Lecture 6:

Classification & Localization

boris. ginzburg@intel.com

Agenda

- ILSVRC 2014
- Overfeat: integrated classification, localization, and detection
  - Classification with Localization
  - Detection.

ILSVRC-2014

http://www.image-net.org/challenges/LSVRC/2014/

Classification & Localization:

- Assign to each image label. 5 guesses.
- A bounding box of the main object must be returned and must match with the ground truth by 50% (using the PASCAL criterion of union over intersection). Each returned bounding box must be labeled with the correct class. similar to classification, 5 guesses are allowed per image

Detection:

- there can be any number of object in each image (including zero). False positives are penalized

ILSVRC-2014

Task 1: Detection

For each image, algorithms will produce a set of annotations (c_i, b_i, s_i) of class labels c_i, bounding boxes b_i and confidence scores s_i. This set is expected to contain each instance of each of the 200 object categories. Objects which were not annotated will be penalized, as will be duplicate detections (two annotations for the same object instance). The winner of the detection challenge will be the team which achieves first place accuracy on the most object categories.

Task 2: Classification and localization

In this task, given an image an algorithm will produce 5 class labels $c_i, i = 1, \dots 5$ in decreasing order of confidence and 5 bounding boxes $b_i, i = 1, \dots 5$, one for each class label. The quality of a labeling will be evaluated based on the label that best matches the ground truth label for the image. The idea is to allow an algorithm to identify multiple objects in an image and not be penalized if one of the objects identified was in fact present, but not included in the ground truth.

The ground truth labels for the image are $C_k, k = 1, \dots n$ with n class labels. For each ground truth class label C_k, the ground truth bounding boxes are $B_{km}, m = 1 \dots M_k$, where M_k is the number of instances of the k^{th} object in the current image.

Let $d(c_i, C_k) = 0$ if $c_i = C_k$ and 1 otherwise. Let $f(b_i, B_k) = 0$ if b_i and B_k have more than 50% overlap, and 1 otherwise. The error of the algorithm on an individual image will be computed using two metrics:

- Classification-only:

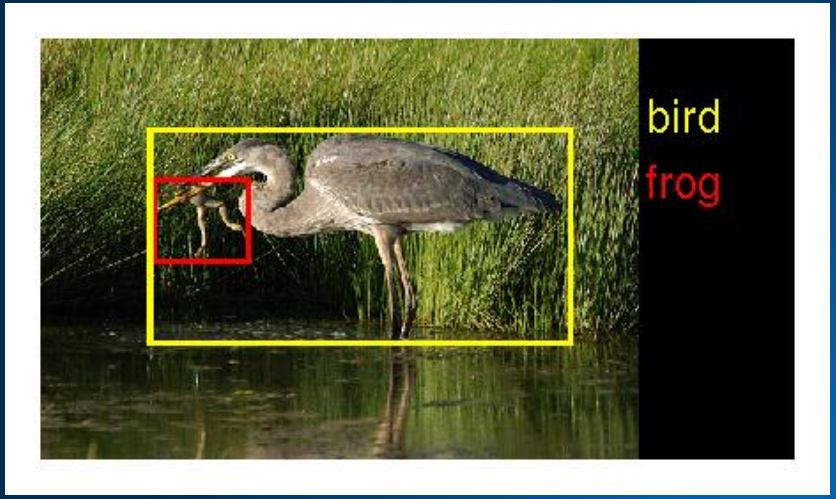
$$e = \frac{1}{n} \cdot \sum_k \min_i d(c_i, C_k)$$

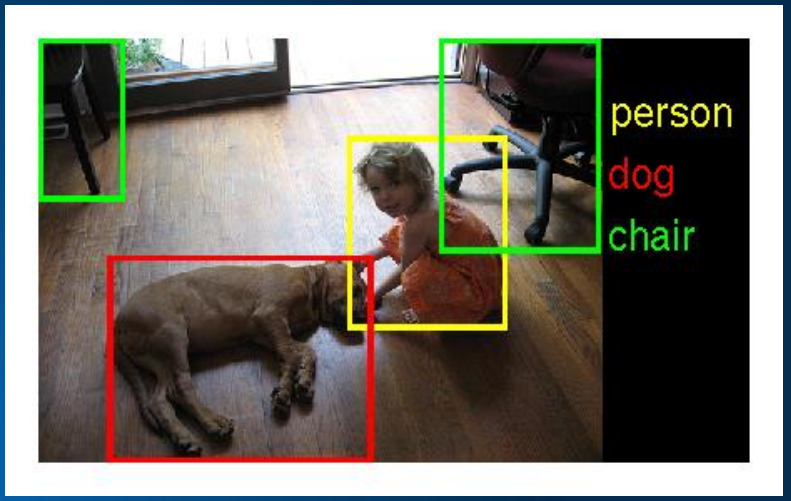
- Classification-with-localization:

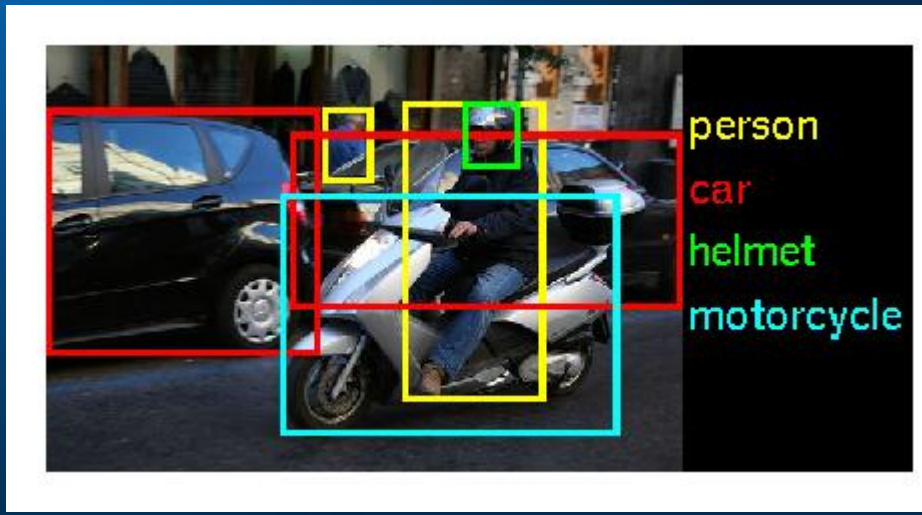
$$e = \frac{1}{n} \cdot \sum_k min_i min_m max\{d(c_i, C_k), f(b_i, B_{km})\}$$

The classification-only and classification-with-localization errors of the algorithm is the average corresponding error across all test images.

Detection: Examples

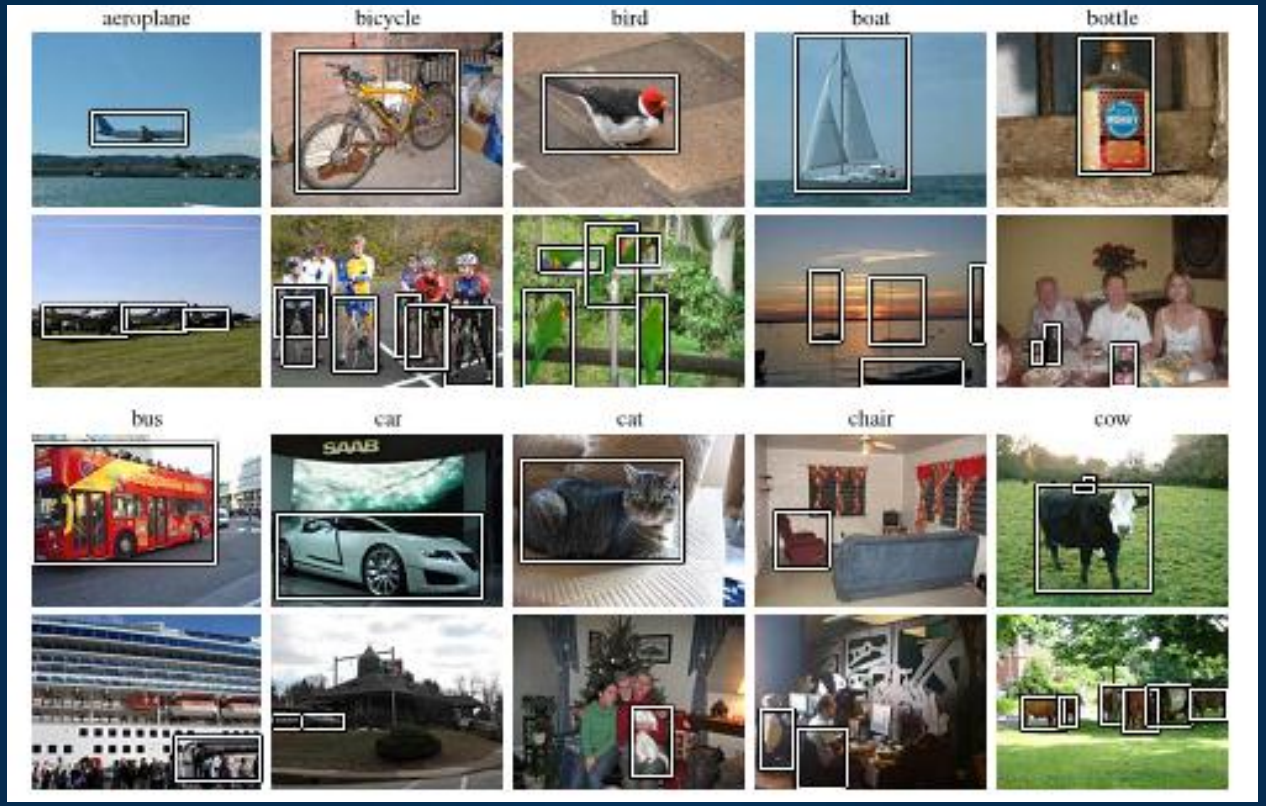






Detection: PASCAL VOC

- http://pascallin.ecs.soton.ac.uk/challenges/VOC/
- 20 classes:



Detection: ILSVRC 2014

- http://image-net.org/challenges/LSVRC/2014/

		PASCAL 2012	ILSVRC 2013	ILSVRC 2014
	# classes	20	200	200
Training	# images	5717	395909	**456567**
	# objects	13609	345854	**478807**
Validation	# images	5823	20121	20121
	# objects	13841	55502	55502
testing	# images	10991	40152	40152
	# objects			

Detection paradigms

1. CNN + Regression
2. Overfeat
3. Regions with CNN

OVERFEAT

Overfeat: Integrated classification, localization & detection

http://cilvr.nyu.edu/doku.php?id=software:overfeat:start

Training a convolutional network to simultaneously classify, locate and detect objects in images boost the classification accuracy and the detection and localization accuracy.

3 ideas:

1. apply a ConvNet at multiple locations in the image, in a sliding window fashion, and over multiple scales.
2. train the system to not only produce a distribution over categories for each window, but also to produce a prediction of the location and size of the bounding box containing the object relative to that of the viewing window
3. accumulate the evidence for each categories at each location and size.

Overfeat: "accurate" net topology

input 3x221x221

1. convo: 7×7 stride 2×2; ReLU; maxpool: 3×3 stride 3×3; output: 96x36x36
2. convo: 7×7 stride 1×1; ReLU; maxpool: 2×2 stride 2×2; output: 256x15x15
3. convo: 3×3 stride 1×1 0-padded; ReLU; output: 512x15x15
4. convo: 3×3 stride 1×1 0-padded; ReLU; output: 512x15x15
5. convo: 3×3 stride 1×1 0-padded; ReLU; output: 1024x15x15
6. convo: 3×3 stride 1×1 0-padded; ReLU; maxpool: 3×3 stride 3×3; output: 1024x5x5
7. convo: 5×5 stride 1×1; ReLU; output: 4096x1x1
8. full; ReLU; output: 4096x1x1
9. full; output: 1000x1x1
10. softmax; output: 1000x1x1

Feature Extraction: 3 x [231x231] → 1024 x [5x5], with total down-sampling is (2x3x2x3):1=36:1

Overfeat: topology summary

Layers 1-5 are similar to Alexnet: conv. layer with ReLU, and max pooling, but with the following differences:

1. no contrast normalization
2. pooling regions are non-overlapping
3. Smaller stride to improve accuracy

Layer	1	2	3	4	5	6	7	Output 8
Stage	conv + max	conv + max	conv	conv	conv + max	full	full	full
# channels	96	256	384	384	256	4096	4096	1000
Filter size	7x7	7x7	3x3	3x3	3x3	-	-	-
Conv. stride	2x2	1x1	1x1	1x1	1x1	-	-	-
Pooling size	3x3	2x2	-	-	3x3	-	-	-
Pooling stride	3x3	2x2	-	-	3x3	-	-	-
Zero-Padding size	-	-	1x1x1x1	1x1x1x1	1x1x1x1	-	-	-
Spatial input size	221x221	36x36	15x15	15x15	15x15	5x5	1x1	1x1

Overfeat: classification

Let's takes image, and apply sliding window [231x231], For each window we will take best score. Feature extractor has sub-smapling 36:1. If we slide window with step 36, then output feature will slide with step 1

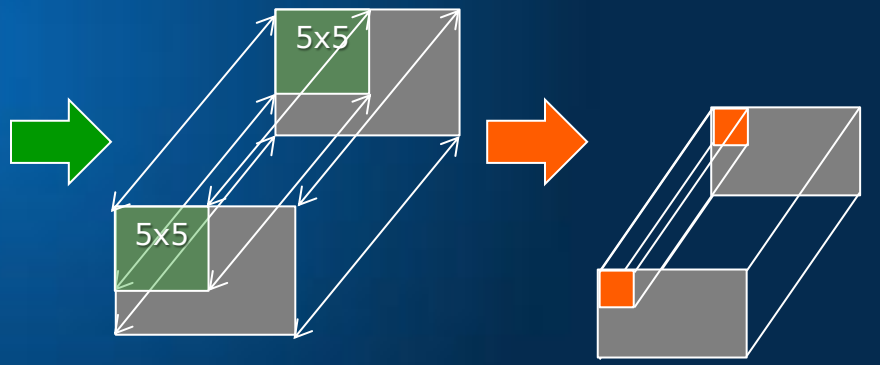


Image: 340x270 Features: 8x6 Best score: 4x2

Overfeat: classification

2 adjacent windows share many computations. Let's do all windows in parallel.

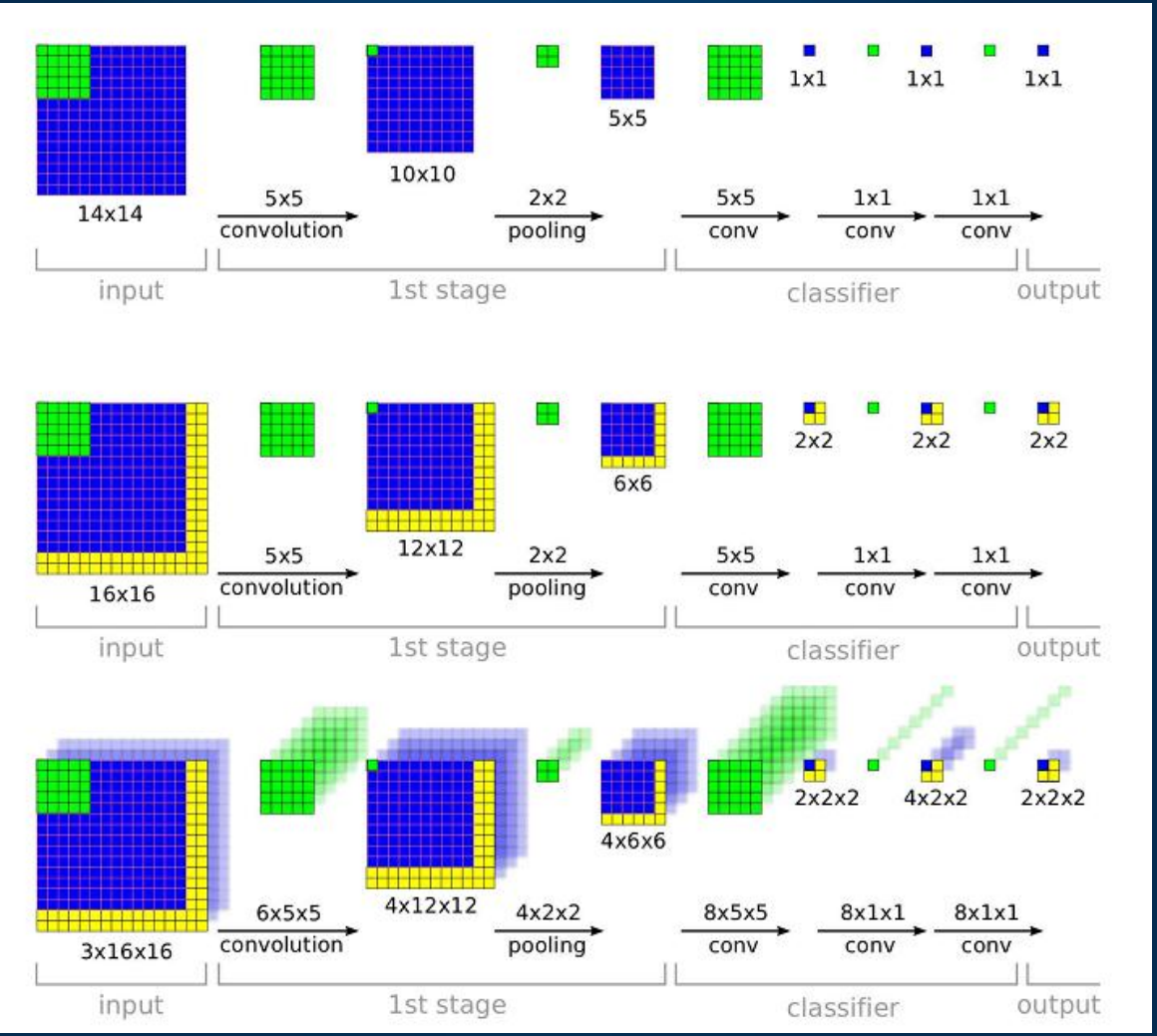
Feature extraction:

- The filters are convolved across the entire image in one pass. This far more efficient than sliding a fixed-size feature extractor over the image and then aggregating the results from different locations.

Classifier :

- Two last fully connected layers can be done in parallel too, but we should take care of right offsets.

Overfeat: classification



Overfeat: classification

Feature Extraction:

we compute first 5 layers for whole image. First 5 layers before pooling correspond to 12:1 “subsampling” .

Classifier:

The classifier has a fixed-size 5x5 input and is exhaustively applied to the layer 5 maps. We will shift the classifier’s viewing window by 1 pixel through pooling layers without subsampling.

In the end we have [MxN] x C scores, where M, N are sliding windows index, and C – number of classes.

Quiz: How to choose 5 best options?

Input	Layer 5 Before pooling	Layer 5 After pool 3x3	Classifier map
245x245	17x17	[3x3] x [5x5]	[3x3] x C
281x 317	20x23	[6x9] x [5x5]	[6x9] x C

Overfeat: scaling and data augmentation

To locate objects in different sizes we can rescale image to 6 scales:

- The typical ratio from one scale to another is about ~1.4 (this number differs for each scale since dimensions are adjusted to
- fit exactly the stride of our network)

Data augmentation: horizontal flipping.

Final post-processing:

- For each class we took local spatial max for resulting windows,
- take top-1/ top-5 .

Overfeat: boosting

Boosting: train 7 different models with different init weights, and select the best result

Approach	Top-1 error %	Top-5 error %
Krizhevsky *et al.* [15]	40.7	18.2
OverFeat - 1 fast model, scale 1, coarse stride	39.28	17.12
OverFeat - 1 fast model, scale 1, fine stride	39.01	16.97
OverFeat - 1 fast model, 4 scales (1,2,4,6), fine stride	38.57	16.39
OverFeat - 1 fast model, 6 scales (1-6), fine stride	38.12	16.27
OverFeat - 1 big model, 4 scales, fine stride	35.74	14.18
OverFeat - 7 fast models, 4 scales, fine stride	35.10	13.86
OverFeat - 7 big models, 4 scales, fine stride	33.96	13.24

Overfeat: "fast" net topology

Input 3x231x231

1. convo: 11×11 stride 4×4; ReLU; maxpool: 2×2 stride 2×2; output: 96x24x24
2. convo: 5×5 stride 1×1; ReLU; maxpool: 2×2 stride 2×2; output: 256x12x12
3. convo: 3×3 stride 1×1 0-padded; ReLU; output: 512x12x12
4. convo: 3×3 stride 1×1 0-padded; ReLU; output: 1024x12x12
5. convo: 3×3 stride 1×1 0-padded; ReLU; maxpool: 2×2 stride 2×2; output: 1024x6x6
6. convo: 6×6 stride 1×1; ReLU; output: 3072x1x1
7. full; ReLU; output : 4096x1x1
8. full; output: 1000x1x1
9. softmax; output: 1000x1x1

Overfeat : training details

1. Data augmentation:
   - Each image is down-sampled so that the smallest dimension is 256 pixels. We then extract 5 random crops (and their horizontal flips) of size 221x221 pixels
2. Weight initialization
   - randomly with $(\mu, \sigma) = (0, 1 \times 10^{-2})$.
3. Training:
   - SGD with learning rate = 5×10^{-2} and is decreased by ½ after (30, 50, 60, 70, 80) epochs, momentum =0.6 , ℓ_2 weight decay =1×10^{-5} ;
   - Dropout in FC layers.

Overfeat: localization

1. Starting from our classification-trained network, fix the feature extraction layers (1-5) and replace the classifier layers by a regression network:
   - Regression net takes as input the pooled feature maps from layer 5. It has 2 fully-connected hidden layers of size 4096 and 1024 channels, respectively. The output layer: has 4 units for each class, which specify the coordinates for the bounding box edges.
2. Train regression net:
   - using an ℓ_2 loss between the predicted and true bounding box for each example.
   - training use the same set of scales as in multi-scale classification.
   - compare the prediction of the regressor at each spatial location with the ground-truth bounding box, shifted into the frame of reference

Overfeat: localization

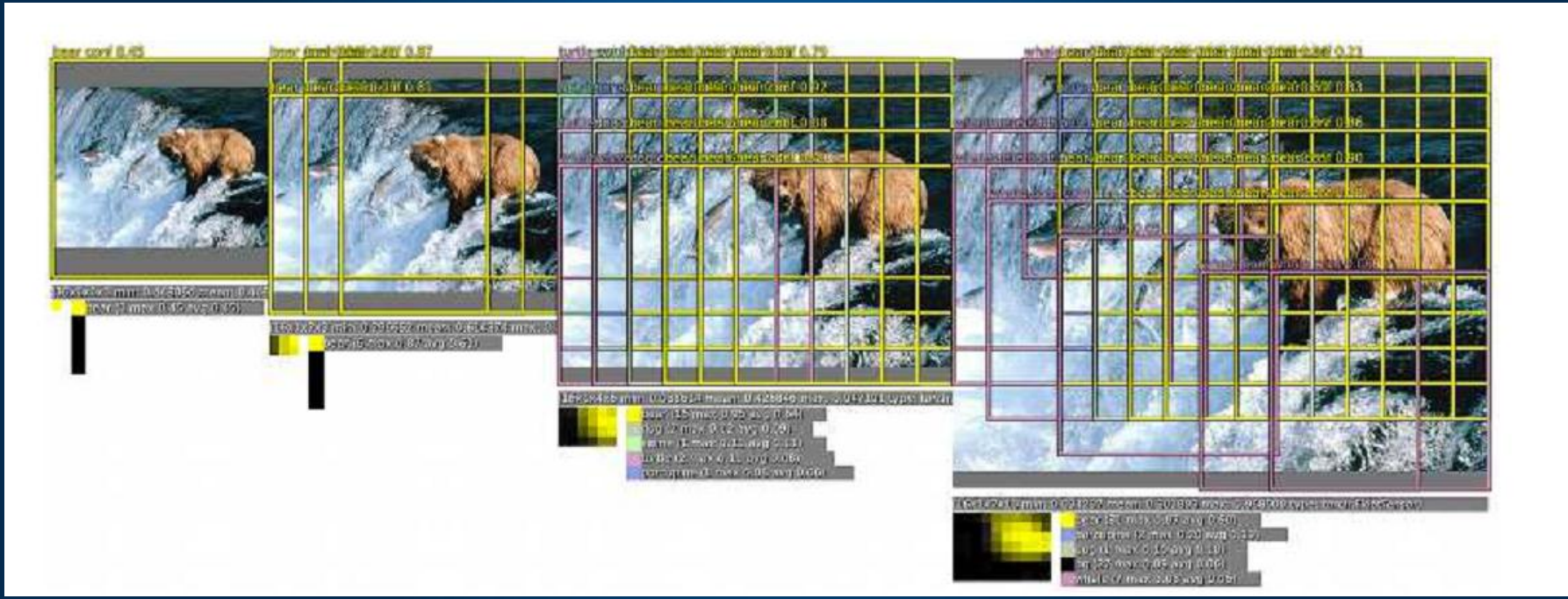
3. Bounding boxes are merged & accumulated
   a) Assign to C_s the set of classes in the top -5 for each scale $s \in 1 \ldots 6$, found by taking the maximum detection class outputs across spatial locations for that scale.
   b) Assign to B_s the set of bounding boxes predicted by the regressor network for each class in C_s, across all spatial locations at scale s.
   c) Assign $B \leftarrow \bigcup_s B_s$
   d) Repeat merging until done:
      a. $(b1, b2) = \text{argmin}_{b1 \neq b2 \in B}$ *match_score (b1, b2)*
      b. If (match_score(b1, b2) > t), then stop;
      c. Otherwise, set $B \leftarrow B \setminus \{b1, b2\} \cup$ box_merge(b1, b2)

Here match_score = the sum of the distance between centers of the two bounding boxes and the intersection area of the boxes.

box merge compute the average of the bounding boxes' coordinates.

Overfeat: localization pipleine

1. The raw classifier/detector outputs a class and a confidence for each location:



Overfeat: localization pipleine

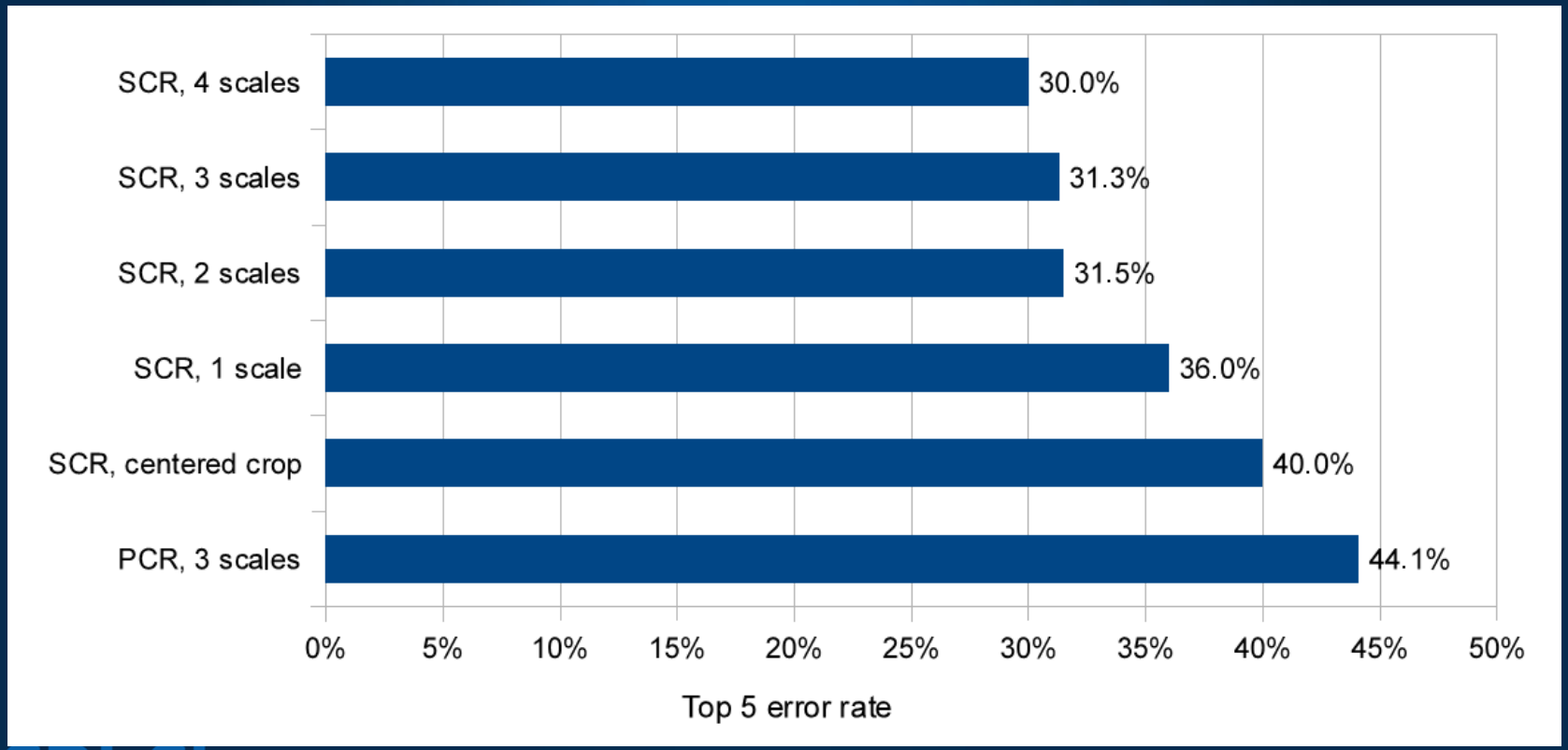
2. The regression then predicts the location scale of the object with respect to each window:

Overfeat: localization pipleine

3. Bounding boxes are merged & accumulated

Single-class Regression vs Per- Class Regression

Using a different top layer for each class in the regressor network for each class (Per-Class Regressor (PCR) surprisingly did not outperform using only a single network shared among all classes (44.1% vs. 31.3%).



Overfeat: Detection

The detection task differ from localization in that there can be any number of object in each image (including zero), and that false positives are penalized by the mean average precision (mAP) measure

The main difference with the localization task, is the necessity to predict a background class when no object is present. Traditionally, negative examples are initially taken at random for training. Then the most offending negative errors are added to the training set in bootstrapping passes.

REGIONS WITH CNN

R-CNN: Regions with CNN features

R. Girshick et al , Berkeley "Rich feature hierarchies..."

http://www.cs.berkeley.edu/~rbg/slides/rcnn-cvpr14-slides.pdf

Source: https://github.com/rbgirshick/rcnn // requires Matlab

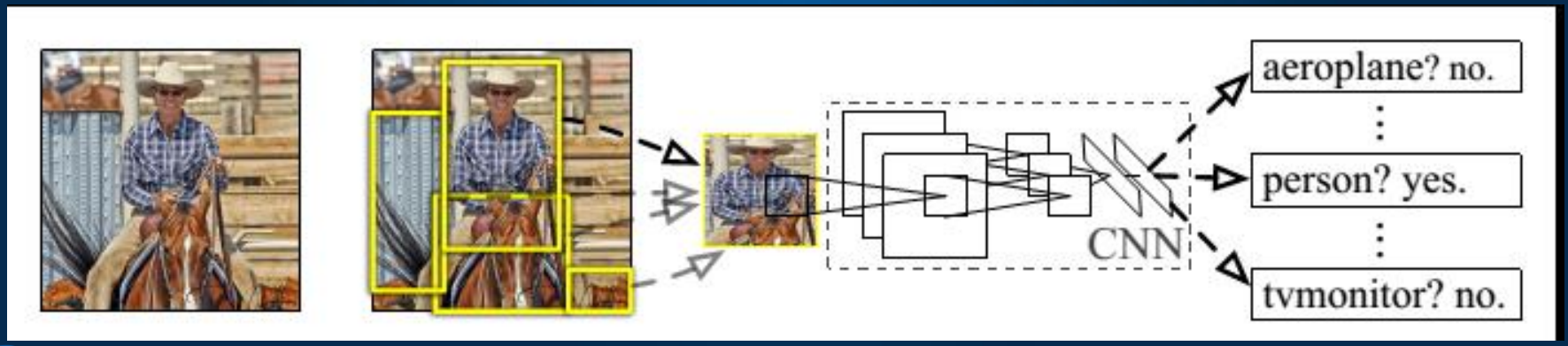
Regions with CNN detection approach:

1. generates ~2000 category-independent regions for the input image,
2. extracts a fixed-length feature vector from each region using a CNN,
3. classifies each region with category-specific linear SVM

R-CNN outperforms OverFeat, with a mAP = 31.4% vs 24.3%.

R-CNN: architecture

1. Region detection → 2000 regions , see http://www.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-124.pdf
2. Region croped and scaled to [227 x 227] → feature extraction with Imagenet: 5 convolutional layers + 2FC → 4096 features
3. SVM for 200 classes
4. Greedy non-maximum suppression for each class: rejects a region if it has an *intersection-over-union (IoU)* overlap with a higher scoring selected region larger than a learned threshold



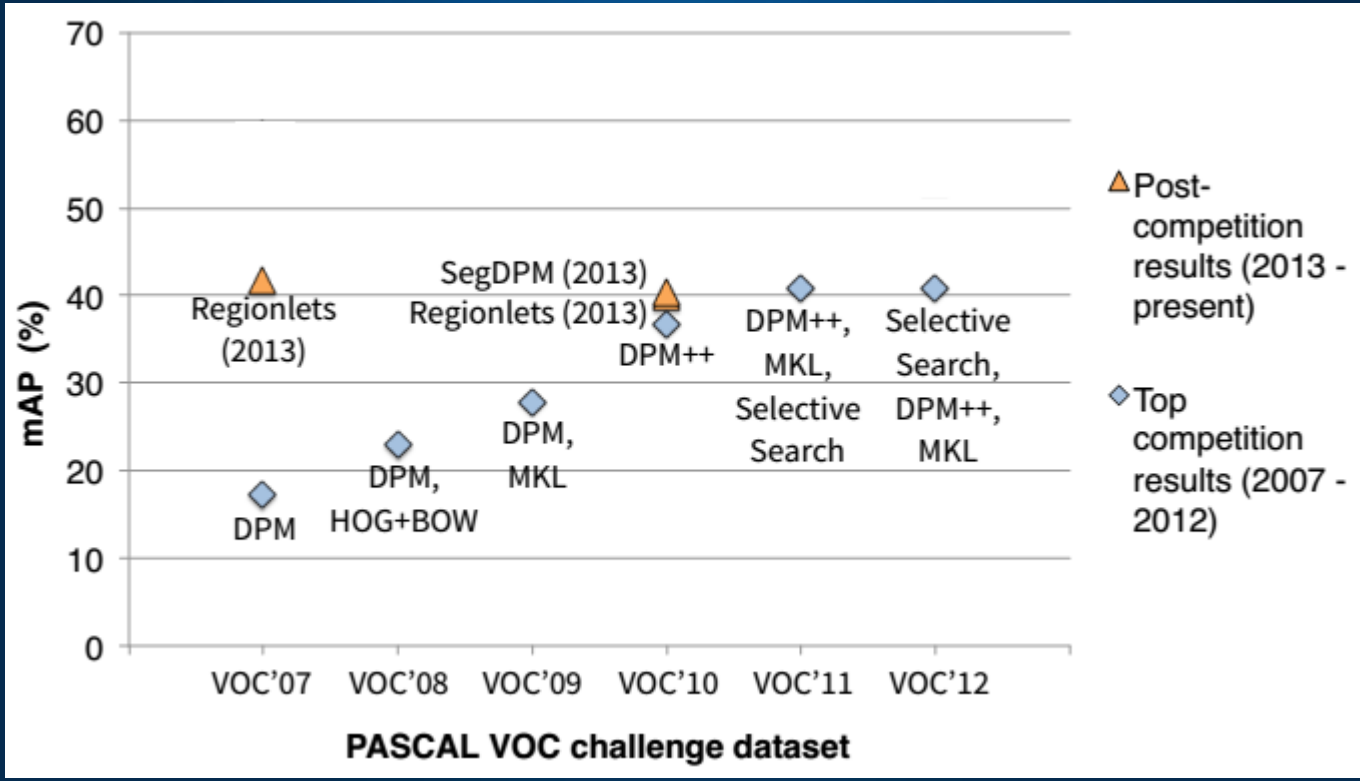
R-CNN Training

The principle idea is to train feature extraction CNN on a large auxiliary dataset (ILSVRC), followed by domain specific fine-tuning on a small dataset (PASCAL):

- Pre-training: Train Imagenet
- Replace last layer with FC layer to N+1 outputs (N classes + 1 "background"; VOC N=20, ILSVRC N=200)
- Training:
  - For each region: if IoU > ½ - positive example, otherwise – negative (background).
  - Batch = 128 = 32 positive + 96 background
  - Init weights random
  - SGD with $\lambda = 0.001$

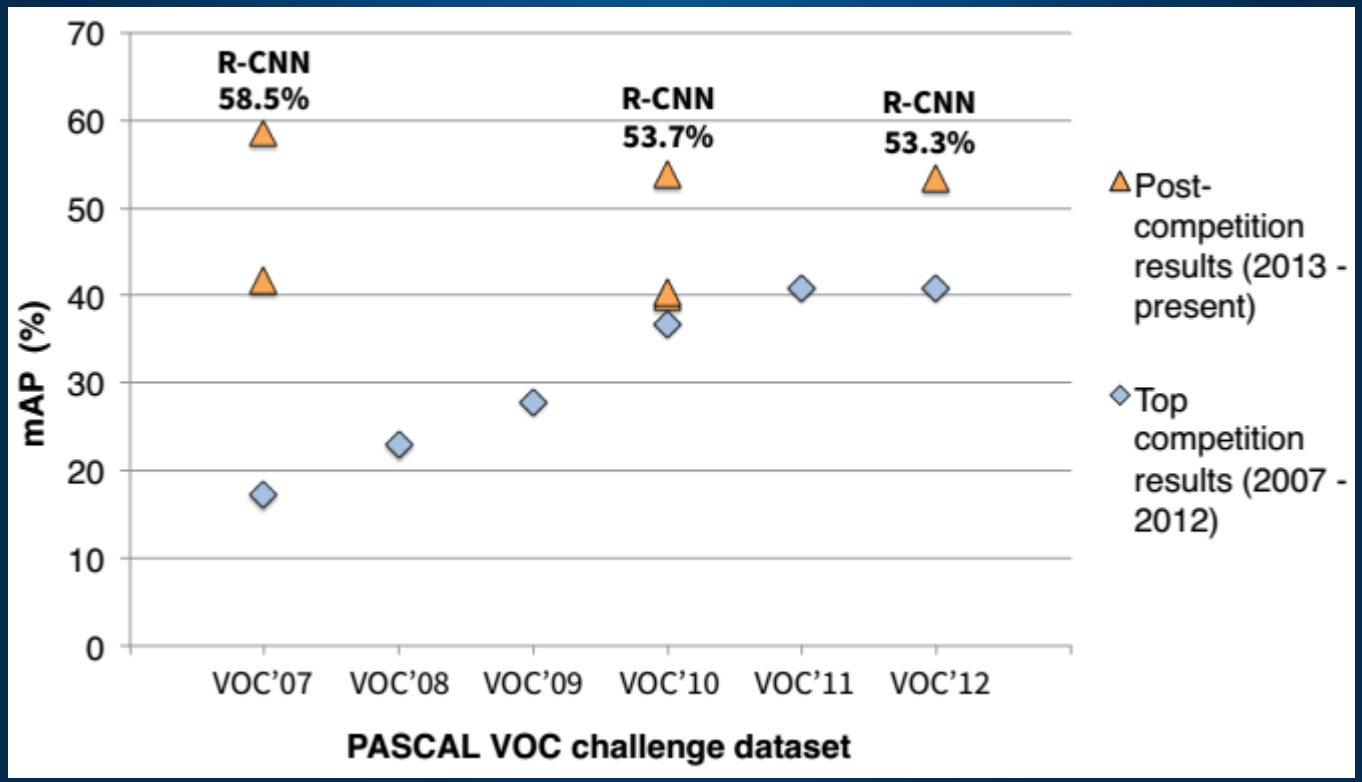
R-CNN: PASCAL VOC performance

2012 SIFT, HOG,...

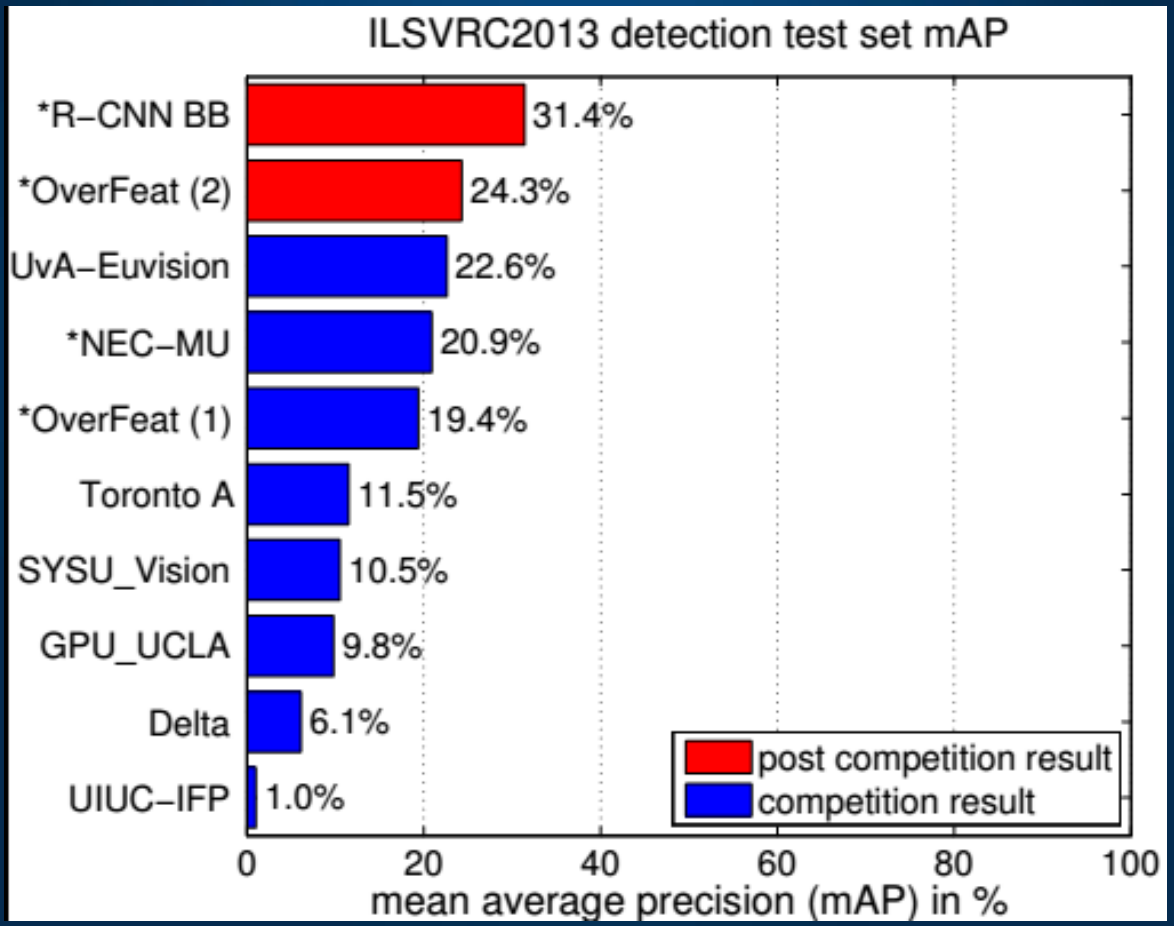


R-CNN: PASCAL VOC performance

2014: Regions with CNN

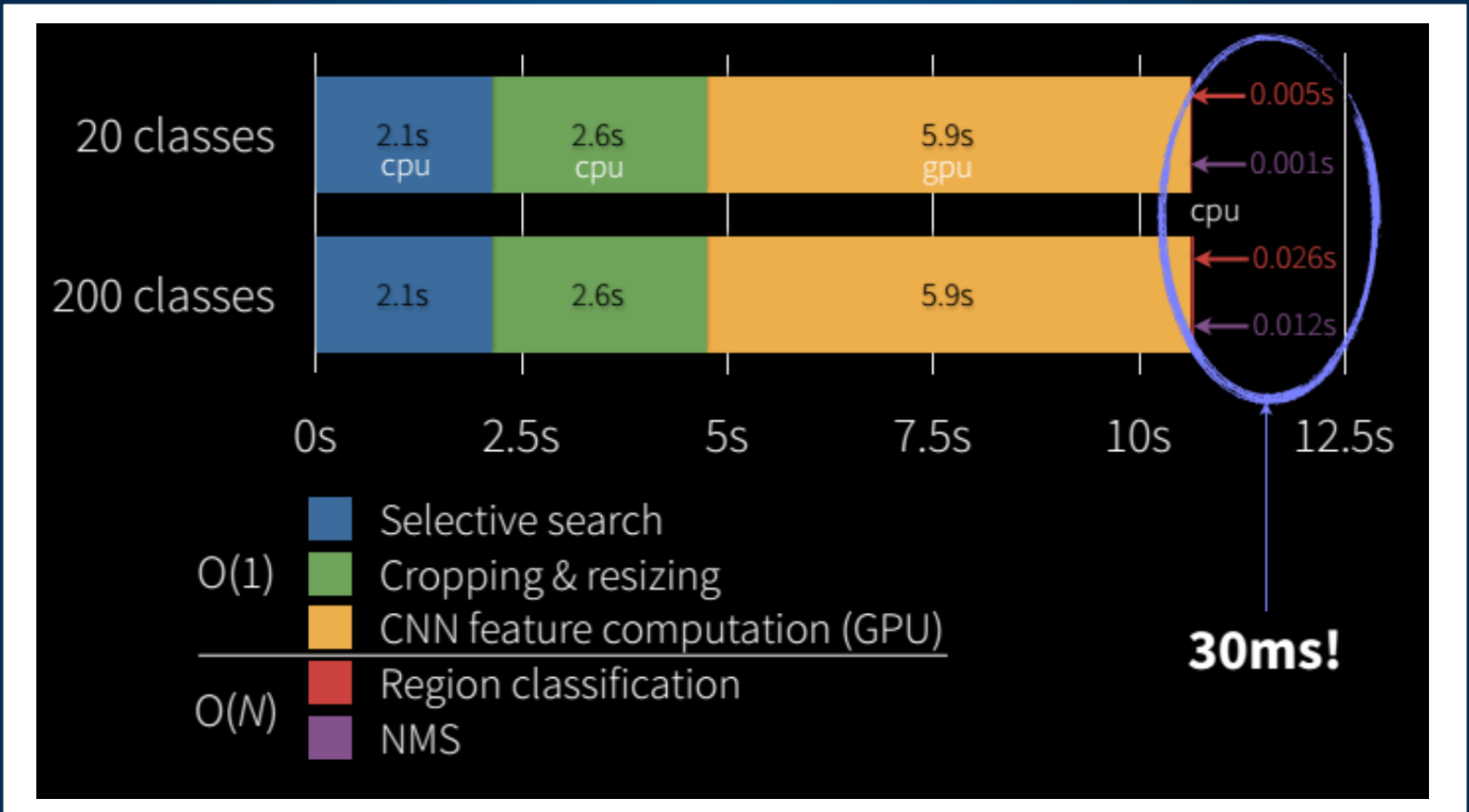


R-CNN: ILSVRC 2013 performance



R-CNN speed and

- R-CNN detection time/frame



R-CNN CODE

https://github.com/rbgirshick/rcnn

Requires Matlab!

See also:

1. Kaiming He et al, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition"

Exercises & Projects

Exercise:

- Implement Overfeat network; train classifier.

Projects:

- Install R-CNN
- Re-implement R-CNN in pure Python/C++ to eliminate Matlab dependency

BACKUP

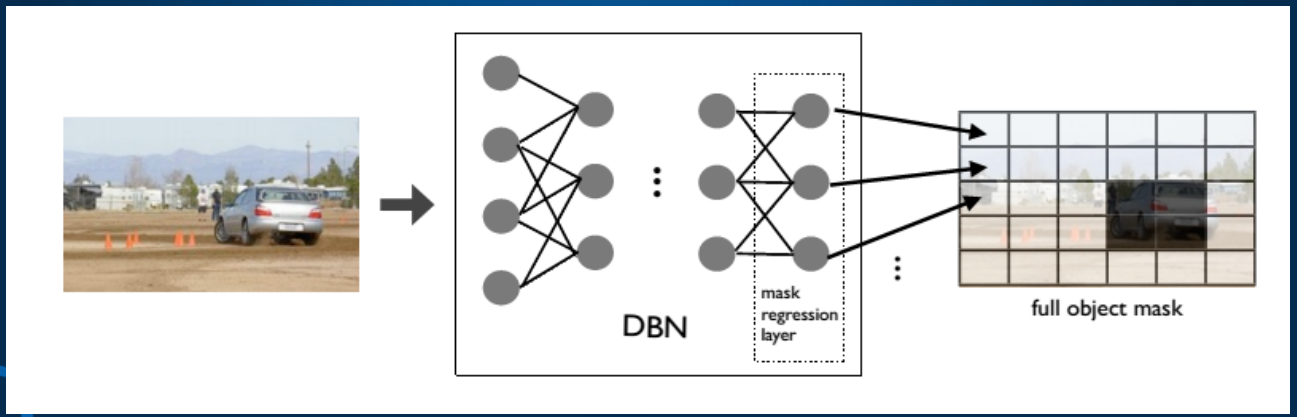
CNN - REGRESSION

CNN regression

Szegedy et all (Google) 2010, "Deep Neural Networks for Object Detection"

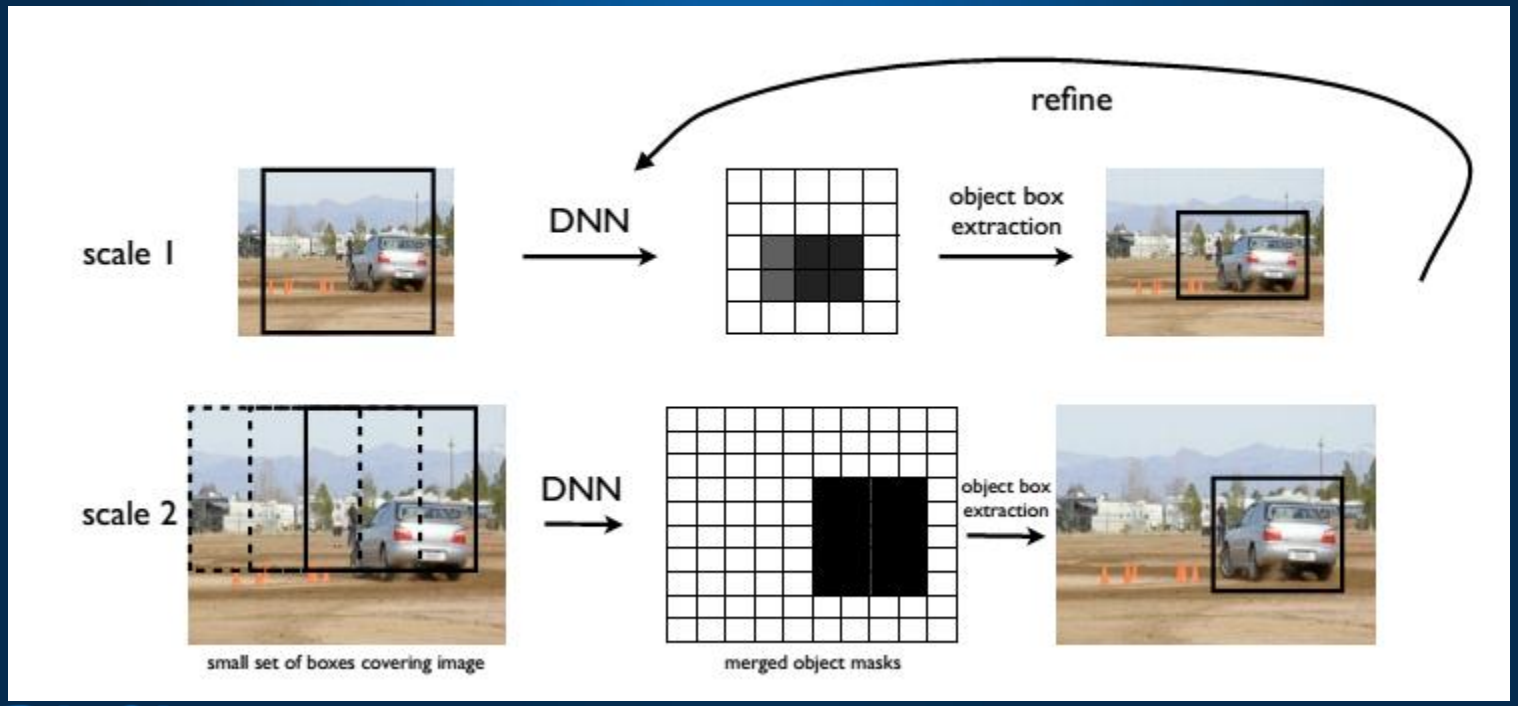
- start with Alexnet,
- replace last soft-max layer with regression layer which generates an binary mask "d x d" : 1 if pixel is inside box, 0- otherwise;
- train net by minimizing L_2 error vs ground truth mask *m:*

$$\min_{\Theta} \sum_{(x,m)\in D} ||(Diag(m) + \lambda I)^{1/2}(DNN(x;\Theta) - m)||_2^2,$$



CNN regression

Multi-scale



CNN regression

Issues:

1. Overlapping masks for multiple touching objects
2. Localization accuracy
3. Recognition of small objects

Issue1:

- To deal with multiple touching objects, we generate not one but several masks, each representing either the full object or part of it.
- we use one network to predict the object box mask and four additional networks to predict four halves of the box: bottom, top, left and right halves