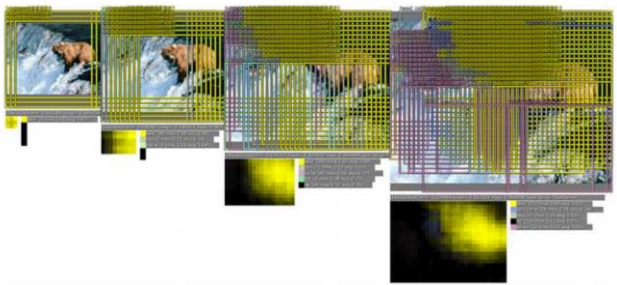
0.8

Classification score:
P(cat)

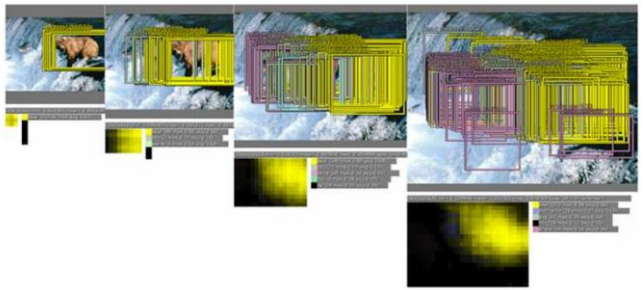
Sliding Window: Overfeat

In practice use many sliding window locations and multiple scales

Window positions + score maps



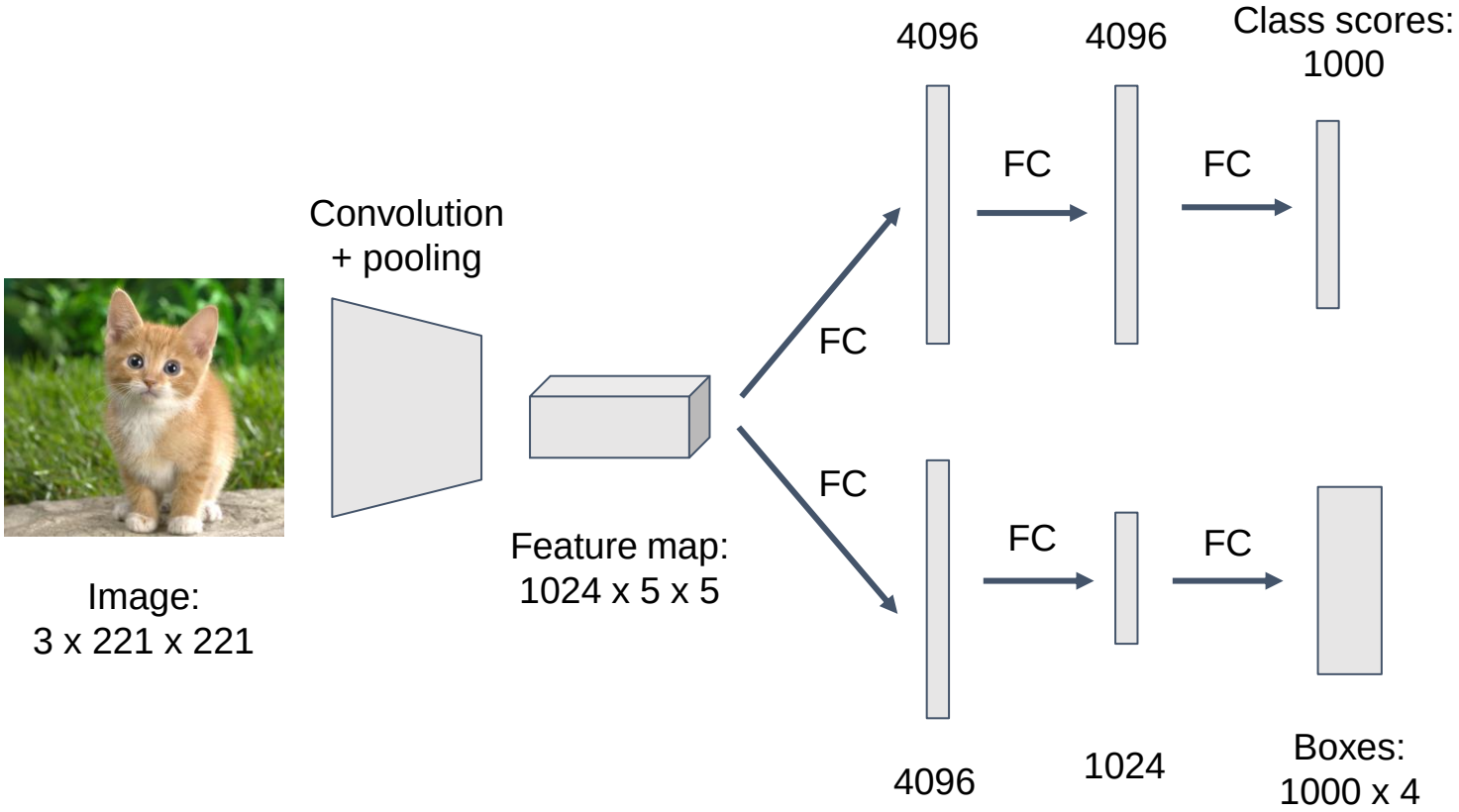
Box regression outputs



Final Predictions

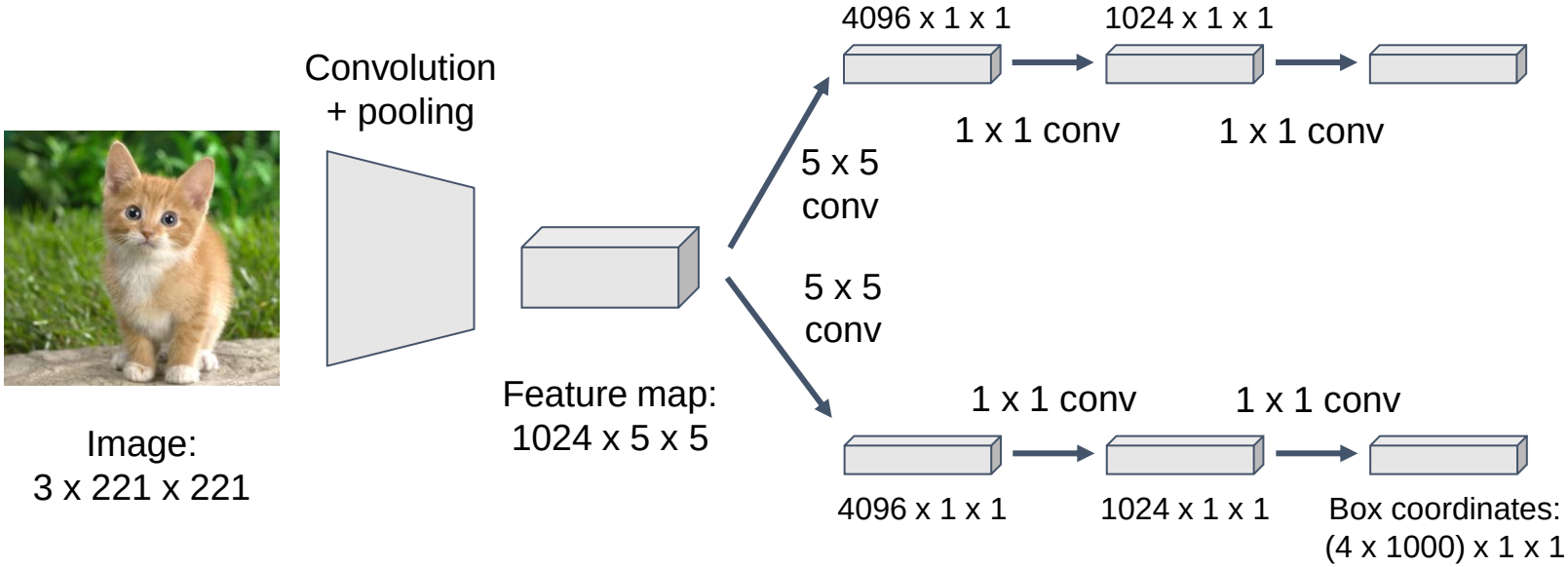
Sermanet et al, “Integrated Recognition, Localization and Detection using Convolutional Networks”, ICLR 2014

Efficient Sliding Window: Overfeat



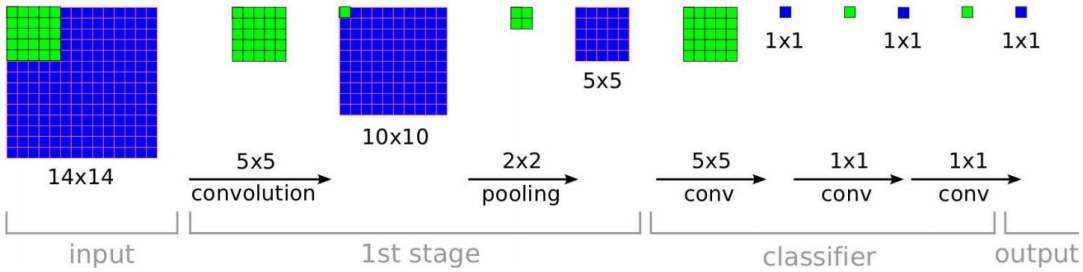
Efficient Sliding Window: Overfeat

Efficient sliding window by converting fully-connected layers into convolutions

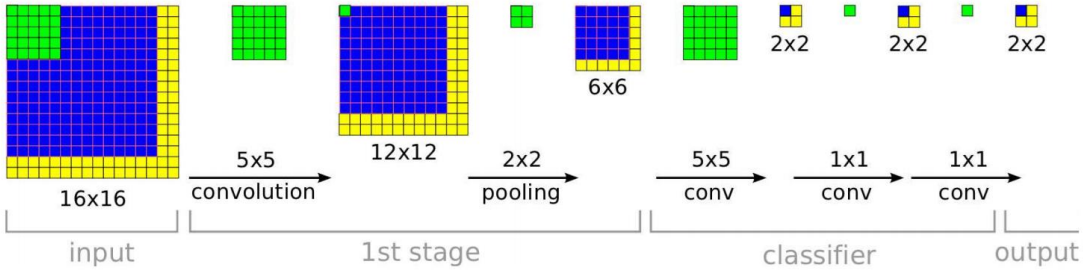


Efficient Sliding Window: Overfeat

Training time: Small image, 1 x 1 classifier output

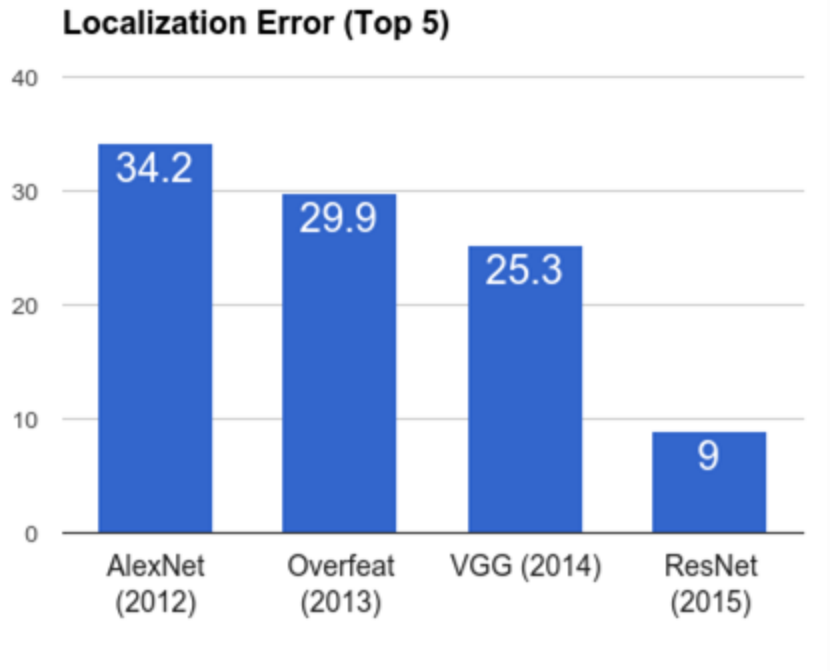


Test time: Larger image, 2 x 2 classifier output, only extra compute at yellow regions



Sermanet et al, “Integrated Recognition, Localization and Detection using Convolutional Networks”, ICLR 2014

ImageNet Classification + Localization



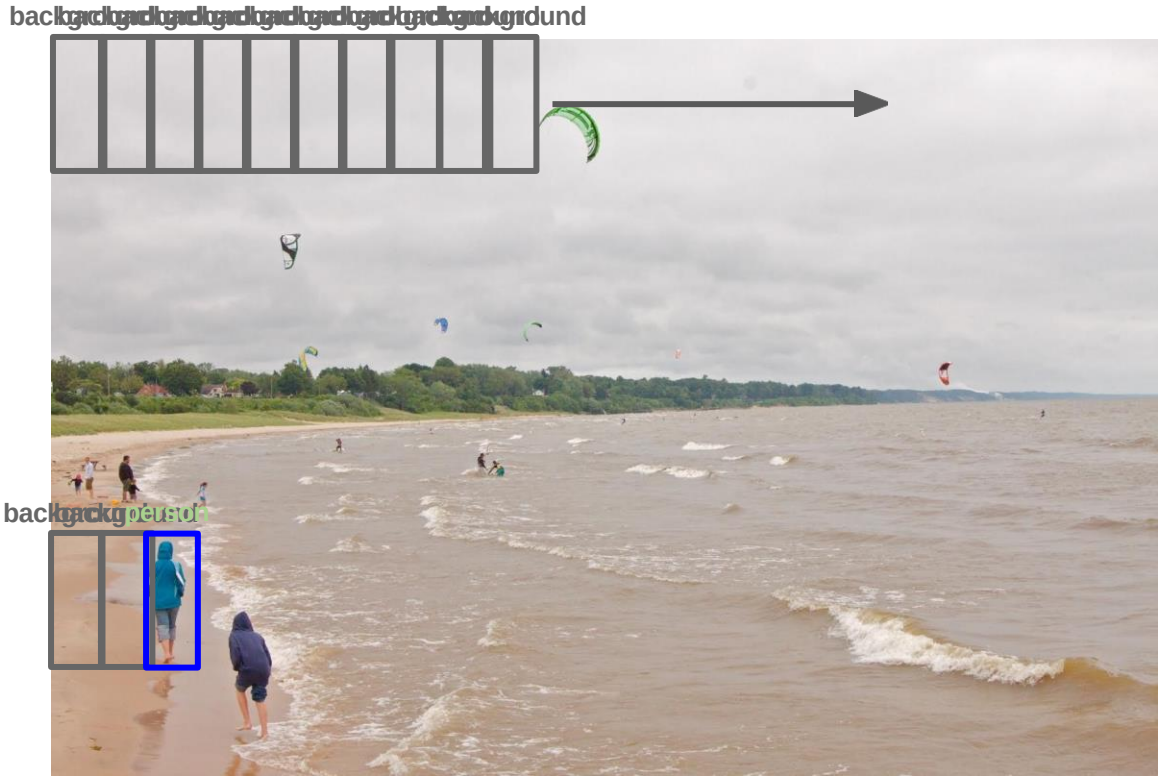
AlexNet: Localization method not published

Overfeat: Multiscale convolutional regression with box merging

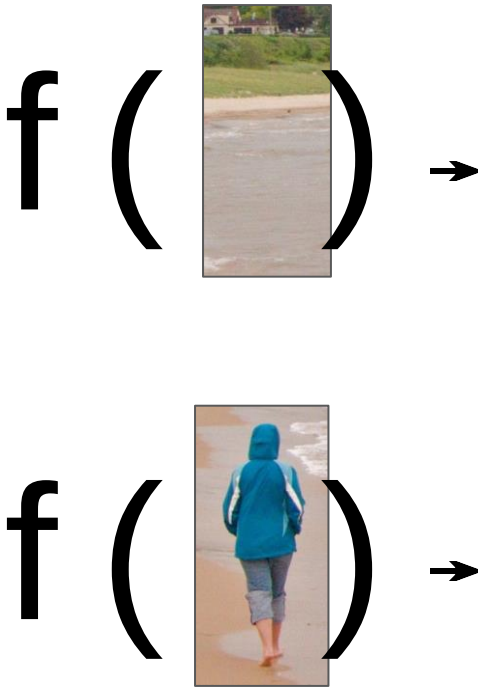
VGG: Same as Overfeat, but fewer scales and locations; simpler method, gains all due to deeper features

ResNet: Different localization method (Region Proposal Network - RPN) and much deeper features

“Sliding Window” Detection

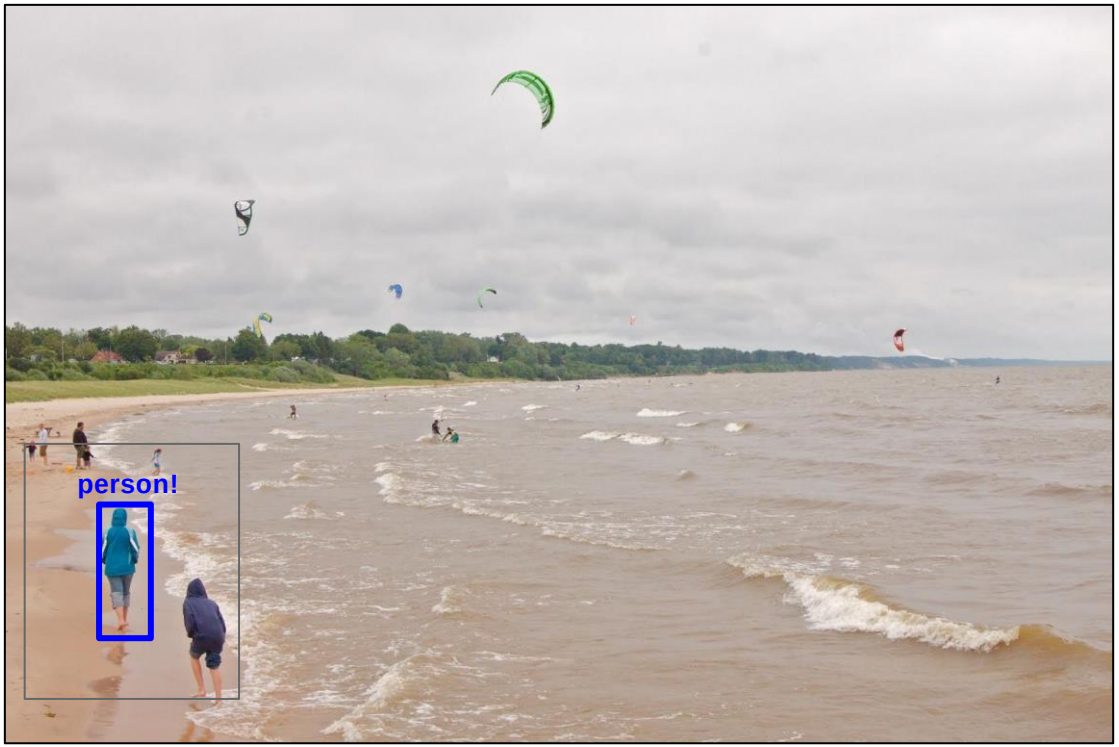


“Sliding Window” Detection



Compute within-region features, then classify

“Sliding Window” Detection



f ( )

f ( )

Typical to enlarge region to include some “context”

Sliding window placement

Slide over *fine grid*
in x, y, scale, aspect ratio

Slow and Accurate

Slide over *coarse grid*
in x, y, scale, aspect ratio

Fast and Not-so-accurate

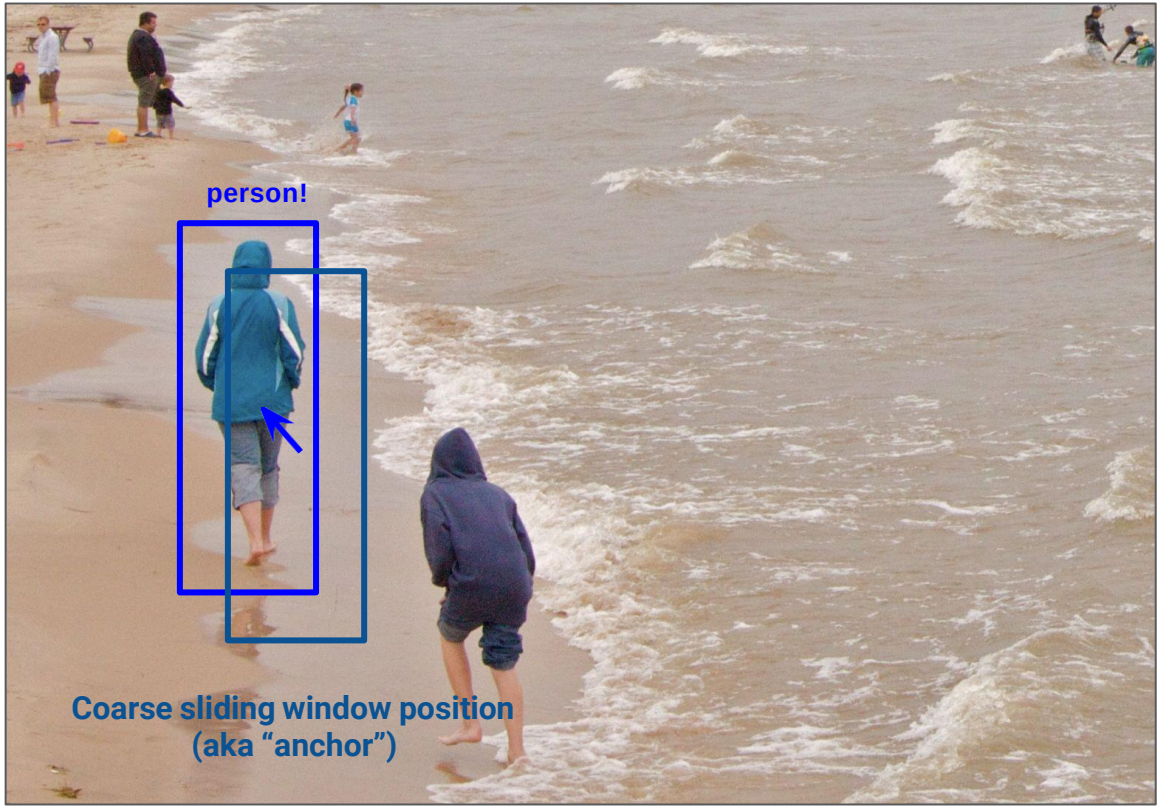
(… or can it be?)

Bounding Box Regression

Idea:

Also predict continuous offset from anchor to “snap” onto object

Bounding Box Regression



Idea:

Also predict continuous offset from anchor to “snap” onto object

Typical Training Objective

Common to use other location losses here...

Per-anchor Loss:

$$L(\text{anchor } \mathbf{a}) = \alpha * \delta(\mathbf{a} \text{ has matching groundtruth}) * L_2(\mathbf{t}^{loc}, W^{loc} \cdot \mathbf{v}_{ij})$$

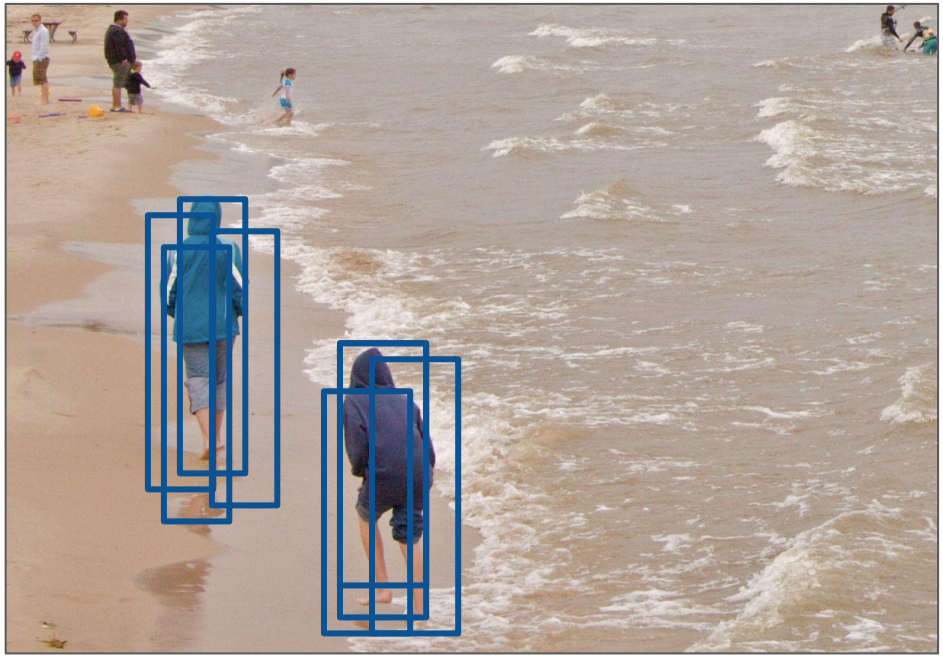
$$+ \beta * \text{SoftMaxCrossEntropy}(\mathbf{t}^{cls}, W^{cls} \cdot \mathbf{v}_{ij})$$

Total Loss: Average per-anchor loss over anchors

Challenge: Dealing with class imbalance (usually way more negative anchors (class 0) than positive anchors)

Solutions: Subsampling negative anchors, downweighting the loss contribution of negatives, hard mining, etc...

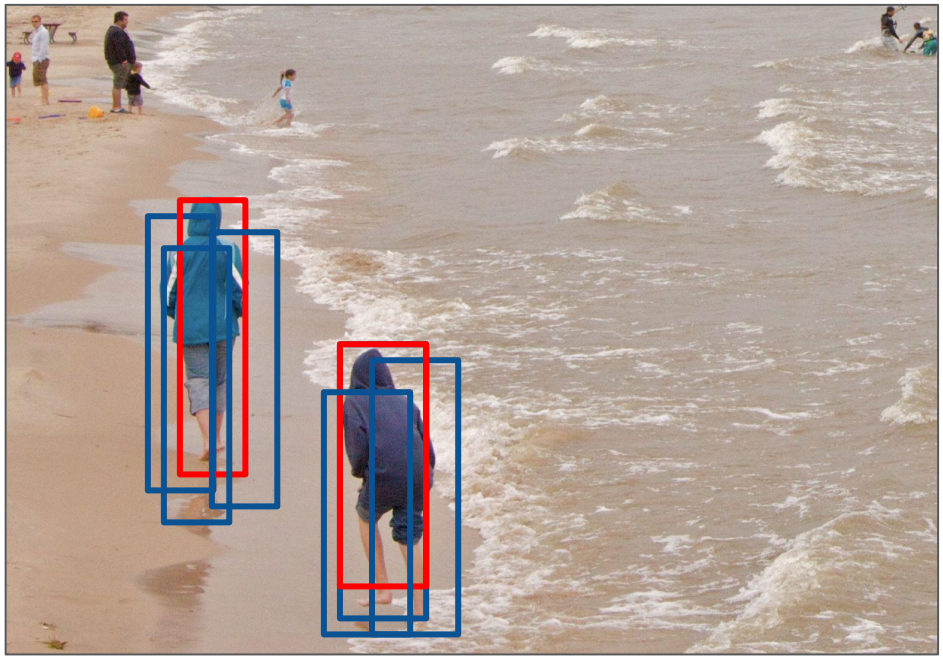
Dealing with multiple detections of the same object



Duplicate detection problem: Typically many anchors will detect the same underlying object and give slightly different boxes, with slightly different scores.

Solution: remove detections if they overlap too much with another higher scoring detection.

Non Max Suppression (NMS)



Algorithm:

1. **Sort detections in decreasing order with respect to score**
2. **Iterate through sorted detections:**
   Reject a detection if it overlaps with a previous (unrejected) detection with IOU greater than some threshold
3. **Return all unrejected detections**

Some shortcomings of NMS to remember:

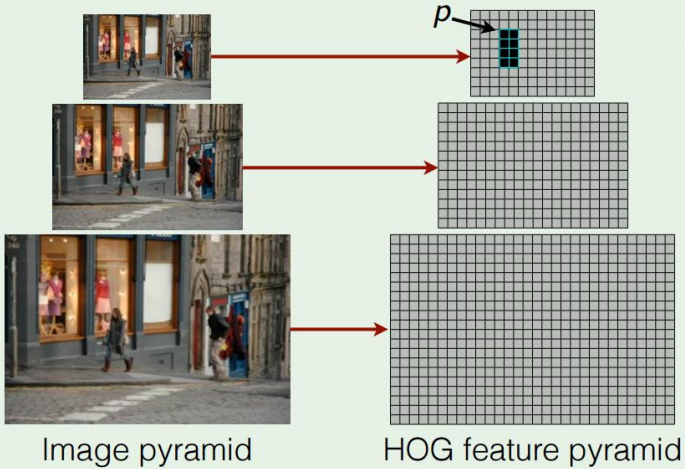
- **Imposes a hard limitation on how close objects can be in order to be detected**
- **Similar classes do not suppress each other**

Detection as Classification

Problem: Need to test many positions and scales

Solution: If your classifier is fast enough, just do it

Histogram of Oriented Gradients

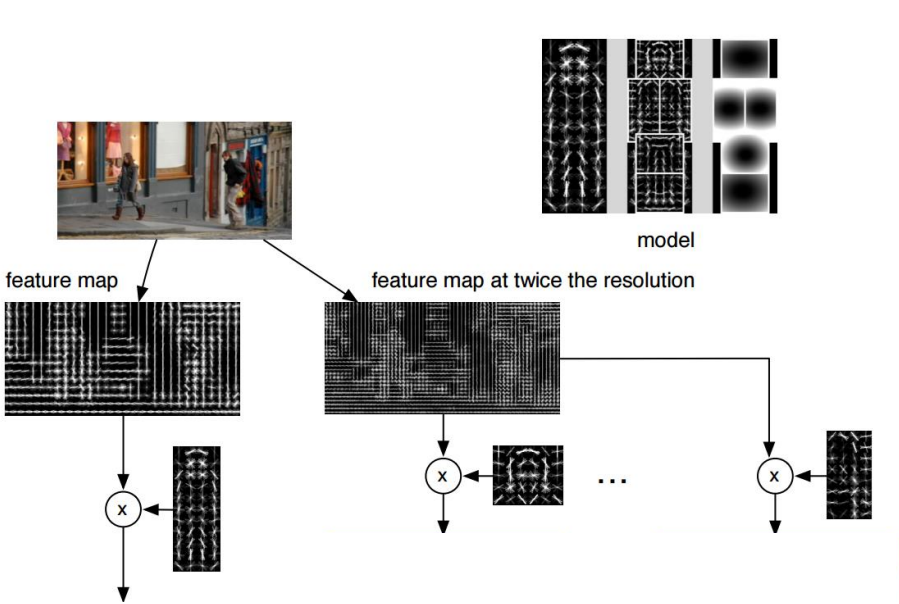


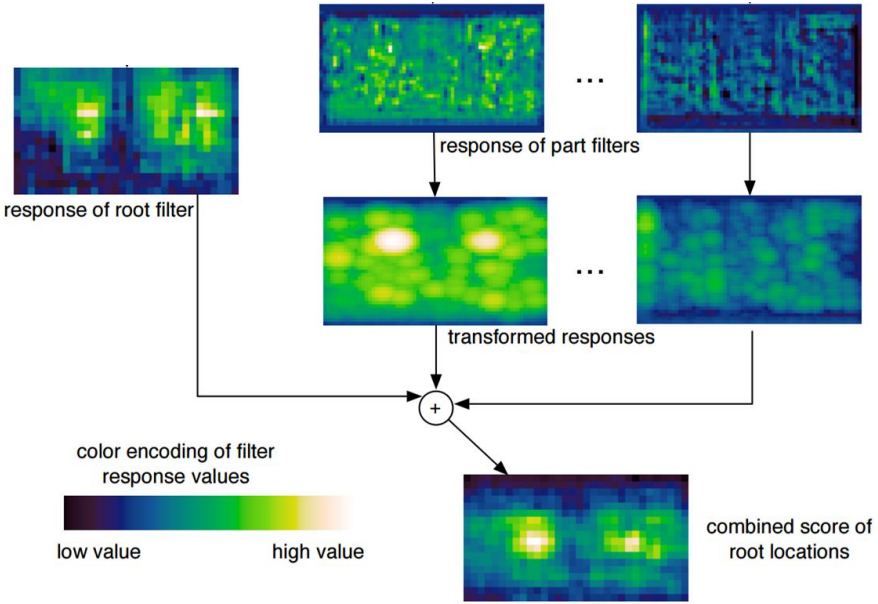
$$score(I, p) = \mathbf{w} \cdot \phi(I, p)$$

- Compute HOG of the whole image at multiple resolutions
- Score every subwindow of the feature pyramid
- Apply non-maxima suppression

Dalal and Triggs, “Histograms of Oriented Gradients for Human Detection”, CVPR 2005
Slide credit: Ross Girshick

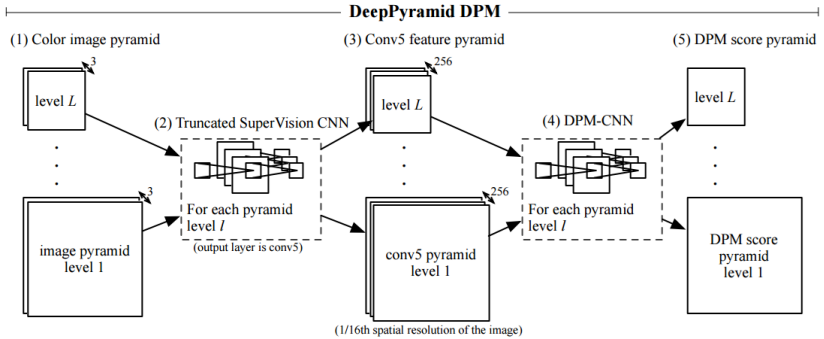
Deformable Parts Model (DPM)

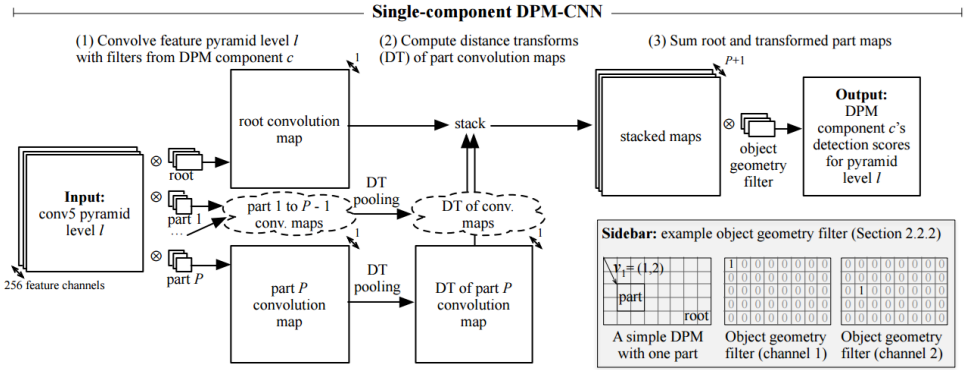




Felzenszwalb et al, “Object Detection with Discriminatively Trained Part Based Models”, PAMI 2010

Aside: Deformable Parts Models are CNNs?





Girschick et al, “Deformable Part Models are Convolutional Neural Networks”, CVPR 2015

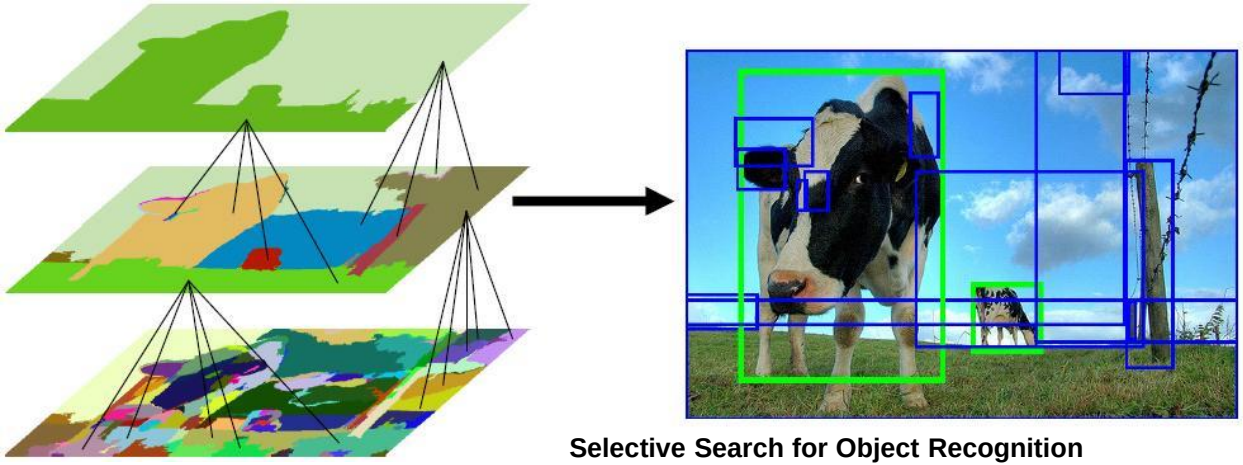
Detection as Classification

Problem: Need to test many positions and scales,
and use a computationally demanding classifier (CNN)

Solution: Only look at a tiny subset of possible positions

Idea 3: Object proposals

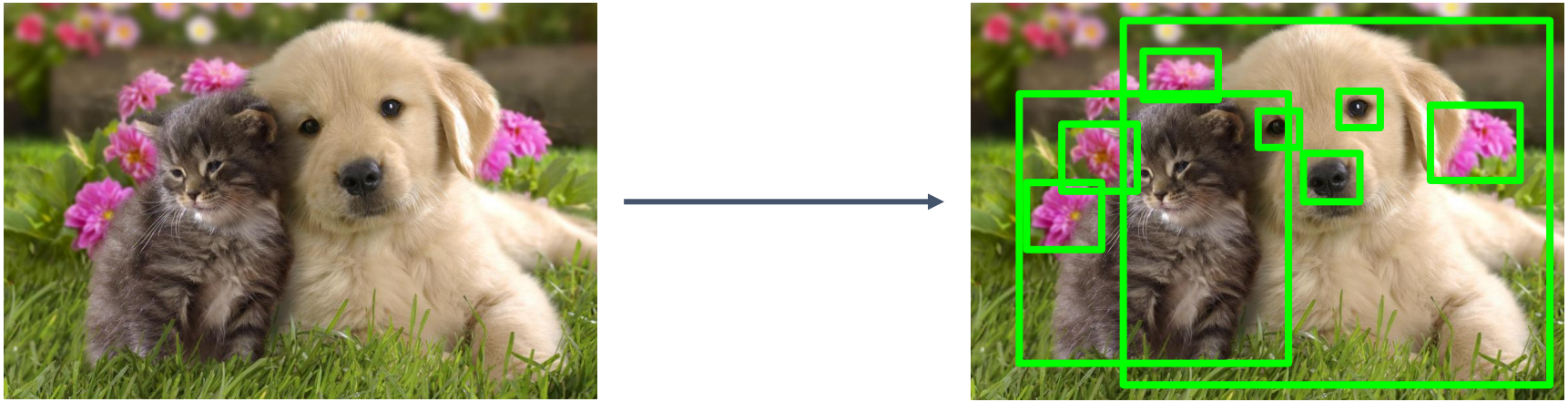
- Use segmentation to produce ~5K candidates



Selective Search for Object Recognition
J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, A. W. M. Smeulders
In International Journal of Computer Vision 2013.

Region Proposals

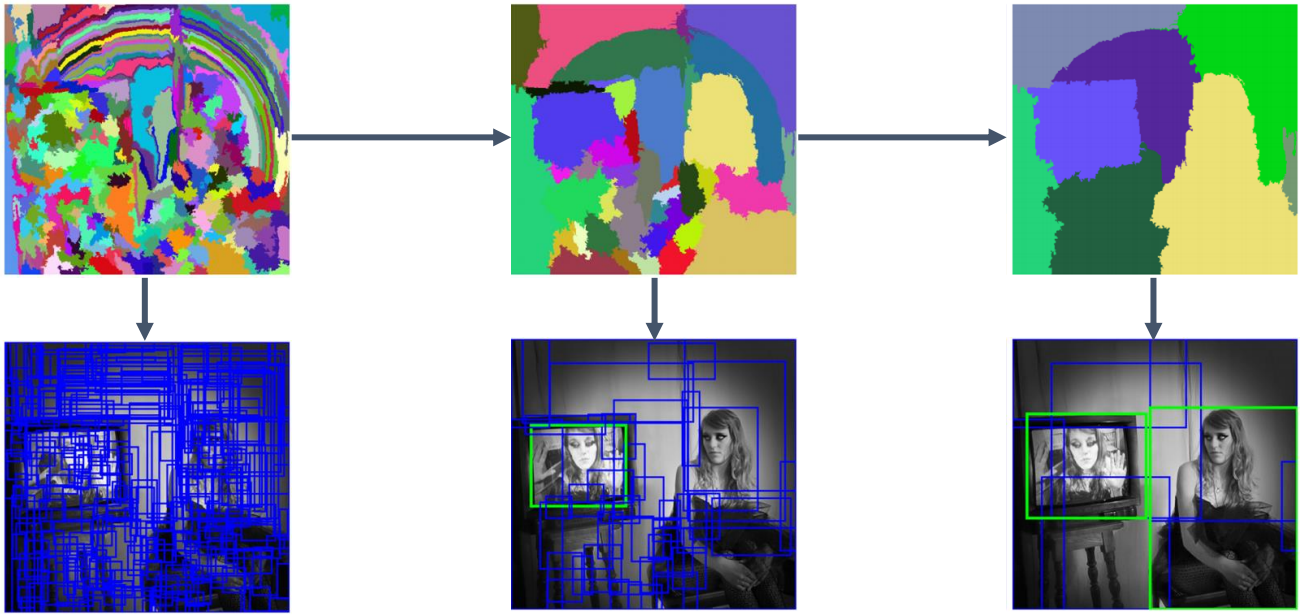
- Find “blobby” image regions that are likely to contain objects
- “Class-agnostic” object detector
- Look for “blob-like” regions



Region Proposals: Selective Search

Bottom-up segmentation, merging regions at multiple scales

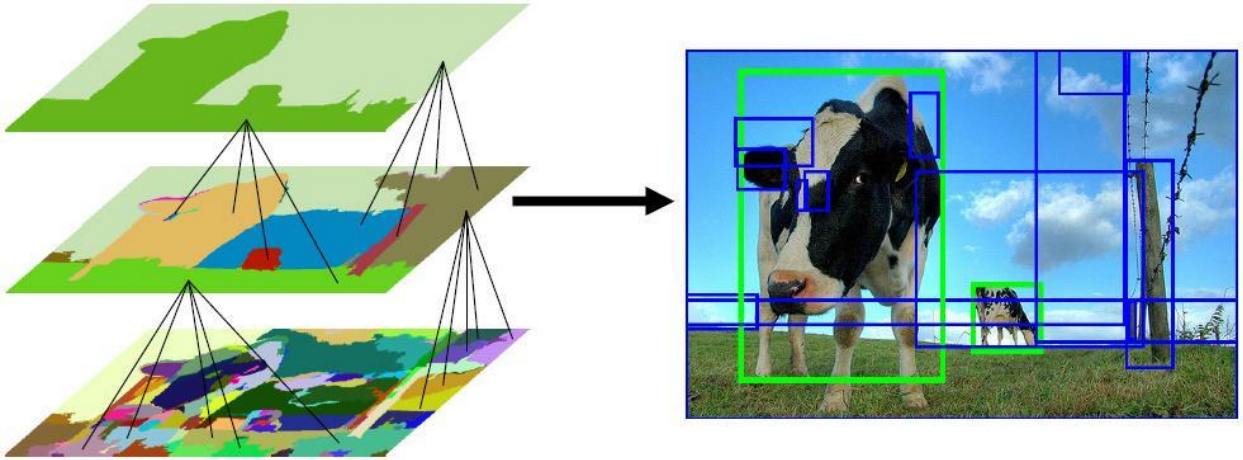
Convert regions to boxes



Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013

Region Proposals: Selective Search

- Use segmentation to produce less (~5K) candidates

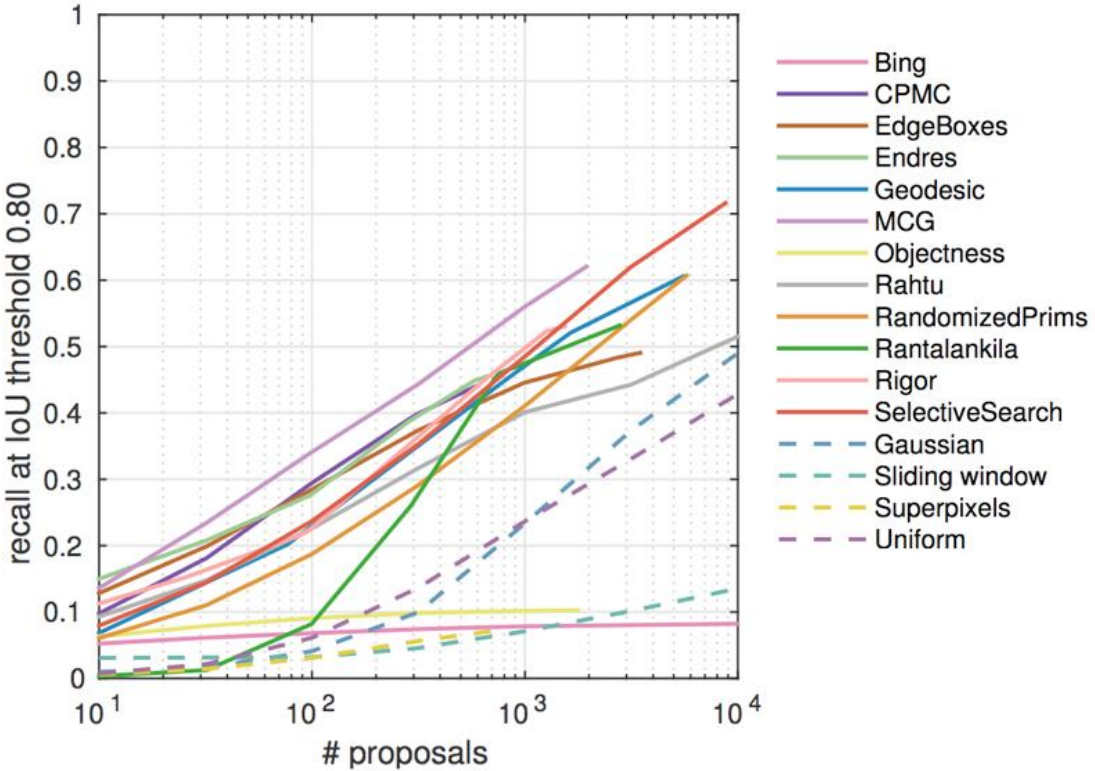


Object proposals

- Many different segmentation algorithms (k-means on color, k-means on color+position, N-cuts….)
- Many hyperparameters (number of clusters, weights on edges)
- Try everything!
  - Every cluster is a candidate object
  - Thousands of segmentations -> thousands of candidate objects

Object proposals

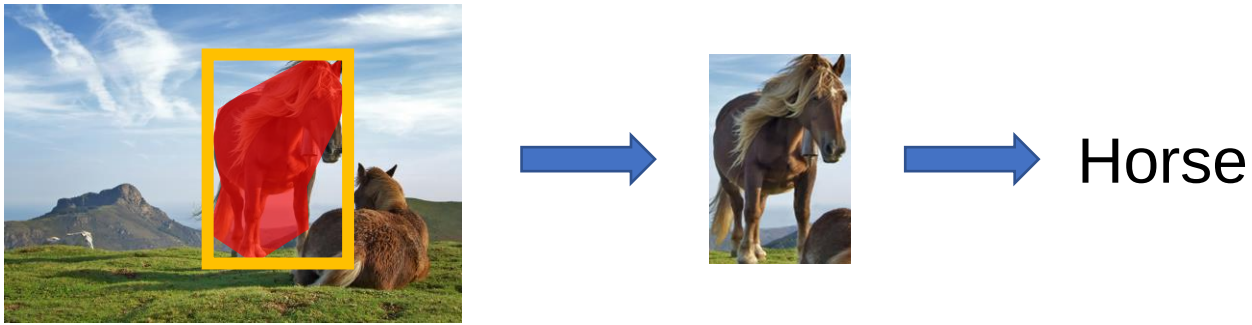
- Tens of ways of generating candidates (“proposals”)
- What fraction of ground truth objects have proposals near them?



What makes for effective detection proposals? J. Hosang, R. Benenson, P. Dollar, B. Schiele. In TPAMI

What do we do with proposals?

- Each proposal is a group of pixels
- Take a tight-fitting box and *classify it*
- *Can leverage any image classification approach*

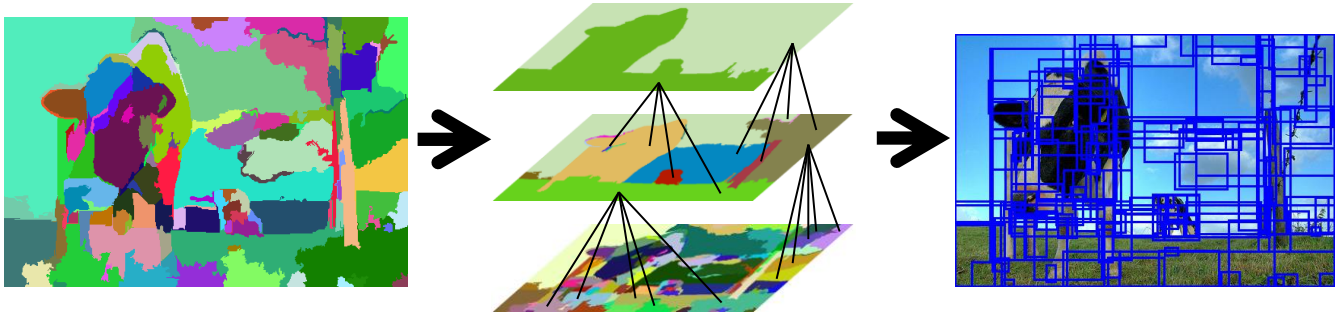


Segmentation as Selective Search

- Rethink segmentation:
  - High Recall
  - Coarse locations are sufficient (boxes)
  - Fast to compute

Segmentation as Selective Search

- An image is intrinsically hierarchical. A segmentation at a single scale cannot find all objects
- Use all locations from a hierarchical grouping



Oversegmentation
(Felzenszwalb 2004)

Hierarchical grouping
of segments

Object hypotheses from
all hierarchy levels

Segmentation as Selective Search

Color cues work best

Texture cues work, color fails

- No single segmentation strategy works everywhere
- We need a set of complementary segmentation strategies

Segmentation as Selective Search

- Hierarchical Grouping
- Use of a variety of color spaces with complementary invariance properties
- Different grouping criteria: Colour, Texture, Size, Insideness
- 2 methods:
  - Fast: uses 8 different hierarchical groupings
  - Quality: uses 80 different hierarchical groupings

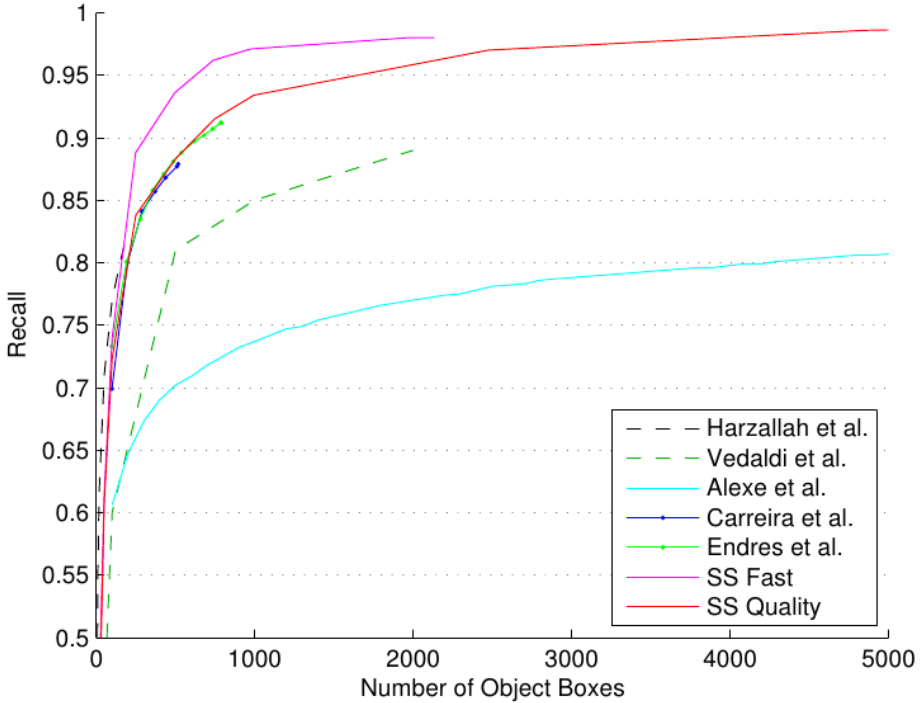
Van de Sande, et al, ICCV 2011

Segmentation as Selective Search

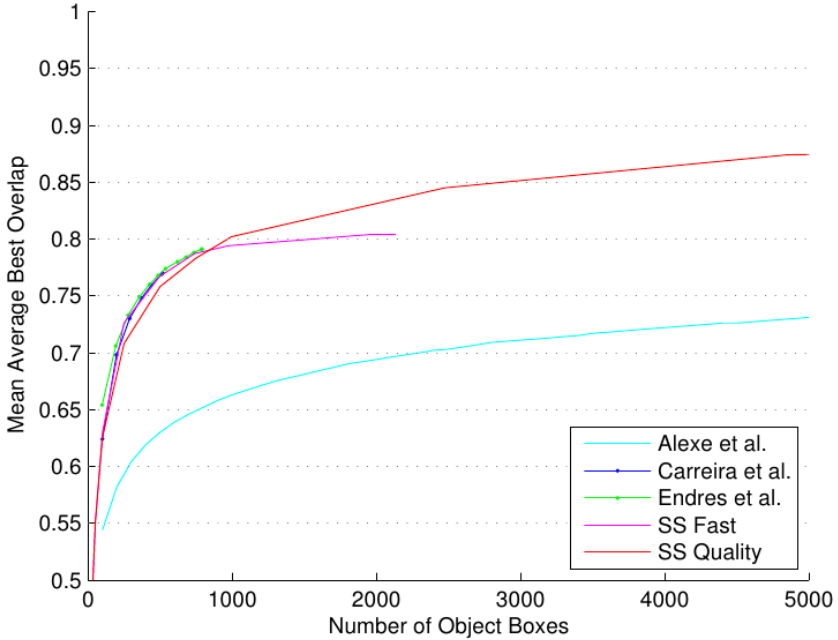
Version	Diversification Strategies	MABO	# windows	time (s)
Single Grouping	HSV C+R+S+F $k = 100$	0.693	362	0.71
Structured Sampling Fast	HSV, Lab C+T+S+F, T+S+F $k = 50, 100$	0.799	2147	3.79
Structured Sampling Quality	HSV, Lab, rgI, H, I C+T+S+F, T+S+F, F, S $k = 50, 100, 150, 300$	0.878	10,108	17.15

MABO: Mean Average Best Overlap
rgI is normalized R and G and intensity. H is the Hue from HSV.
C = color, T = texture, S = Size, F = Fill/Insideness
k is the parameter for the initial oversegmentation. Higher k means fewer, larger initial regions

Evaluation of Locations



Evaluation of Locations

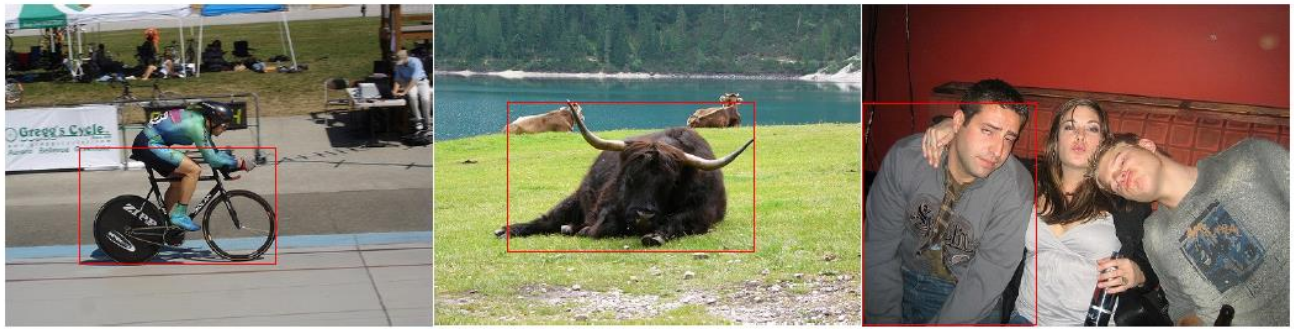


Evaluation of Locations

method	recall	MABO	# windows
Arbelaez *et al.* [2]	0.752	0.649	418
Alexe *et al.* [1]	0.824	0.747	10,000
Harzallah *et al.* [14]	0.830	-	200 per class
Carreira *et al.* [3]	0.879	0.770 ± 0.084	517
Endres *et al.* [8]	0.912	0.791 ± 0.082	790
Felzenszwalb *et al.* [10]	0.933	0.829 ± 0.052	100,352 per class
Vedaldi *et al.* [29]	0.940	-	10,000 per class
Single Grouping	0.840	0.690	289
SS "Fast"	0.980	0.804 ± 0.046	2,134
SS "Quality"	0.991	0.879 ± 0.039	10,097

Evaluation of Locations

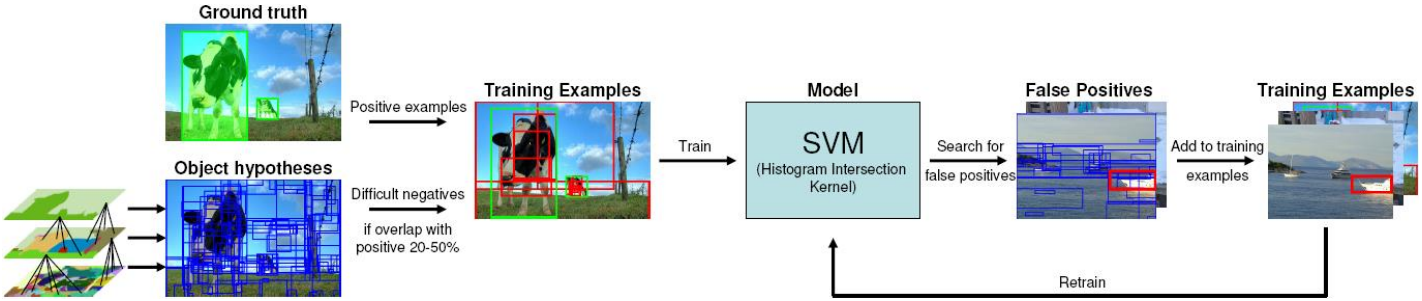
- What does a .88 Best Overlap score mean?



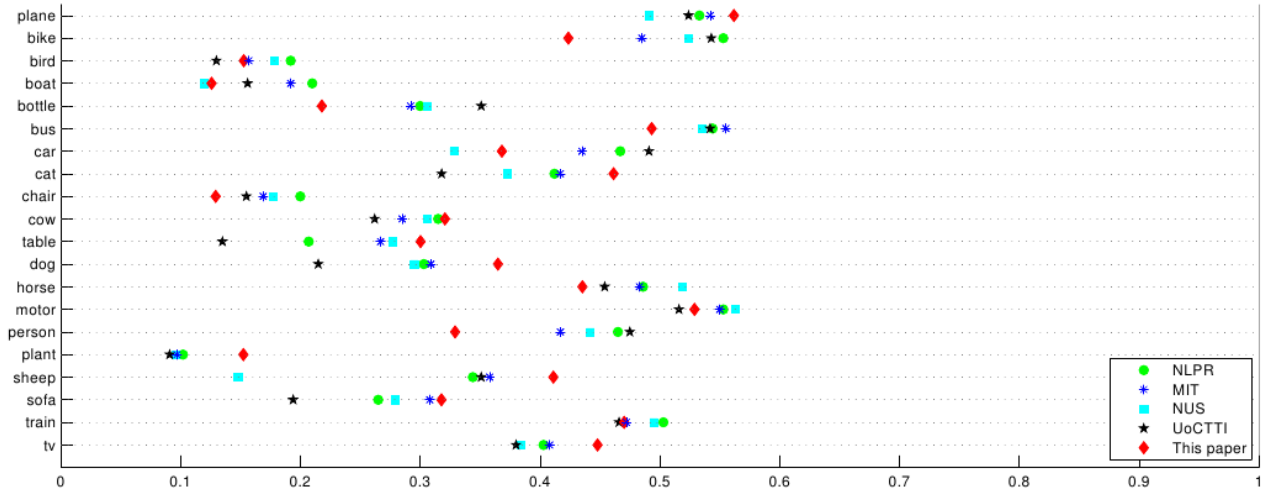
Overlap 88.4% Overlap 87.9% Overlap 87.4%

Selective Search in Object Localization

- Goal: To identify and find the location of the objects. An object is found if the Pascal Overlap (MABO) score > 50%



Selective Search in Object Localization



Pascal VOC 2010

- Best results for 9 out of 20 object classes
- Works especially well on non-rigid object classes
- All competing methods are based on exhaustive search with HOG-features

Van de Sande, et al. ICCV 2011

Proposal methods results

	VOC 2007	VOC 2010
DPM v5 (Girshick et al. 2011)	33.7%	29.6%
UVA sel. search (Uijlings et al. 2013)		35.1%

Reference systems

Slide credit : Ross Girshick

Proposal methods results

	VOC 2007	VOC 2010
DPM v5 (Girshick et al. 2011)	33.7%	29.6%
UVA sel. search (Uijlings et al. 2013)		35.1%
Regionlets (Wang et al. 2013)	**41.7%**	39.7%
SegDPM (Fidler et al. 2013)		**40.4%**

Reference systems

Slide credit : Ross Girshick

Conclusions Selective Search

- Results in a small yet high quality set of potential object locations
- Works by rethinking segmentation:
  - Focus more on Recall than Precision
  - Hierarchical grouping to deal with objects at multiple scales
  - Multiple complementary strategies to deal with high variety in image conditions
- Enables use of more expensive features

R-CNN

Input image

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014..

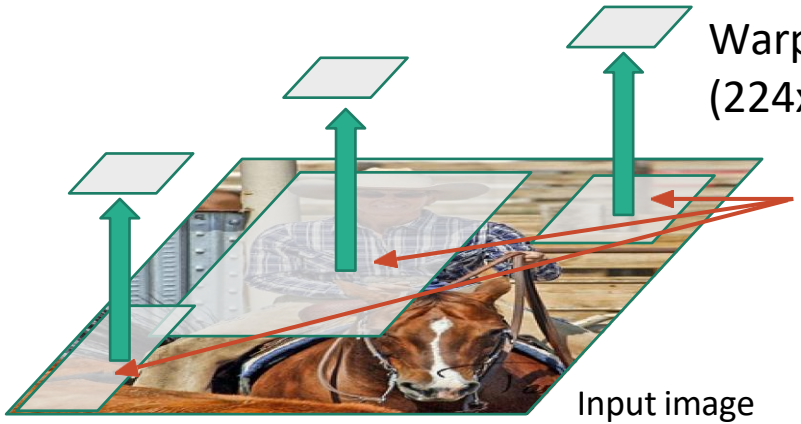
R-CNN

Input image

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

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.

R-CNN



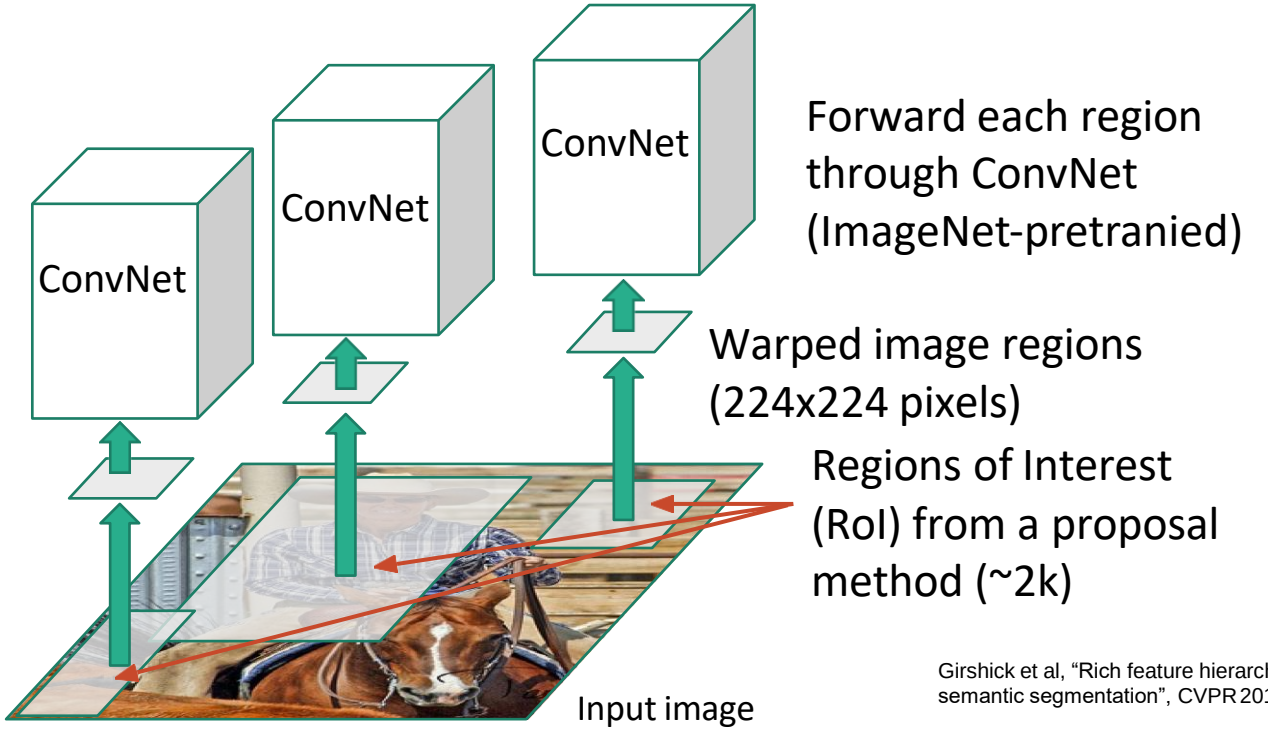
Warped image regions (224x224 pixels)

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

Input image

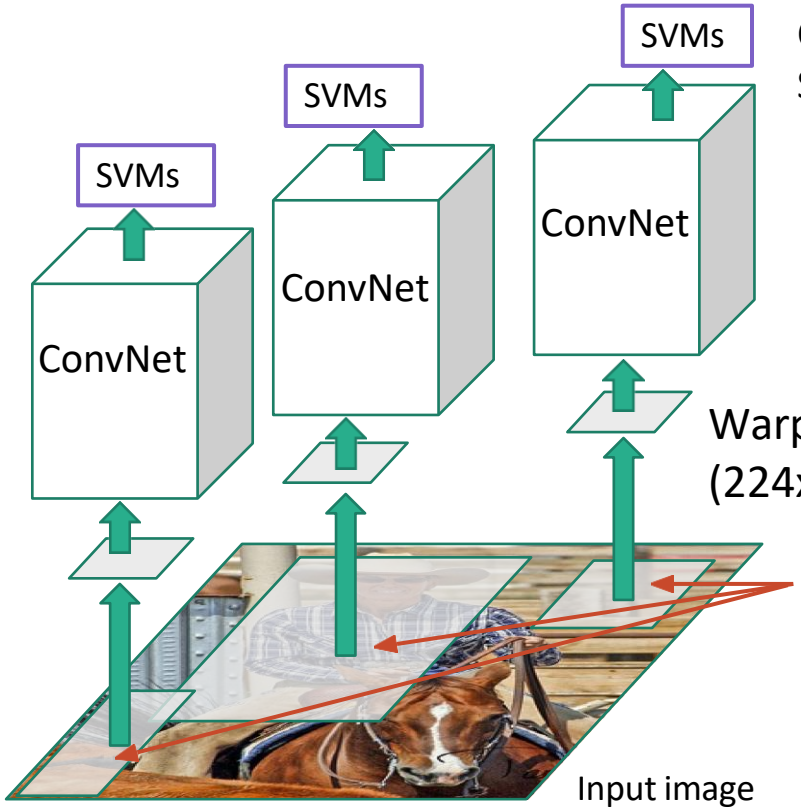
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014..

R-CNN



Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014..

R-CNN



Classify regions with SVMs

Forward each region through ConvNet (ImageNet-pretranied)

Warped image regions (224x224 pixels)

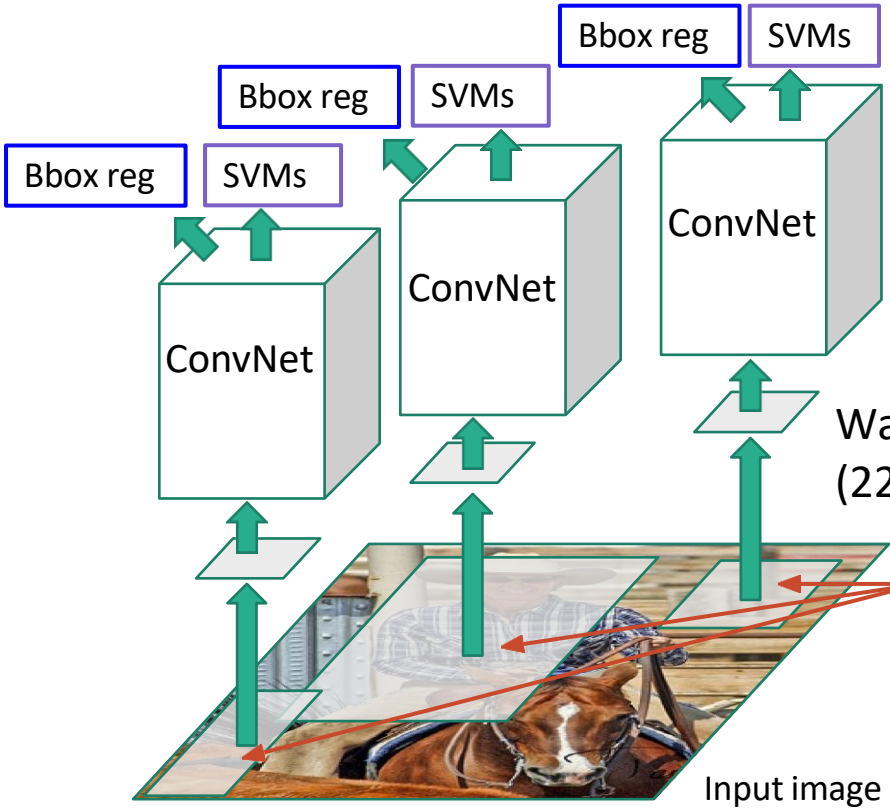
Regions of Interest (RoI) from a proposal method (~2k)

Input image

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014..

R-CNN

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



Classify regions with SVMs

Forward each region through ConvNet (ImageNet-pretranied)

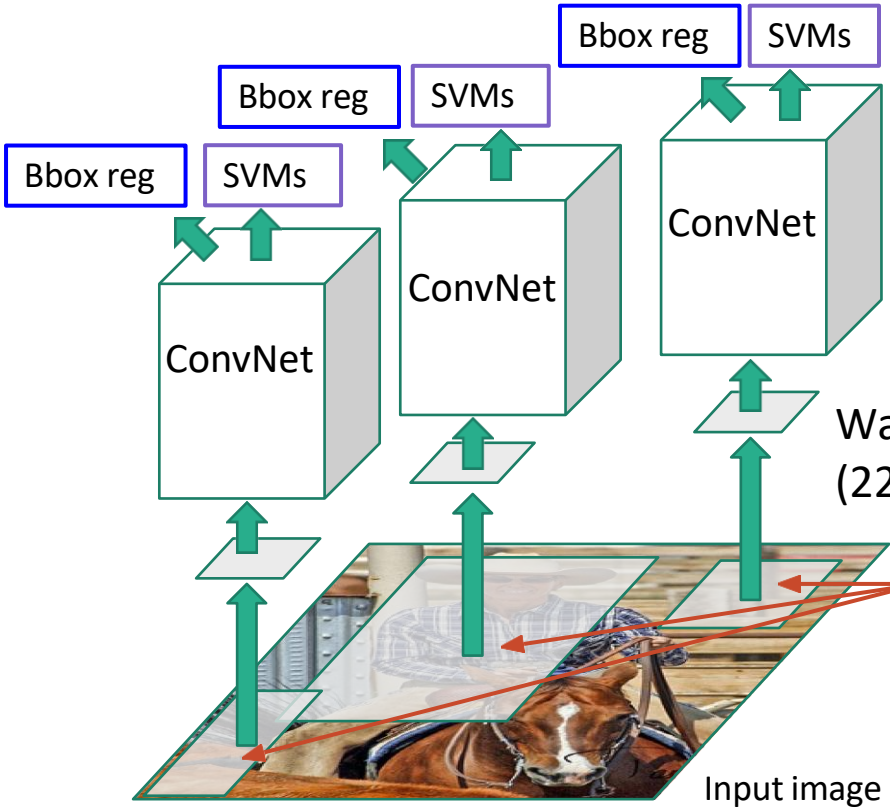
Warped image regions (224x224 pixels)

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

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014..

R-CNN

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



Classify regions with SVMs

Forward each region through ConvNet

Warped image regions (224x224 pixels)

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

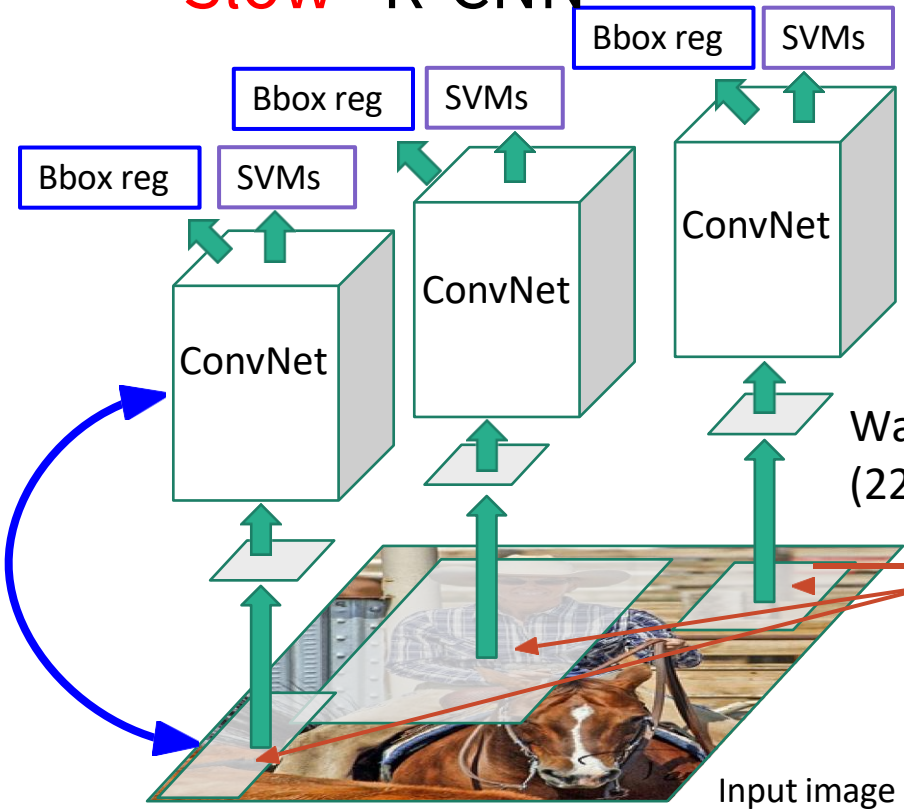
Input image

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

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014..

"Slow" R-CNN

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



Classify regions with SVMs

Forward each region through ConvNet

Warped image regions (224x224 pixels)

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

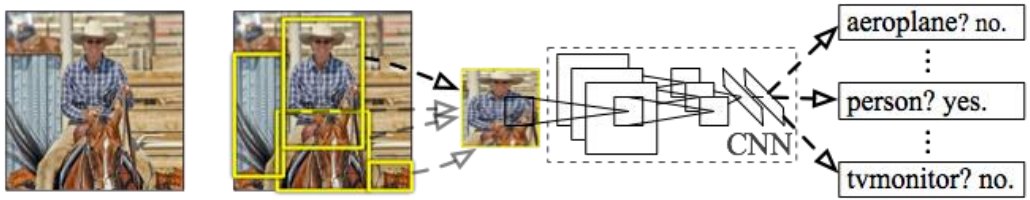
Input image

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

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

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014..

R-CNN: Regions with CNN features



Input image | Extract region proposals (~2k / image) | Compute CNN features | Classify regions (linear SVM)

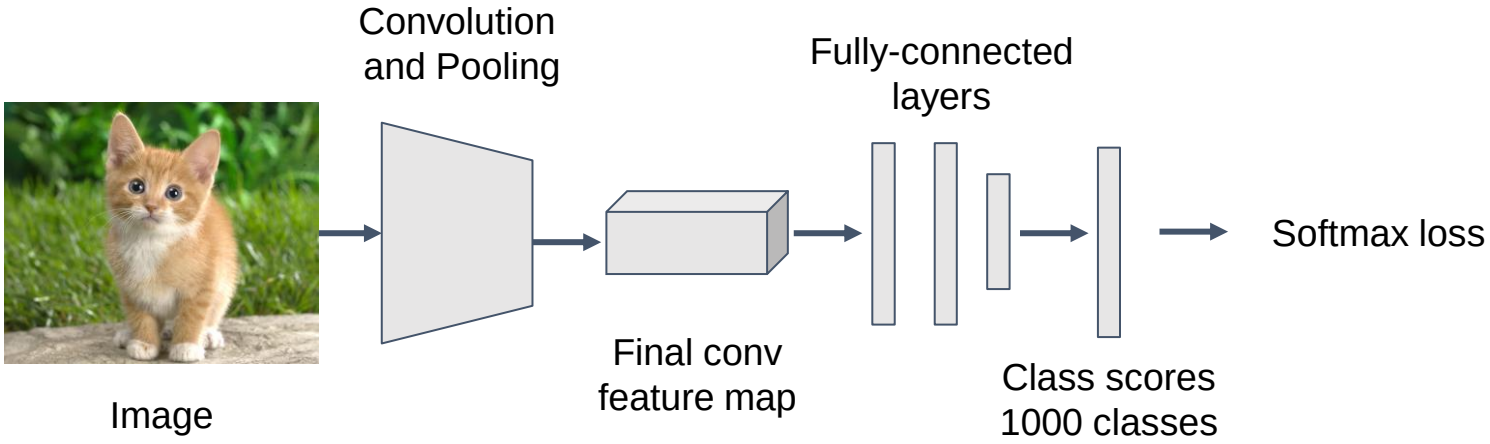
Compared to previous state-of-the-art methods, R-CNN is relatively simple and does not rely on an ensemble of classifiers

Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation
R. Girshick, J. Donahue, T. Darrell, J. Malik
IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014

Slide credit : Ross Girshick

R-CNN Training

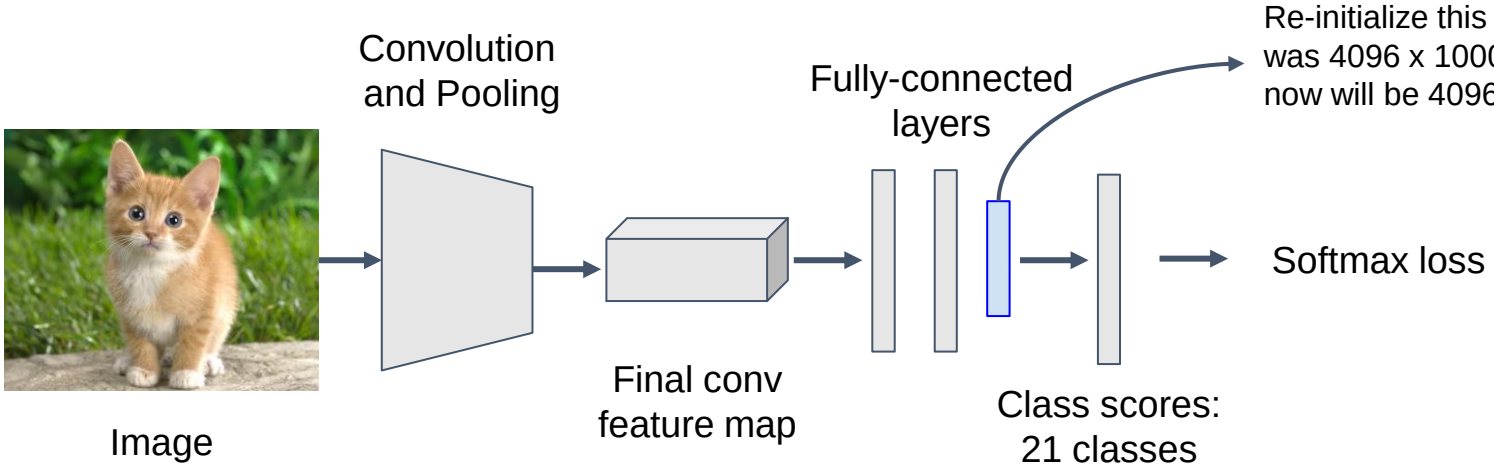
Step 1: Train (or download) a classification model for ImageNet (AlexNet)



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/negative regions from detection images

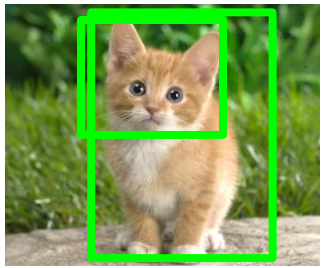


R-CNN Training

Step 3: Extract features

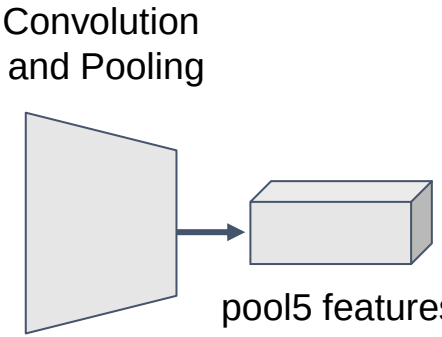
- Extract region proposals for all images
- For each region: warp to CNN input size, run forward through CNN, save pool5 features to disk
- Have a big hard drive: features are ~200GB for PASCAL dataset!

Image

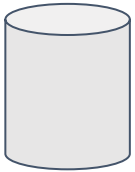


Region Proposals

Crop + Warp



Forward pass



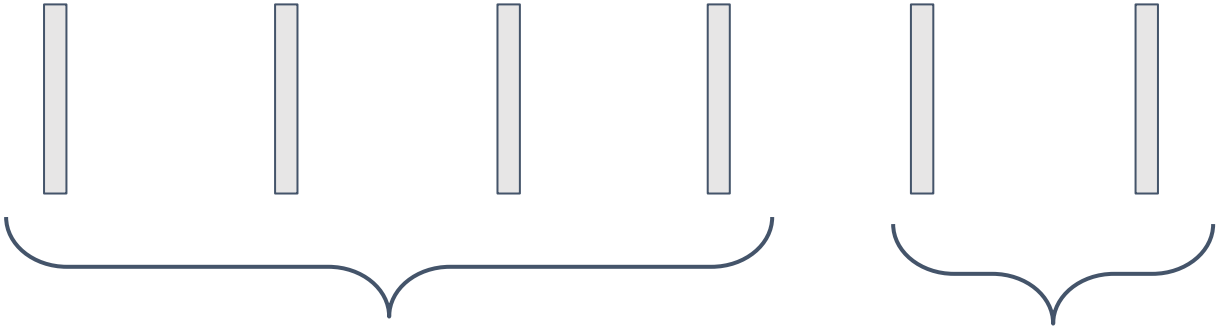
Save to disk

R-CNN Training

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

Training image regions

Cached region features



Positive samples for cat SVM

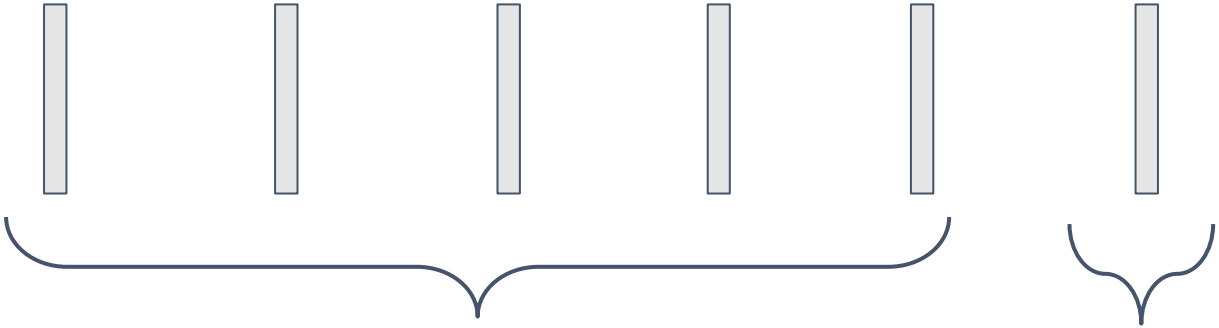
Negative samples for cat SVM

R-CNN Training

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

Training image regions

Cached region features



Negative samples for dog SVM

Positive samples for dog SVM

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

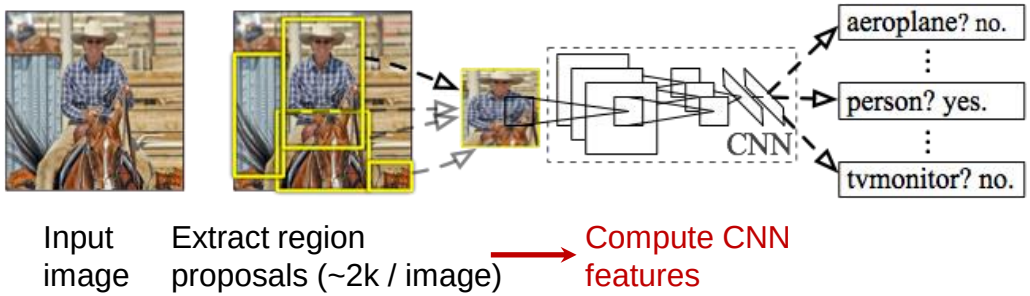
Regression targets
(dx, dy, dw, dh)
Normalized coordinates

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

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

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

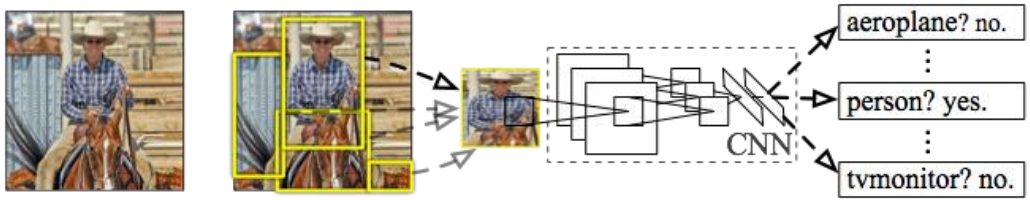
R-CNN at test time: Step 1



a. Crop

Slide credit : Ross Girshick

R-CNN at test time: Step 2



Input image

Extract region proposals (~2k / image)

Compute CNN features

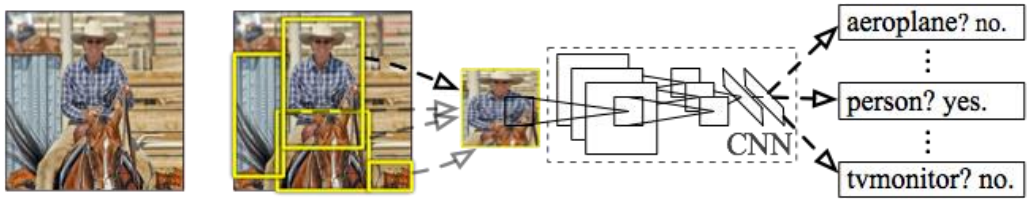
a. Crop

227 x 227

b. Scale (anisotropic)

Slide credit : Ross Girshick

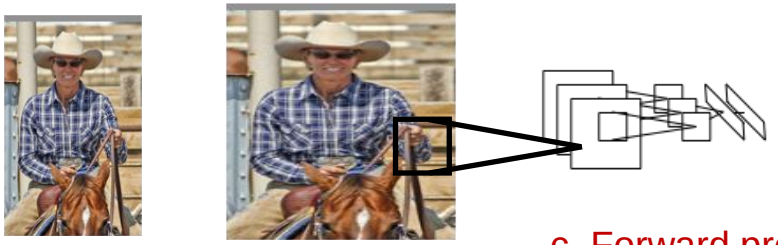
R-CNN at test time: Step 2



Input image

Extract region proposals (~2k / image)

Compute CNN features



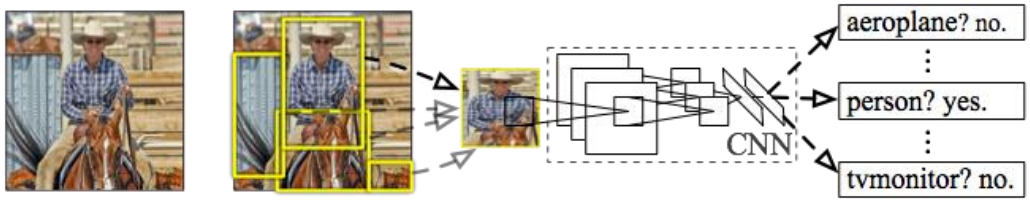
1. Crop

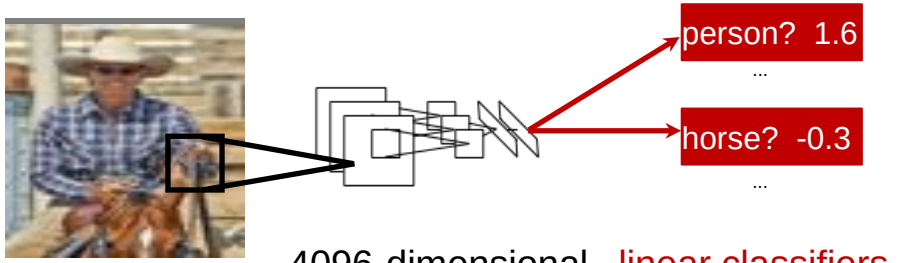
b. Scale (anisotropic)

c. Forward propagate
Output: “fc_7” features

Slide credit : Ross Girshick

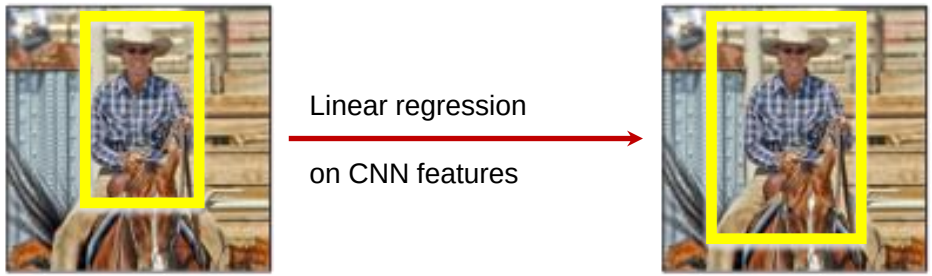
R-CNN at test time: Step 3





Slide credit : Ross Girshick

Step 4: Object proposal refinement



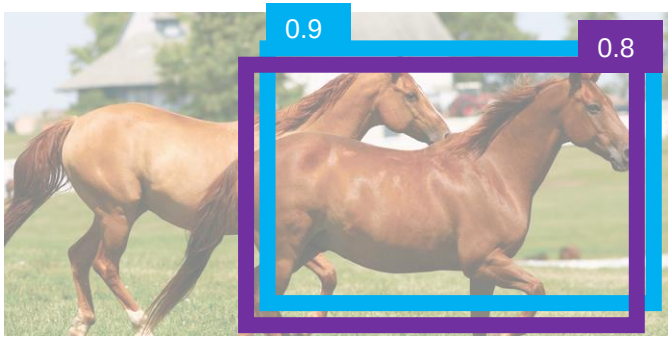
Original
proposal

Predicted
object bounding box

Bounding-box regression

Slide credit : Ross Girshick

Other details - Non-max suppression



How do we deal with multiple detections on the same object?

Other details - Non-max suppression

- Go down the list of detections starting from highest scoring
- Eliminate any detection that overlaps highly with a higher scoring detection
- Separate, heuristic step

Object Detection: Datasets

	PASCAL VOC (2010)	ImageNet Detection (ILSVRC 2014)	MS-COCO (2014)
Number of classes	20	**200**	80
Number of images (train + val)	~20k	**~470k**	~120k
Mean objects per image	2.4	1.1	**7.2**

Object Detection: Evaluation

Use “mean average precision” (mAP)

Compute average precision (AP) separately for each class, then average over classes

A detection is a true positive if it has IoU with a ground-truth box greater than some threshold (usually 0.5) (mAP@0.5)

Combine all detections from all test images to draw a precision / recall curve for each class; AP is area under the curve

TL;DR mAP is a number from 0 to 100; high is good

R-CNN results on PASCAL

	VOC 2007	VOC 2010
DPM v5 (Girshick et al. 2011)	33.7%	29.6%
UVA sel. search (Uijlings et al. 2013)		35.1%
Regionlets (Wang et al. 2013)	**41.7%**	39.7%
SegDPM (Fidler et al. 2013)		**40.4%**

Reference systems

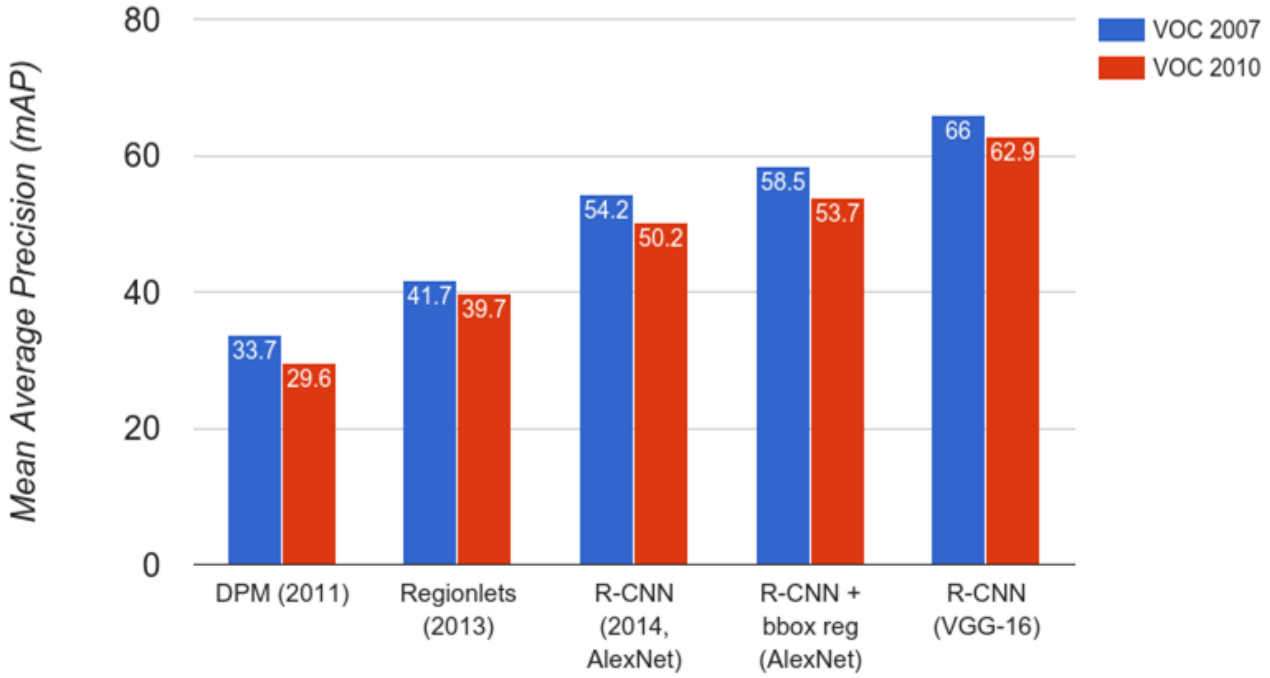
Slide credit : Ross Girshick

R-CNN results on PASCAL

	VOC 2007	VOC 2010
DPM v5 (Girshick et al. 2011)	33.7%	29.6%
UVA sel. search (Uijlings et al. 2013)		35.1%
Regionlets (Wang et al. 2013)	41.7%	39.7%
SegDPM (Fidler et al. 2013)		40.4%
R-CNN	54.2%	50.2%
R-CNN + bbox regression	58.5%	53.7%

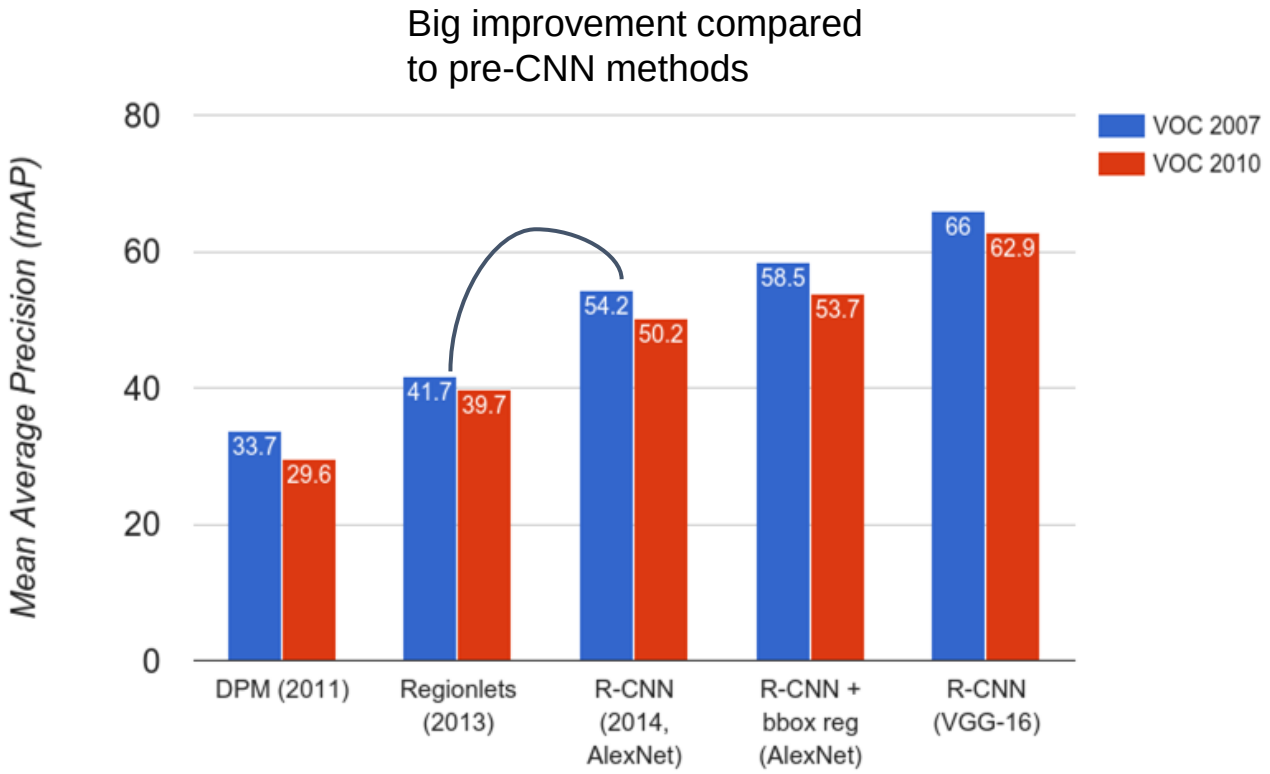
Slide credit : Ross Girshick

R-CNN Results

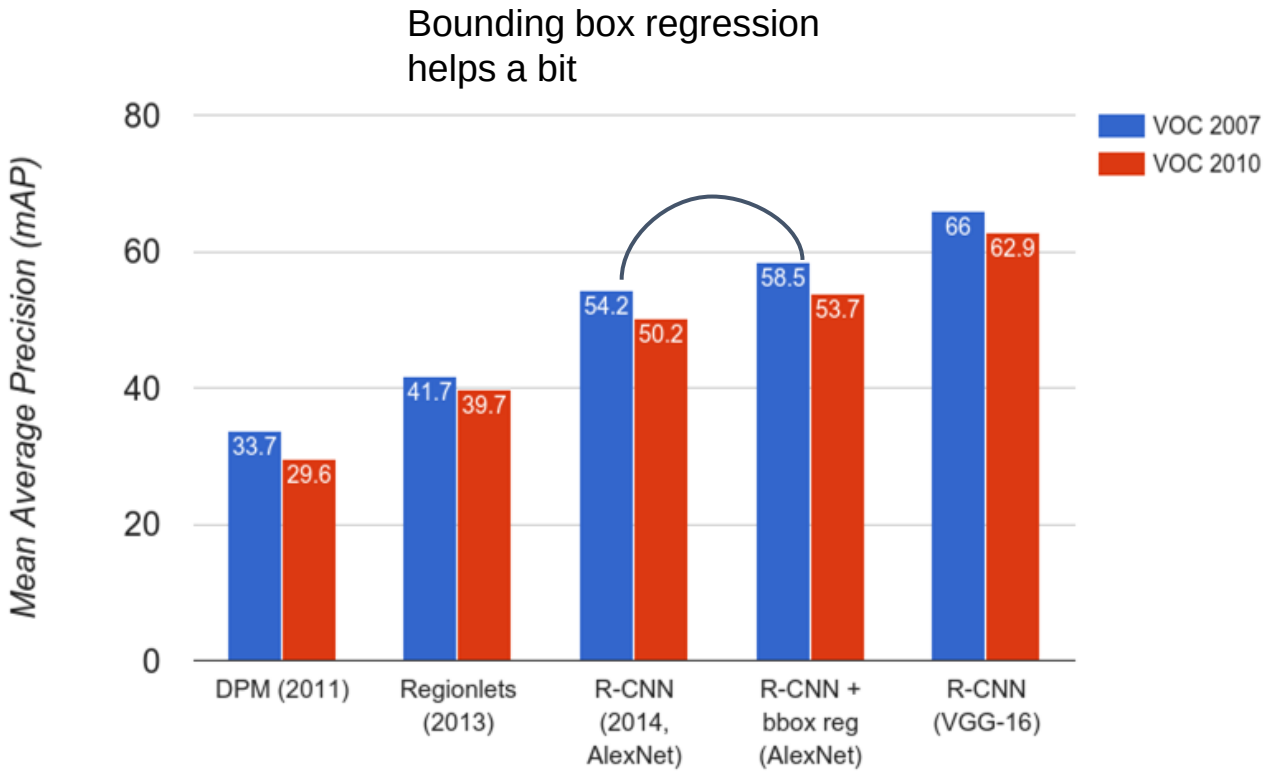


Wang et al, “Regionlets for Generic Object Detection”, ICCV 2013

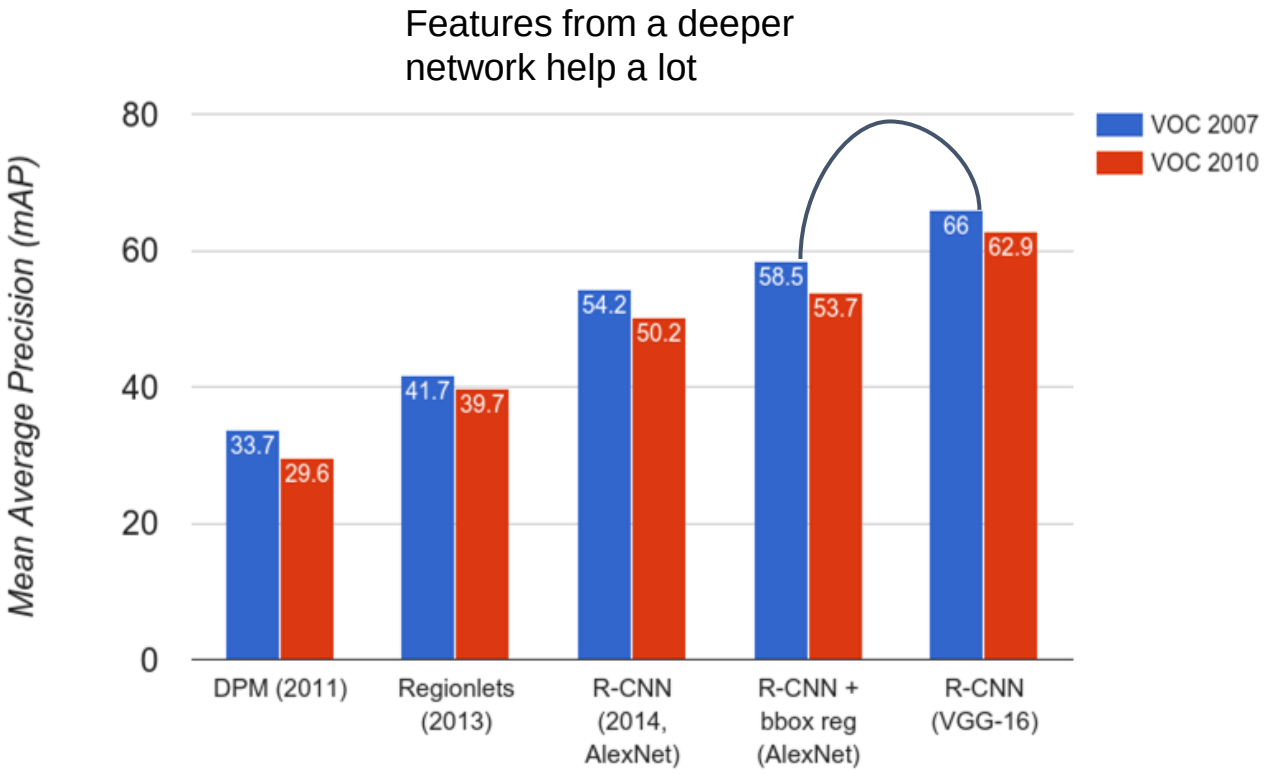
R-CNN Results



R-CNN Results



R-CNN Results



R-CNN Problems

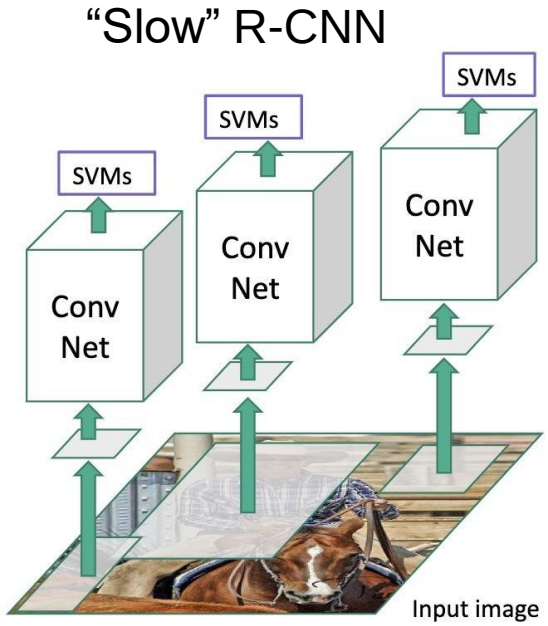
1. Slow at test-time: need to run a full forward pass of CNN for each region proposal

2. SVMs and regressors are post-hoc: CNN features not updated in response to SVMs and regressors

3. Complex multistage training pipeline

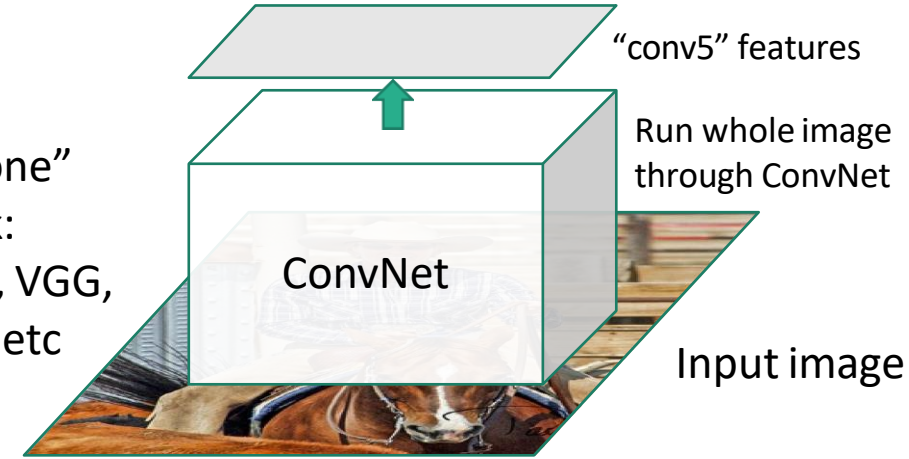
Fast R-CNN

Input image

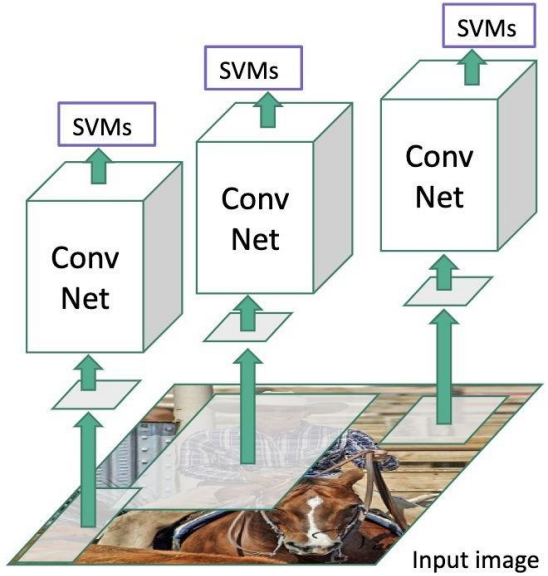


Fast R-CNN

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



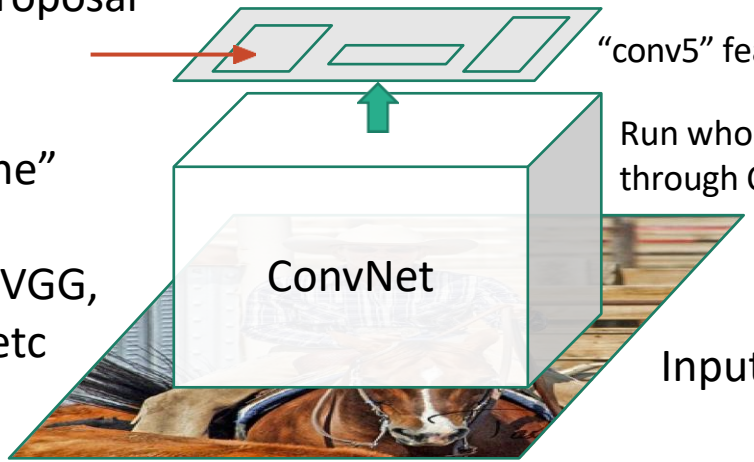
“Slow” R-CNN



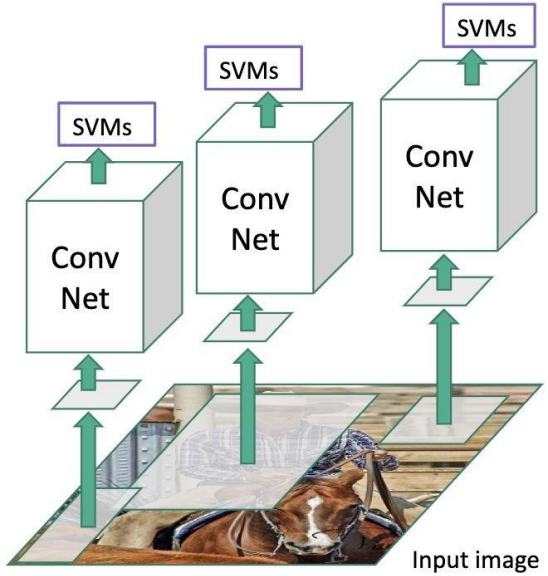
Fast R-CNN

Regions of Interest (RoIs) from a proposal method

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



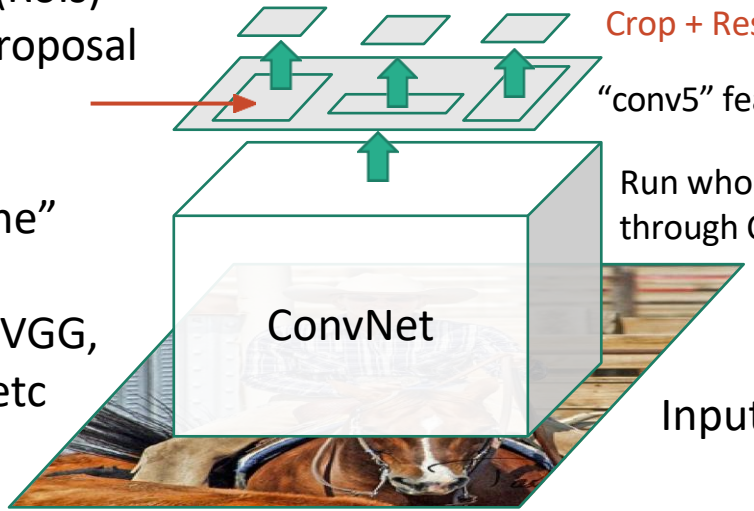
"Slow" R-CNN



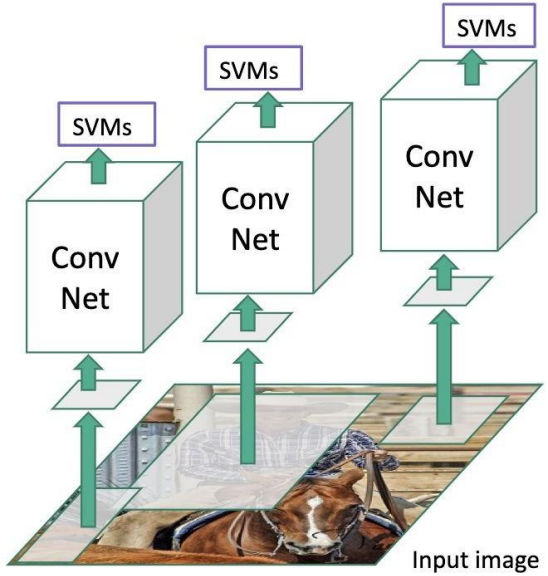
Fast R-CNN

Regions of Interest (RoIs) from a proposal method

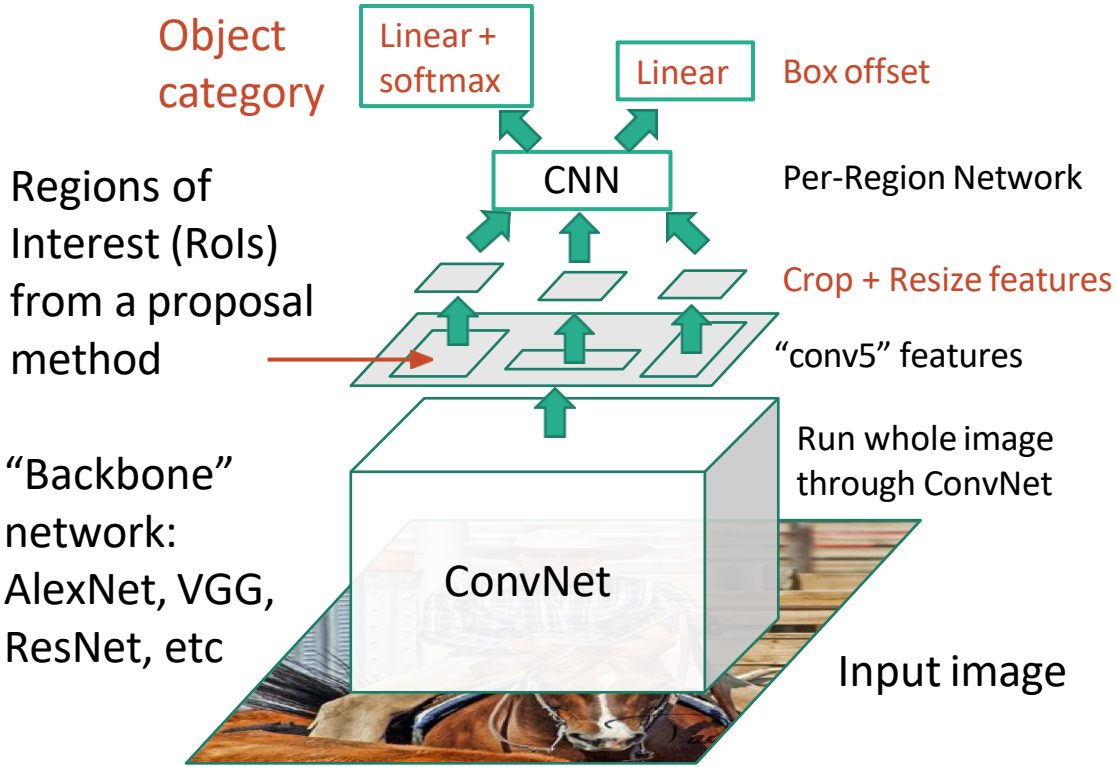
"Backbone" network: AlexNet, VGG, ResNet, etc

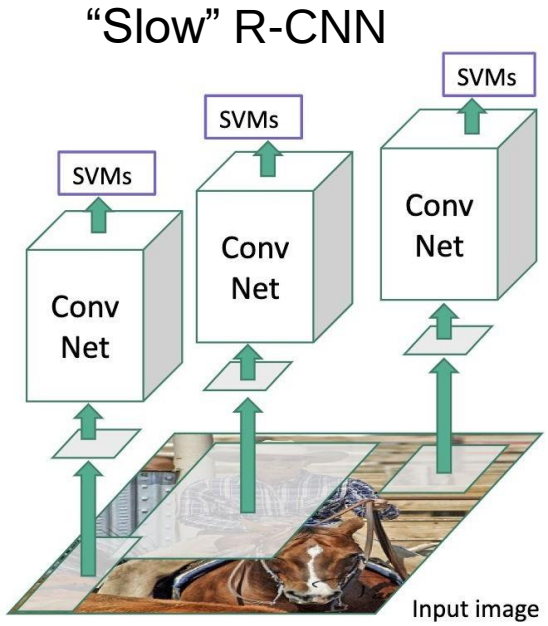


"Slow" R-CNN

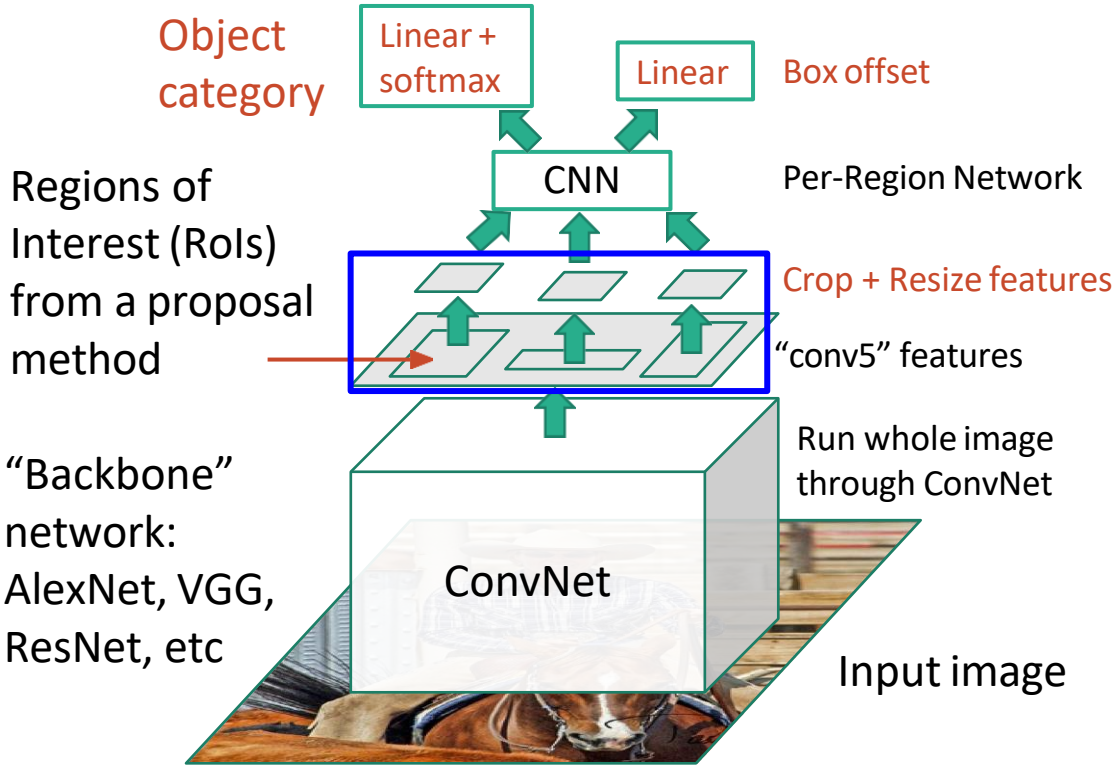


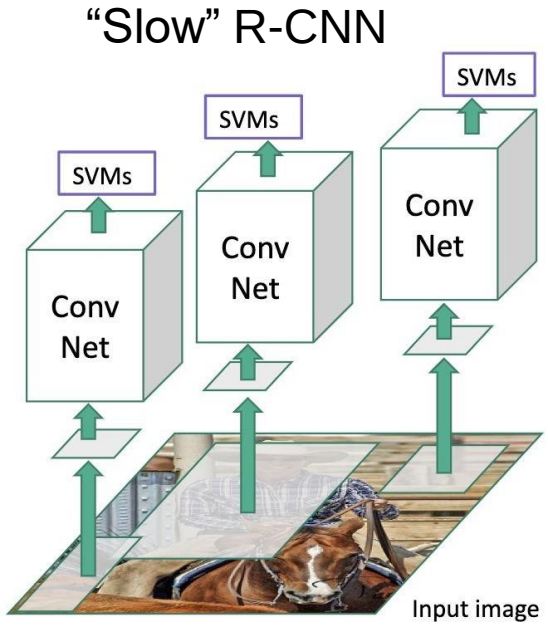
Fast R-CNN

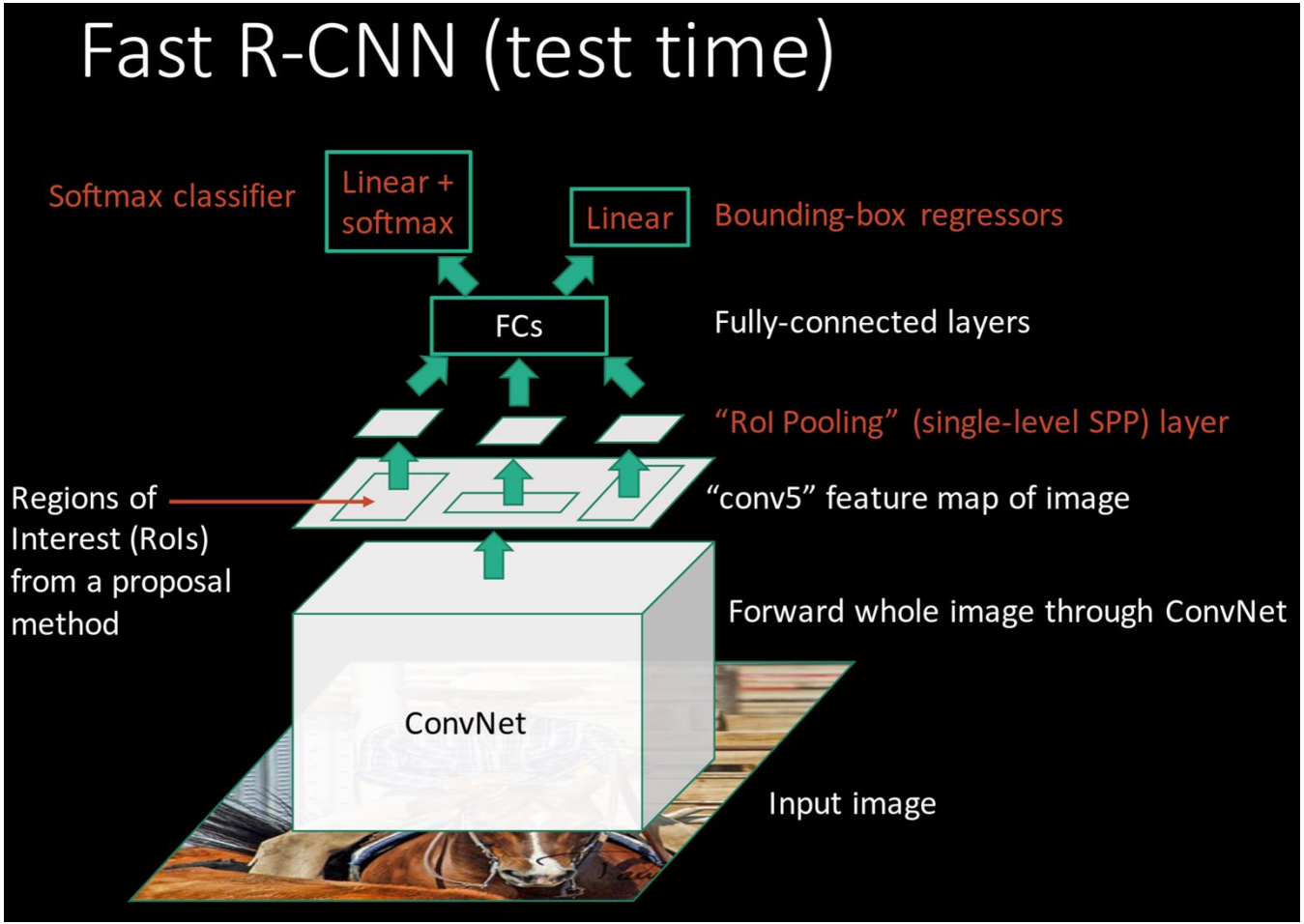

Object category
Linear + softmax
Linear
Box offset
CNN
Per-Region Network
Regions of Interest (RoIs) from a proposal method
Crop + Resize features
"conv5" features
Run whole image through ConvNet
"Backbone" network: AlexNet, VGG, ResNet, etc
ConvNet
Input image


"Slow" R-CNN
SVMs
SVMs
SVMs
Conv Net
Conv Net
Conv Net
Input image

Fast R-CNN

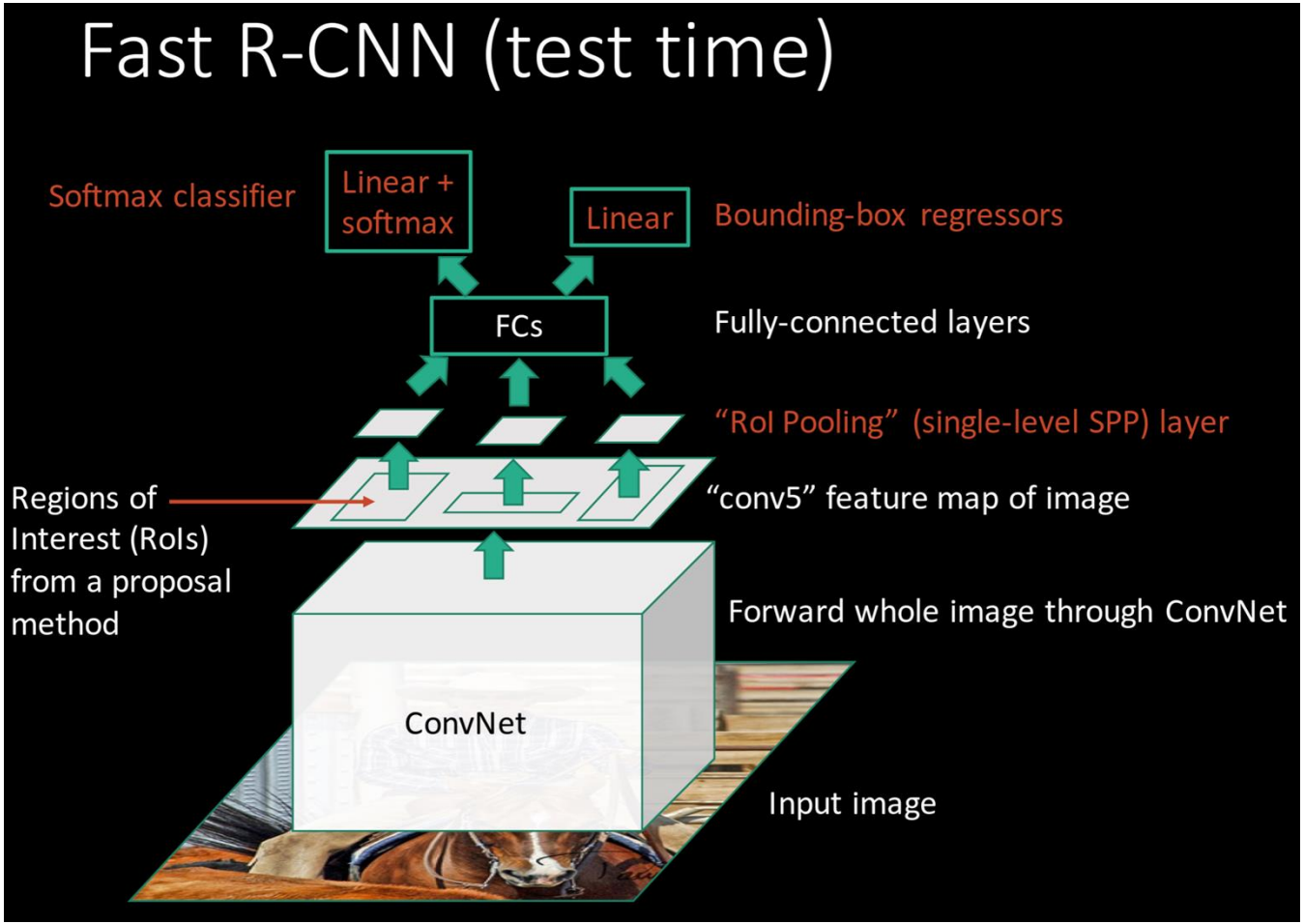

Object category
Linear + softmax
Linear
Box offset
CNN
Per-Region Network
Regions of Interest (RoIs) from a proposal method
Crop + Resize features
"conv5" features
Run whole image through ConvNet
"Backbone" network: AlexNet, VGG, ResNet, etc
ConvNet
Input image


"Slow" R-CNN
SVMs
SVMs
SVMs
Conv Net
Conv Net
Conv Net
Input image



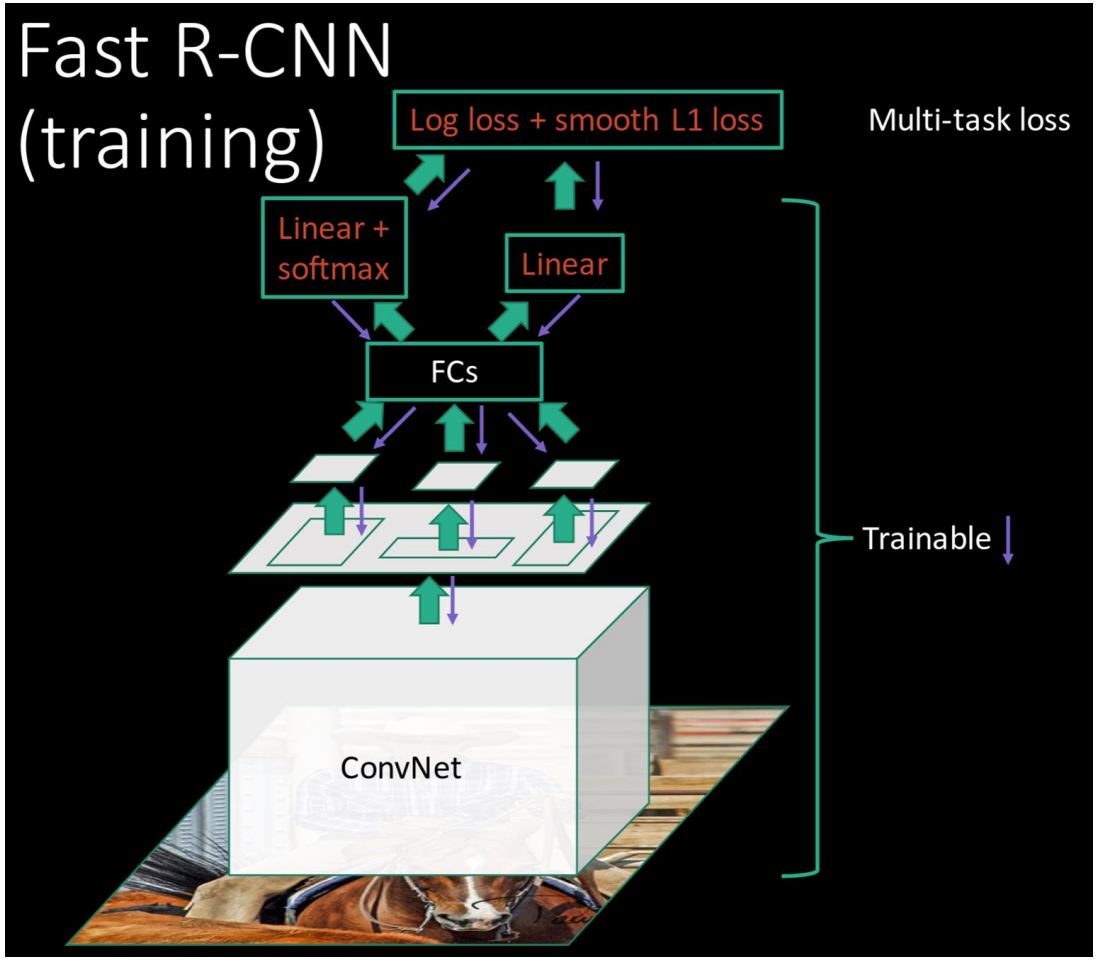
Girschick, "Fast R-CNN", ICCV 2015

Slide credit: Ross Girschick



R-CNN Problem #1:
Slow at test-time due to independent forward passes of the CNN

Solution:
Share computation of convolutional layers between proposals for an image



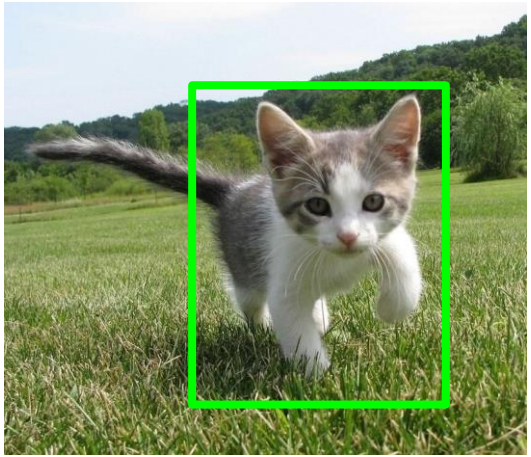
R-CNN Problem #2:
Post-hoc training: CNN not updated in response to final classifiers and regressors

R-CNN Problem #3:
Complex training pipeline

Solution:
Just train the whole system end-to-end all at once!

Slide credit: Ross Girschick

Cropping Features: RoI Pool



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

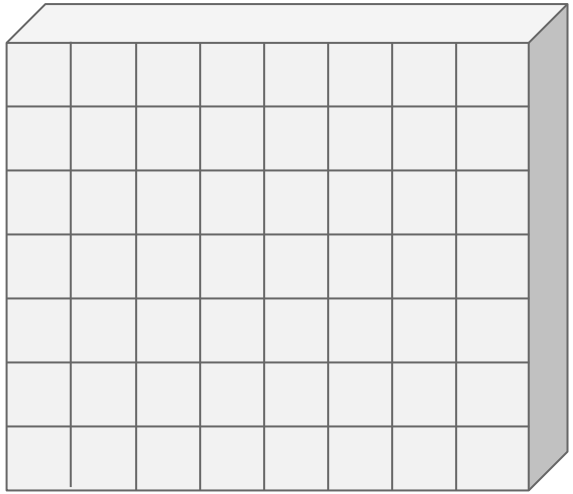
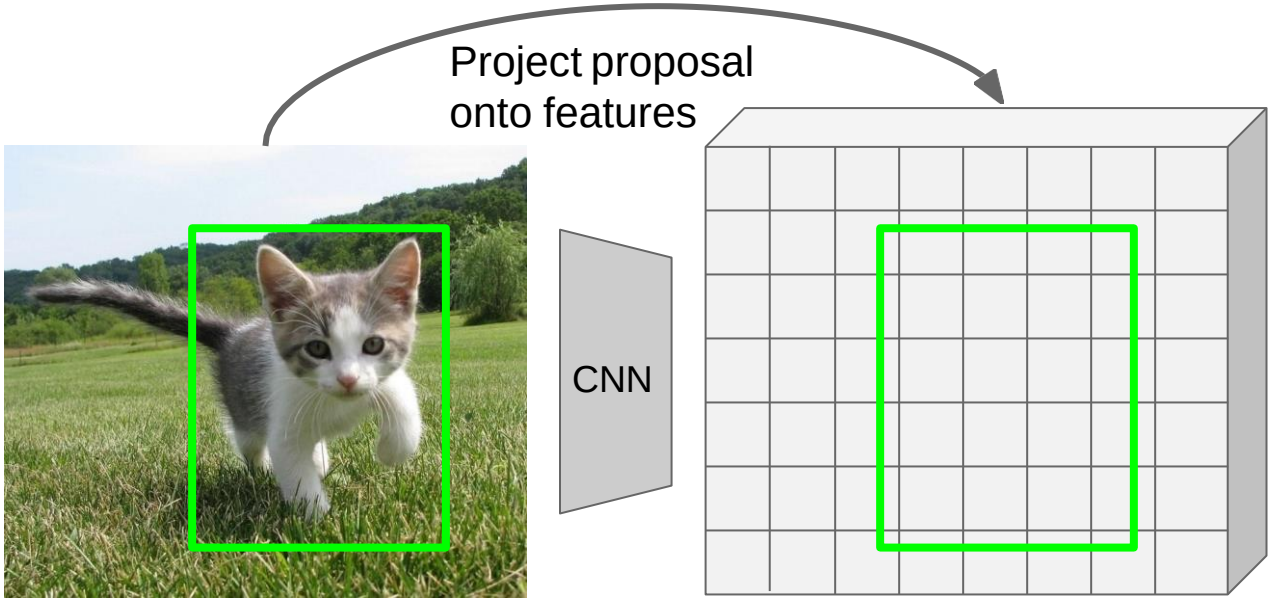


Image features: C x H x W
(e.g. 512 x 20 x 15)

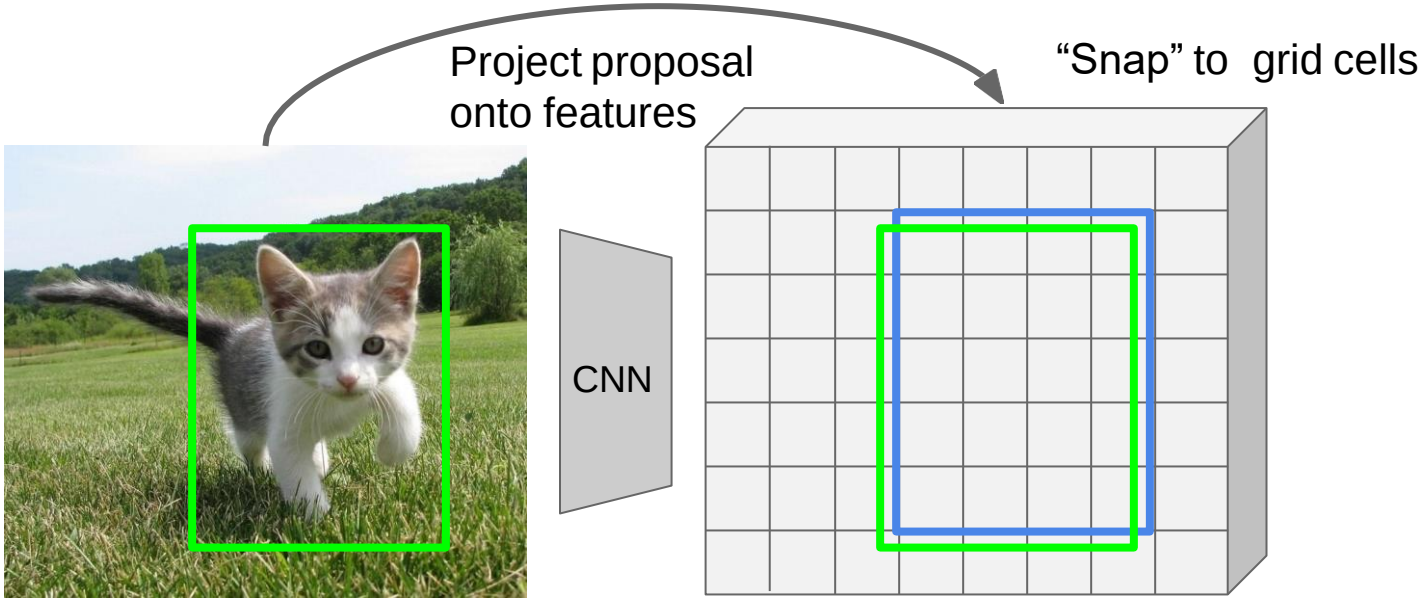
Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool



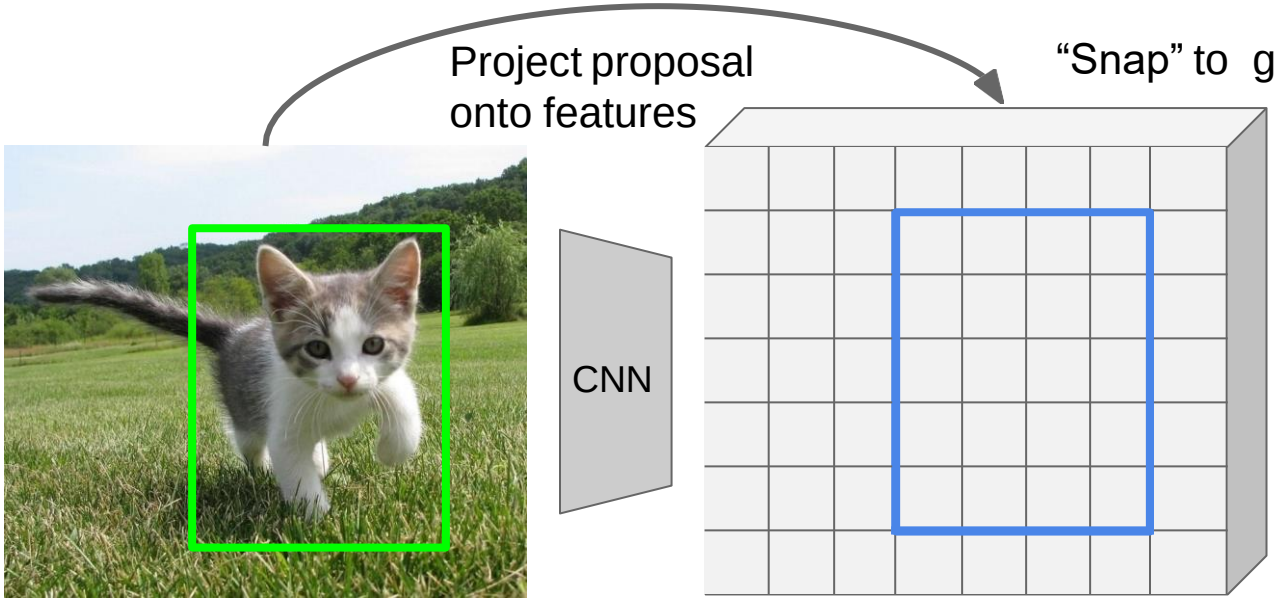
Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool



Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool



Q: how do we resize the 512 x 20 x 15 region to, e.g., a 512 x 2 x 2 tensor?

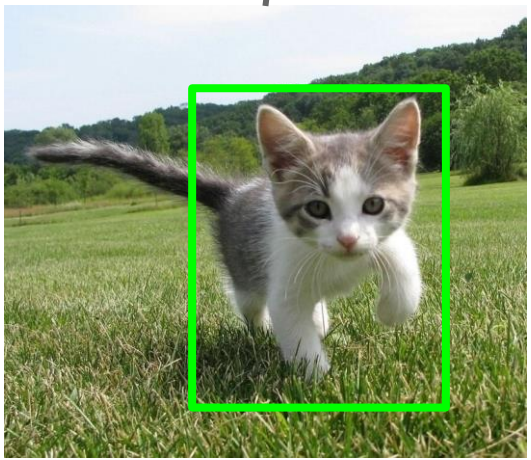
Girshick, “Fast R-CNN”, ICCV 2015.

Cropping Features: RoI Pool

Project proposal onto features

"Snap" to grid cells

Divide into 2x2 grid of (roughly) equal subregions



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

CNN

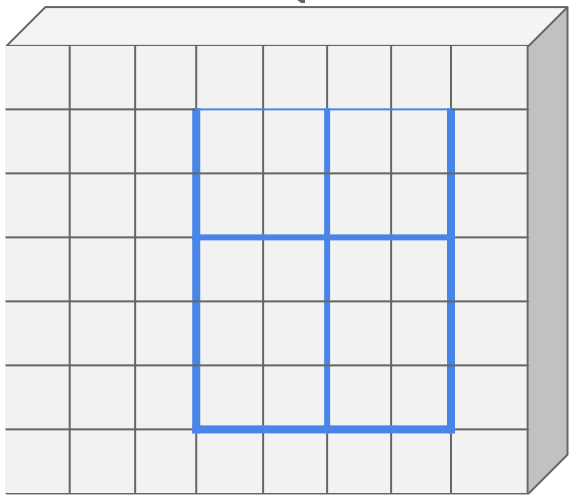
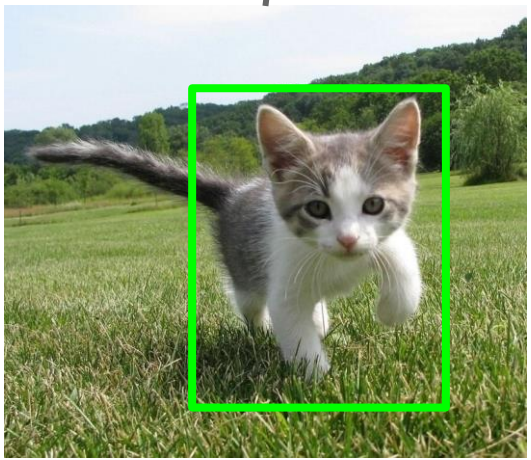


Image features: C x H x W
(e.g. 512 x 20 x 15)

Q: how do we resize the 512 x 20 x 15 region to, e.g., a 512 x 2 x 2 tensor?

Girshick, "Fast R-CNN", ICCV 2015.

Cropping Features: RoI Pool



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

Project proposal onto features

CNN

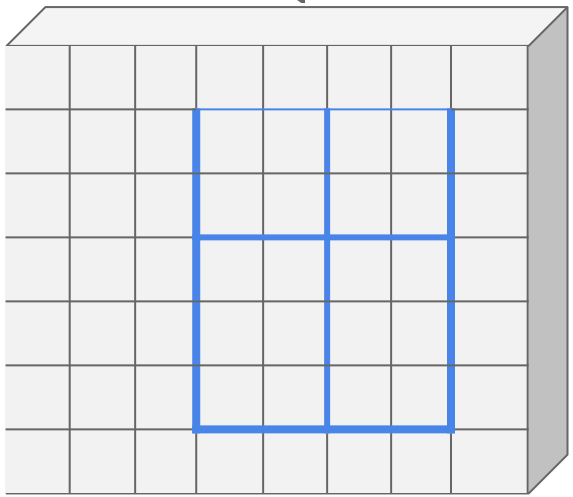
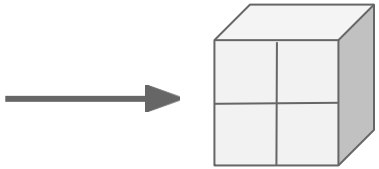


Image features: C x H x W
(e.g. 512 x 20 x 15)

"Snap" to grid cells

Divide into 2x2 grid of (roughly) equal subregions

Max-pool within each subregion



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

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

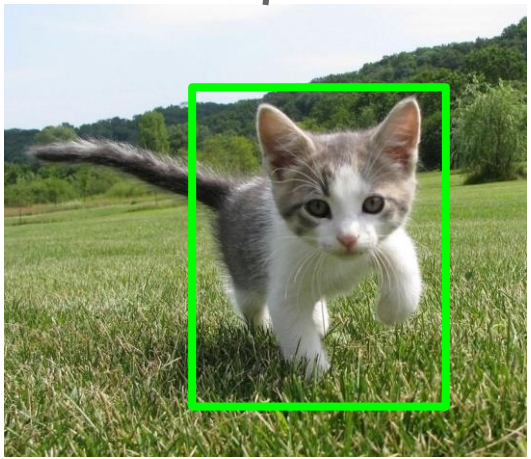
Girshick, "Fast R-CNN", ICCV 2015.

Cropping Features: RoI Pool

Project proposal onto features

"Snap" to grid cells

Divide into 2x2 grid of (roughly) equal subregions



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

CNN

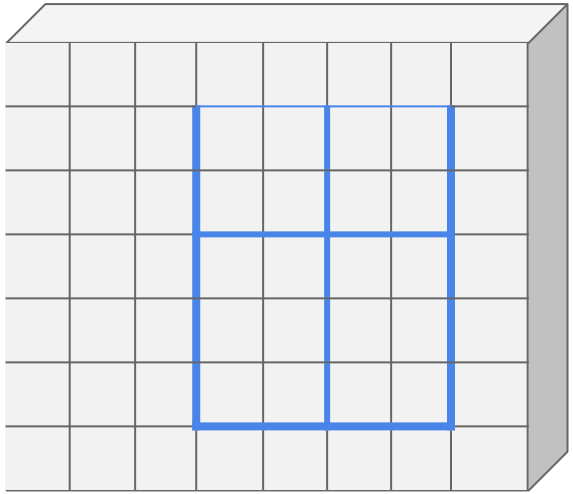
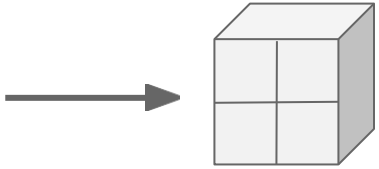


Image features: C x H x W
(e.g. 512 x 20 x 15)

Max-pool within each subregion



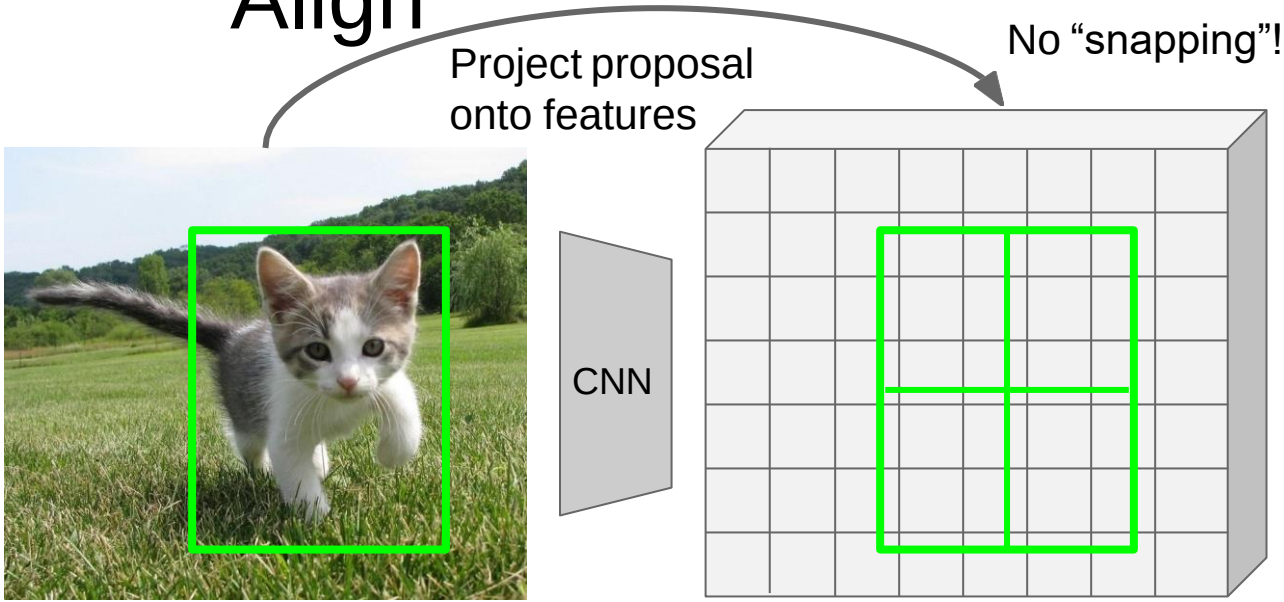
Region features
(here 512 x 2 x 2;
In practice e.g 512 x 7 x 7)

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

Problem: Region features slightly misaligned

Girshick, "Fast R-CNN", ICCV 2015.

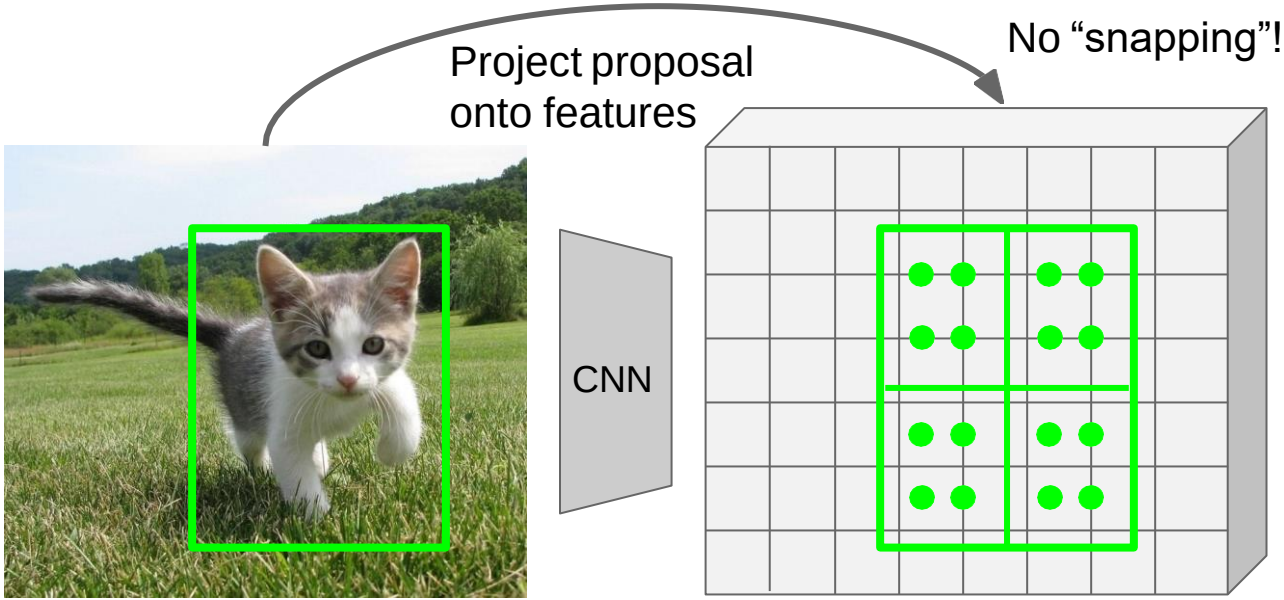
Cropping Features: RoI Align



He et al, “Mask R-CNN”, ICCV 2017

Cropping Features: RoI Align

Sample at regular points in each subregion using bilinear interpolation



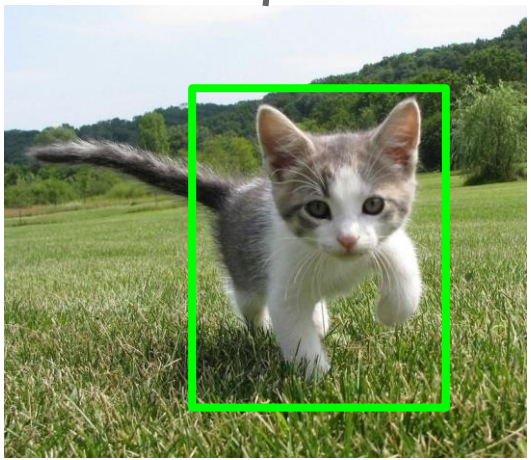
He et al, “Mask R-CNN”, ICCV 2017

Cropping Features: RoI Align

Project proposal onto features

No “snapping”!

Sample at regular points in each subregion using bilinear interpolation



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

CNN

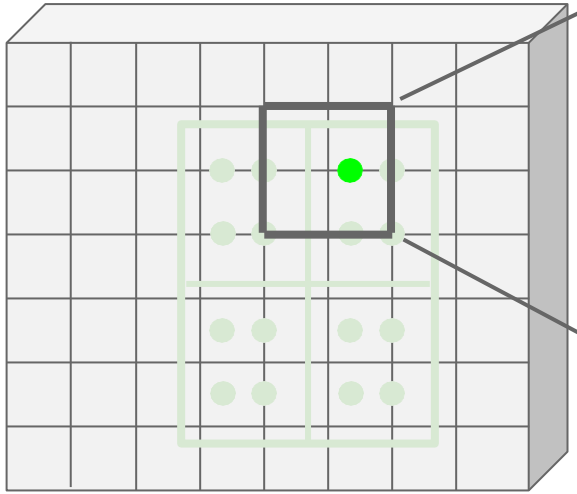
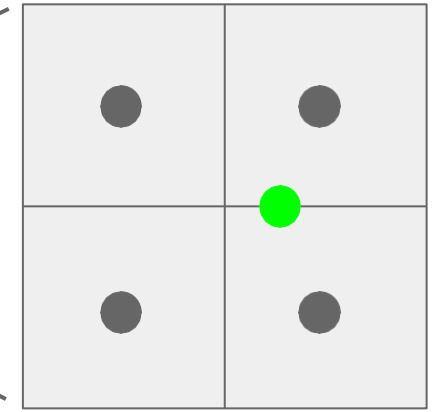


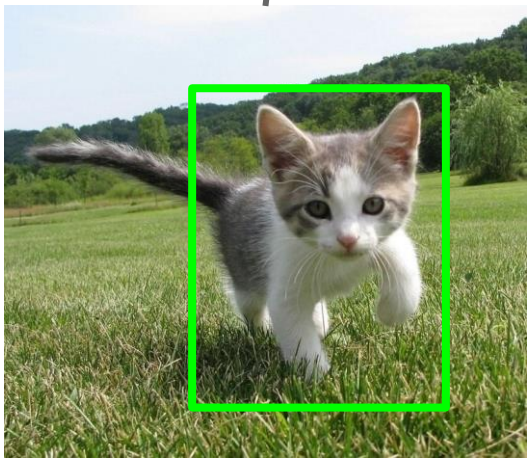
Image features: C x H x W
(e.g. 512 x 20 x 15)



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

He et al, “Mask R-CNN”, ICCV 2017

Cropping Features: RoI Align



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

Project proposal onto features

No “snapping”!

CNN

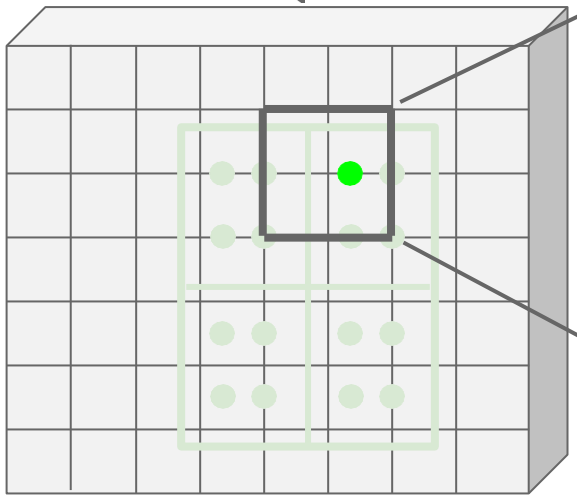
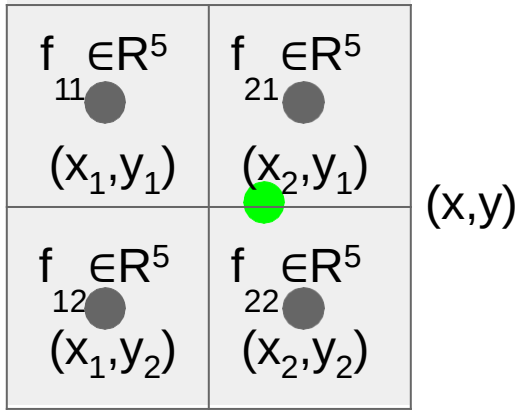


Image features: C x H x W
(e.g. 512 x 20 x 15)

Sample at regular points in each subregion using bilinear interpolation



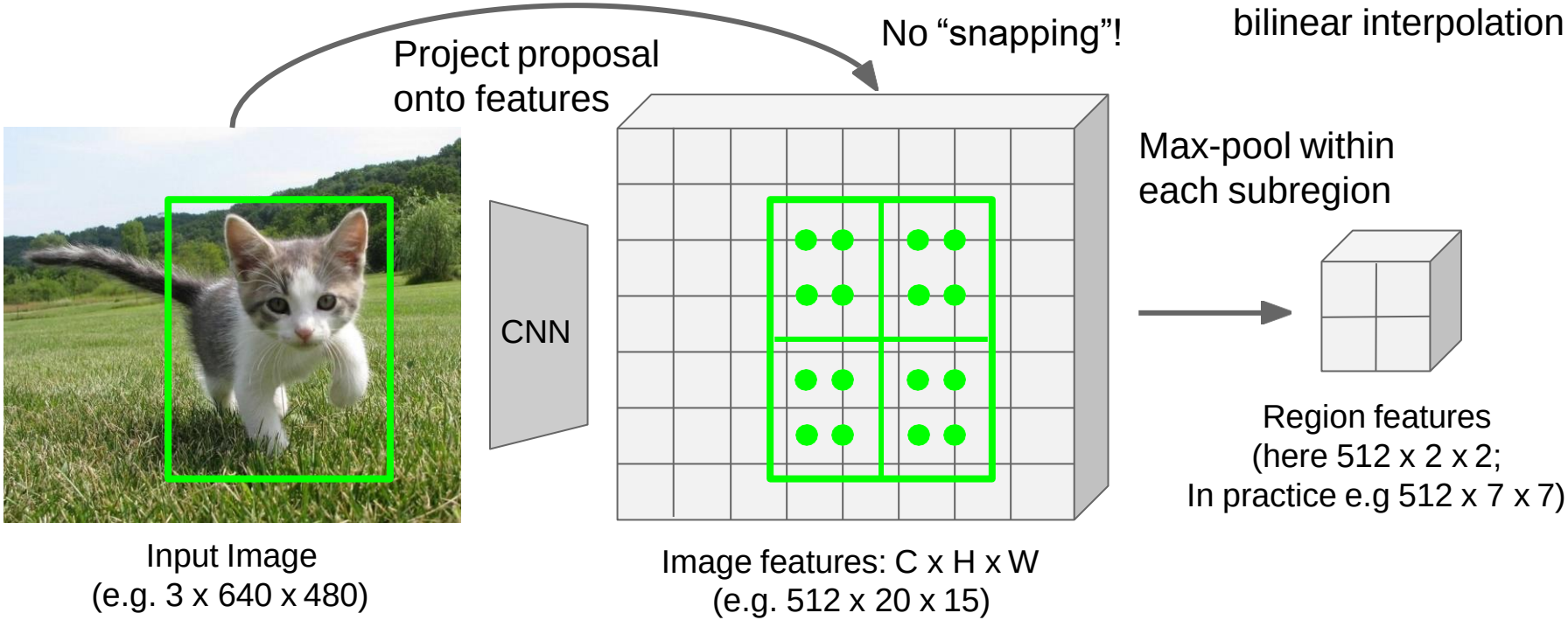
Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

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

He et al, “Mask R-CNN”, ICCV 2017

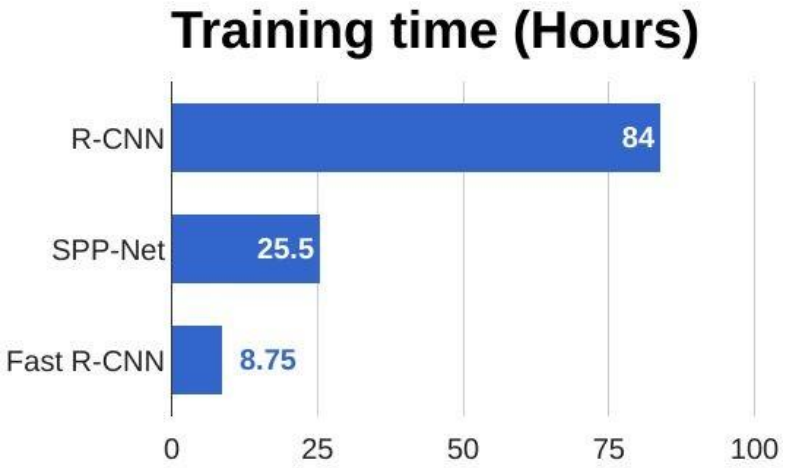
Cropping Features: RoI Align

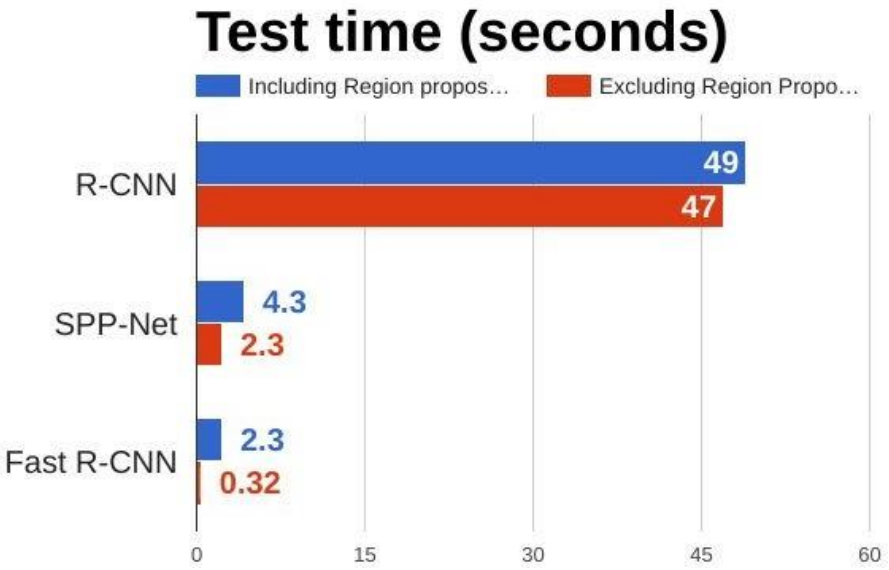
Sample at regular points in each subregion using bilinear interpolation



He et al, “Mask R-CNN”, ICCV 2017

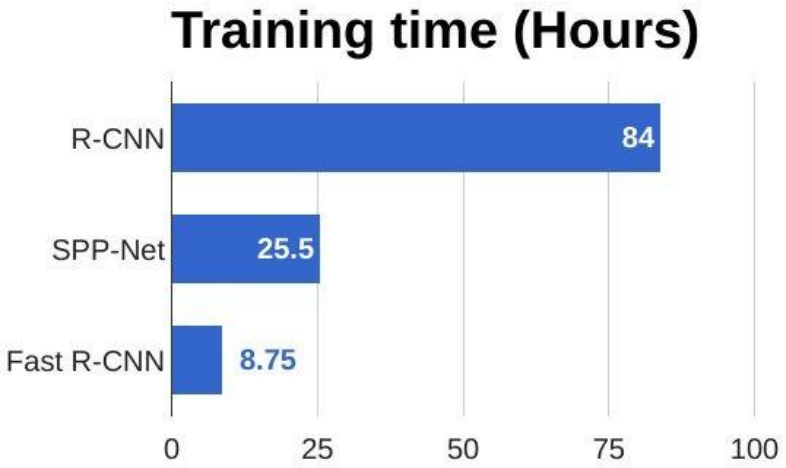
R-CNN vs Fast R-CNN

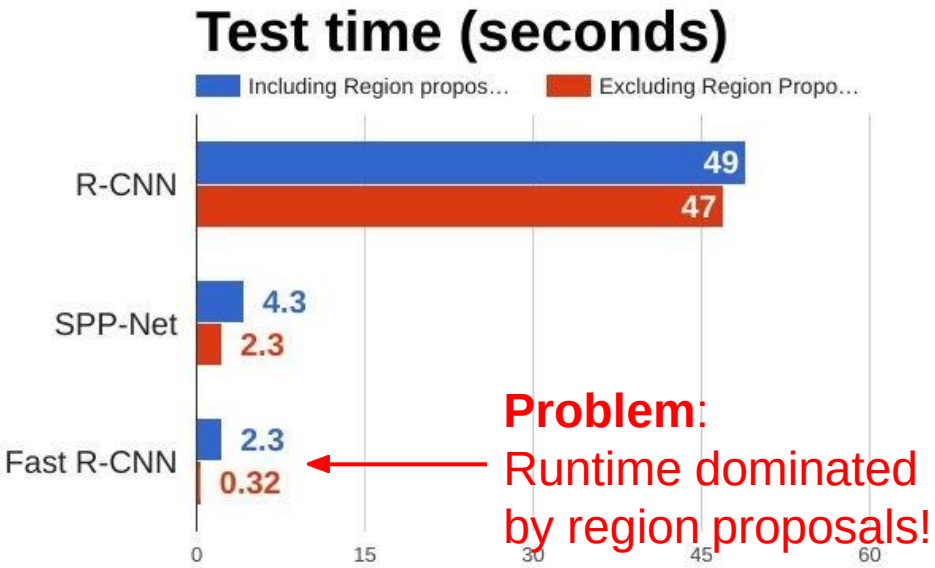




Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
He et al, “Spatial pyramid pooling in deep convolutional networks for visual recognition”, ECCV 2014
Girshick, “Fast R-CNN”, ICCV 2015

R-CNN vs Fast R-CNN





Problem:
Runtime dominated by region proposals!

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